Vehicle Dynamics Blockset™

Reference

MATLAB&SIMULINK®

MathWorks®

How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

Technical support: www.mathworks.com/support/contact_us

Phone: 508-647-7000

The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

Contents

10 | Blocks in Reference Applications

11 | Classes

Steering and Suspension Blocks

Dynamic Steering

Dynamic steering for Ackerman, rack-and-pinion, and parallel steering mechanisms
Library: Vehicle Dynamics Blockset / Steering
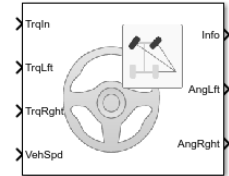
Description

The Dynamic Steering block implements dynamic steering to calculate the wheel angles for Ackerman, rack-and-pinion, and parallel steering mechanisms. The block uses the steering wheel input torque, right wheel torque, and left wheel torque to calculate the wheel angles. The block uses the vehicle coordinate system.

If you select **Power assist**, you can specify a torque assist lookup table that is a function of the vehicle speed and steering wheel input torque. The block uses the steering wheel input torque and torque assist to calculate the steering dynamics.

To specify the steering type, use the **Type** parameter.

Setting	Block Implementation
Ackerman	Ideal Ackerman steering. Wheel angles have a common turning circle center.
Rack and pinion	Ideal rack-and-pinion steering. Gears convert the steering rotation into linear motion.
Parallel	Parallel steering. Wheel angles are equal.

To specify the type of data for the steering mechanism, use the **Parametrized by** parameter.

Setting	Block Implementation
Constant	Steering mechanism uses constant parameter data.
Lookup table	Steering mechanism implements tables for parameter data.

Use the **Location** parameter to specify front or rear steering.

Setting	Implementation
Front	Front steering
Rear	Rear steering

Dynamics

To calculate the steering dynamics, the Dynamic Steering block models the steering wheel, shaft, steering mechanism, hysteresis, and, optionally, power assist.

Calculation	Equations
Steering column and steering shaft dynamics	$J_1 \ddot{\theta}_1 = \tau_{in} - b_2 \dot{\theta}_1 - \tau_{hys}$ $J_2 \ddot{\theta}_2 = \tau_{eq} - b_3 \dot{\theta}_2 + \tau_{hys} - \tau_{fric}$
Hysteresis spring damper	$\delta = \theta_1 - \theta_2$ $\Delta\delta = \delta_{current} - \delta_{previous}$ $\tau_{hys} = \left(b_1 \dot{\delta} - k_1 \delta\right)\left(1 + \exp\left(-\dfrac{\|\Delta\delta\|}{\beta}\right)\right)$ $\beta = \begin{cases} \beta_u & \text{when } \delta > 0 \\ \beta_l & \text{when } \delta \leq 0 \end{cases}$
Optional power assist	$\tau_{ast} = f_{trq}(v, \tau_{in})$ $J_1 \ddot{\theta}_1 = \tau_{in} + \tau_{ast} - b_2 \dot{\theta}_1 - \tau_{hys}$ $J_2 \ddot{\theta}_2 = \tau_{eq} + \tau_{ast} - b_3 \dot{\theta}_2 + \tau_{hys} - \tau_{fric}$

The illustration and equations use these variables.

J_1	Steering wheel inertia
J_2	Steering mechanism inertia
$\theta_1, \dot{\theta}_1, \ddot{\theta}_1$	Steering wheel angle, angular velocity, and angular acceleration, respectively
$\theta_2, \dot{\theta}_2, \ddot{\theta}_2$	Shaft angle, angular velocity, and angular acceleration, respectively
b_1, k_1	Hysteresis spring and viscous damping coefficients, respectively

b_2	Steering wheel viscous damping coefficient
b_3	Steering mechanism damping coefficient
τ_{hys}	Hysteresis spring damping torque
τ_{fric}	Steering mechanism friction torque
τ_{eq}	Wheel equivalent torque
τ_{ast}	Torque assist
β_u , β_l	Upper and lower hysteresis modifiers, respectively
v	Vehicle speed
f_{trq}	Torque assist lookup table

Steering Types

Ackerman

For ideal Ackerman steering, the wheel angles have a common turning circle.

To calculate the steering angles, the block uses these equations.

$$\cot(\delta_L) - \cot(\delta_R) = \frac{TW}{WB}$$

$$\delta_{vir} = \frac{\delta_{in}}{\gamma}$$

$$\delta_L = \tan^{-1}\left(\frac{WB\tan(\delta_{vir})}{WB - 0.5TW\tan(\delta_{vir})}\right)$$

$$\delta_R = \tan^{-1}\left(\frac{WB\tan(\delta_{vir})}{WB + 0.5TW\tan(\delta_{vir})}\right)$$

The illustration and equations use these variables.

δ_{in}	Steering angle
δ_L	Left wheel angle
δ_R	Right wheel angle
δ_{vir}	Virtual wheel angle
TW	Track width
WB	Wheel base
γ	Steering ratio

Rack-and-Pinion

For ideal rack-and-pinion steering, the gears convert the steering rotation into linear motion.

To calculate the steering angles, the block uses these equations.

$$l_1 = \frac{TW - l_{rack}}{2} - \Delta P$$

$$l_2{}^2 = l_1{}^2 + D^2$$

$$\Delta P = r\delta_{in}$$

$$\beta = \frac{\pi}{2} - \tan^{-1}\left[\frac{D}{l_1}\right] - \cos^{-1}\left[\frac{l_{arm}{}^2 + l_2{}^2 - l_{rod}{}^2}{2l_{arm}l_2}\right]$$

The illustration and equations use these variables.

δ_{in}	Steering wheel angle
δ_L	Left wheel angle
δ_R	Right wheel angle
TW	Track width
r	Pinion radius
ΔP	Linear change in rack position
D	Distance between front axis and rack
l_{rack}	Rack casing length
l_{arm}	Steering arm length
l_{rod}	Tie rod length

Parallel

For parallel steering, the wheel angles are equal.

To calculate the steering angles, the block uses this equation.

$$\delta_R = \delta_L = \frac{\delta_{in}}{\gamma}$$

The illustration and equations use these variables.

δ_{in}	Steering wheel angle
δ_L	Left wheel angle
δ_R	Right wheel angle
γ	Steering ratio

Ports

Input

TrqIn — Torque
scalar

Torque, τ_{in}, in N·m.

TrqLft — Left wheel torque
scalar

Left wheel torque, τ_L, in N·m.

TrqRght — Right wheel torque
scalar

Right wheel torque, τ_R, in N·m.

VehSpd — Vehicle speed
scalar

Vehicle speed, v, in m/s.

Dependencies

To create VehSpd, select **Power assist**.

Output

Info — Bus signal
bus

Bus signal contains these block calculations.

Signal	Description	Unit
StrgWhlAng	Steering wheel angle	rad
StrgWhlSpd	Steering wheel angular velocity	rad/s
ShftAng	Shaft angle	rad
ShftSpd	Shaft angular velocity	rad/s
AngLft	Left wheel angle	rad
SpdLft	Left wheel angular velocity	rad/s
AngRght	Right wheel angle	rad
SpdRght	Right wheel angular velocity	rad/s
TrqAst	Torque assist	N·m
PwrAst	Power assist	W
PwrLoss	Power loss	W
InstStrgRatio	Instantaneous steering ratio	NA

AngLft — Left wheel angle
scalar

Left wheel angle, δ_L, in rad.

AngRght — Right wheel angle
scalar

Right wheel angle, δ_R, in rad.

Parameters

Type — Select steering type
Rack and pinion (default) | Ackerman | Parallel

To specify the steering type, use the **Type** parameter.

Setting	Block Implementation
Ackerman	Ideal Ackerman steering. Wheel angles have a common turning circle center.
Rack and pinion	Ideal rack-and-pinion steering. Gears convert the steering rotation into linear motion.
Parallel	Parallel steering. Wheel angles are equal.

Dependencies

This table summarizes the **Type** and **Parametrized by** parameter dependencies.

Type	Parameterized By	Creates Parameters
Ackerman	Constant	**Track width, TrckWdth**
		Wheel base, WhlBase
		Steering range, StrgRng
		Steering ratio, StrgRatio
	Lookup table	**Track width, TrckWdth**
		Wheel base, WhlBase
		Steering range, StrgRng
		Steering angle breakpoints, StrgAngBpts
		Steering ratio table, StrgRatioTbl
Rack and pinion	Constant	**Track width, TrckWdth**
		Steering range, StrgRng
		Steering arm length, StrgArmLngth
		Rack casing length, RckCsLngth
		Tie rod length, TieRodLngth
		Distance between front axis and rack, D
		Pinion radius, PnnRadius

Type	Parameterized By	Creates Parameters
	Lookup table	**Track width, TrckWdth**
		Steering range, StrgRng
		Steering angle breakpoints, StrgAngBpts
		Steering arm length, StrgArmLngth
		Rack casing length, RckCsLngth
		Tie rod length, TieRodLngth
		Distance between front axis and rack, D
		Pinion radius, PnnRadiusTbl
Parallel	Constant	**Steering range, StrgRng**
		Steering ratio, StrgRatio
	Lookup table	**Steering range, StrgRng**
		Steering angle breakpoints, StrgAngBpts
		Steering ratio table, StrgRatioTbl

Parametrized by — Select parameterization
Lookup table (default) | Constant

To specify the type of data for the steering mechanism, use the **Parametrized by** parameter.

Setting	Block Implementation
Constant	Steering mechanism uses constant parameter data.
Lookup table	Steering mechanism implements tables for parameter data.

Dependencies

This table summarizes the **Type** and **Parametrized by** parameter dependencies.

Type	Parameterized By	Creates Parameters
Ackerman	Constant	**Track width, TrckWdth**
		Wheel base, WhlBase
		Steering range, StrgRng
		Steering ratio, StrgRatio

Type	Parameterized By	Creates Parameters
	Lookup table	**Track width, TrckWdth**
		Wheel base, WhlBase
		Steering range, StrgRng
		Steering angle breakpoints, StrgAngBpts
		Steering ratio table, StrgRatioTbl
Rack and pinion	Constant	**Track width, TrckWdth**
		Steering range, StrgRng
		Steering arm length, StrgArmLngth
		Rack casing length, RckCsLngth
		Tie rod length, TieRodLngth
		Distance between front axis and rack, D
		Pinion radius, PnnRadius
	Lookup table	**Track width, TrckWdth**
		Steering range, StrgRng
		Steering angle breakpoints, StrgAngBpts
		Steering arm length, StrgArmLngth
		Rack casing length, RckCsLngth
		Tie rod length, TieRodLngth
		Distance between front axis and rack, D
		Pinion radius, PnnRadiusTbl
Parallel	Constant	**Steering range, StrgRng**
		Steering ratio, StrgRatio
	Lookup table	**Steering range, StrgRng**
		Steering angle breakpoints, StrgAngBpts
		Steering ratio table, StrgRatioTbl

Power assist — Specify power assist
on (default) | off

If you select **Power assist**, you can specify a torque assist lookup table, f_{trq}, that is a function of the vehicle speed, v, and steering wheel input torque, τ_{in}.

$$\tau_{ast} = f_{trq}(v, \tau_{in})$$

The block uses the steering wheel input torque and torque assist to calculate the steering dynamics.

Dependencies
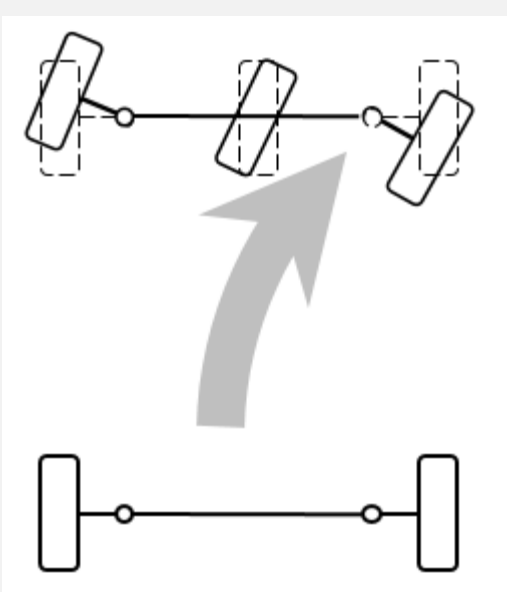
Selecting **Power assist** creates the VehSpd input port and these parameters.
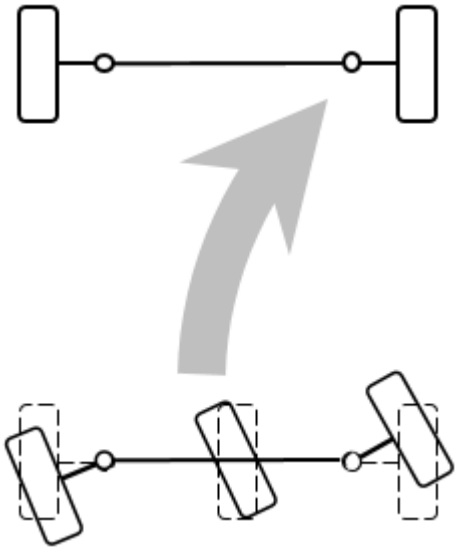
Power Assist	Parameters
on	**Steering wheel torque breakpoints, TrqBpts**
	Vehicle speed breakpoints, VehSpdBpts
	Assisting torque table, TrqTbl
	Assisting torque limit, TrqLmt
	Assisting power limit, PwrLmt
	Assisting torque efficiency, Eta
	Cutoff frequency, omega_c

Location — Select location

Front (default) | **Rear**

Use the **Location** parameter to specify front or rear steering.

Setting	Implementation
Front	Front steering

Setting	Implementation
Rear	Rear steering

General

Track width, TrckWdth — Width
1 | scalar

Track width, *TW*, in m.

Dependencies

To create this parameter, set **Type** to Ackerman or Rack and pinion.

Wheel base, WhlBase — Base
1.524 (default) | scalar

Wheel base, *WB*, in m.

Dependencies

To create this parameter, set **Type** to Ackerman.

Steering range, StrgRng — Range
1.25*pi (default) | scalar

Steering range, in rad. The block limits the wheel angles to remain within the steering range.

Steering ratio, StrgRatio — Ratio
13.5 (default) | scalar

Steering ratio, γ, dimensionless.

Dependencies

To create this parameter:

- Set **Type** to `Ackerman` or `Parallel`.
- Set **Parametrized by** to `Constant`.

Steering angle breakpoints, StrgAngBpts — Breakpoints
`[-6.2832 -5.0265 -3.7699 -2.5133 -1.2566 0 1.2566 2.5133 3.7699 5.0265 6.2832]` (default) | `vector`

Steering angle breakpoints, in rad.

Dependencies

To create this parameter, set **Parametrized by** to `Lookup table`.

Steering ratio table, StrgRatioTbl — Table
`[13.5000 13.3750 13.2500 13.1250 13.0000 13.0000 13.0000 13.1250 13.2500 13.3750 13.5000]` (default) | `vector`

Steering ratio table, γ, dimensionless.

Dependencies

To create this parameter:

- Set **Type** to `Ackerman` or `Parallel`.
- Set **Parametrized by** to `Lookup table`.

Rack-and-Pinion

Steering arm length, StrgArmLngth — Length
`0.1` (default) | `scalar`

Steering arm length, l_{arm}, in m.

Dependencies

To create this parameter, set **Type** to `Rack and pinion`.

Rack casing length, RckCsLngth — Length
`0.5` (default) | `scalar`

Rack casing length, l_{rack}, in m.

Dependencies

To create this parameter, set **Type** to `Rack and pinion`.

Tie rod length, TieRodLngth — Length
`0.248` (default) | `scalar`

Tie rod length, l_{rod}, in m.

Dependencies

To create this parameter, set **Type** to `Rack and pinion`.

Distance between front axis and rack, D — Distance
`0.2` (default) | `scalar`

Distance between front axis and rack, *D*, in m.

Dependencies

To create this parameter, set **Type** to Rack and pinion.

Pinion radius, PnnRadius — Radius
0.0057 (default) | scalar

Pinion radius, *r*, in m.

Dependencies

To create this parameter:

- Set **Type** to Rack and pinion.
- Set **Parametrized by** to Constant.

Pinion radius table, PnnRadiusTbl — Table
[0.0055 0.0055 0.0056 0.0057 0.0057 0.0057 0.0058 0.0057 0.0056 0.0055
0.0055] (default) | vector

Pinion radius table, *r*, in m.

Dependencies

To create this parameter:

- Set **Type** to Rack and pinion.
- Set **Parametrized by** to Lookup table.

Dynamics

Steering wheel inertia, J1 — Inertia
0.1 (default) | scalar

Steering wheel inertia, J_1, in kg*m^2.

Steering mechanism inertia, J2 — Inertia
0.01 (default) | scalar

Steering mechanism inertia, J_2, in kg*m^2.

Upper hysteresis modifier, beta_u — Upper hysteresis modifier
0.1 (default) | scalar

Upper hysteresis modifier, β_u, dimensionless.

Lower hysteresis modifier, beta_l — Lower hysteresis modifier
0.1 (default) | scalar

Lower hysteresis modifier, β_l, dimensionless.

Hysteresis viscous damping, b1 — Damping
0.001 (default) | scalar

Hysteresis damping, b_1, in N·m·s/rad.

Hysteresis stiffness, k1 — Stiffness
30 (default) | scalar

Hysteresis stiffness, k_1, in N·m/rad.

Steering wheel damping, b2 — Damping
1 (default) | scalar

Steering wheel damping, b_2, in N·m·s/rad.

Steering mechanism damping, b3 — Damping
0.001 (default) | scalar

Steering mechanism damping, b_3, in N·m·s/rad.

Initial steering angle, theta_o — Angle
0 (default) | scalar

Initial steering angle, θ_0, in rad.

Initial steering angular velocity, omega_o — Angular velocity
0 (default) | scalar

Initial steering angular velocity, ω_o, in rad/s.

Friction torque, FricTrq — Torque
0 (default) | scalar

Friction torque, τ_{fric}, in N·m.

Power Assist

Steering wheel torque breakpoints, TrqBpts — Breakpoints
[-100 0 100] (default) | 1-by-M vector

Steering wheel torque breakpoints, in N·m.

Dependencies

Selecting **Power assist** creates the VehSpd input port and these parameters.

Power Assist	Parameters
on	**Steering wheel torque breakpoints, TrqBpts**
	Vehicle speed breakpoints, VehSpdBpts
	Assisting torque table, TrqTbl
	Assisting torque limit, TrqLmt
	Assisting power limit, PwrLmt
	Assisting torque efficiency, Eta
	Cutoff frequency, omega_c

Vehicle speed breakpoints, VehSpdBpts — Breakpoints
[0 20] (default) | 1-by-N vector

Vehicle speed breakpoints, in m/s.

Dependencies

Selecting **Power assist** creates the VehSpd input port and these parameters.

Power Assist	Parameters
on	**Steering wheel torque breakpoints, TrqBpts**
	Vehicle speed breakpoints, VehSpdBpts
	Assisting torque table, TrqTbl
	Assisting torque limit, TrqLmt
	Assisting power limit, PwrLmt
	Assisting torque efficiency, Eta
	Cutoff frequency, omega_c

Assisting torque table, TrqTbl — 2D torque table
[0 -100;0 0;0 100] (default) | M-by-N matrix

Assisting torque table, f_{trq}, in N·m.

The torque assist lookup table is a function of the vehicle speed, v, and steering wheel input torque, τ_{in}.

$$\tau_{ast} = f_{trq}(v, \tau_{in})$$

The block uses the steering wheel input torque and torque assist to calculate the steering dynamics.

Dependencies

Selecting **Power assist** creates the VehSpd input port and these parameters.

Power Assist	Parameters
on	**Steering wheel torque breakpoints, TrqBpts**
	Vehicle speed breakpoints, VehSpdBpts
	Assisting torque table, TrqTbl
	Assisting torque limit, TrqLmt
	Assisting power limit, PwrLmt
	Assisting torque efficiency, Eta
	Cutoff frequency, omega_c

`Assisting torque limit, TrqLmt` — Torque limit

100 (default) | `scalar`

Assisting torque limit, in N·m.

Dependencies

Selecting **Power assist** creates the `VehSpd` input port and these parameters.

Power Assist	Parameters
on	**Steering wheel torque breakpoints, TrqBpts**
	Vehicle speed breakpoints, VehSpdBpts
	Assisting torque table, TrqTbl
	Assisting torque limit, TrqLmt
	Assisting power limit, PwrLmt
	Assisting torque efficiency, Eta
	Cutoff frequency, omega_c

`Assisting power limit, PwrLmt` — Power limit

1000 (default) | `scalar`

Assisting power limit, in N·m/s.

Dependencies

Selecting **Power assist** creates the `VehSpd` input port and these parameters.

Power Assist	Parameters
on	**Steering wheel torque breakpoints, TrqBpts**
	Vehicle speed breakpoints, VehSpdBpts
	Assisting torque table, TrqTbl
	Assisting torque limit, TrqLmt
	Assisting power limit, PwrLmt
	Assisting torque efficiency, Eta
	Cutoff frequency, omega_c

`Assisting torque efficiency, Eta` — Efficiency

1 (default) | `scalar`

Assisting torque efficiency, dimensionless.

Dependencies

Selecting **Power assist** creates the `VehSpd` input port and these parameters.

Power Assist	Parameters
on	**Steering wheel torque breakpoints, TrqBpts**
	Vehicle speed breakpoints, VehSpdBpts
	Assisting torque table, TrqTbl
	Assisting torque limit, TrqLmt
	Assisting power limit, PwrLmt
	Assisting torque efficiency, Eta
	Cutoff frequency, omega_c

`Cutoff frequency, omega_c` — **Cutoff frequency**
200 (default) | `scalar`

Cutoff frequency, in rad/s.

Dependencies

Selecting **Power assist** creates the `VehSpd` input port and these parameters.

Power Assist	Parameters
on	**Steering wheel torque breakpoints, TrqBpts**
	Vehicle speed breakpoints, VehSpdBpts
	Assisting torque table, TrqTbl
	Assisting torque limit, TrqLmt
	Assisting power limit, PwrLmt
	Assisting torque efficiency, Eta
	Cutoff frequency, omega_c

References

[1] Crolla, David, David Foster, et al. *Encyclopedia of Automotive Engineering*. Volume 4, Part 5 *(Chassis Systems)* and Part 6 *(Electrical and Electronic Systems)*. Chichester, West Sussex, United Kingdom: John Wiley & Sons Ltd, 2015.

[2] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[3] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Kinematic Steering | Mapped Steering

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2018a

Kinematic Steering

Kinematic steering for Ackerman, rack-and-pinion, and parallel steering mechanisms
Library: Vehicle Dynamics Blockset / Steering
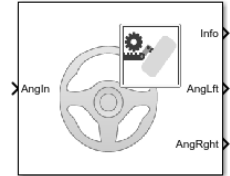
Description

The Kinematic Steering block implements a steering model to determine the left and right wheel angles for Ackerman, rack-and-pinion, and parallel steering mechanisms. The block uses the vehicle coordinate system.

To specify the steering type, use the **Type** parameter.

Setting	Block Implementation
Ackerman	Ideal Ackerman steering. Wheel angles have a common turning circle center.
Rack and pinion	Ideal rack-and-pinion steering. Gears convert the steering rotation into linear motion.
Parallel	Parallel steering. Wheel angles are equal.

To specify the type of data for the steering mechanism, use the **Parametrized by** parameter.

Setting	Block Implementation
Constant	Steering mechanism uses constant parameter data.
Lookup table	Steering mechanism implements tables for parameter data.

Use the **Location** parameter to specify front or rear steering.

Setting	Implementation
Front	Front steering
Rear	Rear steering

Steering Types

Ackerman

For ideal Ackerman steering, the wheel angles have a common turning circle.

WB

TW

Turning Circle

To calculate the steering angles, the block uses these equations.

$$\cot(\delta_L) - \cot(\delta_R) = \frac{TW}{WB}$$

$$\delta_{vir} = \frac{\delta_{in}}{\gamma}$$

$$\delta_L = \tan^{-1}\left(\frac{WB\tan(\delta_{vir})}{WB - 0.5TW\tan(\delta_{vir})}\right)$$

$$\delta_R = \tan^{-1}\left(\frac{WB\tan(\delta_{vir})}{WB + 0.5TW\tan(\delta_{vir})}\right)$$

The illustration and equations use these variables.

δ_{in}	Steering angle
δ_L	Left wheel angle
δ_R	Right wheel angle
δ_{vir}	Virtual wheel angle
TW	Track width
WB	Wheel base
γ	Steering ratio

Rack-and-Pinion

For ideal rack-and-pinion steering, the gears convert the steering rotation into linear motion.

To calculate the steering angles, the block uses these equations.

$$l_1 = \frac{TW - l_{rack}}{2} - \Delta P$$

$$l_2{}^2 = l_1{}^2 + D^2$$

$$\Delta P = r\delta_{in}$$

$$\beta = \frac{\pi}{2} - \tan^{-1}\left[\frac{D}{l_1}\right] - \cos^{-1}\left[\frac{l_{arm}{}^2 + l_2{}^2 - l_{rod}{}^2}{2l_{arm}l_2}\right]$$

1-25

The illustration and equations use these variables.

δ_{in}	Steering wheel angle
δ_L	Left wheel angle
δ_R	Right wheel angle
TW	Track width
r	Pinion radius
ΔP	Linear change in rack position
D	Distance between front axis and rack
l_{rack}	Rack casing length
l_{arm}	Steering arm length
l_{rod}	Tie rod length

Parallel

For parallel steering, the wheel angles are equal.

To calculate the steering angles, the block uses this equation.

$$\delta_R = \delta_L = \frac{\delta_{in}}{\gamma}$$

The illustration and equations use these variables.

δ_{in}	Steering wheel angle
δ_L	Left wheel angle
δ_R	Right wheel angle
γ	Steering ratio

Ports

Input

AngIn — Steering angle
scalar

Steering angle, δ_{in}, in rad.

Output

Info — Bus signal
bus

Bus signal contains this block calculation.

Signal	Description	Variable	Unit
InstStrgRatio	Instantaneous steering ratio	γ	NA

AngLft — Left wheel angle
scalar

Left wheel angle, δ_L, in rad.

AngRght — Right wheel angle
scalar

Right wheel angle, δ_R, in rad.

Parameters

Type — Select steering type
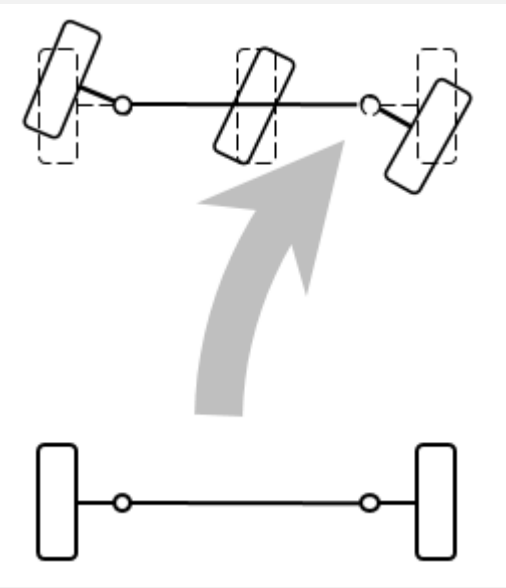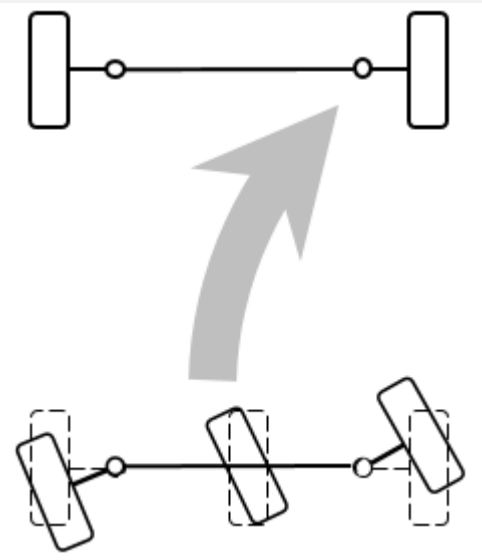Ackerman (default) | Rack and pinion | Parallel

To specify the steering type, use the **Type** parameter.

Setting	Block Implementation
Ackerman	Ideal Ackerman steering. Wheel angles have a common turning circle center.
Rack and pinion	Ideal rack-and-pinion steering. Gears convert the steering rotation into linear motion.
Parallel	Parallel steering. Wheel angles are equal.

Dependencies

This table summarizes the **Type** and **Parametrized by** parameter dependencies.

Type	Parameterized By	Creates Parameters
Ackerman	Constant	**Track width, TrckWdth**
		Wheel base, WhlBase
		Deadband, Db
		Steering range, StrgRng
		Steering ratio, StrgRatio
	Lookup table	**Track width, TrckWdth**
		Wheel base, WhlBase
		Deadband, Db
		Steering range, StrgRng
		Steering angle breakpoints, StrgAngBpts
		Steering ratio table, StrgRatioTbl
Rack and pinion	Constant	**Track width, TrckWdth**
		Deadband, Db
		Steering range, StrgRng
		Steering arm length, StrgArmLngth
		Rack casing length, RckCsLngth
		Tie rod length, TieRodLngth
		Distance between front axis and rack, D
		Pinion radius, PnnRadius

Type	Parameterized By	Creates Parameters
	Lookup table	**Track width, TrckWdth**
		Deadband, Db
		Steering range, StrgRng
		Steering angle breakpoints, StrgAngBpts
		Steering arm length, StrgArmLngth
		Rack casing length, RckCsLngth
		Tie rod length, TieRodLngth
		Distance between front axis and rack, D
		Pinion radius, PnnRadiusTbl
Parallel	Constant	**Deadband, Db**
		Steering range, StrgRng
		Steering ratio, StrgRatio
	Lookup table	**Deadband, Db**
		Steering range, StrgRng
		Steering angle breakpoints, StrgAngBpts
		Steering ratio table, StrgRatioTbl

`Parametrized by` — Select parameterization
`Constant` (default) | `Lookup table`

To specify the type of data for the steering mechanism, use the **Parametrized by** parameter.

Setting	Block Implementation
`Constant`	Steering mechanism uses constant parameter data.
`Lookup table`	Steering mechanism implements tables for parameter data.

Dependencies

This table summarizes the **Type** and **Parametrized by** parameter dependencies.

Type	Parameterized By	Creates Parameters
Ackerman	Constant	**Track width, TrckWdth** **Wheel base, WhlBase** **Deadband, Db** **Steering range, StrgRng** **Steering ratio, StrgRatio**
	Lookup table	**Track width, TrckWdth** **Wheel base, WhlBase** **Deadband, Db** **Steering range, StrgRng** **Steering angle breakpoints, StrgAngBpts** **Steering ratio table, StrgRatioTbl**
Rack and pinion	Constant	**Track width, TrckWdth** **Deadband, Db** **Steering range, StrgRng** **Steering arm length, StrgArmLngth** **Rack casing length, RckCsLngth** **Tie rod length, TieRodLngth** **Distance between front axis and rack, D** **Pinion radius, PnnRadius**
	Lookup table	**Track width, TrckWdth** **Deadband, Db** **Steering range, StrgRng** **Steering angle breakpoints, StrgAngBpts** **Steering arm length, StrgArmLngth** **Rack casing length, RckCsLngth** **Tie rod length, TieRodLngth** **Distance between front axis and rack, D** **Pinion radius, PnnRadiusTbl**

Type	Parameterized By	Creates Parameters
Parallel	Constant	**Deadband, Db**
		Steering range, StrgRng
		Steering ratio, StrgRatio
	Lookup table	**Deadband, Db**
		Steering range, StrgRng
		Steering angle breakpoints, StrgAngBpts
		Steering ratio table, StrgRatioTbl

Location — Select location
Front (default) | **Rear**

Use the **Location** parameter to specify front or rear steering.

Setting	Implementation
Front	Front steering

Setting	Implementation
Rear	Rear steering

Normalization factor, NrmFctr — Adjust the steering angle
scalar

Factor, Nrm_{Fctr}, that the block uses to adjust the steering ratio, γ or pinion radius, r. The block can only normalize if you have **Parametrized by** set to Constant.

To adjust the steering ratio or pinion radius, click **Normalize**.

Steering Type	Normalization
Ackerman Parallel	Block updates the **Steering ratio, StrgRatio** parameter to the normalized value, γ_{nrm}, specified by this equation. $$\gamma_{nrm} = \frac{1}{Nrm_{Fctr}}$$
Rack and pinion	Block updates the **Pinion radius, PnnRadius** parameter to using the normalization factor, Nrm_{Fctr}.

General

Track width, TrckWdth — Width
1 (default) | scalar

Track width, *TW*, in m.

Dependencies

To create this parameter, set **Type** to Ackerman or Rack and pinion.

Wheel base, WhlBase — Base
1.524 (default) | scalar

Wheel base, *WB*, in m.

Dependencies

To create this parameter, set **Type** to `Ackerman`.

Deadband, Db — Deadband
0 (default) | scalar

Deadband steering angle before pinion engages the gear, in rad.

Steering range, StrgRng — Range
1.25*pi (default) | scalar

Steering range, in rad. The block limits the wheel angles to remain within the steering range.

Steering ratio, StrgRatio — Ratio
100 (default) | scalar

Steering ratio, γ, dimensionless.

Dependencies

To create this parameter:

- Set **Type** to `Ackerman` or `Parallel`.
- Set **Parametrized by** to `Constant`.

Steering angle breakpoints, StrgAngBpts — Breakpoints
[-6.2832 -5.0265 -3.7699 -2.5133 -1.2566 0 1.2566 2.5133 3.7699 5.0265 6.2832] (default) | vector

Steering angle breakpoints, in rad.

Dependencies

To create this parameter, set **Parametrized by** to `Lookup table`.

Steering ratio table, StrgRatioTbl — Table
[13.5000 13.3750 13.2500 13.1250 13.0000 13.0000 13.0000 13.1250 13.2500 13.3750 13.5000] (default) | vector

Steering ratio table, γ, dimensionless.

Dependencies

To create this parameter:

- Set **Type** to `Ackerman` or `Parallel`.
- Set **Parametrized by** to `Lookup table`.

Rack-and-Pinion

Steering arm length, StrgArmLngth — Length
0.1 (default) | scalar

Steering arm length, l_{arm}, in m.

Dependencies

To create this parameter, set **Type** to Rack and pinion.

Rack casing length, RckCsLngth — Length
0.5 (default) | scalar

Rack casing length, l_{rack}, in m.

Dependencies

To create this parameter, set **Type** to Rack and pinion.

Tie rod length, TieRodLngth — Length
0.248 (default) | scalar

Tie rod length, l_{rod}, in m.

Dependencies

To create this parameter, set **Type** to Rack and pinion.

Distance between front axis and rack, D — Distance
0.2 (default) | scalar

Distance between front axis and rack, D, in m.

Dependencies

To create this parameter, set **Type** to Rack and pinion.

Pinion radius, PnnRadius — Radius
0.0057 (default) | scalar

Pinion radius, r, in m.

Dependencies

To create this parameter:

- Set **Type** to Rack and pinion.
- Set **Parametrized by** to Constant.

Pinion radius table, PnnRadiusTbl — Table
[0.0055 0.0055 0.0056 0.0057 0.0057 0.0057 0.0058 0.0057 0.0056 0.0055 0.0055] (default) | vector

Pinion radius table, r, in m.

Dependencies

To create this parameter:

- Set **Type** to Rack and pinion.
- Set **Parametrized by** to Lookup table.

References

[1] Crolla, David, David Foster, et al. *Encyclopedia of Automotive Engineering*. Volume 4, Part 5 *(Chassis Systems)* and Part 6 *(Electrical and Electronic Systems)*. Chichester, West Sussex, United Kingdom: John Wiley & Sons Ltd, 2015.

[2] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[3] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Dynamic Steering | Mapped Steering

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2018a

Mapped Steering

Mapped steering with speed-dependent option
Library: Vehicle Dynamics Blockset / Steering
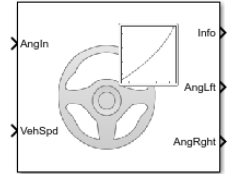
Description

The Mapped Steering block implements lookup tables to calculate the right and left wheel angles. Use the **Speed dependent** parameter to implement a speed-dependent table for the angle calculations. The block uses the vehicle coordinate system.

Speed Dependent	Implementation	Calculations
on (default)	Block uses three tables: • f_s — Function of vehicle speed • f_L — Function of superimposed steering wheel angle • f_R — Function of superimposed steering wheel angle	$\delta_{SpdF} = f_s(v)$ $\delta_{SuprImp} = \delta_{SpdF} \cdot \delta_{in}$ $\delta_L = f_L(\delta_{SuprImp})$ $\delta_R = f_R(\delta_{SuprImp})$
off	Block uses two tables: • f_L — Function of steering wheel angle • f_R — Function of steering wheel angle	$\delta_L = f_L(\delta_{in})$ $\delta_R = f_R(\delta_{in})$

The equations use these variables.

δ_{in}	Steering wheel angle
δ_{SpdF}	Steering wheel angle speed factor
$\delta_{SuprImp}$	Superimposed steering wheel angle
δ_L, δ_R	Left and right wheel angles, respectively

Ports

Input

AngIn — Steering angle
scalar

Steering angle, δ_{in}, in rad.

VehSpd — Vehicle speed
scalar

Vehicle speed, Veh_{spd}, in m/s.

Dependencies

To enable this port, select **Speed dependent**.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal	Description	Variable	Unit
AngLft	Left wheel angle	δ_L	rad
AngRght	Left wheel angle	δ_R	rad

AngLft — Left wheel angle
scalar

Left wheel angle, δ_L, in rad.

AngRght — Right wheel angle
scalar

Right wheel angle, δ_R, in rad.

Parameters

Steering angle breakpoints, StrgAngBpts — Steering angle breakpoints
[-1.5*pi 1.5*pi] (default) | vector

Steering angle breakpoints, in rad.

Left wheel angle table, WhlLftTbl — Left wheel angle table
[-1.5*pi 1.5*pi]/13.5 (default) | vector

Left wheel angle table, δ_L, in rad.

Right wheel angle table, WhlRghtTbl — Right wheel angle table
[-1.5*pi 1.5*pi]/13.5 (default) | vector

Right wheel angle table, δ_R, in rad.

Vehicle speed breakpoints, VehSpdBpts — Vehicle speed breakpoints
[-1 1] (default) | vector

Vehicle speed breakpoints, in m/s.

Dependencies

To create this parameter, select **Speed dependent**.

Superimposed speed factor table, SpdFctTbl — Speed factor
[1 1] (default) | vector

Superimposed speed factor table, f_s, dimensionless. The table is a factor of vehicle speed, v.

Dependencies

To create this parameter, select **Speed dependent**.

References

[1] Crolla, David, David Foster, et al. *Encyclopedia of Automotive Engineering*. Volume 4, Part 5 *(Chassis Systems)* and Part 6 *(Electrical and Electronic Systems)*. Chichester, West Sussex, United Kingdom: John Wiley & Sons Ltd, 2015.

[2] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[3] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Dynamic Steering | Kinematic Steering

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2018a

Independent Suspension - Double Wishbone

Double wishbone independent suspension
Library: Vehicle Dynamics Blockset / Suspension
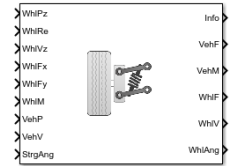
Description

The Independent Suspension - Double Wishbone block implements an independent double wishbone suspension for multiple axles with multiple tracks per axle.

The block models the suspension compliance, damping, and geometric effects as functions of the relative positions and velocities of the vehicle and wheel carrier with axle-specific compliance and damping parameters. Using the suspension compliance and damping, the block calculates the suspension force on the vehicle and wheel. The block uses the Z-down coordinate system (defined in SAE J670).

For Each	You Can Specify
Axle	• Multiple tracks
	• An anti-sway bar for axles with two tracks
	• Suspension parameters
Track	• Steering angles

The block contains energy-storing spring elements and energy-dissipating damper elements. It does not contain energy-storing mass elements. The block assumes that the vehicle (sprung) and wheel (unsprung) blocks connected to the block store the mass-related suspension energy.

This table summarizes the block parameter settings for a vehicle with:

• Two axles
• Two tracks per axle
• Steering angle input for both tracks on the front axle
• An anti-sway bar on the front axle

Parameter	Setting
Number of axles, NumAxl	2
Number of tracks by axle, NumTracksByAxl	[2 2]
Steered axle enable by axle, StrgEnByAxl	[1 0]
Anti-sway axle enable by axle, AntiSwayEnByAxl	[1 0]

Suspension Compliance and Damping

The block uses a linear spring and damper to model the vertical dynamic effects of the suspension system. Using the relative positions and velocities of the vehicle and wheel carrier, the block

calculates the vertical suspension forces on the wheel and vehicle. The block uses a linear equation that relates the vertical damping and compliance to the suspension height, suspension height rate of change, and absolute value of the steering angles.

The block implements this equation.

$$F_{wz_{a,t}} = F_{z0_a} + k_{z_a}(z_{v_{a,t}} - z_{w_{a,t}} + m_{hsteer_a}|\delta_{steer_{a,t}}|) + c(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}) + F_{zhstop_{a,t}} + F_{zaswy_{a,t}}$$

The damping coefficient, c, depends on the **Enable active damping** parameter setting.

Enable active damping Setting	Damping
off	Constant, $c = c_{z_a}$
on	Lookup table that is a function of active damper duty cycle and actuator velocity $$c = f\left(duty, (\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}})\right)$$

The block assumes that the suspension elements have no mass. Therefore, the suspension forces and moments applied to the vehicle are equal to the suspension forces and moments applied to the wheel.

$$F_{vx_{a,t}} = F_{wx_{a,t}}$$

$$F_{vy_{a,t}} = F_{wy_{a,t}}$$

$$F_{vz_{a,t}} = -F_{wz_{a,t}}$$

$$M_{vx_{a,t}} = M_{wx_{a,t}} + F_{wy_{a,t}}(Re_{wy_{a,t}} + H_{a,t})$$

$$M_{vy_{a,t}} = M_{wy_{a,t}} + F_{wx_{a,t}}\left(Re_{wx_{a,t}} + H_{a,t}\right)$$

$$M_{vz_{a,t}} = M_{wz_{a,t}}$$

The block sets the wheel positions and velocities equal to the vehicle lateral and longitudinal positions and velocities.

$$x_{w_{a,t}} = x_{v_{a,t}}$$

$$y_{w_{a,t}} = y_{v_{a,t}}$$

$$\dot{x}_{w_{a,t}} = \dot{x}_{v_{a,t}}$$

$$\dot{y}_{w_{a,t}} = \dot{y}_{v_{a,t}}$$

The equations use these variables.

$F_{wz_{a,t}}$, $M_{wz_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed z-axis
$F_{wx_{a,t}}$, $M_{wx_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed x-axis
$F_{wy_{a,t}}$, $M_{wy_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed y-axis

$F_{vz_{a,t}}$, $M_{vz_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed z-axis
$F_{vx_{a,t}}$, $M_{vx_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed x-axis
$F_{vy_{a,t}}$, $M_{vy_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed y-axis
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle a
k_{z_a}	Vertical spring constant applied to tracks on axle a
m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
c_{z_a}	Vertical damping constant applied to tracks on axle a
$Re_{w_{a,t}}$	Effective wheel radius for axle a, track t
$F_{zhstop_{a,t}}$	Vertical hardstop force at axle a, track t, along the vehicle-fixed z-axis
$F_{zaswy_{a,t}}$	Vertical anti-sway force at axle a, track t, along the vehicle-fixed z-axis
$z_{v_{a,t}}$, $\dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$, $\dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{v_{a,t}}$, $\dot{x}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{w_{a,t}}$, $\dot{x}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$y_{v_{a,t}}$, $\dot{y}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$y_{w_{a,t}}$, $\dot{y}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$H_{a,t}$	Suspension height at axle a, track t
$Re_{w_{a,t}}$	Effective wheel radius at axle a, track t

Hardstop Forces

The hardstop feedback force, $F_{zhstop_{a,t}}$, that the block applies depends on whether the suspension is compressing or extending. The block applies the force:

- In compression, when the suspension is compressed more than the maximum distance specified by the **Suspension maximum height, Hmax** parameter.
- In extension, when the suspension extension is greater than maximum extension specified by the **Suspension maximum height, Hmax** parameter.

To calculate the force, the block uses a stiffness based on a hyperbolic tangent and exponential scaling.

Anti-Sway Bar

Optionally, the block implements an anti-sway bar force, $F_{zaswy_{a,t}}$, for axles that have two tracks. This figure shows how the anti-sway bar transmits torque between two independent suspension tracks on a shared axle. Each independent suspension applies a torque to the anti-sway bar via a radius arm that extends from the anti-sway bar back to the independent suspension connection point.

To calculate the sway bar force, the block implements these equations.

Calculation	Equation
Anti-sway bar angular deflection for a given axle and track, $\Delta\Theta_{a,t}$	$\theta_{0a} = \tan^{-1}\left(\frac{z_0}{r}\right)$ $\Delta\theta_{a,t} = \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,t}} + z_{v_{a,t}}}{r}\right)$
Anti-sway bar twist angle, Θ_a	$\theta_a = -\tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,1}} + z_{v_{a,1}}}{r}\right)$ $- \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,2}} + z_{v_{a,2}}}{r}\right)$
Anti-sway bar torque, τ_a	$\tau_a = k_a\theta_a$
Anti-sway bar forces applied to the wheel on axle a, track t along wheel-fixed z-axis	$F_{zaswy_{a,1}} = \left(\frac{\tau_a}{r}\right)\cos\left(\theta_{0a} - \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,1}} + z_{v_{a,1}}}{r}\right)\right)$ $F_{zaswy_{a,2}} = \left(\frac{\tau_a}{r}\right)\cos\left(\theta_{0a} - \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,2}} + z_{v_{a,2}}}{r}\right)\right)$

The equations and figure use these variables.

τ_a	Anti-sway bar torque
θ	Anti-sway bar twist angle
θ_{0a}	Initial anti-sway bar twist angle
$\Delta\Theta_{a,t}$	Anti-sway bar angular deflection at axle a, track t
r	Anti-sway bar arm radius
z_0	Vertical distance from anti-sway bar connection point to anti-sway bar centerline
$F_{zsway_{a,t}}$	Anti-sway bar force applied to the wheel on axle a, track t along wheel-fixed z-axis
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Wheel displacement at axle a, track t, along the vehicle-fixed z-axis

Camber, Caster, and Toe Angles

To calculate the camber, caster, and toe angles, block uses linear functions of the suspension height and steering angle.

$$\xi_{a,t} = \xi_{0a} + m_{hcamber_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{cambersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\eta_{a,t} = \eta_{0a} + m_{hcaster_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{castersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\zeta_{a,t} = \zeta_{0a} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equations use these variables.

$\xi_{a,t}$	Camber angle of wheel on axle a, track t
$\eta_{a,t}$	Caster angle of wheel on axle a, track t
$\zeta_{a,t}$	Toe angle of wheel on axle a, track t
ξ_{0a}, η_{0a}, ζ_{0a}	Nominal suspension axle a camber, caster, and toe angles, respectively, at zero steering angle
$m_{hcamber_a}$, $m_{hcaster_a}$, m_{htoe_a}	Camber, caster, and toe angles, respectively, versus suspension height slope for axle a
$m_{cambersteer_a}$, $m_{castersteer_a}$, $m_{toesteer_a}$	Camber, caster, and toe angles, respectively, versus steering angle slope for axle a
m_{hsteer_a}	Steering angle versus vertical force slope for axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Steering Angles

Optionally, you can input steering angles for the tracks. To calculate the steering angles for the wheels, the block offsets the input steering angles with a linear function of the suspension height.

$$\delta_{whlsteer_{a,t}} = \delta_{steer_{a,t}} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equation uses these variables.

$m_{toesteer_a}$	Axle a toe angle versus steering angle slope
m_{hsteer_a}	Axle a steering angle versus vertical force slope
m_{htoe_a}	Axle a toe angle versus suspension height slope
$\delta_{whlsteer_{a,t}}$	Wheel steering angle for axle a, track t
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Power and Energy

The block calculates these suspension characteristics for each axle, a, track, t.

Calculation	Equation
Dissipated power, $P_{susp_{a,t}}$	$P_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Absorbed energy, $E_{susp_{a,t}}$	$E_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Suspension height, $H_{a,t}$	$H_{a,t} = -\left(z_{v_{a,t}} - z_{w_{a,t}} + \dfrac{F_{z0_a}}{k_{z_a}} + m_{hsteer_a}\left\|\delta_{steer_{a,t}}\right\|\right)$
Distance from wheel carrier center to tire/road interface	$z_{wtr_{a,t}} = Re_{w_{a,t}} + H_{a,t}$

The equations use these variables.

m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle `a`
$\delta_{steer_{a,t}}$	Steering angle input for axle `a`, track `t`
$Re_{w_{a,t}}$	Axle `a`, track `t` effective wheel radius from wheel carrier center to tire/road interface
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle `a`
$z_{wtr_{a,t}}$	Distance from wheel carrier center to tire/road interface, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle `a`, track `t`, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle `a`, track `t`, along the vehicle-fixed z-axis

Ports

Input

WhlPz — Track z-axis displacement
`array`

Track displacement, z_w, along wheel-fixed z-axis, in m. Array dimensions are `1` by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlPz`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlPz} = z_w = \begin{bmatrix} z_{w_{1,1}} & z_{w_{1,2}} & z_{w_{2,1}} & z_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlPz(1,1)	1	1
WhlPz(1,2)	1	2
WhlPz(1,3)	2	1
WhlPz(1,4)	2	2

WhlRe — Wheel effective radius
`array`

Effective wheel radius, Re_w, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlRe:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlRe} = Re_w = \begin{bmatrix} Re_{w_{1,1}} & Re_{w_{1,2}} & Re_{w_{2,1}} & Re_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlRe(1,1)	1	1
WhlRe(1,2)	1	2
WhlRe(1,3)	2	1
WhlRe(1,4)	2	2

WhlVz — Track z-axis velocity
array

Track velocity, $\dot{z}_w$, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlVz:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlVz} = \dot{z}_w = \begin{bmatrix} \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlVz(1,1)	1	1
WhlVz(1,2)	1	2
WhlVz(1,3)	2	1
WhlVz(1,4)	2	2

WhlFx — Longitudinal wheel force on vehicle
array

Longitudinal wheel force applied to vehicle, F_{wx}, along the vehicle-fixed x-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFx:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlFx} = F_{wx} = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlFx(1,1)	1	1
WhlFx(1,2)	1	2
WhlFx(1,3)	2	1
WhlFx(1,4)	2	2

WhlFy — Lateral wheel force on vehicle
array

Lateral wheel force applied to vehicle, F_{wy}, along the vehicle-fixed y-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFy:

- Signal array dimensions are [1x4].

- Array dimensions are axle by track.

$$\text{WhlFy} = F_{wy} = \begin{bmatrix} F_{wy1,1} & F_{wy1,2} & F_{wy2,1} & F_{wy2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFy(1,1)	1	1
WhlFy(1,2)	1	2
WhlFy(1.3)	2	1
WhlFy(1,4)	2	2

WhlM — Suspension moment on wheel
array

Longitudinal, lateral, and vertical suspension moments at axle a, track t, applied to the wheel at the axle wheel carrier reference coordinate, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlM(1,...) — Suspension moment applied to the wheel about the vehicle-fixed x-axis (longitudinal)

- WhlM(2,...) — Suspension moment applied to the wheel about the vehicle-fixed y-axis (lateral)

- WhlM(3,...) — Suspension moment applied to the wheel about the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlM:

- Signal dimensions are [3x4].

- Signal contains suspension moments applied to four wheels according to their axle and track locations.

$$\text{WhlM} = M_w = \begin{bmatrix} M_{wx1,1} & M_{wx1,2} & M_{wx2,1} & M_{wx2,2} \\ M_{wy1,1} & M_{wy1,2} & M_{wy2,1} & M_{wy2,2} \\ M_{wz1,1} & M_{wz1,2} & M_{wz2,1} & M_{wz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
WhlM(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlM(1,2)	1	2	
WhlM(1,3)	2	1	
WhlM(1,4)	2	2	
WhlM(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlM(2,2)	1	2	
WhlM(2,3)	2	1	
WhlM(2,4)	2	2	
WhlM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlM(3,2)	1	2	
WhlM(3,3)	2	1	
WhlM(3,4)	2	2	

VehP — Vehicle displacement
array

Vehicle displacement from axle `a`, track `t` along vehicle-fixed coordinate system, in m. Array dimensions are 3 by the total number of tracks on the vehicle.

- VehP(1,...) — Vehicle displacement from track, x_v, along the vehicle-fixed x-axis
- VehP(2,...) — Vehicle displacement from track, y_v, along the vehicle-fixed y-axis
- VehP(3,...) — Vehicle displacement from track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehP`:

- Signal dimensions are [3x4].

- Signal contains four track displacements according to their axle and track locations.

$$\text{VehP} = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} x_{v1,1} & x_{v1,2} & x_{v2,1} & x_{v2,2} \\ y_{v1,1} & y_{v1,2} & y_{v2,1} & y_{v2,2} \\ z_{v1,1} & z_{v1,2} & z_{v2,1} & z_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehP(1,1)	1	1	Vehicle-fixed x-axis
VehP(1,2)	1	2	
VehP(1,3)	2	1	
VehP(1,4)	2	2	
VehP(2,1)	1	1	Vehicle-fixed y-axis
VehP(2,2)	1	2	
VehP(2,3)	2	1	
VehP(2,4)	2	2	

Array Element	Axle	Track	Axis
VehP(3,1)	1	1	Vehicle-fixed z-axis
VehP(3,2)	1	2	
VehP(3,3)	2	1	
VehP(3,4)	2	2	

VehV — Vehicle velocity
array

Vehicle velocity at axle `a`, track `t` along vehicle-fixed coordinate system, in m. Input array dimensions are 3 by `a*t`.

- `VehV(1,...)` — Vehicle velocity at track, x_v, along the vehicle-fixed x-axis
- `VehV(2,...)` — Vehicle velocity at track, y_v, along the vehicle-fixed y-axis
- `VehV(3,...)` — Vehicle velocity at track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehV`:

- Signal dimensions are `[3x4]`.
- Signal contains 4 track velocities according to their axle and track locations.

$$\text{VehV} = \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{z}_v \end{bmatrix} = \begin{bmatrix} \dot{x}_{v1,1} & \dot{x}_{v1,2} & \dot{x}_{v2,1} & \dot{x}_{v2,2} \\ \dot{y}_{v1,1} & \dot{y}_{v1,2} & \dot{y}_{v2,1} & \dot{y}_{v2,2} \\ \dot{z}_{v1,1} & \dot{z}_{v1,2} & \dot{z}_{v2,1} & \dot{z}_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehV(1,1)	1	1	Vehicle-fixed x-axis
VehV(1,2)	1	2	
VehV(1,3)	2	1	
VehV(1,4)	2	2	
VehV(2,1)	1	1	Vehicle-fixed y-axis
VehV(2,2)	1	2	
VehV(2,3)	2	1	
VehV(2,4)	2	2	
VehV(3,1)	1	1	Vehicle-fixed z-axis
VehV(3,2)	1	2	
VehV(3,3)	2	1	
VehV(3,4)	2	2	

StrgAng — Steering angle, optional
array

Optional steering angle for each wheel, δ. Input array dimensions are 1 by the number of steered tracks.

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.

- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer1,1} & \delta_{steer1,2} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Output

Info — Bus signal
bus

Bus signal containing block values. The signals are arrays that depend on the track location.

For example, here are the indices for a two-axle, two-track vehicle. The total number of tracks is four.

- 1D array signal (1-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

- 3D array signal (3-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

Array Element	Axle	Track
(2,1)	1	1
(2,2)	1	2
(2,3)	2	1
(2,4)	2	2
(3,1)	1	1
(3,2)	1	2
(3,3)	2	1
(3,4)	2	2

Signal	Description	Array Signal	Variable	Units
Camber	Wheel angles according to the axle and track location.	1D	$\mathrm{WhlAng}[1,...] = \xi = [\xi_{a,t}]$	rad
Caster			$\mathrm{WhlAng}[2,...] = \eta = [\eta_{a,t}]$	
Toe			$\mathrm{WhlAng}[3,...] = \zeta = [\zeta_{a,t}]$	
Height	Suspension height	1D	H	m
Power	Suspension power dissipation	1D	P_{susp}	W
Energy	Suspension absorbed energy	1D	E_{susp}	J
VehF	Suspension forces applied to the vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\mathrm{VehF} = F_v =$$ $$\begin{bmatrix} F_{vx1,1} & F_{vx1,2} & F_{vx2,1} & F_{vx2,2} \\ F_{vy1,1} & F_{vy1,2} & F_{vy2,1} & F_{vy2,2} \\ F_{vz1,1} & F_{vz1,2} & F_{vz2,1} & F_{vz2,2} \end{bmatrix}$$	N

Signal	Description	Array Signal	Variable	Units
VehM	Suspension moments applied to vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\text{VehM} = M_v =$$ $$\begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$$	N·m
WhlF	Suspension force applied to wheel	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlF} = F_w =$$ $$\begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$	N
WhlP	Track displacement	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlP} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} =$$ $$\begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{wtr_{1,1}} & z_{wtr_{1,2}} & z_{wtr_{2,1}} & z_{wtr_{2,2}} \end{bmatrix}$$	m

Signal	Description	Array Signal	Variable	Units
WhlV	Track velocity	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix}$$ $=$ $$\begin{bmatrix} \dot{x}_{w1,1} & \dot{x}_{w1,2} & \dot{x}_{w2,1} & \dot{x}_{w2,2} \\ \dot{y}_{w1,1} & \dot{y}_{w1,2} & \dot{y}_{w2,1} & \dot{y}_{w2,2} \\ \dot{z}_{w1,1} & \dot{z}_{w1,2} & \dot{z}_{w2,1} & \dot{z}_{w2,2} \end{bmatrix}$$	m/s
WhlAng	Wheel camber, caster, toe angles	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix}$$ $$= \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$	rad

VehF — Suspension force on vehicle
`array`

Longitudinal, lateral, and vertical suspension force at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehF(1,...)` — Suspension force applied to vehicle along the vehicle-fixed *x*-axis (longitudinal)
- `VehF(2,...)` — Suspension force applied to vehicle along the vehicle-fixed *y*-axis (lateral)
- `VehF(3,...)` — Suspension force applied to vehicle along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehF`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension forces applied to the vehicle according to the axle and track locations.

$$\text{VehF} = F_v = \begin{bmatrix} F_{vx1,1} & F_{vx1,2} & F_{vx2,1} & F_{vx2,2} \\ F_{vy1,1} & F_{vy1,2} & F_{vy2,1} & F_{vy2,2} \\ F_{vz1,1} & F_{vz1,2} & F_{vz2,1} & F_{vz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
VehF(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehF(1,2)	1	2	
VehF(1,3)	2	1	
VehF(1,4)	2	2	
VehF(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehF(2,2)	1	2	
VehF(2,3)	2	1	
VehF(2,4)	2	2	
VehF(3,1)	1	1	Vehicle-fixed *z*-axis (vertical)
VehF(3,2)	1	2	
VehF(3,3)	2	1	
VehF(3,4)	2	2	

VehM — Suspension moment on vehicle
array

Longitudinal, lateral, and vertical suspension moment at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N·m. Array dimensions are `3` by the total number of tracks on the vehicle.

- `VehM(1,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *x*-axis (longitudinal)
- `VehM(2,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *y*-axis (lateral)
- `VehM(3,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehM`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension moments applied to vehicle according to the axle and track locations.

$$\text{VehM} = M_v = \begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
VehM(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehM(1,2)	1	2	
VehM(1,3)	2	1	
VehM(1,4)	2	2	
VehM(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehM(2,2)	1	2	

Array Element	Axle	Track	Moment Axis
VehM(2,3)	2	1	
VehM(2,4)	2	2	
VehM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
VehM(3,2)	1	2	
VehM(3,3)	2	1	
VehM(3,4)	2	2	

WhlF — Suspension force on wheel
array

Longitudinal, lateral, and vertical suspension forces at axle `a`, track `t`, applied to the wheel at the axle wheel carrier reference coordinate, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlF(1,...)` — Suspension force on wheel along the vehicle-fixed x-axis (longitudinal)
- `WhlF(2,...)` — Suspension force on wheel along the vehicle-fixed y-axis (lateral)
- `WhlF(3,...)` — Suspension force on wheel along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlF`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlF} = F_w = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlF(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlF(1,2)	1	2	
WhlF(1,3)	2	1	
WhlF(1,4)	2	2	
WhlF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlF(2,2)	1	2	
WhlF(2,3)	2	1	
WhlF(2,4)	2	2	
WhlF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

WhlV — Track velocity
array

Longitudinal, lateral, and vertical track velocity at axle a, track t, in m/s. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlV(1,...) — Track velocity along the vehicle-fixed x-axis (longitudinal)
- WhlV(2,...) — Track velocity along the vehicle-fixed y-axis (lateral)
- WhlV(3,...) — Track velocity along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlV:

- Signal dimensions are [3x4].
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix} = \begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlV(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlV(1,2)	1	2	
WhlV(1,3)	2	1	
WhlV(1,4)	2	2	
WhlV(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlV(2,2)	1	2	
WhlV(2,3)	2	1	
WhlV(2,4)	2	2	
WhlV(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlV(3,2)	1	2	
WhlV(3,3)	2	1	
WhlV(3,4)	2	2	

WhlAng — Wheel camber, caster, toe angles
array

Camber, caster, and toe angles at axle a, track t, in rad. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlAng(1,...) — Camber angle
- WhlAng(2,...) — Caster angle
- WhlAng(3,...) — Toe angle

For example, for a two-axle vehicle with two tracks per axle, the WhlAng:

- Signal dimensions are [3x4].
- Signal contains wheel angles according to the axle and track locations.

$$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$

Array Element	Axle	Track	Angle
WhlAng(1,1)	1	1	Camber
WhlAng(1,2)	1	2	
WhlAng(1,3)	2	1	
WhlAng(1,4)	2	2	
WhlAng(2,1)	1	1	Caster
WhlAng(2,2)	1	2	
WhlAng(2,3)	2	1	
WhlAng(2,4)	2	2	
WhlAng(3,1)	1	1	Toe
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

Parameters

Enable active damping — Include damping
off (default) | off

Include damping

Dependencies

Selecting this parameter creates:

- **Damping coefficient map, f_act_susp_cz**
- **Damping actuator duty cycle breakpoints, f_act_susp_duty_bpt**
- **Damping actuator velocity breakpoints, f_act_susp_zdot_bpt**

Number of axles, NumAxl — Number of axles
2 (default) | scalar

Number of axles, N_a, dimensionless.

Number of tracks by axle, NumTracksByAxl — Number of tracks per axle
[2 2] (default) | vector

Number of tracks per axle, Nt_a, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example, [1,2] represents one track on axle 1 and two tracks on axle 2.

Steered axle enable by axle, StrgEnByAxl — Boolean vector to enable axle steering
[1 0] (default) | vector

Boolean vector that enables axle steering, En_{steer}, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example:

- [1 0] — For a two-axle vehicle, enables axle 1 steering and disables axle 2 steering
- [1 1] — For a two-axle vehicle, enables axle 1 and axle 2 steering

Dependencies

Setting any element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to [1 0]. The input signal array dimensions are [1x2].
- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer_{1,1}} & \delta_{steer_{1,2}} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Anti-sway axle enable by axle, AntiSwayEnByAxl — Boolean vector to enable axle anti-sway
[0 0] (default) | vector

Boolean vector that enables axle anti-sway for axle a, dimensionless. For example, [1 0] enables axle 1 anti-sway and disables axle 2 anti-sway. Vector is 1 by the number of vehicle axles, N_a.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

Suspension

Compliance and Damping - Passive

Suspension spring constant, Kz — Suspension spring constant
64370 (default) | scalar | vector

Linear vertical spring constant for independent suspension tracks on axle a, k_{z_a}, in N/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension spring preload, F0z — Suspension spring preload
9810 (default) | scalar | vector

Vertical preload spring force applied to the wheels on the axle at wheel carrier reference coordinates, F_{z0_a}, in N. Positive preload forces:

- Cause the vehicle to lift.
- Point along the negative vehicle-fixed z-axis.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension shock damping constant, Cz — Suspension shock damping constant
10000 (default) | scalar | vector

Linear vertical damping constant for independent suspension tracks on axle a, c_{z_a}, in Ns/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

To create this parameter, clear **Enable active damping**.

Suspension maximum height, Hmax — Height
0.5 (default) | scalar | vector

Maximum suspension extension or minimum suspension compression height, H_{max}, for axle a before the suspension reaches a hardstop, in m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Compliance and Damping - Active

Damping coefficient map, f_act_susp_cz — Lookup table
[10000 10000;10000 10000] (default) | M-by-N array

Damping coefficient table as a function of active duty cycle and actuator compression velocity, in N·s/m. Each value specifies the damping for a specific combination of actuator duty cycle and velocity. The array dimensions must match the duty cycle, M, and actuator velocity, N, breakpoint vector dimensions.

Dependencies

To create this parameter, clear **Enable active damping**.

Damping actuator duty cycle breakpoints, f_act_susp_duty_bpt — Duty cycle breakpoints
[0 1] (default) | 1-by-M vector

Damping actuator duty cycle breakpoints, dimensionless.

Dependencies

To create this parameter, clear **Enable active damping**.

Damping actuator velocity breakpoints, f_act_susp_zdot_bpt — Velocity breakpoints
[-1 1] (default) | 1-by-N vector

Damping actuator velocity breakpoints, in m/s.

Dependencies

To create this parameter, clear **Enable active damping**.

Geometry

Toe angle at steering center, Toe — Toe angle
0.0349 (default) | scalar

Nominal suspension toe angle at zero steering angle, ζ_{0a}, in rad.

Roll steer vs suspension height slope, RollStrgSlp — Steer angle suspension slope
-0.2269 (default) | scalar | vector

Roll steer angle versus suspension height, m_{htoe_a}, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Toe angle vs steering angle slope, ToeStrgSlp — Toe angle steering slope
0.01 (default) | scalar | vector

Toe angle versus steering angle slope, $m_{toesteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Caster angle at steering center, Caster — Caster angle at steering center
0.0698 (default) | scalar

Nominal suspension caster angle at zero steering angle, η_{0a}, in rad.

Caster angle vs suspension height slope, CasterHslp — Caster angle versus suspension height slope
-0.2269 (default) | scalar | vector

Caster angle versus suspension height, $m_{hcaster_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Caster angle vs steering angle slope, CasterStrgSlp — Caster angle versus steering angle slope
0.01 (default) | scalar | vector

Caster angle versus steering angle slope, $m_{castersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

    - **Toe angle vs steering angle slope, ToeStrgSlp**
    - **Caster angle vs steering angle slope, CasterStrgSlp**
    - **Camber angle vs steering angle slope, CamberStrgSlp**
    - **Suspension height vs steering angle slope, StrgHgtSlp**

Camber angle at steering center, Camber — Camber angle at steering center
0.0698 (default) | scalar

Nominal suspension camber angle at zero steering angle, ξ_{0a}, in rad.

Camber angle vs suspension height slope, CamberHslp — Camber angle versus suspension height slope
-0.2269 (default) | scalar | vector

Camber angle versus suspension height, $m_{hcamber_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Camber angle vs steering angle slope, CamberStrgSlp — Camber angle versus steering angle slope
0.01 (default) | scalar | vector

Camber angle versus steering angle slope, $m_{cambersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:
  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Suspension height vs steering angle slope, StrgHgtSlp — Suspension height versus steering angle slope
0.1432 (default) | scalar | vector

Steering angle to vertical force slope applied at suspension wheel carrier reference point, m_{hsteer_a}, in m/rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:
  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Anti-Sway

Anti-sway arm radius, AntiSwayR — Anti-sway arm radius
0.2 (default) | scalar | vector

Anti-sway arm radius, r, in m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

`Anti-sway arm neutral angle, AntiSwayNtrlAng` **— Anti-sway arm neutral angle**
0.5236 (default) | scalar | vector

Anti-sway arm neutral angle, θ_{0a}, at nominal suspension height, in rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

`Anti-sway torsion spring constant, AntiSwayTrsK` **— Anti-sway torsion spring constant**
5.7296e+03 (default) | scalar | vector

Anti-sway bar torsion spring constant, k_a, in N·m/rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[2] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[3] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary*. ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

Extended Capabilities

C/C++ Code Generation
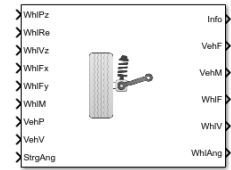Generate C and C++ code using Simulink® Coder™.

See Also

Independent Suspension - MacPherson | Independent Suspension - Mapped

Introduced in R2018a

Independent Suspension - MacPherson

MacPherson independent suspension
Library: Vehicle Dynamics Blockset / Suspension

Description

The Independent Suspension - MacPherson block implements an independent MacPherson suspension for multiple axles with multiple tracks per axle.

The block models the suspension compliance, damping, and geometric effects as functions of the relative positions and velocities of the vehicle and wheel carrier with axle-specific compliance and damping parameters. Using the suspension compliance and damping, the block calculates the suspension force on the vehicle and wheel. The block uses the Z-down coordinate system (defined in SAE J670).

For Each	You Can Specify
Axle	• Multiple tracks • An anti-sway bar for axles with two tracks • Suspension parameters
Track	• Steering angles

The block contains energy-storing spring elements and energy-dissipating damper elements. It does not contain energy-storing mass elements. The block assumes that the vehicle (sprung) and wheel (unsprung) blocks connected to the block store the mass-related suspension energy.

This table summarizes the block parameter settings for a vehicle with:

• Two axles
• Two tracks per axle
• Steering angle input for both tracks on the front axle
• An anti-sway bar on the front axle

Parameter	Setting
Number of axles, NumAxl	2
Number of tracks by axle, NumTracksByAxl	[2 2]
Steered axle enable by axle, StrgEnByAxl	[1 0]
Anti-sway axle enable by axle, AntiSwayEnByAxl	[1 0]

Suspension Compliance and Damping

The block uses a linear spring and damper to model the vertical dynamic effects of the suspension system. Using the relative positions and velocities of the vehicle and wheel carrier, the block

calculates the vertical suspension forces on the wheel and vehicle. The block uses a linear equation that relates the vertical damping and compliance to the suspension height, suspension height rate of change, and absolute value of the steering angles.

The block implements this equation.

$$F_{wz_{a,t}} = F_{z0_a} + k_{z_a}(z_{v_{a,t}} - z_{w_{a,t}} + m_{hsteer_a}|\delta_{steer_{a,t}}|) + c(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}) + F_{zhstop_{a,t}} + F_{zaswy_{a,t}}$$

The damping coefficient, c, depends on the **Enable active damping** parameter setting.

Enable active damping Setting	Damping
off	Constant, $c = c_{z_a}$
on	Lookup table that is a function of active damper duty cycle and actuator velocity $c = f\left(duty, (\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}})\right)$

The block assumes that the suspension elements have no mass. Therefore, the suspension forces and moments applied to the vehicle are equal to the suspension forces and moments applied to the wheel.

$$F_{vx_{a,t}} = F_{wx_{a,t}}$$

$$F_{vy_{a,t}} = F_{wy_{a,t}}$$

$$F_{vz_{a,t}} = -F_{wz_{a,t}}$$

$$M_{vx_{a,t}} = M_{wx_{a,t}} + F_{wy_{a,t}}(Re_{wy_{a,t}} + H_{a,t})$$

$$M_{vy_{a,t}} = M_{wy_{a,t}} + F_{wx_{a,t}}\left(Re_{wx_{a,t}} + H_{a,t}\right)$$

$$M_{vz_{a,t}} = M_{wz_{a,t}}$$

The block sets the wheel positions and velocities equal to the vehicle lateral and longitudinal positions and velocities.

$$x_{w_{a,t}} = x_{v_{a,t}}$$

$$y_{w_{a,t}} = y_{v_{a,t}}$$

$$\dot{x}_{w_{a,t}} = \dot{x}_{v_{a,t}}$$

$$\dot{y}_{w_{a,t}} = \dot{y}_{v_{a,t}}$$

The equations use these variables.

$F_{wz_{a,t}}, M_{wz_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed z-axis
$F_{wx_{a,t}}, M_{wx_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed x-axis
$F_{wy_{a,t}}, M_{wy_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed y-axis

$F_{vz_{a,t}}, M_{vz_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed z-axis
$F_{vx_{a,t}}, M_{vx_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed x-axis
$F_{vy_{a,t}}, M_{vy_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed y-axis
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle a
k_{z_a}	Vertical spring constant applied to tracks on axle a
m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
c_{z_a}	Vertical damping constant applied to tracks on axle a
$Re_{w_{a,t}}$	Effective wheel radius for axle a, track t
$F_{zhstop_{a,t}}$	Vertical hardstop force at axle a, track t, along the vehicle-fixed z-axis
$F_{zaswy_{a,t}}$	Vertical anti-sway force at axle a, track t, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{v_{a,t}}, \dot{x}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{w_{a,t}}, \dot{x}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$y_{v_{a,t}}, \dot{y}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$y_{w_{a,t}}, \dot{y}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$H_{a,t}$	Suspension height at axle a, track t
$Re_{w_{a,t}}$	Effective wheel radius at axle a, track t

Hardstop Forces

The hardstop feedback force, $F_{zhstop_{a,t}}$, that the block applies depends on whether the suspension is compressing or extending. The block applies the force:

- In compression, when the suspension is compressed more than the maximum distance specified by the **Suspension maximum height, Hmax** parameter.
- In extension, when the suspension extension is greater than maximum extension specified by the **Suspension maximum height, Hmax** parameter.

To calculate the force, the block uses a stiffness based on a hyperbolic tangent and exponential scaling.

Anti-Sway Bar

Optionally, the block implements an anti-sway bar force, $F_{zaswy_{a,t}}$, for axles that have two tracks. This figure shows how the anti-sway bar transmits torque between two independent suspension tracks on a shared axle. Each independent suspension applies a torque to the anti-sway bar via a radius arm that extends from the anti-sway bar back to the independent suspension connection point.

To calculate the sway bar force, the block implements these equations.

Calculation	Equation
Anti-sway bar angular deflection for a given axle and track, $\Delta\Theta_{a,t}$	$\theta_{0a} = \tan^{-1}\left(\frac{z_0}{r}\right)$ $\Delta\theta_{a,t} = \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,t}} + z_{v_{a,t}}}{r}\right)$
Anti-sway bar twist angle, Θ_a	$\theta_a = -\tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,1}} + z_{v_{a,1}}}{r}\right)$ $\quad - \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,2}} + z_{v_{a,2}}}{r}\right)$
Anti-sway bar torque, τ_a	$\tau_a = k_a\theta_a$
Anti-sway bar forces applied to the wheel on axle a, track t along wheel-fixed z-axis	$F_{zaswy_{a,1}} = \left(\frac{\tau_a}{r}\right)\cos\left(\theta_{0a} - \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,1}} + z_{v_{a,1}}}{r}\right)\right)$ $F_{zaswy_{a,2}} = \left(\frac{\tau_a}{r}\right)\cos\left(\theta_{0a} - \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,2}} + z_{v_{a,2}}}{r}\right)\right)$

The equations and figure use these variables.

τ_a	Anti-sway bar torque
θ	Anti-sway bar twist angle
θ_{0a}	Initial anti-sway bar twist angle
$\Delta\Theta_{a,t}$	Anti-sway bar angular deflection at axle a, track t
r	Anti-sway bar arm radius
z_0	Vertical distance from anti-sway bar connection point to anti-sway bar centerline
$F_{zsway_{a,t}}$	Anti-sway bar force applied to the wheel on axle a, track t along wheel-fixed z-axis
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Wheel displacement at axle a, track t, along the vehicle-fixed z-axis

Camber, Caster, and Toe Angles

To calculate the camber, caster, and toe angles, block uses linear functions of the suspension height and steering angle.

$$\xi_{a,t} = \xi_{0a} + m_{hcamber_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{cambersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\eta_{a,t} = \eta_{0a} + m_{hcaster_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{castersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\zeta_{a,t} = \zeta_{0a} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equations use these variables.

$\xi_{a,t}$	Camber angle of wheel on axle a, track t
$\eta_{a,t}$	Caster angle of wheel on axle a, track t
$\zeta_{a,t}$	Toe angle of wheel on axle a, track t
ξ_{0a}, η_{0a}, ζ_{0a}	Nominal suspension axle a camber, caster, and toe angles, respectively, at zero steering angle
$m_{hcamber_a}$, $m_{hcaster_a}$, m_{htoe_a}	Camber, caster, and toe angles, respectively, versus suspension height slope for axle a
$m_{cambersteer_a}$, $m_{castersteer_a}$, $m_{toesteer_a}$	Camber, caster, and toe angles, respectively, versus steering angle slope for axle a
m_{hsteer_a}	Steering angle versus vertical force slope for axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Steering Angles

Optionally, you can input steering angles for the tracks. To calculate the steering angles for the wheels, the block offsets the input steering angles with a linear function of the suspension height.

$$\delta_{whlsteer_{a,t}} = \delta_{steer_{a,t}} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equation uses these variables.

$m_{toesteer_a}$	Axle a toe angle versus steering angle slope
m_{hsteer_a}	Axle a steering angle versus vertical force slope
m_{htoe_a}	Axle a toe angle versus suspension height slope
$\delta_{whlsteer_{a,t}}$	Wheel steering angle for axle a, track t
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Power and Energy

The block calculates these suspension characteristics for each axle, a, track, t.

Calculation	Equation
Dissipated power, $P_{susp_{a,t}}$	$P_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Absorbed energy, $E_{susp_{a,t}}$	$E_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Suspension height, $H_{a,t}$	$H_{a,t} = -\left(z_{v_{a,t}} - z_{w_{a,t}} + \dfrac{F_{z0_a}}{k_{z_a}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right)$
Distance from wheel carrier center to tire/road interface	$z_{wtr_{a,t}} = Re_{w_{a,t}} + H_{a,t}$

The equations use these variables.

m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle `a`
$\delta_{steer_{a,t}}$	Steering angle input for axle `a`, track `t`
$Re_{w_{a,t}}$	Axle `a`, track `t` effective wheel radius from wheel carrier center to tire/road interface
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle `a`
$z_{wtr_{a,t}}$	Distance from wheel carrier center to tire/road interface, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle `a`, track `t`, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle `a`, track `t`, along the vehicle-fixed z-axis

Ports

Input

WhlPz — Track z-axis displacement
array

Track displacement, z_w, along wheel-fixed z-axis, in m. Array dimensions are `1` by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlPz`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlPz} = z_w = \begin{bmatrix} z_{w_{1,1}} & z_{w_{1,2}} & z_{w_{2,1}} & z_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlPz(1,1)	1	1
WhlPz(1,2)	1	2
WhlPz(1,3)	2	1
WhlPz(1,4)	2	2

WhlRe — Wheel effective radius
array

Effective wheel radius, Re_w, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlRe`:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlRe} = Re_w = \begin{bmatrix} Re_{w1,1} & Re_{w1,2} & Re_{w2,1} & Re_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlRe(1,1)	1	1
WhlRe(1,2)	1	2
WhlRe(1,3)	2	1
WhlRe(1,4)	2	2

`WhlVz` — Track z-axis velocity
array

Track velocity, $\dot{z}_w$, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlVz`:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlVz} = \dot{z}_w = \begin{bmatrix} \dot{z}_{w1,1} & \dot{z}_{w1,2} & \dot{z}_{w2,1} & \dot{z}_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlVz(1,1)	1	1
WhlVz(1,2)	1	2
WhlVz(1,3)	2	1
WhlVz(1,4)	2	2

`WhlFx` — Longitudinal wheel force on vehicle
array

Longitudinal wheel force applied to vehicle, F_{wx}, along the vehicle-fixed x-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlFx`:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlFx} = F_{wx} = \begin{bmatrix} F_{wx1,1} & F_{wx1,2} & F_{wx2,1} & F_{wx2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFx(1,1)	1	1
WhlFx(1,2)	1	2
WhlFx(1,3)	2	1
WhlFx(1,4)	2	2

WhlFy — Lateral wheel force on vehicle
array

Lateral wheel force applied to vehicle, F_{wy}, along the vehicle-fixed y-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFy:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlFy} = F_{wy} = \begin{bmatrix} F_{wy1,1} & F_{wy1,2} & F_{wy2,1} & F_{wy2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFy(1,1)	1	1
WhlFy(1,2)	1	2
WhlFy(1.3)	2	1
WhlFy(1,4)	2	2

WhlM — Suspension moment on wheel
array

Longitudinal, lateral, and vertical suspension moments at axle a, track t, applied to the wheel at the axle wheel carrier reference coordinate, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlM(1,...) — Suspension moment applied to the wheel about the vehicle-fixed x-axis (longitudinal)
- WhlM(2,...) — Suspension moment applied to the wheel about the vehicle-fixed y-axis (lateral)
- WhlM(3,...) — Suspension moment applied to the wheel about the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlM:

- Signal dimensions are [3x4].
- Signal contains suspension moments applied to four wheels according to their axle and track locations.

$$\text{WhlM} = M_w = \begin{bmatrix} M_{wx1,1} & M_{wx1,2} & M_{wx2,1} & M_{wx2,2} \\ M_{wy1,1} & M_{wy1,2} & M_{wy2,1} & M_{wy2,2} \\ M_{wz1,1} & M_{wz1,2} & M_{wz2,1} & M_{wz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
WhlM(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlM(1,2)	1	2	
WhlM(1,3)	2	1	
WhlM(1,4)	2	2	
WhlM(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlM(2,2)	1	2	
WhlM(2,3)	2	1	
WhlM(2,4)	2	2	
WhlM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlM(3,2)	1	2	
WhlM(3,3)	2	1	
WhlM(3,4)	2	2	

VehP — Vehicle displacement
array

Vehicle displacement from axle `a`, track `t` along vehicle-fixed coordinate system, in m. Array dimensions are 3 by the total number of tracks on the vehicle.

- VehP(1,...) — Vehicle displacement from track, x_v, along the vehicle-fixed x-axis
- VehP(2,...) — Vehicle displacement from track, y_v, along the vehicle-fixed y-axis
- VehP(3,...) — Vehicle displacement from track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the VehP:

- Signal dimensions are [3x4].
- Signal contains four track displacements according to their axle and track locations.

$$\text{VehP} = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} x_{v1,1} & x_{v1,2} & x_{v2,1} & x_{v2,2} \\ y_{v1,1} & y_{v1,2} & y_{v2,1} & y_{v2,2} \\ z_{v1,1} & z_{v1,2} & z_{v2,1} & z_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehP(1,1)	1	1	Vehicle-fixed x-axis
VehP(1,2)	1	2	
VehP(1,3)	2	1	
VehP(1,4)	2	2	
VehP(2,1)	1	1	Vehicle-fixed y-axis
VehP(2,2)	1	2	
VehP(2,3)	2	1	
VehP(2,4)	2	2	

Array Element	Axle	Track	Axis
VehP(3,1)	1	1	Vehicle-fixed z-axis
VehP(3,2)	1	2	
VehP(3,3)	2	1	
VehP(3,4)	2	2	

VehV — Vehicle velocity
array

Vehicle velocity at axle `a`, track `t` along vehicle-fixed coordinate system, in m. Input array dimensions are 3 by `a*t`.

- VehV(1,...) — Vehicle velocity at track, x_v, along the vehicle-fixed x-axis
- VehV(2,...) — Vehicle velocity at track, y_v, along the vehicle-fixed y-axis
- VehV(3,...) — Vehicle velocity at track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehV`:

- Signal dimensions are [3x4].
- Signal contains 4 track velocities according to their axle and track locations.

$$\text{VehV} = \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{z}_v \end{bmatrix} = \begin{bmatrix} \dot{x}_{v1,1} & \dot{x}_{v1,2} & \dot{x}_{v2,1} & \dot{x}_{v2,2} \\ \dot{y}_{v1,1} & \dot{y}_{v1,2} & \dot{y}_{v2,1} & \dot{y}_{v2,2} \\ \dot{z}_{v1,1} & \dot{z}_{v1,2} & \dot{z}_{v2,1} & \dot{z}_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehV(1,1)	1	1	Vehicle-fixed x-axis
VehV(1,2)	1	2	
VehV(1,3)	2	1	
VehV(1,4)	2	2	
VehV(2,1)	1	1	Vehicle-fixed y-axis
VehV(2,2)	1	2	
VehV(2,3)	2	1	
VehV(2,4)	2	2	
VehV(3,1)	1	1	Vehicle-fixed z-axis
VehV(3,2)	1	2	
VehV(3,3)	2	1	
VehV(3,4)	2	2	

StrgAng — Steering angle, optional
array

Optional steering angle for each wheel, δ. Input array dimensions are 1 by the number of steered tracks.

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.
- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer1,\,1} & \delta_{steer1,\,2} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Output

Info — Bus signal
bus

Bus signal containing block values. The signals are arrays that depend on the track location.

For example, here are the indices for a two-axle, two-track vehicle. The total number of tracks is four.

- 1D array signal (1-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

- 3D array signal (3-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

Array Element	Axle	Track
(2,1)	1	1
(2,2)	1	2
(2,3)	2	1
(2,4)	2	2
(3,1)	1	1
(3,2)	1	2
(3,3)	2	1
(3,4)	2	2

Signal	Description	Array Signal	Variable	Units
Camber	Wheel angles according to the axle and track location.	1D	$\text{WhlAng}[1,\ldots] = \xi = [\xi_{a,t}]$	rad
Caster			$\text{WhlAng}[2,\ldots] = \eta = [\eta_{a,t}]$	
Toe			$\text{WhlAng}[3,\ldots] = \zeta = [\zeta_{a,t}]$	
Height	Suspension height	1D	H	m
Power	Suspension power dissipation	1D	P_{susp}	W
Energy	Suspension absorbed energy	1D	E_{susp}	J
VehF	Suspension forces applied to the vehicle	3D	For a two-axle, two tracks per axle vehicle: $\text{VehF} = F_v =$ $\begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$	N

Signal	Description	Array Signal	Variable	Units
VehM	Suspension moments applied to vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\text{VehM} = M_v =$$ $$\begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$$	N·m
WhlF	Suspension force applied to wheel	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlF} = F_w =$$ $$\begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$	N
WhlP	Track displacement	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlP} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} =$$ $$\begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{wtr_{1,1}} & z_{wtr_{1,2}} & z_{wtr_{2,1}} & z_{wtr_{2,2}} \end{bmatrix}$$	m

Signal	Description	Array Signal	Variable	Units
WhlV	Track velocity	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix}$$ $$=$$ $$\begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$	m/s
WhlAng	Wheel camber, caster, toe angles	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix}$$ $$= \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$	rad

VehF — Suspension force on vehicle
`array`

Longitudinal, lateral, and vertical suspension force at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehF(1,...)` — Suspension force applied to vehicle along the vehicle-fixed *x*-axis (longitudinal)
- `VehF(2,...)` — Suspension force applied to vehicle along the vehicle-fixed *y*-axis (lateral)
- `VehF(3,...)` — Suspension force applied to vehicle along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehF`:

- Signal dimensions are [3x4].
- Signal contains suspension forces applied to the vehicle according to the axle and track locations.

$$\text{VehF} = F_v = \begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
VehF(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehF(1,2)	1	2	
VehF(1,3)	2	1	
VehF(1,4)	2	2	
VehF(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehF(2,2)	1	2	
VehF(2,3)	2	1	
VehF(2,4)	2	2	
VehF(3,1)	1	1	Vehicle-fixed *z*-axis (vertical)
VehF(3,2)	1	2	
VehF(3,3)	2	1	
VehF(3,4)	2	2	

VehM — Suspension moment on vehicle
array

Longitudinal, lateral, and vertical suspension moment at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehM(1,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *x*-axis (longitudinal)
- `VehM(2,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *y*-axis (lateral)
- `VehM(3,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehM`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension moments applied to vehicle according to the axle and track locations.

$$\text{VehM} = M_v = \begin{bmatrix} M_{vx1,1} & M_{vx1,2} & M_{vx2,1} & M_{vx2,2} \\ M_{vy1,1} & M_{vy1,2} & M_{vy2,1} & M_{vy2,2} \\ M_{vz1,1} & M_{vz1,2} & M_{vz2,1} & M_{vz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
VehM(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehM(1,2)	1	2	
VehM(1,3)	2	1	
VehM(1,4)	2	2	
VehM(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehM(2,2)	1	2	

Array Element	Axle	Track	Moment Axis
VehM(2,3)	2	1	
VehM(2,4)	2	2	
VehM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
VehM(3,2)	1	2	
VehM(3,3)	2	1	
VehM(3,4)	2	2	

WhlF — Suspension force on wheel
array

Longitudinal, lateral, and vertical suspension forces at axle a, track t, applied to the wheel at the axle wheel carrier reference coordinate, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlF(1,...) — Suspension force on wheel along the vehicle-fixed x-axis (longitudinal)
- WhlF(2,...) — Suspension force on wheel along the vehicle-fixed y-axis (lateral)
- WhlF(3,...) — Suspension force on wheel along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlF:

- Signal dimensions are [3x4].
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlF} = F_w = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlF(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlF(1,2)	1	2	
WhlF(1,3)	2	1	
WhlF(1,4)	2	2	
WhlF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlF(2,2)	1	2	
WhlF(2,3)	2	1	
WhlF(2,4)	2	2	
WhlF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

WhlV — Track velocity
array

Longitudinal, lateral, and vertical track velocity at axle `a`, track `t`, in m/s. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlV(1,...)` — Track velocity along the vehicle-fixed *x*-axis (longitudinal)
- `WhlV(2,...)` — Track velocity along the vehicle-fixed *y*-axis (lateral)
- `WhlV(3,...)` — Track velocity along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlV`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix} = \begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
`WhlV(1,1)`	1	1	Vehicle-fixed *x*-axis (longitudinal)
`WhlV(1,2)`	1	2	
`WhlV(1,3)`	2	1	
`WhlV(1,4)`	2	2	
`WhlV(2,1)`	1	1	Vehicle-fixed *y*-axis (lateral)
`WhlV(2,2)`	1	2	
`WhlV(2,3)`	2	1	
`WhlV(2,4)`	2	2	
`WhlV(3,1)`	1	1	Vehicle-fixed *z*-axis (vertical)
`WhlV(3,2)`	1	2	
`WhlV(3,3)`	2	1	
`WhlV(3,4)`	2	2	

WhlAng — Wheel camber, caster, toe angles
array

Camber, caster, and toe angles at axle `a`, track `t`, in rad. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlAng(1,...)` — Camber angle
- `WhlAng(2,...)` — Caster angle
- `WhlAng(3,...)` — Toe angle

For example, for a two-axle vehicle with two tracks per axle, the `WhlAng`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel angles according to the axle and track locations.

$$
\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}
$$

Array Element	Axle	Track	Angle
WhlAng(1,1)	1	1	Camber
WhlAng(1,2)	1	2	
WhlAng(1,3)	2	1	
WhlAng(1,4)	2	2	
WhlAng(2,1)	1	1	Caster
WhlAng(2,2)	1	2	
WhlAng(2,3)	2	1	
WhlAng(2,4)	2	2	
WhlAng(3,1)	1	1	Toe
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

Parameters

Enable active damping — Include damping
off (default) | off

Include damping

Dependencies

Selecting this parameter creates:

- **Damping coefficient map, f_act_susp_cz**
- **Damping actuator duty cycle breakpoints, f_act_susp_duty_bpt**
- **Damping actuator velocity breakpoints, f_act_susp_zdot_bpt**

Number of axles, NumAxl — Number of axles
2 (default) | scalar

Number of axles, N_a, dimensionless.

Number of tracks by axle, NumTracksByAxl — Number of tracks per axle
[2 2] (default) | vector

Number of tracks per axle, Nt_a, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example, [1,2] represents one track on axle 1 and two tracks on axle 2.

Steered axle enable by axle, StrgEnByAxl — Boolean vector to enable axle steering
[1 0] (default) | vector

Boolean vector that enables axle steering, En_{steer}, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example:

- [1 0] — For a two-axle vehicle, enables axle 1 steering and disables axle 2 steering
- [1 1] — For a two-axle vehicle, enables axle 1 and axle 2 steering

Dependencies

Setting any element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the StrgAng port, set **Steered axle enable by axle, StrgEnByAxl** to [1 0]. The input signal array dimensions are [1x2].
- The StrgAng signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer_{1,1}} & \delta_{steer_{1,2}} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Anti-sway axle enable by axle, AntiSwayEnByAxl — Boolean vector to enable axle anti-sway
[0 0] (default) | vector

Boolean vector that enables axle anti-sway for axle a, dimensionless. For example, [1 0] enables axle 1 anti-sway and disables axle 2 anti-sway. Vector is 1 by the number of vehicle axles, N_a.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

Suspension

Compliance and Damping - Passive

Suspension spring constant, Kz — Suspension spring constant
64370 (default) | scalar | vector

Linear vertical spring constant for independent suspension tracks on axle a, k_{z_a}, in N/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension spring preload, F0z — Suspension spring preload
9810 (default) | scalar | vector

Vertical preload spring force applied to the wheels on the axle at wheel carrier reference coordinates, F_{z0_a}, in N. Positive preload forces:

- Cause the vehicle to lift.
- Point along the negative vehicle-fixed z-axis.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension shock damping constant, Cz — Suspension shock damping constant
10000 (default) | scalar | vector

Linear vertical damping constant for independent suspension tracks on axle a, c_{z_a}, in Ns/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

To create this parameter, clear **Enable active damping**.

Suspension maximum height, Hmax — Height
0.5 (default) | scalar | vector

Maximum suspension extension or minimum suspension compression height, H_{max}, for axle a before the suspension reaches a hardstop, in m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Compliance and Damping - Active

Damping coefficient map, f_act_susp_cz — Lookup table
[10000 10000;10000 10000] (default) | M-by-N array

Damping coefficient table as a function of active duty cycle and actuator compression velocity, in N·s/m. Each value specifies the damping for a specific combination of actuator duty cycle and velocity. The array dimensions must match the duty cycle, M, and actuator velocity, N, breakpoint vector dimensions.

Dependencies

To create this parameter, clear **Enable active damping**.

Damping actuator duty cycle breakpoints, f_act_susp_duty_bpt — Duty cycle breakpoints
[0 1] (default) | 1-by-M vector

Damping actuator duty cycle breakpoints, dimensionless.

Dependencies

To create this parameter, clear **Enable active damping**.

Damping actuator velocity breakpoints, f_act_susp_zdot_bpt — Velocity breakpoints
[-1 1] (default) | 1-by-N vector

Damping actuator velocity breakpoints, in m/s.

Dependencies

To create this parameter, clear **Enable active damping**.

Geometry

Toe angle at steering center, Toe — Toe angle
0.0349 (default) | scalar

Nominal suspension toe angle at zero steering angle, ζ_{0a}, in rad.

Roll steer vs suspension height slope, RollStrgSlp — Steer angle suspension slope
-0.2269 (default) | scalar | vector

Roll steer angle versus suspension height, m_{htoe_a}, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Toe angle vs steering angle slope, ToeStrgSlp — Toe angle steering slope
0.01 (default) | scalar | vector

Toe angle versus steering angle slope, $m_{toesteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Caster angle at steering center, Caster — Caster angle at steering center
0.0698 (default) | scalar

Nominal suspension caster angle at zero steering angle, η_{0a}, in rad.

Caster angle vs suspension height slope, CasterHslp — Caster angle versus suspension height slope
-0.2269 (default) | scalar | vector

Caster angle versus suspension height, $m_{hcaster_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Caster angle vs steering angle slope, CasterStrgSlp — Caster angle versus steering angle slope
0.01 (default) | scalar | vector

Caster angle versus steering angle slope, $m_{castersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Camber angle at steering center, Camber — Camber angle at steering center
0.0698 (default) | scalar

Nominal suspension camber angle at zero steering angle, ξ_{0a}, in rad.

Camber angle vs suspension height slope, CamberHslp — Camber angle versus suspension height slope
-0.2269 (default) | scalar | vector

Camber angle versus suspension height, $m_{hcamber_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Camber angle vs steering angle slope, CamberStrgSlp — Camber angle versus steering angle slope
0.01 (default) | scalar | vector

Camber angle versus steering angle slope, $m_{cambersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:
  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

`Suspension height vs steering angle slope, StrgHgtSlp` — Suspension height versus steering angle slope
`0.1432` (default) | `scalar` | `vector`

Steering angle to vertical force slope applied at suspension wheel carrier reference point, m_{hsteer_a}, in m/rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:
  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Anti-Sway

`Anti-sway arm radius, AntiSwayR` — Anti-sway arm radius
`0.2` (default) | `scalar` | `vector`

Anti-sway arm radius, r, in m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

`Anti-sway arm neutral angle, AntiSwayNtrlAng` — Anti-sway arm neutral angle
`0.5236` (default) | `scalar` | `vector`

Anti-sway arm neutral angle, θ_{0a}, at nominal suspension height, in rad.

Vector is `1` by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

`Anti-sway torsion spring constant, AntiSwayTrsK` — Anti-sway torsion spring constant
`5.7296e+03` (default) | `scalar` | `vector`

Anti-sway bar torsion spring constant, k_a, in N·m/rad.

Vector is `1` by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[2] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[3] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary*. ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

Extended Capabilities

C/C++ Code Generation
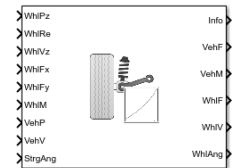Generate C and C++ code using Simulink® Coder™.

See Also

Independent Suspension - Double Wishbone | Independent Suspension - Mapped

Introduced in R2018a

Independent Suspension - Mapped

Mapped independent suspension
Library: Vehicle Dynamics Blockset / Suspension

Description

The Independent Suspension - Mapped block implements a mapped independent suspension for multiple axles with multiple tracks per axle. You can use the block to model suspension geometry, compliance, and damping effects from measured or simulated suspension response data.

The block models the suspension compliance, damping, and geometric effects as functions of the relative positions and velocities of the vehicle and wheel carrier with axle-specific compliance and damping parameters. Using the suspension compliance and damping, the block calculates the suspension force on the vehicle and wheel. The block uses the Z-down coordinate system (defined in SAE J670).

For Each	You Can Specify
Axle	• Multiple tracks • An anti-sway bar for axles with two tracks • Suspension parameters
Track	• Steering angles

The block contains energy-storing spring elements and energy-dissipating damper elements. It does not contain energy-storing mass elements. The block assumes that the vehicle (sprung) and wheel (unsprung) blocks connected to the block store the mass-related suspension energy.

This table summarizes the block parameter settings for a vehicle with:

• Two axles
• Two tracks per axle
• Steering angle input for both tracks on the front axle
• An anti-sway bar on the front axle

Parameter	Setting
Number of axles, NumAxl	2
Number of tracks by axle, NumTracksByAxl	[2 2]
Steered axle enable by axle, StrgEnByAxl	[1 0]
Anti-sway axle enable by axle, AntiSwayEnByAxl	[1 0]

Suspension Compliance and Damping

The block uses a lookup table that relates the vertical damping and compliance to the suspension height, suspension height rate of change, and steering angle. You can calibrate the wheel force lookup table so that steering angle changes from the nominal center position generate a force that increases the vehicle height.

The block implements these equations.

$$F_{wzlookup_a} = f(z_{v_{a,t}} - z_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}})$$

$$F_{wz_{a,t}} = F_{wzlookup_a} + F_{zaswy_{a,t}}$$

The block assumes that the suspension elements have no mass. Therefore, the suspension forces and moments applied to the vehicle are equal to the suspension forces and moments applied to the wheel.

$$F_{vx_{a,t}} = F_{wx_{a,t}}$$
$$F_{vy_{a,t}} = F_{wy_{a,t}}$$
$$F_{vz_{a,t}} = -F_{wz_{a,t}}$$
$$M_{vx_{a,t}} = M_{wx_{a,t}} + F_{wy_{a,t}}(Re_{wy_{a,t}} + H_{a,t})$$
$$M_{vy_{a,t}} = M_{wy_{a,t}} + F_{wx_{a,t}}(Re_{wx_{a,t}} + H_{a,t})$$
$$M_{vz_{a,t}} = M_{wz_{a,t}}$$

The block sets the wheel positions and velocities equal to the vehicle lateral and longitudinal positions and velocities.

$$x_{w_{a,t}} = x_{v_{a,t}}$$
$$y_{w_{a,t}} = y_{v_{a,t}}$$
$$\dot{x}_{w_{a,t}} = \dot{x}_{v_{a,t}}$$
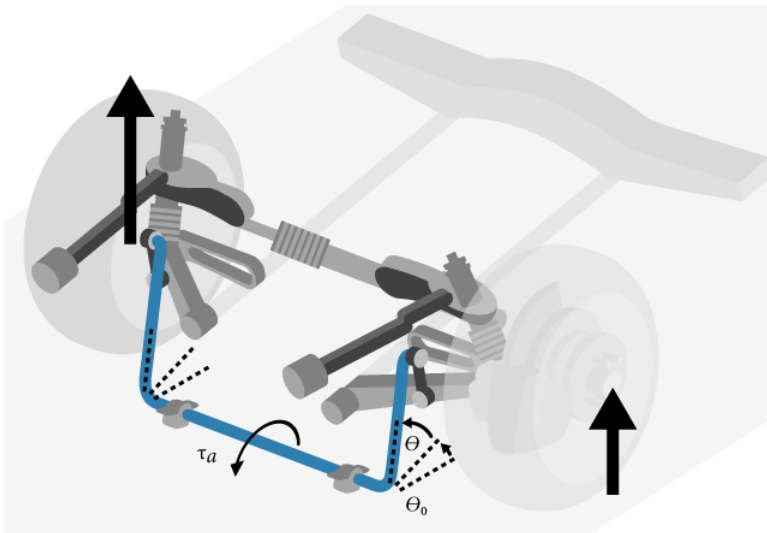$$\dot{y}_{w_{a,t}} = \dot{y}_{v_{a,t}}$$

The equations use these variables.

$F_{wz_{a,t}}, M_{wz_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed z-axis
$F_{wx_{a,t}}, M_{wx_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed x-axis
$F_{wy_{a,t}}, M_{wy_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed y-axis
$F_{vz_{a,t}}, M_{vz_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed z-axis
$F_{vx_{a,t}}, M_{vx_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed x-axis
$F_{vy_{a,t}}, M_{vy_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed y-axis

F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle **a**
k_{z_a}	Vertical spring constant applied to tracks on axle **a**
m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle **a**
$\delta_{steer_{a,t}}$	Steering angle input for axle **a**, track **t**
c_{z_a}	Vertical damping constant applied to tracks on axle **a**
$Re_{w_{a,t}}$	Effective wheel radius for axle **a**, track **t**
$F_{zhstop_{a,t}}$	Vertical hardstop force at axle **a**, track **t**, along the vehicle-fixed z-axis
$F_{zaswy_{a,t}}$	Vertical anti-sway force at axle **a**, track **t**, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle **a**, track **t**, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle **a**, track **t**, along the vehicle-fixed z-axis
$x_{v_{a,t}}, \dot{x}_{v_{a,t}}$	Vehicle displacement and velocity at axle **a**, track **t**, along the vehicle-fixed z-axis
$x_{w_{a,t}}, \dot{x}_{w_{a,t}}$	Track displacement and velocity at axle **a**, track **t**, along the vehicle-fixed z-axis
$y_{v_{a,t}}, \dot{y}_{v_{a,t}}$	Vehicle displacement and velocity at axle **a**, track **t**, along the vehicle-fixed y-axis
$y_{w_{a,t}}, \dot{y}_{w_{a,t}}$	Track displacement and velocity at axle **a**, track **t**, along the vehicle-fixed y-axis
$H_{a,t}$	Suspension height at axle **a**, track **t**
$Re_{w_{a,t}}$	Effective wheel radius at axle **a**, track **t**

Anti-Sway Bar

Optionally, the block implements an anti-sway bar force, $F_{zaswy_{a,t}}$, for axles that have two tracks. This figure shows how the anti-sway bar transmits torque between two independent suspension tracks on a shared axle. Each independent suspension applies a torque to the anti-sway bar via a radius arm that extends from the anti-sway bar back to the independent suspension connection point.

To calculate the sway bar force, the block implements these equations.

Calculation	Equation
Anti-sway bar angular deflection for a given axle and track, $\Delta\Theta_{a,t}$	$\theta_{0a} = \tan^{-1}\left(\frac{z_0}{r}\right)$ $\Delta\theta_{a,t} = \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,t}} + z_{v_{a,t}}}{r}\right)$
Anti-sway bar twist angle, Θ_a	$\theta_a = -\tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,1}} + z_{v_{a,1}}}{r}\right)$ $-\tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,2}} + z_{v_{a,2}}}{r}\right)$
Anti-sway bar torque, τ_a	$\tau_a = k_a\theta_a$
Anti-sway bar forces applied to the wheel on axle a, track t along wheel-fixed z-axis	$F_{zaswy_{a,1}} = \left(\frac{\tau_a}{r}\right)\cos\left(\theta_{0a} - \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,1}} + z_{v_{a,1}}}{r}\right)\right)$ $F_{zaswy_{a,2}} = \left(\frac{\tau_a}{r}\right)\cos\left(\theta_{0a} - \tan^{-1}\left(\frac{r\tan\theta_{0a} - z_{w_{a,2}} + z_{v_{a,2}}}{r}\right)\right)$

The equations and figure use these variables.

τ_a	Anti-sway bar torque
θ	Anti-sway bar twist angle
θ_{0a}	Initial anti-sway bar twist angle
$\Delta\Theta_{a,t}$	Anti-sway bar angular deflection at axle a, track t
r	Anti-sway bar arm radius
z_0	Vertical distance from anti-sway bar connection point to anti-sway bar centerline
$F_{zsway_{a,t}}$	Anti-sway bar force applied to the wheel on axle a, track t along wheel-fixed z-axis
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Wheel displacement at axle a, track t, along the vehicle-fixed z-axis

Camber, Caster, and Toe Angles

To calculate the camber, caster, and toe angles, the block uses a lookup table, $G_{alookup}$, that is a function of the suspension height and steering angle.

$$[\xi_{a,t} \ \eta_{a,t} \ \zeta_{a,t}] = G_{alookup}f(z_{w_{a,t}} - z_{v_{a,t}}, \delta_{steer_{a,t}})$$

The equations use these variables.

$\xi_{a,t}$	Camber angle of wheel on axle a, track t
$\eta_{a,t}$	Caster angle of wheel on axle a, track t
$\zeta_{a,t}$	Toe angle of wheel on axle a, track t
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along vehicle-fixed z-axis
$z_{w_{a,t}}$	Wheel displacement at axle a, track t, along vehicle-fixed z-axis

Steering Angles

Optionally, you can input steering angles for the tracks. To calculate the steering angles for the wheels, the block offsets the input steering angles as a function of the suspension height. For the calculation, the block uses a lookup table, $G_{alookup}$, that is a function of the suspension position and steering angle.

$$\delta_{whlsteer_{a,t}} = \delta_{steer_{a,t}} + G_{alookup}f\left(z_{w_{a,t}} - z_{v_{a,t}}, \delta_{steer_{a,t}}\right)$$

The equation uses these variables.

$\delta_{whlsteer_{a,t}}$	Wheel steering angle for axle a, track t
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Wheel displacement at axle a, track t, along the vehicle-fixed z-axis

Power and Energy

The block calculates these suspension characteristics for each axle, a, track, t.

Calculation	Equation
Dissipated power, $P_{susp_{a,t}}$	$P_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Absorbed energy, $E_{susp_{a,t}}$	$E_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Suspension height, $H_{a,t}$	$H_{a,t} = -\left(z_{v_{a,t}} - z_{w_{a,t}} - \text{median}(f\_susp\_dz\_bp)\right)$
Distance from wheel carrier center to tire/road interface	$z_{wtr_{a,t}} = Re_{w_{a,t}} + H_{a,t}$

The equations use these variables.

m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$Re_{w_{a,t}}$	Axle a, track t effective wheel radius from wheel carrier center to tire/road interface
$f\_susp\_dz\_bp$	Vertical axis suspension height breakpoints
$z_{wtr_{a,t}}$	Distance from wheel carrier center to tire/road interface, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis

Ports

Input

WhlPz — Track z-axis displacement
array

Track displacement, z_w, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlPz`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlPz} = z_w = \begin{bmatrix} z_{w1,1} & z_{w1,2} & z_{w2,1} & z_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlPz(1,1)	1	1
WhlPz(1,2)	1	2
WhlPz(1,3)	2	1
WhlPz(1,4)	2	2

WhlRe — Wheel effective radius
array

Effective wheel radius, Re_w, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlRe`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlRe} = Re_w = \begin{bmatrix} Re_{w1,1} & Re_{w1,2} & Re_{w2,1} & Re_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlRe(1,1)	1	1
WhlRe(1,2)	1	2
WhlRe(1,3)	2	1
WhlRe(1,4)	2	2

WhlVz — Track z-axis velocity
array

Track velocity, $\dot{z}_w$, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlVz`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlVz} = \dot{z}_w = \begin{bmatrix} \dot{z}_{w1,1} & \dot{z}_{w1,2} & \dot{z}_{w2,1} & \dot{z}_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlVz(1,1)	1	1
WhlVz(1,2)	1	2
WhlVz(1,3)	2	1
WhlVz(1,4)	2	2

WhlFx — Longitudinal wheel force on vehicle
array

Longitudinal wheel force applied to vehicle, F_{wx}, along the vehicle-fixed *x*-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFx:

- Signal array dimensions are [1x4].

- Array dimensions are axle by track.

$$\text{WhlFx} = F_{wx} = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlFx(1,1)	1	1
WhlFx(1,2)	1	2
WhlFx(1,3)	2	1
WhlFx(1,4)	2	2

WhlFy — Lateral wheel force on vehicle
array

Lateral wheel force applied to vehicle, F_{wy}, along the vehicle-fixed *y*-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFy:

- Signal array dimensions are [1x4].

- Array dimensions are axle by track.

$$\text{WhlFy} = F_{wy} = \begin{bmatrix} F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlFy(1,1)	1	1
WhlFy(1,2)	1	2
WhlFy(1.3)	2	1
WhlFy(1,4)	2	2

WhlM — Suspension moment on wheel
array

Longitudinal, lateral, and vertical suspension moments at axle a, track t, applied to the wheel at the axle wheel carrier reference coordinate, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlM(1,...)` — Suspension moment applied to the wheel about the vehicle-fixed *x*-axis (longitudinal)
- `WhlM(2,...)` — Suspension moment applied to the wheel about the vehicle-fixed *y*-axis (lateral)
- `WhlM(3,...)` — Suspension moment applied to the wheel about the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlM`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension moments applied to four wheels according to their axle and track locations.

$$\text{WhlM} = M_w = \begin{bmatrix} M_{wx_{1,1}} & M_{wx_{1,2}} & M_{wx_{2,1}} & M_{wx_{2,2}} \\ M_{wy_{1,1}} & M_{wy_{1,2}} & M_{wy_{2,1}} & M_{wy_{2,2}} \\ M_{wz_{1,1}} & M_{wz_{1,2}} & M_{wz_{2,1}} & M_{wz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
`WhlM(1,1)`	1	1	Vehicle-fixed *x*-axis (longitudinal)
`WhlM(1,2)`	1	2	
`WhlM(1,3)`	2	1	
`WhlM(1,4)`	2	2	
`WhlM(2,1)`	1	1	Vehicle-fixed *y*-axis (lateral)
`WhlM(2,2)`	1	2	
`WhlM(2,3)`	2	1	
`WhlM(2,4)`	2	2	
`WhlM(3,1)`	1	1	Vehicle-fixed *z*-axis (vertical)
`WhlM(3,2)`	1	2	
`WhlM(3,3)`	2	1	
`WhlM(3,4)`	2	2	

VehP — Vehicle displacement
`array`

Vehicle displacement from axle a, track t along vehicle-fixed coordinate system, in m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehP(1,...)` — Vehicle displacement from track, x_v, along the vehicle-fixed *x*-axis
- `VehP(2,...)` — Vehicle displacement from track, y_v, along the vehicle-fixed *y*-axis
- `VehP(3,...)` — Vehicle displacement from track, z_v, along the vehicle-fixed *z*-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehP`:

- Signal dimensions are `[3x4]`.

- Signal contains four track displacements according to their axle and track locations.

$$
\text{VehP} = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} x_{v1,1} & x_{v1,2} & x_{v2,1} & x_{v2,2} \\ y_{v1,1} & y_{v1,2} & y_{v2,1} & y_{v2,2} \\ z_{v1,1} & z_{v1,2} & z_{v2,1} & z_{v2,2} \end{bmatrix}
$$

Array Element	Axle	Track	Axis
VehP(1,1)	1	1	Vehicle-fixed x-axis
VehP(1,2)	1	2	
VehP(1,3)	2	1	
VehP(1,4)	2	2	
VehP(2,1)	1	1	Vehicle-fixed y-axis
VehP(2,2)	1	2	
VehP(2,3)	2	1	
VehP(2,4)	2	2	
VehP(3,1)	1	1	Vehicle-fixed z-axis
VehP(3,2)	1	2	
VehP(3,3)	2	1	
VehP(3,4)	2	2	

VehV — Vehicle velocity
`array`

Vehicle velocity at axle `a`, track `t` along vehicle-fixed coordinate system, in m. Input array dimensions are 3 by `a*t`.

- `VehV(1,...)` — Vehicle velocity at track, x_v, along the vehicle-fixed x-axis
- `VehV(2,...)` — Vehicle velocity at track, y_v, along the vehicle-fixed y-axis
- `VehV(3,...)` — Vehicle velocity at track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehV`:

- Signal dimensions are [3x4].
- Signal contains 4 track velocities according to their axle and track locations.

$$
\text{VehV} = \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{z}_v \end{bmatrix} = \begin{bmatrix} \dot{x}_{v1,1} & \dot{x}_{v1,2} & \dot{x}_{v2,1} & \dot{x}_{v2,2} \\ \dot{y}_{v1,1} & \dot{y}_{v1,2} & \dot{y}_{v2,1} & \dot{y}_{v2,2} \\ \dot{z}_{v1,1} & \dot{z}_{v1,2} & \dot{z}_{v2,1} & \dot{z}_{v2,2} \end{bmatrix}
$$

Array Element	Axle	Track	Axis
VehV(1,1)	1	1	Vehicle-fixed x-axis
VehV(1,2)	1	2	
VehV(1,3)	2	1	

Array Element	Axle	Track	Axis
VehV(1,4)	2	2	
VehV(2,1)	1	1	Vehicle-fixed y-axis
VehV(2,2)	1	2	
VehV(2,3)	2	1	
VehV(2,4)	2	2	
VehV(3,1)	1	1	Vehicle-fixed z-axis
VehV(3,2)	1	2	
VehV(3,3)	2	1	
VehV(3,4)	2	2	

StrgAng — Steering angle, optional
array

Optional steering angle for each wheel, δ. Input array dimensions are 1 by the number of steered tracks.

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the StrgAng port, set **Steered axle enable by axle, StrgEnByAxl** to [1 0]. The input signal array dimensions are [1x2].
- The StrgAng signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer_{1,1}} & \delta_{steer_{1,2}} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:
  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Output

Info — Bus signal
bus

Bus signal containing block values. The signals are arrays that depend on the track location.

For example, here are the indices for a two-axle, two-track vehicle. The total number of tracks is four.

- 1D array signal (1-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

- 3D array signal (3-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2
(2,1)	1	1
(2,2)	1	2
(2,3)	2	1
(2,4)	2	2
(3,1)	1	1
(3,2)	1	2
(3,3)	2	1
(3,4)	2	2

Signal	Description	Array Signal	Variable	Units
Camber	Wheel angles according to the axle and track location.	1D	$\text{WhlAng}[1, \ldots] = \xi = [\xi_{a,t}]$	rad
Caster			$\text{WhlAng}[2, \ldots] = \eta = [\eta_{a,t}]$	
Toe			$\text{WhlAng}[3, \ldots] = \zeta = [\zeta_{a,t}]$	
Height	Suspension height	1D	H	m
Power	Suspension power dissipation	1D	P_{susp}	W
Energy	Suspension absorbed energy	1D	E_{susp}	J

Signal	Description	Array Signal	Variable	Units
VehF	Suspension forces applied to the vehicle	3D	For a two-axle, two tracks per axle vehicle: $\text{VehF} = F_v =$ $\begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$	N
VehM	Suspension moments applied to vehicle	3D	For a two-axle, two tracks per axle vehicle: $\text{VehM} = M_v =$ $\begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$	N·m
WhlF	Suspension force applied to wheel	3D	For a two-axle, two tracks per axle vehicle: $\text{WhlF} = F_w =$ $\begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$	N

Signal	Description	Array Signal	Variable	Units
WhlP	Track displacement	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlP} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} =$$ $$\begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{wtr_{1,1}} & z_{wtr_{1,2}} & z_{wtr_{2,1}} & z_{wtr_{2,2}} \end{bmatrix}$$	m
WhlV	Track velocity	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix}$$ $$=$$ $$\begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$	m/s
WhlAng	Wheel camber, caster, toe angles	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix}$$ $$= \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$	rad

VehF — Suspension force on vehicle

array

Longitudinal, lateral, and vertical suspension force at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N. Array dimensions are `3` by the total number of tracks on the vehicle.

- `VehF(1,...)` — Suspension force applied to vehicle along the vehicle-fixed *x*-axis (longitudinal)
- `VehF(2,...)` — Suspension force applied to vehicle along the vehicle-fixed *y*-axis (lateral)
- `VehF(3,...)` — Suspension force applied to vehicle along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehF`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension forces applied to the vehicle according to the axle and track locations.

$$\text{VehF} = F_v = \begin{bmatrix} F_{vx1,1} & F_{vx1,2} & F_{vx2,1} & F_{vx2,2} \\ F_{vy1,1} & F_{vy1,2} & F_{vy2,1} & F_{vy2,2} \\ F_{vz1,1} & F_{vz1,2} & F_{vz2,1} & F_{vz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
`VehF(1,1)`	1	1	Vehicle-fixed *x*-axis (longitudinal)
`VehF(1,2)`	1	2	
`VehF(1,3)`	2	1	
`VehF(1,4)`	2	2	
`VehF(2,1)`	1	1	Vehicle-fixed *y*-axis (lateral)
`VehF(2,2)`	1	2	
`VehF(2,3)`	2	1	
`VehF(2,4)`	2	2	
`VehF(3,1)`	1	1	Vehicle-fixed *z*-axis (vertical)
`VehF(3,2)`	1	2	
`VehF(3,3)`	2	1	
`VehF(3,4)`	2	2	

VehM — Suspension moment on vehicle
`array`

Longitudinal, lateral, and vertical suspension moment at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N·m. Array dimensions are `3` by the total number of tracks on the vehicle.

- `VehM(1,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *x*-axis (longitudinal)
- `VehM(2,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *y*-axis (lateral)
- `VehM(3,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehM`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension moments applied to vehicle according to the axle and track locations.

$$\text{VehM} = M_v = \begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
VehM(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehM(1,2)	1	2	
VehM(1,3)	2	1	
VehM(1,4)	2	2	
VehM(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehM(2,2)	1	2	
VehM(2,3)	2	1	
VehM(2,4)	2	2	
VehM(3,1)	1	1	Vehicle-fixed *z*-axis (vertical)
VehM(3,2)	1	2	
VehM(3,3)	2	1	
VehM(3,4)	2	2	

WhlF — Suspension force on wheel
array

Longitudinal, lateral, and vertical suspension forces at axle `a`, track `t`, applied to the wheel at the axle wheel carrier reference coordinate, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlF(1,...)` — Suspension force on wheel along the vehicle-fixed *x*-axis (longitudinal)
- `WhlF(2,...)` — Suspension force on wheel along the vehicle-fixed *y*-axis (lateral)
- `WhlF(3,...)` — Suspension force on wheel along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlF`:

- Signal dimensions are [3x4].
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlF} = F_w = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlF(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
WhlF(1,2)	1	2	
WhlF(1,3)	2	1	
WhlF(1,4)	2	2	

Array Element	Axle	Track	Force Axis
WhlF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlF(2,2)	1	2	
WhlF(2,3)	2	1	
WhlF(2,4)	2	2	
WhlF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

WhlV — Track velocity
array

Longitudinal, lateral, and vertical track velocity at axle `a`, track `t`, in m/s. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlV(1,...)` — Track velocity along the vehicle-fixed x-axis (longitudinal)
- `WhlV(2,...)` — Track velocity along the vehicle-fixed y-axis (lateral)
- `WhlV(3,...)` — Track velocity along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlV`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix} = \begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlV(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlV(1,2)	1	2	
WhlV(1,3)	2	1	
WhlV(1,4)	2	2	
WhlV(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlV(2,2)	1	2	
WhlV(2,3)	2	1	
WhlV(2,4)	2	2	
WhlV(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlV(3,2)	1	2	
WhlV(3,3)	2	1	
WhlV(3,4)	2	2	

WhlAng — Wheel camber, caster, toe angles
array

Camber, caster, and toe angles at axle `a`, track `t`, in rad. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlAng(1,...)` — Camber angle
- `WhlAng(2,...)` — Caster angle
- `WhlAng(3,...)` — Toe angle

For example, for a two-axle vehicle with two tracks per axle, the `WhlAng`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel angles according to the axle and track locations.

$$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$

Array Element	Axle	Track	Angle
`WhlAng(1,1)`	1	1	Camber
`WhlAng(1,2)`	1	2	
`WhlAng(1,3)`	2	1	
`WhlAng(1,4)`	2	2	
`WhlAng(2,1)`	1	1	Caster
`WhlAng(2,2)`	1	2	
`WhlAng(2,3)`	2	1	
`WhlAng(2,4)`	2	2	
`WhlAng(3,1)`	1	1	Toe
`WhlF(3,2)`	1	2	
`WhlF(3,3)`	2	1	
`WhlF(3,4)`	2	2	

Parameters

Axles

Number of axles, NumAxl — Number of axles
2 (default) | scalar

Number of axles, N_a, dimensionless.

Number of tracks by axle, NumTracksByAxl — Number of tracks per axle
[2 2] (default) | vector

Number of tracks per axle, Nt_a, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example, `[1,2]` represents one track on axle 1 and two tracks on axle 2.

Steered axle enable by axle, StrgEnByAxl — Boolean vector to enable axle steering
[1 0] (default) | vector

Boolean vector that enables axle steering, En_{steer}, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example:

- [1 0] — For a two-axle vehicle, enables axle 1 steering and disables axle 2 steering
- [1 1] — For a two-axle vehicle, enables axle 1 and axle 2 steering

Dependencies

Setting any element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

    - **Toe angle vs steering angle slope, ToeStrgSlp**
    - **Caster angle vs steering angle slope, CasterStrgSlp**
    - **Camber angle vs steering angle slope, CamberStrgSlp**
    - **Suspension height vs steering angle slope, StrgHgtSlp**

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the StrgAng port, set **Steered axle enable by axle, StrgEnByAxl** to [1 0]. The input signal array dimensions are [1x2].
- The StrgAng signal contains two steering angles according to their axle and track locations.

    $$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer_{1,1}} & \delta_{steer_{1,2}} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Anti-sway axle enable by axle, AntiSwayEnByAxl — Boolean vector to enable axle anti-sway
[0 0] (default) | vector

Boolean vector that enables axle anti-sway for axle a, dimensionless. For example, [1 0] enables axle 1 anti-sway and disables axle 2 anti-sway. Vector is 1 by the number of vehicle axles, N_a.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

Suspension

Mapped

Axle breakpoints, f_susp_axl_bp — Breakpoints
[1 2] (default) | 1-by-P array

Axle breakpoints, dimensionless.

Vertical axis suspension height breakpoints, f_susp_dz_bp — Breakpoints
1-by-M array

Vertical axis suspension height breakpoints, in m.

Vertical axis suspension height velocity breakpoints, f_susp_dzdot_bp — Breakpoints
1-by-N array

Vertical axis suspension height velocity breakpoints, in m/s.

Vertical axis suspension force and moment responses, f_susp_fmz — Output array
zeros(31,31,61,2,4) (default) | M-by-N-by-O-by-P-by-4 array

Array of output values as a function of:

- Vertical suspension height, *M*
- Vertical suspension height velocity, *N*
- Steering angle, *O*
- Axle, *P*
- 4 output types

  - 1 — Vertical force, in N·m
  - 2 — User-defined
  - 3 — Stored energy, in J
  - 4 — Absorbed power, in W

The array dimensions must match the breakpoint dimensions

Suspension geometry responses, f_susp_geom — Suspension geometry responses
zeros(31,61,2,3) (default) | M-by-O-by-P-by-3 array

Array of geometric suspension values as a function of:

- Vertical suspension height, *M*
- Steering angle, *O*
- Axle, *P*
- 3 output types

  - 1 — Camber angle, in rad
  - 2 — Caster angle, in rad
  - 3 — Toe angle, in rad

The array dimensions must match the breakpoint dimensions

Steering angle breakpoints, f_susp_strgdelta_bp — Steering angle breakpoints
1-by-0 array

Steering angle breakpoints, in rad.

Anti-Sway

Anti-sway arm radius, AntiSwayR — Anti-sway arm radius
0.2 (default) | scalar | vector

Anti-sway arm radius, r, in m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

Anti-sway arm neutral angle, AntiSwayNtrlAng — Anti-sway arm neutral angle
0.5236 (default) | scalar | vector

Anti-sway arm neutral angle, θ_{0a}, at nominal suspension height, in rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

Anti-sway torsion spring constant, AntiSwayTrsK — Anti-sway torsion spring constant
5.7296e+03 (default) | scalar | vector

Anti-sway bar torsion spring constant, k_a, in N·m/rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Anti-sway axle enable by axle, AntiSwayEnByAxl** vector to 1 creates these anti-sway parameters:

- **Anti-sway arm radius, AntiSwayR**
- **Anti-sway arm neutral angle, AntiSwayNtrlAng**
- **Anti-sway torsion spring constant, AntiSwayTrsK**

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[2] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[3] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary*. ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Independent Suspension - Double Wishbone | Independent Suspension - MacPherson

Introduced in R2018a

Solid Axle Suspension - Mapped

Mapped solid axle suspension
Library: Vehicle Dynamics Blockset / Suspension
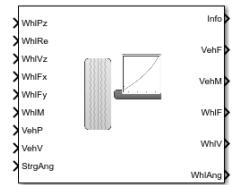
Description

The Solid Axle Suspension - Mapped block implements a mapped solid axle suspension for multiple axles with multiple tracks per axle.

The block models the suspension compliance, damping, and geometric effects as functions of the track positions and velocities, with axle-specific compliance and damping parameters. Using the track position and velocity, the block calculates the vertical track position and suspension forces on the vehicle and wheel. The block uses the Z-down (defined in SAE J670) and a solid axle coordinate system. The solid axle coordinate system, shown here, is aligned with the Z-down vehicle coordinate system, with the *x*-axis in the direction of forward vehicle motion.

For Each	You Can Specify
Axle	• Multiple tracks.
	• Suspension parameters.
Track	• Steering angles.

The block contains energy-storing spring elements and energy-dissipating damper elements. The block also stores energy via the axle roll angular acceleration and axle center of mass vertical and lateral acceleration.

This table summarizes the block parameter settings for a vehicle with:

- Two axles.
- Two tracks per axle.
- Steering angle input for both tracks on the front axle.

Parameter	Setting
Number of axles, NumAxl	2
Number of tracks by axle, NumTracksByAxl	[2 2]
Steered axle enable by axle, StrgEnByAxl	[1 0]

Suspension Compliance and Damping

The block uses a lookup table that relates the vertical damping and compliance to the suspension height, suspension height rate of change, and steering angle. You can calibrate the wheel force lookup table so that steering angle changes from the nominal center position generate a force that increases the vehicle height. Specifically, the block:

Uses	To Calculate
- Longitudinal and lateral displacement and velocity of the vehicle. - Longitudinal and lateral displacement and velocity of the track. - Vertical wheel forces applied to the vehicle.	- Suspension forces applied to the axle center. - Vertical displacements and velocities of the vehicle and track. - Longitudinal, lateral, and vertical suspension forces and moments applied to the vehicle. - Longitudinal, lateral, and vertical suspension forces and moments applied to the wheel.

To calculate the dynamics of the axle, the block implements these equations. The block neglects the effects of:

- Lateral and longitudinal translational velocity.
- Angular velocity about the vertical and lateral axes.

$$
\begin{bmatrix} \ddot{x}_a \\ \ddot{y}_a \\ \ddot{z}_a \end{bmatrix} = \frac{1}{M_a} \begin{bmatrix} F_{xa} \\ F_{ya} \\ F_{za} \end{bmatrix} + \begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{z}_a \end{bmatrix} \times \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \frac{1}{M_a} \begin{bmatrix} 0 \\ 0 \\ F_{za} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{z}_a \end{bmatrix} \times \begin{bmatrix} p \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} 0 \\ p\dot{z}_a \\ \frac{F_{za}}{M_a} + g \end{bmatrix}
$$

$$
\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \left[\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \right] \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1}
$$

$$
= \left[\begin{bmatrix} M_x \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} p \\ q \\ 0 \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ 0 \\ 0 \end{bmatrix} \right] \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1} = \begin{bmatrix} \frac{M_x}{I_{xx}} \\ 0 \\ 0 \end{bmatrix}
$$

For the forces and moments, the block uses lookup tables.

$$F_{wz_{a,t}} = f(z_{v_{a,t}} - z_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}})$$

$$M_{vz_{a,t}} = f(z_{v_{a,t}} - z_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}})$$

The suspension forces and moments applied to the vehicle are equal to the suspension forces and moments applied to the wheel.

$$F_{vx_{a,t}} = F_{wx_{a,t}}$$

$$F_{vy_{a,t}} = F_{wy_{a,t}}$$

$$F_{vz_{a,t}} = -F_{wz_{a,t}}$$

$$M_{vx_{a,t}} = M_{wx_{a,t}} + F_{wy_{a,t}}(Re_{wy_{a,t}} + H_{a,t})$$

$$M_{vy_{a,t}} = M_{wy_{a,t}} + F_{wx_{a,t}}\left(Re_{wx_{a,t}} + H_{a,t}\right)$$

$$M_{vz_{a,t}} = M_{wz_{a,t}}$$

The equations use these variables.

$F_{wz_{a,t}}, M_{wz_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed z-axis
$F_{wx_{a,t}}, M_{wx_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed x-axis
$F_{wy_{a,t}}, M_{wy_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed y-axis
$F_{vz_{a,t}}, M_{vz_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed z-axis
$F_{vx_{a,t}}, M_{vx_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed x-axis
$F_{vy_{a,t}}, M_{vy_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed y-axis
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle a
k_{z_a}	Vertical spring constant applied to tracks on axle a
m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
c_{z_a}	Vertical damping constant applied to tracks on axle a
$Re_{w_{a,t}}$	Effective wheel radius for axle a, track t
$F_{zhstop_{a,t}}$	Vertical hardstop force at axle a, track t, along the vehicle-fixed z-axis
$F_{zaswy_{a,t}}$	Vertical anti-sway force at axle a, track t, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{v_{a,t}}, \dot{x}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{w_{a,t}}, \dot{x}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$y_{v_{a,t}}, \dot{y}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed y-axis

$y_{w_{a,t}}, \dot{y}_{w_{a,t}}$ Track displacement and velocity at axle a, track t, along the vehicle-fixed y-axis

$H_{a,t}$ Suspension height at axle a, track t

$Re_{w_{a,t}}$ Effective wheel radius at axle a, track t

Camber, Caster, and Toe Angles

To calculate the camber, caster, and toe angles, the block uses a lookup table, $G_{alookup}$, that is a function of the suspension height and steering angle.

$$[\xi_{a,t} \; \eta_{a,t} \; \zeta_{a,t}] = G_{alookup}f(z_{w_{a,t}} - z_{v_{a,t}}, \delta_{steer_{a,t}})$$

The equations use these variables.

$\xi_{a,t}$ Camber angle of wheel on axle a, track t

$\eta_{a,t}$ Caster angle of wheel on axle a, track t

$\zeta_{a,t}$ Toe angle of wheel on axle a, track t

$\delta_{steer_{a,t}}$ Steering angle input for axle a, track t

$z_{v_{a,t}}$ Vehicle displacement at axle a, track t, along vehicle-fixed z-axis

$z_{w_{a,t}}$ Wheel displacement at axle a, track t, along vehicle-fixed z-axis

Steering Angles

Optionally, you can input steering angles for the tracks. To calculate the steering angles for the wheels, the block offsets the input steering angles as a function of the suspension height. For the calculation, the block uses a lookup table, $G_{alookup}$, that is a function of the suspension position and steering angle.

$$\delta_{whlsteer_{a,t}} = \delta_{steer_{a,t}} + G_{alookup}f(z_{w_{a,t}} - z_{v_{a,t}}, \delta_{steer_{a,t}})$$

The equation uses these variables.

$\delta_{whlsteer_{a,t}}$ Wheel steering angle for axle a, track t

$\delta_{steer_{a,t}}$ Steering angle input for axle a, track t

$z_{v_{a,t}}$ Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis

$z_{w_{a,t}}$ Wheel displacement at axle a, track t, along the vehicle-fixed z-axis

Power and Energy

The block calculates these suspension characteristics for each axle, a, track, t.

Calculation	Equation
Dissipated power, $P_{susp_{a,t}}$	$P_{susp_{a,t}} = F_{wzlookup_a}(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}})$
Absorbed energy, $E_{susp_{a,t}}$	$E_{susp_{a,t}} = F_{wzlookup_a}(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}})$
Suspension height, $H_{a,t}$	$H_{a,t} = -(z_{v_{a,t}} - z_{w_{a,t}} - \text{median}(f\_susp\_dz\_bp))$

Calculation	Equation
Distance from wheel carrier center to tire/road interface	$z_{wtr_{a,t}} = Re_{w_{a,t}} + H_{a,t}$

The equations use these variables.

m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle **a**
$\delta_{steer_{a,t}}$	Steering angle input for axle **a**, track **t**
$Re_{w_{a,t}}$	Axle **a**, track **t** effective wheel radius from wheel carrier center to tire/road interface
$f\_susp\_dz\_bp$	Vertical axis suspension height breakpoints
$z_{wtr_{a,t}}$	Distance from wheel carrier center to tire/road interface, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle **a**, track **t**, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle **a**, track **t**, along the vehicle-fixed z-axis

Ports

Input

WhlPz — Track z-axis displacement
`array`

Track displacement, z_w, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlPz`:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlPz} = z_w = \begin{bmatrix} z_{w1,1} & z_{w1,2} & z_{w2,1} & z_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
`WhlPz(1,1)`	1	1
`WhlPz(1,2)`	1	2
`WhlPz(1,3)`	2	1
`WhlPz(1,4)`	2	2

WhlRe — Wheel effective radius
`array`

Effective wheel radius, Re_w, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlRe`:

- Signal array dimensions are [1x4].

- Array dimensions are axle by track.

$$\text{WhlRe} = Re_w = \begin{bmatrix} Re_{w1,1} & Re_{w1,2} & Re_{w2,1} & Re_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlRe(1,1)	1	1
WhlRe(1,2)	1	2
WhlRe(1,3)	2	1
WhlRe(1,4)	2	2

WhlVz — Track z-axis velocity
array

Track velocity, $\dot{z}_w$, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlVz:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlVz} = \dot{z}_w = \begin{bmatrix} \dot{z}_{w1,1} & \dot{z}_{w1,2} & \dot{z}_{w2,1} & \dot{z}_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlVz(1,1)	1	1
WhlVz(1,2)	1	2
WhlVz(1,3)	2	1
WhlVz(1,4)	2	2

WhlFx — Longitudinal wheel force on vehicle
array

Longitudinal wheel force applied to vehicle, F_{wx}, along the vehicle-fixed x-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFx:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlFx} = F_{wx} = \begin{bmatrix} F_{wx1,1} & F_{wx1,2} & F_{wx2,1} & F_{wx2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFx(1,1)	1	1
WhlFx(1,2)	1	2
WhlFx(1,3)	2	1
WhlFx(1,4)	2	2

WhlFy — Lateral wheel force on vehicle
array

Lateral wheel force applied to vehicle, F_{wy}, along the vehicle-fixed y-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFy:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlFy} = F_{wy} = \begin{bmatrix} F_{wy1,1} & F_{wy1,2} & F_{wy2,1} & F_{wy2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFy(1,1)	1	1
WhlFy(1,2)	1	2
WhlFy(1.3)	2	1
WhlFy(1,4)	2	2

WhlM — Suspension moment on wheel
array

Longitudinal, lateral, and vertical suspension moments at axle a, track t, applied to the wheel at the axle wheel carrier reference coordinate, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlM(1,...) — Suspension moment applied to the wheel about the vehicle-fixed x-axis (longitudinal)
- WhlM(2,...) — Suspension moment applied to the wheel about the vehicle-fixed y-axis (lateral)
- WhlM(3,...) — Suspension moment applied to the wheel about the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlM:

- Signal dimensions are [3x4].
- Signal contains suspension moments applied to four wheels according to their axle and track locations.

$$\text{WhlM} = M_w = \begin{bmatrix} M_{wx1,1} & M_{wx1,2} & M_{wx2,1} & M_{wx2,2} \\ M_{wy1,1} & M_{wy1,2} & M_{wy2,1} & M_{wy2,2} \\ M_{wz1,1} & M_{wz1,2} & M_{wz2,1} & M_{wz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
WhlM(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlM(1,2)	1	2	
WhlM(1,3)	2	1	
WhlM(1,4)	2	2	
WhlM(2,1)	1	1	Vehicle-fixed y-axis (lateral)

Array Element	Axle	Track	Moment Axis
WhlM(2,2)	1	2	
WhlM(2,3)	2	1	
WhlM(2,4)	2	2	
WhlM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlM(3,2)	1	2	
WhlM(3,3)	2	1	
WhlM(3,4)	2	2	

VehP — Vehicle displacement
array

Vehicle displacement from axle `a`, track `t` along vehicle-fixed coordinate system, in m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehP(1,...)` — Vehicle displacement from track, x_v, along the vehicle-fixed x-axis
- `VehP(2,...)` — Vehicle displacement from track, y_v, along the vehicle-fixed y-axis
- `VehP(3,...)` — Vehicle displacement from track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehP`:

- Signal dimensions are `[3x4]`.
- Signal contains four track displacements according to their axle and track locations.

$$\text{VehP} = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} x_{v1,1} & x_{v1,2} & x_{v2,1} & x_{v2,2} \\ y_{v1,1} & y_{v1,2} & y_{v2,1} & y_{v2,2} \\ z_{v1,1} & z_{v1,2} & z_{v2,1} & z_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehP(1,1)	1	1	Vehicle-fixed x-axis
VehP(1,2)	1	2	
VehP(1,3)	2	1	
VehP(1,4)	2	2	
VehP(2,1)	1	1	Vehicle-fixed y-axis
VehP(2,2)	1	2	
VehP(2,3)	2	1	
VehP(2,4)	2	2	
VehP(3,1)	1	1	Vehicle-fixed z-axis
VehP(3,2)	1	2	
VehP(3,3)	2	1	
VehP(3,4)	2	2	

VehV — Vehicle velocity
array

Vehicle velocity at axle `a`, track `t` along vehicle-fixed coordinate system, in m. Input array dimensions are 3 by `a*t`.

- `VehV(1,...)` — Vehicle velocity at track, x_v, along the vehicle-fixed x-axis
- `VehV(2,...)` — Vehicle velocity at track, y_v, along the vehicle-fixed y-axis
- `VehV(3,...)` — Vehicle velocity at track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehV`:

- Signal dimensions are `[3x4]`.
- Signal contains 4 track velocities according to their axle and track locations.

$$\text{VehV} = \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{z}_v \end{bmatrix} = \begin{bmatrix} \dot{x}_{v1,1} & \dot{x}_{v1,2} & \dot{x}_{v2,1} & \dot{x}_{v2,2} \\ \dot{y}_{v1,1} & \dot{y}_{v1,2} & \dot{y}_{v2,1} & \dot{y}_{v2,2} \\ \dot{z}_{v1,1} & \dot{z}_{v1,2} & \dot{z}_{v2,1} & \dot{z}_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
`VehV(1,1)`	1	1	Vehicle-fixed x-axis
`VehV(1,2)`	1	2	
`VehV(1,3)`	2	1	
`VehV(1,4)`	2	2	
`VehV(2,1)`	1	1	Vehicle-fixed y-axis
`VehV(2,2)`	1	2	
`VehV(2,3)`	2	1	
`VehV(2,4)`	2	2	
`VehV(3,1)`	1	1	Vehicle-fixed z-axis
`VehV(3,2)`	1	2	
`VehV(3,3)`	2	1	
`VehV(3,4)`	2	2	

StrgAng — Steering angle, optional
array

Optional steering angle for each wheel, δ. Input array dimensions are 1 by the number of steered tracks.

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.
- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer1,1} & \delta_{steer1,2} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Output

Info — Bus signal
bus

Bus signal containing block values. The signals are arrays that depend on the track location.

For example, here are the indices for a two-axle, two-track vehicle. The total number of tracks is four.

- 1D array signal (1-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

- 3D array signal (3-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2
(2,1)	1	1
(2,2)	1	2
(2,3)	2	1
(2,4)	2	2
(3,1)	1	1
(3,2)	1	2

Array Element	Axle	Track
(3,3)	2	1
(3,4)	2	2

Signal	Description	Array Signal	Variable	Units
Camber	Wheel angles according to the axle.	1D	$\text{WhlAng}[1, \ldots] = \xi = [\xi_{a,t}]$	rad
Caster			$\text{WhlAng}[2, \ldots] = \eta = [\eta_{a,t}]$	
Toe			$\text{WhlAng}[3, \ldots] = \zeta = [\zeta_{a,t}]$	
Height	Suspension height	1D	H	m
Power	Suspension power dissipation	1D	P_{susp}	W
Energy	Suspension absorbed energy	1D	E_{susp}	J
VehF	Suspension forces applied to the vehicle	3D	For a two-axle, two tracks per axle vehicle: $\text{VehF} = F_v =$ $\begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$	N
VehM	Suspension moments applied to vehicle	3D	For a two-axle, two tracks per axle vehicle: $\text{VehM} = M_v =$ $\begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$	N·m

Signal	Description	Array Signal	Variable	Units
WhlF	Suspension force applied to wheel	3D	For a two-axle, two tracks per axle vehicle: $\text{WhlF} = F_w =$ $\begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$	N
WhlP	Track displacement	3D	For a two-axle, two tracks per axle vehicle: $\text{WhlP} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} =$ $\begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{wtr_{1,1}} & z_{wtr_{1,2}} & z_{wtr_{2,1}} & z_{wtr_{2,2}} \end{bmatrix}$	m
WhlV	Track velocity	3D	For a two-axle, two tracks per axle vehicle: $\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix}$ $=$ $\begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$	m/s

Signal	Description	Array Signal	Variable	Units
WhlAng	Wheel camber, caster, toe angles	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix}$$ $$= \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$	rad

VehF — Suspension force on vehicle
array

Longitudinal, lateral, and vertical suspension force at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehF(1,...)` — Suspension force applied to vehicle along the vehicle-fixed x-axis (longitudinal)
- `VehF(2,...)` — Suspension force applied to vehicle along the vehicle-fixed y-axis (lateral)
- `VehF(3,...)` — Suspension force applied to vehicle along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehF`:

- Signal dimensions are [3x4].
- Signal contains suspension forces applied to the vehicle according to the axle and track locations.

$$\text{VehF} = F_v = \begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
VehF(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
VehF(1,2)	1	2	
VehF(1,3)	2	1	
VehF(1,4)	2	2	
VehF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
VehF(2,2)	1	2	
VehF(2,3)	2	1	
VehF(2,4)	2	2	
VehF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
VehF(3,2)	1	2	
VehF(3,3)	2	1	
VehF(3,4)	2	2	

VehM — Suspension moment on vehicle
`array`

Longitudinal, lateral, and vertical suspension moment at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N·m. Array dimensions are `3` by the total number of tracks on the vehicle.

- `VehM(1,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *x*-axis (longitudinal)
- `VehM(2,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *y*-axis (lateral)
- `VehM(3,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehM`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension moments applied to vehicle according to the axle and track locations.

$$\text{VehM} = M_v = \begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
VehM(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehM(1,2)	1	2	
VehM(1,3)	2	1	
VehM(1,4)	2	2	
VehM(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehM(2,2)	1	2	
VehM(2,3)	2	1	
VehM(2,4)	2	2	
VehM(3,1)	1	1	Vehicle-fixed *z*-axis (vertical)
VehM(3,2)	1	2	
VehM(3,3)	2	1	
VehM(3,4)	2	2	

WhlF — Suspension force on wheel
`array`

Longitudinal, lateral, and vertical suspension forces at axle `a`, track `t`, applied to the wheel at the axle wheel carrier reference coordinate, in N. Array dimensions are `3` by the total number of tracks on the vehicle.

- `WhlF(1,...)` — Suspension force on wheel along the vehicle-fixed *x*-axis (longitudinal)
- `WhlF(2,...)` — Suspension force on wheel along the vehicle-fixed *y*-axis (lateral)
- `WhlF(3,...)` — Suspension force on wheel along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlF`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlF} = F_w = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlF(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlF(1,2)	1	2	
WhlF(1,3)	2	1	
WhlF(1,4)	2	2	
WhlF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlF(2,2)	1	2	
WhlF(2,3)	2	1	
WhlF(2,4)	2	2	
WhlF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

`WhlV` — Track velocity
`array`

Longitudinal, lateral, and vertical track velocity at axle `a`, track `t`, in m/s. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlV(1,...)` — Track velocity along the vehicle-fixed x-axis (longitudinal)
- `WhlV(2,...)` — Track velocity along the vehicle-fixed y-axis (lateral)
- `WhlV(3,...)` — Track velocity along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlV`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix} = \begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlV(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)

Array Element	Axle	Track	Force Axis
WhlV(1,2)	1	2	
WhlV(1,3)	2	1	
WhlV(1,4)	2	2	
WhlV(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
WhlV(2,2)	1	2	
WhlV(2,3)	2	1	
WhlV(2,4)	2	2	
WhlV(3,1)	1	1	Vehicle-fixed *z*-axis (vertical)
WhlV(3,2)	1	2	
WhlV(3,3)	2	1	
WhlV(3,4)	2	2	

WhlAng — Wheel camber, caster, toe angles
array

Camber, caster, and toe angles at axle `a`, track `t`, in rad. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlAng(1,...)` — Camber angle
- `WhlAng(2,...)` — Caster angle
- `WhlAng(3,...)` — Toe angle

For example, for a two-axle vehicle with two tracks per axle, the `WhlAng`:

- Signal dimensions are [3x4].
- Signal contains wheel angles according to the axle and track locations.

$$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$

Array Element	Axle	Track	Angle
WhlAng(1,1)	1	1	Camber
WhlAng(1,2)	1	2	
WhlAng(1,3)	2	1	
WhlAng(1,4)	2	2	
WhlAng(2,1)	1	1	Caster
WhlAng(2,2)	1	2	
WhlAng(2,3)	2	1	
WhlAng(2,4)	2	2	
WhlAng(3,1)	1	1	Toe
WhlF(3,2)	1	2	

Array Element	Axle	Track	Angle
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

Parameters

Axles

Number of axles, NumAxl — Number of axles
2 (default) | scalar

Number of axles, N_a, dimensionless.

Number of tracks by axle, NumTracksByAxl — Number of tracks per axle
[2 2] (default) | vector

Number of tracks per axle, Nt_a, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example, [1,2] represents one track on axle 1 and two tracks on axle 2.

Steered axle enable by axle, StrgEnByAxl — Boolean vector to enable axle steering
[1 0] (default) | vector

Boolean vector that enables axle steering, En_{steer}, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example:

- [1 0]—For a two-axle vehicle, enables axle 1 steering and disables axle 2 steering
- [1 1]—For a two-axle vehicle, enables axle 1 and axle 2 steering

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1:

- Creates input port StrgAng.
- Creates these parameters

    - **Toe angle vs steering angle slope, ToeStrgSlp**
    - **Caster angle vs steering angle slope, CasterStrgSlp**
    - **Camber angle vs steering angle slope, CamberStrgSlp**
    - **Suspension height vs steering angle slope, StrgHgtSlp**

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the StrgAng port, set **Steered axle enable by axle, StrgEnByAxl** to [1 0]. The input signal array dimensions are [1x2].
- The StrgAng signal contains two steering angles according to their axle and track locations.

    $$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer_{1,1}} & \delta_{steer_{1,2}} \end{bmatrix}$$

Array Element	Axle	Track
`StrgAng(1,1)`	1	1
`StrgAng(1,2)`	1	2

Axle and wheels lumped principal moments of inertia about longitudinal axis, AxlIxx — Inertia
300 (default) | vector

Axle and wheels lumped principal moments of inertia about longitudinal axis, AxleIxx a, in kg*m^2.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Axle and wheels lumped mass, AxlM — Mass
[2 2] (default) | vector

Axle and wheels lumped mass, a, in kg.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Track hardpoint coordinates relative to axle center, TrackCoords — Point
[0 0 0 0;-1 1 -1 1;0 0 0 0] (default) | array

Track hardpoint coordinates, Tc_t, along the solid axle x, y, and z-axes, in m.

For example, for a two-axle vehicle with two tracks per axle, the `TrackCoords` array:

*   Dimensions are [3x4].
*   Contains four track hardpoints coordinates according to their axle and track locations.

$$Tc_t = \begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{w_{1,1}} & z_{w_{1,2}} & z_{w_{2,1}} & z_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Axis
`TrackCoords(1, 1)`	1	1	Solid axle x-axis
`TrackCoords(1, 2)`	1	2	
`TrackCoords(1, 3)`	2	1	
`TrackCoords(1, 4)`	2	2	
`TrackCoords(2, 1)`	1	1	Solid axle y-axis
`TrackCoords(2, 2)`	1	2	

Array Element	Axle	Track	Axis
TrackCoords(2,3)	2	1	
TrackCoords(2,4)	2	2	
TrackCoords(3,1)	1	1	Solid axle z-axis
TrackCoords(3,2)	1	2	
TrackCoords(3,3)	2	1	
TrackCoords(3,4)	2	2	

Suspension hardpoint coordinates relative to axle center, SuspCoords — Point
[0 0 0 0;-1 1 -1 1;0 0 0 0] (default) | array

Suspension hardpoint coordinates, Sc_t, along the solid axle x-, y-, and z-axes, in m.

For example, for a two-axle vehicle with two tracks per axle, the SuspCoords array:

- Dimensions are [3x4].
- Contains four track hardpoints coordinates according to their axle and track locations.

$$Sc_t = \begin{bmatrix} x_{s1,1} & x_{s1,2} & x_{s2,1} & x_{s2,2} \\ y_{s1,1} & y_{s1,2} & y_{s2,1} & y_{s2,2} \\ z_{s1,1} & z_{s1,2} & z_{s2,1} & z_{s2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
SuspCoords(1,1)	1	1	Solid axle x-axis
SuspCoords(1,2)	1	2	
SuspCoords(1,3)	2	1	
SuspCoords(1,4)	2	2	
SuspCoords(2,1)	1	1	Solid axle y-axis
SuspCoords(2,2)	1	2	
SuspCoords(2,3)	2	1	
SuspCoords(2,4)	2	2	

Array Element	Axle	Track	Axis
SuspCoords(3,1)	1	1	Solid axle z-axis
SuspCoords(3,2)	1	2	
SuspCoords(3,3)	2	1	
SuspCoords(3,4)	2	2	

Wheel and axle interface compliance constant, KzWhlAxl — Spring rate
6437000 (default) | scalar

Wheel and axle interface compliance constant, K_z, in N/m.

Wheel and axle interface compliance preload, F0zWhlAxl — Spring rate
9810 (default) | scalar

Wheel and axle interface compliance preload, F_{0z}, in N.

Wheel and axle interface damping constant, CzWhlAxl — Damping
10000 (default) | scalar

Wheel and axle interface damping constant, C_z, in m.

Suspension

Mapped

Axle breakpoints, f_susp_axl_bp — Breakpoints
[1 2] (default) | 1-by-P array

Axle breakpoints, dimensionless.

Vertical axis suspension height breakpoints, f_susp_dz_bp — Breakpoints
1-by-M array

Vertical axis suspension height breakpoints, in m.

Vertical axis suspension height velocity breakpoints, f_susp_dzdot_bp — Breakpoints
1-by-N array

Vertical axis suspension height velocity breakpoints, in m/s.

Vertical axis suspension force and moment responses, f_susp_fmz — Output array
zeros(31,31,61,2,4) (default) | M-by-N-by-O-by-P-by-4 array

Array of output values as a function of:

- Vertical suspension height, M
- Vertical suspension height velocity, N

- Steering angle, *O*
- Axle, *P*
- 4 output types

  - 1 — Vertical force, in N·m
  - 2 — User-defined
  - 3 — Stored energy, in J
  - 4 — Absorbed power, in W

The array dimensions must match the breakpoint dimensions

Suspension geometry responses, f_susp_geom — Suspension geometry responses
zeros(31,61,2,3) (default) | M-by-0-by-P-by-3 array

Array of geometric suspension values as a function of:

- Vertical suspension height, *M*
- Steering angle, *O*
- Axle, *P*
- 3 output types

  - 1 — Camber angle, in rad
  - 2 — Caster angle, in rad
  - 3 — Toe angle, in rad

The array dimensions must match the breakpoint dimensions

Steering angle breakpoints, f_susp_strgdelta_bp — Steering angle breakpoints
1-by-0 array

Steering angle breakpoints, in rad.

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[2] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[3] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary*. ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Solid Axle Suspension | Solid Axle Suspension - Coil Spring | Solid Axle Suspension - Leaf Spring

Introduced in R2018a

Solid Axle Suspension

Solid axle suspension for multiple axles
Library: Vehicle Dynamics Blockset / Suspension
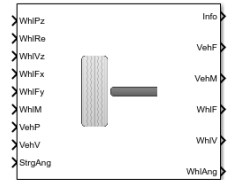
Description

The Solid Axle Suspension block implements a solid axle suspension for multiple axles with multiple tracks per axle.

The block models the suspension compliance, damping, and geometric effects as functions of the track positions and velocities, with axle-specific compliance and damping parameters. Using the track position and velocity, the block calculates the vertical track position and suspension forces on the vehicle and wheel. The block uses the Z-down (defined in SAE J670) and a solid axle coordinate system. The solid axle coordinate system, shown here, is aligned with the Z-down vehicle coordinate system, with the *x*-axis in the direction of forward vehicle motion.

For Each	You Can Specify
Axle	• Multiple tracks. • Suspension parameters.
Track	• Steering angles.

The block contains energy-storing spring elements and energy-dissipating damper elements. The block also stores energy via the axle roll angular acceleration and axle center of mass vertical and lateral acceleration.

This table summarizes the block parameter settings for a vehicle with:

- Two axles.

- Two tracks per axle.

- Steering angle input for both tracks on the front axle.

Parameter	Setting
Number of axles, NumAxl	2
Number of tracks by axle, NumTracksByAxl	[2 2]
Steered axle enable by axle, StrgEnByAxl	[1 0]

Suspension Compliance and Damping

The block uses a linear spring and damper to model the vertical dynamic effects of the suspension system on the vehicle and wheel. Specifically, the block:

Uses	To Calculate
- Longitudinal and lateral displacement and velocity of the vehicle. - Longitudinal and lateral displacement and velocity of the track. - Vertical wheel forces applied to the vehicle.	- Suspension forces applied to the axle center. - Vertical displacements and velocities of the vehicle and track. - Longitudinal, lateral and vertical suspension forces and moments applied to the vehicle. - Longitudinal, lateral and vertical suspension forces and moments applied to the wheel.

To calculate the dynamics of the axle, the block implements these equations. The block neglects the effects of:

- Lateral and longitudinal translational velocity.

- Angular velocity about the vertical and lateral axes.

$$
\begin{bmatrix} \ddot{x}_a \\ \ddot{y}_a \\ \ddot{z}_a \end{bmatrix} = \frac{1}{M_a} \begin{bmatrix} F_{xa} \\ F_{ya} \\ F_{za} \end{bmatrix} + \begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{z}_a \end{bmatrix} \times \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \frac{1}{M_a} \begin{bmatrix} 0 \\ 0 \\ F_{za} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{z}_a \end{bmatrix} \times \begin{bmatrix} p \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} 0 \\ p\dot{z}_a \\ \frac{F_{za}}{M_a} + g \end{bmatrix}
$$

$$
\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \left[\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \right] \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1}
$$

$$
= \left[\begin{bmatrix} M_x \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} p \\ q \\ 0 \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ 0 \\ 0 \end{bmatrix} \right] \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1} = \begin{bmatrix} \frac{M_x}{I_{xx}} \\ 0 \\ 0 \end{bmatrix}
$$

The net vertical force on the axle center of mass is the sum of the wheel and suspension forces acting on the axle.

$$F_{za} = \sum_{t=1}^{Nta} \left(F_{wz_{a,t}} + F_{z0_a} + k_{za}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{za}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right) \right)$$

The net moment about the roll axis of the solid axle suspension accounts for the hardpoint coordinates of the suspension and tracks.

$$M_x = \sum_{t=1}^{Nta} \left(F_{wz_{a,t}} y_{w_t} + \left(F_{z0_a} + k_{za}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{za}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right) \right) y_{s_t} \right.$$

$$\left. + M_{wx_{a,t}} \frac{I_{xx}}{I_{xx} + M_a y_{w_t}} \right)$$

Block parameters provide the track and suspension hardpoints coordinates.

$$Tc_t = \begin{bmatrix} x_{w_1} & x_{w_2} & \cdots \\ y_{w_1} & y_{w_2} & \cdots \\ z_{w_1} & z_{w_2} & \cdots \end{bmatrix}$$

$$Sc_t = \begin{bmatrix} x_{s_1} & x_{s_2} & \cdots \\ y_{s_1} & y_{s_2} & \cdots \\ z_{s_1} & z_{s_2} & \cdots \end{bmatrix}$$

The block uses Euler angles to transform the track and suspension displacements, velocities, and accelerations to the vehicle coordinate system.

To calculate the suspension forces applied to the vehicle, the block implements this equation.

$$F_{vz_{a,t}} = -\left(F_{z0_a} + k_{za}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{za}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right) + F_{zhstop_{a,t}} \right)$$

The suspension forces and moments applied to the vehicle are equal to the suspension forces and moments applied to the wheel.

$$F_{vx_{a,t}} = F_{wx_{a,t}}$$

$$F_{vy_{a,t}} = F_{wy_{a,t}}$$

$$F_{vz_{a,t}} = -F_{wz_{a,t}}$$

$$M_{vx_{a,t}} = M_{wx_{a,t}} + F_{wy_{a,t}}(Re_{wy_{a,t}} + H_{a,t})$$

$$M_{vy_{a,t}} = M_{wy_{a,t}} + F_{wx_{a,t}}\left(Re_{wx_{a,t}} + H_{a,t}\right)$$

$$M_{vz_{a,t}} = M_{wz_{a,t}}$$

To calculate the vertical force applied to the suspension at the track location, the block implements a stiff spring-damper.

$$F_{wz_{a,t}} = -Fwa_{z0} - kwa_z\left(z_{wa,t} - z_{sa,t}\right) - cwa_z\left(\dot{z}_{wa,t} - \dot{z}_{sa,t}\right)$$

The equations use these variables.

$F_{wz_{a,t}}, M_{wz_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed z-axis
$F_{wx_{a,t}}, M_{wx_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed x-axis
$F_{wy_{a,t}}, M_{wy_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed y-axis
$F_{vz_{a,t}}, M_{vz_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed z-axis
$F_{vx_{a,t}}, M_{vx_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed x-axis
$F_{vy_{a,t}}, M_{vy_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed y-axis
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle a
k_{z_a}	Vertical spring constant applied to tracks on axle a
m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
c_{z_a}	Vertical damping constant applied to tracks on axle a
$Re_{w_{a,t}}$	Effective wheel radius for axle a, track t
$F_{zhstop_{a,t}}$	Vertical hardstop force at axle a, track t, along the vehicle-fixed z-axis
$F_{zaswy_{a,t}}$	Vertical anti-sway force at axle a, track t, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{v_{a,t}}, \dot{x}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{w_{a,t}}, \dot{x}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$y_{v_{a,t}}, \dot{y}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$y_{w_{a,t}}, \dot{y}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$H_{a,t}$	Suspension height at axle a, track t
$Re_{w_{a,t}}$	Effective wheel radius at axle a, track t

Hardstop Forces

The hardstop feedback force, $F_{zhstop_{a,t}}$, that the block applies depends on whether the suspension is compressing or extending. The block applies the force:

- In compression, when the suspension is compressed more than the maximum distance specified by the **Suspension maximum height, Hmax** parameter.
- In extension, when the suspension extension is greater than maximum extension specified by the **Suspension maximum height, Hmax** parameter.

To calculate the force, the block uses a stiffness based on a hyperbolic tangent and exponential scaling.

Camber, Caster, and Toe Angles

To calculate the camber, caster, and toe angles, block uses linear functions of the suspension height and steering angle.

$$\xi_{a,t} = \xi_{0a} + m_{hcamber_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{cambersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\eta_{a,t} = \eta_{0a} + m_{hcaster_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{castersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\zeta_{a,t} = \zeta_{0a} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equations use these variables.

$\xi_{a,t}$	Camber angle of wheel on axle a, track t
$\eta_{a,t}$	Caster angle of wheel on axle a, track t
$\zeta_{a,t}$	Toe angle of wheel on axle a, track t
ξ_{0a}, η_{0a}, ζ_{0a}	Nominal suspension axle a camber, caster, and toe angles, respectively, at zero steering angle
$m_{hcamber_a}$, $m_{hcaster_a}$, m_{htoe_a}	Camber, caster, and toe angles, respectively, versus suspension height slope for axle a
$m_{cambersteer_a}$, $m_{castersteer_a}$, $m_{toesteer_a}$	Camber, caster, and toe angles, respectively, versus steering angle slope for axle a
m_{hsteer_a}	Steering angle versus vertical force slope for axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Steering Angles

Optionally, you can input steering angles for the tracks. To calculate the steering angles for the wheels, the block offsets the input steering angles with a linear function of the suspension height.

$$\delta_{whlsteer_{a,t}} = \delta_{steer_{a,t}} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equation uses these variables.

$m_{toesteer_a}$	Axle a toe angle versus steering angle slope
m_{hsteer_a}	Axle a steering angle versus vertical force slope
m_{htoe_a}	Axle a toe angle versus suspension height slope
$\delta_{whlsteer_{a,t}}$	Wheel steering angle for axle a, track t
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Power and Energy

The block calculates these suspension characteristics for each axle, a, track, t.

Calculation	Equation
Dissipated power, $P_{susp_{a,t}}$	$P_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Absorbed energy, $E_{susp_{a,t}}$	$E_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Suspension height, $H_{a,t}$	$H_{a,t} = -\left(z_{v_{a,t}} - z_{w_{a,t}} + \dfrac{F_{z0_a}}{k_{z_a}} + m_{hsteer_a}\left\|\delta_{steer_{a,t}}\right\|\right)$
Distance from wheel carrier center to tire/road interface	$z_{wtr_{a,t}} = Re_{w_{a,t}} + H_{a,t}$

The equations use these variables.

m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$Re_{w_{a,t}}$	Axle a, track t effective wheel radius from wheel carrier center to tire/road interface
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle a
$z_{wtr_{a,t}}$	Distance from wheel carrier center to tire/road interface, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis

Ports

Input

WhlPz — Track z-axis displacement
array

Track displacement, z_w, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlPz:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlPz} = z_w = \begin{bmatrix} z_{w_{1,1}} & z_{w_{1,2}} & z_{w_{2,1}} & z_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlPz(1,1)	1	1
WhlPz(1,2)	1	2
WhlPz(1,3)	2	1
WhlPz(1,4)	2	2

WhlRe — Wheel effective radius
array

Effective wheel radius, Re_w, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlRe:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlRe} = Re_w = \begin{bmatrix} Re_{w_{1,1}} & Re_{w_{1,2}} & Re_{w_{2,1}} & Re_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlRe(1,1)	1	1
WhlRe(1,2)	1	2
WhlRe(1,3)	2	1
WhlRe(1,4)	2	2

WhlVz — Track z-axis velocity
array

Track velocity, $\dot{z}_w$, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlVz:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlVz} = \dot{z}_w = \begin{bmatrix} \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlVz(1,1)	1	1
WhlVz(1,2)	1	2
WhlVz(1,3)	2	1
WhlVz(1,4)	2	2

WhlFx — Longitudinal wheel force on vehicle
array

Longitudinal wheel force applied to vehicle, F_{wx}, along the vehicle-fixed x-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFx:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlFx} = F_{wx} = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlFx(1,1)	1	1
WhlFx(1,2)	1	2
WhlFx(1,3)	2	1
WhlFx(1,4)	2	2

WhlFy — Lateral wheel force on vehicle
array

Lateral wheel force applied to vehicle, F_{wy}, along the vehicle-fixed y-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFy:

- Signal array dimensions are [1x4].

- Array dimensions are axle by track.

$$\text{WhlFy} = F_{wy} = \begin{bmatrix} F_{wy1,1} & F_{wy1,2} & F_{wy2,1} & F_{wy2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFy(1,1)	1	1
WhlFy(1,2)	1	2
WhlFy(1.3)	2	1
WhlFy(1,4)	2	2

WhlM — Suspension moment on wheel
array

Longitudinal, lateral, and vertical suspension moments at axle a, track t, applied to the wheel at the axle wheel carrier reference coordinate, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlM(1,...) — Suspension moment applied to the wheel about the vehicle-fixed x-axis (longitudinal)

- WhlM(2,...) — Suspension moment applied to the wheel about the vehicle-fixed y-axis (lateral)

- WhlM(3,...) — Suspension moment applied to the wheel about the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlM:

- Signal dimensions are [3x4].

- Signal contains suspension moments applied to four wheels according to their axle and track locations.

$$\text{WhlM} = M_w = \begin{bmatrix} M_{wx1,1} & M_{wx1,2} & M_{wx2,1} & M_{wx2,2} \\ M_{wy1,1} & M_{wy1,2} & M_{wy2,1} & M_{wy2,2} \\ M_{wz1,1} & M_{wz1,2} & M_{wz2,1} & M_{wz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
WhlM(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlM(1,2)	1	2	
WhlM(1,3)	2	1	
WhlM(1,4)	2	2	
WhlM(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlM(2,2)	1	2	
WhlM(2,3)	2	1	
WhlM(2,4)	2	2	
WhlM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlM(3,2)	1	2	
WhlM(3,3)	2	1	
WhlM(3,4)	2	2	

VehP — Vehicle displacement
array

Vehicle displacement from axle `a`, track `t` along vehicle-fixed coordinate system, in m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehP(1,...)` — Vehicle displacement from track, x_v, along the vehicle-fixed x-axis
- `VehP(2,...)` — Vehicle displacement from track, y_v, along the vehicle-fixed y-axis
- `VehP(3,...)` — Vehicle displacement from track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehP`:

- Signal dimensions are [3x4].
- Signal contains four track displacements according to their axle and track locations.

$$
\text{VehP} = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} x_{v1,1} & x_{v1,2} & x_{v2,1} & x_{v2,2} \\ y_{v1,1} & y_{v1,2} & y_{v2,1} & y_{v2,2} \\ z_{v1,1} & z_{v1,2} & z_{v2,1} & z_{v2,2} \end{bmatrix}
$$

Array Element	Axle	Track	Axis
VehP(1,1)	1	1	Vehicle-fixed x-axis
VehP(1,2)	1	2	
VehP(1,3)	2	1	
VehP(1,4)	2	2	
VehP(2,1)	1	1	Vehicle-fixed y-axis
VehP(2,2)	1	2	
VehP(2,3)	2	1	
VehP(2,4)	2	2	

Array Element	Axle	Track	Axis
VehP(3,1)	1	1	Vehicle-fixed z-axis
VehP(3,2)	1	2	
VehP(3,3)	2	1	
VehP(3,4)	2	2	

VehV — Vehicle velocity
array

Vehicle velocity at axle `a`, track `t` along vehicle-fixed coordinate system, in m. Input array dimensions are 3 by `a*t`.

- VehV(1,...) — Vehicle velocity at track, x_v, along the vehicle-fixed x-axis
- VehV(2,...) — Vehicle velocity at track, y_v, along the vehicle-fixed y-axis
- VehV(3,...) — Vehicle velocity at track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehV`:

- Signal dimensions are [3x4].
- Signal contains 4 track velocities according to their axle and track locations.

$$\text{VehV} = \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{z}_v \end{bmatrix} = \begin{bmatrix} \dot{x}_{v1,1} & \dot{x}_{v1,2} & \dot{x}_{v2,1} & \dot{x}_{v2,2} \\ \dot{y}_{v1,1} & \dot{y}_{v1,2} & \dot{y}_{v2,1} & \dot{y}_{v2,2} \\ \dot{z}_{v1,1} & \dot{z}_{v1,2} & \dot{z}_{v2,1} & \dot{z}_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehV(1,1)	1	1	Vehicle-fixed x-axis
VehV(1,2)	1	2	
VehV(1,3)	2	1	
VehV(1,4)	2	2	
VehV(2,1)	1	1	Vehicle-fixed y-axis
VehV(2,2)	1	2	
VehV(2,3)	2	1	
VehV(2,4)	2	2	
VehV(3,1)	1	1	Vehicle-fixed z-axis
VehV(3,2)	1	2	
VehV(3,3)	2	1	
VehV(3,4)	2	2	

StrgAng — Steering angle, optional
array

Optional steering angle for each wheel, δ. Input array dimensions are 1 by the number of steered tracks.

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.
- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer1,1} & \delta_{steer1,2} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Output

`Info` — Bus signal
bus

Bus signal containing block values. The signals are arrays that depend on the track location.

For example, here are the indices for a two-axle, two-track vehicle. The total number of tracks is four.

- 1D array signal (1-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

- 3D array signal (3-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

Array Element	Axle	Track
(2,1)	1	1
(2,2)	1	2
(2,3)	2	1
(2,4)	2	2
(3,1)	1	1
(3,2)	1	2
(3,3)	2	1
(3,4)	2	2

Signal	Description	Array Signal	Variable	Units
Camber	Wheel angles according to the axle.	1D	$\text{WhlAng}[1, \ldots] = \xi = \left[\xi_{a,t}\right]$	rad
Caster			$\text{WhlAng}[2, \ldots] = \eta = \left[\eta_{a,t}\right]$	
Toe			$\text{WhlAng}[3, \ldots] = \zeta = \left[\zeta_{a,t}\right]$	
Height	Suspension height	1D	H	m
Power	Suspension power dissipation	1D	P_{susp}	W
Energy	Suspension absorbed energy	1D	E_{susp}	J
VehF	Suspension forces applied to the vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\text{VehF} = F_v =$$ $$\begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$$	N

Signal	Description	Array Signal	Variable	Units
VehM	Suspension moments applied to vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\text{VehM} = M_v =$$ $$\begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$$	N·m
WhlF	Suspension force applied to wheel	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlF} = F_w =$$ $$\begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$	N
WhlP	Track displacement	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlP} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} =$$ $$\begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{wtr_{1,1}} & z_{wtr_{1,2}} & z_{wtr_{2,1}} & z_{wtr_{2,2}} \end{bmatrix}$$	m

Signal	Description	Array Signal	Variable	Units
WhlV	Track velocity	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix}$$ = $$\begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$	m/s
WhlAng	Wheel camber, caster, toe angles	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix}$$ $$= \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$	rad

VehF — Suspension force on vehicle
`array`

Longitudinal, lateral, and vertical suspension force at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehF(1,...)` — Suspension force applied to vehicle along the vehicle-fixed *x*-axis (longitudinal)
- `VehF(2,...)` — Suspension force applied to vehicle along the vehicle-fixed *y*-axis (lateral)
- `VehF(3,...)` — Suspension force applied to vehicle along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehF`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension forces applied to the vehicle according to the axle and track locations.

$$\text{VehF} = F_v = \begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
VehF(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehF(1,2)	1	2	
VehF(1,3)	2	1	
VehF(1,4)	2	2	
VehF(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehF(2,2)	1	2	
VehF(2,3)	2	1	
VehF(2,4)	2	2	
VehF(3,1)	1	1	Vehicle-fixed *z*-axis (vertical)
VehF(3,2)	1	2	
VehF(3,3)	2	1	
VehF(3,4)	2	2	

VehM — Suspension moment on vehicle
array

Longitudinal, lateral, and vertical suspension moment at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehM(1,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *x*-axis (longitudinal)
- `VehM(2,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *y*-axis (lateral)
- `VehM(3,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehM`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension moments applied to vehicle according to the axle and track locations.

$$
\text{VehM} = M_v = \begin{bmatrix} M_{vx1,1} & M_{vx1,2} & M_{vx2,1} & M_{vx2,2} \\ M_{vy1,1} & M_{vy1,2} & M_{vy2,1} & M_{vy2,2} \\ M_{vz1,1} & M_{vz1,2} & M_{vz2,1} & M_{vz2,2} \end{bmatrix}
$$

Array Element	Axle	Track	Moment Axis
VehM(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehM(1,2)	1	2	
VehM(1,3)	2	1	
VehM(1,4)	2	2	
VehM(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehM(2,2)	1	2	

Array Element	Axle	Track	Moment Axis
VehM(2,3)	2	1	
VehM(2,4)	2	2	
VehM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
VehM(3,2)	1	2	
VehM(3,3)	2	1	
VehM(3,4)	2	2	

WhlF — Suspension force on wheel
array

Longitudinal, lateral, and vertical suspension forces at axle `a`, track `t`, applied to the wheel at the axle wheel carrier reference coordinate, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlF(1,...)` — Suspension force on wheel along the vehicle-fixed x-axis (longitudinal)
- `WhlF(2,...)` — Suspension force on wheel along the vehicle-fixed y-axis (lateral)
- `WhlF(3,...)` — Suspension force on wheel along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlF`:

- Signal dimensions are [3x4].
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlF} = F_w = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlF(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlF(1,2)	1	2	
WhlF(1,3)	2	1	
WhlF(1,4)	2	2	
WhlF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlF(2,2)	1	2	
WhlF(2,3)	2	1	
WhlF(2,4)	2	2	
WhlF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

WhlV — Track velocity
array

Longitudinal, lateral, and vertical track velocity at axle `a`, track `t`, in m/s. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlV(1,...)` — Track velocity along the vehicle-fixed *x*-axis (longitudinal)
- `WhlV(2,...)` — Track velocity along the vehicle-fixed *y*-axis (lateral)
- `WhlV(3,...)` — Track velocity along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlV`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$
\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix} = \begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}
$$

Array Element	Axle	Track	Force Axis
`WhlV(1,1)`	1	1	Vehicle-fixed *x*-axis (longitudinal)
`WhlV(1,2)`	1	2	
`WhlV(1,3)`	2	1	
`WhlV(1,4)`	2	2	
`WhlV(2,1)`	1	1	Vehicle-fixed *y*-axis (lateral)
`WhlV(2,2)`	1	2	
`WhlV(2,3)`	2	1	
`WhlV(2,4)`	2	2	
`WhlV(3,1)`	1	1	Vehicle-fixed *z*-axis (vertical)
`WhlV(3,2)`	1	2	
`WhlV(3,3)`	2	1	
`WhlV(3,4)`	2	2	

WhlAng — Wheel camber, caster, toe angles
array

Camber, caster, and toe angles at axle `a`, track `t`, in rad. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlAng(1,...)` — Camber angle
- `WhlAng(2,...)` — Caster angle
- `WhlAng(3,...)` — Toe angle

For example, for a two-axle vehicle with two tracks per axle, the `WhlAng`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel angles according to the axle and track locations.

$$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$

Array Element	Axle	Track	Angle
`WhlAng(1,1)`	1	1	Camber
`WhlAng(1,2)`	1	2	
`WhlAng(1,3)`	2	1	
`WhlAng(1,4)`	2	2	
`WhlAng(2,1)`	1	1	Caster
`WhlAng(2,2)`	1	2	
`WhlAng(2,3)`	2	1	
`WhlAng(2,4)`	2	2	
`WhlAng(3,1)`	1	1	Toe
`WhlF(3,2)`	1	2	
`WhlF(3,3)`	2	1	
`WhlF(3,4)`	2	2	

Parameters

Axles

Number of axles, NumAxl — Number of axles
2 (default) | `scalar`

Number of axles, N_a, dimensionless.

Number of tracks by axle, NumTracksByAxl — Number of tracks per axle
[2 2] (default) | `vector`

Number of tracks per axle, Nt_a, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example, [1,2] represents one track on axle 1 and two tracks on axle 2.

Steered axle enable by axle, StrgEnByAxl — Boolean vector to enable axle steering
[1 0] (default) | `vector`

Boolean vector that enables axle steering, En_{steer}, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example:

- [1 0]—For a two-axle vehicle, enables axle 1 steering and disables axle 2 steering
- [1 1]—For a two-axle vehicle, enables axle 1 and axle 2 steering

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1:

- Creates input port `StrgAng`.

- Creates these parameters

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.
- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer1,1} & \delta_{steer1,2} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Axle and wheels lumped principal moments of inertia about longitudinal axis, AxlIxx — Inertia
300 (default) | `vector`

Axle and wheels lumped principal moments of inertia about longitudinal axis, AxleIxx a, in kg*m^2.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Axle and wheels lumped mass, AxlM — Mass
[2 2] (default) | `vector`

Axle and wheels lumped mass, a, in kg.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Track hardpoint coordinates relative to axle center, TrackCoords — Point
[0 0 0 0;-1 1 -1 1;0 0 0 0] (default) | `array`

Track hardpoint coordinates, Tc_t, along the solid axle x, y, and z-axes, in m.

For example, for a two-axle vehicle with two tracks per axle, the `TrackCoords` array:

- Dimensions are [3x4].
- Contains four track hardpoints coordinates according to their axle and track locations.

$$Tc_t = \begin{bmatrix} x_{w1,1} & x_{w1,2} & x_{w2,1} & x_{w2,2} \\ y_{w1,1} & y_{w1,2} & y_{w2,1} & y_{w2,2} \\ z_{w1,1} & z_{w1,2} & z_{w2,1} & z_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
TrackCoords(1, 1)	1	1	Solid axle x-axis
TrackCoords(1, 2)	1	2	
TrackCoords(1, 3)	2	1	
TrackCoords(1, 4)	2	2	
TrackCoords(2, 1)	1	1	Solid axle y-axis
TrackCoords(2, 2)	1	2	
TrackCoords(2, 3)	2	1	
TrackCoords(2, 4)	2	2	
TrackCoords(3, 1)	1	1	Solid axle z-axis
TrackCoords(3, 2)	1	2	
TrackCoords(3, 3)	2	1	
TrackCoords(3, 4)	2	2	

Suspension hardpoint coordinates relative to axle center, SuspCoords — Point
[0 0 0 0;-1 1 -1 1;0 0 0 0] (default) | array

Suspension hardpoint coordinates, Sc_t, along the solid axle x-, y-, and z-axes, in m.

For example, for a two-axle vehicle with two tracks per axle, the SuspCoords array:

- Dimensions are [3x4].

- Contains four track hardpoints coordinates according to their axle and track locations.

$$Sc_t = \begin{bmatrix} x_{s1,1} & x_{s1,2} & x_{s2,1} & x_{s2,2} \\ y_{s1,1} & y_{s1,2} & y_{s2,1} & y_{s2,2} \\ z_{s1,1} & z_{s1,2} & z_{s2,1} & z_{s2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
SuspCoords(1,1)	1	1	Solid axle x-axis
SuspCoords(1,2)	1	2	

Array Element	Axle	Track	Axis
SuspCoords(1,3)	2	1	
SuspCoords(1,4)	2	2	
SuspCoords(2,1)	1	1	Solid axle y-axis
SuspCoords(2,2)	1	2	
SuspCoords(2,3)	2	1	
SuspCoords(2,4)	2	2	
SuspCoords(3,1)	1	1	Solid axle z-axis
SuspCoords(3,2)	1	2	
SuspCoords(3,3)	2	1	
SuspCoords(3,4)	2	2	

Wheel and axle interface compliance constant, KzWhlAxl — Spring rate
6437000 (default) | scalar

Wheel and axle interface compliance constant, K_z, in N/m.

Wheel and axle interface compliance preload, F0zWhlAxl — Spring rate
9810 (default) | scalar

Wheel and axle interface compliance preload, F_{0z}, in N.

Wheel and axle interface damping constant, CzWhlAxl — Damping
10000 (default) | scalar

Wheel and axle interface damping constant, C_z, in m.

Suspension

Compliance and Damping - Passive

Suspension spring constant, Kz — Suspension spring constant
64370 (default) | scalar | vector

Linear vertical spring constant for independent suspension tracks on axle a, k_{z_a}, in N/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension spring preload, F0z — Suspension spring preload
9810 (default) | scalar | vector

Vertical preload spring force applied to the wheels on the axle at wheel carrier reference coordinates, F_{z0_a}, in N. Positive preload forces:

- Cause the vehicle to lift.
- Point along the negative vehicle-fixed z-axis.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension shock damping constant, Cz — Suspension shock damping constant
10000 (default) | scalar | vector

Linear vertical damping constant for independent suspension tracks on axle a, c_{z_a}, in Ns/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

To create this parameter, clear **Enable active damping**.

Suspension maximum height, Hmax — Height
0.5 (default) | scalar | vector

Maximum suspension extension or minimum suspension compression height, H_{max}, for axle a before the suspension reaches a hardstop, in m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Geometry

Toe angle at steering center, Toe — Toe angle
0.0349 (default) | scalar

Nominal suspension toe angle at zero steering angle, ζ_{0a}, in rad.

Roll steer vs suspension height slope, RollStrgSlp — Steer angle suspension slope
-0.2269 (default) | scalar | vector

Roll steer angle versus suspension height, m_{htoe_a}, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Toe angle vs steering angle slope, ToeStrgSlp — Toe angle steering slope
0.01 (default) | scalar | vector

Toe angle versus steering angle slope, $m_{toesteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

`Caster angle at steering center, Caster` — Caster angle at steering center
`0.0698` (default) | `scalar`

Nominal suspension caster angle at zero steering angle, η_{0a}, in rad.

`Caster angle vs suspension height slope, CasterHslp` — Caster angle versus suspension height slope
`-0.2269` (default) | `scalar` | `vector`

Caster angle versus suspension height, $m_{hcaster_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

`Caster angle vs steering angle slope, CasterStrgSlp` — Caster angle versus steering angle slope
`0.01` (default) | `scalar` | `vector`

Caster angle versus steering angle slope, $m_{castersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

`Camber angle at steering center, Camber` — Camber angle at steering center
`0.0698` (default) | `scalar`

Nominal suspension camber angle at zero steering angle, ξ_{0a}, in rad.

`Camber angle vs suspension height slope, CamberHslp` — Camber angle versus suspension height slope
`-0.2269` (default) | `scalar` | `vector`

Camber angle versus suspension height, $m_{hcamber_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Camber angle vs steering angle slope, CamberStrgSlp — Camber angle versus steering angle slope
`0.01` (default) | `scalar` | `vector`

Camber angle versus steering angle slope, $m_{cambersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Suspension height vs steering angle slope, StrgHgtSlp — Suspension height versus steering angle slope
`0.1432` (default) | `scalar` | `vector`

Steering angle to vertical force slope applied at suspension wheel carrier reference point, m_{hsteer_a}, in m/rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[2] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[3] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary.* ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Solid Axle Suspension - Coil Spring | Solid Axle Suspension - Leaf Spring | Solid Axle Suspension - Mapped

Introduced in R2018a

Solid Axle Suspension - Coil Spring

Solid axle suspension with coil spring
Library: Vehicle Dynamics Blockset / Suspension
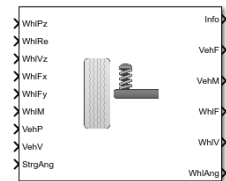
Description

The Solid Axle Suspension - Coil Spring block implements a solid axle suspension with a coil spring for multiple axles with multiple tracks per axle.

The block models the suspension compliance, damping, and geometric effects as functions of the track positions and velocities, with axle-specific compliance and damping parameters. Using the track position and velocity, the block calculates the vertical track position and suspension forces on the vehicle and wheel. The block uses the Z-down (defined in SAE J670) and a solid axle coordinate system. The solid axle coordinate system, shown here, is aligned with the Z-down vehicle coordinate system, with the *x*-axis in the direction of forward vehicle motion.

For Each	You Can Specify
Axle	• Multiple tracks.
	• Suspension parameters.
Track	• Steering angles.

The block contains energy-storing spring elements and energy-dissipating damper elements. The block also stores energy via the axle roll angular acceleration and axle center of mass vertical and lateral acceleration.

This table summarizes the block parameter settings for a vehicle with:

- Two axles.
- Two tracks per axle.
- Steering angle input for both tracks on the front axle.

Parameter	Setting
Number of axles, NumAxl	2
Number of tracks by axle, NumTracksByAxl	[2 2]
Steered axle enable by axle, StrgEnByAxl	[1 0]

Suspension Compliance and Damping

The block uses a linear spring and damper to model the vertical dynamic effects of the suspension system on the vehicle and wheel. Specifically, the block:

Uses	To Calculate
- Longitudinal and lateral displacement and velocity of the vehicle. - Longitudinal and lateral displacement and velocity of the track. - Vertical wheel forces applied to the vehicle.	- Suspension forces applied to the axle center. - Vertical displacements and velocities of the vehicle and track. - Longitudinal, lateral and vertical suspension forces and moments applied to the vehicle. - Longitudinal, lateral and vertical suspension forces and moments applied to the wheel.

To calculate the dynamics of the axle, the block implements these equations. The block neglects the effects of:

- Lateral and longitudinal translational velocity.
- Angular velocity about the vertical and lateral axes.

$$
\begin{bmatrix} \ddot{x}_a \\ \ddot{y}_a \\ \ddot{z}_a \end{bmatrix} = \frac{1}{M_a} \begin{bmatrix} F_{xa} \\ F_{ya} \\ F_{za} \end{bmatrix} + \begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{z}_a \end{bmatrix} \times \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \frac{1}{M_a} \begin{bmatrix} 0 \\ 0 \\ F_{za} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{z}_a \end{bmatrix} \times \begin{bmatrix} p \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} 0 \\ p\dot{z}_a \\ \frac{F_{za}}{M_a} + g \end{bmatrix}
$$

$$
\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \left[\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \right] \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1}
$$

$$
= \left[\begin{bmatrix} M_x \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} p \\ q \\ 0 \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ 0 \\ 0 \end{bmatrix} \right] \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1} = \begin{bmatrix} \frac{M_x}{I_{xx}} \\ 0 \\ 0 \end{bmatrix}
$$

The net vertical force on the axle center of mass is the sum of the wheel and suspension forces acting on the axle.

$$F_{za} = \sum_{t=1}^{Nta} \left(F_{wz_{a,t}} + F_{z0_a} + k_{z_a}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{z_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right) \right)$$

The net moment about the roll axis of the solid axle suspension accounts for the hardpoint coordinates of the suspension and tracks.

$$M_x = \sum_{t=1}^{Nta} \left(F_{wz_{a,t}} y_{w_t} + \left(F_{z0_a} + k_{z_a}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{z_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right)\right) y_{s_t} \right.$$
$$\left. + M_{wx_{a,t}}\frac{I_{xx}}{I_{xx} + M_a y_{w_t}} \right)$$

Block parameters provide the track and suspension hardpoints coordinates.

$$Tc_t = \begin{bmatrix} x_{w_1} & x_{w_2} & \cdots \\ y_{w_1} & y_{w_2} & \cdots \\ z_{w_1} & z_{w_2} & \cdots \end{bmatrix}$$

$$Sc_t = \begin{bmatrix} x_{s_1} & x_{s_2} & \cdots \\ y_{s_1} & y_{s_2} & \cdots \\ z_{s_1} & z_{s_2} & \cdots \end{bmatrix}$$

The block uses Euler angles to transform the track and suspension displacements, velocities, and accelerations to the vehicle coordinate system.

To calculate the suspension forces applied to the vehicle, the block implements this equation.

$$F_{vz_{a,t}} = -\left(F_{z0_a} + k_{z_a}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{z_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right) + F_{zhstop_{a,t}} \right)$$

The suspension forces and moments applied to the vehicle are equal to the suspension forces and moments applied to the wheel.

$$F_{vx_{a,t}} = F_{wx_{a,t}}$$

$$F_{vy_{a,t}} = F_{wy_{a,t}}$$

$$F_{vz_{a,t}} = -F_{wz_{a,t}}$$

$$M_{vx_{a,t}} = M_{wx_{a,t}} + F_{wy_{a,t}}(Re_{wy_{a,t}} + H_{a,t})$$

$$M_{vy_{a,t}} = M_{wy_{a,t}} + F_{wx_{a,t}}\left(Re_{wx_{a,t}} + H_{a,t}\right)$$

$$M_{vz_{a,t}} = M_{wz_{a,t}}$$

To calculate the vertical force applied to the suspension at the track location, the block implements a stiff spring-damper.

$$F_{wz_{a,t}} = -Fwa_{z0} - kwa_z\left(z_{wa,t} - z_{sa,t}\right) - cwa_z\left(\dot{z}_{wa,t} - \dot{z}_{sa,t}\right)$$

The equations use these variables.

$F_{wz_{a,t}}, M_{wz_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed z-axis
$F_{wx_{a,t}}, M_{wx_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed x-axis
$F_{wy_{a,t}}, M_{wy_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed y-axis
$F_{vz_{a,t}}, M_{vz_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed z-axis
$F_{vx_{a,t}}, M_{vx_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed x-axis
$F_{vy_{a,t}}, M_{vy_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed y-axis
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle a
k_{z_a}	Vertical spring constant applied to tracks on axle a
m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
c_{z_a}	Vertical damping constant applied to tracks on axle a
$Re_{w_{a,t}}$	Effective wheel radius for axle a, track t
$F_{zhstop_{a,t}}$	Vertical hardstop force at axle a, track t, along the vehicle-fixed z-axis
$F_{zaswy_{a,t}}$	Vertical anti-sway force at axle a, track t, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{v_{a,t}}, \dot{x}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{w_{a,t}}, \dot{x}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$y_{v_{a,t}}, \dot{y}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$y_{w_{a,t}}, \dot{y}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$H_{a,t}$	Suspension height at axle a, track t
$Re_{w_{a,t}}$	Effective wheel radius at axle a, track t

Hardstop Forces

The hardstop feedback force, $F_{zhstop_{a,t}}$, that the block applies depends on whether the suspension is compressing or extending. The block applies the force:

- In compression, when the suspension is compressed more than the maximum distance specified by the **Suspension maximum height, Hmax** parameter.
- In extension, when the suspension extension is greater than maximum extension specified by the **Suspension maximum height, Hmax** parameter.

To calculate the force, the block uses a stiffness based on a hyperbolic tangent and exponential scaling.

Camber, Caster, and Toe Angles

To calculate the camber, caster, and toe angles, block uses linear functions of the suspension height and steering angle.

$$\xi_{a,t} = \xi_{0a} + m_{hcamber_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{cambersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\eta_{a,t} = \eta_{0a} + m_{hcaster_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{castersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\zeta_{a,t} = \zeta_{0a} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equations use these variables.

$\xi_{a,t}$	Camber angle of wheel on axle a, track t
$\eta_{a,t}$	Caster angle of wheel on axle a, track t
$\zeta_{a,t}$	Toe angle of wheel on axle a, track t
ξ_{0a}, η_{0a}, ζ_{0a}	Nominal suspension axle a camber, caster, and toe angles, respectively, at zero steering angle
$m_{hcamber_a}$, $m_{hcaster_a}$, m_{htoe_a}	Camber, caster, and toe angles, respectively, versus suspension height slope for axle a
$m_{cambersteer_a}$, $m_{castersteer_a}$, $m_{toesteer_a}$	Camber, caster, and toe angles, respectively, versus steering angle slope for axle a
m_{hsteer_a}	Steering angle versus vertical force slope for axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Steering Angles

Optionally, you can input steering angles for the tracks. To calculate the steering angles for the wheels, the block offsets the input steering angles with a linear function of the suspension height.

$$\delta_{whlsteer_{a,t}} = \delta_{steer_{a,t}} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equation uses these variables.

$m_{toesteer_a}$	Axle a toe angle versus steering angle slope
m_{hsteer_a}	Axle a steering angle versus vertical force slope
m_{htoe_a}	Axle a toe angle versus suspension height slope
$\delta_{whlsteer_{a,t}}$	Wheel steering angle for axle a, track t
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Power and Energy

The block calculates these suspension characteristics for each axle, a, track, t.

Calculation	Equation
Dissipated power, $P_{susp_{a,t}}$	$P_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Absorbed energy, $E_{susp_{a,t}}$	$E_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Suspension height, $H_{a,t}$	$H_{a,t} = -\left(z_{v_{a,t}} - z_{w_{a,t}} + \dfrac{F_{z0_a}}{k_{z_a}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right)$
Distance from wheel carrier center to tire/road interface	$z_{wtr_{a,t}} = Re_{w_{a,t}} + H_{a,t}$

The equations use these variables.

m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$Re_{w_{a,t}}$	Axle a, track t effective wheel radius from wheel carrier center to tire/road interface
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle a
$z_{wtr_{a,t}}$	Distance from wheel carrier center to tire/road interface, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis

Ports

Input

WhlPz — Track z-axis displacement
array

Track displacement, z_w, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlPz:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlPz} = z_w = \begin{bmatrix} z_{w_{1,1}} & z_{w_{1,2}} & z_{w_{2,1}} & z_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlPz(1,1)	1	1
WhlPz(1,2)	1	2
WhlPz(1,3)	2	1
WhlPz(1,4)	2	2

WhlRe — Wheel effective radius
array

Effective wheel radius, Re_w, in m. Array dimensions are `1` by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlRe`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlRe} = Re_w = \begin{bmatrix} Re_{w_{1,1}} & Re_{w_{1,2}} & Re_{w_{2,1}} & Re_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
`WhlRe(1,1)`	1	1
`WhlRe(1,2)`	1	2
`WhlRe(1,3)`	2	1
`WhlRe(1,4)`	2	2

`WhlVz` — Track z-axis velocity
`array`

Track velocity, $\dot{z}_w$, along wheel-fixed z-axis, in m. Array dimensions are `1` by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlVz`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlVz} = \dot{z}_w = \begin{bmatrix} \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
`WhlVz(1,1)`	1	1
`WhlVz(1,2)`	1	2
`WhlVz(1,3)`	2	1
`WhlVz(1,4)`	2	2

`WhlFx` — Longitudinal wheel force on vehicle
`array`

Longitudinal wheel force applied to vehicle, F_{wx}, along the vehicle-fixed x-axis. Array dimensions are `1` by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlFx`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlFx} = F_{wx} = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlFx(1,1)	1	1
WhlFx(1,2)	1	2
WhlFx(1,3)	2	1
WhlFx(1,4)	2	2

WhlFy — Lateral wheel force on vehicle
array

Lateral wheel force applied to vehicle, F_{wy}, along the vehicle-fixed y-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFy:

- Signal array dimensions are [1x4].

- Array dimensions are axle by track.

$$\text{WhlFy} = F_{wy} = \begin{bmatrix} F_{wy1,1} & F_{wy1,2} & F_{wy2,1} & F_{wy2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFy(1,1)	1	1
WhlFy(1,2)	1	2
WhlFy(1.3)	2	1
WhlFy(1,4)	2	2

WhlM — Suspension moment on wheel
array

Longitudinal, lateral, and vertical suspension moments at axle a, track t, applied to the wheel at the axle wheel carrier reference coordinate, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlM(1,...) — Suspension moment applied to the wheel about the vehicle-fixed x-axis (longitudinal)

- WhlM(2,...) — Suspension moment applied to the wheel about the vehicle-fixed y-axis (lateral)

- WhlM(3,...) — Suspension moment applied to the wheel about the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlM:

- Signal dimensions are [3x4].

- Signal contains suspension moments applied to four wheels according to their axle and track locations.

$$\text{WhlM} = M_w = \begin{bmatrix} M_{wx1,1} & M_{wx1,2} & M_{wx2,1} & M_{wx2,2} \\ M_{wy1,1} & M_{wy1,2} & M_{wy2,1} & M_{wy2,2} \\ M_{wz1,1} & M_{wz1,2} & M_{wz2,1} & M_{wz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
WhlM(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlM(1,2)	1	2	
WhlM(1,3)	2	1	
WhlM(1,4)	2	2	
WhlM(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlM(2,2)	1	2	
WhlM(2,3)	2	1	
WhlM(2,4)	2	2	
WhlM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlM(3,2)	1	2	
WhlM(3,3)	2	1	
WhlM(3,4)	2	2	

VehP — Vehicle displacement
array

Vehicle displacement from axle `a`, track `t` along vehicle-fixed coordinate system, in m. Array dimensions are 3 by the total number of tracks on the vehicle.

- VehP(1,...) — Vehicle displacement from track, x_v, along the vehicle-fixed x-axis
- VehP(2,...) — Vehicle displacement from track, y_v, along the vehicle-fixed y-axis
- VehP(3,...) — Vehicle displacement from track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the VehP:

- Signal dimensions are [3x4].
- Signal contains four track displacements according to their axle and track locations.

$$\text{VehP} = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} x_{v1,1} & x_{v1,2} & x_{v2,1} & x_{v2,2} \\ y_{v1,1} & y_{v1,2} & y_{v2,1} & y_{v2,2} \\ z_{v1,1} & z_{v1,2} & z_{v2,1} & z_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehP(1,1)	1	1	Vehicle-fixed x-axis
VehP(1,2)	1	2	
VehP(1,3)	2	1	
VehP(1,4)	2	2	
VehP(2,1)	1	1	Vehicle-fixed y-axis
VehP(2,2)	1	2	
VehP(2,3)	2	1	
VehP(2,4)	2	2	

Array Element	Axle	Track	Axis
VehP(3,1)	1	1	Vehicle-fixed z-axis
VehP(3,2)	1	2	
VehP(3,3)	2	1	
VehP(3,4)	2	2	

VehV — Vehicle velocity
array

Vehicle velocity at axle `a`, track `t` along vehicle-fixed coordinate system, in m. Input array dimensions are 3 by `a*t`.

- VehV(1,...) — Vehicle velocity at track, x_v, along the vehicle-fixed x-axis
- VehV(2,...) — Vehicle velocity at track, y_v, along the vehicle-fixed y-axis
- VehV(3,...) — Vehicle velocity at track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehV`:

- Signal dimensions are [3x4].
- Signal contains 4 track velocities according to their axle and track locations.

$$\text{VehV} = \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{z}_v \end{bmatrix} = \begin{bmatrix} \dot{x}_{v1,1} & \dot{x}_{v1,2} & \dot{x}_{v2,1} & \dot{x}_{v2,2} \\ \dot{y}_{v1,1} & \dot{y}_{v1,2} & \dot{y}_{v2,1} & \dot{y}_{v2,2} \\ \dot{z}_{v1,1} & \dot{z}_{v1,2} & \dot{z}_{v2,1} & \dot{z}_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehV(1,1)	1	1	Vehicle-fixed x-axis
VehV(1,2)	1	2	
VehV(1,3)	2	1	
VehV(1,4)	2	2	
VehV(2,1)	1	1	Vehicle-fixed y-axis
VehV(2,2)	1	2	
VehV(2,3)	2	1	
VehV(2,4)	2	2	
VehV(3,1)	1	1	Vehicle-fixed z-axis
VehV(3,2)	1	2	
VehV(3,3)	2	1	
VehV(3,4)	2	2	

StrgAng — Steering angle, optional
array

Optional steering angle for each wheel, δ. Input array dimensions are 1 by the number of steered tracks.

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.

- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer1,1} & \delta_{steer1,2} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Output

Info — Bus signal
bus

Bus signal containing block values. The signals are arrays that depend on the track location.

For example, here are the indices for a two-axle, two-track vehicle. The total number of tracks is four.

- 1D array signal (1-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

- 3D array signal (3-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

Array Element	Axle	Track
(2,1)	1	1
(2,2)	1	2
(2,3)	2	1
(2,4)	2	2
(3,1)	1	1
(3,2)	1	2
(3,3)	2	1
(3,4)	2	2

Signal	Description	Array Signal	Variable	Units
Camber	Wheel angles according to the axle.	1D	$\text{WhlAng}[1, \ldots] = \xi = [\xi_{a,t}]$	rad
Caster			$\text{WhlAng}[2, \ldots] = \eta = [\eta_{a,t}]$	
Toe			$\text{WhlAng}[3, \ldots] = \zeta = [\zeta_{a,t}]$	
Height	Suspension height	1D	H	m
Power	Suspension power dissipation	1D	P_{susp}	W
Energy	Suspension absorbed energy	1D	E_{susp}	J
VehF	Suspension forces applied to the vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\text{VehF} = F_v =$$ $$\begin{bmatrix} F_{vx1,1} & F_{vx1,2} & F_{vx2,1} & F_{vx2,2} \\ F_{vy1,1} & F_{vy1,2} & F_{vy2,1} & F_{vy2,2} \\ F_{vz1,1} & F_{vz1,2} & F_{vz2,1} & F_{vz2,2} \end{bmatrix}$$	N

Signal	Description	Array Signal	Variable	Units
VehM	Suspension moments applied to vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\text{VehM} = M_v =$$ $$\begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$$	N·m
WhlF	Suspension force applied to wheel	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlF} = F_w =$$ $$\begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$	N
WhlP	Track displacement	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlP} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} =$$ $$\begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{wtr_{1,1}} & z_{wtr_{1,2}} & z_{wtr_{2,1}} & z_{wtr_{2,2}} \end{bmatrix}$$	m

Signal	Description	Array Signal	Variable	Units
WhlV	Track velocity	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix}$$ $$=$$ $$\begin{bmatrix} \dot{x}_{w1,1} & \dot{x}_{w1,2} & \dot{x}_{w2,1} & \dot{x}_{w2,2} \\ \dot{y}_{w1,1} & \dot{y}_{w1,2} & \dot{y}_{w2,1} & \dot{y}_{w2,2} \\ \dot{z}_{w1,1} & \dot{z}_{w1,2} & \dot{z}_{w2,1} & \dot{z}_{w2,2} \end{bmatrix}$$	m/s
WhlAng	Wheel camber, caster, toe angles	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix}$$ $$= \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$	rad

VehF — Suspension force on vehicle
`array`

Longitudinal, lateral, and vertical suspension force at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehF(1,...)` — Suspension force applied to vehicle along the vehicle-fixed *x*-axis (longitudinal)
- `VehF(2,...)` — Suspension force applied to vehicle along the vehicle-fixed *y*-axis (lateral)
- `VehF(3,...)` — Suspension force applied to vehicle along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehF`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension forces applied to the vehicle according to the axle and track locations.

$$\text{VehF} = F_v = \begin{bmatrix} F_{vx1,1} & F_{vx1,2} & F_{vx2,1} & F_{vx2,2} \\ F_{vy1,1} & F_{vy1,2} & F_{vy2,1} & F_{vy2,2} \\ F_{vz1,1} & F_{vz1,2} & F_{vz2,1} & F_{vz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
VehF(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehF(1,2)	1	2	
VehF(1,3)	2	1	
VehF(1,4)	2	2	
VehF(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehF(2,2)	1	2	
VehF(2,3)	2	1	
VehF(2,4)	2	2	
VehF(3,1)	1	1	Vehicle-fixed *z*-axis (vertical)
VehF(3,2)	1	2	
VehF(3,3)	2	1	
VehF(3,4)	2	2	

VehM — Suspension moment on vehicle
array

Longitudinal, lateral, and vertical suspension moment at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehM(1,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *x*-axis (longitudinal)
- `VehM(2,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *y*-axis (lateral)
- `VehM(3,...)` — Suspension moment applied to the vehicle about the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehM`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension moments applied to vehicle according to the axle and track locations.

$$\text{VehM} = M_v = \begin{bmatrix} M_{vx1,1} & M_{vx1,2} & M_{vx2,1} & M_{vx2,2} \\ M_{vy1,1} & M_{vy1,2} & M_{vy2,1} & M_{vy2,2} \\ M_{vz1,1} & M_{vz1,2} & M_{vz2,1} & M_{vz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
VehM(1,1)	1	1	Vehicle-fixed *x*-axis (longitudinal)
VehM(1,2)	1	2	
VehM(1,3)	2	1	
VehM(1,4)	2	2	
VehM(2,1)	1	1	Vehicle-fixed *y*-axis (lateral)
VehM(2,2)	1	2	

Array Element	Axle	Track	Moment Axis
VehM(2,3)	2	1	
VehM(2,4)	2	2	
VehM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
VehM(3,2)	1	2	
VehM(3,3)	2	1	
VehM(3,4)	2	2	

WhlF — Suspension force on wheel
array

Longitudinal, lateral, and vertical suspension forces at axle `a`, track `t`, applied to the wheel at the axle wheel carrier reference coordinate, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlF(1,...)` — Suspension force on wheel along the vehicle-fixed x-axis (longitudinal)
- `WhlF(2,...)` — Suspension force on wheel along the vehicle-fixed y-axis (lateral)
- `WhlF(3,...)` — Suspension force on wheel along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlF`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlF} = F_w = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlF(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlF(1,2)	1	2	
WhlF(1,3)	2	1	
WhlF(1,4)	2	2	
WhlF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlF(2,2)	1	2	
WhlF(2,3)	2	1	
WhlF(2,4)	2	2	
WhlF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

WhlV — Track velocity
array

Longitudinal, lateral, and vertical track velocity at axle `a`, track `t`, in m/s. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlV(1,...)` — Track velocity along the vehicle-fixed *x*-axis (longitudinal)
- `WhlV(2,...)` — Track velocity along the vehicle-fixed *y*-axis (lateral)
- `WhlV(3,...)` — Track velocity along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlV`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$
\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix} = \begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}
$$

Array Element	Axle	Track	Force Axis
`WhlV(1,1)`	1	1	Vehicle-fixed *x*-axis (longitudinal)
`WhlV(1,2)`	1	2	
`WhlV(1,3)`	2	1	
`WhlV(1,4)`	2	2	
`WhlV(2,1)`	1	1	Vehicle-fixed *y*-axis (lateral)
`WhlV(2,2)`	1	2	
`WhlV(2,3)`	2	1	
`WhlV(2,4)`	2	2	
`WhlV(3,1)`	1	1	Vehicle-fixed *z*-axis (vertical)
`WhlV(3,2)`	1	2	
`WhlV(3,3)`	2	1	
`WhlV(3,4)`	2	2	

`WhlAng` — Wheel camber, caster, toe angles
`array`

Camber, caster, and toe angles at axle `a`, track `t`, in rad. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlAng(1,...)` — Camber angle
- `WhlAng(2,...)` — Caster angle
- `WhlAng(3,...)` — Toe angle

For example, for a two-axle vehicle with two tracks per axle, the `WhlAng`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel angles according to the axle and track locations.

$$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$

Array Element	Axle	Track	Angle
`WhlAng(1,1)`	1	1	Camber
`WhlAng(1,2)`	1	2	
`WhlAng(1,3)`	2	1	
`WhlAng(1,4)`	2	2	
`WhlAng(2,1)`	1	1	Caster
`WhlAng(2,2)`	1	2	
`WhlAng(2,3)`	2	1	
`WhlAng(2,4)`	2	2	
`WhlAng(3,1)`	1	1	Toe
`WhlF(3,2)`	1	2	
`WhlF(3,3)`	2	1	
`WhlF(3,4)`	2	2	

Parameters

Axles

Number of axles, NumAxl — Number of axles
2 (default) | `scalar`

Number of axles, N_a, dimensionless.

Number of tracks by axle, NumTracksByAxl — Number of tracks per axle
[2 2] (default) | `vector`

Number of tracks per axle, Nt_a, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example, [1,2] represents one track on axle 1 and two tracks on axle 2.

Steered axle enable by axle, StrgEnByAxl — Boolean vector to enable axle steering
[1 0] (default) | `vector`

Boolean vector that enables axle steering, En_{steer}, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example:

- [1 0]—For a two-axle vehicle, enables axle 1 steering and disables axle 2 steering
- [1 1]—For a two-axle vehicle, enables axle 1 and axle 2 steering

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1:

- Creates input port `StrgAng`.

- Creates these parameters

    - **Toe angle vs steering angle slope, ToeStrgSlp**
    - **Caster angle vs steering angle slope, CasterStrgSlp**
    - **Camber angle vs steering angle slope, CamberStrgSlp**
    - **Suspension height vs steering angle slope, StrgHgtSlp**

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.

- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer_{1,1}} & \delta_{steer_{1,2}} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Axle and wheels lumped principal moments of inertia about longitudinal axis, AxlIxx — Inertia
300 (default) | vector

Axle and wheels lumped principal moments of inertia about longitudinal axis, AxleIxx a, in kg*m^2.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Axle and wheels lumped mass, AxlM — Mass
[2 2] (default) | vector

Axle and wheels lumped mass, a, in kg.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Track hardpoint coordinates relative to axle center, TrackCoords — Point
[0 0 0 0;-1 1 -1 1;0 0 0 0] (default) | array

Track hardpoint coordinates, Tc_t, along the solid axle x, y, and z-axes, in m.

For example, for a two-axle vehicle with two tracks per axle, the `TrackCoords` array:

- Dimensions are [3x4].

- Contains four track hardpoints coordinates according to their axle and track locations.

$$Tc_t = \begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{w_{1,1}} & z_{w_{1,2}} & z_{w_{2,1}} & z_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Axis
TrackCoords(1, 1)	1	1	Solid axle x-axis
TrackCoords(1, 2)	1	2	
TrackCoords(1, 3)	2	1	
TrackCoords(1, 4)	2	2	
TrackCoords(2, 1)	1	1	Solid axle y-axis
TrackCoords(2, 2)	1	2	
TrackCoords(2, 3)	2	1	
TrackCoords(2, 4)	2	2	
TrackCoords(3, 1)	1	1	Solid axle z-axis
TrackCoords(3, 2)	1	2	
TrackCoords(3, 3)	2	1	
TrackCoords(3, 4)	2	2	

Suspension hardpoint coordinates relative to axle center, SuspCoords — Point
`[0 0 0 0;-1 1 -1 1;0 0 0 0]` (default) | `array`

Suspension hardpoint coordinates, Sc_t, along the solid axle x-, y-, and z-axes, in m.

For example, for a two-axle vehicle with two tracks per axle, the `SuspCoords` array:

- Dimensions are `[3x4]`.
- Contains four track hardpoints coordinates according to their axle and track locations.

$$Sc_t = \begin{bmatrix} x_{s1,1} & x_{s1,2} & x_{s2,1} & x_{s2,2} \\ y_{s1,1} & y_{s1,2} & y_{s2,1} & y_{s2,2} \\ z_{s1,1} & z_{s1,2} & z_{s2,1} & z_{s2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
SuspCoords(1,1)	1	1	Solid axle x-axis
SuspCoords(1,2)	1	2	

Array Element	Axle	Track	Axis
SuspCoords(1,3)	2	1	
SuspCoords(1,4)	2	2	
SuspCoords(2,1)	1	1	Solid axle y-axis
SuspCoords(2,2)	1	2	
SuspCoords(2,3)	2	1	
SuspCoords(2,4)	2	2	
SuspCoords(3,1)	1	1	Solid axle z-axis
SuspCoords(3,2)	1	2	
SuspCoords(3,3)	2	1	
SuspCoords(3,4)	2	2	

Wheel and axle interface compliance constant, KzWhlAxl — Spring rate
6437000 (default) | scalar

Wheel and axle interface compliance constant, K_z, in N/m.

Wheel and axle interface compliance preload, F0zWhlAxl — Spring rate
9810 (default) | scalar

Wheel and axle interface compliance preload, F_{0z}, in N.

Wheel and axle interface damping constant, CzWhlAxl — Damping
10000 (default) | scalar

Wheel and axle interface damping constant, C_z, in m.

Suspension

Compliance and Damping - Passive

Suspension spring constant, Kz — Suspension spring constant
64370 (default) | scalar | vector

Linear vertical spring constant for independent suspension tracks on axle a, k_{z_a}, in N/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension spring preload, F0z — Suspension spring preload
9810 (default) | scalar | vector

Vertical preload spring force applied to the wheels on the axle at wheel carrier reference coordinates, F_{z0_a}, in N. Positive preload forces:

- Cause the vehicle to lift.
- Point along the negative vehicle-fixed z-axis.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension shock damping constant, Cz — Suspension shock damping constant
10000 (default) | scalar | vector

Linear vertical damping constant for independent suspension tracks on axle a, c_{z_a}, in Ns/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

To create this parameter, clear **Enable active damping**.

Suspension maximum height, Hmax — Height
0.5 (default) | scalar | vector

Maximum suspension extension or minimum suspension compression height, H_{max}, for axle a before the suspension reaches a hardstop, in m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Geometry

Toe angle at steering center, Toe — Toe angle
0.0349 (default) | scalar

Nominal suspension toe angle at zero steering angle, ζ_{0a}, in rad.

Roll steer vs suspension height slope, RollStrgSlp — Steer angle suspension slope
-0.2269 (default) | scalar | vector

Roll steer angle versus suspension height, m_{htoe_a}, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Toe angle vs steering angle slope, ToeStrgSlp — Toe angle steering slope
0.01 (default) | scalar | vector

Toe angle versus steering angle slope, $m_{toesteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Caster angle at steering center, Caster — Caster angle at steering center
`0.0698` (default) | `scalar`

Nominal suspension caster angle at zero steering angle, η_{0a}, in rad.

Caster angle vs suspension height slope, CasterHslp — Caster angle versus suspension height slope
`-0.2269` (default) | `scalar` | `vector`

Caster angle versus suspension height, $m_{hcaster_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Caster angle vs steering angle slope, CasterStrgSlp — Caster angle versus steering angle slope
`0.01` (default) | `scalar` | `vector`

Caster angle versus steering angle slope, $m_{castersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Camber angle at steering center, Camber — Camber angle at steering center
`0.0698` (default) | `scalar`

Nominal suspension camber angle at zero steering angle, ξ_{0a}, in rad.

Camber angle vs suspension height slope, CamberHslp — Camber angle versus suspension height slope
`-0.2269` (default) | `scalar` | `vector`

Camber angle versus suspension height, $m_{hcamber_a}$, in rad/m.

1-179

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

`Camber angle vs steering angle slope, CamberStrgSlp` — Camber angle versus steering angle slope
`0.01` (default) | `scalar` | `vector`

Camber angle versus steering angle slope, $m_{cambersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

`Suspension height vs steering angle slope, StrgHgtSlp` — Suspension height versus steering angle slope
`0.1432` (default) | `scalar` | `vector`

Steering angle to vertical force slope applied at suspension wheel carrier reference point, m_{hsteer_a}, in m/rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[2] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[3] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary.* ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Solid Axle Suspension | Solid Axle Suspension - Leaf Spring | Solid Axle Suspension - Mapped

Introduced in R2018a

Solid Axle Suspension - Leaf Spring

Solid axle suspension with leaf spring
Library: Vehicle Dynamics Blockset / Suspension
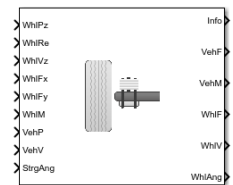
Description

The Solid Axle Suspension - Leaf Spring block implements a solid axle suspension with a coil spring for multiple axles with multiple tracks per axle.

The block models the suspension compliance, damping, and geometric effects as functions of the track positions and velocities, with axle-specific compliance and damping parameters. Using the track position and velocity, the block calculates the vertical track position and suspension forces on the vehicle and wheel. The block uses the Z-down (defined in SAE J670) and a solid axle coordinate system. The solid axle coordinate system, shown here, is aligned with the Z-down vehicle coordinate system, with the *x*-axis in the direction of forward vehicle motion.

For Each	You Can Specify
Axle	• Multiple tracks.
	• Suspension parameters.
Track	• Steering angles.

The block contains energy-storing spring elements and energy-dissipating damper elements. The block also stores energy via the axle roll angular acceleration and axle center of mass vertical and lateral acceleration.

This table summarizes the block parameter settings for a vehicle with:

- Two axles.

- Two tracks per axle.

- Steering angle input for both tracks on the front axle.

Parameter	Setting
Number of axles, NumAxl	2
Number of tracks by axle, NumTracksByAxl	[2 2]
Steered axle enable by axle, StrgEnByAxl	[1 0]

Suspension Compliance and Damping

The block uses a linear spring and damper to model the vertical dynamic effects of the suspension system on the vehicle and wheel. Specifically, the block:

Uses	To Calculate
- Longitudinal and lateral displacement and velocity of the vehicle. - Longitudinal and lateral displacement and velocity of the track. - Vertical wheel forces applied to the vehicle.	- Suspension forces applied to the axle center. - Vertical displacements and velocities of the vehicle and track. - Longitudinal, lateral and vertical suspension forces and moments applied to the vehicle. - Longitudinal, lateral and vertical suspension forces and moments applied to the wheel.

To calculate the dynamics of the axle, the block implements these equations. The block neglects the effects of:

- Lateral and longitudinal translational velocity.

- Angular velocity about the vertical and lateral axes.

$$
\begin{bmatrix} \ddot{x}_a \\ \ddot{y}_a \\ \ddot{z}_a \end{bmatrix} = \frac{1}{M_a} \begin{bmatrix} F_{xa} \\ F_{ya} \\ F_{za} \end{bmatrix} + \begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{z}_a \end{bmatrix} \times \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \frac{1}{M_a} \begin{bmatrix} 0 \\ 0 \\ F_{za} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{z}_a \end{bmatrix} \times \begin{bmatrix} p \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} 0 \\ p\dot{z}_a \\ \frac{F_{za}}{M_a} + g \end{bmatrix}
$$

$$
\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \left[\begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \right] \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1}
$$

$$
= \left[\begin{bmatrix} M_x \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} p \\ q \\ 0 \end{bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} p \\ 0 \\ 0 \end{bmatrix} \right] \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1} = \begin{bmatrix} \frac{M_x}{I_{xx}} \\ 0 \\ 0 \end{bmatrix}
$$

The net vertical force on the axle center of mass is the sum of the wheel and suspension forces acting on the axle.

$$F_{za} = \sum_{t=1}^{Nta} \left(F_{wz_{a,t}} + F_{z0_a} + k_{za}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{za}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right)\right)$$

The net moment about the roll axis of the solid axle suspension accounts for the hardpoint coordinates of the suspension and tracks.

$$M_x = \sum_{t=1}^{Nta} \left(F_{wz_{a,t}}y_{w_t} + \left(F_{z0_a} + k_{za}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{za}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right)\right)y_{s_t}$$

$$+ M_{wx_{a,t}}\frac{I_{xx}}{I_{xx} + M_a y_{w_t}}\right)$$

Block parameters provide the track and suspension hardpoints coordinates.

$$Tc_t = \begin{bmatrix} x_{w_1} & x_{w_2} & \cdots \\ y_{w_1} & y_{w_2} & \cdots \\ z_{w_1} & z_{w_2} & \cdots \end{bmatrix}$$

$$Sc_t = \begin{bmatrix} x_{s_1} & x_{s_2} & \cdots \\ y_{s_1} & y_{s_2} & \cdots \\ z_{s_1} & z_{s_2} & \cdots \end{bmatrix}$$

The block uses Euler angles to transform the track and suspension displacements, velocities, and accelerations to the vehicle coordinate system.

To calculate the suspension forces applied to the vehicle, the block implements this equation.

$$F_{vz_{a,t}} = -\left(F_{z0_a} + k_{za}\left(z_{v_{a,t}} - z_{s_{a,t}} + m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + c_{za}\left(\dot{z}_{v_{a,t}} - \dot{z}_{s_{a,t}}\right) + F_{zhstop_{a,t}}\right)$$

The suspension forces and moments applied to the vehicle are equal to the suspension forces and moments applied to the wheel.

$$F_{vx_{a,t}} = F_{wx_{a,t}}$$

$$F_{vy_{a,t}} = F_{wy_{a,t}}$$

$$F_{vz_{a,t}} = -F_{wz_{a,t}}$$

$$M_{vx_{a,t}} = M_{wx_{a,t}} + F_{wy_{a,t}}(Re_{wy_{a,t}} + H_{a,t})$$

$$M_{vy_{a,t}} = M_{wy_{a,t}} + F_{wx_{a,t}}\left(Re_{wx_{a,t}} + H_{a,t}\right)$$

$$M_{vz_{a,t}} = M_{wz_{a,t}}$$

To calculate the vertical force applied to the suspension at the track location, the block implements a stiff spring-damper.

$$F_{wz_{a,t}} = -Fwa_{z0} - kwa_z\left(z_{wa,t} - z_{sa,t}\right) - cwa_z\left(\dot{z}_{wa,t} - \dot{z}_{sa,t}\right)$$

The equations use these variables.

$F_{wz_{a,t}}, M_{wz_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed z-axis
$F_{wx_{a,t}}, M_{wx_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed x-axis
$F_{wy_{a,t}}, M_{wy_{a,t}}$	Suspension force and moment applied to the wheel on axle a, track t along wheel-fixed y-axis
$F_{vz_{a,t}}, M_{vz_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed z-axis
$F_{vx_{a,t}}, M_{vx_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed x-axis
$F_{vy_{a,t}}, M_{vy_{a,t}}$	Suspension force and moment applied to the vehicle on axle a, track t along wheel-fixed y-axis
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle a
k_{z_a}	Vertical spring constant applied to tracks on axle a
m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
c_{z_a}	Vertical damping constant applied to tracks on axle a
$Re_{w_{a,t}}$	Effective wheel radius for axle a, track t
$F_{zhstop_{a,t}}$	Vertical hardstop force at axle a, track t, along the vehicle-fixed z-axis
$F_{zaswy_{a,t}}$	Vertical anti-sway force at axle a, track t, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{v_{a,t}}, \dot{x}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$x_{w_{a,t}}, \dot{x}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed z-axis
$y_{v_{a,t}}, \dot{y}_{v_{a,t}}$	Vehicle displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$y_{w_{a,t}}, \dot{y}_{w_{a,t}}$	Track displacement and velocity at axle a, track t, along the vehicle-fixed y-axis
$H_{a,t}$	Suspension height at axle a, track t
$Re_{w_{a,t}}$	Effective wheel radius at axle a, track t

Hardstop Forces

The hardstop feedback force, $F_{zhstop_{a,t}}$, that the block applies depends on whether the suspension is compressing or extending. The block applies the force:

- In compression, when the suspension is compressed more than the maximum distance specified by the **Suspension maximum height, Hmax** parameter.
- In extension, when the suspension extension is greater than maximum extension specified by the **Suspension maximum height, Hmax** parameter.

To calculate the force, the block uses a stiffness based on a hyperbolic tangent and exponential scaling.

Camber, Caster, and Toe Angles

To calculate the camber, caster, and toe angles, block uses linear functions of the suspension height and steering angle.

$$\xi_{a,t} = \xi_{0a} + m_{hcamber_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{cambersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\eta_{a,t} = \eta_{0a} + m_{hcaster_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{castersteer_a}\left|\delta_{steer_{a,t}}\right|$$

$$\zeta_{a,t} = \zeta_{0a} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equations use these variables.

$\xi_{a,t}$	Camber angle of wheel on axle a, track t
$\eta_{a,t}$	Caster angle of wheel on axle a, track t
$\zeta_{a,t}$	Toe angle of wheel on axle a, track t
ξ_{0a}, η_{0a}, ζ_{0a}	Nominal suspension axle a camber, caster, and toe angles, respectively, at zero steering angle
$m_{hcamber_a}$, $m_{hcaster_a}$, m_{htoe_a}	Camber, caster, and toe angles, respectively, versus suspension height slope for axle a
$m_{cambersteer_a}$, $m_{castersteer_a}$, $m_{toesteer_a}$	Camber, caster, and toe angles, respectively, versus steering angle slope for axle a
m_{hsteer_a}	Steering angle versus vertical force slope for axle a
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Steering Angles

Optionally, you can input steering angles for the tracks. To calculate the steering angles for the wheels, the block offsets the input steering angles with a linear function of the suspension height.

$$\delta_{whlsteer_{a,t}} = \delta_{steer_{a,t}} + m_{htoe_a}\left(z_{w_{a,t}} - z_{v_{a,t}} - m_{hsteer_a}\left|\delta_{steer_{a,t}}\right|\right) + m_{toesteer_a}\left|\delta_{steer_{a,t}}\right|$$

The equation uses these variables.

$m_{toesteer_a}$	Axle a toe angle versus steering angle slope
m_{hsteer_a}	Axle a steering angle versus vertical force slope
m_{htoe_a}	Axle a toe angle versus suspension height slope
$\delta_{whlsteer_{a,t}}$	Wheel steering angle for axle a, track t
$\delta_{steer_{a,t}}$	Steering angle input for axle a, track t
$z_{v_{a,t}}$	Vehicle displacement at axle a, track t, along the vehicle-fixed z-axis
$z_{w_{a,t}}$	Track displacement at axle a, track t, along the vehicle-fixed z-axis

Power and Energy

The block calculates these suspension characteristics for each axle, a, track, t.

Calculation	Equation
Dissipated power, $P_{susp_{a,t}}$	$P_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Absorbed energy, $E_{susp_{a,t}}$	$E_{susp_{a,t}} = F_{wzlookup_a}\left(\dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \dot{z}_{v_{a,t}} - \dot{z}_{w_{a,t}}, \delta_{steer_{a,t}}\right)$
Suspension height, $H_{a,t}$	$H_{a,t} = -\left(z_{v_{a,t}} - z_{w_{a,t}} + \dfrac{F_{z0_a}}{k_{z_a}} + m_{hsteer_a}\left\|\delta_{steer_{a,t}}\right\|\right)$
Distance from wheel carrier center to tire/road interface	$z_{wtr_{a,t}} = Re_{w_{a,t}} + H_{a,t}$

The equations use these variables.

m_{hsteer_a}	Steering angle to vertical force slope applied at wheel carrier for tracks on axle `a`
$\delta_{steer_{a,t}}$	Steering angle input for axle `a`, track `t`
$Re_{w_{a,t}}$	Axle `a`, track `t` effective wheel radius from wheel carrier center to tire/road interface
F_{z0_a}	Vertical suspension spring preload force applied to the wheels on axle `a`
$z_{wtr_{a,t}}$	Distance from wheel carrier center to tire/road interface, along the vehicle-fixed z-axis
$z_{v_{a,t}}, \dot{z}_{v_{a,t}}$	Vehicle displacement and velocity at axle `a`, track `t`, along the vehicle-fixed z-axis
$z_{w_{a,t}}, \dot{z}_{w_{a,t}}$	Track displacement and velocity at axle `a`, track `t`, along the vehicle-fixed z-axis

Ports

Input

WhlPz — Track z-axis displacement
array

Track displacement, z_w, along wheel-fixed z-axis, in m. Array dimensions are `1` by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlPz`:

- Signal array dimensions are `[1x4]`.
- Array dimensions are axle by track.

$$\text{WhlPz} = z_w = \begin{bmatrix} z_{w_{1,1}} & z_{w_{1,2}} & z_{w_{2,1}} & z_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track
WhlPz(1,1)	1	1
WhlPz(1,2)	1	2
WhlPz(1,3)	2	1
WhlPz(1,4)	2	2

WhlRe — Wheel effective radius
array

Effective wheel radius, Re_w, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlRe`:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlRe} = Re_w = \begin{bmatrix} Re_{w1,1} & Re_{w1,2} & Re_{w2,1} & Re_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlRe(1,1)	1	1
WhlRe(1,2)	1	2
WhlRe(1,3)	2	1
WhlRe(1,4)	2	2

WhlVz — Track z-axis velocity
array

Track velocity, $\dot{z}_w$, along wheel-fixed z-axis, in m. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlVz`:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlVz} = \dot{z}_w = \begin{bmatrix} \dot{z}_{w1,1} & \dot{z}_{w1,2} & \dot{z}_{w2,1} & \dot{z}_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlVz(1,1)	1	1
WhlVz(1,2)	1	2
WhlVz(1,3)	2	1
WhlVz(1,4)	2	2

WhlFx — Longitudinal wheel force on vehicle
array

Longitudinal wheel force applied to vehicle, F_{wx}, along the vehicle-fixed x-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the `WhlFx`:

- Signal array dimensions are [1x4].
- Array dimensions are axle by track.

$$\text{WhlFx} = F_{wx} = \begin{bmatrix} F_{wx1,1} & F_{wx1,2} & F_{wx2,1} & F_{wx2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFx(1,1)	1	1
WhlFx(1,2)	1	2
WhlFx(1,3)	2	1
WhlFx(1,4)	2	2

WhlFy — Lateral wheel force on vehicle
array

Lateral wheel force applied to vehicle, F_{wy}, along the vehicle-fixed y-axis. Array dimensions are 1 by the total number of tracks on the vehicle.

For example, for a two-axle vehicle with two tracks per axle, the WhlFy:

- Signal array dimensions are [1x4].

- Array dimensions are axle by track.

$$\text{WhlFy} = F_{wy} = \begin{bmatrix} F_{wy1,1} & F_{wy1,2} & F_{wy2,1} & F_{wy2,2} \end{bmatrix}$$

Array Element	Axle	Track
WhlFy(1,1)	1	1
WhlFy(1,2)	1	2
WhlFy(1.3)	2	1
WhlFy(1,4)	2	2

WhlM — Suspension moment on wheel
array

Longitudinal, lateral, and vertical suspension moments at axle **a**, track **t**, applied to the wheel at the axle wheel carrier reference coordinate, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlM(1,...) — Suspension moment applied to the wheel about the vehicle-fixed x-axis (longitudinal)

- WhlM(2,...) — Suspension moment applied to the wheel about the vehicle-fixed y-axis (lateral)

- WhlM(3,...) — Suspension moment applied to the wheel about the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlM:

- Signal dimensions are [3x4].

- Signal contains suspension moments applied to four wheels according to their axle and track locations.

$$\text{WhlM} = M_w = \begin{bmatrix} M_{wx1,1} & M_{wx1,2} & M_{wx2,1} & M_{wx2,2} \\ M_{wy1,1} & M_{wy1,2} & M_{wy2,1} & M_{wy2,2} \\ M_{wz1,1} & M_{wz1,2} & M_{wz2,1} & M_{wz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
WhlM(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlM(1,2)	1	2	
WhlM(1,3)	2	1	
WhlM(1,4)	2	2	
WhlM(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlM(2,2)	1	2	
WhlM(2,3)	2	1	
WhlM(2,4)	2	2	
WhlM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlM(3,2)	1	2	
WhlM(3,3)	2	1	
WhlM(3,4)	2	2	

VehP — Vehicle displacement
array

Vehicle displacement from axle `a`, track `t` along vehicle-fixed coordinate system, in m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehP(1,...)` — Vehicle displacement from track, x_v, along the vehicle-fixed x-axis
- `VehP(2,...)` — Vehicle displacement from track, y_v, along the vehicle-fixed y-axis
- `VehP(3,...)` — Vehicle displacement from track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehP`:

- Signal dimensions are `[3x4]`.
- Signal contains four track displacements according to their axle and track locations.

$$\text{VehP} = \begin{bmatrix} x_v \\ y_v \\ z_v \end{bmatrix} = \begin{bmatrix} x_{v1,1} & x_{v1,2} & x_{v2,1} & x_{v2,2} \\ y_{v1,1} & y_{v1,2} & y_{v2,1} & y_{v2,2} \\ z_{v1,1} & z_{v1,2} & z_{v2,1} & z_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehP(1,1)	1	1	Vehicle-fixed x-axis
VehP(1,2)	1	2	
VehP(1,3)	2	1	
VehP(1,4)	2	2	
VehP(2,1)	1	1	Vehicle-fixed y-axis
VehP(2,2)	1	2	
VehP(2,3)	2	1	
VehP(2,4)	2	2	

Array Element	Axle	Track	Axis
VehP(3,1)	1	1	Vehicle-fixed z-axis
VehP(3,2)	1	2	
VehP(3,3)	2	1	
VehP(3,4)	2	2	

VehV — Vehicle velocity
array

Vehicle velocity at axle `a`, track `t` along vehicle-fixed coordinate system, in m. Input array dimensions are 3 by `a*t`.

- `VehV(1,...)` — Vehicle velocity at track, x_v, along the vehicle-fixed x-axis
- `VehV(2,...)` — Vehicle velocity at track, y_v, along the vehicle-fixed y-axis
- `VehV(3,...)` — Vehicle velocity at track, z_v, along the vehicle-fixed z-axis

For example, for a two-axle vehicle with two tracks per axle, the `VehV`:

- Signal dimensions are `[3x4]`.
- Signal contains 4 track velocities according to their axle and track locations.

$$\text{VehV} = \begin{bmatrix} \dot{x}_v \\ \dot{y}_v \\ \dot{z}_v \end{bmatrix} = \begin{bmatrix} \dot{x}_{v1,1} & \dot{x}_{v1,2} & \dot{x}_{v2,1} & \dot{x}_{v2,2} \\ \dot{y}_{v1,1} & \dot{y}_{v1,2} & \dot{y}_{v2,1} & \dot{y}_{v2,2} \\ \dot{z}_{v1,1} & \dot{z}_{v1,2} & \dot{z}_{v2,1} & \dot{z}_{v2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
VehV(1,1)	1	1	Vehicle-fixed x-axis
VehV(1,2)	1	2	
VehV(1,3)	2	1	
VehV(1,4)	2	2	
VehV(2,1)	1	1	Vehicle-fixed y-axis
VehV(2,2)	1	2	
VehV(2,3)	2	1	
VehV(2,4)	2	2	
VehV(3,1)	1	1	Vehicle-fixed z-axis
VehV(3,2)	1	2	
VehV(3,3)	2	1	
VehV(3,4)	2	2	

StrgAng — Steering angle, optional
array

Optional steering angle for each wheel, δ. Input array dimensions are 1 by the number of steered tracks.

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.
- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer1,1} & \delta_{steer1,2} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Output

`Info` — Bus signal
bus

Bus signal containing block values. The signals are arrays that depend on the track location.

For example, here are the indices for a two-axle, two-track vehicle. The total number of tracks is four.

- 1D array signal (1-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

- 3D array signal (3-by-4)

Array Element	Axle	Track
(1,1)	1	1
(1,2)	1	2
(1,3)	2	1
(1,4)	2	2

Array Element	Axle	Track
(2,1)	1	1
(2,2)	1	2
(2,3)	2	1
(2,4)	2	2
(3,1)	1	1
(3,2)	1	2
(3,3)	2	1
(3,4)	2	2

Signal	Description	Array Signal	Variable	Units
Camber	Wheel angles according to the axle.	1D	$\text{WhlAng}[1, \ldots] = \xi = [\xi_{a,t}]$	rad
Caster			$\text{WhlAng}[2, \ldots] = \eta = [\eta_{a,t}]$	
Toe			$\text{WhlAng}[3, \ldots] = \zeta = [\zeta_{a,t}]$	
Height	Suspension height	1D	H	m
Power	Suspension power dissipation	1D	P_{susp}	W
Energy	Suspension absorbed energy	1D	E_{susp}	J
VehF	Suspension forces applied to the vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\text{VehF} = F_v =$$ $$\begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$$	N

Signal	Description	Array Signal	Variable	Units
VehM	Suspension moments applied to vehicle	3D	For a two-axle, two tracks per axle vehicle: $$\text{VehM} = M_v =$$ $$\begin{bmatrix} M_{vx_{1,1}} & M_{vx_{1,2}} & M_{vx_{2,1}} & M_{vx_{2,2}} \\ M_{vy_{1,1}} & M_{vy_{1,2}} & M_{vy_{2,1}} & M_{vy_{2,2}} \\ M_{vz_{1,1}} & M_{vz_{1,2}} & M_{vz_{2,1}} & M_{vz_{2,2}} \end{bmatrix}$$	N·m
WhlF	Suspension force applied to wheel	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlF} = F_w =$$ $$\begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$	N
WhlP	Track displacement	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlP} = \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} =$$ $$\begin{bmatrix} x_{w_{1,1}} & x_{w_{1,2}} & x_{w_{2,1}} & x_{w_{2,2}} \\ y_{w_{1,1}} & y_{w_{1,2}} & y_{w_{2,1}} & y_{w_{2,2}} \\ z_{wtr_{1,1}} & z_{wtr_{1,2}} & z_{wtr_{2,1}} & z_{wtr_{2,2}} \end{bmatrix}$$	m

Signal	Description	Array Signal	Variable	Units
WhlV	Track velocity	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix}$$ $$= $$ $$\begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$	m/s
WhlAng	Wheel camber, caster, toe angles	3D	For a two-axle, two tracks per axle vehicle: $$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix}$$ $$= \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$	rad

VehF — Suspension force on vehicle
`array`

Longitudinal, lateral, and vertical suspension force at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehF(1,...)` — Suspension force applied to vehicle along the vehicle-fixed *x*-axis (longitudinal)
- `VehF(2,...)` — Suspension force applied to vehicle along the vehicle-fixed *y*-axis (lateral)
- `VehF(3,...)` — Suspension force applied to vehicle along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehF`:

- Signal dimensions are [3x4].
- Signal contains suspension forces applied to the vehicle according to the axle and track locations.

$$\text{VehF} = F_v = \begin{bmatrix} F_{vx_{1,1}} & F_{vx_{1,2}} & F_{vx_{2,1}} & F_{vx_{2,2}} \\ F_{vy_{1,1}} & F_{vy_{1,2}} & F_{vy_{2,1}} & F_{vy_{2,2}} \\ F_{vz_{1,1}} & F_{vz_{1,2}} & F_{vz_{2,1}} & F_{vz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
VehF(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
VehF(1,2)	1	2	
VehF(1,3)	2	1	
VehF(1,4)	2	2	
VehF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
VehF(2,2)	1	2	
VehF(2,3)	2	1	
VehF(2,4)	2	2	
VehF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
VehF(3,2)	1	2	
VehF(3,3)	2	1	
VehF(3,4)	2	2	

VehM — Suspension moment on vehicle
array

Longitudinal, lateral, and vertical suspension moment at axle `a`, track `t`, applied to the vehicle at the suspension connection point, in N·m. Array dimensions are 3 by the total number of tracks on the vehicle.

- `VehM(1,...)` — Suspension moment applied to the vehicle about the vehicle-fixed x-axis (longitudinal)
- `VehM(2,...)` — Suspension moment applied to the vehicle about the vehicle-fixed y-axis (lateral)
- `VehM(3,...)` — Suspension moment applied to the vehicle about the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `VehM`:

- Signal dimensions are `[3x4]`.
- Signal contains suspension moments applied to vehicle according to the axle and track locations.

$$\text{VehM} = M_v = \begin{bmatrix} M_{vx1,1} & M_{vx1,2} & M_{vx2,1} & M_{vx2,2} \\ M_{vy1,1} & M_{vy1,2} & M_{vy2,1} & M_{vy2,2} \\ M_{vz1,1} & M_{vz1,2} & M_{vz2,1} & M_{vz2,2} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
VehM(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
VehM(1,2)	1	2	
VehM(1,3)	2	1	
VehM(1,4)	2	2	
VehM(2,1)	1	1	Vehicle-fixed y-axis (lateral)
VehM(2,2)	1	2	

Array Element	Axle	Track	Moment Axis
VehM(2,3)	2	1	
VehM(2,4)	2	2	
VehM(3,1)	1	1	Vehicle-fixed z-axis (vertical)
VehM(3,2)	1	2	
VehM(3,3)	2	1	
VehM(3,4)	2	2	

WhlF — Suspension force on wheel
array

Longitudinal, lateral, and vertical suspension forces at axle a, track t, applied to the wheel at the axle wheel carrier reference coordinate, in N. Array dimensions are 3 by the total number of tracks on the vehicle.

- WhlF(1,...) — Suspension force on wheel along the vehicle-fixed x-axis (longitudinal)
- WhlF(2,...) — Suspension force on wheel along the vehicle-fixed y-axis (lateral)
- WhlF(3,...) — Suspension force on wheel along the vehicle-fixed z-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the WhlF:

- Signal dimensions are [3x4].
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlF} = F_w = \begin{bmatrix} F_{wx_{1,1}} & F_{wx_{1,2}} & F_{wx_{2,1}} & F_{wx_{2,2}} \\ F_{wy_{1,1}} & F_{wy_{1,2}} & F_{wy_{2,1}} & F_{wy_{2,2}} \\ F_{wz_{1,1}} & F_{wz_{1,2}} & F_{wz_{2,1}} & F_{wz_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
WhlF(1,1)	1	1	Vehicle-fixed x-axis (longitudinal)
WhlF(1,2)	1	2	
WhlF(1,3)	2	1	
WhlF(1,4)	2	2	
WhlF(2,1)	1	1	Vehicle-fixed y-axis (lateral)
WhlF(2,2)	1	2	
WhlF(2,3)	2	1	
WhlF(2,4)	2	2	
WhlF(3,1)	1	1	Vehicle-fixed z-axis (vertical)
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

WhlV — Track velocity
array

Longitudinal, lateral, and vertical track velocity at axle `a`, track `t`, in m/s. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlV(1,...)` — Track velocity along the vehicle-fixed *x*-axis (longitudinal)
- `WhlV(2,...)` — Track velocity along the vehicle-fixed *y*-axis (lateral)
- `WhlV(3,...)` — Track velocity along the vehicle-fixed *z*-axis (vertical)

For example, for a two-axle vehicle with two tracks per axle, the `WhlV`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel forces applied to the vehicle according to the axle and track locations.

$$\text{WhlV} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{z}_w \end{bmatrix} = \begin{bmatrix} \dot{x}_{w_{1,1}} & \dot{x}_{w_{1,2}} & \dot{x}_{w_{2,1}} & \dot{x}_{w_{2,2}} \\ \dot{y}_{w_{1,1}} & \dot{y}_{w_{1,2}} & \dot{y}_{w_{2,1}} & \dot{y}_{w_{2,2}} \\ \dot{z}_{w_{1,1}} & \dot{z}_{w_{1,2}} & \dot{z}_{w_{2,1}} & \dot{z}_{w_{2,2}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
`WhlV(1,1)`	1	1	Vehicle-fixed *x*-axis (longitudinal)
`WhlV(1,2)`	1	2	
`WhlV(1,3)`	2	1	
`WhlV(1,4)`	2	2	
`WhlV(2,1)`	1	1	Vehicle-fixed *y*-axis (lateral)
`WhlV(2,2)`	1	2	
`WhlV(2,3)`	2	1	
`WhlV(2,4)`	2	2	
`WhlV(3,1)`	1	1	Vehicle-fixed *z*-axis (vertical)
`WhlV(3,2)`	1	2	
`WhlV(3,3)`	2	1	
`WhlV(3,4)`	2	2	

WhlAng — Wheel camber, caster, toe angles
array

Camber, caster, and toe angles at axle `a`, track `t`, in rad. Array dimensions are 3 by the total number of tracks on the vehicle.

- `WhlAng(1,...)` — Camber angle
- `WhlAng(2,...)` — Caster angle
- `WhlAng(3,...)` — Toe angle

For example, for a two-axle vehicle with two tracks per axle, the `WhlAng`:

- Signal dimensions are `[3x4]`.
- Signal contains wheel angles according to the axle and track locations.

$$\text{WhlAng} = \begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} \xi_{1,1} & \xi_{1,2} & \xi_{2,1} & \xi_{2,2} \\ \eta_{1,1} & \eta_{1,2} & \eta_{2,1} & \eta_{2,2} \\ \zeta_{1,1} & \zeta_{1,2} & \zeta_{2,1} & \zeta_{2,2} \end{bmatrix}$$

Array Element	Axle	Track	Angle
WhlAng(1,1)	1	1	Camber
WhlAng(1,2)	1	2	
WhlAng(1,3)	2	1	
WhlAng(1,4)	2	2	
WhlAng(2,1)	1	1	Caster
WhlAng(2,2)	1	2	
WhlAng(2,3)	2	1	
WhlAng(2,4)	2	2	
WhlAng(3,1)	1	1	Toe
WhlF(3,2)	1	2	
WhlF(3,3)	2	1	
WhlF(3,4)	2	2	

Parameters

Axles

Number of axles, NumAxl — Number of axles
2 (default) | scalar

Number of axles, N_a, dimensionless.

Number of tracks by axle, NumTracksByAxl — Number of tracks per axle
[2 2] (default) | vector

Number of tracks per axle, Nt_a, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example, [1,2] represents one track on axle 1 and two tracks on axle 2.

Steered axle enable by axle, StrgEnByAxl — Boolean vector to enable axle steering
[1 0] (default) | vector

Boolean vector that enables axle steering, En_{steer}, dimensionless. Vector is 1 by the number of vehicle axles, N_a. For example:

- [1 0]—For a two-axle vehicle, enables axle 1 steering and disables axle 2 steering
- [1 1]—For a two-axle vehicle, enables axle 1 and axle 2 steering

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1:

- Creates input port StrgAng.

- Creates these parameters

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

For example, for a two-axle vehicle with two tracks per axle, you can input steering angles for both wheels on the first axle.

- To create the `StrgAng` port, set **Steered axle enable by axle, StrgEnByAxl** to `[1 0]`. The input signal array dimensions are `[1x2]`.
- The `StrgAng` signal contains two steering angles according to their axle and track locations.

$$\text{StrgAng} = \delta_{steer} = \begin{bmatrix} \delta_{steer1,1} & \delta_{steer1,2} \end{bmatrix}$$

Array Element	Axle	Track
StrgAng(1,1)	1	1
StrgAng(1,2)	1	2

Axle and wheels lumped principal moments of inertia about longitudinal axis, AxlIxx — Inertia
300 (default) | vector

Axle and wheels lumped principal moments of inertia about longitudinal axis, AxleIxx a, in kg*m^2.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Axle and wheels lumped mass, AxlM — Mass
[2 2] (default) | vector

Axle and wheels lumped mass, a, in kg.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Track hardpoint coordinates relative to axle center, TrackCoords — Point
[0 0 0 0;-1 1 -1 1;0 0 0 0] (default) | array

Track hardpoint coordinates, Tc_t, along the solid axle x, y, and z-axes, in m.

For example, for a two-axle vehicle with two tracks per axle, the `TrackCoords` array:

- Dimensions are `[3x4]`.
- Contains four track hardpoints coordinates according to their axle and track locations.

$$Tc_t = \begin{bmatrix} x_{w1,1} & x_{w1,2} & x_{w2,1} & x_{w2,2} \\ y_{w1,1} & y_{w1,2} & y_{w2,1} & y_{w2,2} \\ z_{w1,1} & z_{w1,2} & z_{w2,1} & z_{w2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
TrackCoords(1, 1)	1	1	Solid axle x-axis
TrackCoords(1, 2)	1	2	
TrackCoords(1, 3)	2	1	
TrackCoords(1, 4)	2	2	
TrackCoords(2, 1)	1	1	Solid axle y-axis
TrackCoords(2, 2)	1	2	
TrackCoords(2, 3)	2	1	
TrackCoords(2, 4)	2	2	
TrackCoords(3, 1)	1	1	Solid axle z-axis
TrackCoords(3, 2)	1	2	
TrackCoords(3, 3)	2	1	
TrackCoords(3, 4)	2	2	

Suspension hardpoint coordinates relative to axle center, SuspCoords — Point
[0 0 0 0;-1 1 -1 1;0 0 0 0] (default) | array

Suspension hardpoint coordinates, Sc_t, along the solid axle x-, y-, and z-axes, in m.

For example, for a two-axle vehicle with two tracks per axle, the SuspCoords array:

• Dimensions are [3x4].

• Contains four track hardpoints coordinates according to their axle and track locations.

$$Sc_t = \begin{bmatrix} x_{s1,1} & x_{s1,2} & x_{s2,1} & x_{s2,2} \\ y_{s1,1} & y_{s1,2} & y_{s2,1} & y_{s2,2} \\ z_{s1,1} & z_{s1,2} & z_{s2,1} & z_{s2,2} \end{bmatrix}$$

Array Element	Axle	Track	Axis
SuspCoords(1,1)	1	1	Solid axle x-axis
SuspCoords(1,2)	1	2	

Array Element	Axle	Track	Axis
SuspCoords(1,3)	2	1	
SuspCoords(1,4)	2	2	
SuspCoords(2,1)	1	1	Solid axle y-axis
SuspCoords(2,2)	1	2	
SuspCoords(2,3)	2	1	
SuspCoords(2,4)	2	2	
SuspCoords(3,1)	1	1	Solid axle z-axis
SuspCoords(3,2)	1	2	
SuspCoords(3,3)	2	1	
SuspCoords(3,4)	2	2	

Wheel and axle interface compliance constant, KzWhlAxl — Spring rate
6437000 (default) | scalar

Wheel and axle interface compliance constant, K_z, in N/m.

Wheel and axle interface compliance preload, F0zWhlAxl — Spring rate
9810 (default) | scalar

Wheel and axle interface compliance preload, F_{0z}, in N.

Wheel and axle interface damping constant, CzWhlAxl — Damping
10000 (default) | scalar

Wheel and axle interface damping constant, C_z, in m.

Suspension

Compliance and Damping - Passive

Suspension spring constant, Kz — Suspension spring constant
64370 (default) | scalar | vector

Linear vertical spring constant for independent suspension tracks on axle a, k_{z_a}, in N/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension spring preload, F0z — Suspension spring preload
9810 (default) | scalar | vector

Vertical preload spring force applied to the wheels on the axle at wheel carrier reference coordinates, F_{z0_a}, in N. Positive preload forces:

- Cause the vehicle to lift.
- Point along the negative vehicle-fixed z-axis.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Suspension shock damping constant, Cz — Suspension shock damping constant
10000 (default) | scalar | vector

Linear vertical damping constant for independent suspension tracks on axle a, c_{z_a}, in Ns/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

To create this parameter, clear **Enable active damping**.

Suspension maximum height, Hmax — Height
0.5 (default) | scalar | vector

Maximum suspension extension or minimum suspension compression height, H_{max}, for axle a before the suspension reaches a hardstop, in m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Geometry

Toe angle at steering center, Toe — Toe angle
0.0349 (default) | scalar

Nominal suspension toe angle at zero steering angle, ζ_{0a}, in rad.

Roll steer vs suspension height slope, RollStrgSlp — Steer angle suspension slope
-0.2269 (default) | scalar | vector

Roll steer angle versus suspension height, m_{htoe_a}, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Toe angle vs steering angle slope, ToeStrgSlp — Toe angle steering slope
0.01 (default) | scalar | vector

Toe angle versus steering angle slope, $m_{toesteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Caster angle at steering center, Caster — Caster angle at steering center
0.0698 (default) | scalar

Nominal suspension caster angle at zero steering angle, η_{0a}, in rad.

Caster angle vs suspension height slope, CasterHslp — Caster angle versus suspension height slope
-0.2269 (default) | scalar | vector

Caster angle versus suspension height, $m_{hcaster_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Caster angle vs steering angle slope, CasterStrgSlp — Caster angle versus steering angle slope
0.01 (default) | scalar | vector

Caster angle versus steering angle slope, $m_{castersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port `StrgAng`.
- Parameters:

  - **Toe angle vs steering angle slope, ToeStrgSlp**
  - **Caster angle vs steering angle slope, CasterStrgSlp**
  - **Camber angle vs steering angle slope, CamberStrgSlp**
  - **Suspension height vs steering angle slope, StrgHgtSlp**

Camber angle at steering center, Camber — Camber angle at steering center
0.0698 (default) | scalar

Nominal suspension camber angle at zero steering angle, ξ_{0a}, in rad.

Camber angle vs suspension height slope, CamberHslp — Camber angle versus suspension height slope
-0.2269 (default) | scalar | vector

Camber angle versus suspension height, $m_{hcamber_a}$, in rad/m.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Camber angle vs steering angle slope, CamberStrgSlp — Camber angle versus steering angle slope
0.01 (default) | scalar | vector

Camber angle versus steering angle slope, $m_{cambersteer_a}$, dimensionless.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

    - **Toe angle vs steering angle slope, ToeStrgSlp**
    - **Caster angle vs steering angle slope, CasterStrgSlp**
    - **Camber angle vs steering angle slope, CamberStrgSlp**
    - **Suspension height vs steering angle slope, StrgHgtSlp**

Suspension height vs steering angle slope, StrgHgtSlp — Suspension height versus steering angle slope
0.1432 (default) | scalar | vector

Steering angle to vertical force slope applied at suspension wheel carrier reference point, m_{hsteer_a}, in m/rad.

Vector is 1 by the number of vehicle axles, N_a. If you provide a scalar value, the block uses that value for all axles.

Dependencies

Setting an element of the **Steered axle enable by axle, StrgEnByAxl** vector to 1 creates:

- Input port StrgAng.
- Parameters:

    - **Toe angle vs steering angle slope, ToeStrgSlp**
    - **Caster angle vs steering angle slope, CasterStrgSlp**
    - **Camber angle vs steering angle slope, CamberStrgSlp**
    - **Suspension height vs steering angle slope, StrgHgtSlp**

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[2] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[3] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary.* ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Solid Axle Suspension | Solid Axle Suspension - Coil Spring | Solid Axle Suspension - Mapped

Introduced in R2018a

Drivetrain Blocks

Rotational Inertia

Ideal mechanical rotational inertia

Library: Powertrain Blockset / Drivetrain / Couplings
Vehicle Dynamics Blockset / Powertrain / Drivetrain / Couplings
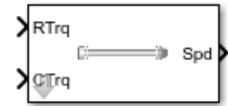
Description

The Rotational Inertia block implements an ideal mechanical rotational inertia.

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Variable	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrR	Mechanical power from base shaft	P_{TR}	$P_{TR} = T_R\omega$
		PwrC	Mechanical power from follower shaft	P_{TC}	$P_{TC} = T_C\omega$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrDampLoss	Power loss due to damping	P_d	$P_d = -b\lvert\omega\rvert^2$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	PwrStoredShft	Rate change of stored internal torsional energy	P_s	$P_s = \omega\dot{\omega}J$

The equations use these variables.

T_R	Input torque
T_C	Output torque

ω	Driveshaft angular velocity
J	Rotational inertia
b	Rotational viscous damping
P_d	Power loss due to damping
P_s	Rate change of stored internal torsional energy

Ports

Input

RTrq — Input torque
scalar

Applied input driveshaft torque, T_R, in N·m.

Dependencies

To enable this port, for **Port Configuration**, select `Simulink`.

CTrq — Output torque
scalar

Load driveshaft torque, T_C, in N·m.

Dependencies

To enable this port, for **Port Configuration**, select `Simulink`.

R — Angular velocity and torque
two-way connector port

Angular velocity in rad/s. Torque is in N·m.

Dependencies

To enable this port, for **Port Configuration**, select `Two-way connection`.

Inertia — Input
scalar

Rotational inertia, in kg·m^2.

Dependencies

To create the `Inertia` port, select **External inertia input**.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal			Description	Variable	Units
Trq	R		Applied input driveshaft torque	T_R	N·m
	C		Output driveshaft torque	T_C	N·m
	Damp		Damping torque	$T_d = b\omega$	N·m
Spd			Angular driveshaft speed	ω	rad/s
PwrInfo	PwrTrnsfrd	PwrR	Mechanical power from base shaft	P_{TR}	W
		PwrC	Mechanical power from follower shaft	P_{TC}	W
	PwrNotTrnsf rd	PwrDampLos s	Power loss due to damping	P_d	W
	PwrStored	PwrStoredS hft	Rate change of stored internal torsional energy	P_s	W

Dependencies

To enable this port, select **Output Info bus**.

Spd — Driveshaft speed
scalar

Angular driveshaft speed, ω, in rad/s.

Dependencies

To enable this port, for **Port Configuration**, select Simulink.

C — Angular velocity and torque
two-way connector port

Angular velocity in rad/s. Torque is in N·m.

Dependencies

To enable this port, for **Port Configuration**, select Two-way connection.

Parameters

Block Options

Port Configuration — Specify configuration
Simulink (default) | Two-way connection

Specify the port configuration.

Dependencies

Specifying Simulink creates these ports:

- RTrq
- CTrq
- Spd

Specifying Two-way connection creates these ports:

- R
- C

Output Info bus — Selection
off (default) | on

Select to create the Info output port.

External inertia input — Input rotational inertia
off (default) | on

Dependencies

To create the Inertia port, select **External inertia input**.

Parameters

Rotational inertia, J — Inertia
.01 (default) | scalar

Rotational inertia, in kg·m^2.

Dependencies

To enable this parameter, clear **Input rotational inertia**.

Torsional damping, b — Damping
.001 (default) | scalar

Torsional damping, in N·m· s/rad.

Initial velocity, omega_o — Angular
0 (default) | scalar

Initial angular velocity, in rad/s.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Split Torsional Compliance | Torsional Compliance

Introduced in R2017a

Split Torsional Compliance

Split torsional coupler

Library: Powertrain Blockset / Drivetrain / Couplings

Vehicle Dynamics Blockset / Powertrain / Drivetrain / Couplings
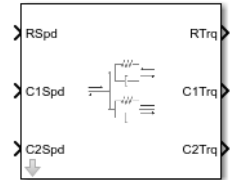
Description

The Split Torsional Compliance block implements parallel spring-damper coupling between shafts. You can specify the type of coupling by selecting one of the **Coupling Configuration** parameters:

- `Shaft split` — Single input shaft coupled to two output shafts
- `Shaft merge` — Two input shafts coupled to a single output shaft

In fuel economy and emissions studies, you can use the Split Torsional Compliance block to model mechanical rotational compliance between common driveline elements such as motors, planetary gears, and clutches. For example, use the `Shaft split` configuration to couple a motor and two planetary gear sets. Use the `Shaft merge` configuration to couple a dual clutch transmission to an output shaft.

Shaft Split

For the `Shaft split` configuration, the block implements this schematic and equations.
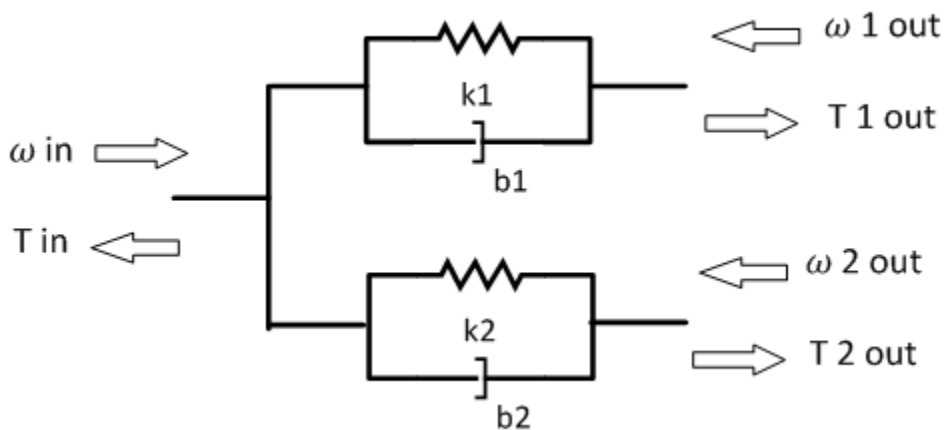
$$T_{in} = -(\omega_{in} - \omega_{1out})b_1 - (\omega_{in} - \omega_{2out})b_2 - \theta_1 k_1 - \theta_2 k_2$$

$$T_{1out} = (\omega_{in} - \omega_{1out})b_1 + \theta_1 k_1$$

$$T_{2out} = (\omega_{in} - \omega_{2out})b_2 + \theta_2 k_2$$

$$\dot{\theta}_1 = (\omega_{in} - \omega_{1out})$$

$$\dot{\theta}_2 = (\omega_{in} - \omega_{2out})$$

To account for frequency-dependent damping, both damping terms incorporate a low-pass filter.

The equations use these variables.

T_{in}	Resulting applied input reaction torque
ω_{in}	Input shaft rotational velocity
T_{1out}	Resulting applied torque to first output shaft
ω_{1out}	First output shaft rotational velocity
T_{2out}	Resulting applied torque to second output shaft
ω_{2out}	Second output shaft rotational velocity
θ_1, θ_2	First, second shaft rotation, respectively
b_1, b_2	First, second shaft viscous damping, respectively
k_1, k_2	First, second shaft torsional stiffness, respectively

Shaft Merge

For the `Shaft merge` configuration, the block implements this schematic and equations.
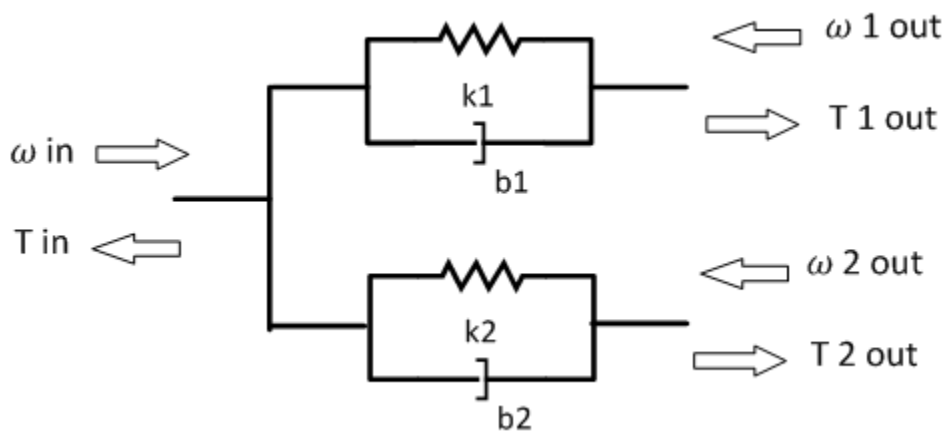
$$T_{out} = (-\omega_{out} + \omega_{1in})b_1 + (-\omega_{out} + \omega_{2in})b_2 + \theta_1 k_1 + \theta_2 k_2$$
$$T_{1out} = (\omega_{out} - \omega_{1in})b_1 - \theta_1 k_1$$
$$T_{2out} = (\omega_{out} - \omega_{2in})b_2 - \theta_2 k_2$$

$$\dot{\theta}_1 = (\omega_{1in} - \omega_{out})$$
$$\dot{\theta}_2 = (\omega_{2in} - \omega_{out})$$

To account for frequency-dependent damping, both damping terms incorporate a low-pass filter.

The equations use these variables.

T_{out}	Resulting applied output torque
ω_{out}	Output shaft rotational velocity
T_{1in}	Resulting reaction torque to first input shaft

ω_{1in}		First input shaft rotational velocity			
T_{2in}		Resulting reaction torque to second input shaft			
ω_{2in}		Second input shaft rotational velocity			
θ_1, θ_2		First, second shaft rotation, respectively			
b_1, b_2		First, second shaft viscous damping, respectively			
k_1, k_2		First, second shaft torsional stiffness, respectively			

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Variable	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrR	For the Shaft split configuration, mechanical power from input shaft	P_{TR}	$P_{TR} = -T_R\omega_R$
		PwrC1	For the Shaft split configuration, mechanical power from first output shaft	P_{TC1}	$P_{TC1} = -T_{C1}\omega_{C1}$
		PwrC2	For the Shaft split configuration, mechanical power from second output shaft	P_{TC2}	$P_{TC2} = -T_{C2}\omega_{C2}$
		PwrC	For the Shaft merge configuration, mechanical power from output shaft	P_{TC}	$P_{TC} = T_C\omega_C$
		PwrR1	For the Shaft merge configuration, mechanical power from first input shaft	P_{TR1}	$P_{TR1} = T_{R1}\omega_{R1}$

Bus Signal		PwrR2	Description	Variable	Equations
		PwrR2	For the `Shaft merge` configuration, mechanical power from second input shaft	P_{TR2}	$P_{TR2} = T_{R2}\omega_{R2}$
`PwrNotTrnsfrd` — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss		PwrDampLoss	Mechanical damping loss	P_d	$P_d = -\left(b_1\left|\dot{\theta}_1\right|^2 + b_2\left|\dot{\theta}_2\right|^2\right)$
`PwrStored` — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease		PwrStoredShft	Rate change in spring energy	P_s	$P_s = \left(k_1\theta_1\dot{\theta}_1 + k_2\theta_2\dot{\theta}_2\right)$

The equations use these variables.

T_R	Shaft R torque
T_C	Shaft C torque
ω_R	Shaft R angular velocity
ω_C	Shaft C angular velocity
θ	Coupled shaft rotation
k	Shaft torsional stiffness
b	Rotational viscous damping
P_t	Total mechanical power
P_d	Power loss due to damping
P_s	Rate change of stored spring energy

Ports

Input

RSpd — Input shaft speed
`scalar`

Input shaft rotational velocity, ω_{in}, in rad/s.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft split`

C1Spd — First output shaft speed
`scalar`

First output shaft rotational velocity, ω_{1out}, in rad/s.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft split`

C2Spd — Second output shaft speed
`scalar`

Second output shaft rotational velocity, ω_{2out}, in rad/s.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft split`

CSpd — Input speed
`scalar`

Output shaft rotational velocity, ω_{out}, in rad/s.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft merge`

R1Spd — First input shaft speed
`scalar`

First input shaft rotational velocity, ω_{1in}, in rad/s.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft merge`

R2Spd — Second input shaft speed
`scalar`

Second input shaft rotational velocity, ω_{2in}, in rad/s.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft merge`

R — Input shaft angular velocity and torque
two-way connector port

Input shaft angular velocity, ω_{in}, in rad/s and torque, T_{in}, in N·m.

Dependencies

To enable this port, select:

- **Port Configuration**>`Two-way connection`
- **Coupling Configuration**>`Shaft split`

R1 — First input shaft angular velocity and torque
two-way connector port

First input shaft angular velocity, ω_{1in}, in rad/s and torque, T_{1in}, in N·m.

Dependencies

To enable this port, select:

- **Port Configuration**>`Two-way connection`
- **Coupling Configuration**>`Shaft merge`

R2 — Second input shaft angular velocity and torque
two-way connector port

Second input shaft angular velocity, ω_{2in}, in rad/s and torque, T_{2in}, in N·m.

Dependencies

To enable this port, select:

- **Port Configuration**>`Two-way connection`
- **Coupling Configuration**>`Shaft merge`

Output

`Info` — Bus signal
bus

If you set **Coupling Configuration** to `Shaft split`, the Info bus contains these signals.

Signal		Description	Variable	Units
Trq	R	Input shaft torque	T_{in}	N·m
	C1	First output shaft torque	T_{1out}	N·m

Signal			Description	Variable	Units
	C2		Second output shaft torque	T_{2out}	N·m
	Damp	C1	First output shaft damping torque	$b_1\omega_{1out}$	N·m
		C2	Second output shaft damping torque	$b_2\omega_{2out}$	N·m
	Spring	C1	First output shaft spring torque	$k_1\theta_1$	N·m
		C2	Second output shaft spring torque	$k_2\theta_2$	N·m
Spd	R		Input shaft angular velocity	ω_{in}	rad/s
	C1		First output shaft angular velocity	ω_{1out}	rad/s
	C2		Second output shaft angular velocity	ω_{2out}	rad/s
	deltadot1		Difference in input and first output shaft angular velocity	$\dot{\theta}_1$	rad/s
	deltadot2		Difference in input and second output shaft angular velocity	$\dot{\theta}_2$	rad/s
PwrInfo	PwrTrnsfrd	PwrR	Mechanical power from input shaft	P_{TR}	W
		PwrC1	Mechanical power from first output shaft	P_{TC1}	W
		PwrC2	Mechanical power from second output shaft	P_{TC2}	W
	PwrNotTrnsfrd	PwrDampLoss	Mechanical damping loss	P_d	W
	PwrStored	PwrStoredShft	Rate change of stored internal torsional energy	P_s	W

If you set **Coupling Configuration** to Shaft merge, the Info bus contains these signals.

Signal			Description	Variable	Units
Trq	C		Output shaft torque	T_{out}	N·m
	R1		First input shaft torque	T_{1in}	N·m
	R2		Second input shaft torque	T_{2in}	N·m
	Damp	R1	First input shaft damping torque	$b_1\omega_{1in}$	N·m
		R2	Second in shaft damping torque	$b_2\omega_{2in}$	N·m
	Spring	R1	First input shaft spring torque	$k_1\theta_1$	N·m
		R2	Second in shaft spring torque	$k_2\theta_2$	N·m
Spd	C		Output shaft angular velocity	ω_{out}	rad/s
	R1		First input shaft angular velocity	ω_{1in}	rad/s

Signal			Description	Variable	Units
	R2		Second input shaft angular velocity	ω_{2in}	rad/s
	deltadot1		Difference in first input and output shaft angular velocity	$\dot{\theta}_1$	rad/s
	deltadot2		Difference in second input and output shaft angular velocity	$\dot{\theta}_2$	rad/s
PwrInfo	PwrTrnsfrd	PwrC	Mechanical power from output shaft	P_{TC}	W
		PwrR1	Mechanical power from first input shaft	P_{TR1}	W
		PwrR2	Mechanical power from second input shaft	P_{TR2}	W
	PwrNotTrnsfrd	PwrDampLoss	Mechanical damping loss	P_d	W
	PwrStored	PwrStoredShft	Rate change of stored internal torsional energy	P_s	W

Dependencies

To enable this port, select **Output Info bus**.

RTrq — Input shaft torque
scalar

Input shaft torque, T_{in}, in N·m.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to Simulink
- **Coupling Configuration** to Shaft split

C1Trq — First output shaft torque
scalar

First output shaft torque, T_{1out}, in N·m.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to Simulink
- **Coupling Configuration** to Shaft split

C2Trq — Second output shaft torque
scalar

Second output shaft torque, T_{2out}, in N·m.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft split`

CTrq — Output shaft torque
scalar

Output shaft torque, T_{out}, in N·m.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft merge`

R1Trq — First input shaft torque
scalar

First input shaft torque, T_{1in}, in N·m.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft merge`

R2Trq — Second input shaft torque
scalar

Second input shaft torque, T_{2in}, in N·m.

Dependencies

To enable this port, set both of these parameters:

- **Port Configuration** to `Simulink`
- **Coupling Configuration** to `Shaft merge`

C1 — First output shaft angular velocity and torque
two-way connector port

First output shaft angular velocity, ω_{1out}, in rad/s and torque, T_{1out}, in N·m.

Dependencies

To enable this port, select:

- **Port Configuration**>`Two-way connection`
- **Coupling Configuration**>`Shaft split`

C2 — Second output shaft angular velocity and torque
two-way connector port

Second output shaft angular velocity, ω_{2out}, in rad/s and torque, T_{2out}, in N·m.

Dependencies

To enable this port, select:

- **Port Configuration**>Two-way connection
- **Coupling Configuration**>Shaft split

C — Output shaft angular velocity and torque
two-way connector port

Output shaft angular velocity, ω_{out}, in rad/s and torque, T_{out}, in N·m.

Dependencies

To enable this port, select:

- **Port Configuration**>Two-way connection
- **Coupling Configuration**>Shaft merge

Parameters

Block Options

Port Configuration — Specify configuration
Simulink (default) | Two-way connection

Specify the port configuration.

Coupling Configuration — Specify configuration
Shaft split (default) | Shaft merge

Specify the coupling type.

Output Info bus — Selection
off (default) | on

Select to create the Info output port.

Coupling 1

Torsional stiffness, k1 — Stiffness
5e4 (default) | scalar

Rotational inertia, k_1, in N·m/rad.

Torsional damping, b1 — Damping
1e2 (default) | scalar

Torsional damping, b_1, in N·m· s/rad.

Damping cutoff frequency, omega1_c — Frequency
3000 (default) | scalar

Damping cutoff frequency, in rad/s.

Coupling 2

Torsional stiffness, k2 — Stiffness
5e4 (default) | scalar

Rotational inertia, k_2, in N·m/rad.

Torsional damping, b2 — Damping
1e2 (default) | scalar

Torsional damping, b_2, in N·m· s/rad.

Damping cutoff frequency, omega2_c — Frequency
3000 (default) | scalar

Damping cutoff frequency, in rad/s.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Rotational Inertia | Torsional Compliance

Introduced in R2017b

Torsional Compliance

Parallel spring-damper

Library: Powertrain Blockset / Drivetrain / Couplings
                      Vehicle Dynamics Blockset / Powertrain / Drivetrain /
                      Couplings

Description

The Torsional Compliance block implements a parallel spring-damper to couple two rotating driveshafts. The block uses the driveshaft angular velocities, torsional stiffness, and torsional damping to determine the torques.

$$T_R = -(\omega_R - \omega_C)b - \theta k$$

$$T_C = (\omega_R - \omega_C)b + \theta k$$

$$\dot{\theta} = (\omega_R - \omega_C)$$

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Variable	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrR	Mechanical power from driveshaft R	P_{TR}	$P_{TR} = T_R\omega_R$
		PwrC	Mechanical power from driveshaft C	P_{TC}	$P_{TC} = T_c\omega_c$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrDampLoss	Mechanical damping loss	P_d	$P_d = -b\left\|\dot{\theta}\right\|^2$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	PwrStoredShft	Rate change in spring energy	P_s	$P_s = -\theta k\dot{\theta}$

The equations use these variables.

T_R	Driveshaft R torque
T_C	Driveshaft C torque
ω_R	Driveshaft R angular velocity
ω_C	Driveshaft C angular velocity
θ	Coupled driveshaft rotation
k	Driveshaft torsional stiffness
b	Rotational viscous damping
P_d	Power loss due to damping
P_s	Rate change of stored spring energy

Ports

Input

RSpd — Driveshaft R angular velocity
scalar

Input driveshaft angular velocity, in rad/s.

Dependencies

To enable this port, for **Port Configuration**, select Simulink.

CSpd — Driveshaft C angular velocity
scalar

Output driveshaft angular velocity, in rad/s.

Dependencies

To enable this port, for **Port Configuration**, select Simulink.

R — Angular velocity and torque
two-way connector port

Angular velocity in rad/s. Torque is in N·m.

Dependencies

To enable this port, for **Port Configuration**, select Two-way connection.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal		Description	Variable	Units
Trq	R	Input driveshaft torque	T_R	N·m
	C	Output driveshaft torque	T_C	N·m

Signal			Description	Variable	Units
	Damp		Damping torque	$T_s = b\dot\theta$	N·m
	Spring		Spring torque	$T_d = k\theta$	N·m
Spd	R		Input driveshaft angular velocity	ω_R	rad/s
	C		Output driveshaft angular velocity	ω_C	rad/s
	deltadot		Difference in input and output driveshaft angular velocity	$\dot\theta$	rad/s
PwrInfo	PwrTrnsfrd	PwrR	Mechanical power from driveshaft R	P_{TR}	W
		PwrC	Mechanical power from driveshaft C	P_{TC}	W
	PwrNotTrnsfrd	PwrDampLoss	Power loss due to damping	P_d	W
	PwrStored	PwrStoredShft	Rate change of stored internal kinetic energy	P_s	W

Dependencies

To enable this port, select **Output Info bus**.

RTrq — Driveshaft R torque
scalar

Input drive shaft torque, in N·m.

Dependencies

To enable this port, for **Port Configuration**, select Simulink.

CTrq — Driveshaft C torque
scalar

Applied output driveshaft torque, in N·m.

Dependencies

To enable this port, for **Port Configuration**, select Simulink.

C — Angular velocity and torque
two-way connector port

Angular velocity in rad/s. Torque is in N·m.

Dependencies

To enable this port, for **Port Configuration**, select Two-way connection.

Parameters

Block Options

Port Configuration — Specify configuration
Simulink (default) | Two-way connection

Specify the port configuration.

Dependencies

Specifying Simulink creates these ports:

- RSpd
- CSpd
- RTrq
- CTrq

Specifying Two-way connection creates these ports:

- R
- C

Output Info bus — Selection
off (default) | on

Select to create the Info output port.

Torsional stiffness, k — Inertia
1e4 (default) | scalar

Torsional stiffness, in N·m/rad.

Torsional damping, b — Damping
1e2 (default) | scalar

Torsional damping, in N·m· s/rad.

Initial deflection, theta_o — Angular
0 (default) | scalar

Initial deflection, in rad.

Initial velocity difference, domega_o — Angular
0 (default) | scalar

Initial velocity difference, in rad/s.

Damping cut-off frequency, omega_c — Frequency
3000 (default) | scalar

Damping cut-off frequency, in rad/s.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Rotational Inertia | Split Torsional Compliance

Introduced in R2017a

Active Differential

Spur or planetary active differential gear
Library: Vehicle Dynamics Blockset / Powertrain / Drivetrain / Final Drive Unit

Spur gears, superposition clutches
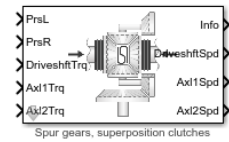
Description

The Active Differential block implements an active differential to account for the power transfer from the transmission to the axles. The block models the active differential as an open differential coupled to either a spur or planetary differential gear set. The block uses external pressure signals to regulate the clutch pressure to either speed up or slow down each axle rotation.

Use the block in hardware-in-the-loop (HIL) and optimization workflows to dynamically couple the driveshaft to the wheel axles when you want to direct the transmission torque to a specific axle. For detailed front wheel driving studies, use the block to couple the driveshaft to universal joints. The block is suitable to use in system-level closed-loop control studies, for example, yaw stability and torque vectoring. All the parameters are tunable.

To specify the active differential, open the **Active Differential** parameters and specify **Active differential type**.

Setting	Block Implementation
Spur gears, superposition clutches	Clutches are in superposition through a three-gang gear system and a differential case
Double planetary gears, stationary clutches	Clutches are fixed to the carrier and axles through double planetary gear sets

Use the **Open Differential** parameter **Crown wheel (ring gear) located** to specify the open differential location, either to the left or right of the center-line.

Depending on the available data, to specify the method to couple the different torques applied to the axles, use the **Slip Coupling** parameter **Coupling type**.

Setting	Block Implementation
Pre-loaded ideal clutch	Torque modeled as a dry clutch with constant friction coefficients
Slip speed dependent torque data	Torque determined from a lookup table that is a function of slip-speed and clutch pressure

The Active Differential block does not include a controller or external clutch actuator dynamics. Use this information to control the input clutch pressure. The info bus contains the slip speeds at clutch 1, $\Delta\omega_{cl1}$, and clutch 2, $\Delta\omega_{cl2}$.

Input Axle Torque	$\Delta\omega_{cl1}$	$\Delta\omega_{cl2}$	Input Clutch Pressure
Positive axle 1 torque	> 0	N/A	Increase clutch 1 pressure
Positive axle 1 torque	< 0	N/A	Disengage clutch 1 and 2
Positive axle 2 torque	N/A	> 0	Increase clutch 1 pressure
Positive axle 2 torque	N/A	< 0	Disengage clutch 1 and 2

Differentials

The Active Differential block implements these equations to represent the mechanical dynamic response for the superposition and stationary clutch configurations. To determine the gear ratios, the block uses the clutch speed and the number of teeth for each gear pair. The allowable wheel speed difference (AWSD) limits the wheel speed difference for positive torque.

Mechanical Dynamic Response	Equations	
	Superposition Clutches and Spur Gearing	Stationary Clutches and Planetary Gearing
Crown gear	$\dot{\omega}_d(J_d + J_{gs}) = T_d - \omega_d b_d - T_i$	$\dot{\omega}_d(\ J_d + J_{s1} + J_{s2}) = T_d - \omega_d b_d - T_i$
Axle 1	$\dot{\omega}_1 J_1 = \eta T_1 - \omega_1 b_1 - T_{i1}$	$\dot{\omega}_1(J_1 + J_{r1}) = T_1 - \omega_1 b_1 - T_{i1}$
Axle 2	$\dot{\omega}_2 J_2 = \eta T_2 - \omega_2 b_2 - T_{i2}$	$\dot{\omega}_2(J_{axle2} + J_{r1}) = T_2 - \omega_2 b_2 - T_{i2}$
Gear ratios	$\dfrac{\omega_{cl1}}{\omega_d} = N_{s1} = \dfrac{z_1 z_6}{z_4 z_3}$ $\dfrac{\omega_{cl2}}{\omega_d} = N_{s2} = \dfrac{z_1 z_5}{z_4 z_2}$	$\dfrac{\omega_{cl1}}{\omega_d} = N_{p1} = \dfrac{z_1 z_6}{z_4 z_3}$ $\dfrac{\omega_{cl2}}{\omega_d} = N_{p2} = \dfrac{z_1 z_5}{z_4 z_2}$
Rigid Coupling Constraints	$T_1 = \dfrac{NT_i}{2} - \dfrac{N_{s2}}{2}T_{cl2} + \dfrac{N_{s1}}{2}T_{cl1}$ $T_2 = \dfrac{NT_i}{2} + (1 - \dfrac{N_{s2}}{2})T_{cl2} - (1 - \dfrac{N_{s1}}{2})T_{cl1}$ $\omega_d = \dfrac{N}{2}(\omega_1 + \omega_2)$	$T_1 = \dfrac{NT_i}{2} - \dfrac{N_{p2}}{(N_{p2}-1)2}T_{cl2} + \dfrac{(2-N_{p1})}{(N_{p1}-1)2}T_{cl1}$ $T_2 = \dfrac{NT_i}{2} + \dfrac{(2-N_{p2})}{(N_{p2}-1)2}T_{cl2} - \dfrac{N_{p1}}{(N_{p1}-1)2}T_{cl1}$ $\omega_d = \dfrac{N}{2}(\omega_1 + \omega_2)$
Allowable wheel speed difference (AWSD)	$\overline{\Delta\omega}_{max} = (N_{s2} - N_{s1}) \cdot 100\%$	$\overline{\Delta\omega}_{max} = (N_{p1,2} - 1) \cdot 100\%$

Superposition Clutches and Spur Gearing

These superposition clutch illustrations show the clutch configuration and schematic for torque transfer to the left wheel.

Torque transfer to left wheel

Propeller shaft

Active yaw control (AYC) differential

Hydraulic multi-plate clutch (left clutch)

Left axle

Right axle
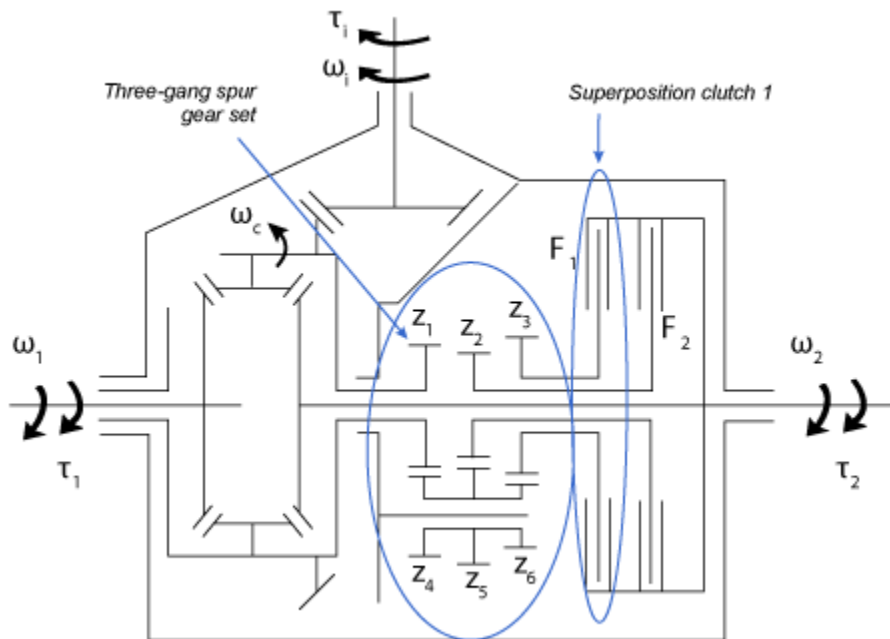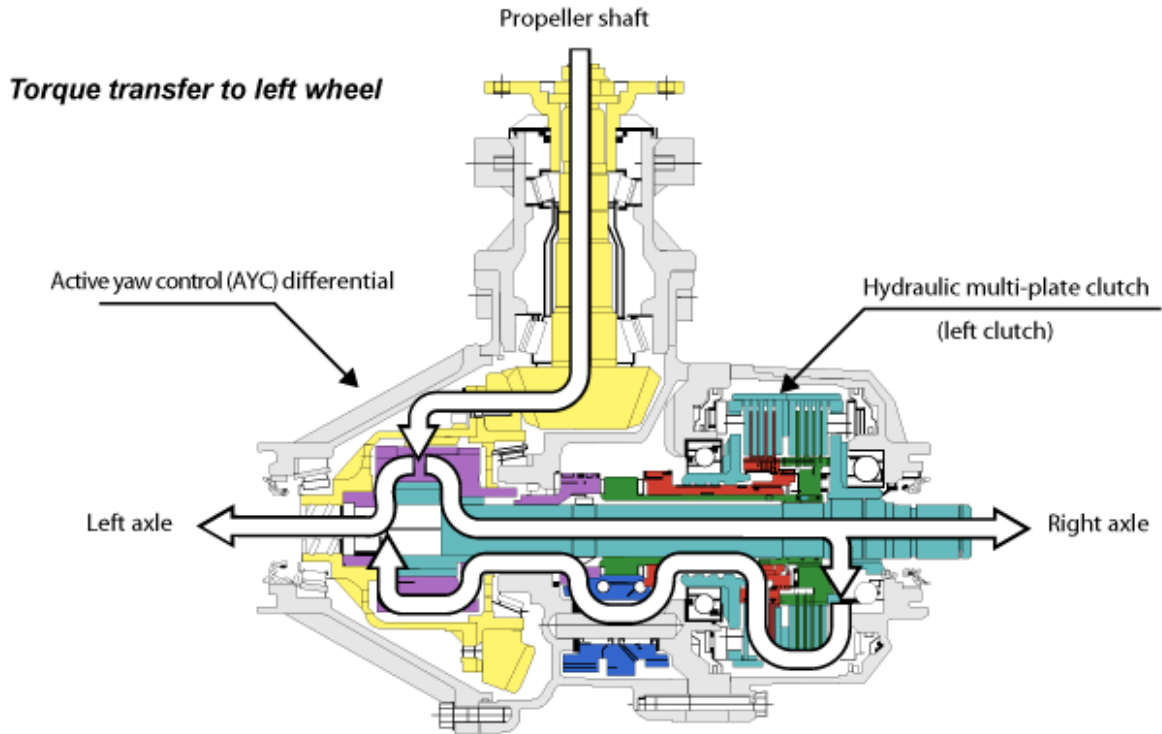
Stationary Clutches and Planetary Gearing

The illustrations show the stationary clutch configuration and schematic.
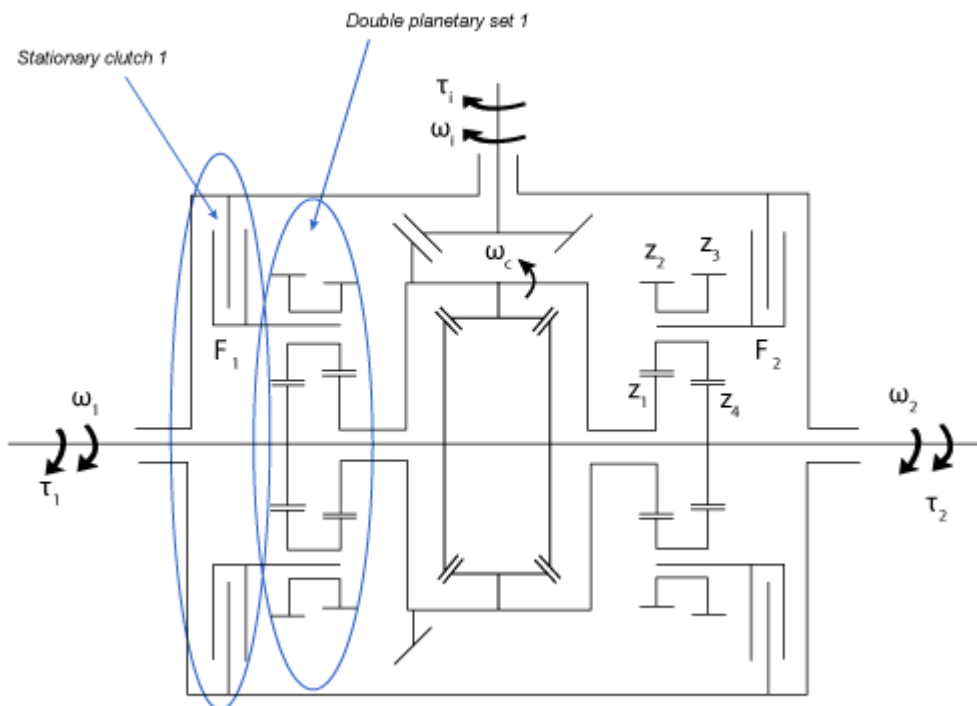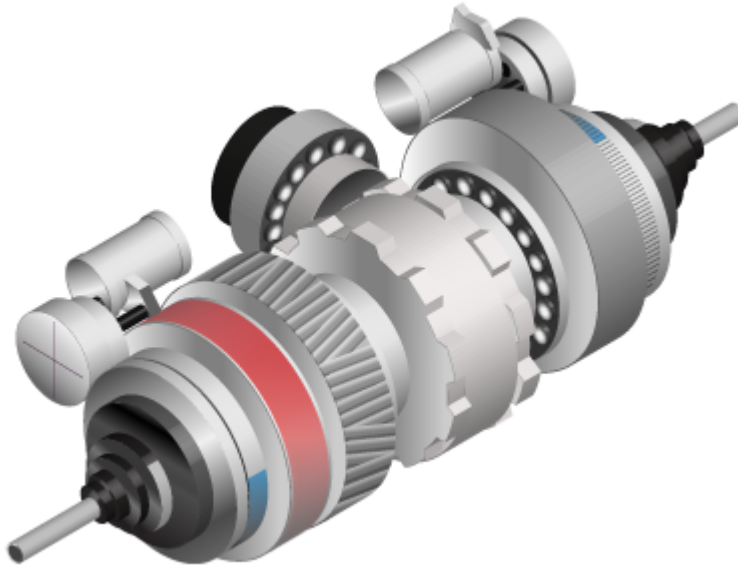
Slip Coupling

For both the ideal clutch and slip-speed configurations, the slip coupling is a function of the slip-speed and clutch pressure. The slip-speed depends on the slip velocity at each of the clutch interfaces.

$$\varpi = [\Delta\omega_{c1}, \Delta\omega_{c2}]$$

Ideal Clutch

The ideal clutch coupling model uses the axle slip speed, clutch pressure, and friction to calculate the clutch torque. The friction coefficient is a function of the slip speed.

$$T_C = F_T N_d \mu(|\varpi|) R_{eff} \tanh(4\varpi)$$

To calculate the total clutch force, the block uses the effective radius, clutch pressure, and clutch preload force.

$$F_T = \quad F_C + P_{1,2}A_{eff}, \quad F_T \geq 0$$

The disc radii determine the effective clutch radius over which the clutch force acts.

$$R_{eff} = \frac{2(R_o^3 - R_i^3)}{3(R_o^2 - R_i^2)}$$

Slip-Speed

To calculate the clutch torque, the slip speed coupling model uses torque data that is a function of slip speed and clutch pressure. The angular velocities of the axles determine the slip speed.

$$T_C = T_C(\varpi, \quad P_{1,2})$$

The equations use these variables.

A_{eff}	Effective clutch pressure area
b_d	Crown gear linear viscous damping
b_1, b_2	Axle 1 and 2 linear viscous damping, respectively
F_c, F_T	Clutch preload force and total force, respectively
J_d	Carrier rotational inertia
J_{gc}	Three-gang gear rotational inertia
J_{c1}, J_{c2}	Planetary carrier 1 and 2 rotational inertia, respectively
J_{r1}, J_{r2}	Planetary ring gear 1 and 2 rotational inertia, respectively
J_{s1}, J_{s2}	Planetary sun gear 1 and 2 rotational inertia, respectively
J_1, J_2	Axle 1 and 2 rotational inertia, respectively
N	Carrier-to-drive shaft gear ratio
N_d	Number of disks
N_{s1}, N_{s2}	Clutch 1 and 2 carrier-to-spur gear ratio, respectively
N_{p1}, N_{p2}	Planetary 1 and 2 carrier-to-axle gear ratio, respectively
P_1, P_2	Clutch 1 and 2 pressure, respectively

R_{eff}	Effective clutch radius
R_i, R_o	Annular disk inner and outer radius, respectively
T_c	Clutch torque
T_{cl1}, T_{cl2}	Clutch 1 and 2 coupling torque, respectively
T_d	Driveshaft torque
T_1, T_2	Axle 1 and 2 torque, respectively
T_i	Axle internal resistance torque
T_{i1}, T_{i2}	Axle 1 and 2 internal resistance torque
ω_d	Driveshaft angular velocity
ϖ	Slip speed
ω_1, ω_2	Axle 1 and 2 angular velocity, respectively
$\Delta\omega_{cl1}$, $\Delta\omega_{cl2}$	Clutch 1 and 2 slip speed at interface, respectively
ω_{cl1}, ω_{cl2}	Clutch 1 and 2 angular velocity, respectively
μ	Clutch coefficient of friction
z_i	Number of teeth on gear i

Ports

Inputs

Prs1 — Clutch 1 pressure
scalar

Clutch 1 pressure, P_1, in Pa.

Prs2 — Clutch 2 pressure
scalar

Clutch 2 pressure, P_2, in Pa.

DriveshftTrq — Driveshaft torque
scalar

Applied input torque, T_d, typically from the engine driveshaft, in N·m.

Axl1Trq — Torque
scalar

Axle 1 torque, T_1, in N·m.

Axl2Trq — Torque
scalar

Axle 2 torque, T_2, in N·m.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal		Description	Units
Driveshft	DriveshftTrq	Drive shaft torque	N·m
	DriveshftSpd	Drive shaft angular velocity	rad/s
Axl1	Axl1Trq	Axle 1 torque	N·m
	Axl1Spd	Axle 1 angular velocity	rad/s
Axl2	Axl2Trq	Axle 2 torque	N·m
	Axl2Spd	Axle 2 angular velocity	rad/s
Cplng	CplngTrq1	Clutch 1 coupling torque	N·m
	CplngTrq2	Clutch 2 coupling torque	N·m
	CplngSlipSpd1	Clutch 1 slip speed	rad/s
	CplngSlipSpd2	Clutch 2 slip speed	rad/s
	CplngPrs1	Clutch 1 input pressure	Pa
	CplngPrs2	Clutch 2 input pressure	Pa

DriveshftSpd — Angular velocity
scalar

Driveshaft angular velocity, ω_d, in rad/s.

Axl1Spd — Angular velocity
scalar

Axle 1 angular velocity, ω_1, in rad/s.

Axl2Spd — Angular velocity
scalar

Axle 2 angular velocity, ω_2, in rad/s.

Parameters

Active Differential

Active differential type — Differential
Spur gears, superposition clutches (default) | Double planetary gears, stationary clutches

Specify the type of active differential.

Setting	Block Implementation
Spur gears, superposition clutches	Clutches are in superposition through a three-gang gear system and a differential case
Double planetary gears, stationary clutches	Clutches are fixed to the carrier and axles through double planetary gear sets

Clutch 1 to differential case gear ratio, Ns1 — Clutch 1-spur gear ratio
.875 (default) | scalar

Clutch 1-to-carrier spur gear ratio, N_{s1}, dimensionless.

Dependencies

To enable the spur gear parameters, select Spur gears, superposition clutches for the **Active differential type** parameter.

Clutch 2 to differential case gear ratio, Ns2 — Clutch 2-spur gear ratio
1.125 (default) | scalar

Clutch 2-to-carrier spur gear ratio, N_{s2}, dimensionless.

Dependencies

To enable the spur gear parameters, select Spur gears, superposition clutches for the **Active differential type** parameter.

Three-gang gear inertia, Jgc — Rotational inertia
.003 (default) | scalar

Three-gang gear rotational inertia, J_{gc}, in kg·m^2.

Dependencies

To enable the spur gear parameters, select Spur gears, superposition clutches for the **Active differential type** parameter.

Axle 1 planetary carrier to axle gear ratio, Np1 — Planetary 1 carrier gear ratio
1.125 (default) | scalar

Planetary 1 carrier-to-axle gear ratio, N_{p1}, dimensionless.

Dependencies

To enable the planetary gear parameters, select Double planetary gears, stationary clutches for the **Active differential type** parameter.

Axle 1 sun gear inertia, Js1 — Planetary 1 sun gear inertia
.001 (default) | scalar

Planetary 1 sun gear inertia, J_{s1}, in kg·m^2.

Dependencies

To enable the planetary gear parameters, select Double planetary gears, stationary clutches for the **Active differential type** parameter.

Axle 1 carrier inertia, Jc1 — Planetary 1 carrier inertia
.001 (default) | scalar

Planetary 1 carrier inertia, J_{c1}, in kg·m^2.

Dependencies

To enable the planetary gear parameters, select Double planetary gears, stationary clutches for the **Active differential type** parameter.

Axle 1 ring inertia, Jr1 — Planetary 1 ring gear inertia
.002 (default) | scalar

Planetary 1 ring gear inertia, J_{r1}, kg·m^2.

Dependencies

To enable the planetary gear parameters, select Double planetary gears, stationary clutches for the **Active differential type** parameter.

Axle 2 planetary carrier to axle gear ratio, Np2 — Planetary 2 carrier gear ratio
1.125 (default) | scalar

Planetary 2 carrier-to-axle gear ratio, N_{p2}, dimensionless.

Dependencies

To enable the planetary gear parameters, select Double planetary gears, stationary clutches for the **Active differential type** parameter.

Axle 2 sun gear inertia, Js2 — Planetary 2 sun gear inertia
.001 (default) | scalar

Planetary 2 sun gear inertia, J_{s2}, in kg·m^2.

Dependencies

To enable the planetary gear parameters, select Double planetary gears, stationary clutches for the **Active differential type** parameter.

Axle 2 carrier inertia, Jc2 — Planetary 2 carrier inertia
.001 (default) | scalar

Planetary 2 carrier inertia, J_{c2}, in kg·m^2.

Dependencies

To enable the planetary gear parameters, select Double planetary gears, stationary clutches for the **Active differential type** parameter.

Axle 2 ring inertia, Jr2 — Planetary 2 ring gear inertia
.002 (default) | scalar

Planetary 2 ring gear inertia, J_{r2}, in kg·m^2.

Dependencies

To enable the planetary gear parameters, select Double planetary gears, stationary clutches for the **Active differential type** parameter.

Open Differential

Crown wheel (ring gear) located — Specify crown wheel connection
To the left of center-line (default) | To the right of center-line

Specify the crown wheel connection to the drive shaft.

Carrier to drive shaft ratio, NC/ND — Ratio
4 (default) | scalar

Carrier-to-drive shaft gear ratio, *N*.

Carrier inertia, Jd — Inertia
.1 (default) | scalar

Rotational inertia of the crown gear assembly, J_d, in kg·m^2. You can include the drive shaft inertia.

Carrier damping, bd — Damping
1e-3 (default) | scalar

Crown gear linear viscous damping, b_d, in N·m·s/rad.

Axle 1 inertia, Jw1 — Inertia
.1 (default) | scalar

Axle 1 rotational inertia, J_1, in kg·m^2.

Axle 1 damping, bw1 — Damping
1e-3 (default) | scalar

Axle 1 linear viscous damping, b_1, in N·m·s/rad.

Axle 2 inertia, Jw2 — Inertia
.1 (default) | scalar

Axle 2 rotational inertia, J_2, in kg·m^2.

Axle 2 damping, bw2 — Damping
1e-3 (default) | scalar

Axle 2 linear viscous damping, b_2, in N·m·s/rad.

Axle 1 initial velocity, omegaw1o — Angular velocity
0 (default) | scalar

Axle 1 initial velocity, ω_{o1}, in rad/s.

Axle 2 initial velocity, omegaw2o — Angular velocity
0 (default) | scalar

Axle 2 initial velocity, ω_{o2}, in rad/s.

Slip Coupling

Coupling type — Torque coupling
Ideal pre-loaded clutch (default) | Slip speed dependent torque data | Input torque dependent torque data

Specify the type of torque coupling.

Setting	Block Implementation
Pre-loaded ideal clutch	Torque modeled as a wet clutch with a constant velocity

Setting	Block Implementation
Slip speed dependent torque data	Torque determined from a lookup table that is a function of slip-speed and clutch pressure

Effective applied pressure area — Pressure area
0.01 (default) | scalar

Effective applied pressure area, in N/m^2.

Dependencies

To enable the clutch parameters, select Ideal pre-loaded clutch for the **Coupling type** parameter.

Number of disks, Ndisks — Torque coupling
4 (default) | scalar

Number of disks.

Dependencies

To enable the clutch parameters, select Ideal pre-loaded clutch for the **Coupling type** parameter.

Effective radius, Reff — Radius
.20 (default) | scalar

The effective radius, R_{eff}, used with the applied clutch friction force to determine the friction force. The effective radius is defined as:

$$R_{eff} = \frac{2(R_o{}^3 - R_i{}^3)}{3(R_o{}^2 - R_i{}^2)}$$

The equation uses these variables.

R_o	Annular disk outer radius
R_i	Annular disk inner radius

Dependencies

To enable the clutch parameters, select Ideal pre-loaded clutch for the **Coupling type** parameter.

Nominal preload force, Fc — Force
500 (default) | scalar

Nominal preload force, in N.

Dependencies

To enable the clutch parameters, select Ideal pre-loaded clutch for the **Coupling type** parameter.

Friction coefficient vector, mu — Friction
[.16 0.13 0.115 0.11 0.105 0.1025 0.10125 .10125] (default) | vector

Friction coefficient vector.

Dependencies

To enable the clutch parameters, select `Ideal pre-loaded clutch` for the **Coupling type** parameter.

`Slip speed vector, dw` — Angular velocity
[0 10 20 40 60 80 100 500] (default) | vector

Slip speed vector, in rad/s.

To enable the clutch parameters, select `Ideal pre-loaded clutch` for the **Coupling type** parameter.

`Torque - slip speed matrix, TdPdw` — Clutch torque
[-1000,-500,-90,-50,-5,0,5,50,90,500,1000].*ones(11) (default) | matrix

Torque matrix, T_c, in N·m.

Dependencies

To enable the slip speed parameters, select `Slip speed dependent torque data` for the **Coupling type** parameter.

`Clutch pressure vector, pT` — Clutch pressure breakpoints
[0 1e3 5e3 7e3 1e4 2e4 5e4 1e5 5e5 1e6 5e6] (default) | vector

Clutch pressure breakpoints vector, $P_{1,2}$, in Pa.

Dependencies

To enable the slip speed parameters, select `Slip speed dependent torque data` for the **Coupling type** parameter.

`Slip speed vector, dwT` — Slip speed breakpoints
[-500 -200, -175, -100, - 50, 0, 50, 100, 175, 200, 500] (default) | vector

Slip speed breakpoints vector, ω, in rad/s.

Dependencies

To enable the slip speed parameters, select `Slip speed dependent torque data` for the **Coupling type** parameter.

`Coupling time constant, tauC` — Constant
.01 (default) | scalar

Coupling time constant, in s.

References

[1] Deur, J., Ivanović, V., Hancock, M., and Assadian, F. "Modeling of Active Differential Dynamics." In ASME proceedings. *Transportation Systems*. Vol. 17, pp: 427-436.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Limited Slip Differential | Open Differential

Introduced in R2018b

Limited Slip Differential

Limited differential as a planetary bevel gear

Library: Powertrain Blockset / Drivetrain / Final Drive Unit
Vehicle Dynamics Blockset / Powertrain / Drivetrain / Final
Drive Unit
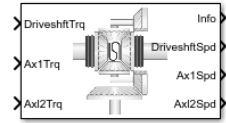
Description

The Limited Slip Differential block implements a differential as a planetary bevel gear train. The block matches the driveshaft bevel gear to the crown (ring) bevel gear. You can specify:

- Carrier-to-driveshaft ratio
- Crown wheel location
- Viscous and damping coefficients for the axles and carrier
- Type of slip coupling

Use the block in system-level driveline analysis to account for the power transfer from the transmission to the wheels. The block is suitable for use in hardware-in-the-loop (HIL) and optimization workflows. All the parameters are tunable.

In a limited slip differential, to prevent one of the wheels from slipping, the differential splits the torque applied to the left and right axles. With different torque applied to the axles, the wheels can move at different angular velocities, preventing slip. The block implements three methods for coupling the different torques applied to the axes:

- Pre-loaded ideal clutch
- Slip speed-dependent torque data
- Input torque dependent torque data

The block uses a coordinate system that produces positive tire and vehicle motion for standard engine, transmission, and differential configurations. The arrows indicate positive motion.

Efficiency

To account for the block efficiency, use the **Efficiency factors** parameter. This table summarizes the block implementation for each setting.

Setting	Implementation
`Constant`	Constant efficiency that you can set with the **Constant efficiency factor, eta** parameter.
`Driveshaft torque, temperature and speed`	Efficiency as a function of base gear input torque, air temperature, and driveshaft speed. Use these parameters to specify the lookup table and breakpoints: • **Efficiency lookup table, eta_tbl** • **Efficiency torque breakpoints, Trq_bpts** • **Efficiency speed breakpoints, omega_bpts** • **Efficiency temperature breakpoints, Temp_bpts** For the air temperature, you can either: • Select **Input temperature** to create an input port. • Set a **Ambient temperature, Tamb** parameter value. To select the interpolation method, use the **Interpolation method** parameter. For more information, see "Interpolation Methods" (Simulink).

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrDriveshft	Mechanical power from driveshaft	$\eta T_d \omega_d$
		PwrAxl1	Mechanical power from axle 1	$\eta T_1 \omega_1$
		PwrAxl2	Mechanical power from axle 2	$\eta T_2 \omega_2$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrMechLoss	Total power loss	$\dot{W}_{loss} = -(P_t + P_d + P_c) + P_s$ $P_t = \eta(T_d\omega_d + T_1\omega_1 + T_2\omega_2)$
		PwrDampLoss	Power loss due to damping	$P_d = -(b_1|\omega_1| + b_2|\omega_2| + b_d|\omega_d|)$
		PwrCplngLoss	Power loss due to clutch	$P_c = T_c|\overline{\omega}|$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	PwrStoredShft	Rate change of stored internal energy	$P_s = -(\omega_1\dot{\omega}_1 J_1 + \omega_2\dot{\omega}_2 J_2 + \omega_d\dot{\omega}_d J_d)$

Dynamics

The Limited Slip Differential block implements these differential equations to represent the mechanical dynamic response for the crown gear, left axle, and right axle.

Mechanical Dynamic Response	Differential Equation
Crown Gear	$\dot{\omega}_d J_d = \eta T_d - \omega_d b_d - T_i$
Left Axle	$\dot{\omega}_1 J_1 = \eta T_1 - \omega_1 b_1 - T_{i1}$
Right Axle	$\dot{\omega}_2 J_2 = \eta T_2 - \omega_2 b_2 - T_{i2}$

The block assumes rigid coupling between the crown gear and axles. These constraint equations apply.

$$\eta T_1 = \frac{N}{2}T_i - \frac{1}{2}T_c$$

$$\eta T_2 = \frac{N}{2}T_i + \frac{1}{2}T_c$$

$$\omega_d = \frac{N}{2}(\omega_1 + \omega_2)$$

The equations use these variables.

N	Carrier-to-driveshaft gear ratio
J_d	Rotational inertia of the crown gear assembly
b_d	Crown gear linear viscous damping
ω_d	Driveshaft angular speed
ϖ	Slip speed
J_1	Axle 1 rotational inertia
b_1	Axle 1 linear viscous damping
ω_1	Axle 1 speed
J_2	Axle 2 rotational inertia
b_2	Axle 2 linear viscous damping
ω_2	Axle 2 angular speed
η	Efficiency
T_d	Driveshaft torque
T_1	Axle 1 torque
T_2	Axle 2 torque
T_i	Axle internal resistance torque
T_{i1}	Axle 1 internal resistance torque
T_{i2}	Axle 2 internal resistance torque
μ	Coefficient of friction
R_{eff}	Effective clutch radius
R_o	Annular disk outer radius
R_i	Annular disk inner radius
F_c	Clutch force
T_c	Clutch torque
μ	Coefficient of friction

Table blocks in the Limited Slip Differential have these parameter settings:

- **Interpolation method** — `Linear`
- **Extrapolation method** — `Clip`

Ideal Clutch Coupling

The ideal clutch coupling model uses the axle slip speed and friction to calculate the clutch torque. The friction coefficient is a function of the slip speed.

$$T_c = F_c N \mu(|\varpi|) R_{eff} \tanh(4|\varpi|)$$

The disc radii determine the effective clutch radius over which the clutch force acts.

$$R_{eff} = \frac{2(R_o3 - R_i3)}{3(R_o2 - R_i2)}$$

The angular velocities of the axles determine the slip speed.

$$\varpi = \omega_1 - \omega_2$$

Slip Speed Coupling

To calculate the clutch torque, the slip speed coupling model uses torque data that is a function of slip speed. The angular velocities of the axles determine the slip speed.

$$\varpi = \omega_1 - \omega_2$$

Input Torque Coupling

To calculate the clutch torque, the input torque coupling model uses torque data that is a function of input torque.

The Open Differential block assumes rigid coupling between the crown gear and axles. These constraint equations apply.

$$\eta T_1 = \quad \eta T_2 = \frac{N}{2} T_i$$

$$\omega_{d =} = \frac{N}{2}(\omega_1 + \omega_2)$$

Ports

Inputs

DriveshftTrq — Torque
scalar

Applied input torque, typically from the engine crankshaft, in N·m.

Axl1Trq — Torque
scalar

Axle 1 torque, T_1, in N·m.

Axl2Trq — Torque
scalar

Axle 2 torque, T_2, in N·m.

Temp — Temperature
scalar

Temperature, in K.

Dependencies

To enable this port:

- Set **Efficiency factors** to `Driveshaft torque, speed and temperature`.
- Select **Input temperature**.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal			Description	Units
Driveshft	DriveshftTrq		Driveshaft torque	N·m
	DriveshftSpd		Driveshaft speed	rad/s
Axl1	Axl1Trq		Axle 1 torque	N·m
	Axl1Spd		Axle 1 speed	rad/s
Axl2	Axl2Trq		Axle 2 torque	N·m
	Axl2Spd		Axle 2 speed	rad/s
Cplng	CplngTrq		Torque coupling	N·m
	CplngSlipSpd		Slip speed	rad/s
PwrInfo	PwrTrnsfrd	PwrDriveshft	Mechanical power from driveshaft	W
		PwrAxl1	Mechanical power from axle 1	W
		PwrAxl2	Mechanical power from axle 2	W
	PwrNotTrnsfrd	PwrMechLoss	Total power loss	W
		PwrDampLoss	Power loss due to damping	W
		PwrCplngLoss	Power loss due to clutch	W
	PwrStoredShft	PwrStoredShft	Rate change of stored internal energy	W

DriveshftSpd — Angular speed
scalar

Driveshaft angular speed, ω_d, in rad/s.

Axl1Spd — Angular speed
scalar

Axle 1 angular speed, ω_1, in rad/s.

Axl2Spd — Angular speed
scalar

Axle 2 angular speed, ω_2, in rad/s.

Parameters

Block Options

Efficiency factors — Specify configuration
Constant (default) | Driveshaft torque, speed and temperature

To account for the block efficiency, use the **Efficiency factors** parameter. This table summarizes the block implementation for each setting.

Setting	Implementation
Constant	Constant efficiency that you can set with the **Constant efficiency factor, eta** parameter.
Driveshaft torque, temperature and speed	Efficiency as a function of base gear input torque, air temperature, and driveshaft speed. Use these parameters to specify the lookup table and breakpoints: • **Efficiency lookup table, eta_tbl** • **Efficiency torque breakpoints, Trq_bpts** • **Efficiency speed breakpoints, omega_bpts** • **Efficiency temperature breakpoints, Temp_bpts** For the air temperature, you can either: • Select **Input temperature** to create an input port. • Set a **Ambient temperature, Tamb** parameter value. To select the interpolation method, use the **Interpolation method** parameter. For more information, see "Interpolation Methods" (Simulink).

Interpolation method — Method
Flat | Nearest | Linear point-slope | Linear Lagrange | Cubic spline

For more information, see "Interpolation Methods" (Simulink).

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Input temperature — Create input port
off (default) | on

Select to create input port Temp for the temperature.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Open Differential

Crown wheel (ring gear) located — Specify crown wheel connection
To the left of center-line (default) | To the right of center-line

Specify the crown wheel connection to the driveshaft.

Carrier to drive shaft ratio, NC/ND — Ratio
4 (default) | scalar

Carrier-to-driveshaft gear ratio, N.

Carrier inertia, Jd — Inertia
.1 (default) | scalar

Rotational inertia of the crown gear assembly, J_d, in kg·m^2. You can include the driveshaft inertia.

Carrier damping, bd — Damping
1e-3 (default) | scalar

Crown gear linear viscous damping, b_d, in N·m·s/rad.

Axle 1 inertia, Jw1 — Inertia
.1 (default) | scalar

Axle 1 rotational inertia, J_1, in kg·m^2.

Axle 1 damping, bw1 — Damping
1e-3 (default) | scalar

Axle 1 linear viscous damping, b_1, in N·m·s/rad.

Axle 2 inertia, Jw2 — Inertia
.1 (default) | scalar

Axle 2 rotational inertia, J_2, in kg·m^2.

Axle 2 damping, bw2 — Damping
1e-3 (default) | scalar

Axle 2 linear viscous damping, b_2, in N·m·s/rad.

Axle 1 initial velocity, omegaw1o — Angular velocity
0 (default) | scalar

Axle 1 initial velocity, ω_{o1}, in rad/s.

Axle 2 initial velocity, omegaw2o — Angular velocity
0 (default) | scalar

Axle 2 initial velocity, ω_{o2}, in rad/s.

Constant efficiency factor, eta — Efficiency
1 (default) | scalar

Constant efficiency, η.

Dependencies

To enable this parameter, set **Efficiency factors** to Constant.

Efficiency lookup table, eta_tbl — Lookup table
M-by-N-by-L array

Dimensionless array of values for efficiency as a function of:

- M input torques
- N input speed
- L air temperatures

Each value specifies the efficiency for a specific combination of torque, speed, and temperature. The array size must match the dimensions defined by the torque, speed, and temperature breakpoint vectors.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Efficiency torque breakpoints, Trq_bpts — Torque breakpoints
[25, 50, 75, 100, 150, 200, 250] (default) | 1-by-M vector

Vector of input torque, breakpoints for efficiency, in N·m.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Efficiency speed breakpoints, omega_bpts — Speed breakpoints
[52.4 78.5 105 131 157 183 209 262 314 419 524] (default) | 1-by-N vector

Vector of speed, breakpoints for efficiency, in rad/s.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Efficiency temperature breakpoints, Temp_bpts — Temperature breakpoints
[290 358] (default) | 1-by-L vector

Vector of ambient temperature breakpoints for efficiency, in K.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Ambient temperature, Tamb — Ambient temperature
297.15 (default) | scalar

Ambient air temperature, T_{air}, in K.

Dependencies

To enable this parameter:

- Set **Efficiency factors** to `Driveshaft torque, speed and temperature`.
- Clear **Input temperature**.

Slip Coupling

Coupling type — Torque coupling
`Pre-loaded ideal clutch` (default) | `Slip speed dependent torque data` | `Input torque dependent torque data`

Specify the type of torque coupling.

Number of disks, Ndisks — Torque coupling
`4` (default) | `scalar`

Number of disks.

Dependencies

To enable the ideal clutch parameters, select `Pre-loaded ideal clutch` for the **Coupling type** parameter.

Effective radius, Reff — Radius
`.20` (default) | `scalar`

The effective radius, R_{eff}, used with the applied clutch friction force to determine the friction force. The effective radius is defined as:

$$R_{eff} = \frac{2(R_o{}^3 - R_i{}^3)}{3(R_o{}^2 - R_i{}^2)}$$

The equation uses these variables.

R_o	Annular disk outer radius
R_i	Annular disk inner radius

Dependencies

To enable the clutch parameters, select `Pre-loaded ideal clutch` for the **Coupling type** parameter.

Nominal preload force, Fc — Force
`500` (default) | `scalar`

Nominal preload force, in N.

Dependencies

To enable the clutch parameters, select `Pre-loaded ideal clutch` for the **Coupling type** parameter.

Friction coefficient vector, muc — Friction
`[.16 0.13 0.115 0.11 0.105 0.1025 0.10125]` (default) | `vector`

Friction coefficient vector.

Dependencies

To enable the clutch parameters, select `Pre-loaded ideal clutch` for the **Coupling type** parameter.

Slip speed vector, `dw` — Angular velocity
`[0 10 20 40 60 80 100]` (default) | `vector`

Slip speed vector, in rad/s.

Dependencies

To enable the clutch parameters, select `Pre-loaded ideal clutch` for the **Coupling type** parameter.

Torque - slip speed vector, `Tdw` — Torque
`[-100, -90, -50, -5, 0, 5, 50, 90, 100]` (default) | `vector`

Torque vector, in N·m.

Dependencies

To enable the slip speed parameters, select `Slip speed dependent torque data` for the **Coupling type** parameter.

Slip speed vector, `dwT` — Angular velocity
`[-200, -175, -100, - 50, 0, 50, 100, 175, 200]` (default) | `vector`

Slip speed vector, in rad/s.

Dependencies

To enable the slip speed parameters, select `Slip speed dependent torque data` for the **Coupling type** parameter.

Torque - input torque vector, `TTin` — Torque
`[-200 -175 -100 - 50 0 50 100 175 200]` (default) | `vector`

Torque vector, in N·m.

Dependencies

To enable the input torque parameters, select `Input torque dependent torque data` for the **Coupling type** parameter.

Input torque vector, `Tin` — Torque
`[-200 -175 -100 - 50 0 50 100 175 200]` (default) | `vector`

Torque vector, in N·m.

Dependencies

To enable the input torque parameters, select `Input torque dependent torque data` for the **Coupling type** parameter.

Coupling time constant, `tauC` — Constant
`.01` (default) | `scalar`

Coupling time constant, in s.

References

[1] Deur, J., Ivanović, V., Hancock, M., and Assadian, F. "Modeling of Active Differential Dynamics." In ASME proceedings. *Transportation Systems*. Vol. 17, pp: 427-436.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Open Differential

Introduced in R2017a

Open Differential

Differential as a planetary bevel gear

Library: Powertrain Blockset / Drivetrain / Final Drive Unit
Vehicle Dynamics Blockset / Powertrain / Drivetrain / Final
Drive Unit
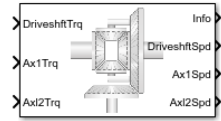
Description

The Open Differential block implements a differential as a planetary bevel gear train. The block matches the driveshaft bevel gear to the crown (ring) bevel gear. You can specify:

- Carrier-to-driveshaft ratio
- Crown wheel location
- Viscous and damping coefficients for the axles and carrier

Use the Open Differential block to:

- Dynamically couple the post-transmission driveshaft to the wheel axles or universal joints
- Model simplified or older drivetrains when optimal traction control does not require passive or active torque vectoring
- Model mechanical power splitting in generic gearbox and drive line scenarios

The block is suitable for use in hardware-in-the-loop (HIL) and optimization workflows. All the parameters are tunable.

The block uses a coordinate system that produces positive tire and vehicle motion for standard engine, transmission, and differential configurations. The arrows indicate positive motion.

Efficiency

To account for the block efficiency, use the **Efficiency factors** parameter. This table summarizes the block implementation for each setting.

Setting	Implementation
`Constant`	Constant efficiency that you can set with the **Constant efficiency factor, eta** parameter.
`Driveshaft torque, temperature and speed`	Efficiency as a function of base gear input torque, air temperature, and driveshaft speed. Use these parameters to specify the lookup table and breakpoints: • **Efficiency lookup table, eta_tbl** • **Efficiency torque breakpoints, Trq_bpts** • **Efficiency speed breakpoints, omega_bpts** • **Efficiency temperature breakpoints, Temp_bpts** For the air temperature, you can either: • Select **Input temperature** to create an input port. • Set a **Ambient temperature, Tamb** parameter value. To select the interpolation method, use the **Interpolation method** parameter. For more information, see "Interpolation Methods" (Simulink).

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrDriveshft	Mechanical power from driveshaft	$\eta T_d \omega_d$
		PwrAxl1	Mechanical power from axle 1	$\eta T_1 \omega_1$
		PwrAxl2	Mechanical power from axle 2	$\eta T_2 \omega_2$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrMechLoss	Total power loss	$\dot{W}_{loss} = -(P_t + P_d) + P_s$ $P_t = \eta T_d \omega_d + \eta T_1 \omega_1 + \eta T_2 \omega_2$
		PwrDampLoss	Power loss due to damping	$P_d = -(b_1\|\omega_1\| + b_2\|\omega_2\| + b_d\|\omega_d\|)$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	PwrStoredShft	Rate change of stored internal energy	$P_s = -(\omega_1\dot{\omega}_1 J_1 + \omega_2\dot{\omega}_2 J_2 + \omega_d\dot{\omega}_d J_d)$

Dynamics

The Open Differential block implements these differential equations to represent the mechanical dynamic response for the crown gear, left axle, and right axle.

Mechanical Dynamic Response	Differential Equation
Crown Gear	$\dot{\omega}_d J_d = \eta T_d - \omega_d b_d - T_i$
Left Axle	$\dot{\omega}_1 J_1 = \eta T_1 - \omega_1 b_1 - T_{i1}$
Right Axle	$\dot{\omega}_2 J_2 = \eta T_2 - \omega_2 b_2 - T_{i2}$

The Open Differential block assumes rigid coupling between the crown gear and axles. These constraint equations apply.

$$\eta T_1 = \eta T_2 = \frac{N}{2} T_i$$

$$\omega_d = \frac{N}{2}(\omega_1 + \omega_2)$$

The equations use these variables.

N	Carrier-to-driveshaft gear ratio
J_d	Rotational inertia of the crown gear assembly
b_d	Crown gear linear viscous damping
ω_d	Driveshaft angular speed
η	Differential efficiency
J_1	Axle 1 rotational inertia
b_1	Axle 1 linear viscous damping
ω_1	Axle 1 speed
J_2	Axle 2 rotational inertia
b_2	Axle 2 linear viscous damping
ω_2	Axle 2 angular speed
T_d	Driveshaft torque
T_1	Axle 1 torque
T_2	Axle 2 torque
T_i	Driveshaft internal resistance torque
T_{i1}	Axle 1 internal resistance torque
T_{i2}	Axle 2 internal resistance torque

Ports

Inputs

DriveshftTrq — Torque
scalar

Applied input torque, typically from the engine crankshaft, in N·m.

Axl1Trq — Torque
scalar

Axle 1 torque, T_1, in N·m.

Axl2Trq — Torque
scalar

Axle 2 torque, T_2, in N·m.

Temp — Temperature
scalar

Temperature, in K.

Dependencies

To enable this port:

- Set **Efficiency factors** to `Driveshaft torque, speed and temperature`.
- Select **Input temperature**.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal			Description	Units
Driveshft	DriveshftTrq		Driveshaft torque	N·m
	DriveshftSpd		Driveshaft speed	rad/s
Axl1	Axl1Trq		Axle 1 torque	N·m
	Axl1Spd		Axle 1 speed	rad/s
Axl2	Axl2Trq		Axle 2 torque	N·m
	Axl2Spd		Axle 2 speed	rad/s
PwrInfo	PwrTrnsfrd	PwrDriveshft	Mechanical power from driveshaft	W
		PwrAxl1	Mechanical power from axle 1	W
		PwrAxl2	Mechanical power from axle 2	W
	PwrTrnsfrd	PwrMechLoss	Total power loss	W
		PwrDampLoss	Power loss due to damping	W
	PwrStored	PwrStoredShft	Rate change of stored internal energy	W

DriveshftSpd — Angular speed
scalar

Driveshaft angular speed, ω_d, in rad/s.

Axl1Spd — Angular speed
scalar

Axle 1 angular speed, ω_1, in rad/s.

Axl2Spd — Angular speed
scalar

Axle 2 angular speed, ω_2, in rad/s.

Parameters

Block Options

Efficiency factors — Specify configuration
Constant (default) | Driveshaft torque, speed and temperature

To account for the block efficiency, use the **Efficiency factors** parameter. This table summarizes the block implementation for each setting.

Setting	Implementation
Constant	Constant efficiency that you can set with the **Constant efficiency factor, eta** parameter.
Driveshaft torque, temperature and speed	Efficiency as a function of base gear input torque, air temperature, and driveshaft speed. Use these parameters to specify the lookup table and breakpoints: • **Efficiency lookup table, eta_tbl** • **Efficiency torque breakpoints, Trq_bpts** • **Efficiency speed breakpoints, omega_bpts** • **Efficiency temperature breakpoints, Temp_bpts** For the air temperature, you can either: • Select **Input temperature** to create an input port. • Set a **Ambient temperature, Tamb** parameter value. To select the interpolation method, use the **Interpolation method** parameter. For more information, see "Interpolation Methods" (Simulink).

Interpolation method — Method
Flat | Nearest | Linear point-slope | Linear Lagrange | Cubic spline

For more information, see "Interpolation Methods" (Simulink).

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Input temperature — Create input port
off (default) | on

Select to create input port Temp for the temperature.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Crown wheel (ring gear) located — Specify crown wheel connection
To the left of center-line (default) | To the right of center-line

Specify the crown wheel connection to the driveshaft.

`Carrier to drive shaft ratio, Ndiff` **— Ratio**
4 (default) | scalar

Carrier-to-driveshaft gear ratio, *N*, dimensionless.

`Carrier inertia, Jd` **— Inertia**
.1 (default) | scalar

Rotational inertia of the crown gear assembly, J_d, in kg·m^2. You can include the driveshaft inertia.

`Carrier damping, bd` **— Damping**
1e-3 (default) | scalar

Crown gear linear viscous damping, b_d, in N·m·s/rad.

`Axle 1 inertia, Jw1` **— Inertia**
.1 (default) | scalar

Axle 1 rotational inertia, J_1, in kg·m^2.

`Axle 1 damping, bw1` **— Damping**
1e-3 (default) | scalar

Axle 1 linear viscous damping, b_1, in N·m·s/rad.

`Axle 2 inertia, Jw2` **— Inertia**
.1 (default) | scalar

Axle 2 rotational inertia, J_2, in kg·m^2.

`Axle 2 damping, bw2` **— Damping**
1e-3 (default) | scalar

Axle 2 linear viscous damping, b_2, in N·m·s/rad.

`Axle 1 initial velocity, omegaw1o` **— Angular velocity**
0 (default) | scalar

Axle 1 initial velocity, ω_{o1}, in rad/s.

`Axle 2 initial velocity, omegaw2o` **— Angular velocity**
0 (default) | scalar

Axle 2 initial velocity, ω_{o2}, in rad/s.

Efficiency

`Constant efficiency factor, eta` **— Efficiency**
1 (default) | scalar

Constant efficiency, η.

Dependencies

To enable this parameter, set **Efficiency factors** to `Constant`.

Efficiency lookup table, eta_tbl — Lookup table
M-by-N-by-L array

Dimensionless array of values for efficiency as a function of:

- M input torques
- N input speed
- L air temperatures

Each value specifies the efficiency for a specific combination of torque, speed, and temperature. The array size must match the dimensions defined by the torque, speed, and temperature breakpoint vectors.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Efficiency torque breakpoints, Trq_bpts — Torque breakpoints
[25, 50, 75, 100, 150, 200, 250] (default) | 1-by-M vector

Vector of input torque, breakpoints for efficiency, in N·m.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Efficiency speed breakpoints, omega_bpts — Speed breakpoints
[52.4 78.5 105 131 157 183 209 262 314 419 524] (default) | 1-by-N vector

Vector of speed, breakpoints for efficiency, in rad/s.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Efficiency temperature breakpoints, Temp_bpts — Temperature breakpoints
[290 358] (default) | 1-by-L vector

Vector of ambient temperature breakpoints for efficiency, in K.

Dependencies

To enable this parameter, set **Efficiency factors** to Driveshaft torque, speed and temperature.

Ambient temperature, Tamb — Ambient temperature
297.15 (default) | scalar

Ambient air temperature, T_{air}, in K.

Dependencies

To enable this parameter:

- Set **Efficiency factors** to `Driveshaft torque, speed and temperature`.
- Clear **Input temperature**.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Limited Slip Differential

Introduced in R2017a

Ideal Fixed Gear Transmission

Ideal fixed gear transmission without clutch or synchronization

Library: Powertrain Blockset / Transmission / Transmission Systems
Vehicle Dynamics Blockset / Powertrain / Transmission
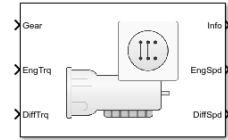
Description

The Ideal Fixed Gear Transmission implements an idealized fixed-gear transmission without a clutch or synchronization. Use the block to model the overall gear ratio and power loss when you do not need a detailed transmission model, for example, in component-sizing, fuel economy, and emission studies. The block implements a transmission model with minimal parameterization or computational cost.

To specify the block efficiency calculation, for **Efficiency factors**, select either of these options.

Setting	Block Implementation
`Gear only`	Efficiency determined from a 1D lookup table that is a function of the gear.
`Gear, input torque, input speed, and temperature`	Efficiency determined from a 4D lookup table that is a function of: • Gear • Input torque • Input speed • Oil temperature

The block uses this equation to determine the transmission dynamics:

$$\dot{\omega}_i \frac{J_N}{N^2} = \eta_N \left(\frac{T_o}{N} + T_i \right) - \frac{\omega_i}{N^2} b_N$$

$$\omega_i = N\omega_o$$

The block filters the gear command signal:

$$\frac{G}{G_{cmd}}(s) = \frac{1}{\tau_s s + 1}$$

Neutral Gear

When **Initial gear number, G_o** is equal to 0, the initial gear is neutral. The block uses these parameters to decouple the input flywheel from the downstream gearing.

• **Initial input velocity, omega_o**
• **Initial neutral input velocity, omegainN_o**

The block uses these equations for the neutral gear speed and flywheel.

$$\dot{\omega}_{neutral}\frac{J_N}{N^2} = \eta_N\frac{T_o}{N} - \frac{\omega_{neutral}}{N^2}b_N$$

$$\omega_{neutral} = N\omega_o$$

$$\dot{\omega}_1 J_F = \eta_{@N=0}T_i - b_{@N=0}\omega_i$$

$$J_F = J_{@N=1} - J_{@N=0}$$

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Varia ble	Equations
PwrIn fo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrEng	Engine power	P_{eng}	$\omega_i T_i$
		PwrDif frntl	Differential power	P_{diff}	$\omega_o T_o$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrEff Loss	Mechanical power loss	$P_{effloss}$	$\omega_o T_o(\eta_N - 1)$
		PwrDam pLoss	Mechanical damping loss	$P_{damploss}$	For G=0: $-\dfrac{b_N\omega_i^2}{\left\|N^2\right\|}$ For $G \neq 0$: $-b_N\omega_i^2 - \dfrac{b_N\omega_{neutral}^2}{\left\|N^2\right\|}$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	PwrSto redTra ns	Rate change in rotational kinetic energy	P_{str}	For G=0: $\dfrac{J_N}{N^2}\dot{\omega}_i\omega_i$ For $G \neq 0$: $J_F\dot{\omega}_i\omega_i + \dfrac{J_N}{N^2}\dot{\omega}_{neutral}\omega_{neutral}$

The equations use these variables.

b_N	Engaged gear viscous damping
J_N	Engaged gear rotational inertia
J_F	Flywheel rotational inertia
η_N	Engaged gear efficiency
G	Engaged gear number
G_{cmd}	Gear number to engage
N	Engaged gear ratio

T_i	Applied input torque, typically from the engine crankshaft or dual mass flywheel damper
T_o	Applied load torque, typically from the differential or drive shaft
ω_o	Initial input drive shaft rotational velocity
ω_i, $\acute{\omega}_i$	Applied drive shaft angular speed and acceleration
ω_{No}	Initial neutral gear input rotational velocity
$\omega_{neutral}$	Neutral gear drive shaft rotational velocity
τ_s	Shift time constant

Ports

Inputs

`Gear` — Gear number to engage
`scalar`

Integer value of gear number to engage, G_{cmd}.

`EngTrq` — Applied input torque
`scalar`

Applied input torque, T_i, typically from the engine crankshaft or dual mass flywheel damper, in N·m.

`DiffTrq` — Applied load torque
`scalar`

Applied load torque, T_o, typically from the differential, in N·m.

`Temp` — Oil temperature
`scalar`

Oil temperature, in K. To determine the efficiency, the block uses a 4D lookup table that is a function of:

- Gear
- Input torque
- Input speed
- Oil temperature

Dependencies

To enable this port, set **Efficiency factors** to `Gear, input torque, input speed, and temperature`.

Output

`Info` — Bus signal
`bus`

Bus signal containing these block calculations.

Signal			Description	Variable	Units
Eng	EngTrq		Applied input torque, typically from the engine crankshaft or dual mass flywheel damper	T_i	N·m
	EngSpd		Applied drive shaft angular speed input	ω_i	rad/s
Diff	DiffTrq		Applied load torque, typically from the differential	T_o	N·m
	DiffSpd		Drive shaft angular speed output	ω_o	rad/s
Trans	TransSpdRatio		Input to output speed ratio at time t	$\Phi(t)$	N/A
	TransEta		Ratio of output power to input power	η_N	N/A
	TransGearCmd		Commanded gear	N_{cmd}	N/A
	TransGear		Engaged gear	N	N/A
PwrInfo	PwrTrnsfrd	PwrEng	Engine power	P_{eng}	W
		PwrDiffrntl	Differential power	P_{diff}	W
	PwrNotTrnsfrd	PwrEffLoss	Mechanical power loss	$P_{effloss}$	W
		PwrDampLoss	Mechanical damping loss	$P_{damploss}$	W
	PwrStored	PwrStoredTrans	Rate change in rotational kinetic energy	P_{str}	W

EngSpd — Angular speed
scalar

Applied drive shaft angular speed input, ω_i, in rad/s.

DiffSpd — Angular speed
scalar

Drive shaft angular speed output, ω_o, in rad/s.

2-59

Parameters

Efficiency factors — Specify efficiency calculation
Gear only (default) | Gear, input torque, input speed, and temperature

To specify the block efficiency calculation, for **Efficiency factors**, select either of these options.

Setting	Block Implementation
Gear only	Efficiency determined from a 1D lookup table that is a function of the gear.
Gear, input torque, input speed, and temperature	Efficiency determined from a 4D lookup table that is a function of: • Gear • Input torque • Input speed • Oil temperature

Dependencies

Setting Parameter To	Enables
Gear only	**Efficiency vector, eta**
Gear, input torque, input speed, and temperature	**Efficiency torque breakpoints, Trq_bpts** **Efficiency speed breakpoints, omega_bpts** **Efficiency temperature breakpoints, Temp_bpts** **Efficiency lookup table, eta_tbl**

Gear property interpolation method — Interpolation
Nearest (default) | Linear | Flat | Cubic spline

Method that the block uses to switch the gear ratio during gear shifting.

Transmission

Gear number vector, G — Specify number of transmission speeds
[-1,0,1,2,3,4,5] (default) | vector

Vector of integer gear commands used to specify the number of transmission speeds. Neutral gear is 0. For example, you can set these parameter values.

To Specify	Set Gear number, G To
Four transmission speeds, including neutral	[0,1,2,3,4]
Three transmission speeds, including neutral and reverse	[-1,0,1,2,3]
Five transmission speeds, including neutral and reverse	[-1,0,1,2,3,4,5]

Vector dimensions for the **Gear number vector**, **Gear ratio vector**, **Inertia vector**, **Damping vector**, and **Efficiency vector** parameters must be equal.

Efficiency torque breakpoints, Trq_bpts — Breakpoints
[25,50,75,100,150,200,250] (default) | vector

Torque breakpoints for efficiency table.

Dependencies

To enable this parameter, set **Efficiency factors** to Gear, input torque, input speed, and temperature.

Efficiency speed breakpoints, omega_bpts — Breakpoints
[52.4 78.5 105 131 157 183 209 262 314 419 524] (default) | vector

Speed breakpoints for efficiency table.

Dependencies

To enable this parameter, set **Efficiency factors** to Gear, input torque, input speed, and temperature.

Efficiency temperature breakpoints, Temp_bpts — Breakpoints
[313 358] (default) | vector

Temperature breakpoints for efficiency table.

Dependencies

To enable this parameter, set **Efficiency factors** to Gear, input torque, input speed, and temperature.

Gear ratio vector, N — Ratio of input speed to output speed
[-4.47,4.47,4.47,2.47,1.47,1,0.8] (default) | vector

Vector of gear ratios (that is, input speed to output speed) with indices corresponding to the ratios specified in **Gear number, G**. For neutral, set the gear ratio to 1. For example, you can set these parameter values.

To Specify Gear Ratios For	Set Gear number, G To	Set Gear ratio, N To
Four transmission speeds, including neutral	[0,1,2,3,4]	[1,4.47,2.47,1.47,1]
Five transmission speeds, including neutral and reverse	[-1,0,1,2,3,4,5]	[-4.47,1,4.47,2.47, 1.47,1,0.8]

Vector dimensions for the **Gear number vector**, **Gear ratio vector**, **Inertia vector**, **Damping vector**, and **Efficiency vector** parameters must be equal.

Inertia vector, Jout — Gear rotational inertia
[0.128 0.01 0.128 0.1 0.062 0.028 0.01] (default) | vector

Vector of gear rotational inertias, J_N, with indices corresponding to the inertias specified in **Gear number, G**, in kg*m^2. For example, you can set these parameter values.

To Specify Inertia For	Set Gear number, G To	Set Inertia, J To
Four gears, including neutral	`[0,1,2,3,4]`	`[0.01,2.28,2.04, 0.32,0.028]`
Inertia for five gears, including reverse and neutral	`[-1,0,1,2,3,4,5]`	`[2.28,0.01,2.28, 2.04,0.32,0.028,0.01]`

Vector dimensions for the **Gear number vector**, **Gear ratio vector**, **Inertia vector**, **Damping vector**, and **Efficiency vector** parameters must be equal.

Damping vector, bout — Gear viscous damping coefficient
`[.003 .001 .003 .0025 .002 .001 .001]` (default) | `vector`

Vector of gear viscous damping coefficients, b_N, with indices corresponding to the coefficients specified in **Gear number, G**, in N·m·s/rad. For example, you can set these parameter values.

To Specify Damping For	Set Gear number, G To	Set Damping, b To
Four gears, including neutral	`[0,1,2,3,4]`	`[0.001,0.003, 0.0025,0.002,0.001]`
Five gears, including reverse and neutral	`[-1,0,1,2,3,4,5]`	`[0.003,0.001, 0.003,0.0025, 0.002,0.001,0.001]`

Vector dimensions for the **Gear number vector**, **Gear ratio vector**, **Inertia vector**, **Damping vector**, and **Efficiency vector** parameters must be equal.

Efficiency vector, eta — Gear efficiency
`[0.9,0.9,0.9,0.9,0.9,0.95,0.95]` (default) | `vector`

Vector of gear mechanical efficiency, η_N, with indices corresponding to the efficiencies specified in **Gear number, G**. For example, you can set these parameter values.

To Specify Efficiency For	Set Gear number, G To	Set Efficiency, eta To
Four gears, including neutral	`[0,1,2,3,4]`	`[0.9,0.9,0.9,0.9,0.95]`
Five gears, including reverse and neutral	`[-1,0,1,2,3,4,5]`	`[0.9,0.9,0.9, 0.9,0.9,0.95,0.95]`

Vector dimensions for the **Gear number vector**, **Gear ratio vector**, **Inertia vector**, **Damping vector**, and **Efficiency vector** parameters must be equal.

Dependencies

To enable this parameter, set **Efficiency factors** to `Gear only`.

Efficiency lookup table, eta_tbl — Gear efficiency
`array`

Table of gear mechanical efficiency, η_N as a function of gear, input torque, input speed, and temperature.

Dependencies

To enable this parameter, set **Efficiency factors** to `Gear, input torque, input speed, and temperature`.

`Initial gear number, G_o` — Gear
`0` (default) | `scalar`

Initial gear number, G_o, dimensionless.

`Initial output velocity, omega_o` — Output speed
`0` (default) | `scalar`

Transmission initial output rotational velocity, ω_o, in rad/s.

`Initial neutral input velocity, omegainN_o` — Neutral gear input speed
`0` (default) | `scalar`

Initial neutral gear input rotational velocity, ω_{No}, in rad/s.

`Shift time constant, tau_s` — Time
`.01` (default) | `scalar`

Shift time constant, τ_s, in s.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Limited Slip Differential | Open Differential

Introduced in R2017a

Wheel and Tire Blocks

Longitudinal Wheel

Longitudinal wheel with disc, drum, or mapped brake

Library: Powertrain Blockset / Drivetrain / Wheels

                                       Vehicle Dynamics Blockset / Wheels and Tires
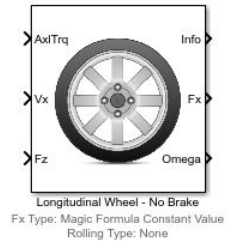
Description

The Longitudinal Wheel block implements the longitudinal behavior of an ideal wheel. You can specify the longitudinal force and rolling resistance calculation method, and brake type. Use the block in driveline and longitudinal vehicle simulations where low frequency tire-road and braking forces are required to determine vehicle acceleration, braking, and wheel-rolling resistance. For example, you can use the block to determine the torque and power requirements for a specified drive cycle or braking event. The block is not suitable for applications that require combined lateral slip.

There are four types of Longitudinal Wheel blocks. Each block implements a different brake type.

Block Name	Brake Type Setting	Brake Implementation
Longitudinal Wheel - No Brake	`None`	None
Longitudinal Wheel - Disc Brake	`Disc`	Brake that converts the brake cylinder pressure into a braking force.
Longitudinal Wheel - Drum Brake	`Drum`	Simplex drum brake that converts the applied force and brake geometry into a net braking torque.
Longitudinal Wheel - Mapped Brake	`Mapped`	Lookup table that is a function of the wheel speed and applied brake pressure.

The block models longitudinal force as a function of wheel slip relative to the road surface. To calculate the longitudinal force, specify one of these **Longitudinal Force** parameters.

Setting	Block Implementation
`Magic Formula constant value`	Magic Formula with constant coefficient for stiffness, shape, peak, and curvature.
`Magic Formula pure longitudinal slip`	Magic Formula with load-dependent coefficients that implement equations 4.E9 through 4.E18 in *Tire and Vehicle Dynamics*.
`Mapped force`	Lookup table that is a function of the normal force and wheel slip ratio.

To calculate the rolling resistance torque, specify one of these **Rolling Resistance** parameters.

Setting	Block Implementation
None	None
Pressure and velocity	Method in *Stepwise Coastdown Methodology for Measuring Tire Rolling Resistance*. The rolling resistance is a function of tire pressure, normal force, and velocity.
ISO 28580	Method specified in ISO 28580:2018, *Passenger car, truck and bus tyre rolling resistance measurement method — Single point test and correlation of measurement results*.
Magic Formula	Magic formula equations from 4.E70 in *Tire and Vehicle Dynamics*. The magic formula is an empirical equation based on fitting coefficients.
Mapped torque	Lookup table that is a function of the normal force and spin axis longitudinal velocity.

To calculate vertical motion, specify one of these **Vertical Motion** parameters.

Setting	Block Implementation
None	Block passes the applied chassis forces directly through to the rolling resistance and longitudinal force calculations.
Mapped stiffness and damping	Vertical motion depends on wheel stiffness and damping. Stiffness is a function of tire sidewall displacement and pressure. Damping is a function of tire sidewall velocity and pressure.

Rotational Wheel Dynamics

The block calculates the inertial response of the wheel subject to:

- Axle losses
- Brake and drive torque
- Tire rolling resistance
- Ground contact through the tire-road interface

The input torque is the summation of the applied axle torque, braking torque, and moment arising from the combined tire torque.

$$T_i = T_a - T_b + T_d$$

For the moment arising from the combined tire torque, the block implements tractive wheel forces and rolling resistance with first order dynamics. The rolling resistance has a time constant parameterized in terms of a relaxation length.

$$T_d(s) = \frac{1}{\frac{|\omega|R_e}{L_e}s + 1}(F_x R_e + M_y)$$

To calculate the rolling resistance torque, you can specify one of these **Rolling Resistance** parameters.

Setting	Block Implementation
None	Block sets rolling resistance, M_y, to zero.
Pressure and velocity	Block uses the method in SAE *Stepwise Coastdown Methodology for Measuring Tire Rolling Resistance*. The rolling resistance is a function of tire pressure, normal force, and velocity. Specifically, $$M_y = R_e\{a + b|V_x| + cV_x^2\}\{F_z{}^\beta p_i{}^\alpha\}\tanh(4V_x)$$
ISO 28580	Block uses the method specified in ISO 28580:2018, *Passenger car, truck and bus tyre rolling resistance measurement method — Single point test and correlation of measurement results*. The method accounts for normal load, parasitic loss, and thermal corrections from test conditions. Specifically, $$M_y = R_e(\frac{F_z C_r}{1 + K_t(T_{amb} - T_{meas})} - F_{pl})\tanh(\omega)$$
Magic Formula	Block calculates the rolling resistance, M_y, using the Magic Formula equations from 4.E70 in *Tire and Vehicle Dynamics*. The magic formula is an empirical equation based on fitting coefficients.
Mapped torque	For the rolling resistance, M_y, the block uses a lookup table that is a function of the normal force and spin axis longitudinal velocity.

If the brakes are enabled, the block determines the braking locked or unlocked condition based on an idealized dry clutch friction model. Based on the lock-up condition, the block implements these friction and dynamic models.

If	Lock-Up Condition	Friction Model	Dynamic Model
$\omega \neq 0$ or $T_S < \|T_i + T_f - \omega b\|$	Unlocked	$T_f = T_k$ where, $T_k = F_c R_{eff} \mu_k \tanh[4(-\omega_d)]$ $T_s = F_c R_{eff} \mu_s$ $R_{eff} = \dfrac{2(R_o{}^3 - R_i{}^3)}{3(R_o{}^2 - R_i{}^2)}$	$\dot{\omega}J = -\omega b + T_i + T_o$
$\omega = 0$ and $T_S \geq \|T_i + T_f - \omega b\|$	Locked	$T_f = T_s$	$\omega = 0$

The equations use these variables.

ω	Wheel angular velocity
a	Velocity-independent force component
b	Linear velocity force component
c	Quadratic velocity force component
L_e	Tire relaxation length
J	Moment of inertia

M_y	Rolling resistance torque
T_a	Applied axle torque
T_b	Braking torque
T_d	Combined tire torque
T_f	Frictional torque
T_i	Net input torque
T_k	Kinetic frictional torque
T_o	Net output torque
T_s	Static frictional torque
F_c	Applied clutch force
F_x	Longitudinal force developed by the tire road interface due to slip
R_{eff}	Effective clutch radius
R_o	Annular disk outer radius
R_i	Annular disk inner radius
R_e	Effective tire radius while under load and for a given pressure
V_x	Longitudinal axle velocity
F_z	Vehicle normal force
C_r	Rolling resistance constant
T_{amb}	Ambient temperature
T_{meas}	Measured temperature for rolling resistance constant
F_{pl}	Parasitic force loss
K_t	Thermal correction factor
α	Tire pressure exponent
β	Normal force exponent
p_i	Tire pressure
μ_s	Coefficient of static friction
μ_k	Coefficient of kinetic friction

Brakes

Disc

If you specify the **Brake Type** parameter `Disc`, the block implements a disc brake. This figure shows the side and front views of a disc brake.

A disc brake converts brake cylinder pressure from the brake cylinder into force. The disc brake applies the force at the brake pad mean radius.

The block uses these equations to calculate brake torque for the disc brake.

$$T = \begin{cases} \dfrac{\mu P \pi B_a{}^2 R_m N_{pads}}{4} & \text{when } N \neq 0 \\[2ex] \dfrac{\mu_{static} P \pi B_a{}^2 R_m N_{pads}}{4} & \text{when } N = 0 \end{cases}$$

$$Rm = \frac{Ro + Ri}{2}$$

The equations use these variables.

T	Brake torque
P	Applied brake pressure
N	Wheel speed
N_{pads}	Number of brake pads in disc brake assembly
μ_{static}	Disc pad-rotor coefficient of static friction
μ	Disc pad-rotor coefficient of kinetic friction
B_a	Brake actuator bore diameter
R_m	Mean radius of brake pad force application on brake rotor
R_o	Outer radius of brake pad

| R_i | Inner radius of brake pad |

Drum

If you specify the **Brake Type** parameter `Drum`, the block implements a static (steady-state) simplex drum brake. A simplex drum brake consists of a single two-sided hydraulic actuator and two brake shoes. The brake shoes do not share a common hinge pin.

The simplex drum brake model uses the applied force and brake geometry to calculate a net torque for each brake shoe. The drum model assumes that the actuators and shoe geometry are symmetrical for both sides, allowing a single set of geometry and friction parameters to be used for both shoes.

The block implements equations that are derived from these equations in *Fundamentals of Machine Elements*.

$$T_{rshoe} = \left(\frac{\pi \mu c r (\cos\theta_2 - \cos\theta_1) B_a 2}{2\mu \left(2r \left(\cos\theta_2 - \cos\theta_1 \right) + a \left(\cos^2\theta_2 - \cos^2\theta_1 \right) \right) + ar(2\theta_1 - 2\theta_2 + \sin2\theta_2 - \sin2\theta_1)} \right) P$$

$$T_{lshoe} = \left(\frac{\pi \mu c r (\cos\theta_2 - \cos\theta_1) B_a 2}{-2\mu \left(2r \left(\cos\theta_2 - \cos\theta_1 \right) + a \left(\cos^2\theta_2 - \cos^2\theta_1 \right) \right) + ar(2\theta_1 - 2\theta_2 + \sin2\theta_2 - \sin2\theta_1)} \right) P$$

$$T = \begin{cases} T_{rshoe} + T_{lshoe} & \text{when } N \neq 0 \\ (T_{rshoe} + T_{lshoe}) \frac{\mu_{static}}{\mu} & \text{when } N = 0 \end{cases}$$

Rotation

The equations use these variables.

T	Brake torque
P	Applied brake pressure
N	Wheel speed

μ_{static}	Disc pad-rotor coefficient of static friction
μ	Disc pad-rotor coefficient of kinetic friction
T_{rshoe}	Right shoe brake torque
T_{lshoe}	Left shoe brake torque
a	Distance from drum center to shoe hinge pin center
c	Distance from shoe hinge pin center to brake actuator connection on brake shoe
r	Drum internal radius
B_a	Brake actuator bore diameter
Θ_1	Angle from shoe hinge pin center to start of brake pad material on shoe
Θ_2	Angle from shoe hinge pin center to end of brake pad material on shoe

Mapped

If you specify the **Brake Type** parameter `Mapped`, the block uses a lookup table to determine the brake torque.

$$T = \begin{cases} f_{brake}(P, N) & \text{when } N \neq 0 \\ \left(\frac{\mu_{static}}{\mu}\right) f_{brake}(P, N) & \text{when } N = 0 \end{cases}$$

The equations use these variables.

T	Brake torque
$f_{brake}(P, N)$	Brake torque lookup table
P	Applied brake pressure
N	Wheel speed
μ_{static}	Friction coefficient of drum pad-face interface under static conditions
μ	Friction coefficient of disc pad-rotor interface

The lookup table for the brake torque, $f_{brake}(P, N)$, is a function of applied brake pressure and wheel speed, where:

- T is brake torque, in N·m.
- P is applied brake pressure, in bar.
- N is wheel speed, in rpm.

Longitudinal Force

To model the Longitudinal Wheel block longitudinal forces, you can use the Magic Formula. The model provides a steady-state *tire characteristic function* $F_x = f(\kappa, F_z)$, the longitudinal force F_x on the tire, based on:

- Vertical load F_z
- Wheel slip κ

The Magic Formula model uses these variables.

Ω	Wheel angular velocity
r_w	Wheel radius
V_x	Wheel hub longitudinal velocity
$r_w \Omega$	Tire tread longitudinal velocity
$V_{sx} = r_w \Omega - V_x$	Wheel slip velocity
$\kappa = V_{sx}/|V_x|$	Wheel slip
F_z, F_{z0}	Vertical load and nominal vertical load on tire
$F_x = f(\kappa, F_z)$	Longitudinal force exerted on the tire at the contact point. Also a characteristic function f of the tire.

Magic Formula Constant Value

If you set **Longitudinal Force** to `Magic Formula constant value`, the block implements the Magic Formula as a specific form of the tire characteristic function, characterized by four dimensionless coefficients (B, C, D, E), or stiffness, shape, peak, and curvature:

$$F_x = f(\kappa, F_z) \ = F_z D \sin\left(C \tan^{-1}\left[\ \{ B\kappa - E[B\kappa - \tan^{-1}(B\kappa) \ ] \} \ \right] \right)$$

The slope of f at $\kappa = 0$ is $BCD \cdot F_z$.

The coefficients are based on empirical tire data. These values are typical sets of constant Magic Formula coefficients for common road conditions.

Surface	B	C	D	E
Dry tarmac	10	1.9	1	0.97
Wet tarmac	12	2.3	0.82	1
Snow	5	2	0.3	1
Ice	4	2	0.1	1

Magic Formula Pure Longitudinal Slip

If you set **Longitudinal Force** to `Magic Formula pure longitudinal slip`, the block implements a more general Magic Formula using dimensionless coefficients that are functions of the tire load. The block implements the longitudinal force equations in Chapter 4 of *Tire and Vehicle Dynamics*, including 4.E9 through 4.E18:

$$F_{x0} = D_x \sin\left(C_x \tan^{-1}\left[\ \{B_x \kappa_x - E_x[B_x \kappa_x - \tan^{-1}(B_x \kappa_x)\ ]\}\ ]\ \right) + S_{Vx}$$

where:

$$\kappa_x = \kappa + S_{Hx}$$

$$C_x = p_{Cx1} \lambda_{Cx}$$

$$D_x = \mu_x F_z \varsigma_1$$

$$\mu_x = (p_{Dx1} + p_{Dx2} df_z)(1 + p_{px3} dp_i + p_{px4} dp_i 2)(1 - p_{Dx3} \gamma^2) \lambda^*_{\mu x}$$

$$E_x = (p_{Ex1} + p_{Ex2} df_z + p_{Ex3} df_z 2)[1 - p_{Ex4} \text{sgn}(\kappa_x)]\ \lambda_{Ex}$$

$$K_{x\kappa} = F_z(p_{Kx1} + p_{Kx2} df_z) \exp(p_{Kx3} df_z)(1 + p_{px1} dp_i + p_{px2} dp_i 2)$$

$$B_x = K_{x\kappa}/(C_x D_x + \varepsilon_x)$$

$$S_{Hx} = p_{Hx1} + p_{Hx2} df_z$$

$$S_{Vx} = F_z \cdot (p_{Vx1} + p_{Vx2} df_z) \lambda_{Vx} \lambda'_{\mu x} \varsigma_1$$

S_{Hx} and S_{Vx} represent offsets to the slip and longitudinal force in the force-slip function, or horizontal and vertical offsets if the function is plotted as a curve. μ_x is the longitudinal load-dependent friction coefficient. ε_x is a small number inserted to prevent division by zero as F_z approaches zero.

Vertical Dynamics

If you select no vertical degrees-of-freedom by setting **Vertical Motion** to None, the block passes the applied chassis forces directly through to the rolling resistance and longitudinal force calculations.

If you set **Vertical Motion** to `Mapped stiffness and damping`, the vertical motion depends on wheel stiffness and damping. Stiffness is a function of tire sidewall displacement and pressure. Damping is a function of tire sidewall velocity and pressure.

$$Fztire(z, \dot{z}, P_{tire}) = F_{zk}(z, P_{tire}) + F_{zb}(\dot{z}, P_{tire})$$

The block determines the vertical response using this differential equation.

$$\ddot{z}m = Fztire - F_z - mg$$

When you disable the vertical degree-of-freedom, the input normal force from the vehicle passes directly to the longitudinal and rolling force calculations.

$$\ddot{z} = \dot{z} = m = 0$$
$$Fztire = mg$$

The block uses the wheel-fixed frame to resolve the vertical forces.
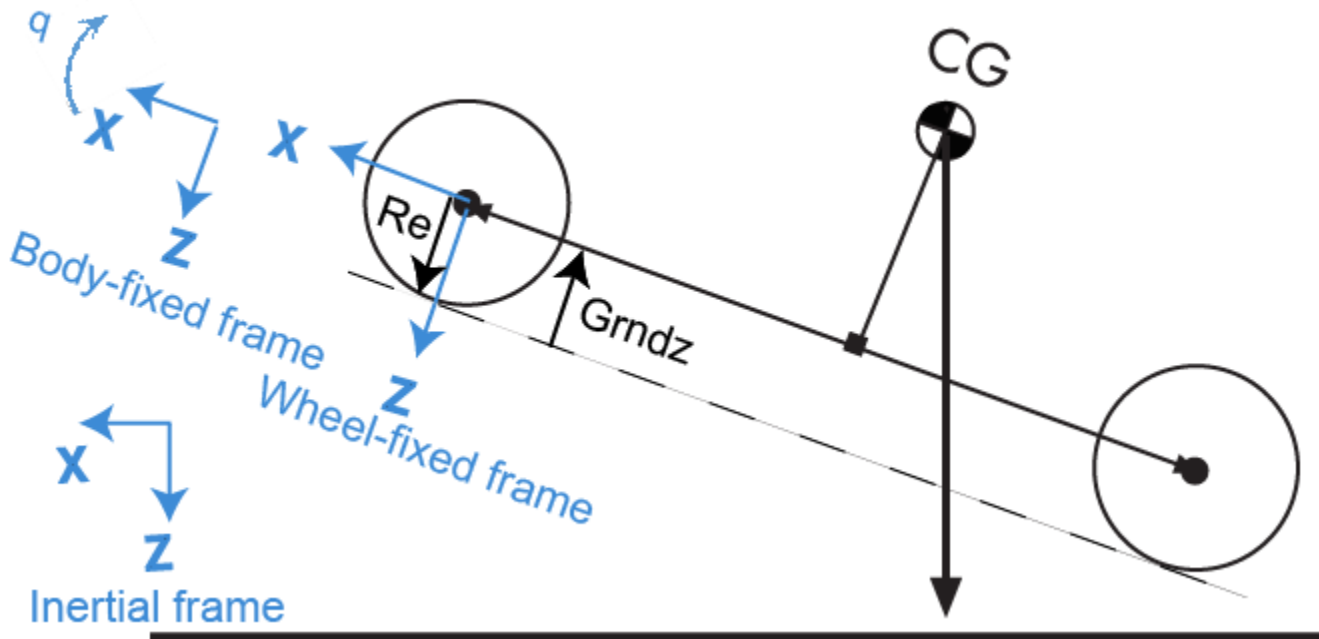
The equations use these variables.

Fztire	Tire normal force along the wheel-fixed *z*-axis
m	Axle mass
F_{zk}	Tire normal force due to wheel stiffness along the wheel-fixed *z*-axis
F_{zb}	Tire normal force due to wheel damping along the wheel-fixed *z*-axis
F_z	Suspension or vehicle normal force along the wheel-fixed *z*-axis
P_{Tire}	Tire pressure
$z, \dot{z}, \ddot{z}$	Tire displacement, velocity, and acceleration, respectively, along the wheel-fixed *z*-axis

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks	PwrRoad	Tractive power applied from the axle	$P_{road} = F_x V_x$
		PwrAxlTrq	External torque applied by the axle to the wheel	$P_T = T\omega$

Bus Signal			Description	Equations
	• Positive signals indicate flow into block • Negative signals indicate flow out of block	`PwrFz`	Vertical force applied to the wheel by the vehicle or suspension	$P_{Fz} = F_z \dot{z}$
	`PwrNotTrnsfrd` — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	`PwrSlip`	Tractive power loss	$P_\kappa = F_x V_x + (-F_{cp}R_e + M_y)\omega$
		`PwrMyRoll`	Rolling resistance power	$P_{My} = M_y \omega$
		`PwrMyBrk`	Braking power	$P_{brk} = M_{brk}\omega$
		`PwrMyb`	Rolling viscous damping loss	$P_b = -b\omega^2$
		`PwrFzDamp`	Vertical damping power	$P_{Fzb} = F_{zb}\dot{z}$
	`PwrStored` — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	`PwrStoredzdot`	Rate of change of vertical kinetic energy	$P_{\dot{z}} = m\ddot{z}\dot{z}$
		`PwrStoredq`	Rate of change of rotational kinetic energy	$P_\omega = I_{yy}\dot{\omega}\omega$
		`PwrStoredFsFzSprng`	Rate of change of stored sidewall potential energy	$P_{Fzk} = F_{zk}\dot{z}_x$
		`PwrStoredGrvty`	Rate of change of gravitational potential energy	$P_g = -mg\dot{Z}$

The equations use these variables.

ω	Wheel angular velocity
b	Linear velocity force component
F_x	Longitudinal force developed by the tire road interface due to slip
F_{cp}	Tire slip force at contact patch
F_z	Vehicle normal force
F_{zb}	Tire normal force due to wheel damping
F_{zk}	Tire normal force due to wheel stiffness
I_{yy}	Wheel rotational inertia
M_{brk}	Braking moment
M_y	Rolling resistance torque
R_e	Effective tire radius while under load and for a given pressure
T	Axle torque applied on wheel
V_x	Longitudinal axle velocity

$z, \dot{z}, \ddot{z}$	Tire displacement, velocity, and acceleration, respectively
ω	Wheel angular velocity
$\dot{Z}$	Vehicle vertical velocity along the vehicle-fixed z-axis

Ports

Input

BrkPrs — Brake pressure
scalar

Brake pressure, in Pa.

Dependencies

To enable this port, for the **Brake Type** parameter, specify one of these types:

- `Disc`
- `Drum`
- `Mapped`

AxlTrq — Axle torque
scalar

Axle torque, T_a, about wheel spin axis, in N·m.

Vx — Velocity
scalar

Axle longitudinal velocity along vehicle(body)-fixed x-axis, in m/s.

Fz — Normal force
scalar

Absolute value of suspension or vehicle normal force along body-fixed z-axis, in N.

Gnd — Ground displacement
scalar

Ground displacement, `Grndz`, along negative wheel-fixed z-axis, in m.
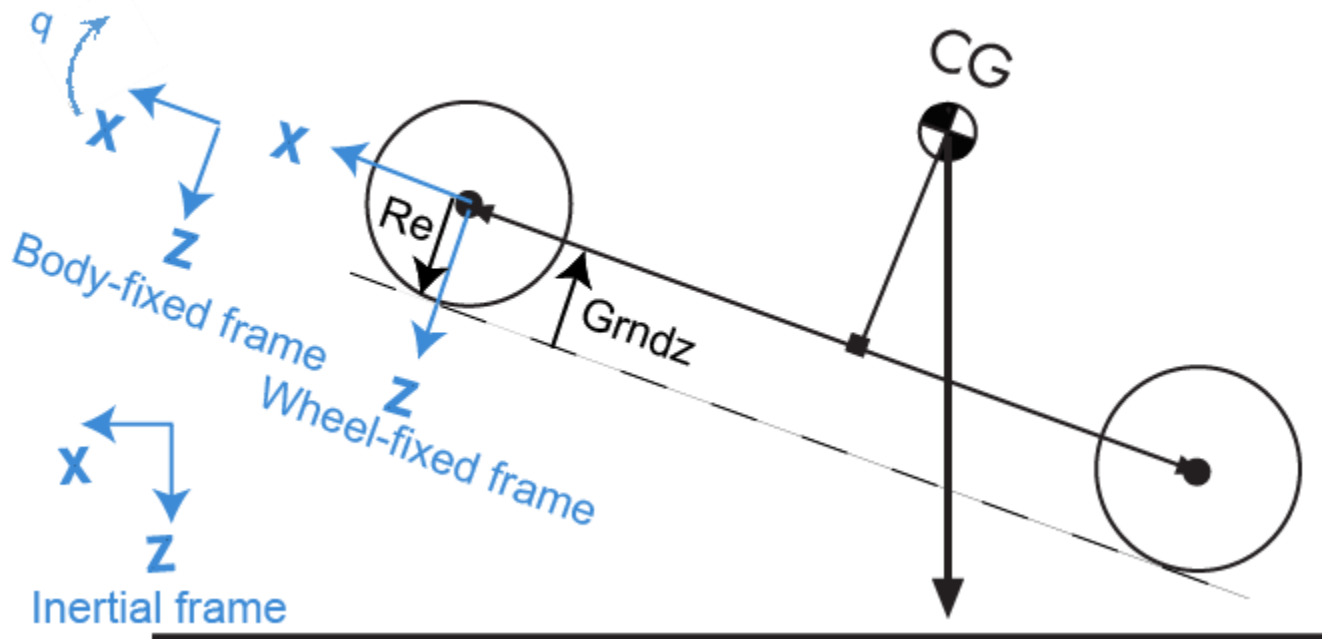
Dependencies

To create Gnd:

- Set **Vertical Motion** to Mapped stiffness and damping.
- On the **Vertical** pane, select **Input ground displacement**.

lam_mux — Friction scaling factor
scalar

Longitudinal friction scaling factor, dimensionless.

Dependencies

To enable this port, select **Input friction scale factor**.

TirePrs — Tire pressure
scalar

Tire pressure, in Pa.

Dependencies

To enable this port:

- Set one of these parameters:

    - **Longitudinal Force** to Magic Formula pure longitudinal slip.
    - **Rolling Resistance** to Pressure and velocity or Magic Formula.
    - **Vertical Motion** to Mapped stiffness and damping.

- On the **Wheel Dynamics** pane, select **Input tire pressure**.

Tamb — Ambient temperature
scalar

Ambient temperature, T_{amb}, in K.

Dependencies

To enable this port:

1   Set **Rolling Resistance** to ISO 28580.
2   On the **Rolling Resistance** pane, select to **Input ambient temperature**.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal	Description	Units
AxlTrq	Axle torque about body-fixed y-axis	N·m
Omega	Wheel angular velocity about body-fixed y-axis	rad/s
Omegadot	Wheel angular acceleration about body-fixed y-axis	rad/s^2
Fx	Longitudinal vehicle force along body-fixed x-axis	N
Fz	Vertical vehicle force along body-fixed z-axis	N
Fzb	Tire normal force due to wheel damping along the wheel-fixed z-axis	N
Fzk	Tire normal force due to wheel stiffness along the wheel-fixed z-axis	N
My	Rolling resistance torque about body-fixed y-axis	N·m
Myb	Rolling resistance torque due to damping about body-fixed y-axis	N·m
Kappa	Slip ratio	NA
Vx	Vehicle longitudinal velocity along body-fixed x-axis	m/s
Re	Wheel effective radius along wheel-fixed z-axis	m

Signal			Description	Units
BrkTrq			Brake torque about body-fixed y-axis	N·m
BrkPrs			Brake pressure	Pa
z			Wheel vertical deflection along wheel-fixed z-axis	m
zdot			Wheel vertical velocity along wheel-fixed z-axis	m/s
zddot			Wheel vertical acceleration along wheel-fixed z-axis	m/s^2
Gndz			Ground displacement along negative of wheel-fixed z-axis (positive input produces wheel lift)	m
GndFz			Vertical wheel force on ground along negative of wheel-fixed z-axis	N
TirePrs			Tire pressure	Pa
Fpatch			Tractive power applied from the axle	
PwrInfo	PwrTrnsfrd	PwrRoad	External torque applied by the axle to the wheel	W
		PwrAxlTrq	Vertical force applied to the wheel by the vehicle or suspension	W
		PwrFz	Tractive power loss	W
	PwrNotTrnsfrd	PwrSlip	Rolling resistance power	W
		PwrMyRoll	Braking power	W
		PwrMyBrk	Rolling viscous damping loss	W
		PwrMyb	Vertical damping power	W
		PwrFzDamp	Rate of change of vertical kinetic energy	W
	PwrStored	PwrStoredzdot	Rate of change of rotational kinetic energy	W
		PwrStoredq	Rate of change of stored sidewall potential energy	W
		PwrStoredFsFzSprng	Rate of change of gravitational potential energy	W
		PwrStoredGrvty	Tractive power applied from the axle	W

Fx — Longitudinal axle force
scalar

Longitudinal force acting on axle, along body-fixed *x*-axis, in N. Positive force acts to move the vehicle forward.

Omega — Wheel angular velocity
scalar

Wheel angular velocity, about body-fixed *y*-axis, in rad/s.

z — Wheel vertical deflection
scalar

Wheel vertical deflection along wheel-fixed *z*-axis, in m.

Dependencies

To enable this port, set **Vertical Motion** to Mapped stiffness and damping.

zdot — Wheel vertical velocity
scalar

Wheel vertical velocity along wheel-fixed *z*-axis, in m/s.

Dependencies

To enable this port, set **Vertical Motion** to Mapped stiffness and damping.

Parameters

Block Options

Longitudinal Force — Select type
Magic Formula constant value (default) | Magic Formula pure longitudinal slip | Mapped force

The block models longitudinal force as a function of wheel slip relative to the road surface. To calculate the longitudinal force, specify one of these **Longitudinal Force** parameters.

Setting	Block Implementation
Magic Formula constant value	Magic Formula with constant coefficient for stiffness, shape, peak, and curvature.
Magic Formula pure longitudinal slip	Magic Formula with load-dependent coefficients that implement equations 4.E9 through 4.E18 in *Tire and Vehicle Dynamics*.
Mapped force	Lookup table that is a function of the normal force and wheel slip ratio.

Dependencies

Selecting	Enables These Parameters
Magic Formula constant value	**Pure longitudinal peak factor, Dx**
	Pure longitudinal shape factor, Cx
	Pure longitudinal stiffness factor, Bx
	Pure longitudinal curvature factor, Ex

Selecting	Enables These Parameters
`Magic Formula pure longitudinal slip`	**Cfx shape factor, PCX1**
	Longitudinal friction at nominal normal load, PDX1
	Frictional variation with load, PDX2
	Frictional variation with camber, PDX3
	Longitudinal curvature at nominal normal load, PEX1
	Variation of curvature factor with load, PEX2
	Variation of curvature factor with square of load, PEX3
	Longitudinal curvature factor with slip, PEX4
	Longitudinal slip stiffness at nominal normal load, PKX1
	Variation of slip stiffness with load, PKX2
	Slip stiffness exponent factor, PKX3
	Horizontal shift in slip ratio at nominal normal load, PHX1
	Variation of horizontal slip ratio with load, PHX2
	Vertical shift in load at nominal normal load, PVX1
	Variation of vertical shift with load, PVX2
	Linear variation of longitudinal slip stiffness with tire pressure, PPX1
	Quadratic variation of longitudinal slip stiffness with tire pressure, PPX2
	Linear variation of peak longitudinal friction with tire pressure, PPX3
	Quadratic variation of peak longitudinal friction with tire pressure, PPX4
	Linear variation of longitudinal slip stiffness with tire pressure, PPX1
	Slip speed decay function scaling factor, lam_muV
	Brake slip stiffness scaling factor, lam_Kxkappa
	Longitudinal shape scaling factor, lam_Cx
	Longitudinal curvature scaling factor, lam_Ex

Selecting	Enables These Parameters
	Longitudinal horizontal shift scaling factor, lam_Hx
	Longitudinal vertical shift scaling factor, lam_Vx
Mapped force	**Slip ratio breakpoints, kappaFx**
	Normal force breakpoints, FzFx
	Longitudinal force map, FxMap

Rolling Resistance — Select type
None (default) | Pressure and velocity | ISO 28580 | Magic Formula | Mapped torque

To calculate the rolling resistance torque, specify one of these **Rolling Resistance** parameters.

Setting	Block Implementation
None	None
Pressure and velocity	Method in *Stepwise Coastdown Methodology for Measuring Tire Rolling Resistance*. The rolling resistance is a function of tire pressure, normal force, and velocity.
ISO 28580	Method specified in ISO 28580:2018, *Passenger car, truck and bus tyre rolling resistance measurement method — Single point test and correlation of measurement results*.
Magic Formula	Magic formula equations from 4.E70 in *Tire and Vehicle Dynamics*. The magic formula is an empirical equation based on fitting coefficients.
Mapped torque	Lookup table that is a function of the normal force and spin axis longitudinal velocity.

Dependencies

Selecting	Parameters
Pressure and velocity	**Velocity independent force coefficient, aMy**
	Linear velocity force component, bMy
	Quadratic velocity force component, cMy
	Tire pressure exponent, alphaMy
	Normal force exponent, betaMy

Selecting	Parameters
`ISO 28580`	**Parasitic losses force, Fpl**
	Rolling resistance constant, Cr
	Thermal correction factor, Kt
	Measured temperature, Tmeas
	Parasitic losses force, Fpl
	Ambient temperature, Tamb
`Magic Formula`	**Rolling resistance torque coefficient, QSY**
	Longitudinal force rolling resistance coefficient, QSY2
	Linear rotational speed rolling resistance coefficient, QSY3
	Quartic rotational speed rolling resistance coefficient, QSY4
	Camber squared rolling resistance torque, QSY5
	Load based camber squared rolling resistance torque, QSY6
	Normal load rolling resistance coefficient, QSY7
	Pressure load rolling resistance coefficient, QSY8
	Rolling resistance scaling factor, lam_My
`Mapped torque`	**Spin axis velocity breakpoints, VxMy**
	Normal force breakpoints, FzMy
	Rolling resistance torque map, MyMap

`Brake Type` — Select type
None | Disc | Drum | Mapped

There are four types of Longitudinal Wheel blocks. Each block implements a different brake type.

Block Name	Brake Type Setting	Brake Implementation
Longitudinal Wheel - No Brake	`None`	None
Longitudinal Wheel - Disc Brake	`Disc`	Brake that converts the brake cylinder pressure into a braking force.
Longitudinal Wheel - Drum Brake	`Drum`	Simplex drum brake that converts the applied force and brake geometry into a net braking torque.

Block Name	Brake Type Setting	Brake Implementation
Longitudinal Wheel - Mapped Brake	Mapped	Lookup table that is a function of the wheel speed and applied brake pressure.

Vertical Motion — Select type
None (default) | Mapped stiffness and damping

To calculate vertical motion, specify one of these **Vertical Motion** parameters.

Setting	Block Implementation
None	Block passes the applied chassis forces directly through to the rolling resistance and longitudinal force calculations.
Mapped stiffness and damping	Vertical motion depends on wheel stiffness and damping. Stiffness is a function of tire sidewall displacement and pressure. Damping is a function of tire sidewall velocity and pressure.

Selecting	Enables These Parameters	Creates These Output Ports
Mapped stiffness and damping	**Wheel and unsprung mass, m** **Initial deflection, zo** **Initial velocity, zdoto** **Gravitational acceleration, g** **Vertical deflection breakpoints, zFz** **Pressure breakpoints, pFz** **Force due to deflection, Fzz** **Vertical velocity breakpoints, zdotFz** **Force due to velocity, Fzzdot** **Ground displacement, Gndz** **Input ground displacement**	z zdot

Longitudinal scaling factor, lam_x — Friction scaling factor
1 (default)

Longitudinal friction scaling factor, dimensionless.

Dependencies

To enable this parameter, clear **Input friction scale factor**.

Input friction scale factor — Selection
Off (default)

Create input port for longitudinal friction scaling factor.

Dependencies

Selecting this parameter:

- Creates input port `lam_mux`.
- Disables parameter **Longitudinal scaling factor, lam_x**.

Wheel Dynamics

`Axle viscous damping coefficient, br` — Damping
`0.001` (default) | `scalar`

Axle viscous damping coefficient, *br,* in N·m·s/rad.

`Wheel inertia, Iyy` — Inertia
`0.8` (default) | `scalar`

Wheel inertia, in kg·m^2.

`Wheel initial angular velocity, omegao` — Wheel speed
`0` (default) | `scalar`

Initial angular velocity of wheel, along body-fixed *y*-axis, in rad/s.

`Relaxation length, Lrel` — Relaxation length
`0.5` (default) | `scalar`
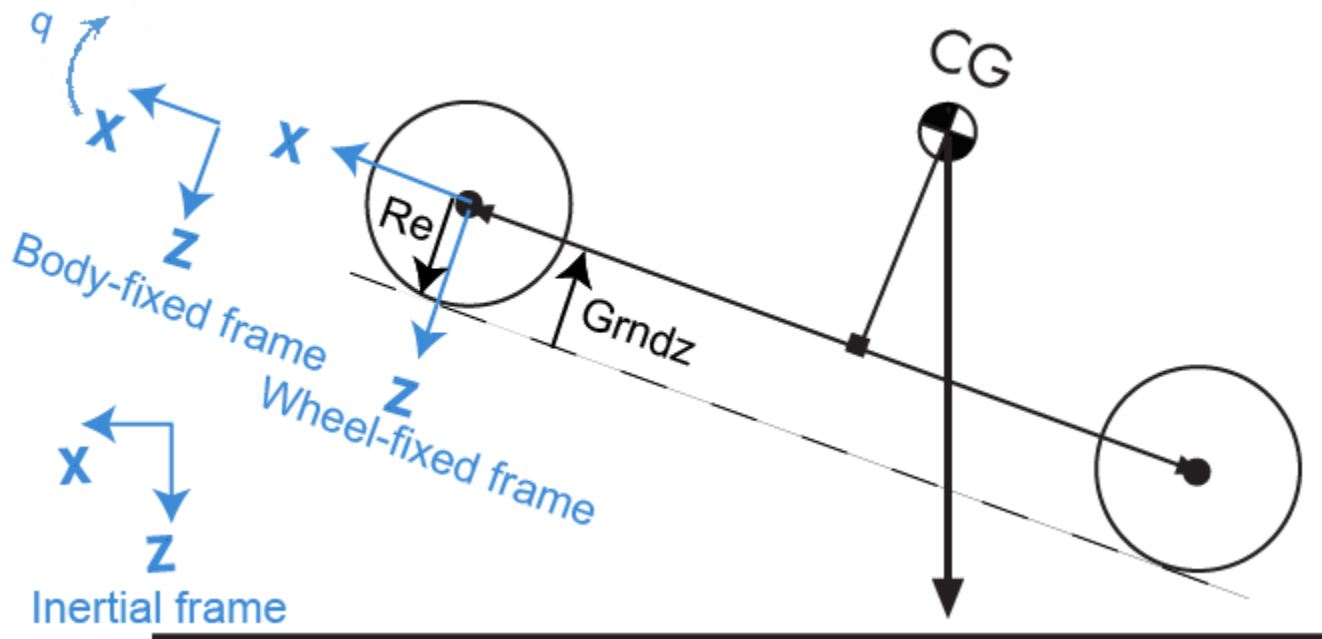
Wheel relaxation length, in m.

`Loaded radius, Re` — Loaded radius
`0.3` (default) | `scalar`

Loaded wheel radius, `Re`, in m.

Unloaded radius, UNLOADED_RADIUS — Unloaded radius
0.4 (default) | scalar

Unloaded wheel radius, in m.

Dependencies

To create this parameter, set **Rolling Resistance** to `Pressure and velocity` or `Magic Formula`.

Nominal longitudinal speed, LONGVL — Speed
16 (default) | scalar

Nominal longitudinal speed along body-fixed *x*-axis, in m/s.

Dependencies

To enable this parameter, set **Longitudinal Force** to `Magic Formula pure longitudinal slip`.

Nominal camber angle, gamma — Camber
0 (default) | scalar

Nominal camber angle, in rad.

Dependencies

To enable this parameter, set either:

• **Longitudinal Force** to `Magic Formula pure longitudinal slip`.
• **Rolling Resistance** to `Magic Formula`.

Nominal pressure, NOMPRES — Pressure
220000 (default) | scalar

Nominal pressure, in Pa.

Dependencies

To enable this parameter, set either:

• **Longitudinal Force** to Magic Formula pure longitudinal slip.
• **Rolling Resistance** to Magic Formula.

Pressure, press — Pressure
220000 (default) | scalar

Pressure, in Pa.

Dependencies

To enable this parameter:

• Set one of these:

   • **Longitudinal Force** to Magic Formula pure longitudinal slip.
   • **Rolling Resistance** to Pressure and velocity or Magic Formula.
   • **Vertical Motion** to Mapped stiffness and damping.
• On the **Wheel Dynamics** pane, clear **Input tire pressure**.

Longitudinal

Magic Formula Constant Value

Pure longitudinal peak factor, Dx — Factor
1 (default) | scalar

Pure longitudinal peak factor, dimensionless.

The coefficients are based on empirical tire data. These values are typical sets of constant Magic Formula coefficients for common road conditions.

Surface	B	C	D	E
Dry tarmac	10	1.9	1	0.97
Wet tarmac	12	2.3	0.82	1
Snow	5	2	0.3	1
Ice	4	2	0.1	1

Dependencies

To create this parameter, select the **Longitudinal Force** parameter Magic Formula constant value.

Pure longitudinal shape factor, Cx — Factor
1.65 (default) | scalar

Pure longitudinal shape factor, dimensionless.

The coefficients are based on empirical tire data. These values are typical sets of constant Magic Formula coefficients for common road conditions.

Surface	B	C	D	E
Dry tarmac	10	1.9	1	0.97
Wet tarmac	12	2.3	0.82	1
Snow	5	2	0.3	1
Ice	4	2	0.1	1

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula constant value`.

Pure longitudinal stiffness factor, Bx — Factor
`10` (default) | `scalar`

Pure longitudinal stiffness factor, dimensionless.

The coefficients are based on empirical tire data. These values are typical sets of constant Magic Formula coefficients for common road conditions.

Surface	B	C	D	E
Dry tarmac	10	1.9	1	0.97
Wet tarmac	12	2.3	0.82	1
Snow	5	2	0.3	1
Ice	4	2	0.1	1

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula constant value`.

Pure longitudinal curvature factor, Ex — Factor
`0.01` (default) | `scalar`

Pure longitudinal curvature factor, dimensionless.

The coefficients are based on empirical tire data. These values are typical sets of constant Magic Formula coefficients for common road conditions.

Surface	B	C	D	E
Dry tarmac	10	1.9	1	0.97
Wet tarmac	12	2.3	0.82	1
Snow	5	2	0.3	1
Ice	4	2	0.1	1

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula constant value`.

Magic Formula Pure Longitudinal Slip

`Cfx shape factor, PCX1` — Factor
`1.6` (default) | `scalar`

Cfx shape factor, PCX1, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

`Longitudinal friction at nominal normal load, PDX1` — Factor
`1` (default) | `scalar`

Longitudinal friction at nominal normal load, PDX1, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

`Frictional variation with load, PDX2` — Factor
`-0.08` (default) | `scalar`

Frictional variation with load, PDX2, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

`Frictional variation with camber, PDX3` — Factor
`0` (default) | `scalar`

Frictional variation with camber, PDX3, 1/rad^2.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

`Longitudinal curvature at nominal normal load, PEX1` — Factor
`0.112` (default) | `scalar`

Longitudinal curvature at nominal normal load, PEX1, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Variation of curvature factor with load, PEX2 — Factor
0.313 (default) | scalar

Variation of curvature factor with load, PEX2, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Variation of curvature factor with square of load, PEX3 — Factor
0 (default) | scalar

Variation of curvature factor with square of load, PEX3, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Longitudinal curvature factor with slip, PEX4 — Factor
0.0016 (default) | scalar

Longitudinal curvature factor with slip, PEX4, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Longitudinal slip stiffness at nominal normal load, PKX1 — Factor
21.7 (default) | scalar

Longitudinal slip stiffness at nominal normal load, PKX1, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Variation of slip stiffness with load, PKX2 — Factor
13.77 (default) | scalar

Variation of slip stiffness with load, PKX2, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Slip stiffness exponent factor, PKX3 — Factor
-0.412 (default) | scalar

Slip stiffness exponent factor, PKX3, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Horizontal shift in slip ratio at nominal normal load, PHX1 — Factor
`2.1585E-4` (default) | `scalar`

Horizontal shift in slip ratio at nominal normal load, PHX1, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Variation of horizontal slip ratio with load, PHX2 — Factor
`0.00115` (default) | `scalar`

Variation of horizontal slip ratio with load, PHX2, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Vertical shift in load at nominal normal load, PVX1 — Factor
`1.5973E-5` (default) | `scalar`

Vertical shift in load at nominal normal load, PVX1, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Variation of vertical shift with load, PVX2 — Factor
`1.043E-4` (default) | `scalar`

Variation of vertical shift with load, PVX2, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Linear variation of longitudinal slip stiffness with tire pressure, PPX1 — Factor
`-0.3489` (default) | `scalar`

Linear variation of longitudinal slip stiffness with tire pressure, PPX1, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Quadratic variation of longitudinal slip stiffness with tire pressure, PPX2 — Factor
`0.382` (default) | `scalar`

Quadratic variation of longitudinal slip stiffness with tire pressure, PPX2, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Linear variation of peak longitudinal friction with tire pressure, PPX3 — Factor
`-0.09634` (default) | `scalar`

Linear variation of peak longitudinal friction with tire pressure, PPX3, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Quadratic variation of peak longitudinal friction with tire pressure, PPX4 — Factor
`0.06447` (default) | `scalar`

Quadratic variation of peak longitudinal friction with tire pressure, PPX4, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Slip speed decay function scaling factor, lam_muV — Factor
`1` (default) | `scalar`

Slip speed decay function scaling factor, lam_muV, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Brake slip stiffness scaling factor, lam_Kxkappa — Factor
`1` (default) | `scalar`

Brake slip stiffness scaling factor, lam_Kxkappa, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Longitudinal shape scaling factor, lam_Cx — Factor
`1` (default) | `scalar`

Longitudinal shape scaling factor, lam_Cx, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter `Magic Formula pure longitudinal slip`.

Longitudinal curvature scaling factor, lam_Ex — Factor
0 (default) | scalar

Longitudinal curvature scaling factor, lam_Ex, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter Magic Formula pure longitudinal slip.

Longitudinal horizontal shift scaling factor, lam_Hx — Factor
1 (default) | scalar

Longitudinal horizontal shift scaling factor, lam_Hx, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter Magic Formula pure longitudinal slip.

Longitudinal vertical shift scaling factor, lam_Vx — Factor
1 (default) | scalar

Longitudinal vertical shift scaling factor, lam_Vx, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter Magic Formula pure longitudinal slip.

Mapped Force

Slip ratio breakpoints, kappaFx — Breakpoints
vector

Slip ratio breakpoints, dimensionless.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter Mapped force.

Normal force breakpoints, FzFx — Breakpoints
vector

Normal force breakpoints, N.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter Mapped force.

Longitudinal force map, FxMap — Lookup table
array

Longitudinal force versus slip ratio and normal force, N.

Dependencies

To create this parameter, select the **Longitudinal Force** parameter Mapped force.

Rolling Resistance

Pressure and Velocity

Velocity independent force coefficient, aMy — Force coefficient
8e-4 (default) | scalar

Velocity-independent force coefficient, *a*, in s/m.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Pressure and velocity.

Linear velocity force component, bMy — Force component
.001 (default) | scalar

Linear velocity force component, *b*, in s/m.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Pressure and velocity.

Quadratic velocity force component, cMy — Force component
1.6e-4 (default) | scalar

Quadratic velocity force component, *c*, in s^2/m^2.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Pressure and velocity.

Tire pressure exponent, alphaMy — Pressure exponent
-0.003 (default) | scalar

Tire pressure exponent, α, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Pressure and velocity.

Normal force exponent, betaMy — Force exponent
0.97 (default) | scalar

Normal force exponent, β, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Pressure and velocity.

ISO 28580

Parasitic losses force, Fpl — Force loss
10 (default) | scalar

Parasitic force loss, F_{pl}, in N.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

Rolling resistance constant, Cr — Constant
1e-3 (default) | scalar

Rolling resistance constant, C_r, in N/kN. ISO 28580 specifies the rolling resistance unit as one newton of tractive resistance for every kilonewtons of normal load.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

Thermal correction factor, Kt — Correction factor
.008 (default) | scalar

Thermal correction factor, K_t, in 1/degC.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

Measured temperature, Tmeas — Temperature
298.15 (default) | scalar

Measured temperature, T_{meas}, in K.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

Ambient temperature, Tamb — Temperature
298.15 (default) | scalar

Measured temperature, T_{amb}, in K.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

Input ambient temperature — Selection
off (default) | on

Select to create input port Tamb.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

Magic Formula

Rolling resistance torque coefficient, QSY1 — Torque coefficient
0.007 (default) | scalar

Rolling resistance torque coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

3-33

Longitudinal force rolling resistance coefficient, QSY2 — Force resistance coefficient
0 (default) | scalar

Longitudinal force rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

Linear rotational speed rolling resistance coefficient, QSY3 — Linear speed coefficient
0.0015 (default) | scalar

Linear rotational speed rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

Quartic rotational speed rolling resistance coefficient, QSY4 — Quartic speed coefficient
8.5e-05 (default) | scalar

Quartic rotational speed rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

Camber squared rolling resistance torque, QSY5 — Camber resistance torque
0 (default) | scalar

Camber squared rolling resistance torque, in $1/\text{rad}^2$.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

Load based camber squared rolling resistance torque, QSY6 — Load resistance torque
0 (default) | scalar

Load based camber squared rolling resistance torque, in $1/\text{rad}^2$.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

Normal load rolling resistance coefficient, QSY7 — Normal resistance coefficient
0.9 (default) | scalar

Normal load rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

Pressure load rolling resistance coefficient, QSY8 — Pressure resistance coefficient
-0.4 (default) | scalar

Pressure load rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

Rolling resistance scaling factor, lam_My — Scale
1 (default) | scalar

Rolling resistance scaling factor, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Magic Formula.

Mapped

Spin axis velocity breakpoints, VxMy — Breakpoints
-20:1:20 (default) | vector

Spin axis velocity breakpoints, in m/s.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Mapped torque.

Normal force breakpoints, FzMy — Breakpoints
0:200:1e4 (default) | vector

Normal force breakpoints, in N.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Mapped torque.

Rolling resistance torque map, MyMap — Lookup table
array

Rolling resistance torque versus axle speed and normal force, in N·m.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter Mapped torque.

Brake

Static friction coefficient, mu_static — Static friction
.3 (default) | scalar

Static friction coefficient, dimensionless.

Dependencies

To enable this parameter, for the **Brake Type** parameter, specify one of these types:

- `Disc`
- `Drum`
- `Mapped`

Kinetic friction coefficient, mu_kinetic — Kinetic friction
.2 (default) | scalar

Kinematic friction coefficient, dimensionless.

Dependencies

To enable this parameter, for the **Brake Type** parameter, specify one of these types:

- `Disc`
- `Drum`
- `Mapped`

Disc

Disc brake actuator bore, disc_abore — Bore distance
.05 (default) | scalar

Disc brake actuator bore, in m.

Dependencies

To enable the disc brake parameters, select `Disc` for the **Brake Type** parameter.

Brake pad mean radius, Rm — Radius
.177 (default) | scalar

Brake pad mean radius, in m.

Dependencies

To enable the disc brake parameters, select `Disc` for the **Brake Type** parameter.

Number of brake pads, num_pads — Count
2 (default) | scalar

Number of brake pads.

Dependencies

To enable the disc brake parameters, select `Disc` for the **Brake Type** parameter.

Drum

Drum brake actuator bore, disc_abore — Bore distance
0.0508 (default) | scalar

Drum brake actuator bore, in m.

Dependencies

To enable the drum brake parameters, select `Drum` for the **Brake Type** parameter.

Shoe pin to drum center distance, drum_a — Distance
0.123 (default) | scalar

Shoe pin to drum center distance, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin center to force application point distance, drum_c — Distance
0.212 (default) | scalar

Shoe pin center to force application point distance, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Drum internal radius, drum_r — Radius
0.15 (default) | scalar

Drum internal radius, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin to pad start angle, drum_theta1 — Angle
0 (default) | scalar

Shoe pin to pad start angle, in deg.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin to pad end angle, drum_theta2 — Angle
126 (default) | scalar

Shoe pin to pad end angle, in deg.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Mapped

Brake actuator pressure breakpoints, brake_p_bpt — Breakpoints
vector

Brake actuator pressure breakpoints, in bar.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Wheel speed breakpoints, brake_n_bpt — Breakpoints
vector

Wheel speed breakpoints, in rpm.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Brake torque map, f_brake_t — Lookup table
array

The lookup table for the brake torque, $f_{brake}(P, N)$, is a function of applied brake pressure and wheel speed, where:

- T is brake torque, in N·m.
- P is applied brake pressure, in bar.
- N is wheel speed, in rpm.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Vertical

Nominal normal force, FNOMIN — Force
2000 (default) | scalar

Nominal rated wheel load along wheel-fixed z-axis, in N.

Dependencies

To enable this parameter, set either:

- **Longitudinal Force** to Magic Formula pure longitudinal slip.
- **Rolling Resistance** to Magic Formula.

Nominal rated load scaling factor, lam_Fzo — Factor
1 (default) | scalar

Nominal rated load scaling factor, dimensionless. Used to scale the normal for specific applications and load conditions.

Dependencies

To enable this parameter, set **Longitudinal Force** to Magic Formula pure longitudinal slip.

Wheel and unsprung mass, m — Mass
10 (default) | scalar

Wheel and unsprung mass, in kg. Used in the vertical motion calculations.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Initial deflection, zo — Deflection
0 (default) | scalar

Initial axle displacement along wheel-fixed z-axis, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Initial velocity, zdoto — Velocity
0 (default) | scalar

Initial axle velocity along wheel-fixed z-axis, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Gravitational acceleration, g — Gravity
9.81 (default) | scalar

Gravitational acceleration, in m/s^2.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Ground displacement, Gndz — Displacement
0 (default) | scalar

Ground displacement, Grndz, along negative wheel-fixed z-axis, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Mapped Stiffness and Damping

Vertical deflection breakpoints, zFz — Breakpoints
[0 .01 .1] (default) | vector

Vector of sidewall deflection breakpoints corresponding to the force table, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Pressure breakpoints, pFz — Breakpoints
[10000 1000000] (default) | vector

Vector of pressure data points corresponding to the force table, in Pa.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Force due to deflection, Fzz — Force
[0 1e3 1e4; 0 1e4 1e5] (default) | vector

Force due to sidewall deflection and pressure along wheel-fixed *z*-axis, in N.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Vertical velocity breakpoints, zdotFz — Breakpoints
[-20 0 20] (default) | scalar

Vector of sidewall velocity breakpoints corresponding to the force due to velocity table, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Force due to velocity, Fzzdot — Force
[500 0 -500;250 0 -250] (default) | array

Force due to sidewall velocity and pressure along wheel-fixed *z*-axis, in N.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Simulation Setup

Minimum normal force, FZMIN — Force
0 (default) | scalar

Minimum normal force, in N. Used with all vertical force calculations.

Maximum normal force, FZMAX — Force
10000 (default) | scalar

Maximum normal force, in N. Used with all vertical force calculations.

Max allowable slip ratio (absolute), kappamax — Ratio
1.5 (default) | scalar

Maximum allowable absolute slip ratio, dimensionless.

Velocity tolerance used to handle low velocity situations, VXLOW — Tolerance
1 (default) | scalar

Velocity tolerance used to handle low-velocity situations, in m/s.

Minimum ambient temperature, TMIN — Tmin
0 (default) | scalar

Minimum ambient temperature, T_{MIN}, in K.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

Maximum ambient temperature, TMAX — Tmax
400 (default) | scalar

Maximum ambient temperature, T_{MAX}, in K.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

References

[1] Highway Tire Committee. *Stepwise Coastdown Methodology for Measuring Tire Rolling Resistance*. Standard J2452_199906. Warrendale, PA: SAE International, June 1999.

[2] Pacejka, H. B. *Tire and Vehicle Dynamics*. 3rd ed. Oxford, United Kingdom: SAE and Butterworth-Heinemann, 2012.

[3] Schmid, Steven R., Bernard J. Hamrock, and Bo O. Jacobson. "Chapter 18: Brakes and Clutches." *Fundamentals of Machine Elements, SI Version*. 3rd ed. Boca Raton, FL: CRC Press, 2014.

[4] Shigley, Joseph E., and Larry Mitchel. *Mechanical Engineering Design*. 4th ed. New York, NY: McGraw Hill, 1983.

[5] ISO 28580:2018. *Passenger car, truck and bus tyre rolling resistance measurement method -- Single point test and correlation of measurement results*. ISO (International Organization for Standardization), 2018.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Combined Slip Wheel 2DOF | Drive Cycle Source | Longitudinal Driver

Introduced in R2017a

Combined Slip Wheel 2DOF

Combined slip 2DOF wheel with disc, drum, or mapped brake
Library: Vehicle Dynamics Blockset / Wheels and Tires
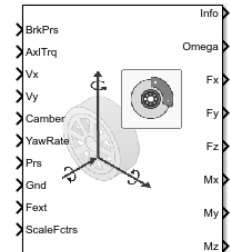
Description

The Combined Slip Wheel 2DOF block implements the longitudinal and lateral behavior of a wheel characterized by the Magic Formula[1] and [2]. Use the block in driveline and vehicle simulations where low frequency tire-road and braking forces are required to determine vehicle acceleration, braking, and wheel-rolling resistance. The block is suitable for applications that require combined lateral slip, for example, in lateral motion and yaw stability studies.

Based on the driveline torque, brake pressure, road height, wheel camber angle, and inflation pressure, the block determines the wheel rotation rate, vertical motion, forces, and moments in all six degrees of freedom (DOF). Use the vertical DOF to study tire-suspension resonances from road profiles or chassis motion.

To implement the Magic Formula, the block uses empirical equations[1] and [2]. The equations use fitting coefficients that correspond to the block parameters.

To update the block parameters with fitting coefficients from a file:

1   On the **Wheel and Tire Parameters > Tire** pane, select **Select file**.
2   Select the tire coefficient file.
3   Select **Apply**.
4   Select **Update mask values from file**. In the dialog box that prompts you for confirmation, click **OK**. The block updates the parameters.

Use the **Brake Type** parameter to select the brake.

Brake Type Setting	Brake Implementation
None	None
Disc	Brake that converts the brake cylinder pressure into a braking force
Drum	Simplex drum brake that converts the applied force and brake geometry into a net braking torque
Mapped	Lookup table that is a function of the wheel speed and applied brake pressure

Rotational Wheel Dynamics

The block calculates the inertial response of the wheel subject to:

- Axle losses
- Brake and drive torque
- Tire rolling resistance
- Ground contact through the tire-road interface

To implement the Magic Formula, the block uses these equations.

Calculation	Equations
Longitudinal force	*Tire and Vehicle Dynamics*[2] equations 4.E9 through 4.E57
Lateral force - pure sideslip	*Tire and Vehicle Dynamics*[2] equations 4.E19 through 4.E30
Lateral force - combined slip	*Tire and Vehicle Dynamics*[2] equations 4.E58 through 4.E67
Vertical dynamics	*Tire and Vehicle Dynamics*[2] equations 4.E68, 4.E1, 4.E2a, and 4.E2b
Overturning couple	*Tire and Vehicle Dynamics*[2] equation 4.E69
Rolling resistance	- *An improved Magic Formula/Swift tyre model that can handle inflation pressure changes*[1] equation 6.1.2 - *Tire and Vehicle Dynamics*[2] equation 4.E70
Aligning moment	*Tire and Vehicle Dynamics*[2] equation 4.E31 through 4.E49
Aligning torque - combined slip	*Tire and Vehicle Dynamics*[2] equation 4.E71 through 4.E78

The input torque is the summation of the applied axle torque, braking torque, and moment arising from the combined tire torque.

$$T_i = T_a - T_b + T_d$$

For the moment arising from the combined tire torque, the block implements tractive wheel forces and rolling resistance with first-order dynamics. The rolling resistance has a time constant parameterized in terms of a relaxation length.

$$T_d(s) = \frac{1}{\frac{|\omega| R_e}{L_e} s + 1}(F_x R_e + M_y)$$

If the brakes are enabled, the block determines the braking locked or unlocked condition based on an idealized dry clutch friction model. Based on the lockup condition, the block implements these friction and dynamic models.

If	Lockup Condition	Friction Model	Dynamic Model
$\omega \neq 0$ or $T_S < \left\lvert T_i + T_f - \omega b \right\rvert$	Unlocked	$T_f = T_k$ where, $T_k = F_c R_{eff} \mu_k \tanh[4(-\omega_d)]$ $T_s = F_c R_{eff} \mu_s$ $R_{eff} = \dfrac{2(R_o3 - R_i3)}{3(R_o2 - R_i2)}$	$\dot{\omega} J = -\omega b + T_i + T_o$
$\omega = 0$ and $T_S \geq \left\lvert T_i + T_f - \omega b \right\rvert$	Locked	$T_f = T_s$	$\omega = 0$

The equations use these variables.

ω	Wheel angular velocity
a	Velocity independent force component
b	Linear velocity force component
c	Quadratic velocity force component
L_e	Tire relaxation length
J	Moment of inertia
M_y	Rolling resistance torque
T_a	Applied axle torque about wheel spin axis
T_b	Braking torque
T_d	Combined tire torque
T_f	Frictional torque
T_i	Net input torque
T_k	Kinetic frictional torque
T_o	Net output torque
T_s	Static frictional torque
F_c	Applied clutch force
F_x	Longitudinal force developed by the tire road interface due to slip
R_{eff}	Effective clutch radius
R_o	Annular disk outer radius
R_i	Annular disk inner radius
R_e	Effective tire radius while under load and for a given pressure
V_x	Longitudinal axle velocity
F_z	Vehicle normal force
α	Tire pressure exponent
β	Normal force exponent

p_i	Tire pressure
μ_s	Coefficient of static friction
μ_k	Coefficient of kinetic friction

Tire and Wheel Coordinate Systems

To resolve the forces and moments, the block uses the Z-Up orientation of the tire and wheel coordinate systems.

- Tire coordinate system axes (X_T, Y_T, Z_T) are fixed in a reference frame attached to the tire. The origin is at the tire contact with the ground.

- Wheel coordinate system axes (X_W, Y_W, Z_W) are fixed in a reference frame attached to the wheel. The origin is at the wheel center.

Z-Up Orientation[1]

Brakes

Disc

If you specify the **Brake Type** parameter `Disc`, the block implements a disc brake. This figure shows the side and front views of a disc brake.

A disc brake converts brake cylinder pressure from the brake cylinder into force. The disc brake applies the force at the brake pad mean radius.

The block uses these equations to calculate brake torque for the disc brake.

$$T = \begin{cases} \dfrac{\mu P \pi B_a{}^2 R_m N_{pads}}{4} & \text{when } N \neq 0 \\[2ex] \dfrac{\mu_{static} P \pi B_a{}^2 R_m N_{pads}}{4} & \text{when } N = 0 \end{cases}$$

$$Rm = \frac{Ro + Ri}{2}$$

The equations use these variables.

T	Brake torque
P	Applied brake pressure
N	Wheel speed
N_{pads}	Number of brake pads in disc brake assembly
μ_{static}	Disc pad-rotor coefficient of static friction
μ	Disc pad-rotor coefficient of kinetic friction
B_a	Brake actuator bore diameter
R_m	Mean radius of brake pad force application on brake rotor
R_o	Outer radius of brake pad

R_i	Inner radius of brake pad

Drum

If you specify the **Brake Type** parameter `Drum`, the block implements a static (steady-state) simplex drum brake. A simplex drum brake consists of a single two-sided hydraulic actuator and two brake shoes. The brake shoes do not share a common hinge pin.

The simplex drum brake model uses the applied force and brake geometry to calculate a net torque for each brake shoe. The drum model assumes that the actuators and shoe geometry are symmetrical for both sides, allowing a single set of geometry and friction parameters to be used for both shoes.

The block implements equations that are derived from these equations in *Fundamentals of Machine Elements*.

$$T_{rshoe} = \left(\frac{\pi \mu c r (\cos\theta_2 - \cos\theta_1) B_a 2}{2\mu\left(2r\left(\cos\theta_2 - \cos\theta_1\right) + a\left(\cos^2\theta_2 - \cos^2\theta_1\right)\right) + ar(2\theta_1 - 2\theta_2 + \sin2\theta_2 - \sin2\theta_1)} \right) P$$

$$T_{lshoe} = \left(\frac{\pi \mu c r (\cos\theta_2 - \cos\theta_1) B_a 2}{-2\mu\left(2r\left(\cos\theta_2 - \cos\theta_1\right) + a\left(\cos^2\theta_2 - \cos^2\theta_1\right)\right) + ar(2\theta_1 - 2\theta_2 + \sin2\theta_2 - \sin2\theta_1)} \right) P$$

$$T = \begin{cases} T_{rshoe} + T_{lshoe} & \text{when } N \neq 0 \\ (T_{rshoe} + T_{lshoe})\frac{\mu_{static}}{\mu} & \text{when } N = 0 \end{cases}$$

Rotation

The equations use these variables.

T	Brake torque
P	Applied brake pressure
N	Wheel speed

μ_{static}	Disc pad-rotor coefficient of static friction
μ	Disc pad-rotor coefficient of kinetic friction
T_{rshoe}	Right shoe brake torque
T_{lshoe}	Left shoe brake torque
a	Distance from drum center to shoe hinge pin center
c	Distance from shoe hinge pin center to brake actuator connection on brake shoe
r	Drum internal radius
B_a	Brake actuator bore diameter
Θ_1	Angle from shoe hinge pin center to start of brake pad material on shoe
Θ_2	Angle from shoe hinge pin center to end of brake pad material on shoe

Mapped

If you specify the **Brake Type** parameter `Mapped`, the block uses a lookup table to determine the brake torque.

$$T = \begin{cases} f_{brake}(P, N) & \text{when } N \neq 0 \\ \left(\frac{\mu_{static}}{\mu}\right) f_{brake}(P, N) & \text{when } N = 0 \end{cases}$$

The equations use these variables.

T	Brake torque
$f_{brake}(P, N)$	Brake torque lookup table
P	Applied brake pressure
N	Wheel speed
μ_{static}	Friction coefficient of drum pad-face interface under static conditions
μ	Friction coefficient of disc pad-rotor interface

The lookup table for the brake torque, $f_{brake}(P, N)$, is a function of applied brake pressure and wheel speed, where:

- T is brake torque, in N·m.
- P is applied brake pressure, in bar.
- N is wheel speed, in rpm.

Ports

Input

BrkPrs — Brake pressure
scalar | N-by-1 vector

Brake pressure, in Pa.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Dependencies

To enable this port, for the **Brake Type** parameter, specify one of these types:

- Disc
- Drum
- Mapped

AxlTrq — Axle torque
scalar | N-by-1 vector

Axle torque, T_a, about wheel spin axis, in N·m.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Vx — Longitudinal velocity
scalar | N-by-1 vector

Axle longitudinal velocity, V_x, along tire-fixed *x*-axis, in m/s.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Vy — Lateral velocity
scalar | N-by-1 vector

Axle lateral velocity, V_y, along tire-fixed *y*-axis, in m/s.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Camber — Camber angle
scalar | N-by-1 vector

Camber angle, ɣ, in rad.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

YawRate — Tire angular velocity
scalar | N-by-1 vector

Tire angular velocity, *r*, about the tire-fixed *z*-axis (yaw rate), in rad/s.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Prs — Tire inflation pressure
scalar | N-by-1 vector

Tire inflation pressure, p_i, in Pa.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Gnd — Ground displacement
scalar | N-by-1 vector

Ground displacement along tire-fixed *z*-axis, in m. Positive input produces wheel lift.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Fext — Axle force applied to tire
scalar | N-by-1 vector

Axle force applied to tire, F_{ext}, along vehicle-fixed *z*-axis (positive input compresses the tire), in N·m.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

ScaleFctrs — Scale factors
27-by-N array

Magic Formula scale factor array. Array dimensions are 27 by the number of wheels, *N*.

The Magic Formula equations use scale factors to account for static or simulation run-time variations. Nominally, most are set to 1.

Array Element	Variable	Scale Factor
ScaleFctrs(1,1)	lam_Fzo	Nominal load
ScaleFctrs(2,1)	lam_mux	Longitudinal peak friction coefficient
ScaleFctrs(3,1)	lam_muy	Lateral peak friction coefficient
ScaleFctrs(4,1)	lam_muV	Slip speed Vs decaying friction
ScaleFctrs(5,1)	lam_Kxkappa	Brake slip stiffness
ScaleFctrs(6,1)	lam_Kyalpha	Cornering stiffness
ScaleFctrs(7,1)	lam_Cx	Longitudinal shape factor
ScaleFctrs(8,1)	lam_Cy	Lateral shape factor
ScaleFctrs(9,1)	lam_Ex	Longitudinal curvature factor
ScaleFctrs(10,1)	lam_Ey	Lateral curvature factor
ScaleFctrs(11,1)	lam_Hx	Longitudinal horizontal shift
ScaleFctrs(12,1)	lam_Hy	Lateral horizontal shift
ScaleFctrs(13,1)	lam_Vx	Longitudinal vertical shift

Array Element	Variable	Scale Factor
ScaleFctrs(14,1)	lam_Vy	Lateral vertical shift
ScaleFctrs(15,1)	lam_Kygamma	Camber force stiffness
ScaleFctrs(16,1)	lam_Kzgamma	Camber torque stiffness
ScaleFctrs(17,1)	lam_t	Pneumatic trail (effecting aligning torque stiffness)
ScaleFctrs(18,1)	lam_Mr	Residual torque
ScaleFctrs(19,1)	lam_xalpha	Alpha influence on *Fx* (kappa)
ScaleFctrs(20,1)	lam_ykappa	Kappa influence on *Fy* (alpha)
ScaleFctrs(21,1)	lam_Vykappa	Induced ply steer *Fy*
ScaleFctrs(22,1)	lam_s	Moment arm of *Fx*
ScaleFctrs(23,1)	lam_Cz	Radial tire stiffness
ScaleFctrs(24,1)	lam_Mx	Overturning couple stiffness
ScaleFctrs(25,1)	lam_VMx	Overturning couple vertical shift
ScaleFctrs(26,1)	lam_My	Rolling resistance moment
ScaleFctrs(27,1)	lam_Mphi	Parking torque *Mz*

Output

Info — Block data
bus

Block data, returned as a bus signal containing these block values.

Signal	Description	Units
AxlTrq	Axle torque about wheel-fixed *y*-axis	N·m
Omega	Wheel angular velocity about wheel-fixed *y*-axis	rad/s
Fx	Longitudinal vehicle force along tire-fixed *x*-axis	N
Fy	Lateral vehicle force along tire-fixed *y*-axis	N
Fz	Vertical vehicle force along tire-fixed *z*-axis	N
Mx	Overturning moment about tire-fixed *x*-axis	N·m
My	Rolling resistance torque about tire-fixed *y*-axis	N·m
Mz	Aligning moment about tire-fixed *z*-axis	N·m
Vx	Vehicle longitudinal velocity along tire-fixed *x*-axis	m/s
Vy	Vehicle lateral velocity along tire-fixed *y*-axis	m/s
Re	Loaded effective radius	m
Kappa	Longitudinal slip ratio	NA
Alpha	Side slip angle	rad
a	Contact patch half length	m
b	Contact patch half width	m

Signal	Description	Units
Gamma	Camber angle	rad
psidot	Tire angular velocity about the tire-fixed z-axis (yaw rate)	rad/s
BrkTrq	Brake torque about vehicle-fixed y-axis	N·m
BrkPrs	Brake pressure	Pa
z	Axle vertical displacement along tire-fixed z-axis	m
zdot	Axle vertical velocity along tire-fixed z-axis	m/s
Gnd	Ground displacement along tire-fixed z-axis (positive input produces wheel lift)	m
GndFz	Vertical sidewall force on ground along tire-fixed z-axis	N
Prs	Tire inflation pressure	Pa

Omega — Wheel angular velocity
scalar | N-by-1 vector

Wheel angular velocity, ω, about wheel-fixed y-axis, in rad/s.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Fx — Longitudinal axle force
scalar | N-by-1 vector

Longitudinal force acting on axle, F_x, along tire-fixed x-axis, in N. Positive force acts to move the vehicle forward.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Fy — Lateral axle force
scalar | N-by-1 vector

Lateral force acting on axle, F_y, along tire-fixed y-axis, in N.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Fz — Vertical axle force
scalar | N-by-1 vector

Vertical force acting on axle, F_z, along tire-fixed z-axis, in N.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Mx — Overturning moment
scalar | N-by-1 vector

Longitudinal moment acting on axle, M_x, about tire-fixed x-axis, in N·m.

3-53

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

My — Rolling resistive moment
scalar | N-by-1 vector

Lateral moment acting on axle, M_y, about tire-fixed *y*-axis, in N·m.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Mz — Aligning moment
scalar | N-by-1 vector

Vertical moment acting on axle, M_z, about tire-fixed *z*-axis, in N·m.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Parameters

Block Options

Brake Type — Select type
None | Disc | Drum | Mapped

Use the **Brake Type** parameter to select the brake.

Brake Type Setting	Brake Implementation
None	None
Disc	Brake that converts the brake cylinder pressure into a braking force
Drum	Simplex drum brake that converts the applied force and brake geometry into a net braking torque
Mapped	Lookup table that is a function of the wheel speed and applied brake pressure

Brake

Static friction coefficient, mu_static — Static friction
.3 (default) | scalar

Static friction coefficient, dimensionless.

Dependencies

To enable this parameter, for the **Brake Type** parameter, specify one of these types:

• Disc
• Drum
• Mapped

Kinetic friction coefficient, mu_kinetic — Kinetic friction
.2 (default) | scalar

Kinematic friction coefficient, dimensionless.

Dependencies

To enable this parameter, for the **Brake Type** parameter, specify one of these types:

- Disc
- Drum
- Mapped

Disc

Disc brake actuator bore, disc_abore — Bore distance
.05 (default) | scalar

Disc brake actuator bore, in m.

Dependencies

To enable the disc brake parameters, select Disc for the **Brake Type** parameter.

Brake pad mean radius, Rm — Radius
.177 (default) | scalar

Brake pad mean radius, in m.

Dependencies

To enable the disc brake parameters, select Disc for the **Brake Type** parameter.

Number of brake pads, num_pads — Count
2 (default) | scalar

Number of brake pads.

Dependencies

To enable the disc brake parameters, select Disc for the **Brake Type** parameter.

Drum

Drum brake actuator bore, disc_abore — Bore distance
0.0508 (default) | scalar

Drum brake actuator bore, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin to drum center distance, drum_a — Distance
0.123 (default) | scalar

Shoe pin to drum center distance, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin center to force application point distance, drum_c — Distance
0.212 (default) | scalar

Shoe pin center to force application point distance, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Drum internal radius, drum_r — Radius
0.15 (default) | scalar

Drum internal radius, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin to pad start angle, drum_theta1 — Angle
0 (default) | scalar

Shoe pin to pad start angle, in deg.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin to pad end angle, drum_theta2 — Angle
126 (default) | scalar

Shoe pin to pad end angle, in deg.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Mapped

Brake actuator pressure breakpoints, brake_p_bpt — Breakpoints
vector

Brake actuator pressure breakpoints, in bar.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Wheel speed breakpoints, brake_n_bpt — Breakpoints
vector

Wheel speed breakpoints, in rpm.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Brake torque map, f_brake_t — Lookup table
array

The lookup table for the brake torque, $f_{brake}(P, N)$, is a function of applied brake pressure and wheel speed, where:

- T is brake torque, in N·m.
- P is applied brake pressure, in bar.
- N is wheel speed, in rpm.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Tire

Tire file or object, tireParamSet — Tire file
vdynPassCar.mat (default) | .tir | .txt

Tire file .tir or object containing empirical data to model tire longitudinal and lateral behavior with the Magic Formula. If you provide an .txt file, make sure the file contains names that correspond to the block parameters.

To update the block parameters with fitting coefficients from a file:

1 On the **Wheel and Tire Parameters > Tire** pane, select **Select file**.
2 Select the tire coefficient file.
3 Select **Apply**.
4 Select **Update mask values from file**. In the dialog box that prompts you for confirmation, click **OK**. The block updates the parameters.

Simulation

Maximum pressure, PRESMAX — Pressure
1003118 (default) | scalar

Maximum pressure, *PRESMAX*, in Pa.

Minimum pressure, PRESMIN — Pressure
9982 (default) | scalar

Minimum pressure, *PRESMIN*, in Pa.

Maximum normal force, FZMAX — Force
10000 (default) | scalar

Maximum normal force, *FZMAX*, in N.

Minimum normal force, FZMIN — Force
100 (default) | scalar

Minimum normal force, *FZMIN*, in N.

Velocity tolerance used to handle low velocity situations, VXLOW — Tolerance
0.1 (default) | scalar

Velocity tolerance used to handle low velocity situations, *VXLOW*, in m/s.

Max allowable slip ratio (absolute), KPUMAX — Ratio
1.5 (default) | scalar

Max allowable slip ratio (absolute), *KPUMAX*, dimensionless.

Minimum allowable slip ratio (absolute), KPUMIN — Ratio
-1.5 (default) | scalar

Minimum allowable slip ratio (absolute), *KPUMIN*, dimensionless.

Max allowable slip angle (absolute), ALPMAX — Angle
1.5708 (default) | scalar

Max allowable slip angle (absolute), *ALPMAX*, in rad.

Minimum allowable slip angle (absolute), ALPMIN — Angle
-1.5708 (default) | scalar

Minimum allowable slip angle (absolute), *ALPMIN*, in rad.

Maximum allowable camber angle, CAMMAX — Angle
0.173 (default) | scalar

Maximum allowable camber angle *CAMMAX*, in rad.

Minimum allowable camber angle, CAMMIN — Angle
-0.173 (default) | scalar

Minimum allowable camber angle, *CAMMIN*, in rad.

Nominal longitudinal speed, LONGVL — Speed
16.7 (default) | scalar

Nominal longitudinal speed, *LONGVL*, in m/s.

Wheel

Initial rotational velocity, omegao — Velocity
0 (default) | scalar

Initial rotational velocity, in rad/s.

Rotational damping, br — Damping
1e-3 (default) | scalar

Rotational damping, *br,* in N·m·s/rad.

Unloaded radius, UNLOADED_RADIUS — Radius
0.309 (default) | scalar

Unloaded radius, *UNLOADED_RADIUS*, in m.

Nominal pressure, NOMPRES — Pressure
224006 (default) | scalar

Nominal pressure, *NOMPRES*, in Pa.

Nominal normal force, FNOMIN — Force
4025 (default) | scalar

Nominal normal force, *FNOMIN*, in N.

Wheel width, WIDTH — Width
0.209 (default) | scalar

Wheel width, *WIDTH*, in m.

Inertial

Wheel mass, MASS — Mass
9.465 (default) | scalar

Wheel mass, *MASS*, in kg.

Rotational inertia (rolling axis), IYY — Inertia
0.74 (default) | scalar

Rotational inertia (rolling axis), *IYY,* in kg·m^2.

Gravity, GRAVITY — Gravity
-9.81 (default) | scalar

Gravity, *GRAVITY*, in m/s^2.

Vertical

Initial tire deflection, zo — Deflection
0 (default) | scalar

Initial tire deflection, *zo,* in m.

Initial wheel vertical velocity (wheel fixed frame), zdoto — Velocity
0 (default) | scalar

Initial wheel vertical velocity (wheel fixed frame), *zdoto,* in m/s.

Effective rolling radius at low load stiffness, BREFF — Stiffness
8.25094594147963 (default) | scalar

Effective rolling radius at low load stiffness, *BREFF*, dimensionless.

Effective rolling radius peak value, DREFF — Radius
0.260468730454265 (default) | scalar

Effective rolling radius peak value, *DREFF*, dimensionless.

Effective rolling radius at high load stiffness, FREFF — Radius
0.0735298544471851 (default) | scalar

Effective rolling radius at high load stiffness, *FREFF*, dimensionless.

Unloaded to nominal rolling radius ratio, Q_RE0 — Ratio
1.00866439868088 (default) | scalar

Unloaded to nominal rolling radius ratio, *Q_RE0*, dimensionless.

Radius rotational speed dependence, Q_V1 — Speed
0.000760413786224011 (default) | scalar

Radius rotational speed dependence, *Q_V1*, dimensionless.

Stiffness rotational speed dependence, Q_V2 — Speed
0.0463384792019201 (default) | scalar

Stiffness rotational speed dependence, *Q_V2*, dimensionless.

Linear load change with deflection, Q_FZ1 — Load change
0 (default) | scalar

Linear load change with deflection, *Q_FZ1*, dimensionless.

Quadratic load change with deflection, Q_FZ2 — Load change
15.6870832810226 (default) | scalar

Quadratic load change with deflection, *Q_FZ2*, dimensionless.

Linear load change with deflection and quadratic camber, Q_FZ3 — Load change
0 (default) | scalar

Linear load change with deflection and quadratic camber, *Q_FZ3*, dimensionless.

Load response to longitudinal force, Q_FCX — Force
0.138643970247602 (default) | scalar

Load response to longitudinal force, *Q_FCX*, dimensionless.

Load response to lateral force, Q_FCY — Force
0.10843499565426 (default) | scalar

Load response to lateral force, *Q_FCY*, dimensionless.

Vertical stiffness change due to lateral load dependency on lateral stiffness, Q_FCY2 — Stiffness
-0.465763352339538 (default) | scalar

Vertical stiffness change due to lateral load dependency on lateral stiffness, *Q_FCY2*, dimensionless.

Stiffness response to pressure, PFZ1 — Stiffness
0.69958166705601 (default) | scalar

Stiffness response to pressure, *PFZ1*, dimensionless.

Vertical tire stiffness, VERTICAL_STIFFNESS — Stiffness
207885.061134007 (default) | scalar

Vertical tire stiffness, *VERTICAL_STIFFNESS*, in N/m.

Vertical tire damping, VERTICAL_DAMPING — Damping
494.649255786991 (default) | scalar

Vertical tire damping, *VERTICAL_DAMPING*, in N*s/m.

Structural

Longitudinal stiffness, LONGITUDINAL_STIFFNESS — Stiffness
388714.175879854 (default) | scalar

Longitudinal stiffness, *LONGITUDINAL_STIFFNESS*, in N/m.

Longitudinal stiffness, LATERAL_STIFFNESS — Stiffness
157087.404633332 (default) | scalar

Longitudinal stiffness, *LATERAL_STIFFNESS*, in N/m.

Linear vertical deflection influence on longitudinal stiffness, PCFX1 — Deflection influence
0.18539229191199 (default) | scalar

Linear vertical deflection influence on longitudinal stiffness, *PCFX1*, dimensionless.

Quadratic vertical deflection influence on longitudinal stiffness, PCFX2 — Deflection influence
0 (default) | scalar

Quadratic vertical deflection influence on longitudinal stiffness, *PCFX2*, dimensionless.

Pressure dependency on longitudinal stiffness, PCFX3 — Pressure dependency
0 (default) | scalar

Pressure dependency on longitudinal stiffness, *PCFX3*, dimensionless.

Linear vertical deflection influence on lateral stiffness, PCFY1 — Deflection influence
0.163183345037677 (default) | scalar

Linear vertical deflection influence on lateral stiffness, *PCFY1*, dimensionless.

3-61

Quadratic vertical deflection influence on lateral stiffness, PCFY2 — Deflection influence
0 (default) | scalar

Quadratic vertical deflection influence on lateral stiffness, *PCFY2*, dimensionless.

Pressure dependency on longitudinal stiffness, PCFY3 — Pressure dependency
0.249094918231954 (default) | scalar

Pressure dependency on longitudinal stiffness, *PCFY3*, dimensionless.

Contact Patch

Contact length square root term, Q_RA1 — Length term
0.675942571700787 (default) | scalar

Contact length square root term, *Q_RA1*, dimensionless.

Contact length linear term, Q_RA2 — Length term
0.73800463546164 (default) | scalar

Contact length linear term, *Q_RA2*, dimensionless.

Contact width root term, Q_RB1 — Width term
1.04487486456705 (default) | scalar

Contact width root term, *Q_RB1*, dimensionless.

Contact width linear term, Q_RB2 — Width term
-1.191763806629 (default) | scalar

Contact width linear term *Q_RB2*, dimensionless.

Longitudinal

Cfx shape factor, PCX1 — Shape factor
1.58523057950359 (default) | scalar

Shape factor, C_{fx}, *PCX1*, dimensionless.

Longitudinal friction at nominal normal load, PDX1 — Friction
1.0239116238178 (default) | scalar

Longitudinal friction at nominal normal load, *PDX1*, dimensionless.

Frictional variation with load, PDX2 — Friction variation
-0.0842405110022724 (default) | scalar

Frictional variation with load, *PDX2*, dimensionless.

Frictional variation with camber, PDX3 — Friction variation
0 (default) | scalar

Frictional variation with camber, *PDX3*, in 1/rad^2.

Longitudinal curvature at nominal normal load, PEX1 — Curvature
0.110312777158455 (default) | scalar

Longitudinal curvature at nominal normal load, *PEX1*, dimensionless.

Variation of curvature factor with load, PEX2 — Curvature variation
0.313468516038146 (default) | scalar

Variation of curvature factor with load, *PEX2*, dimensionless.

Variation of curvature factor with square of load, PEX3 — Curvature variation
0 (default) | scalar

Variation of curvature factor with square of load, *PEX3*, dimensionless.

Longitudinal curvature factor with slip, PEX4 — Curvature
0.0016060905760045 (default) | scalar

Longitudinal curvature factor with slip, *PEX4*, dimensionless.

Longitudinal slip stiffness at nominal normal load, PKX1 — Stiffness
21.3872544684023 (default) | scalar

Longitudinal slip stiffness at nominal normal load, *PKX1*, dimensionless.

Variation of slip stiffness with load, PKX2 — Stiffness variation
14.0006541873175 (default) | scalar

Variation of slip stiffness with load, *PKX1*, dimensionless.

Slip stiffness exponent factor, PKX3 — Slip stiffness
-0.405326109653452 (default) | scalar

Slip stiffness exponent factor, *PKX3*, dimensionless.

Horizontal shift in slip ratio at nominal normal load, PHX1 — Slip ratio shift
0.000212599305364818 (default) | scalar

Horizontal shift in slip ratio at nominal normal load, *PHX1*, dimensionless.

Variation of horizontal slip ratio with load, PHX2 — Slip variation
0.00115950515263055 (default) | scalar

Variation of horizontal slip ratio with load, *PHX2*, dimensionless.

Vertical shift in load at nominal normal load, PVX1 — Load shift
(default) | scalar | 1.59236164471432e-05

Vertical shift in load at nominal normal load, *PVX1*, dimensionless.

Variation of vertical shift with load, PVX2 — Load variation
0.000104321112127671 (default) | scalar

Variation of vertical shift with load, *PVX2*, dimensionless.

Linear variation of longitudinal slip stiffness with tire pressure, PPX1 — Stiffness variation
-0.349461321276586 (default) | scalar

Linear variation of longitudinal slip stiffness with tire pressure, *PPX1*, dimensionless.

Quadratic variation of longitudinal slip stiffness with tire pressure, PPX2 — Stiffness variation
0.387840040616429 (default) | scalar

Quadratic variation of longitudinal slip stiffness with tire pressure, *PPX2*, dimensionless.

Linear variation of peak longitudinal friction with tire pressure, PPX3 — Friction variation
-0.0969947336569324 (default) | scalar

Linear variation of peak longitudinal friction with tire pressure, *PPX3*, dimensionless.

Quadratic variation of peak longitudinal friction with tire pressure, PPX4 — Friction variation
0.0632271859795801 (default) | scalar

Quadratic variation of peak longitudinal friction with tire pressure, *PPX4*, dimensionless.

Combined slip Fx slope factor reduction, RBX1 — Combined slip longitudinal force slope factor reduction
12.7633329850276 (default) | scalar

Combined slip longitudinal force, F_x, slope factor reduction, *RBX1*, dimensionless.

Slip ratio Fx slope reduction variation, RBX2 — Slip ratio longitudinal force slope reduction variation
9.5787123658471 (default) | scalar

Slip ratio longitudinal force, F_x, slope reduction variation, *RBX2*, dimensionless.

Camber influence on combined slip Fx stiffness, RBX3 — Camber influence on combined slip longitudinal force stiffness
0 (default) | scalar

Camber influence on combined slip longitudinal force, F_x, stiffness, *RBX3*, dimensionless.

Shape factor for combined slip Fx reduction, RCX1 — Shape factor for combined slip longitudinal force reduction
1.02796280922059 (default) | scalar

Shape factor for combined slip longitudinal force, F_x, reduction, *RCX1*, dimensionless.

Combined Fx curvature factor, REX1 — Combined longitudinal force curvature factor
-0.45202516851367 (default) | scalar

Combined longitudinal force, F_x, curvature factor, *REX1*, dimensionless.

Combined Fx curvature factor with load, REX2 — Combined longitudinal force curvature factor
-0.47304886171176 (default) | scalar

Combined longitudinal force, F_x, curvature factor with load, *REX2*, dimensionless.

Combined slip Fx shift factor reduction, RHX1 — Combined slip longitudinal force slip factor
0.000233272373115809 (default) | scalar

Combined slip longitudinal force, F_x, shift factor reduction, *RHX1*, dimensionless.

Overturning

Vertical shift of overturning moment, QSX1 — Overturning moment
-0.00788307788961443 (default) | scalar

Vertical shift of overturning moment, *QSX1*, dimensionless.

Overturning moment due to camber, QSX2 — Overturning moment due to camber
0.954629515459396 (default) | scalar

Overturning moment due to camber, *QSX2*, dimensionless.

Overturning moment due to Fy, QSX3 — Overturning moment due to lateral force
0.0155855777667025 (default) | scalar

Overturning moment due to lateral force, *QSX3*, dimensionless.

Mx combined lateral force load and camber, QSX4 — Overturning moment
0.125904607142414 (default) | scalar

Overturning moment, M_x, moment combined lateral force load and camber, *QSX4*, dimensionless.

Mx load effect due to lateral force and camber, QSX5 — Overturning moment
1.01110069703963 (default) | scalar

Overturning moment, M_x, load effect due to lateral force and camber, *QSX5*, dimensionless.

Mx load effect due to B-factor, QSX6 — Overturning moment
20.1938577118433 (default) | scalar

Overturning moment, M_x, load effect due to B-factor, *QSX6*, dimensionless.

Mx due to camber and load, QSX7 — Overturning moment
0.310818120717083 (default) | scalar

Overturning moment, M_x, due to camber and load, *QSX7*, dimensionless.

Mx due to lateral force and load, QSX8 — Overturning moment
-0.0430823751139655 (default) | scalar

Overturning moment, M_x, due to lateral force and load, QSX8, dimensionless.

Mx due to B-factor of lateral force and load, QSX9 — Overturning moment
0.636494404888216 (default) | scalar

Overturning moment, M_x, due to B-factor of lateral force and load, *QSX9*, dimensionless.

Mx due to vertical force and camber, QSX10 — Overturning moment
0.142438060591663 (default) | scalar

Overturning moment, M_x, due to vertical force and camber, *QSX10*, dimensionless.

Mx due to B-factor of vertical force and camber, QSX11 — Overturning moment
0.346766530036373 (default) | scalar

Overturning moment, M_x, due to B-factor of vertical force and camber, *QSX11*, dimensionless.

Mx due to squared camber, QSX12 — Overturning moment
0 (default) | scalar

Overturning moment, M_x, due to squared camber, *QSX12*, dimensionless.

Mx due to lateral force, QSX13 — Overturning moment
0 (default) | scalar

Overturning moment, M_x, due to lateral force, *QSX13*, dimensionless.

Mx due to lateral force with camber, QSX14 — Overturning moment
0 (default) | scalar

Overturning moment, M_x, due to lateral force with camber, *QSX14*, dimensionless.

Mx due to inflation pressure, PPMX1 — Overturning moment due to pressure
0 (default) | scalar

Overturning moment, M_x, due to inflation pressure, *PPMX1*, dimensionless.

Lateral

Cfy shape factor for lateral force, PCY1 — Lateral force shape factor
1.34299950037631 (default) | scalar

Shape factor for lateral force, C_{fy}, *PCY1*, dimensionless.

Lateral friction muy, PDY1 — Lateral friction
0.878267729082195 (default) | scalar

Lateral friction, μ_y, *PDY1*, dimensionless.

Lateral friction variation of muy with load, PDY2 — Lateral friction variation
-0.0644597923147385 (default) | scalar

Variation of lateral friction, μ_y, with load, *PDY2*, dimensionless.

Lateral friction variation of muy with squared camber, PDY3 — Lateral friction variation
0 (default) | scalar

Variation of lateral friction, μ_y, with squared camber, *PDY3*, dimensionless.

Efy lateral curvature at nominal force FZNOM, PEY1 — Lateral curvature at nominal force
-0.809776534470972 (default) | scalar

Lateral curvature, Ef_y, at nominal force, F_{ZNOM}, *PEY1*, dimensionless.

Efy curvature variation with load PEY2 — Lateral curvature variation
-0.600180598867198 (default) | scalar

Lateral curvature, Ef_y, variation with load, *PEY2*, dimensionless.

Efy curvature constant camber dependency, PEY3 — Lateral curvature constant
0.0991732552633487 (default) | scalar

Lateral curvature, Ef_y, constant camber dependency, *PEY3*, dimensionless.

Efy curvature variation with camber, PEY4 — Lateral curvature variation
-6.5577974035573 (default) | scalar

Lateral curvature, Ef_y, variation with camber, *PEY4*, dimensionless.

Efy curvature variation with camber squared, PEY5 — Lateral curvature variation
0 (default) | scalar

Lateral curvature, Ef_y, variation with camber squared, *PEY5*, dimensionless.

Maximum KFy/FZNOM stiffness, PKY1 — Maximum stiffness
-15.5714726518315 (default) | scalar

Maximum lateral force stiffness, KF_y, to nominal force, F_{ZNOM}, ratio, *PKY1*, dimensionless.

Load at maximum KFy/FZNOM stiffness, PKY2 — Load
1.73126522291751 (default) | scalar

Load at maximum lateral force stiffness, KF_y, to nominal force, F_{ZNOM}, ratio, *PKY2*, dimensionless.

KFy/FZNOM stiffness variation with camber, PKY3 — Stiffness variation
0.365349873517229 (default) | scalar

Lateral force stiffness, KF_y, to nominal force, F_{ZNOM}, stiffness variation with camber, *PKY3*, dimensionless.

KFy curvature, PKY4 — Lateral force stiffness curvature
1.98176755955416 (default) | scalar

Lateral force stiffness, KF_y curvature, *PKY4*, dimensionless.

Variation of peak stiffness with squared camber, PKY5 — Stiffness variation
0 (default) | scalar

Variation of peak stiffness with squared camber, *PKY5*, dimensionless.

Fy camber stiffness factor, PKY6 — Lateral force camber stiffness factor
-0.884005199550313 (default) | scalar

Lateral force, F_y, camber stiffness factor, *PKY6*, dimensionless.

Camber stiffness vertical load dependency, PKY7 — Stiffness
-0.237259727611847 (default) | scalar

Camber stiffness vertical load dependency, *PKY7*, dimensionless.

SHY horizontal shift at FZNOM, PHY1 — Horizontal shift at nominal force
-0.00183370557235628 (default) | scalar

Horizontal shift, S_{HY}, at nominal force, F_{ZNOM}, *PHY1*, dimensionless.

SHY variation with load, PHY2 — Horizontal shift variation
0.00346401302716671 (default) | scalar

Horizontal shift, S_{HY}, variation with load, *PHY2*, dimensionless.

Svy/Fz vertical shift at FZNOM, PVY1 — Vertical shift at nominal force
-0.00675427560163264 (default) | scalar

Vertical shift, S_{vy}, at nominal force, F_{ZNOM}, *PVY1*, dimensionless.

Svy/Fz variation with load, PVY2 — Vertical shift variation with load
0.036379218103843 (default) | scalar

Vertical shift, S_{vy}, variation with load, *PVY2*, dimensionless.

Svy/Fz variation with camber, PVY3 — Vertical shift variation with camber
-0.163543984089166 (default) | scalar

Vertical shift, S_{vy}, variation with camber, *PVY3*, dimensionless.

Svy/Fz variation with load and camber, PVY4 — Vertical shift variation with load and camber
-0.491003349625726 (default) | scalar

Vertical shift, S_{vy}, variation with load and camber, *PVY4*, dimensionless.

Cornering stiffness variation with inflation pressure, PPY1 — Stiffness variation with pressure
-0.62059646553185 (default) | scalar

Cornering stiffness variation with inflation pressure, *PPY1*, dimensionless.

Cornering stiffness variation with inflation pressure induced nominal load dependency, PPY2 — Stiffness variation with pressure
-0.0647824842338686 (default) | scalar

Cornering stiffness variation with inflation pressure induced nominal load dependency, *PPY2*, dimensionless.

Linear inflation pressure on peak lateral friction, PPY3 — Pressure
-0.164648843290686 (default) | scalar

Linear inflation pressure on peak lateral friction, *PPY3*, dimensionless.

Quadratic inflation pressure on peak lateral friction, PPY4 — Pressure
0.283193909060098 (default) | scalar

Quadratic inflation pressure on peak lateral friction, *PPY4*, dimensionless.

Inflation pressure effect on camber stiffness, PPY5 — Pressure
0 (default) | scalar

Inflation pressure effect on camber stiffness, *PPY5*, dimensionless.

Combined Fy reduction slope factor, RBY1 — Combined lateral force reduction slope factor
10.7588809173639 (default) | scalar

Combined lateral force, F_y, reduction slope factor, *RBY1*, dimensionless.

Fy slope reduction with slip angle, RBY2 — Lateral force slope reduction with slip angle
7.70420065295029 (default) | scalar

Lateral force, F_y, slope reduction with slip angle, *RBY2*, dimensionless.

Fy shift reduction with slip angle, RBY3 — Lateral force shift reduction with slip angle
3.40283755070691e-06 (default) | scalar

Lateral force, F_y, shift reduction with slip angle, *RBY3*, dimensionless.

Fy combined stiffness variation from camber, RBY4 — Lateral force combined stiffness variation from camber
0 (default) | scalar

Lateral force, F_y, combined stiffness variation from camber, *RBY4*, dimensionless.

Fy combined reduction shape factor, RCY1 — Lateral force combined reduction shape factor
1.06691874059059 (default) | scalar

Lateral force, F_y, combined reduction shape factor, *RCY1*, dimensionless.

Fy combined curvature factor, REY1 — Lateral force combined curvature factor
0.308244924574305 (default) | scalar

Lateral force, F_y, combined curvature factor, *REY1*, dimensionless.

Fy combined curvature factor with load, REY2 — Lateral force combined curvature factor with load
7.48775032604508e-06 (default) | scalar

Lateral force, F_y, combined curvature factor with load, *REY2*, dimensionless.

Fy combined reduction shift factor, RHY1 — Lateral force combined reduction shift factor
0.0118479204960415 (default) | scalar

Lateral force, F_y, combined reduction shift factor, *RHY1*, dimensionless.

Fy combined reduction shift factor with load, RHY2 — Lateral force combined reduction shift factor with load
7.63650177161071e-06 (default) | scalar

3-69

Lateral force, F_y, combined reduction shift factor with load, *RHY2*, dimensionless.

Slip ratio side force Svyk/Muy*Fz at FZNOM, RVY1 — Slip ratio slide force at nominal force
0.0560024425050078 (default) | scalar

Slip ratio side force at nominal force, F_{ZNOM}, *RVY1*, dimensionless.

Side force Svyk/Muy*Fz variation with load, RVY2 — Side force variation with load
7.48487127056197e-06 (default) | scalar

Side force variation with load, *RVY2*, dimensionless.

Side force Svyk/Muy*Fz variation with camber, RVY3 — Side force variation with camber
0 (default) | scalar

Side force variation with camber, *RVY3*, dimensionless.

Side force Svyk/Muy*Fz variation with slip angle, RVY4 — Side force variation with slip angle
98.4047026517149 (default) | scalar

Side force variation with slip angle, *RVY4*, dimensionless.

Side force Svyk/Muy*Fz variation with slip ratio, RVY5 — Side force variation with slip ratio
2.02759273900054 (default) | scalar

Side force variation with slip ratio, *RVY5*, dimensionless.

Side force Svyk/Muy*Fz variation with slip ratio arctangent, RVY6 — Side force variation with slip ratio arctangent
15.7623064923347 (default) | scalar

Side force variation with slip ratio arctangent, *RVY6*, dimensionless.

Rolling

Torque resistance coefficient, QSY1 — Torque resistance
0.00699077564863536 (default) | scalar

Torque resistance coefficient, *QSY1*, dimensionless.

Torque resistance due to Fx, QSY2 — Torque resistance due to longitudinal force
0 (default) | scalar

Torque resistance due to longitudinal force, F_x, *QSY2*, dimensionless.

Torque resistance due to speed, QSY3 — Torque resistance due to speed
(default) | scalar | 0.00153535478298811

Torque resistance due to speed, *QSY3*, dimensionless.

Torque resistance due to speed^4, QSY4 — Torque resistance due to speed
8.55710322455322e-05 (default) | scalar

Torque resistance due to speed^4, *QSY4*, dimensionless.

Torque resistance due to square of camber, QSY5 — Torque resistance due to camber
0 (default) | scalar

Torque resistance due to square of camber, *QSY5*, dimensionless.

Torque resistance due to square of camber and load, QSY6 — Torque resistance due to camber and load
0 (default) | scalar

Torque resistance due to square of camber and load, *QSY6*, dimensionless.

Torque resistance due to load, QSY7 — Torque resistance due to load
0.89443879773793 (default) | scalar

Torque resistance due to load, *QSY7*, dimensionless.

Torque resistance due to pressure, QSY8 — Torque resistance due to pressure
-0.403227641260894 (default) | scalar

Torque resistance due to pressure, *QSY8*, dimensionless.

Aligning

Trail slope factor for trail Bpt at FZNOM, QBZ1 — Trail slope factor at nominal force
11.666656027484 (default) | scalar

Trail slope factor for trail Bpt at nominal force, F_{ZNOM}, *QBZ1*, dimensionless.

Bpt slope variation with load, QBZ2 — Slope variation with load
-1.30355438574485 (default) | scalar

Slope variation with load, *QBZ2*, dimensionless.

Bpt slope variation with square of load, QBZ3 — Slope variation with load
0 (default) | scalar

Slope variation with square of load, *QBZ3*, dimensionless.

Bpt slope variation with camber, QBZ4 — Slope variation with camber
0.177918305399071 (default) | scalar

Slope variation with camber, *QBZ4*, dimensionless.

Bpt slope variation with absolute value of camber, QBZ5 — Slope variation with camber
-0.162481135068033 (default) | scalar

Slope variation with absolute value of camber, *QBZ5*, dimensionless.

Bpt slope variation with square of camber, QBZ6 — Slope variation with camber
0 (default) | scalar

Slope variation with square of camber, *QBZ6*, dimensionless.

Br of Mzr slope scaling factor, QBZ9 — Slope scaling factor
30.1445534682089 (default) | scalar

Slope scaling factor, *QBZ9*, dimensionless.

Br of Mzr cornering stiffness factor, QBZ10 — Cornering stiffness factor
0 (default) | scalar

Br of Mzr cornering stiffness factor, *QBZ10*, dimensionless.

Cpt pneumatic trail shape factor, QCZ1 — Pneumatic trail shape factor
1.29026978207574 (default) | scalar

Pneumatic trail shape factor, C_{pt}, *QCZ1*, dimensionless.

Dpt peak trail, QDZ1 — Peak trail
0.0886972644991399 (default) | scalar

Peak trail, D_{pt}, *QDZ1*, dimensionless.

Dpt peak trail variation with load, QDZ2 — Peak trail variation with load
-0.0058634820161856 (default) | scalar

Peak trail, D_{pt}, variation with load, *QDZ2*, dimensionless.

Dpt peak trail variation with camber, QDZ3 — Peak trail variation with camber
0.3805597481383 (default) | scalar

Peak trail, D_{pt}, variation with camber, *QDZ3*, dimensionless.

Dpt peak trail variation with square of camber, QDZ4 — Peak trail variation with camber
0 (default) | scalar

Peak trail, D_{pt}, variation with square of camber, *QDZ4*, dimensionless.

Dmr peak residual torque, QDZ6 — Peak residual torque
0.00160914918118959 (default) | scalar

Peak residual torque, D_{mr}, *QDZ6*, dimensionless.

Dmr peak residual torque variation with load, QDZ7 — Peak residual torque variation with load
-0.00211036330862154 (default) | scalar

Peak residual torque, D_{mr}, variation with load, *QDZ7*, dimensionless.

Dmr peak residual torque variation with camber, QDZ8 — Peak residual torque variation with camber
-0.13921904912557 (default) | scalar

Peak residual torque, D_{mr}, variation with camber, *QDZ8*, dimensionless.

Dmr peak residual torque variation with camber and load, QDZ9 — Peak residual torque variation with camber and load
0.00816585550028243 (default) | scalar

Peak residual torque, D_{mr}, variation with camber and load, *QDZ9*, dimensionless.

Dmr peak residual torque variation with square of camber, QDZ10 — Peak residual torque variation with camber
0 (default) | scalar

Peak residual torque, D_{mr}, variation with square of camber, *QDZ10*, dimensionless.

Dmr peak residual torque variation with square of load, QDZ11 — Peak residual torque variation with load
0 (default) | scalar

Peak residual torque, D_{mr}, variation with square of load, *QDZ11*, dimensionless.

Ept trail curvature at FZNOM, QEZ1 — Trail curvature at nominal force
-2.19802254534746 (default) | scalar

Trail curvature, E_{pt}, at nominal force, F_{ZNOM}, *QEZ1*, dimensionless.

Ept variation with load, QEZ2 — Trail curvature variation with load
1.01850535789165 (default) | scalar

Trail curvature, E_{pt} variation with load, *QEZ2*, dimensionless.

Ept variation with square of load, QEZ3 — Trail curvature variation with load
0 (default) | scalar

Trail curvature, E_{pt} variation with square of load, *QEZ3*, dimensionless.

Ept variation with sign of alpha-t, QEZ4 — Trail curvature variation
0.278379679112054 (default) | scalar

Trail curvature, E_{pt} variation with sign of alpha-t, *QEZ4*, dimensionless.

Ept variation with sign of alpha-t and camber, QEZ5 — Variation
-0.548414722765386 (default) | scalar

Trail curvature, E_{pt} variation with sign of alpha-t and camber, *QEZ5*, dimensionless.

Sht horizontal trail shift at FZNOM, QHZ1 — Horizontal trail shift at nominal load
0.00148361198869669 (default) | scalar

Horizontal trail shift, Sh_t, at nominal load, F_{ZNOM}, *QHZ1*, dimensionless.

Sht variation with load, QHZ2 — Horizontal trail shift variation with load
0.00202017567585904 (default) | scalar

Horizontal trail shift, Sh_t, variation with load, *QHZ2*, dimensionless.

Sht variation with camber, QHZ3 — Horizontal trail shift variation with camber
0.240316641332066 (default) | scalar

Horizontal trail shift, Sh_t, variation with camber, *QHZ3*, dimensionless.

Sht variation with load and camber, QHZ4 — Horizontal trail shift variation with load and camber
-0.210114035738085 (default) | scalar

Horizontal trail shift, Sh_t, variation with load and camber, *QHZ4*, dimensionless.

Inflation pressure influence on trail length, PPZ1 — Pressure influence on trail length
-0.443927982388615 (default) | scalar

Inflation pressure influence on trail length, *PPZ1*, dimensionless.

Inflation pressure influence on residual aligning torque, PPZ2 — Pressure influence on aligning torque
0 (default) | scalar

Inflation pressure influence on residual aligning torque, *PPZ2*, dimensionless.

Nominal value of s/R0: effect of Fx on Mz, SSZ1 — Effect of longitudinal force on aligning torque
0.00914999585397887 (default) | scalar

Nominal value of s/R0: effect of longitudinal force, F_x, on aligning torque, M_z, *SSZ1*, dimensionless.

s/R0 variation with lateral to nominal force ratio, SSZ2 — Variation with lateral to nominal force ratio
0.0389830656052735 (default) | scalar

Variation with lateral to nominal force ratio, *SSZ2*, dimensionless.

s/R0 variation with camber, SSZ3 — Variation with camber
0 (default) | scalar

Variation with camber, *SSZ3*, dimensionless.

s/R0 variation with camber and load, SSZ4 — Variation with camber and load
0 (default) | scalar

Variation with camber and load, *SSZ4*, dimensionless.

Turnslip

Fx peak reduction due to spin, PDXP1 — Longitudinal force peak reduction due to spin
0.396718274678791 (default) | scalar

Longitudinal force, F_x, peak reduction due to spin, *PDXP1*, dimensionless.

Fx peak reduction due to spin with varying load, PDXP2 — Longitudinal force peak reduction due to spin
0 (default) | scalar

Longitudinal force, F_x, peak reduction due to spin with varying load, *PDXP2*, dimensionless.

Fx peak reduction due to spin with slip ratio, PDXP3 — Longitudinal force peak reduction due to spin
0 (default) | scalar

Longitudinal force, F_x, peak reduction due to spin with slip ratio, *PDXP3*, dimensionless.

Cornering stiffness reduction due to spin, PKYP1 — Stiffness reduction due to spin
1.00587060633573 (default) | scalar

Cornering stiffness reduction due to spin, *PKYP1*, dimensionless.

Fy peak reduction due to spin, PDYP1 — Lateral force peak reduction due to spin
0.40483048923438 (default) | scalar

Lateral force, F_y, peak reduction due to spin, *PDYP1*, dimensionless.

Fy peak reduction due to spin with varying load, PDYP2 — Lateral force peak reduction due to spin
0 (default) | scalar

Lateral force, F_y, peak reduction due to spin with varying load, *PDYP2*, dimensionless.

Fy peak reduction due to spin with slip angle, PDYP3 — Lateral force peak reduction due to spin
0 (default) | scalar

Lateral force, F_y, peak reduction due to spin with slip angle, *PDYP3*, dimensionless.

Fy peak reduction due to square root of spin, PDYP4 — Lateral force peak reduction due to spin
0 (default) | scalar

Lateral force, F_y, peak reduction due to square root of spin, *PDYP4*, dimensionless.

Fy vs. slip angle response lateral shift limit, PHYP1 — Lateral force versus slip angle response
0.992731942067188 (default) | scalar

Lateral force, F_y, versus slip angle response lateral shift limit, *PHYP1*, dimensionless.

Fy vs. slip angle response max lateral shift limit, PHYP2 — Lateral force versus slip angle response
0.150352945499914 (default) | scalar

Lateral force, F_y, versus slip angle response max lateral shift limit, *PHYP2*, dimensionless.

Fy vs. slip angle response max lateral shift limit with load, PHYP3 — Lateral force versus slip angle response
0 (default) | scalar

Lateral force, F_y, versus slip angle response max lateral shift limit with load, *PHYP3*, dimensionless.

Fy vs. slip angle response lateral shift curvature factor, PHYP4 — Lateral force versus slip angle response
-3.98988682633674 (default) | scalar

Lateral force, F_y, versus slip angle response lateral shift curvature factor, *PHYP4*, dimensionless.

Camber stiffness reduction due to spin, PECP1 — Camber stiffness reduction
0.502379254513503 (default) | scalar

Camber stiffness reduction due to spin, *PECP1*, dimensionless.

Camber stiffness reduction due to spin with load, PECP2 — Camber stiffness reduction
0 (default) | scalar

Camber stiffness reduction due to spin with load, *PECP2*, dimensionless.

Turn slip pneumatic trail reduction factor, QDTP1 — Turn slip pneumatic trail reduction factor
10.0178654287947 (default) | scalar

Turn slip pneumatic trail reduction factor, *QDTP1*, dimensionless.

Turn moment for constant turning and zero longitudinal speed, QCRP1 — Turn moment for constant turning
0.200627747200702 (default) | scalar

Turn moment for constant turning and zero longitudinal speed, *QCRP1*, dimensionless.

Turn slip moment increase with spin at 90deg slip angle, QCRP2 — Turn slip moment
0.0995735881599772 (default) | scalar

Turn slip moment increase with spin at 90-degree slip angle, *QCRP2*, dimensionless.

Residual spin torque reduction from side slip, QBRP1 — Residual spin torque reduction
0.0997107127960717 (default) | scalar

Residual spin torque reduction from side slip, *QBRP1*, dimensionless.

Turn slip moment peak magnitude, QDRP1 — Turn slip moment peak magnitude
1.01432910167423 (default) | scalar

Turn slip moment peak magnitude, *QDRP1*, dimensionless.

Turn slip moment curvature, QDRP2 — Turn slip moment curvature
0 (default) | scalar

Turn slip moment curvature, *QDRP2*, dimensionless.

References

[1] Besselink, I. J, M., A. J. C. Schmeitz, and H. B. Pacejka. "An improved Magic Formula/Swift tyre model that can handle inflation pressure changes." *Vehicle System Dynamics - International Journal of Vehicle Mechanics and Mobility*. Vol. 48, 2010. doi: 10.1080/00423111003748088.

[2] Pacejka, H. B. *Tire and Vehicle Dynamics*. 3rd ed. Oxford, United Kingdom: SAE and Butterworth-Heinemann, 2012.

[3] Schmid, Steven R., Bernard J. Hamrock, and Bo O. Jacobson. "Chapter 18: Brakes and Clutches." *Fundamentals of Machine Elements, SI Version*. 3rd ed. Boca Raton, FL: CRC Press, 2014.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Longitudinal Wheel

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2018a

Fiala Wheel 2DOF

Fiala wheel 2DOF wheel with disc, drum, or mapped brake
Library: Vehicle Dynamics Blockset / Wheels and Tires
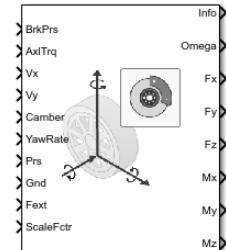
Description

The Fiala Wheel 2DOF block implements a simplified tire with lateral and longitudinal slip capability based on the E. Fiala model[1]. The block uses a translational friction model to calculate the forces and moments during combined longitudinal and lateral slip, requiring fewer parameters than the Combined Slip Wheel 2DOF block. If you do not have the tire coefficients needed by the Magic Formula, consider using this block for studies that do not involve extensive nonlinear combined lateral slip or lateral dynamics. If your study does require nonlinear combined slip or lateral dynamics, consider using the Combined Slip Wheel 2DOF block.

The block determines the wheel rotation rate, vertical motion, and forces and moments in all six degrees-of-freedom (DOFs) based on the driveline torque, brake pressure, road height, wheel camber angle, and inflation pressure. You can use this block for these types of analyses:

- Driveline and vehicle simulations that require low frequency tire-road and braking forces for vehicle acceleration, braking, and wheel rolling resistance calculations with minimal tire parameters.

- Wheel interaction with an idealized road surface.

- Ride and handling maneuvers for vehicles undergoing mild combined slip. For this analysis, you can connect the block to driveline and chassis components such as differentials, suspension, and vehicle body systems.

- Yaw stability. For this analyses, you can connect this block to more detailed braking system models.

- Tire stiffness and unsprung mass interactions with ground variations, load transfer, or chassis motion using the block vertical DOF.

The block integrates rotational wheel, vertical mass, and braking dynamics models. For the slip-dependent tire forces and moments, the block implements the Fiala tire model.

Use the **Brake Type** parameter to select the brake.

Brake Type Setting	Brake Implementation
None	None
Disc	Brake that converts the brake cylinder pressure into a braking force

Brake Type Setting	Brake Implementation
Drum	Simplex drum brake that converts the applied force and brake geometry into a net braking torque
Mapped	Lookup table that is a function of the wheel speed and applied brake pressure

To calculate the rolling resistance torque, specify one of these **Rolling Resistance** parameters.

Setting	Block Implementation
None	None
Pressure and velocity	Method in *Stepwise Coastdown Methodology for Measuring Tire Rolling Resistance.* The rolling resistance is a function of tire pressure, normal force, and velocity.
ISO 28580	Method specified in ISO 28580:2018, *Passenger car, truck and bus tyre rolling resistance measurement method — Single point test and correlation of measurement results.*
Magic Formula	Magic formula equations from 4.E70 in *Tire and Vehicle Dynamics*. The magic formula is an empirical equation based on fitting coefficients.
Mapped torque	Lookup table that is a function of the normal force and spin axis longitudinal velocity.

To calculate vertical motion, specify one of these **Vertical Motion** parameters.

Setting	Block Implementation
None	Block passes the applied chassis forces directly through to the rolling resistance and longitudinal force calculations.
Mapped stiffness and damping	Vertical motion depends on wheel stiffness and damping. Stiffness is a function of tire sidewall displacement and pressure. Damping is a function of tire sidewall velocity and pressure.

Rotational Wheel Dynamics

The block calculates the inertial response of the wheel subject to:

- Axle losses
- Brake and drive torque
- Tire rolling resistance
- Ground contact through the tire-road interface

The input torque is the summation of the applied axle torque, braking torque, and moment arising from the combined tire torque.

$$T_i = T_a - T_b + T_d$$

For the moment arising from the combined tire torque, the block implements tractive wheel forces and rolling resistance with first-order dynamics. The rolling resistance has a time constant parameterized in terms of a relaxation length.

$$T_d(s) = \frac{1}{\frac{|\omega| R_e}{L_e} s + 1}(F_x R_e + M_y)$$

To calculate the rolling resistance torque, you can specify one of these **Rolling Resistance** parameters.

Setting	Block Implementation
None	Block sets rolling resistance, M_y, to zero.
Pressure and velocity	Block uses the method in SAE *Stepwise Coastdown Methodology for Measuring Tire Rolling Resistance*. The rolling resistance is a function of tire pressure, normal force, and velocity. Specifically, $$M_y = R_e\{a + b|V_x| + cV_x{}^2\}\{F_z{}^\beta p_i{}^\alpha\}\tanh(4V_x)$$
ISO 28580	Block uses the method specified in ISO 28580:2018, *Passenger car, truck and bus tyre rolling resistance measurement method — Single point test and correlation of measurement results*. The method accounts for normal load, parasitic loss, and thermal corrections from test conditions. Specifically, $$M_y = R_e(\frac{F_z C_r}{1 + K_t(T_{amb} - T_{meas})} - F_{pl})\tanh(\omega)$$
Magic Formula	Block calculates the rolling resistance, M_y, using the Magic Formula equations from 4.E70 in *Tire and Vehicle Dynamics*. The magic formula is an empirical equation based on fitting coefficients.
Mapped torque	For the rolling resistance, M_y, the block uses a lookup table that is a function of the normal force and spin axis longitudinal velocity.

If the brakes are enabled, the block determines the braking locked or unlocked condition based on an idealized dry clutch friction model. Based on the lock-up condition, the block implements these friction and dynamic models.

If	Lock-Up Condition	Friction Model	Dynamic Model
$\omega \neq 0$ or $T_S < \left|T_i + T_f - \omega b\right|$	Unlocked	$T_f = T_k$ where, $T_k = F_c R_{eff}\mu_k\tanh[4(-\omega_d)]$ $T_s = F_c R_{eff}\mu_s$ $R_{eff} = \dfrac{2(R_o{}^3 - R_i{}^3)}{3(R_o{}^2 - R_i{}^2)}$	$\dot{\omega}J = -\omega b + T_i + T_o$
$\omega = 0$ and $T_S \geq \left|T_i + T_f - \omega b\right|$	Locked	$T_f = T_s$	$\omega = 0$

The equations use these variables.

ω	Wheel angular velocity
a	Velocity-independent force component

b	Linear velocity force component
c	Quadratic velocity force component
L_e	Tire relaxation length
J	Moment of inertia
M_y	Rolling resistance torque
T_a	Applied axle torque
T_b	Braking torque
T_d	Combined tire torque
T_f	Frictional torque
T_i	Net input torque
T_k	Kinetic frictional torque
T_o	Net output torque
T_s	Static frictional torque
F_c	Applied clutch force
F_x	Longitudinal force developed by the tire road interface due to slip
R_{eff}	Effective clutch radius
R_o	Annular disk outer radius
R_i	Annular disk inner radius
R_e	Effective tire radius while under load and for a given pressure
V_x	Longitudinal axle velocity
F_z	Vehicle normal force
C_r	Rolling resistance constant
T_{amb}	Ambient temperature
T_{meas}	Measured temperature for rolling resistance constant
F_{pl}	Parasitic force loss
K_t	Thermal correction factor
α	Tire pressure exponent
β	Normal force exponent
p_i	Tire pressure
μ_s	Coefficient of static friction
μ_k	Coefficient of kinetic friction

Longitudinal Force

The block implements the longitudinal force as a function of wheel slip relative to the road surface using these equations.

Calculation	Equation
Critical slip	$\kappa'_{Critical} = \left\| \dfrac{\mu F_z}{2C_K} \right\|$

Calculation	Equation
Longitudinal force	$$F_x = \begin{cases} C_k \ \ \kappa' & \text{when } \lvert\kappa'\rvert \le \kappa'_{Critical} \\ \tanh(4\kappa')\left(\mu\lvert F_z\rvert - \left\lvert\dfrac{(\mu F_z)^2}{4\kappa' C_\kappa}\right\rvert\right) & \text{when } \lvert\kappa'\rvert > \kappa'_{Critical} \end{cases}$$
Friction coefficient	$\mu = (\mu_s - (\mu_s - \mu_k) \ \kappa_{k\alpha})\lambda_\mu$
Slip coefficient	$\kappa_{k\alpha} = \sqrt{\kappa'^2 + \tan^2(\alpha')}$

The equations use these variables.

κ'	Slip state
F_x	Longitudinal force acting on axle along tire-fixed x-axis,
C_κ	Longitudinal stiffness
F_z	Vertical contact patch normal force along tire-fixed z-axis,
μ	Friction coefficient
μ_s	Coefficient of static friction
μ_k	Coefficient of kinetic friction
$\kappa_{k\alpha}$	Comprehensive slip coefficient
α'	Slip angle state
λ_μ	Friction scaling

Lateral Force

The block implements the lateral force as a function of wheel slip angle state using these equations.

Calculation	Equation
Critical slip angle	$\alpha'_{Critical} = \text{atan}(\dfrac{3\mu\lvert F_z\rvert}{C_a})$
Lateral force	$$F_y = \begin{cases} -\tanh(4\alpha')\mu\lvert F_z\rvert & \text{when } \lvert\alpha'\rvert > \alpha'_{Critical} \\ -\tanh(4\alpha')\mu\lvert F_z\rvert\left(1 - \xi^3\right) + \gamma C_\gamma & \text{when } \lvert\alpha'\rvert \le \alpha'_{Critical} \end{cases}$$ $$\xi = 1 - \dfrac{C_a\lvert\tan(\alpha')\rvert}{3\mu\lvert F_z\rvert}$$

The equations use these variables.

α'	Slip angle state
F_y	Lateral force acting on axle along tire-fixed y-axis,
F_z	Vertical contact patch normal force along tire-fixed z-axis
C_γ	Camber stiffness
C_α	Lateral stiffness per slip angle
μ	Friction coefficient

Vertical Dynamics

For the vertical dynamics, the block implements these equations.

Calculation	Equation
Vertical response	$\ddot{z}m = F_{ztire} + mg - Fz$
Tire normal force	$F_{ztire} = \rho_z k - b\dot{z}$
Vertical sidewall deflection	$\rho_z = z_{gnd} - z, z \geq 0$

The equations use these variables.

z	Tire deflection along tire-fixed z-axis
z_{gnd}	Ground displacement along tire-fixed z-axis
F_{ztire}	Tire normal force along tire-fixed z-axis
F_z	Vertical force acting on axle along tire-fixed z-axis
ρ_z	Vertical sidewall deflection along tire-fixed z-axis
k	Vertical sidewall stiffness
b	Vertical sidewall damping

Overturning, Aligning, and Scaling

This table summarizes the overturning, aligning, and scaling implementation.

Calculation	Implementation
Overturning moment	The Fiala model does not define an overturning moment. The block implements this equation, requiring minimal parameters. $$M_x = F_y R_e \cos(\gamma)$$
Aligning moment	The block implements the aligning moment as a combination of yaw rate damping and slip angle state. $$M_z = \begin{cases} \dot{\psi}b_{M_z} & \text{when } \lvert\alpha'\rvert > \alpha'_{Critical} \\ \tanh(4\alpha')w\mu\lvert F_z\rvert(1-\xi)\xi^3 + \dot{\psi}b_{M_z} & \text{when } \lvert\alpha'\rvert \leq \alpha'_{Critical} \end{cases}$$ $$\xi = 1 - \frac{C_a\lvert\tan(\alpha')\rvert}{3\mu\lvert F_z\rvert}$$
Friction scaling	To vary the coefficient of friction, use the `ScaleFctr` input port.

The equations use these variables.

M_x	Overturning moment acting on axle about tire-fixed x-axis
M_z	Aligning moment acting on axle about tire-fixed z-axis
R_e	Effective contact patch to wheel carrier radial distance
γ	Camber angle
k	Vertical sidewall stiffness

b	Vertical sidewall damping
$\dot{\psi}$	Tire angular velocity about the tire-fixed z-axis (yaw rate)
w	Tire width
α'	Slip angle state
b_{Mz}	Linear yaw rate resistance
F_y	Lateral force acting on axle along tire-fixed y-axis
C_γ	Camber stiffness
C_α	Lateral stiffness per slip angle
μ	Friction coefficient
F_z	Vertical contact patch normal force along tire-fixed z-axis

Tire and Wheel Coordinate Systems

To resolve the forces and moments, the block uses the Z-Up orientation of the tire and wheel coordinate systems.

- Tire coordinate system axes (X_T, Y_T, Z_T) are fixed in a reference frame attached to the tire. The origin is at the tire contact with the ground.

- Wheel coordinate system axes (X_W, Y_W, Z_W) are fixed in a reference frame attached to the wheel. The origin is at the wheel center.

Z-Up Orientation[2]
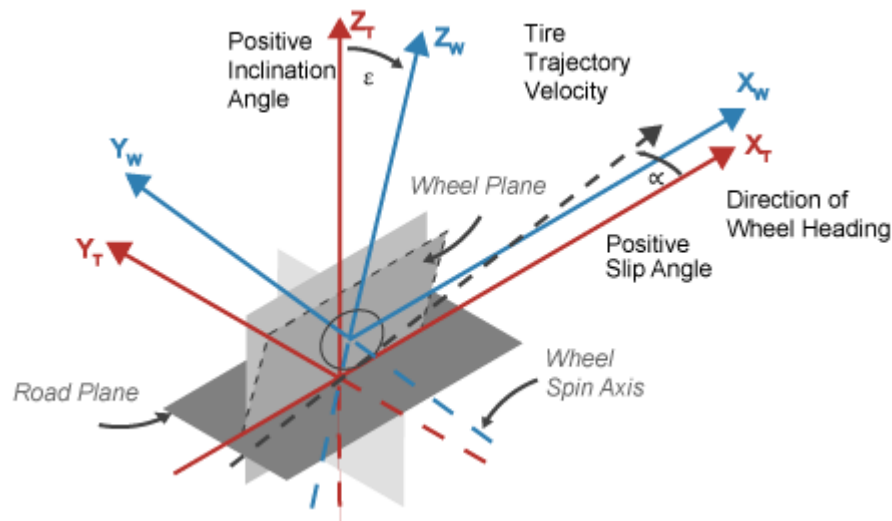
Brakes

Disc

If you specify the **Brake Type** parameter `Disc`, the block implements a disc brake. This figure shows the side and front views of a disc brake.

---

2. Reprinted with permission Copyright © 2008 SAE International. Further distribution of this material is not permitted without prior permission from SAE.

A disc brake converts brake cylinder pressure from the brake cylinder into force. The disc brake applies the force at the brake pad mean radius.

The block uses these equations to calculate brake torque for the disc brake.

$$T = \begin{cases} \dfrac{\mu P \pi B_a{}^2 R_m N_{pads}}{4} & \text{when } N \neq 0 \\[2mm] \dfrac{\mu_{static} P \pi B_a{}^2 R_m N_{pads}}{4} & \text{when } N = 0 \end{cases}$$

$$Rm = \frac{Ro + Ri}{2}$$

The equations use these variables.

T	Brake torque
P	Applied brake pressure
N	Wheel speed
N_{pads}	Number of brake pads in disc brake assembly
μ_{static}	Disc pad-rotor coefficient of static friction
μ	Disc pad-rotor coefficient of kinetic friction
B_a	Brake actuator bore diameter
R_m	Mean radius of brake pad force application on brake rotor
R_o	Outer radius of brake pad

R_i Inner radius of brake pad

Drum

If you specify the **Brake Type** parameter `Drum`, the block implements a static (steady-state) simplex drum brake. A simplex drum brake consists of a single two-sided hydraulic actuator and two brake shoes. The brake shoes do not share a common hinge pin.

The simplex drum brake model uses the applied force and brake geometry to calculate a net torque for each brake shoe. The drum model assumes that the actuators and shoe geometry are symmetrical for both sides, allowing a single set of geometry and friction parameters to be used for both shoes.

The block implements equations that are derived from these equations in *Fundamentals of Machine Elements*.

$$T_{rshoe} = \left(\frac{\pi \mu c r (\cos\theta_2 - \cos\theta_1) B_a 2}{2\mu(2r(\cos\theta_2 - \cos\theta_1) + a(\cos^2\theta_2 - \cos^2\theta_1)) + ar(2\theta_1 - 2\theta_2 + \sin2\theta_2 - \sin2\theta_1)} \right) P$$

$$T_{lshoe} = \left(\frac{\pi \mu c r (\cos\theta_2 - \cos\theta_1) B_a 2}{-2\mu(2r(\cos\theta_2 - \cos\theta_1) + a(\cos^2\theta_2 - \cos^2\theta_1)) + ar(2\theta_1 - 2\theta_2 + \sin2\theta_2 - \sin2\theta_1)} \right) P$$

$$T = \begin{cases} T_{rshoe} + T_{lshoe} & \text{when } N \neq 0 \\ (T_{rshoe} + T_{lshoe}) \frac{\mu_{static}}{\mu} & \text{when } N = 0 \end{cases}$$

Rotation

The equations use these variables.

T	Brake torque
P	Applied brake pressure
N	Wheel speed

μ_{static}	Disc pad-rotor coefficient of static friction
μ	Disc pad-rotor coefficient of kinetic friction
T_{rshoe}	Right shoe brake torque
T_{lshoe}	Left shoe brake torque
a	Distance from drum center to shoe hinge pin center
c	Distance from shoe hinge pin center to brake actuator connection on brake shoe
r	Drum internal radius
B_a	Brake actuator bore diameter
Θ_1	Angle from shoe hinge pin center to start of brake pad material on shoe
Θ_2	Angle from shoe hinge pin center to end of brake pad material on shoe

Mapped

If you specify the **Brake Type** parameter `Mapped`, the block uses a lookup table to determine the brake torque.

$$T = \begin{cases} f_{brake}(P, N) & \text{when } N \neq 0 \\ \left(\frac{\mu_{static}}{\mu}\right) f_{brake}(P, N) & \text{when } N = 0 \end{cases}$$

The equations use these variables.

T	Brake torque
$f_{brake}(P, N)$	Brake torque lookup table
P	Applied brake pressure
N	Wheel speed
μ_{static}	Friction coefficient of drum pad-face interface under static conditions
μ	Friction coefficient of disc pad-rotor interface

The lookup table for the brake torque, $f_{brake}(P, N)$, is a function of applied brake pressure and wheel speed, where:

- T is brake torque, in N·m.
- P is applied brake pressure, in bar.
- N is wheel speed, in rpm.

Ports

Input

BrkPrs — Brake pressure
scalar | N-by-1 vector

Brake pressure, in Pa.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Dependencies

To enable this port, for the **Brake Type** parameter, specify one of these types:

- Disc
- Drum
- Mapped

AxlTrq — Axle torque
scalar | N-by-1 vector

Axle torque, T_a, about wheel spin axis, in N·m.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Vx — Longitudinal velocity
scalar | N-by-1 vector

Axle longitudinal velocity, V_x, along tire-fixed *x*-axis, in m/s.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Vy — Lateral velocity
scalar | N-by-1 vector

Axle lateral velocity, V_y, along tire-fixed *y*-axis, in m/s.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Camber — Camber angle
scalar | N-by-1 vector

Camber angle, ɣ, in rad.

Vector is the number of wheels, *N*, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

YawRate — Tire angular velocity
scalar | N-by-1 vector

Tire angular velocity, *r*, about the tire-fixed *z*-axis (yaw rate), in rad/s.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Prs — Tire inflation pressure
scalar | N-by-1 vector

Tire inflation pressure, p_i, in Pa.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Gnd — Ground displacement
scalar | N-by-1 vector

Ground displacement along tire-fixed z-axis, in m. Positive input produces wheel lift.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Fext — Axle force applied to tire
scalar | N-by-1 vector

Axle force applied to tire, F_{ext}, along vehicle-fixed z-axis (positive input compresses the tire), in N·m.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

ScaleFctrs — Scale factor
scalar | N-by-1 vector

Scale factor to account for variations in the coefficient of friction.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Output

Info — Block data
bus

Block data, returned as a bus signal containing these block values.

Signal	Description	Units
AxlTrq	Axle torque about wheel-fixed y-axis	N·m
Omega	Wheel angular velocity about wheel-fixed y-axis	rad/s
Fx	Longitudinal vehicle force along tire-fixed x-axis	N
Fy	Lateral vehicle force along tire-fixed y-axis	N
Fz	Vertical vehicle force along tire-fixed z-axis	N
Mx	Overturning moment about tire-fixed x-axis	N·m
My	Rolling resistance torque about tire-fixed y-axis	N·m
Mz	Aligning moment about tire-fixed z-axis	N·m

Signal	Description	Units
Vx	Vehicle longitudinal velocity along tire-fixed x-axis	m/s
Vy	Vehicle lateral velocity along tire-fixed y-axis	m/s
Re	Loaded effective radius	m
Kappa	Longitudinal slip ratio	NA
Alpha	Side slip angle	rad
a	Contact patch half length	m
b	Contact patch half width	m
Gamma	Camber angle	rad
psidot	Tire angular velocity about the tire-fixed z-axis (yaw rate)	rad/s
BrkTrq	Brake torque about the vehicle-fixed y-axis	N·m
BrkPrs	Brake pressure	Pa
z	Axle vertical displacement along tire-fixed z-axis	m
zdot	Axle vertical velocity along tire-fixed z-axis	m/s
Gnd	Ground displacement along tire-fixed z-axis (positive input produces wheel lift)	m
GndFz	Vertical sidewall force on ground along tire-fixed z-axis	N
Prs	Tire inflation pressure	Pa

Omega — Wheel angular velocity
scalar | N-by-1 vector

Wheel angular velocity, ω, about wheel-fixed y-axis, in rad/s.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Fx — Longitudinal axle force
scalar | N-by-1 vector

Longitudinal force acting on axle, F_x, along tire-fixed x-axis, in N. Positive force acts to move the vehicle forward.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Fy — Lateral axle force
scalar | N-by-1 vector

Lateral force acting on axle, F_y, along tire-fixed y-axis, in N.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Fz — Vertical axle force
scalar | N-by-1 vector

Vertical force acting on axle, F_z, along tire-fixed z-axis, in N.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Mx — Overturning moment
scalar | N-by-1 vector

Longitudinal moment acting on axle, M_x, about tire-fixed x-axis, in N·m.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

My — Rolling resistive moment
scalar | N-by-1 vector

Lateral moment acting on axle, M_y, about tire-fixed y-axis, in N·m.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Mz — Aligning moment
scalar | N-by-1 vector

Vertical moment acting on axle, M_z, about tire-fixed z-axis, in N·m.

Vector is the number of wheels, N, by 1. If you provide a scalar value, the block assumes that number of wheels is one.

Parameters

Block Options

Brake Type — Select type
None | Disc | Drum | Mapped

Use the **Brake Type** parameter to select the brake.

Brake Type Setting	Brake Implementation
None	None
Disc	Brake that converts the brake cylinder pressure into a braking force
Drum	Simplex drum brake that converts the applied force and brake geometry into a net braking torque
Mapped	Lookup table that is a function of the wheel speed and applied brake pressure

Rolling Resistance — Select type
None (default) | Pressure and velocity | ISO 28580 | Magic Formula | Mapped torque

To calculate the rolling resistance torque, specify one of these **Rolling Resistance** parameters.

Setting	Block Implementation
None	None
Pressure and velocity	Method in *Stepwise Coastdown Methodology for Measuring Tire Rolling Resistance*. The rolling resistance is a function of tire pressure, normal force, and velocity.
ISO 28580	Method specified in ISO 28580:2018, *Passenger car, truck and bus tyre rolling resistance measurement method — Single point test and correlation of measurement results*.
Magic Formula	Magic formula equations from 4.E70 in *Tire and Vehicle Dynamics*. The magic formula is an empirical equation based on fitting coefficients.
Mapped torque	Lookup table that is a function of the normal force and spin axis longitudinal velocity.

Dependencies

Selecting	Parameters
Pressure and velocity	**Velocity independent force coefficient, aMy**
	Linear velocity force component, bMy
	Quadratic velocity force component, cMy
	Tire pressure exponent, alphaMy
	Normal force exponent, betaMy
ISO 28580	**Parasitic losses force, Fpl**
	Rolling resistance constant, Cr
	Thermal correction factor, Kt
	Measured temperature, Tmeas
	Parasitic losses force, Fpl
	Ambient temperature, Tamb

Selecting	Parameters
Magic Formula	**Rolling resistance torque coefficient, QSY**
	Longitudinal force rolling resistance coefficient, QSY2
	Linear rotational speed rolling resistance coefficient, QSY3
	Quartic rotational speed rolling resistance coefficient, QSY4
	Camber squared rolling resistance torque, QSY5
	Load based camber squared rolling resistance torque, QSY6
	Normal load rolling resistance coefficient, QSY7
	Pressure load rolling resistance coefficient, QSY8
	Rolling resistance scaling factor, lam_My
Mapped torque	**Spin axis velocity breakpoints, VxMy**
	Normal force breakpoints, FzMy
	Rolling resistance torque map, MyMap

Vertical Motion — Select type
None (default) | Mapped stiffness and damping

To calculate vertical motion, specify one of these **Vertical Motion** parameters.

Setting	Block Implementation
None	Block passes the applied chassis forces directly through to the rolling resistance and longitudinal force calculations.
Mapped stiffness and damping	Vertical motion depends on wheel stiffness and damping. Stiffness is a function of tire sidewall displacement and pressure. Damping is a function of tire sidewall velocity and pressure.

Selecting	Enables These Parameters
`Mapped stiffness and damping`	**Wheel mass, MASS**
	Initial tire deflection, zo
	Initial velocity, zdoto
	Initial wheel vertical velocity (wheel fixed frame), zdoto
	Vertical deflection breakpoints, zFz
	Pressure breakpoints, pFz
	Force due to deflection, Fzz
	Vertical velocity breakpoints, zdotFz
	Force due to velocity, Fzzdot

Longitudinal and Lateral

Longitudinal stiffness, Ckappa — Longitudinal stiffness
1e7 (default) | scalar

Longitudinal stiffness, C_κ, in N.

Lateral stiffness per slip angle, Calpha — Lateral stiffness
4.5e4 (default) | scalar

Lateral stiffness per slip angle, C_α, in N/rad.

Camber stiffness, Cgamma — Camber stiffness
1e3 (default) | scalar

Camber stiffness, C_γ, in N/rad.

Kinematic friction, muMin — Friction
.8 (default) | scalar

Kinematic friction, μ_k, dimensionless.

Static friction, muMax — Friction
1 (default) | scalar

Static friction, μ_s, dimensionless.

Longitudinal relaxation length, Lrelx — Length
.05 (default) | scalar

Longitudinal relaxation length, L_{relx}, in m.

Lateral relaxation length, Lrely — Length
.15 (default) | scalar

Lateral relaxation length, L_{rely}, in m/rad.

Rolling

`Rotational damping, br` — Damping
`1e-3` (default) | scalar

Rotational damping, *br*, in N·m·s/rad.

`Rotational inertia (rolling axis), IYY` — Inertia
`0.74` (default) | scalar

Rotational inertia (rolling axis), *IYY*, in kg·m^2.

`Initial rotational velocity, omegao` — Velocity
`0` (default) | scalar

Initial rotational velocity, in rad/s.

`Unloaded radius, UNLOADED_RADIUS` — Radius
`0.309384029954441` (default) | scalar

Unloaded radius, in m.

Pressure and Velocity

`Velocity independent force coefficient, aMy` — Force coefficient
`8e-4` (default) | scalar

Velocity-independent force coefficient, *a*, in s/m.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Pressure and velocity`.

`Linear velocity force component, bMy` — Force component
`.001` (default) | scalar

Linear velocity force component, *b*, in s/m.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Pressure and velocity`.

`Quadratic velocity force component, cMy` — Force component
`1.6e-4` (default) | scalar

Quadratic velocity force component, *c*, in s^2/m^2.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Pressure and velocity`.

`Tire pressure exponent, alphaMy` — Pressure exponent
`-0.003` (default) | scalar

Tire pressure exponent, α, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Pressure and velocity`.

Normal force exponent, betaMy — Force exponent
0.97 (default) | scalar

Normal force exponent, β, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Pressure and velocity`.

ISO 28580

Parasitic losses force, Fpl — Force loss
10 (default) | scalar

Parasitic force loss, F_{pl}, in N.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `ISO 28580`.

Rolling resistance constant, Cr — Constant
1e-3 (default) | scalar

Rolling resistance constant, C_r, in N/kN. ISO 28580 specifies the rolling resistance unit as one newton of tractive resistance for every kilonewtons of normal load.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `ISO 28580`.

Thermal correction factor, Kt — Correction factor
.008 (default) | scalar

Thermal correction factor, K_t, in 1/degC.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `ISO 28580`.

Measured temperature, Tmeas — Temperature
298.15 (default) | scalar

Measured temperature, T_{meas}, in K.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `ISO 28580`.

Ambient temperature, Tamb — Temperature
298.15 (default) | scalar

Measured temperature, T_{amb}, in K.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `ISO 28580`.

Input ambient temperature — Selection
`off` (default) | `on`

Select to create input port `Tamb`.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `ISO 28580`.

Magic Formula

Rolling resistance torque coefficient, QSY1 — Torque coefficient
`0.007` (default) | `scalar`

Rolling resistance torque coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Longitudinal force rolling resistance coefficient, QSY2 — Force resistance coefficient
`0` (default) | `scalar`

Longitudinal force rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Linear rotational speed rolling resistance coefficient, QSY3 — Linear speed coefficient
`0.0015` (default) | `scalar`

Linear rotational speed rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Quartic rotational speed rolling resistance coefficient, QSY4 — Quartic speed coefficient
`8.5e-05` (default) | `scalar`

Quartic rotational speed rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Camber squared rolling resistance torque, QSY5 — Camber resistance torque
`0` (default) | `scalar`

Camber squared rolling resistance torque, in 1/rad^2.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Load based camber squared rolling resistance torque, QSY6 — Load resistance torque
`0` (default) | `scalar`

Load based camber squared rolling resistance torque, in 1/rad^2.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Normal load rolling resistance coefficient, QSY7 — Normal resistance coefficient
`0.9` (default) | `scalar`

Normal load rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Pressure load rolling resistance coefficient, QSY8 — Pressure resistance coefficient
`-0.4` (default) | `scalar`

Pressure load rolling resistance coefficient, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Rolling resistance scaling factor, lam_My — Scale
`1` (default) | `scalar`

Rolling resistance scaling factor, dimensionless.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Magic Formula`.

Mapped

Spin axis velocity breakpoints, VxMy — Breakpoints
`-20:1:20` (default) | `vector`

Spin axis velocity breakpoints, in m/s.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Mapped torque`.

Normal force breakpoints, FzMy — Breakpoints
`0:200:1e4` (default) | `vector`

Normal force breakpoints, in N.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Mapped torque`.

Rolling resistance torque map, MyMap — Lookup table
`array`

Rolling resistance torque versus axle speed and normal force, in N·m.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter `Mapped torque`.

Aligning

Wheel width, WIDTH — Width
`0.209` (default) | `scalar`

Wheel width, *WIDTH*, in m.

Linear yaw rate resistance, bMz — Resistance
`0` | `scalar`

Linear yaw rate resistance, b_{Mz}, in N·m·s/rad.

Brake

Static friction coefficient, mu_static — Static friction
`.3` (default) | `scalar`

Static friction coefficient, dimensionless.

Dependencies

To enable this parameter, for the **Brake Type** parameter, specify one of these types:

- `Disc`
- `Drum`
- `Mapped`

Kinetic friction coefficient, mu_kinetic — Kinetic friction
`.2` (default) | `scalar`

Kinematic friction coefficient, dimensionless.

Dependencies

To enable this parameter, for the **Brake Type** parameter, specify one of these types:

- `Disc`
- `Drum`
- `Mapped`

Disc

Disc brake actuator bore, disc_abore — Bore distance
`.05` (default) | `scalar`

Disc brake actuator bore, in m.

Dependencies

To enable the disc brake parameters, select Disc for the **Brake Type** parameter.

Brake pad mean radius, Rm — Radius
.177 (default) | scalar

Brake pad mean radius, in m.

Dependencies

To enable the disc brake parameters, select Disc for the **Brake Type** parameter.

Number of brake pads, num_pads — Count
2 (default) | scalar

Number of brake pads.

Dependencies

To enable the disc brake parameters, select Disc for the **Brake Type** parameter.

Drum

Drum brake actuator bore, disc_abore — Bore distance
0.0508 (default) | scalar

Drum brake actuator bore, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin to drum center distance, drum_a — Distance
0.123 (default) | scalar

Shoe pin to drum center distance, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin center to force application point distance, drum_c — Distance
0.212 (default) | scalar

Shoe pin center to force application point distance, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Drum internal radius, drum_r — Radius
0.15 (default) | scalar

Drum internal radius, in m.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin to pad start angle, drum_theta1 — Angle
0 (default) | scalar

Shoe pin to pad start angle, in deg.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Shoe pin to pad end angle, drum_theta2 — Angle
126 (default) | scalar

Shoe pin to pad end angle, in deg.

Dependencies

To enable the drum brake parameters, select Drum for the **Brake Type** parameter.

Mapped

Brake actuator pressure breakpoints, brake_p_bpt — Breakpoints
vector

Brake actuator pressure breakpoints, in bar.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Wheel speed breakpoints, brake_n_bpt — Breakpoints
vector

Wheel speed breakpoints, in rpm.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Brake torque map, f_brake_t — Lookup table
array

The lookup table for the brake torque, $f_{brake}(P, N)$, is a function of applied brake pressure and wheel speed, where:

- T is brake torque, in N·m.
- P is applied brake pressure, in bar.
- N is wheel speed, in rpm.

Dependencies

To enable the mapped brake parameters, select Mapped for the **Brake Type** parameter.

Vertical

Wheel mass, m — Mass
9.46491996974568 (default) | scalar

Wheel mass, in kg. Used in the vertical motion calculations.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Initial tire deflection, zo — Deflection
0 (default) | scalar

Initial axle displacement along wheel-fixed z-axis, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Initial wheel vertical velocity (wheel fixed frame), zdoto — Velocity
0 (default) | scalar

Initial axle velocity along wheel-fixed z-axis, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Gravitational acceleration, GRAVITY — Gravity
-9.81 (default) | scalar

Gravitational acceleration, in m/s^2.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Mapped Stiffness and Damping

Vertical deflection breakpoints, zFz — Breakpoints
[0 .01 .1] (default) | vector

Vector of sidewall deflection breakpoints corresponding to the force table, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Pressure breakpoints, pFz — Breakpoints
[10000 1000000] (default) | vector

Vector of pressure data points corresponding to the force table, in Pa.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Force due to deflection, Fzz — Force
[0 1e3 1e4; 0 1e4 1e5] (default) | vector

Force due to sidewall deflection and pressure along wheel-fixed z-axis, in N.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Vertical velocity breakpoints, zdotFz — Breakpoints
[-20 0 20] (default) | scalar

Vector of sidewall velocity breakpoints corresponding to the force due to velocity table, in m.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Force due to velocity, Fzzdot — Force
[500 0 -500;250 0 -250] (default) | array

Force due to sidewall velocity and pressure along wheel-fixed z-axis, in N.

Dependencies

To enable this parameter, set **Vertical Motion** to Mapped stiffness and damping.

Simulation

Maximum normal force, FZMAX — Force
10000 (default) | scalar

Maximum normal force, in N. Used with all vertical force calculations.

Minimum normal force, FZMIN — Force
0 (default) | scalar

Minimum normal force, in N. Used with all vertical force calculations.

Maximum pressure, PRESMAX — Pressure
1003118 (default) | scalar

Maximum pressure, *PRESMAX*, in Pa.

Minimum pressure, PRESMIN — Pressure
9982 (default) | scalar

Minimum pressure, *PRESMIN*, in Pa.

Max allowable slip ratio (absolute), KPUMAX — Ratio
1.5 (default) | scalar

Max allowable slip ratio (absolute), *KPUMAX*, dimensionless.

Minimum allowable slip ratio (absolute), KPUMIN — Ratio
-1.5 (default) | scalar

Minimum allowable slip ratio (absolute), *KPUMIN*, dimensionless.

Max allowable slip angle (absolute), ALPMAX — Angle
1.5708 (default) | scalar

Max allowable slip angle (absolute), *ALPMAX*, in rad.

Minimum allowable slip angle (absolute), ALPMIN — Angle
-1.5708 (default) | scalar

Minimum allowable slip angle (absolute), *ALPMIN*, in rad.

Maximum allowable camber angle, CAMMAX — Angle
0.173 (default) | scalar

Maximum allowable camber angle *CAMMAX*, in rad.

Minimum allowable camber angle, CAMMIN — Angle
-0.173 (default) | scalar

Minimum allowable camber angle, *CAMMIN*, in rad.

Minimum ambient temperature, TMIN — Tmin
0 (default) | scalar

Minimum ambient temperature, T_{MIN}, in K.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

Maximum ambient temperature, TMAX — Tmax
400 (default) | scalar

Maximum ambient temperature, T_{MAX}, in K.

Dependencies

To create this parameter, select the **Rolling Resistance** parameter ISO 28580.

References

[1] Fiala, E. "Seitenkrafte am Rollenden Luftreifen." *VDI Zeitschrift, V.D.I.*. Vol 96, 1954.

[2] Highway Tire Committee. *Stepwise Coastdown Methodology for Measuring Tire Rolling Resistance*. Standard J2452_199906. Warrendale, PA: SAE International, June 1999.

[3] ISO 28580:2018. *Passenger car, truck and bus tyre rolling resistance measurement method — Single point test and correlation of measurement results*. ISO (International Organization for Standardization), 2018.

[4] Pacejka, H. B. *Tire and Vehicle Dynamics*. 3rd ed. Oxford, UK: SAE and Butterworth-Heinemann, 2012.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Longitudinal Wheel

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2019a

Propulsion Blocks

Mapped Motor

Mapped motor and drive electronics operating in torque-control mode
Library: Powertrain Blockset / Propulsion / Electric Motors
              Vehicle Dynamics Blockset / Powertrain / Propulsion
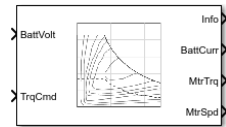
Description

The Mapped Motor block implements a mapped motor and drive electronics operating in torque-control mode. The output torque tracks the torque reference demand and includes a motor-response and drive-response time constant. Use the block for fast system-level simulations when you do not know detailed motor parameters, for example, for motor power and torque tradeoff studies. The block assumes that the speed fluctuations due to mechanical load do not affect the motor torque tracking.

You can specify:

- Port configuration — Input torque or speed.
- Electrical torque range — Torque speed envelope or maximum motor power and torque.
- Electrical loss — Single operating point, measured efficiency, or measured loss. If you have Model-Based Calibration Toolbox™, you can virtually calibrate the measured loss tables.

Electrical Torque

To specify the range of torque and speed that the block allows, on the **Electrical Torque** tab, for **Parametrized by**, select one of these options.

Setting	Block Implementation
Tabulated torque-speed envelope	Range specified as a set of speed data points and corresponding maximum torque values.
Maximum torque and power	Range specified with maximum torque and maximum power.

For either method, the block implements an envelope similar to this.
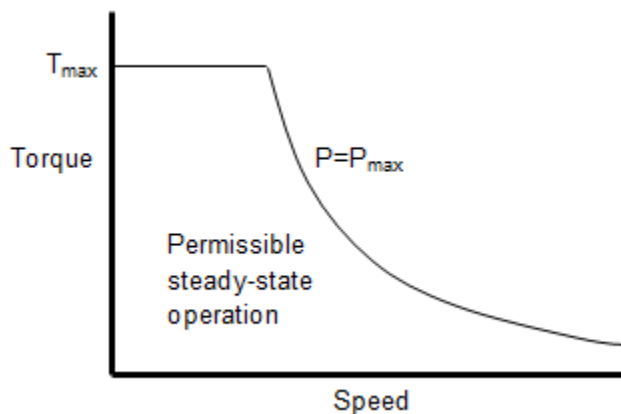
Electrical Losses

To specify the electrical losses, on the **Electrical Losses** tab, for **Parameterize losses by**, select one of these options.

Setting	Block Implementation
Single efficiency measurement	Sum of these terms, measured at a single measurement point: • Fixed losses independent of torque and speed, P_0. Use P_0 to account for fixed converter losses. • A torque-dependent electrical loss $k\tau^2$, where k is a constant and τ is the torque. Represents ohmic losses in the copper windings. • A speed-dependent electrical loss $k_w\omega^2$, where k_w is a constant and ω is the speed. Represents iron losses due to eddy currents.
Tabulated loss data	Loss lookup table that is a function of motor speeds and load torques. If you have Model-Based Calibration Toolbox, click **Calibrate Maps** to virtually calibrate the 2D lookup tables using measured data.
Tabulated loss data with temperature	Loss lookup table that is a function of motor speeds, load torques, and operating temperature. If you have Model-Based Calibration Toolbox, click **Calibrate Maps** to virtually calibrate the 3D lookup tables using measured data.
Tabulated efficiency data	2D efficiency lookup table that is a function of motor speeds and load torques: • Converts the efficiency values you provide into losses and uses the tabulated losses for simulation. • Ignores efficiency values you provide for zero speed or zero torque. Losses are assumed zero when either torque or speed is zero. • Uses linear interpolation to determine losses. Provide tabulated data for low speeds and low torques, as required, to get the desired level of accuracy for lower power conditions. • Does not extrapolate loss values for speed and torque magnitudes that exceed the range of the table.

Setting	Block Implementation
Tabulated efficiency data with temperature	3D efficiency lookup table that is a function of motor speeds, load torques, and operating temperature: • Converts the efficiency values you provide into losses and uses the tabulated losses for simulation. • Ignores efficiency values you provide for zero speed or zero torque. Losses are assumed zero when either torque or speed is zero. • Uses linear interpolation to determine losses. Provide tabulated data for low speeds and low torques, as required, to get the desired level of accuracy for lower power conditions. • Does not extrapolate loss values for speed, torque, or temperature magnitudes that exceed the range of the table.

For best practice, use `Tabulated loss data` instead of `Tabulated efficiency data`:

• Efficiency becomes ill defined for zero speed or zero torque.

• You can account for fixed losses that are still present for zero speed or torque.

Note Due to system losses, the motor can draw a current when the motor torque is zero.

Virtual Calibration

If you have Model-Based Calibration Toolbox, you can virtually calibrate the measured loss lookup tables.

1 On the **Electrical Losses** tab, set **Parameterize losses by** to either:

   • `Tabulated loss data`
   • `Tabulated loss data with temperature`

2 Click **Calibrate Maps**.

The dialog box steps through these tasks.

Task	Description
Import Loss Data	Import this loss data from a file. For example, open *<matlabroot>*/toolbox/ autoblks/autoblksshared/mbctemplates/MappedMotorDataset.xlsx. For more information, see "Using Data" (Model-Based Calibration Toolbox). <table><tr><th>Parameterize losses by</th><th>Required Data</th></tr><tr><td>Tabulated loss data</td><td>• Motor speed, rad/s • Motor torque, N·m • Power loss, W</td></tr><tr><td>Tabulated loss data with temperature</td><td>• Motor speed, rad/s • Motor torque, N·m • Motor temperature, K • Power loss, W</td></tr></table> Collect motor data at steady-state operating conditions. Data should cover the motor speed, torque, and temperature operating range. To filter or edit the data, select **Edit in Application**. The Model-Based Calibration Toolbox Data Editor opens.
Generate Response Models	Model-Based Calibration Toolbox uses test plans to fit data to Gaussian process models (GPMs). To assess or adjust the response model fit, select **Edit in Application**. The Model-Based Calibration Toolbox Model Browser opens. For more information, see "Model Assessment" (Model-Based Calibration Toolbox).
Generate Calibration	Model-Based Calibration Toolbox calibrates the response models and generates calibrated tables. To assess or adjust the calibration, select **Edit in Application**. The Model-Based Calibration Toolbox CAGE Browser opens. For more information, see "Calibration Lookup Tables" (Model-Based Calibration Toolbox).

Task	Description
Update block parameters	Update these parameters with the calibration.

Parameterize losses by	Parameters
Tabulated loss data	• **Vector of speeds(w) for tabulated losses, w_eff_bp** • **Vector of torques (T) for tabulated losses, T_eff_bp** • **Corresponding losses, losses_table**
Tabulated loss data with temperature	• **Vector of speeds(w) for tabulated losses, w_eff_bp** • **Vector of torques (T) for tabulated losses, T_eff_bp** • **Vector of temperatures for tabulated losses, Temp_eff_bp** • **Corresponding losses, losses_table_3d**

Battery Current

The block calculates the battery current using the mechanical power, power loss, and battery voltage. Positive current indicates battery discharge. Negative current indicates battery charge.

$$BattAmp = \frac{MechPwr + PwrLoss}{BattVolt}$$

The equation uses these variables.

BattVolt   Battery voltage
MechPwr   Mechanical power
PwrLoss   Power loss
BattCurr   Battery current

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Variable	Equations
PwrInfo	PwrTrnsfrd • Positive signals indicate power flow into the block. • Negative signals indicate power flow out of the block.	PwrMtr	Mechanical power	P_{mot}	$P_{mot} = \omega_m T_e$
		PwrBus	Electrical power	P_{bus}	$P_{bus} = P_{mot} + P_{loss}$
	PwrNotTrnsfrd • Negative signals indicate power loss.	PwrLoss	Motor power loss	P_{loss}	$P_{stored} = \omega_m \dot{\omega}_m J$

Bus Signal		Description	Variable	Equations
PwrStored • Positive signals indicate power gain.	PwrStor edShft	Motor power stored	P_{str}	$P_{loss} = -(P_{mot} + P_{loss} - P_{stored})$

The equations use these variables.

T_e	Motor output shaft torque
ω	Motor shaft speed
J	Motor inertia

Ports

Input

BattVolt — Battery voltage
scalar

Battery voltage, *BattVolt*, in V.

TrqCmd — Commanded motor torque
scalar

Commanded motor torque, Trq_{cmd}, in N·m.

Dependencies

To create this input port, for the **Port configuration**, select Torque.

MtrSpd — Motor output shaft speed
scalar

Motor shaft speed, Mtr_{spd}, in rad/s.

Dependencies

To create this input port, for the **Port configuration**, select Speed.

Output

Info — Bus signal
bus

The bus signal contains these block calculations.

Signal			Description	Units
MechPwr			Mechanical power	rad
PwrLoss			Internal inverter and motor power loss	N·m
PwrInfo	PwrTrnsfrd	PwrMtr	Mechanical power	W
		PwrBus	Electrical power	W

Signal		Description	Units
PwrNotTrnsfrd	PwrLoss	Motor power loss	W
PwrStored	PwrStored Shft	Motor power stored	W

BattCurr — Battery current
scalar

Battery current draw or demand, I_{batt}, in A.

MtrTrq — Motor torque
scalar

Motor output shaft torque, Mtr_{trq}, in N·m.

MtrSpd — Motor shaft speed
scalar

Motor shaft speed, Mtr_{spd}, in rad/s.

Dependencies

To create this output port, for the **Port configuration**, select Torque.

Parameters

Block Options

Port configuration — Select port configuration
Torque (default) | Speed

This table summarizes the port configurations.

Port Configuration	Creates Ports
Torque	Outpost MtrSpd
Speed	Input MtrSpd

Calibrate Maps — Calibrate tables with measured data
selection

If you have Model-Based Calibration Toolbox, you can virtually calibrate the measured loss lookup tables.

1   On the **Electrical Losses** tab, set **Parameterize losses by** to either:

   - Tabulated loss data
   - Tabulated loss data with temperature

2   Click **Calibrate Maps**.

The dialog box steps through these tasks.

Task	Description
Import Loss Data	Import this loss data from a file. For example, open *<matlabroot>*/toolbox/ autoblks/autoblksshared/mbctemplates/MappedMotorDataset.xlsx. For more information, see "Using Data" (Model-Based Calibration Toolbox).

Parameterize losses by	Required Data
Tabulated loss data	• Motor speed, rad/s • Motor torque, N·m • Power loss, W
Tabulated loss data with temperature	• Motor speed, rad/s • Motor torque, N·m • Motor temperature, K • Power loss, W

	Collect motor data at steady-state operating conditions. Data should cover the motor speed, torque, and temperature operating range. To filter or edit the data, select **Edit in Application**. The Model-Based Calibration Toolbox Data Editor opens.
Generate Response Models	Model-Based Calibration Toolbox uses test plans to fit data to Gaussian process models (GPMs). To assess or adjust the response model fit, select **Edit in Application**. The Model-Based Calibration Toolbox Model Browser opens. For more information, see "Model Assessment" (Model-Based Calibration Toolbox).
Generate Calibration	Model-Based Calibration Toolbox calibrates the response models and generates calibrated tables. To assess or adjust the calibration, select **Edit in Application**. The Model-Based Calibration Toolbox CAGE Browser opens. For more information, see "Calibration Lookup Tables" (Model-Based Calibration Toolbox).

Task	Description		
Update block parameters	Update these parameters with the calibration.		
	Parameterize losses by	**Parameters**	
	Tabulated loss data	• **Vector of speeds(w) for tabulated losses, w_eff_bp** • **Vector of torques (T) for tabulated losses, T_eff_bp** • **Corresponding losses, losses_table**	
	Tabulated loss data with temperature	• **Vector of speeds(w) for tabulated losses, w_eff_bp** • **Vector of torques (T) for tabulated losses, T_eff_bp** • **Vector of temperatures for tabulated losses, Temp_eff_bp** • **Corresponding losses, losses_table_3d**	

Electrical Torque

Parameterized by — Select type
Tabulated torque-speed envelope (default) | Maximum torque and power

Setting	Block Implementation
Tabulated torque-speed envelope	Range specified as a set of speed data points and corresponding maximum torque values.
Maximum torque and power	Range specified with maximum torque and maximum power.

For either method, the block implements an envelope similar to this.
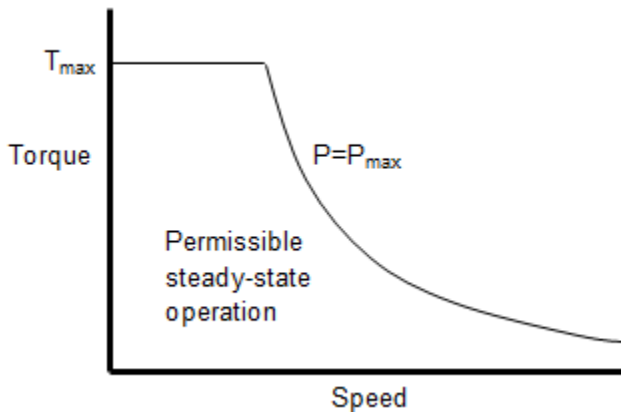
Vector of rotational speeds, w_t — Rotational speeds
[0 375 750 800] (default) | vector

Rotational speeds for permissible steady-state operation, in rad/s. To avoid poor performance due to an infinite slope in the torque-speed curve, specify a vector of rotational speeds that does not contain duplicate consecutive values.

Dependencies

To create this parameter, for the **Parameterized by** parameter, select `Tabulated torque-speed envelope`.

Vector of maximum torque values, T_t — Torque
[0.09 0.08 0.07 0] (default) | `vector`

Maximum torque values for permissible steady state, in N·m.

Dependencies

To create this parameter, for the **Parameterized by** parameter, select `Tabulated torque-speed envelope`.

Maximum torque, torque_max — Torque
.1 (default) | `scalar`

The maximum permissible motor torque, in N·m.

Dependencies

To create this parameter, for the **Parameterized by** parameter, select `Maximum torque and power`.

Maximum power, power_max — Power
30 (default) | `scalar`

The maximum permissible motor power, in W.

Dependencies

To create this parameter, for the **Parameterized by** parameter, select `Maximum torque and power`.

Torque control time constant, Tc — Time constant
0.02 (default) | `scalar`

Time constant with which the motor driver tracks a torque demand, in s.

Electrical Losses

Parameterize losses by — Select type
`Single efficiency measurement` (default) | `Tabulated loss data` | `Tabulated efficiency data`

Setting	Block Implementation
`Single efficiency measurement`	Sum of these terms, measured at a single measurement point: • Fixed losses independent of torque and speed, P_0. Use P_0 to account for fixed converter losses. • A torque-dependent electrical loss $k\tau^2$, where k is a constant and τ is the torque. Represents ohmic losses in the copper windings. • A speed-dependent electrical loss $k_w\omega^2$, where k_w is a constant and ω is the speed. Represents iron losses due to eddy currents.
`Tabulated loss data`	Loss lookup table that is a function of motor speeds and load torques. If you have Model-Based Calibration Toolbox, click **Calibrate Maps** to virtually calibrate the 2D lookup tables using measured data.
`Tabulated loss data with temperature`	Loss lookup table that is a function of motor speeds, load torques, and operating temperature. If you have Model-Based Calibration Toolbox, click **Calibrate Maps** to virtually calibrate the 3D lookup tables using measured data.
`Tabulated efficiency data`	2D efficiency lookup table that is a function of motor speeds and load torques: • Converts the efficiency values you provide into losses and uses the tabulated losses for simulation. • Ignores efficiency values you provide for zero speed or zero torque. Losses are assumed zero when either torque or speed is zero. • Uses linear interpolation to determine losses. Provide tabulated data for low speeds and low torques, as required, to get the desired level of accuracy for lower power conditions. • Does not extrapolate loss values for speed and torque magnitudes that exceed the range of the table.

Setting	Block Implementation
Tabulated efficiency data with temperature	3D efficiency lookup table that is a function of motor speeds, load torques, and operating temperature: • Converts the efficiency values you provide into losses and uses the tabulated losses for simulation. • Ignores efficiency values you provide for zero speed or zero torque. Losses are assumed zero when either torque or speed is zero. • Uses linear interpolation to determine losses. Provide tabulated data for low speeds and low torques, as required, to get the desired level of accuracy for lower power conditions. • Does not extrapolate loss values for speed, torque, or temperature magnitudes that exceed the range of the table.

For best practice, use `Tabulated loss data` instead of `Tabulated efficiency data`:

• Efficiency becomes ill defined for zero speed or zero torque.
• You can account for fixed losses that are still present for zero speed or torque.

Note Due to system losses, the motor can draw a current when the motor torque is zero.

Motor and drive overall efficiency, eff — Efficiency
100 (default) | scalar

The block defines overall efficiency as:

$$\eta = 100 \frac{\tau_0 \omega_0}{\tau_0 \omega_0 + P_0 + k\tau_0^2 + k_w\omega_0^2}$$

The equation uses these variables.

τ_0	Torque at which efficiency is measured
ω_0	Speed at which efficiency is measured
P_0	Fixed losses independent of torque or speed
$k\tau_0^2$	Torque-dependent electrical losses
$k_w\omega^2$	Speed-dependent iron losses

At initialization, the block solves the efficiency equation for *k*. The block neglects losses associated with the rotor damping.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Single efficiency measurement`.

Speed at which efficiency is measured, w_eff — Speed
375 (default) | scalar

Speed at which efficiency is measured, in rad/s.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Single efficiency measurement`.

Torque at which efficiency is measured, T_eff — Torque
`0.08` (default) | `scalar`

Torque at which efficiency is measured, in N·m.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Single efficiency measurement`.

Iron losses, Piron — Power
`0` (default) | `scalar`

Iron losses at the speed and torque at which efficiency is defined, in W.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Single efficiency measurement`.

Fixed losses independent of torque and speed, Pbase — Power
`0` (default) | `scalar`

Fixed electrical loss associated with the driver when the motor current and torque are zero, in W.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Single efficiency measurement`.

Vector of speeds (w) for tabulated losses, w_eff_bp — Breakpoints
`[-8000 -4000 0 4000 8000]` (default) | 1-by-M vector

Speed breakpoints for lookup table when calculating losses, in rad/s. Array dimensions are 1 by the number of speed breakpoints, M.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select one of these:

- `Tabulated loss data`
- `Tabulated loss data with temperature`
- `Tabulated efficiency data`
- `Tabulated efficiency data with temperature`

Vector of torques (T) for tabulated losses, T_eff_bp — Breakpoints
`[0 0.03 0.06 0.09]` (default) | 1-by-N vector

Torque breakpoints for lookup table when calculating losses, in N·m. Array dimensions are 1 by the number of torque breakpoints, N.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select one of these:

- `Tabulated loss data`
- `Tabulated loss data with temperature`
- `Tabulated efficiency data`
- `Tabulated efficiency data with temperature`

Vector of temperatures for tabulated losses, Temp_eff_bp — Breakpoints
[233.15 293.15 373.15] (default) | 1-by-L vector

Temperature breakpoints for lookup table when calculating losses, in K. Array dimensions are 1 by the number of temperature breakpoints, L.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select one of these:

- `Tabulated loss data with temperature`
- `Tabulated efficiency data with temperature`

Corresponding losses, losses_table — 2D lookup table
M-by-N matrix

Array of values for electrical losses as a function of speed and torque, in W. Each value specifies the losses for a specific combination of speed and torque. The array dimensions must match the speed, M, and torque, N, breakpoint vector dimensions.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Tabulated loss data`.

Corresponding losses, losses_table_3d — 3D lookup table
M-by-N-by-L array

Array of values for electrical losses as a function of speed, torque, and temperature, in W. Each value specifies the losses for a specific combination of speed, torque, and temperature. The array dimensions must match the speed, M, torque, N, and temperature, L, breakpoint vector dimensions.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Tabulated loss data with temperature`.

Corresponding efficiency, efficiency_table — 2D lookup table
M-by-N matrix

Array of efficiency as a function of speed and torque, in %. Each value specifies the losses for a specific combination of speed and torque. The array dimensions must match the speed, M, and torque, N, breakpoint vector dimensions.

The block ignores efficiency values for zero speed or zero torque. Losses are zero when either torque or speed is zero. The block uses linear interpolation.

To get the desired level of accuracy for lower power conditions, you can provide tabulated data for low speeds and low torques.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Tabulated efficiency data`.

Corresponding efficiency, efficiency_table_3d — 3D lookup table
M-by-N-by-L array

Array of efficiency as a function of speed and torque, in %. Each value specifies the losses for a specific combination of speed and torque. The array dimensions must match the speed, M, torque, N, and temperature, L, breakpoint vector dimensions.

The block ignores efficiency values for zero speed or zero torque. Losses are zero when either torque or speed is zero. The block uses linear interpolation.

To get the desired level of accuracy for lower power conditions, you can provide tabulated data for low speeds and low torques.

Dependencies

To create this parameter, for the **Parameterize losses by** parameter, select `Tabulated efficiency data`.

Mechanical

Rotor inertia, J — Inertia
5e-6 (default) | scalar

Rotor resistance to change in motor motion, in kg*m^2. The value can be zero.

Dependencies

To create this parameter, for the **Port configuration** parameter, select `Torque`.

Rotor damping, b — Damping
1e-5 (default) | scalar

Rotor damping, in N·m/(rad/s). The value can be zero.

Dependencies

To create this parameter, for the **Port configuration** parameter, select `Torque`.

Initial rotor speed, omega_o — Speed
0 (default) | scalar

Rotor speed at the start of the simulation, in rad/s.

Dependencies

To create this parameter, for the **Port configuration** parameter, select `Torque`.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Open Differential

Introduced in R2017a

Mapped CI Engine

Compression-ignition engine model using lookup tables

Library: Powertrain Blockset / Propulsion / Combustion Engines
Vehicle Dynamics Blockset / Powertrain / Propulsion
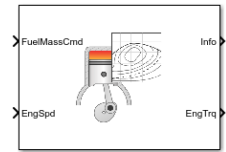
Description

The Mapped CI Engine block implements a mapped compression-ignition (CI) engine model using power, air mass flow, fuel flow, exhaust temperature, efficiency, and emission performance lookup tables. You can use the block for:

- Hardware-in-the-loop (HIL) engine control design
- Vehicle-level fuel economy and performance simulations

The lookup tables, developed with the Model-Based Calibration Toolbox, are functions of injected fuel mass, F, engine torque, T, engine speed, N, and engine temperature, $Temp_{Eng}$.

Input Command Setting	Input Engine Temperature Parameter Setting	Lookup Tables
Fuel mass	off	$f(F,N)$
	on	$f(F,N,Temp_{Eng})$
Torque	off	$f(T,N)$
	on	$f(T,N,Temp_{Eng})$

The block enables you to specify lookup tables for these engine characteristics:

- Power
- Air
- Fuel
- Temperature
- Efficiency
- Hydrocarbon (HC) emissions
- Carbon monoxide (CO) emissions
- Nitric oxide and nitrogen dioxide (NOx) emissions
- Carbon dioxide (CO_2) emissions
- Particulate matter (PM) emissions

To bound the Mapped CI Engine block output, the block does not extrapolate the lookup table data.

Virtual Calibration

If you have Model-Based Calibration Toolbox, click **Calibrate Maps** to virtually calibrate the 2D lookup tables using measured data. The dialog box steps through these tasks.

Task	Description
Import firing data	Import this loss data from a file. For example, open *<matlabroot>*/toolbox/autoblks/autodemos/projectsrc/CIDynamometer/CalMappedEng/CiEngineData.xlsx. For more information, see "Using Data" (Model-Based Calibration Toolbox). {{NESTED_TABLE}} Collect firing data at steady-state operating conditions when injectors deliver the fuel. Data should cover the engine speed and torque operating range. Model-Based Calibration Toolbox uses the firing data boundary as the maximum torque. To filter or edit the data, select **Edit in Application**. The Model-Based Calibration Toolbox Data Editor opens.
Import non-firing data	Import this non-firing data from a file. For example, open *<matlabroot>*/toolbox/autoblks/autodemos/projectsrc/CIDynamometer/CalMappedEng/CiEngineData.xlsx. • Engine speed, rpm • Engine torque, N·m Collect non-firing (motoring) data at steady-state operating conditions when fuel is cut off. All non-firing torque points must be less than zero. Non-firing data is a function of engine speed only.
Generate response models	For both firing and non-firing data, the Model-Based Calibration Toolbox uses test plans to fit data to Gaussian process models (GPMs). To assess or adjust the response model fit, select **Edit in Application**. The Model-Based Calibration Toolbox Model Browser opens. For more information, see "Model Assessment" (Model-Based Calibration Toolbox).

Nested table within "Import firing data":

Input command	Required Data	Optional Data
Fuel mass	• Engine speed, rpm • Commanded fuel mass per injection, mg • Engine torque, N·m	• Air mass flow rate, kg/s • Brake specific fuel consumption, g/(kW·h) • CO_2 mass flow rate, kg/s
Torque	• Engine speed, rpm • Engine torque, N·m	• CO mass flow rate, kg/s • Exhaust temperature, K • Fuel mass flow rate, kg/s • HC mass flow rate, kg/s • NOx mass flow rate, kg/s • Particulate matter mass flow rate, kg/s

Task	Description
Generate calibration	Model-Based Calibration Toolbox calibrates the firing and non-firing response models and generates calibrated tables.
	To assess or adjust the calibration, select **Edit in Application**. The Model-Based Calibration Toolbox CAGE Browser opens. For more information, see "Calibration Lookup Tables" (Model-Based Calibration Toolbox).
Update block parameters	Update the block lookup table and breakpoint parameters with the calibration.

Cylinder Air Mass

The block calculates the normalized cylinder air mass using these equations.

$$M_{Nom} = \frac{P_{std}V_d}{N_{cyl}R_{air}T_{std}}$$

$$L = \frac{\left(\frac{60s}{min}\right)Cps \cdot \dot{m}_{air}}{\left(\frac{1000g}{Kg}\right)N_{cyl} \cdot N \cdot M_{Nom}}$$

The equations use these variables.

L	Normalized cylinder air mass
M_{Nom}	Nominal engine cylinder air mass at standard temperature and pressure, piston at bottom dead center (BDC) maximum volume, in kg
Cps	Crankshaft revolutions per power stroke, rev/stroke
P_{std}	Standard pressure
T_{std}	Standard temperature
R_{air}	Ideal gas constant for air and burned gas mixture
V_d	Displaced volume
N_{cyl}	Number of engine cylinders
N	Engine speed
$\dot{m}_{intk}$	Engine air mass flow, in g/s

Turbocharger Lag

To model turbocharger lag, select **Include turbocharger lag effect**. Turbocharger lag limits the maximum fuel mass per injection. To model the maximum fuel mass per injection, the block uses a first-order system with a time constant. At low torque, the engine does not require boost to provide sufficient air flow. When the requested fuel mass requires boost, the block uses a time constant to determine the maximum fuel mass per injection. The block uses these equations for the specified **Input command** setting.

Calculation	Input command Parameter Setting	
	Fuel mass	**Torque**
Dynamic torque	$\dfrac{dF_{max}}{dt} = \dfrac{1}{\tau_{eng}}(F_{cmd} - F_{max})$	$\dfrac{dT_{max}}{dt} = \dfrac{1}{\tau_{eng}}(T_{cmd} - T_{max})$
Fuel mass per injection or torque - with turbocharger lag	$F =$ $\begin{cases} F_{cmd} & \text{when } F_{cmd} < F_{\max} \\ F_{max} & \text{when } F_{cmd} \geq F_{\max} \end{cases}$	$T_{target} =$ $\begin{cases} T_{cmd} & \text{when } T_{cmd} < T_{\max} \\ T_{max} & \text{when } T_{cmd} \geq T_{\max} \end{cases}$
Fuel mass per injection or torque- without turbocharger lag	$F = F_{cmd} = F_{max}$	$T_{target} = T_{cmd} = T_{max}$
Boost time constant	$\tau_{bst} =$ $\begin{cases} \tau_{bst,\,rising} & \text{when } F_{cmd} > F_{\max} \\ \tau_{bst,\,falling} & \text{when } F_{cmd} \leq F_{\max} \end{cases}$	$\tau_{bst} =$ $\begin{cases} \tau_{bst,\,rising} & \text{when } T_{cmd} > T_{\max} \\ \tau_{bst,\,falling} & \text{when } T_{cmd} \leq T_{\max} \end{cases}$
Final time constant	$\tau_{eng} = \begin{cases} \tau_{nat} & \text{when } T_{brake} < f_{bst}(N) \\ \tau_{bst} & \text{when } T_{brake} \geq f_{bst}(N) \end{cases}$	

The equations use these variables.

T_{brake}	Brake torque
F	Fuel mass per injection
F_{cmd}, F_{max}	Commanded and maximum fuel mass per injection, respectively
T_{target}, T_{cmd}, T_{max}	Target, commanded, and maximum torque, respectively
τ_{bst}	Boost time constant
$\tau_{bst,rising}$, $\tau_{bst,falling}$	Boost rising and falling time constant, respectively
τ_{eng}	Final time constant
τ_{nat}	Time constant below the boost torque speed line
$f_{bst}(N)$	Boost torque/speed line
N	Engine speed

Fuel Flow

To calculate the fuel economy for high-fidelity models, the block uses the volumetric fuel flow.

$$Q_{fuel} = \frac{\dot{m}_{fuel}}{\left(\frac{1000kg}{m^3}\right)Sg_{fuel}}$$

The equation uses these variables.

$\dot{m}_{fuel}$	Fuel mass flow
Sg_{fuel}	Specific gravity of fuel

Q_{fuel} Volumetric fuel flow

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrCrkshft	Crankshaft power	$-\tau_{eng}\omega$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrFuel	Fuel input power	$\dot{m}_{fuel}LHV$
		PwrLoss	Power loss	$\tau_{eng}\omega$ $-\dot{m}_{fuel}LHV$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease		*Not used*	

The equations use these variables.

LHV	Fuel lower heating value
ω	Engine speed, rad/s
$\dot{m}_{fuel}$	Fuel mass flow
τ_{eng}	Fuel mass per injection time constant

Ports

Input

FuelMassCmd — Injected fuel mass command
scalar

Injected fuel mass command, *F*, in mg/inj.

Dependencies

To enable this port, for **Input command**, select Fuel mass.

TrqCmd — Torque command
scalar

Torque command, *T*, in N·m.

Dependencies

To enable this port, for **Input command**, select Torque.

EngSpd — Engine speed
scalar

Engine speed, *N*, in rpm.

EngTemp — Engine temperature
scalar

Engine temperature, $Temp_{Eng}$, in K.

Dependencies

To enable this port, select **Input engine temperature**.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal	Description	Units
IntkGasMassFlw	Engine air mass flow output	kg/s
NrmlzdAirChrg	Normalized engine cylinder air mass	N/A
Afr	Air-fuel ratio (AFR)	N/A
FuelMassFlw	Engine fuel flow output	kg/s
FuelVolFlw	Volumetric fuel flow	m³/s
ExhManGasTemp	Engine exhaust gas temperature	K
EngTrq	Engine torque output	N·m
EngSpd	Engine speed	rpm
CrkAng	Engine crankshaft absolute angle $$\int_0^{(360)Cps} EngSpd \frac{180}{30} d\theta$$ where *Cps* is crankshaft revolutions per power stroke.	degrees crank angle
Bsfc	Engine brake-specific fuel consumption (BSFC)	g/kWh
EoHC	Engine out hydrocarbon emission mass flow	kg/s
EoCO	Engine out carbon monoxide emission mass flow rate	kg/s
EoNOx	Engine out nitric oxide and nitrogen dioxide emissions mass flow	kg/s
EoCO2	Engine out carbon dioxide emission mass flow	kg/s

Signal			Description	Units
EoPM			Engine out particulate matter emission mass flow	kg/s
PwrInfo	PwrTrnsfrd	PwrCrkshft	Crankshaft power	W
	PwrNotTrnsfrd	PwrFuel	Fuel input power	W
		PwrLoss	Power loss	W
	PwrStored		*Not used*	

EngTrq — Power
scalar

Engine power, T_{brake}, in N·m.

Parameters

Block Options

Input command — Table functions
Fuel mass (default) | Torque

The lookup tables, developed with the Model-Based Calibration Toolbox, are functions of injected fuel mass, *F*, engine torque, *T,* engine speed, *N*, and engine temperature, $Temp_{Eng}$.

Input Command Setting	Input Engine Temperature Parameter Setting	Lookup Tables
Fuel mass	off	$f(F,N)$
	on	$f(F,N,Temp_{Eng})$
Torque	off	$f(T,N)$
	on	$f(T,N,Temp_{Eng})$

Dependencies

- Selecting Fuel mass enables **Breakpoints for commanded fuel mass input, f_tbrake_f_bpt**.

- Selecting Torque enables **Breakpoints for commanded torque input, f_tbrake_t_bpt**.

- Selecting **Input engine temperature** enables **Breakpoints for temperature input, f_tbrake_engtmp_bpt**.

Include turbocharger lag effect — Increase time constant
off (default)

To model turbocharger lag, select **Include turbocharger lag effect**. Turbocharger lag limits the maximum fuel mass per injection. To model the maximum fuel mass per injection, the block uses a first-order system with a time constant. At low torque, the engine does not require boost to provide sufficient air flow. When the requested fuel mass requires boost, the block uses a time constant to determine the maximum fuel mass per injection. The block uses these equations for the specified **Input command** setting.

Calculation	Input command Parameter Setting	
	Fuel mass	**Torque**
Dynamic torque	$\dfrac{dF_{max}}{dt} = \dfrac{1}{\tau_{eng}}(F_{cmd} - F_{max})$	$\dfrac{dT_{max}}{dt} = \dfrac{1}{\tau_{eng}}(T_{cmd} - T_{max})$
Fuel mass per injection or torque - with turbocharger lag	$F =$ $\begin{cases} F_{cmd} & \text{when } F_{cmd} < F_{max} \\ F_{max} & \text{when } F_{cmd} \geq F_{max} \end{cases}$	$T_{target} =$ $\begin{cases} T_{cmd} & \text{when } T_{cmd} < T_{max} \\ T_{max} & \text{when } T_{cmd} \geq T_{max} \end{cases}$
Fuel mass per injection or torque- without turbocharger lag	$F = F_{cmd} = F_{max}$	$T_{target} = T_{cmd} = T_{max}$
Boost time constant	$\tau_{bst} =$ $\begin{cases} \tau_{bst,rising} & \text{when } F_{cmd} > F_{max} \\ \tau_{bst,falling} & \text{when } F_{cmd} \leq F_{max} \end{cases}$	$\tau_{bst} =$ $\begin{cases} \tau_{bst,rising} & \text{when } T_{cmd} > T_{max} \\ \tau_{bst,falling} & \text{when } T_{cmd} \leq T_{max} \end{cases}$
Final time constant	$\tau_{eng} = \begin{cases} \tau_{nat} & \text{when } T_{brake} < f_{bst}(N) \\ \tau_{bst} & \text{when } T_{brake} \geq f_{bst}(N) \end{cases}$	

The equations use these variables.

T_{brake}	Brake torque
F	Fuel mass per injection
F_{cmd}, F_{max}	Commanded and maximum fuel mass per injection, respectively
T_{target}, T_{cmd}, T_{max}	Target, commanded, and maximum torque, respectively
τ_{bst}	Boost time constant
$\tau_{bst,rising}$, $\tau_{bst,falling}$	Boost rising and falling time constant, respectively
τ_{eng}	Final time constant
τ_{nat}	Time constant below the boost torque speed line
$f_{bst}(N)$	Boost torque/speed line
N	Engine speed

Dependencies

Selecting **Include turbocharger lag effect** enables these parameters:

- **Boost torque line, f_tbrake_bst**
- **Time constant below boost line, tau_nat**
- **Rising maximum fuel mass boost time constant, tau_bst_rising**
- **Falling maximum fuel mass boost time constant, tau_bst_falling**

Input engine temperature — Create input port
off (default) | on

Select this to create the EngTemp input port.

The lookup tables, developed with the Model-Based Calibration Toolbox, are functions of injected fuel mass, *F*, engine torque, *T*, engine speed, *N*, and engine temperature, $Temp_{Eng}$.

Input Command Setting	Input Engine Temperature Parameter Setting	Lookup Tables
`Fuel mass`	off	$f(F,N)$
	on	$f(F,N,Temp_{Eng})$
`Torque`	off	$f(T,N)$
	on	$f(T,N,Temp_{Eng})$

Configuration

`Calibrate Maps` — Calibrate tables with measured data
selection

If you have Model-Based Calibration Toolbox, click **Calibrate Maps** to virtually calibrate the 2D lookup tables using measured data. The dialog box steps through these tasks.

Task	Description
Import firing data	Import this loss data from a file. For example, open *\<matlabroot\>*`/toolbox/autoblks/autodemos/projectsrc/CIDynamometer/CalMappedEng/CiEngineData.xlsx`.

For more information, see "Using Data" (Model-Based Calibration Toolbox).

Input command	Required Data	Optional Data
`Fuel mass`	• Engine speed, rpm • Commanded fuel mass per injection, mg • Engine torque, N·m	• Air mass flow rate, kg/s • Brake specific fuel consumption, g/(kW·h) • CO_2 mass flow rate, kg/s
`Torque`	• Engine speed, rpm • Engine torque, N·m	• CO mass flow rate, kg/s • Exhaust temperature, K • Fuel mass flow rate, kg/s • HC mass flow rate, kg/s • NOx mass flow rate, kg/s • Particulate matter mass flow rate, kg/s

Collect firing data at steady-state operating conditions when injectors deliver the fuel. Data should cover the engine speed and torque operating range. Model-Based Calibration Toolbox uses the firing data boundary as the maximum torque.

To filter or edit the data, select **Edit in Application**. The Model-Based Calibration Toolbox Data Editor opens.

Task	Description
Import non-firing data	Import this non-firing data from a file. For example, open *<matlabroot>/* `toolbox/autoblks/autodemos/projectsrc/CIDynamometer/` `CalMappedEng/CiEngineData.xlsx`. • Engine speed, rpm • Engine torque, N·m Collect non-firing (motoring) data at steady-state operating conditions when fuel is cut off. All non-firing torque points must be less than zero. Non-firing data is a function of engine speed only.
Generate response models	For both firing and non-firing data, the Model-Based Calibration Toolbox uses test plans to fit data to Gaussian process models (GPMs). To assess or adjust the response model fit, select **Edit in Application**. The Model-Based Calibration Toolbox Model Browser opens. For more information, see "Model Assessment" (Model-Based Calibration Toolbox).
Generate calibration	Model-Based Calibration Toolbox calibrates the firing and non-firing response models and generates calibrated tables. To assess or adjust the calibration, select **Edit in Application**. The Model-Based Calibration Toolbox CAGE Browser opens. For more information, see "Calibration Lookup Tables" (Model-Based Calibration Toolbox).
Update block parameters	Update the block lookup table and breakpoint parameters with the calibration.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Breakpoints for commanded fuel mass input, f_tbrake_f_bpt — Breakpoints
1-by-M vector

Breakpoints, in mg/inj.

Dependencies

Setting **Input command** to Fuel mass enables this parameter.

Breakpoints for commanded torque input, f_tbrake_t_bpt — Breakpoints
1-by-M vector

Breakpoints, in N·m.

Dependencies

Setting **Input command** to Torque enables this parameter.

Breakpoints for engine speed input, f_tbrake_n_bpt — Breakpoints
1-by-N vector

Breakpoints, in rpm.

Breakpoints for temperature input, f_tbrake_engtmp_bpt — Breakpoints
[233.15 273.15 373.15] (default) | 1-by-L vector

Breakpoints, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

Number of cylinders, NCyl — Number
4 (default) | scalar

Number of cylinders.

Crank revolutions per power stroke, Cps — Crank revolutions
2 (default) | scalar

Crank revolutions per power stroke.

Total displaced volume, Vd — Volume
0.0015 (default) | scalar

Volume displaced by engine, in m^3.

Fuel lower heating value, Lhv — Heating value
45e6 (default) | scalar

Fuel lower heating value, *LHV*, in J/kg.

Fuel specific gravity, Sg — Specific gravity
0.832 (default) | scalar

Specific gravity of fuel, Sg_{fuel}, dimensionless.

Ideal gas constant air, Rair — Constant
287 (default) | scalar

Ideal gas constant of air and residual gas entering the engine intake port, in J/(kg·K).

Air standard pressure, Pstd — Pressure
101325 (default) | scalar

Standard air pressure, in Pa.

Air standard temperature, Tstd — Temperature
293.15 (default) | scalar

Standard air temperature, in K.

Boost torque line, f_tbrake_bst — Boost lag
[90,95,95,95,96,100,104,104,104,100,95,85,75,67,60,55] (default) | 1-by-M vector

Boost torque line, $f_{bst}(N)$, in N·m.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Time constant below boost line — Time constant below
0.1 (default) | scalar

Time constant below boost line, τ_{nat}, in s.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Rising maximum fuel mass boost time constant, tau_bst_rising — Rising time constant
1.0 (default) | scalar

Rising maximum fuel mass boost time constant, $\tau_{bst,rising}$, in s.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Falling maximum fuel mass boost time constant, tau_bst_falling — Falling time constant
0.7 (default) | scalar
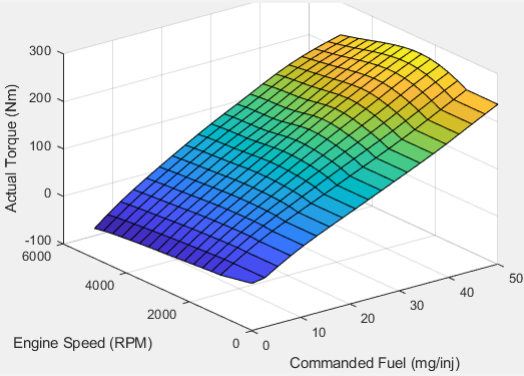
Falling maximum fuel mass boost time constant, $\tau_{bst,falling}$, in s.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Turbocharger time constant blend fuel mass fraction, f_blend_frac — Time constant
0.01 (default) | scalar

Turbocharger time constant blend fuel mass fraction, in s.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Power

Brake torque map, f_tbrake — 2D lookup table
M-by-N matrix

Input Command Setting	Description
`Fuel mass`	The engine brake torque lookup table is a function of commanded fuel mass and engine speed, $T_{brake} = f(F, N)$, where: • T_{brake} is engine torque, in N·m. • F is commanded fuel mass, in mg per injection. • N is engine speed, in rpm.
`Torque`	The engine brake torque lookup table is a function of target torque and engine speed, $T_{brake} = f(T_{target}, N)$, where: • T_{brake} is engine torque, in N·m. • T_{target} is target torque, in N·m. • N is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot brake torque map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Brake torque map, f_tbrake_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
`Fuel mass`	The engine brake torque lookup table is a function of commanded fuel mass and engine speed, $T_{brake} = f(F, N, Temp_{Eng})$, where: • T_{brake} is engine torque, in N·m. • F is commanded fuel mass, in mg per injection. • $Temp_{Eng}$ is engine temperature, in K.

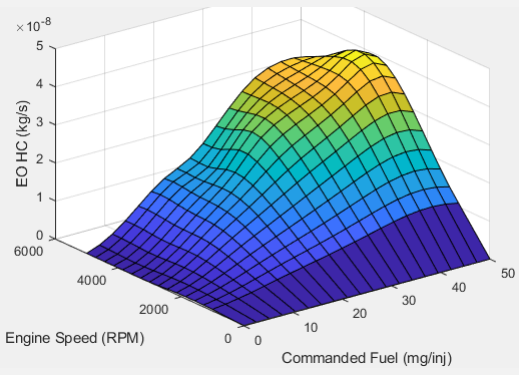| Input Command Setting	Description
Torque	The engine brake torque lookup table is a function of target torque and engine speed, $T_{brake} = f(T_{target}, N, Temp_{Eng})$, where: • T_{brake} is engine torque, in N·m. • T_{target} is target torque, in N·m. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

Air
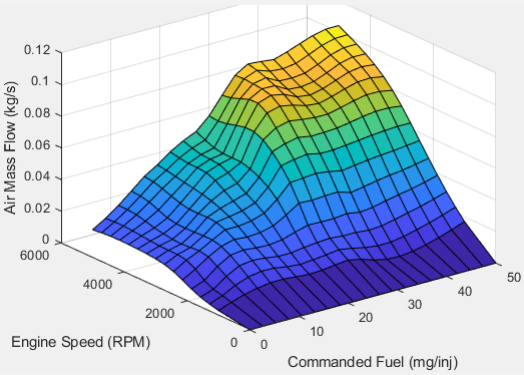
Air mass flow map, f_air — 2D lookup table
M-by-N matrix

Input Command Setting	Description
Fuel mass	The air mass flow lookup table is a function of commanded fuel mass and engine speed, $\dot{m}_{intk} = f(F_{max}, N)$, where: • $\dot{m}_{intk}$ is engine air mass flow, in kg/s. • F_{max} is commanded fuel mass, in mg per injection. • N is engine speed, in rpm.
Torque	The air mass flow lookup table is a function of maximum torque and engine speed, $\dot{m}_{intk} = f(T_{max}, N)$, where: • $\dot{m}_{intk}$ is engine air mass flow, in kg/s. • T_{max} is maximum torque, in N·m. • N is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot air mass map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Air mass flow map, f_air_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
Fuel mass	The air mass flow lookup table is a function of commanded fuel mass and engine speed, $\dot{m}_{intk} = f(F_{max}, N, Temp_{Eng})$, where: • $\dot{m}_{intk}$ is engine air mass flow, in kg/s. • F_{max} is commanded fuel mass, in mg per injection. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
Torque	The air mass flow lookup table is a function of maximum torque and engine speed, $\dot{m}_{intk} = f(T_{max}, N, Temp_{Eng})$, where: • $\dot{m}_{intk}$ is engine air mass flow, in kg/s. • T_{max} is maximum torque, in N·m. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
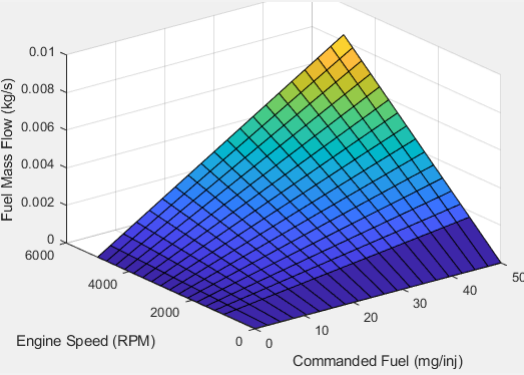
Dependencies

To enable this parameter, select **Input engine temperature**.

Fuel

Fuel flow map, f_fuel — 2D lookup table
M-by-N matrix

Input Command Setting	Description
Fuel mass	The engine fuel flow lookup table is a function of commanded fuel mass and engine speed, *MassFlow*= $f(F, N)$, where: • *MassFlow* is engine fuel mass flow, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm.
Torque	The engine fuel flow lookup table is a function of target torque and engine speed, *MassFlow* = $f(T_{target}, N)$, where: • *MassFlow* is engine fuel mass flow, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot fuel flow map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Fuel flow map, f_fuel_3d — 3D lookup table
M-by-N-by-L array

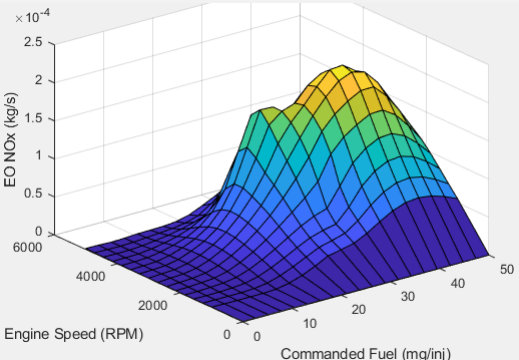| Input Command Setting	Description
Fuel mass	The engine fuel flow lookup table is a function of commanded fuel mass, engine speed, and engine temperature, $MassFlow = f(F, N, Temp_{Eng})$, where: • $MassFlow$ is engine fuel mass flow, in kg/s. • F is commanded fuel mass, in mg per injection. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
Torque	The engine fuel flow lookup table is a function of target torque and engine speed, and engine temperature, $MassFlow = f(T_{target}, N, Temp_{Eng})$, where: • $MassFlow$ is engine fuel mass flow, in kg/s. • T_{target} is target torque, in N·m. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
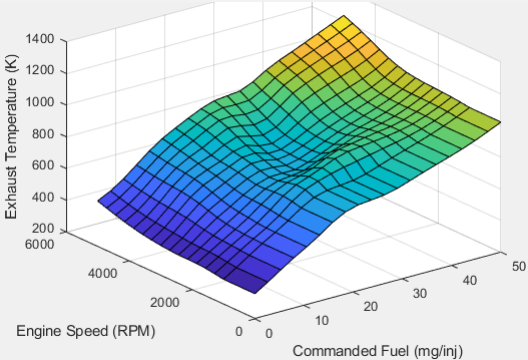
Dependencies

To enable this parameter, select **Input engine temperature**.

Temperature

Exhaust temperature map, f_texh — 2D lookup table
M-by-N matrix

Input Command Setting	Description
Fuel mass	The engine exhaust temperature table is a function of commanded fuel mass and engine speed, $T_{exh} = f(F, N)$, where: • T_{exh} is exhaust temperature, in K. • F is commanded fuel mass, in mg per injection. • N is engine speed, in rpm.

Input Command Setting	Description
Torque	The engine exhaust temperature table is a function of target torque and engine speed, $T_{exh} = f(T_{target}, N)$, where: • T_{exh} is exhaust temperature, in K. • T_{target} is target torque, in N·m. • N is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot exhaust temperature map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Exhaust temperature map, f_texh_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
Fuel mass	The engine exhaust temperature table is a function of commanded fuel mass and engine speed, $T_{exh} = f(F, N, Temp_{Eng})$, where: • T_{exh} is exhaust temperature, in K. • F is commanded fuel mass, in mg per injection. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
Torque	The engine exhaust temperature table is a function of target torque and engine speed, $T_{exh} = f(T_{target}, N, Temp_{Eng})$, where: • T_{exh} is exhaust temperature, in K. • T_{target} is target torque, in N·m. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
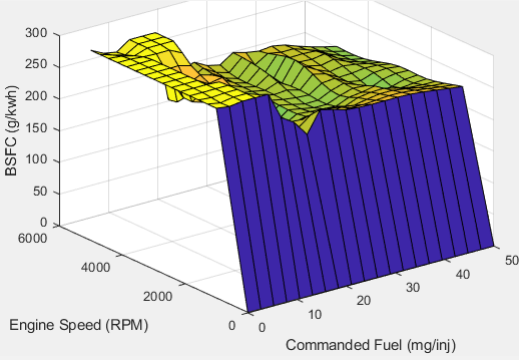
Dependencies

To enable this parameter, select **Input engine temperature**.

Efficiency

BSFC map, f_eff — 2D lookup table
M-by-N matrix

Input Command Setting	Description
Fuel mass	The brake-specific fuel consumption (BSFC) efficiency is a function of commanded fuel mass and engine speed, $BSFC = f(F, N)$, where: • $BSFC$ is BSFC, in g/kWh. • F is commanded fuel mass, in mg per injection. • N is engine speed, in rpm.
Torque	The brake-specific fuel consumption (BSFC) efficiency is a function of target torque and engine speed, $BSFC = f(T_{target}, N)$, where: • $BSFC$ is BSFC, in g/kWh. • T_{target} is target torque, in N·m. • N is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot BSFC map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

BSFC map, f_eff_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
Fuel mass	The brake-specific fuel consumption (BSFC) efficiency is a function of commanded fuel mass and engine speed, $BSFC = f(F, N, Temp_{Eng})$, where: • $BSFC$ is BSFC, in g/kWh. • F is commanded fuel mass, in mg per injection. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.

Input Command Setting	Description
Torque	The brake-specific fuel consumption (BSFC) efficiency is a function of target torque and engine speed, $BSFC = f(T_{target}, N, Temp_{Eng})$, where: • $BSFC$ is BSFC, in g/kWh. • T_{target} is target torque, in N·m. • N is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.

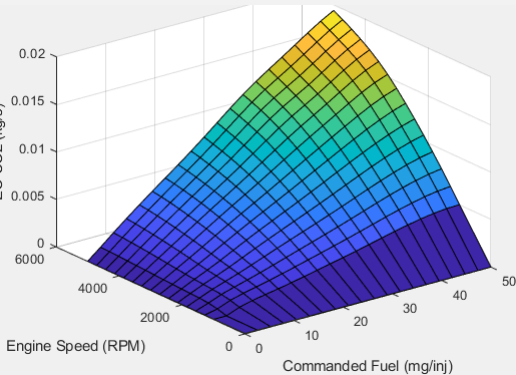**Dependencies**

To enable this parameter, select **Input engine temperature**.

HC

EO HC map, f_hc — 2D lookup table
M-by-N matrix

Input Command Setting	Description
Fuel mass	The engine-out hydrocarbon emissions are a function of commanded fuel mass and engine speed, $EO\ HC = f(F, N)$, where: • $EO\ HC$ is engine-out hydrocarbon emissions, in kg/s. • F is commanded fuel mass, in mg per injection. • N is engine speed, in rpm.
Torque	The engine-out hydrocarbon emissions are a function of target torque and engine speed, $EO\ HC = f(T_{target}, N)$, where: • $EO\ HC$ is engine-out hydrocarbon emissions, in kg/s. • T_{target} is target torque, in N·m. • N is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot EO HC map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO HC map, f_hc_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
Fuel mass	The engine-out hydrocarbon emissions are a function of commanded fuel mass and engine speed, *EO HC*= $f(F, N, Temp_{Eng})$, where: • *EO HC* is engine-out hydrocarbon emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
Torque	The engine-out hydrocarbon emissions are a function of target torque and engine speed, *EO HC* = $f(T_{target}, N, Temp_{Eng})$, where: • *EO HC* is engine-out hydrocarbon emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

CO
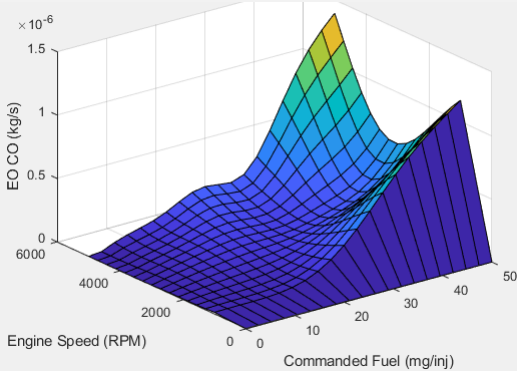
EO CO map, f_co — 2D lookup table
M-by-N matrix

Input Command Setting	Description
Fuel mass	The engine-out carbon monoxide emissions are a function of commanded fuel mass and engine speed, *EO CO*= ƒ(*F*, *N*), where: • *EO CO* is engine-out carbon monoxide emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm.
Torque	The engine-out carbon monoxide emissions are a function of target torque and engine speed, $EO\ CO = ƒ(T_{target}, N)$, where: • *EO CO* is engine-out carbon monoxide emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot EO CO map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO CO map, f_co_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
`Fuel mass`	The engine-out carbon monoxide emissions are a function of commanded fuel mass and engine speed, *EO CO*= $f(F, N, Temp_{Eng})$, where: • *EO CO* is engine-out carbon monoxide emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
`Torque`	The engine-out carbon monoxide emissions are a function of target torque and engine speed, *EO CO* = $f(T_{target}, N, Temp_{Eng})$, where: • *EO CO* is engine-out carbon monoxide emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

NOx

EO NOx map, f_nox — 2D lookup table
M-by-N matrix

Input Command Setting	Description
`Fuel mass`	The engine-out nitric oxide and nitrogen dioxide emissions are a function of commanded fuel mass and engine speed, *EO NOx*= $f(F, N)$, where: • *EO NOx* is engine-out nitric oxide and nitrogen dioxide emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm.

Input Command Setting	Description
Torque	The engine-out nitric oxide and nitrogen dioxide emissions are a function of target torque and engine speed, *EO NOx* = $f(T_{target}, N)$, where: • *EO NOx* is engine-out nitric oxide and nitrogen dioxide emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot EO NOx map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO NOx map, f_nox_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
Fuel mass	The engine-out nitric oxide and nitrogen dioxide emissions are a function of commanded fuel mass, engine speed, and engine temperature, *EO NOx* = $f(F, N, Temp_{Eng})$, where: • *EO NOx* is engine-out nitric oxide and nitrogen dioxide emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
Torque	The engine-out nitric oxide and nitrogen dioxide emissions are a function of target torque, engine speed, and engine temperature, *EO NOx* = $f(T_{target}, N, Temp_{Eng})$, where: • *EO NOx* is engine-out nitric oxide and nitrogen dioxide emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

CO2

EO CO2 map, f_co2 — 2D lookup table
M-by-N matrix

Input Command Setting	Description
Fuel mass	The engine-out carbon dioxide emissions are a function of commanded fuel mass and engine speed, *EO CO2*= ƒ(*F*, *N*), where: • *EO CO2* is engine-out carbon dioxide emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm.
Torque	The engine-out carbon dioxide emissions are a function of target torque and engine speed, *EO CO2* = ƒ(T_{target}, *N*), where: • *EO CO2* is engine-out carbon dioxide emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot CO2 map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO CO2 map, f_co2_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
Fuel mass	The engine-out carbon dioxide emissions are a function of commanded fuel mass, engine speed, and engine temperature, *EO CO2*= $f(F, N, Temp_{Eng})$, where: • *EO CO2* is engine-out carbon dioxide emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
Torque	The engine-out carbon dioxide emissions are a function of target torque, engine speed, and engine temperature, *EO CO2* = $f(T_{target}, N, Temp_{Eng})$, where: • *EO CO2* is engine-out carbon dioxide emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

PM

EO PM map, f_pm — 2D lookup table
M-by-N matrix

Input Command Setting	Description
Fuel mass	The engine-out PM emissions are a function of commanded fuel mass and engine speed, where: • *EO PM* is engine-out PM emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm.
Torque	The engine-out PM emissions are a function of target torque and engine speed, *EO PM* = $f(T_{target}, N)$, where: • *EO PM* is engine-out PM emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot EO PM map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO PM map, f_pm_3d — 3D lookup table
M-by-N-by-L array

Input Command Setting	Description
Fuel mass	The engine-out PM emissions are a function of commanded fuel mass, engine speed, and engine temperature, where: • *EO PM* is engine-out PM emissions, in kg/s. • *F* is commanded fuel mass, in mg per injection. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.
Torque	The engine-out PM emissions are a function of target torque, engine speed, and engine temperature, $EO\ PM = f(T_{target}, N, T)$, where: • *EO PM* is engine-out PM emissions, in kg/s. • T_{target} is target torque, in N·m. • *N* is engine speed, in rpm. • $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Mapped Motor | Mapped SI Engine

Topics
"Engine Calibration Maps"
"Model-Based Calibration Toolbox"

Introduced in R2017a

Mapped SI Engine

Spark-ignition engine model using lookup tables
Library: Powertrain Blockset / Propulsion / Combustion Engines
                                  Vehicle Dynamics Blockset / Powertrain / Propulsion
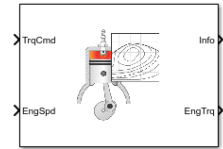
Description

The Mapped SI Engine block implements a mapped spark-ignition (SI) engine model using power, air mass flow, fuel flow, exhaust temperature, efficiency, and emission performance lookup tables. You can use the block for:

- Hardware-in-the-loop (HIL) engine control design
- Vehicle-level fuel economy and performance simulations

The block enables you to specify lookup tables for these engine characteristics. The lookup tables, developed with the Model-Based Calibration Toolbox, are functions of commanded torque, T_{cmd}, brake torque, T_{brake}, and engine speed, N. If you select **Input engine temperature**, the tables are also a function of engine temperature, $Temp_{Eng}$.

Table	Input Engine Temperature Parameter Setting	
	off	**on**
Power	$f(T_{cmd},N)$	$f(T_{cmd},N,Temp_{Eng})$
Air	$f(T_{brake},N)$	$f(T_{brake},N,Temp_{Eng})$
Fuel		
Temperature		
Efficiency		
HC		
CO		
NOx		
CO2		
PM		

To bound the Mapped SI Engine block output, the block does not extrapolate the lookup table data.

Virtual Calibration

If you have Model-Based Calibration Toolbox, click **Calibrate Maps** to virtually calibrate the 2D lookup tables using measured data. The dialog box steps through these tasks.

Task	Description
Import firing data	Import this loss data from a file. For example, open *<matlabroot>*/toolbox/autoblks/autodemos/projectsrc/SIDynamometer/CalMappedEng/SiEngineData.xlsx. For more information, see "Using Data" (Model-Based Calibration Toolbox). <table><tr><td>**Required Data**</td><td>**Optional Data**</td></tr><tr><td>• Engine speed, rpm • Engine torque, N·m</td><td>• Air mass flow rate, kg/s • Brake specific fuel consumption, g/(kW·h) • CO_2 mass flow rate, kg/s • CO mass flow rate, kg/s • Exhaust temperature, K • Fuel mass flow rate, kg/s • HC mass flow rate, kg/s • NOx mass flow rate, kg/s • Particulate matter mass flow rate, kg/s</td></tr></table> Collect firing data at steady-state operating conditions when injectors deliver the fuel. Data should cover the engine speed and torque operating range. Model-Based Calibration Toolbox uses the firing data boundary as the maximum torque. To filter or edit the data, select **Edit in Application**. The Model-Based Calibration Toolbox Data Editor opens.
Import non-firing data	Import this non-firing data from a file. For example, open *<matlabroot>*/toolbox/autoblks/autodemos/projectsrc/SIDynamometer/CalMappedEng/SiEngineData.xlsx. • Engine speed, rpm • Engine torque, N·m Collect non-firing (motoring) data at steady-state operating conditions when fuel is cut off. All non-firing torque points must be less than zero. Non-firing data is a function of engine speed only.
Generate response models	For both firing and non-firing data, the Model-Based Calibration Toolbox uses test plans to fit data to Gaussian process models (GPMs). To assess or adjust the response model fit, select **Edit in Application**. The Model-Based Calibration Toolbox Model Browser opens. For more information, see "Model Assessment" (Model-Based Calibration Toolbox).
Generate calibration	Model-Based Calibration Toolbox calibrates the firing and non-firing response models and generates calibrated tables. To assess or adjust the calibration, select **Edit in Application**. The Model-Based Calibration Toolbox CAGE Browser opens. For more information, see "Calibration Lookup Tables" (Model-Based Calibration Toolbox).

Task	Description
Update block parameters	Update the block lookup table and breakpoint parameters with the calibration.

Cylinder Air Mass

The block calculates the normalized cylinder air mass using these equations.

$$M_{Nom} = \frac{P_{std}V_d}{N_{cyl}R_{air}T_{std}}$$

$$L = \frac{\left(\frac{60s}{min}\right)Cps \cdot \dot{m}_{air}}{\left(\frac{1000g}{Kg}\right)N_{cyl} \cdot N \cdot M_{Nom}}$$

The equations use these variables.

L	Normalized cylinder air mass
M_{Nom}	Nominal engine cylinder air mass at standard temperature and pressure, piston at bottom dead center (BDC) maximum volume, in kg
Cps	Crankshaft revolutions per power stroke, rev/stroke
P_{std}	Standard pressure
T_{std}	Standard temperature
R_{air}	Ideal gas constant for air and burned gas mixture
V_d	Displaced volume
N_{cyl}	Number of engine cylinders
N	Engine speed
$\dot{m}_{intk}$	Engine air mass flow, in g/s

Turbocharger Lag

To model turbocharger lag, select **Include turbocharger lag effect**. During throttle control, the time constant models the manifold filling and emptying dynamics. When the torque request requires a turbocharger boost, the block uses a larger time constant to represent the turbocharger lag. The block uses these equations.

Dynamic torque	$\dfrac{dT_{brake}}{dt} = \dfrac{1}{\tau_{eng}}(T_{stdy} - T_{brake})$
Boost time constant	$\tau_{bst} = \begin{cases} \tau_{bst,\,rising} & \text{when } T_{stdy} > T_{brake} \\ \tau_{bst,\,falling} & \text{when } T_{stdy} \le T_{brake} \end{cases}$
Final time constant	$\tau_{eng} = \begin{cases} \tau_{thr} & \text{when } T_{brake} < f_{bst}(N) \\ \tau_{bst} & \text{when } T_{brake} \ge f_{bst}(N) \end{cases}$

The equations use these variables.

T_{brake}	Brake torque

T_{stdy}	Steady-state target torque
τ_{bst}	Boost time constant
$\tau_{bst,rising}$, $\tau_{bst,falling}$	Boost rising and falling time constant, respectively
τ_{eng}	Final time constant
τ_{thr}	Time constant during throttle control
$f_{bst}(N)$	Boost torque speed line
N	Engine speed

Fuel Flow

To calculate the fuel economy for high-fidelity models, the block uses the volumetric fuel flow.

$$Q_{fuel} = \frac{\dot{m}_{fuel}}{\left(\frac{1000kg}{m^3}\right)Sg_{fuel}}$$

The equation uses these variables.

$\dot{m}_{fuel}$	Fuel mass flow
Sg_{fuel}	Specific gravity of fuel
Q_{fuel}	Volumetric fuel flow

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrCrkshft	Crankshaft power	$-\tau_{eng}\omega$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrFuel	Fuel input power	$\dot{m}_{fuel}LHV$
		PwrLoss	Power loss	$\tau_{eng}\omega - \dot{m}_{fuel}LHV$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	*Not used*		

The equations use these variables.

LHV	Fuel lower heating value

ω	Engine speed, rad/s
$\dot{m}_{fuel}$	Fuel mass flow
τ_{eng}	Fuel mass per injection time constant

Ports

Input

TrqCmd — Commanded torque
scalar

Torque, T_{cmd}, in N·m.

EngSpd — Engine speed
scalar

Engine speed, N, in rpm.

EngTemp — Engine temperature
scalar

Engine temperature, $Temp_{Eng}$, in K.

Dependencies

To enable this port, select **Input engine temperature**.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal	Description	Units
IntkGassMassFlw	Engine air mass flow output	kg/s
NrmlzdAirChrg	Normalized engine cylinder air mass	N/A
Afr	Air-fuel ratio (AFR)	N/A
FuelMassFlw	Engine fuel flow output	kg/s
FuelVolFlw	Volumetric fuel flow	m³/s
ExhManGasTemp	Engine exhaust gas temperature	K
EngTrq	Engine torque output	N·m
EngSpd	Engine speed	rpm

Signal			Description	Units
CrkAng			Engine crankshaft absolute angle $$\int_{0}^{(360)Cps} EngSpd\frac{180}{30}d\theta$$ where *Cps* is crankshaft revolutions per power stroke.	degrees crank angle
Bsfc			Engine brake-specific fuel consumption (BSFC)	g/kWh
EoHC			Engine out hydrocarbon emission mass flow	kg/s
EoCO			Engine out carbon monoxide emission mass flow rate	kg/s
EoNOx			Engine out nitric oxide and nitrogen dioxide emissions mass flow	kg/s
EoCO2			Engine out carbon dioxide emission mass flow	kg/s
EoPM			Engine out particulate matter emission mass flow	kg/s
PwrInfo	PwrTrnsfrd	PwrCrkshft	Crankshaft power	W
	PwrNotTrnsfrd	PwrFuel	Fuel input power	W
		PwrLoss	Power loss	W
	PwrStored		*Not used*	

EngTrq — Engine brake torque
scalar

Engine brake torque, T_{brake}, in N·m.

Parameters

Block Options

Include turbocharger lag effect — Increase time constant
off (default)

To model turbocharger lag, select **Include turbocharger lag effect**. During throttle control, the time constant models the manifold filling and emptying dynamics. When the torque request requires a turbocharger boost, the block uses a larger time constant to represent the turbocharger lag. The block uses these equations.

Dynamic torque	$\dfrac{dT_{brake}}{dt} = \dfrac{1}{\tau_{eng}}(T_{stdy} - T_{brake})$

Boost time constant	$\tau_{bst} = \begin{cases} \tau_{bst,\,rising} & \text{when } T_{stdy} > T_{brake} \\ \tau_{bst,\,falling} & \text{when } T_{stdy} \le T_{brake} \end{cases}$
Final time constant	$\tau_{eng} = \begin{cases} \tau_{thr} & \text{when } T_{brake} < f_{bst}(N) \\ \tau_{bst} & \text{when } T_{brake} \ge f_{bst}(N) \end{cases}$

The equations use these variables.

T_{brake}	Brake torque
T_{stdy}	Steady-state target torque
τ_{bst}	Boost time constant
$\tau_{bst,rising}$, $\tau_{bst,falling}$	Boost rising and falling time constant, respectively
τ_{eng}	Final time constant
τ_{thr}	Time constant during throttle control
$f_{bst}(N)$	Boost torque speed line
N	Engine speed

Dependencies

Selecting **Include turbocharger lag effect** enables these parameters:

- **Boost torque line, f_tbrake_bst**
- **Time constant below boost line, tau_thr**
- **Rising torque boost time constant, tau_bst_rising**
- **Falling torque boost time constant, tau_bst_falling**

Input engine temperature — Create input port
off (default) | on

Select this to create the EngTemp input port.

The block enables you to specify lookup tables for these engine characteristics. The lookup tables, developed with the Model-Based Calibration Toolbox, are functions of commanded torque, T_{cmd}, brake torque, T_{brake}, and engine speed, N. If you select **Input engine temperature**, the tables are also a function of engine temperature, $Temp_{Eng}$.

Table	Input Engine Temperature Parameter Setting	
	off	on
Power	$f(T_{cmd},N)$	$f(T_{cmd},N,Temp_{Eng})$
Air	$f(T_{brake},N)$	$f(T_{brake},N,Temp_{Eng})$
Fuel		
Temperature		
Efficiency		
HC		
CO		

Table	Input Engine Temperature Parameter Setting	
	off	**on**
NOx		
CO2		
PM		

Configuration

`Calibrate Maps` **— Calibrate tables with measured data**
`selection`

If you have Model-Based Calibration Toolbox, click **Calibrate Maps** to virtually calibrate the 2D lookup tables using measured data. The dialog box steps through these tasks.

Task	Description
Import firing data	Import this loss data from a file. For example, open *<matlabroot>*`/toolbox/autoblks/autodemos/projectsrc/SIDynamometer/CalMappedEng/SiEngineData.xlsx`.

For more information, see "Using Data" (Model-Based Calibration Toolbox).

Required Data	Optional Data
• Engine speed, rpm • Engine torque, N·m	• Air mass flow rate, kg/s • Brake specific fuel consumption, g/(kW·h) • CO2 mass flow rate, kg/s • CO mass flow rate, kg/s • Exhaust temperature, K • Fuel mass flow rate, kg/s • HC mass flow rate, kg/s • NOx mass flow rate, kg/s • Particulate matter mass flow rate, kg/s

Collect firing data at steady-state operating conditions when injectors deliver the fuel. Data should cover the engine speed and torque operating range. Model-Based Calibration Toolbox uses the firing data boundary as the maximum torque.

To filter or edit the data, select **Edit in Application**. The Model-Based Calibration Toolbox Data Editor opens.

Task	Description
Import non-firing data	Import this non-firing data from a file. For example, open *\<matlabroot\>*/`toolbox/autoblks/autodemos/projectsrc/SIDynamometer/` `CalMappedEng/SiEngineData.xlsx`. • Engine speed, rpm • Engine torque, N·m Collect non-firing (motoring) data at steady-state operating conditions when fuel is cut off. All non-firing torque points must be less than zero. Non-firing data is a function of engine speed only.
Generate response models	For both firing and non-firing data, the Model-Based Calibration Toolbox uses test plans to fit data to Gaussian process models (GPMs). To assess or adjust the response model fit, select **Edit in Application**. The Model-Based Calibration Toolbox Model Browser opens. For more information, see "Model Assessment" (Model-Based Calibration Toolbox).
Generate calibration	Model-Based Calibration Toolbox calibrates the firing and non-firing response models and generates calibrated tables. To assess or adjust the calibration, select **Edit in Application**. The Model-Based Calibration Toolbox CAGE Browser opens. For more information, see "Calibration Lookup Tables" (Model-Based Calibration Toolbox).
Update block parameters	Update the block lookup table and breakpoint parameters with the calibration.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Breakpoints for commanded torque, f_tbrake_t_bpt — Breakpoints
1-by-M vector

Breakpoints, in N·m.

Breakpoints for engine speed input, f_tbrake_n_bpt — Breakpoints
1-by-N vector

Breakpoints, in rpm.

Breakpoints for temperature input, f_tbrake_engtmp_bpt — Breakpoints
[233.15 273.15 373.15] (default) | 1-by-L vector

Breakpoints, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

Number of cylinders, NCyl — Number
4 (default) | scalar

Number of cylinders.

Crank revolutions per power stroke, Cps — Crank revolutions
2 (default) | scalar

Crank revolutions per power stroke.

Total displaced volume, Vd — Volume
0.0015 (default) | scalar

Volume displaced by engine, in m^3.

Fuel lower heating value, Lhv — Heating value
45e6 (default) | scalar

Fuel lower heating value, *LHV*, in J/kg.

Fuel specific gravity, Sg — Specific gravity
0.745 (default) | scalar

Specific gravity of fuel, Sg_{fuel}, dimensionless.

Ideal gas constant air, Rair — Constant
287 (default) | scalar

Ideal gas constant of air and residual gas entering the engine intake port, in J/(kg*K).

Air standard pressure, Pstd — Pressure
101325 (default) | scalar

Standard air pressure, in Pa.

Air standard temperature, Tstd — Temperature
293.15 (default) | scalar

Standard air temperature, in K.

Boost torque line, f_tbrake_bst — Boost lag
1-by-M vector

Boost torque line, $f_{bst}(N)$, in N·m.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Time constant below boost line — Time constant below
0.2 (default) | scalar

Time constant below boost line, τ_{thr}, in s.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Rising torque boost time constant, tau_bst_rising — Rising time constant
1.5 (default) | scalar

Rising torque boost time constant, $\tau_{bst,rising}$, in s.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Falling torque boost time constant, tau_bst_falling — Falling time constant
1 (default) | scalar

Falling torque boost time constant, $\tau_{bst,falling}$, in s.

Dependencies

To enable this parameter, select **Include turbocharger lag effect**.

Power

Brake torque map, f_tbrake — 2D lookup table
M-by-N matrix

The engine torque lookup table is a function of commanded engine torque and engine speed, $T = f(T_{cmd}, N)$, where:

- T is engine torque, in N·m.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.
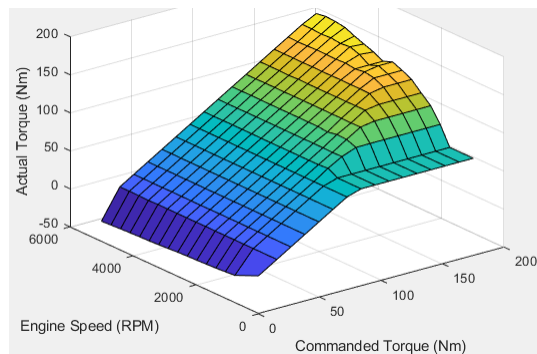
Plot brake torque map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Brake torque map, f_tbrake_3d — 3D lookup table
M-by-N-by-L array

The engine torque lookup table is a function of commanded engine torque, engine speed, and engine temperature, $T = f(T_{cmd}, N, Temp_{Eng})$, where:

- T is engine torque, in N·m.
- T_{cmd} is commanded engine torque, in N·m.

- N is engine speed, in rpm.
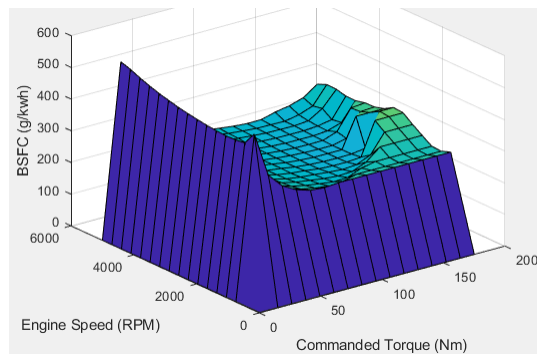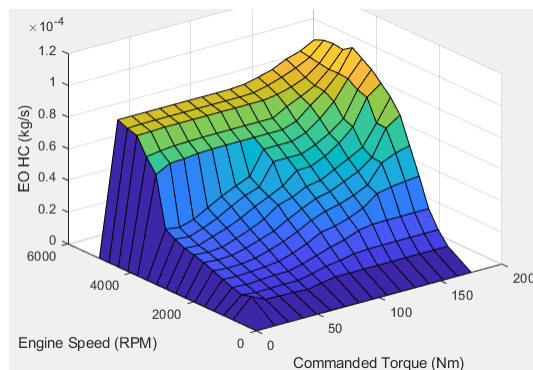- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

Air

Air mass flow map, f_air — 2D lookup table
M-by-N matrix

The engine air mass flow lookup table is a function of commanded engine torque and engine speed, $\dot{m}_{intk} = f(T_{cmd}, N)$, where:

- $\dot{m}_{intk}$ is engine air mass flow, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.
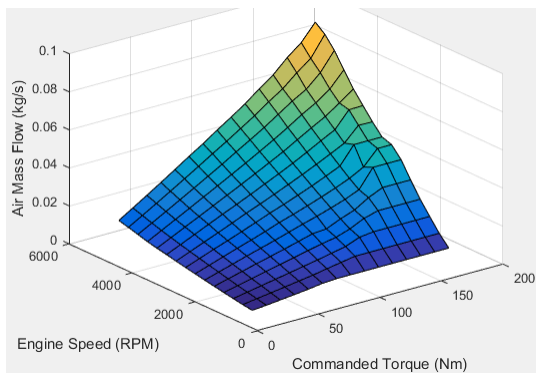
Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot air mass map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Air mass flow map, f_air_3d — 3D lookup table
M-by-N-by-L array

The engine air mass flow lookup table is a function of commanded engine torque, engine speed, and engine temperature, $\dot{m}_{intk} = f(T_{cmd}, N, Temp_{Eng})$, where:

- $\dot{m}_{intk}$ is engine air mass flow, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.

- N is engine speed, in rpm.
- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

Fuel

Fuel flow map, f_fuel — 2D lookup table
M-by-N matrix

The engine fuel mass flow lookup table is a function of commanded engine torque and engine speed, $MassFlow = f(T_{cmd}, N)$, where:

- $MassFlow$ is engine fuel mass flow, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.
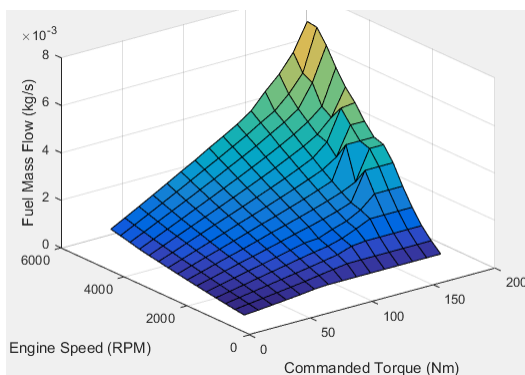
Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot fuel flow map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Fuel flow map, f_fuel_3d — 3D lookup table
M-by-N-by-L array

The engine fuel mass flow lookup table is a function of commanded engine torque, engine speed, and engine temperature, $MassFlow = f(T_{cmd}, N, Temp_{Eng})$, where:

- $MassFlow$ is engine fuel mass flow, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.

- $Temp_{Eng}$ is engine temperature, in K.
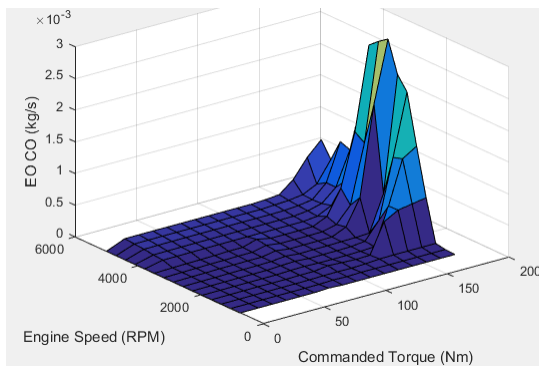
Dependencies

To enable this parameter, select **Input engine temperature**.

Temperature

Exhaust temperature map, f_texh — 2D lookup table
M-by-N matrix

The engine exhaust temperature lookup table is a function of commanded engine torque and engine speed, $T_{exh} = f(T_{cmd}, N)$, where:

- T_{exh} is exhaust temperature, in K.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.
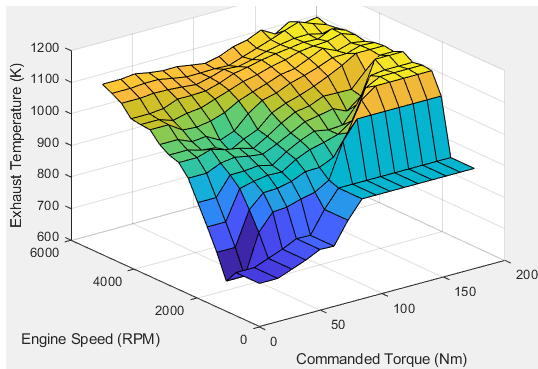
Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot exhaust temperature map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Exhaust temperature map, f_texh_3d — 3D lookup table
array

The engine exhaust temperature lookup table is a function of commanded engine torque, engine speed, and engine temperature, $T_{exh} = f(T_{cmd}, N, Temp_{Eng})$, where:

- T_{exh} is exhaust temperature, in K.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.
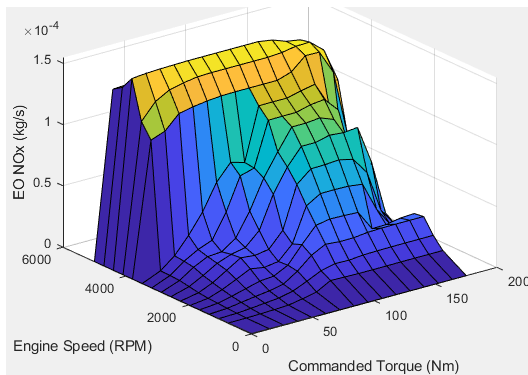- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

Efficiency

BSFC map, f_eff — 2D lookup table
M-by-N-by-L array

The brake-specific fuel consumption (BSFC) efficiency is a function of commanded engine torque and engine speed, $BSFC = f(T_{cmd}, N)$, where:

- $BSFC$ is BSFC, in g/kWh.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot BSFC map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

BSFC map, f_eff_3d — 3D lookup table
M-by-N-by-L array

The brake-specific fuel consumption (BSFC) efficiency is a function of commanded engine torque, engine speed, and engine temperature, $BSFC = f(T_{cmd}, N, Temp_{Eng})$, where:

- $BSFC$ is BSFC, in g/kWh.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.
- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

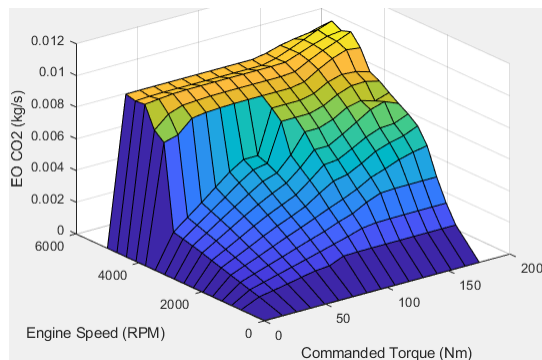To enable this parameter, select **Input engine temperature**.

HC

EO HC map, f_hc — 2D lookup table
M-by-N matrix

The engine-out hydrocarbon emissions are a function of commanded engine torque and engine speed, $EO\ HC = f(T_{cmd}, N)$, where:

- $EO\ HC$ is engine-out hydrocarbon emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot EO HC map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO HC map, f_hc_3d — 3D lookup table
M-by-N-by-L array

The engine-out hydrocarbon emissions are a function of commanded engine torque, engine speed, and engine temperature, $EO\ HC = f(T_{cmd}, N, Temp_{Eng})$, where:

- $EO\ HC$ is engine-out hydrocarbon emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.
- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

CO

EO CO map, f_co — 2D lookup table
M-by-N matrix

The engine-out carbon monoxide emissions are a function of commanded engine torque and engine speed, *EO CO* = $f(T_{cmd}, N)$, where:

- *EO CO* is engine-out carbon monoxide emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- *N* is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot EO CO map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO HC map, f_hc_3d — 3D lookup table
M-by-N-by-L array

The engine-out hydrocarbon emissions are a function of commanded engine torque, engine speed, and engine temperature, *EO HC* = $f(T_{cmd}, N, Temp_{Eng})$, where:

- *EO HC* is engine-out hydrocarbon emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- *N* is engine speed, in rpm.
- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

NOx

EO NOx map, f_nox — 2D lookup table
M-by-N matrix

The engine-out nitric oxide and nitrogen dioxide emissions are a function of commanded engine torque and engine speed, $EO\ NOx = f(T_{cmd}, N)$, where:

- $EO\ NOx$ is engine-out nitric oxide and nitrogen dioxide emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot EO NOx map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO NOx map, f_nox_3d — 3D lookup table
M-by-N-by-L array

The engine-out nitric oxide and nitrogen dioxide emissions are a function of commanded engine torque, engine speed, and engine temperature, $EO\ NOx = f(T_{cmd}, N, Temp_{Eng})$, where:

- $EO\ NOx$ is engine-out nitric oxide and nitrogen dioxide emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- N is engine speed, in rpm.
- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

CO2

EO CO2 map, f_co2 — 2D lookup table
M-by-N matrix

The engine-out carbon dioxide emissions are a function of commanded engine torque and engine speed, *EO CO2* = $f(T_{cmd}, N)$, where:

- *EO CO2* is engine-out carbon dioxide emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- *N* is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot CO2 map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO CO2 map, f_co2_3d — 3D lookup table
M-by-N-by-L array

The engine-out carbon dioxide emissions are a function of commanded engine torque, engine speed, and engine temperature, *EO CO2* = $f(T_{cmd}, N, Temp_{Eng})$, where:

- *EO CO2* is engine-out carbon dioxide emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- *N* is engine speed, in rpm.
- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

PM

EO PM map, f_pm — 2D lookup table
M-by-N matrix

The engine-out particulate matter emissions are a function of commanded engine torque and engine speed, where:

- *EO PM* is engine-out PM emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- *N* is engine speed, in rpm.

Dependencies

To enable this parameter, clear **Input engine temperature**.

Plot EO PM map — Plot table
button

Click to plot table.

Dependencies

To enable this parameter, clear **Input engine temperature**.

EO PM map, f_pm_3d — 3D lookup table
M-by-N-by-L array

The engine-out particulate matter emissions are a function of commanded engine torque, engine speed, and engine temperature, where:

- *EO PM* is engine-out PM emissions, in kg/s.
- T_{cmd} is commanded engine torque, in N·m.
- *N* is engine speed, in rpm.
- $Temp_{Eng}$ is engine temperature, in K.

Dependencies

To enable this parameter, select **Input engine temperature**.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Mapped CI Engine | Mapped Motor

Topics
"Engine Calibration Maps"
"Model-Based Calibration Toolbox"

Introduced in R2017a

Vehicle Dynamics Blocks

Vehicle Body Total Road Load

Vehicle motion using coast-down testing coefficients
Library: Powertrain Blockset / Vehicle Dynamics
                      Vehicle Dynamics Blockset / Vehicle Body
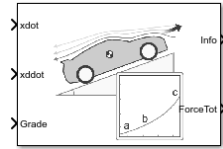
Description

The Vehicle Body Total Road Load block implements a one degree-of-freedom (1DOF) rigid vehicle model using coast-down testing coefficients. You can use this block in a vehicle model to represent the load that the driveline and chassis applies to a transmission or engine. It is suitable for system-level performance, component sizing, fuel economy, or drive cycle tracking studies. The block calculates the dynamic powertrain load with minimal parameterization or computational cost.

You can configure the block for kinematic, force, or total power input.

- Kinematic — Block uses the vehicle longitudinal velocity and acceleration to calculate the tractive force and power.
- Force — Block uses the tractive force to calculate the vehicle longitudinal displacement and velocity.
- Power — Block uses the engine or transmission power to calculate the vehicle longitudinal displacement and velocity.

Dynamics

To calculate the total road load acting on the vehicle, the block implements this equation.

$$F_{road} = a + b\dot{x} + c\dot{x}^2 + mg\sin(\theta)$$

To determine the coefficients a, b, and c, you can use a test procedure similar to the one described in *Road Load Measurement and Dynamometer Simulation Using Coastdown Techniques*. You can also use Simulink® Design Optimization™ to fit the coefficients to measured data.

To calculate the vehicle motion, the block uses Newton's law for rigid bodies.

$$F_{total} = m\ddot{x} + F_{road}$$

Total power input is a product of the total force and longitudinal velocity. Power due to road and gravitational forces is a product of the road force and longitudinal velocity.

$$P_{total} = F_{total}\dot{x}$$
$$P_{road} = F_{road}\dot{x}$$

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Variable	Equations
PwrIn fo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrFxE xt	Externally applied force power	P_{FxExt}	$P_{FxExt} = F_{total}\dot{x}$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrFxD rag	Drag force power	P_D	$P_d = -(a + b\dot{x} + c\dot{x}^2)\dot{x}$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	wrStor edGrvt y	Rate change in gravitational potential energy	P_g	$P_g = -mg\dot{Z}$
		PwrSto redxdo t	Rate in change of longitudinal kinetic energy	P_{xdot}	$P_{\dot{x}} = m\ddot{x}\dot{x}$

The equations use these variables.

a	Steady-state rolling resistance coefficient
b	Viscous driveline and rolling resistance coefficient
c	Aerodynamic drag coefficient
g	Gravitational acceleration
x	Vehicle longitudinal displacement with respect to ground, in the vehicle-fixed frame
$\dot{x}$	Vehicle longitudinal velocity with respect to ground, in the vehicle-fixed frame
$\ddot{x}$	Vehicle longitudinal acceleration with respect to ground, vehicle-fixed frame
m	Vehicle body mass
Θ	Road grade angle
F_{total}	Total force acting on vehicle
F_{road}	Resistive road load due to losses and gravitational load
P_{total}	Total tractive input power
P_{road}	Total power due to losses and gravitational load
$\dot{Z}$	Vehicle vertical velocity along the vehicle-fixed z-axis

Ports

Input

xdot — Vehicle longitudinal velocity
scalar

Vehicle total longitudinal velocity, $\dot{x}$, in m/s.

Dependencies

To enable this port, for the **Input Mode** parameter, select Kinematic.

xddot — Vehicle longitudinal acceleration
scalar

Vehicle total longitudinal acceleration, $\ddot{x}$, in m/s^2.

Dependencies

To enable this port, for the **Input Mode** parameter, select Kinematic.

PwrTot — Tractive input power
scalar

Tractive input power, P_{total}, in W.

Dependencies

To enable this port, for the **Input Mode** parameter, select Power.

ForceTot — Tractive input force
scalar

Tractive input force, F_{total}, in N.

Dependencies

To enable this port, for the **Input Mode** parameter, select Force.

Grade — Road grade angle
scalar

Road grade angle, Θ, in deg.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal				Description	Value	Units
In er	Cg	Disp	X	Vehicle CG displacement along earth-fixed X-axis	Computed	m

Signal				Description	Value	Units
tFrm			Y	Vehicle CG displacement along earth-fixed Y-axis	0	m
			Z	Vehicle CG displacement along earth-fixed Z-axis	Computed	m
		Vel	Xdot	Vehicle CG velocity along earth-fixed X-axis	Computed	m/s
			Ydot	Vehicle CG velocity along earth-fixed Y-axis	0	m/s
			Zdot	Vehicle CG velocity along earth-fixed Z-axis	Computed	m/s
		Ang	phi	Rotation of vehicle-fixed frame about the earth-fixed X-axis (roll)	0	rad
			theta	Rotation of vehicle-fixed frame about the earth-fixed Y-axis (pitch)	Computed	rad
			psi	Rotation of vehicle-fixed frame about the earth-fixed Z-axis (yaw)	0	rad
BdyFrm	Cg	Disp	x	Vehicle CG displacement along the vehicle-fixed x-axis	Computed	m
			y	Vehicle CG displacement along the vehicle-fixed y-axis	0	m
			z	Vehicle CG displacement along the vehicle-fixed z-axis	0	m
		Vel	xdot	Vehicle CG velocity along the vehicle-fixed x-axis	Computed	m/s
			ydot	Vehicle CG velocity along the vehicle-fixed y-axis	0	m/s
			zdot	Vehicle CG velocity along the vehicle-fixed z-axis	0	m/s
		Acc	ax	Vehicle CG acceleration along the vehicle-fixed x-axis	Computed	gn
			ay	Vehicle CG acceleration along the vehicle-fixed y-axis	0	gn
			az	Vehicle CG acceleration along the vehicle-fixed z-axis	0	gn
	Forces	Body	Fx	Net force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy	Net force on vehicle CG along the vehicle-fixed y-axis	0	N
			Fz	Net force on vehicle CG along the vehicle-fixed z-axis	0	N
		Ext	Fx	External force on vehicle CG along the vehicle-fixed x-axis	Computed	N

Signal				Description	Value	Units
			Fy	External force on vehicle CG along the vehicle-fixed y-axis	0	N
			Fz	External force on vehicle CG along the vehicle-fixed z-axis	0	N
		Drag	Fx	Drag force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy	Drag force on vehicle CG along the vehicle-fixed y-axis	0	N
			Fz	Drag force on vehicle CG along the vehicle-fixed z-axis	0	N
		Grvty	Fx	Gravity force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy	Gravity force on vehicle CG along the vehicle-fixed y-axis	0	N
			Fz	Gravity force on vehicle CG along the vehicle-fixed z-axis	Computed	N
	Pwr	PwrExt		Applied external power	Computed	W
		Drag		Power loss due to drag	Computed	W
PwrInfo	PwrTrnsfrd	PwrFxExt		Externally applied force power	P_{FxExt}	W
	PwrNotTrnsfrd	PwrFxDrag		Drag force power	P_D	W
	PwrStored	wrStoredGrvty		Rate change in gravitational potential energy	P_g	W
		PwrStoredxdot		Rate in change of longitudinal kinetic energy	P_{xdot}	W

xdot — Vehicle longitudinal velocity
scalar

Vehicle total longitudinal velocity, $\dot{x}$, in m/s.

Dependencies

To enable this port, for the **Input Mode** parameter, select Power or Force.

ForceTot — Tractive input force
scalar

Tractive input force, F_{total}, in N.

Dependencies

To enable this port, for the **Input Mode** parameter, select Kinematic.

Parameters

Input Mode — Specify input mode
Kinematic (default) | Force | Power

Specify the input type.

- Kinematic — Block uses the vehicle longitudinal velocity and acceleration to calculate the tractive force and power. Use this configuration for powertrain, driveline, and braking system design, or component sizing.
- Force — Block uses the tractive force to calculate the vehicle longitudinal displacement and velocity. Use this configuration for system-level performance, fuel economy, or drive cycle tracking studies.
- Power — Block uses the engine or transmission power to calculate the vehicle longitudinal displacement and velocity. Use this configuration for system-level performance, fuel economy, or drive cycle tracking studies.

Dependencies

This table summarizes the port and input mode configurations.

Input Mode	Creates Ports
Kinematic	xdot
	xddot
Force	Force
Power	Power

Mass — Vehicle body mass
1200 (default) | scalar

Vehicle body mass, *m*, in kg.

Rolling resistance coefficient, a — Rolling
196 (default) | scalar

Steady-state rolling resistance coefficient, *a*, in N.

Rolling and driveline resistance coefficient, b — Rolling and driveline
2.232 (default) | scalar

Viscous driveline and rolling resistance coefficient, *b*, in N*s/m.

Aerodynamic drag coefficient, c — Drag
0.389 (default) | scalar

Aerodynamic drag coefficient, *c*, in N·s^2/m.

Gravitational acceleration, g — Gravity
9.81 (default) | scalar

Gravitational acceleration, *g*, in m/s^2.

Initial position, x_o — Position
0 (default) | scalar

Vehicle longitudinal initial position, in m.

Initial velocity, xdot_o — Velocity
0 (default) | scalar

Vehicle longitudinal initial velocity with respect to ground, in m/s.

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers (SAE), 1992.

[2] Light Duty Vehicle Performance And Economy Measure Committee. *Road Load Measurement and Dynamometer Simulation Using Coastdown Techniques*. Standard J1263_201003. SAE International, March 2010.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Drive Cycle Source | Vehicle Body 1DOF Longitudinal | Vehicle Body 3DOF Longitudinal

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2017a

Vehicle Body 1DOF Longitudinal

Two-axle vehicle in forward and reverse motion
Library: Powertrain Blockset / Vehicle Dynamics
                        Vehicle Dynamics Blockset / Vehicle Body
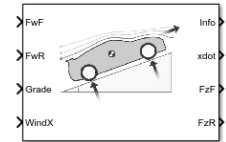
Description

The Vehicle Body 1DOF Longitudinal block implements a one degree-of-freedom (1DOF) rigid vehicle body with constant mass undergoing longitudinal (that is, forward and reverse) motion. Use the block:

- In powertrain and fuel economy studies to represent the vehicle inertial and drag loads when weight transfer from vertical and pitch motions are negligible.
- To determine the engine torque and power required for the vehicle to follow a specified drive cycle.

You can select block options to create input ports for external forces, moments, air temperature, and wind speed.

Block Option Setting	External Input Ports	Description
External forces	FExt	External force applied to vehicle CG in the vehicle-fixed frame.
External moments	MExt	External moment about vehicle CG in the vehicle-fixed frame.
Air temperature	AirTemp	Ambient air temperature. Consider this option if you want to vary the temperature during run-time.
Wind X,Y,Z	WindXYZ	Wind speed along earth-fixed *X*-, *Y*-, and *Z*-axes. If you do not select this option, the block implements input port WindX — Longitudinal wind speed along the earth-fixed *X*-axis.

Vehicle Body Model

The vehicle axles are parallel and form a plane. The longitudinal direction lies in this plane and is perpendicular to the axles. If the vehicle is traveling on an inclined slope, the normal direction is not parallel to gravity but is always perpendicular to the axle-longitudinal plane.

The block uses the net effect of all the forces and torques acting on it to determine the vehicle motion. The longitudinal tire forces push the vehicle forward or backward. The weight of the vehicle acts through its center of gravity (CG). The grade angle changes the direction of the resolved gravitational force acting on the vehicle CG. Similarly, the block resolves the resistive aerodynamic drag force on the vehicle CG.
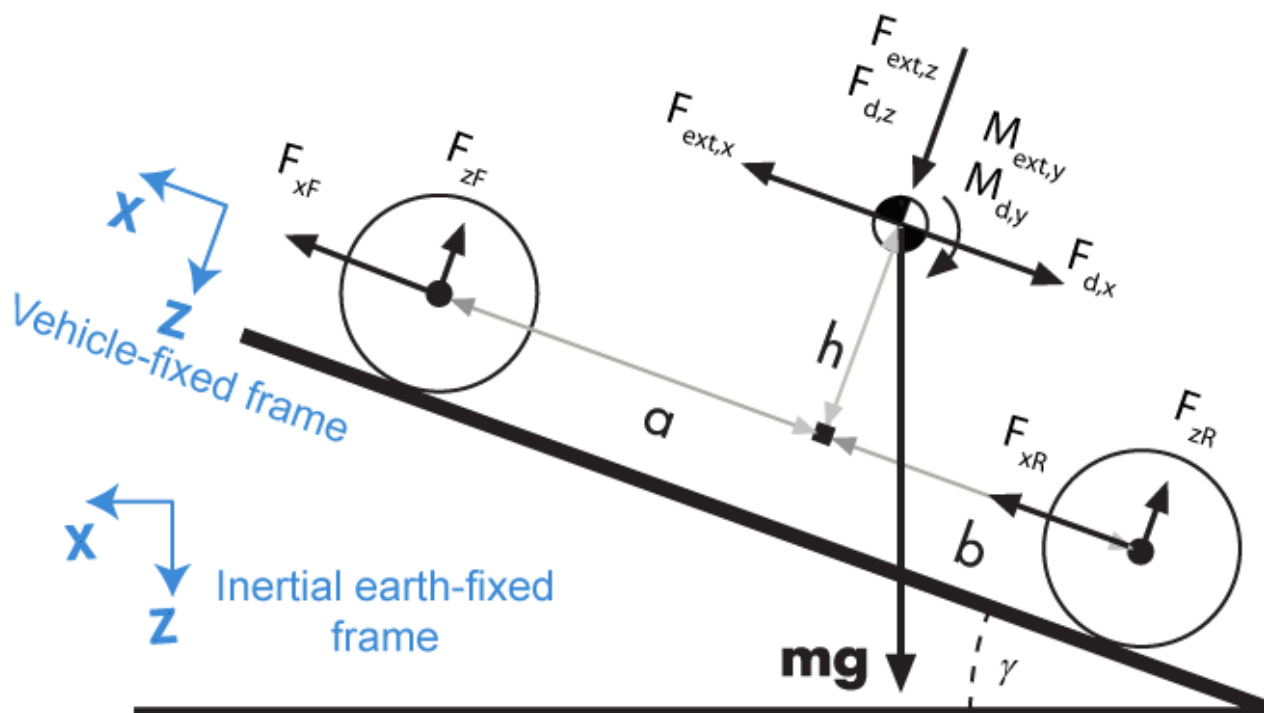
The Vehicle Body 1DOF Longitudinal block implements these equations.

$$F_b = m\ddot{x}$$

$$F_b = F_{xF} + F_{xR} - F_{d,x} + F_{ext,x} - mg\sin\gamma$$

Zero normal acceleration and zero pitch torque determine the normal force on each front and rear axles.

$$F_{zF} = \frac{-M_{ext,y} - M_{d,y} + b(F_{d,z} + F_{ext,z} + mg\cos\gamma) - h(-F_{ext,x} + F_{d,x} + mg\sin\gamma + m\ddot{x})}{N_F(a + b)}$$

$$F_{zR} = \frac{M_{ext,y} + M_{d,y} + a(F_{d,z} + F_{ext,z} + mg\cos\gamma) + h(-F_{ext,x} + F_{d,x} + mg\sin\gamma + m\ddot{x})}{N_R(a + b)}$$

The wheel normal forces satisfy this equation.

$$N_F F_{zF} + N_R F_{zR} - F_{ext,z} = mg\cos\gamma$$

Wind and Drag Forces

The block subtracts the wind speeds from the vehicle velocity components to obtain a net relative airspeed. To calculate the drag force and moments acting on the vehicle, the block uses the net relative airspeed.

$$F_{d,x} = \frac{1}{2TR}C_dA_fP_{abs}(\dot{x}$$

$$F_{d,z} = \frac{1}{2TR}C_lA_fP_{abs}(\dot{x}$$

$$M_{d,y} = \frac{1}{2TR}C_{pm}A_fP_{abs}(\dot{x}(a+b)$$

By default, to calculate the wind speed along the vehicle-fixed *x*-axis, the block uses the longitudinal wind speed along the earth-fixed *X*-axis. If you select **WindX,Y,Z**, the block uses the wind speed along the earth-fixed *X*-, *Y*-, *Z*-axes.

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrFxExt	Externally applied force power	$P_{FxExt} = F_{xExt}\dot{x}$
		PwrFwFx	Longitudinal force power applied at the front axle	$P_{FwFx} = F_{wF}\dot{x}$
		PwrFwRx	Longitudinal force power applied at the rear axle	$P_{FwRx} = F_{wR}\dot{x}$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrFxDrag	Drag force power	$P_d = -\dfrac{0.5C_dA_fP_{abs}\left(\dot{x}^2 - w_x\right)^2}{287.058T}\dot{x}$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	wrStoredGrvty	Rate change in gravitational potential energy	$P_g = -mg\dot{Z}$
		PwrStoredxdot	Rate in change of longitudinal kinetic energy	$P_{\dot{x}} = m\ddot{x}\dot{x}$

The equations use these variables.

F_{xf}, F_{xr}	Longitudinal forces on each wheel at the front and rear ground contact points, respectively

F_{zf}, F_{zr}	Normal load forces on each wheel at the front and rear ground contact points, respectively
F_{wF}, F_{wR}	Longitudinal force on front and rear axles along vehicle-fixed x-axis
F_{xExt}, F_{wR}	External force along the vehicle-fixed x-axis
$F_{d,x}$, $F_{d,z}$	Longitudinal and normal drag force on vehicle CG
$M_{d,y}$	Torque due to drag on vehicle about the vehicle-fixed y-axis
F_d	Aerodynamic drag force
V_x	Velocity of the vehicle. When $V_x > 0$, the vehicle moves forward. When $V_x < 0$, the vehicle moves backward.
N_f, N_r	Number of wheels on front and rear axle, respectively
γ	Angle of road grade
m	Vehicle body mass
a,b	Distance of front and rear axles, respectively, from the normal projection point of vehicle CG onto the common axle plane
h	Height of vehicle CG above the axle plane
C_d	Frontal air drag coefficient
A_f	Frontal area
P_{abs}	Absolute pressure
ρ	Mass density of air
x, $\dot{x}$, $\ddot{x}$	Vehicle longitudinal position, velocity, and acceleration along the vehicle-fixed x-axis
w_x	Wind speed along the vehicle-fixed x-axis
$\dot{Z}$	Vehicle vertical velocity along the vehicle-fixed z-axis

Limitations

The Vehicle Body 1DOF Longitudinal block lets you model only longitudinal dynamics, parallel to the ground and oriented along the direction of motion. The vehicle is assumed to be in pitch and normal equilibrium. The block does not model pitch or vertical movement. To model a vehicle with three degrees-of-freedom (DOF), use the Vehicle Body 3DOF Longitudinal.

Ports

Input

FExt — External force on vehicle CG
`array`

External forces applied to vehicle CG, F_{xext}, F_{yext}, F_{zext}, in vehicle-fixed frame, in N. Signal vector dimensions are `[1x3]` or `[3x1]`.

Dependencies

To enable this port, select **External forces**.

MExt — External moment about vehicle CG
array

External moment about vehicle CG, M_x, M_y, M_z, in the vehicle-fixed frame, in N·m. Signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, select **External moments**.

FwF — Total longitudinal force on front axle
scalar

Longitudinal force on the front axle, F_{xf}, along vehicle-fixed x-axis, in N.

FwR — Total longitudinal force on rear axle
scalar

Longitudinal force on the rear axle, Fw_R, along vehicle-fixed x-axis, in N.

Grade — Road grade angle
scalar

Road grade angle, γ, in deg.

WindX — Longitudinal wind speed
scalar

Longitudinal wind speed, W_w, along earth-fixed X-axis, in m/s.

Dependencies

To enable this port, clear **Wind X,Y,Z components**.

WindXYZ — Wind speed
array

Wind speed, W_w, W_{wY}, W_{wZ} along inertial X-, Y-, and Z-axes, in m/s. Signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, select **Wind X,Y,Z components**.

AirTemp — Ambient air temperature
scalar

Ambient air temperature, T_{air}, in K. Considering this option if you want to vary the temperature during run-time.

Dependencies

To enable this port, select **Air temperature**.

Output

Info — Bus signal
bus

Bus signal containing these block values.

Signal				Description	Value	Units
InertFrm	Cg	Disp	X	Vehicle CG displacement along earth-fixed X-axis	Computed	m
			Y	Vehicle CG displacement along earth-fixed Y-axis	0	m
			Z	Vehicle CG displacement along earth-fixed Z-axis	Computed	m
		Vel	Xdot	Vehicle CG velocity along earth-fixed X-axis	Computed	m/s
			Ydot	Vehicle CG velocity along earth-fixed Y-axis	0	m/s
			Zdot	Vehicle CG velocity along earth-fixed Z-axis	Computed	m/s
		Ang	phi	Rotation of vehicle-fixed frame about the earth-fixed X-axis (roll)	0	rad
			theta	Rotation of vehicle-fixed frame about the earth-fixed Y-axis (pitch)	Computed (input - grade angle)	rad
			psi	Rotation of vehicle-fixed frame about the earth-fixed Z-axis (yaw)	0	rad
	FrntAxl	Disp	X	Front axle displacement along the earth-fixed X-axis	Computed	m
			Y	Front axle displacement along the earth-fixed Y-axis	0	m
			Z	Front axle displacement along the earth-fixed Z-axis	Computed	m
		Vel	Xdot	Front axle velocity along the earth-fixed X-axis	Computed	m/s
			Ydot	Front axle velocity along the earth-fixed Y-axis	0	m/s
			Zdot	Front axle velocity along the earth-fixed Z-axis	Computed	m/s
	RearAxl	Disp	X	Rear axle displacement along the earth-fixed X-axis	Computed	m
			Y	Rear axle displacement along the earth-fixed Y-axis	0	m
			Z	Rear axle displacement along the earth-fixed Z-axis	Computed	m

Signal				Description	Value	Units
		Vel	Xdot	Rear axle velocity along the earth-fixed X-axis	Computed	m/s
			Ydot	Rear axle velocity along the earth-fixed Y-axis	0	m/s
			Zdot	Rear axle velocity along the earth-fixed Z-axis	Computed	m/s
BdyFrm	Cg	Disp	x	Vehicle CG displacement along the vehicle-fixed x-axis	Computed	m
			y	Vehicle CG displacement along the vehicle-fixed y-axis	0	m
			z	Vehicle CG displacement along the vehicle-fixed z-axis	0	m
		Vel	xdot	Vehicle CG velocity along the vehicle-fixed x-axis	Computed	m/s
			ydot	Vehicle CG velocity along the vehicle-fixed y-axis	0	m/s
			zdot	Vehicle CG velocity along the vehicle-fixed z-axis	0	m/s
		AngVel	p	Vehicle angular velocity about the vehicle-fixed x-axis (roll rate)	0	rad/s
			q	Vehicle angular velocity about the vehicle-fixed y-axis (pitch rate)	0	rad/s
			r	Vehicle angular velocity about the vehicle-fixed z-axis (yaw rate)	0	rad/s
		Accel	ax	Vehicle CG acceleration along the vehicle-fixed x-axis	Computed	gn
			ay	Vehicle CG acceleration along the vehicle-fixed y-axis	0	gn
			az	Vehicle CG acceleration along the vehicle-fixed z-axis	0	gn
	Forces	Body	Fx	Net force on vehicle CG along the vehicle-fixed x-axis	0	N

Signal				Description	Value	Units
			Fy	Net force on vehicle CG along the vehicle-fixed y-axis	0	N
			Fz	Net force on vehicle CG along the vehicle-fixed z-axis	0	N
		Ext	Fx	External force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy	External force on vehicle CG along the vehicle-fixed y-axis	Computed	N
			Fz	External force on vehicle CG along the vehicle-fixed z-axis	Computed	N
		FrntAxl	Fx	Longitudinal force on front axle, along the vehicle-fixed x-axis	0	N
			Fy	Lateral force on front axle, along the vehicle-fixed y-axis	0	N
			Fz	Normal force on front axle, along the vehicle-fixed z-axis	Computed	N
		RearAxl	Fx	Longitudinal force on rear axle, along the vehicle-fixed x-axis	0	N
			Fy	Lateral force on rear axle, along the vehicle-fixed y-axis	0	N
			Fz	Normal force on rear axle, along the vehicle-fixed z-axis	Computed	N
		Tires	FrntTire	Fx	Front tire force, along the vehicle-fixed x-axis	0	N
				Fy	Front tire force, along the vehicle-fixed y-axis	0	N
				Fz	Front tire force, along the vehicle-fixed z-axis	Computed	N
			RearTire	Fx	Rear tire force, along the vehicle-fixed x-axis	0	N
				Fy	Rear tire force, along the vehicle-fixed y-axis	0	N

Signal					Description	Value	Units
				Fz	Rear tire force, along the vehicle-fixed z-axis	Computed	N
		Drag	Fx		Drag force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy		Drag force on vehicle CG along the vehicle-fixed y-axis	Computed	N
			Fz		Drag force on vehicle CG along the vehicle-fixed z-axis	Computed	N
		Grvty	Fx		Gravity force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy		Gravity force on vehicle CG along the vehicle-fixed y-axis	0	N
			Fz		Gravity force on vehicle CG along the vehicle-fixed z-axis	Computed	N
	Moments	Body	Mx		Net moment on vehicle CG about the vehicle-fixed x-axis	0	N·m
			My		Net moment on vehicle CG about the vehicle-fixed y-axis	0	N·m
			Mz		Net moment on vehicle CG about the vehicle-fixed z-axis	0	N·m
		Drag	Mx		Drag moment on vehicle CG about the vehicle-fixed x-axis	Computed	N·m
			My		Drag moment on vehicle CG about the vehicle-fixed y-axis	Computed	N·m
			Mz		Drag moment on vehicle CG about the vehicle-fixed z-axis	Computed	N·m
		Ext	Fx		External moment on vehicle CG about the vehicle-fixed x-axis	Computed	N·m
			Fy		External moment on vehicle CG about the vehicle-fixed y-axis	Computed	N·m

Signal				Description	Value	Units
			Fz	External moment on vehicle CG about the vehicle-fixed z-axis	`Computed`	N·m
	FrntAxl	Disp	x	Front axle displacement along the vehicle-fixed x-axis	Computed	m
			y	Front axle displacement along the vehicle-fixed y-axis	0	m
			z	Front axle displacement along the vehicle-fixed z-axis	Computed	m
		Vel	xdot	Front axle velocity along the vehicle-fixed x-axis	Computed	m/s
			ydot	Front axle velocity along the vehicle-fixed y-axis	0	m/s
			zdot	Front axle velocity along the vehicle-fixed z-axis	Computed	m/s
		Steer	WhlAngFL	Front left wheel steering angle	Computed	rad
			WhlAngFR	Front right wheel steering angle	Computed	rad
	RearAxl	Disp	x	Rear axle displacement along the vehicle-fixed x-axis	Computed	m
			y	Rear axle displacement along the vehicle-fixed y-axis	0	m
			z	Rear axle displacement along the vehicle-fixed z-axis	Computed	m
		Vel	xdot	Rear axle velocity along the vehicle-fixed x-axis	Computed	m/s
			ydot	Rear axle velocity along the vehicle-fixed y-axis	0	m/s
			zdot	Rear axle velocity along the vehicle-fixed z-axis	Computed	m/s
		Steer	WhlAngRL	Rear left wheel steering angle	Computed	rad
			WhlAngRR	Rear right wheel steering angle	Computed	rad
	Pwr	PwrExt		Applied external power	Computed	W
		Drag		Power loss due to drag	Computed	W

Signal			Description	Value	Units
PwrInfo	PwrTrns frd	PwrFxExt	Externally applied force power	Computed	W
		PwrFwFx	Longitudinal force power applied at the front axle	Computed	W
		PwrFwRx	Longitudinal force power applied at the rear axle	Computed	W
	PwrNotT rnsfrd	PwrFxDrag	Drag force power	Computed	W
	PwrStor ed	wrStoredGrvty	Rate change in gravitational potential energy	Computed	W
		PwrStoredxdot	Rate in change of longitudinal kinetic energy	Computed	W

xdot — Vehicle body longitudinal velocity
scalar

Vehicle body longitudinal velocity along the earth-fixed reference frame X-axis, in m/s.

FzF — Front axle normal force
scalar

Normal load force on the front axle, F_{zf}, along vehicle-fixed z-axis, in N.

FzR — Rear axle normal force
scalar

Normal force on rear axle, F_{zr}, along the vehicle-fixed z-axis, in N.

Parameters

Options

External forces — FExt input port
off (default) | on

Specify to create input port FExt.

External moments — MExt input port
off (default) | on

Specify to create input port MExt.

Air temperature — AirTemp input port
off (default) | on

Specify to create input port AirTemp.

Wind X,Y,Z components — WindXYZ input port
off (default) | on

Specify to create input port WindXYZ.

Longitudinal

Number of wheels on front axle, NF — Front wheel count
2 (default) | scalar

Number of wheels on front axle, N_F. The value is dimensionless.

Number of wheels on rear axle, NR — Rear wheel count
2 (default) | scalar

Number of wheels on rear axle, N_R. The value is dimensionless.

Mass, m — Vehicle mass
1500 (default) | scalar

Vehicle mass, M, in kg.

Horizontal distance from CG to front axle, a — Front axle distance
1.4 (default) | scalar

Horizontal distance a from the vehicle CG to the front wheel axle, in m.

Horizontal distance from CG to rear axle, b — Rear axle distance
1.8 (default) | scalar

Horizontal distance b from the vehicle CG to the rear wheel axle, in m.

CG height above axles, h — Height
.35 (default) | scalar

Height of vehicle CG above the ground, h, in m.

Drag coefficient, Cd — Drag
.3 (default) | scalar

Air drag coefficient, C_d. The value is dimensionless.

Lift coefficient, Cl — Lift
0 (default) | scalar

Air lift coefficient, C_l. The value is dimensionless.

Pitch drag moment coefficient, Cpm — Pitch drag
0 (default) | scalar

Pitch drag moment coefficient, C_{pm}. The value is dimensionless.

Frontal area, Af — Area
4 (default) | scalar

Effective vehicle cross-sectional area, A, to calculate the aerodynamic drag force on the vehicle, in m^2.

Initial position, x_o — Position
0 (default) | scalar

Vehicle body longitudinal initial position along the vehicle-fixed x-axis, x_o, in m.

Initial velocity, xdot_o — Velocity
0 (default) | scalar

Vehicle body longitudinal initial velocity along the vehicle-fixed x-axis, $\dot{x}_0$, in m/s.

Environment

Absolute air pressure, Pabs — Pressure
101325 (default) | scalar

Environmental air absolute pressure, P_{abs}, in Pa.

Air temperature, T — Ambient air temperature
273 (default) | scalar

Ambient air temperature, T_{air}, in K.

Dependencies

To enable this parameter, clear **Air temperature**.

Gravitational acceleration, g — Gravity
9.81 (default) | scalar

Gravitational acceleration, g, in m/s^2.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Vehicle Body 3DOF Longitudinal | Vehicle Body Total Road Load

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2017a

Vehicle Body 3DOF Longitudinal

3DOF rigid vehicle body to calculate longitudinal, vertical, and pitch motion
Library: Powertrain Blockset / Vehicle Dynamics
                    Vehicle Dynamics Blockset / Vehicle Body
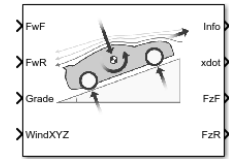
Description

The Vehicle Body 3DOF Longitudinal block implements a three degrees-of-freedom (3DOF) rigid vehicle body model with configurable axle stiffness to calculate longitudinal, vertical, and pitch motion. The block accounts for body mass, aerodynamic drag, road incline, and weight distribution between the axles due to acceleration and the road profile.

You can specify the type of axle attachment to the vehicle:

- Grade angle — Vertical axle displacement from road surface to axles remains constant. The block uses tabular stiffness and damping parameters to model the suspension forces acting between the vehicle body and axles.

- Axle displacement — Axles have input-provided vertical displacement and velocity with respect to the road grade. The block uses tabular stiffness and damping parameters to model the suspension forces acting between the vehicle body and axle.

- External suspension — Axles have externally applied forces for coupling the vehicle body to custom suspension models.

If the weight transfer from vertical and pitch motions are not negligible, consider using this block to represent vehicle motion in powertrain and fuel economy studies. For example, in studies with heavy breaking or acceleration or road profiles that contain larger vertical changes.

The block uses rigid-body vehicle motion, suspension system forces, and wind and drag forces to calculate the normal forces on the front and rear axles. The block resolves the force components and moments on the rigid vehicle body frame:

$$F_x = F_{wF} + F_{wR} - F_{d,x} - F_{sx,F} - F_{sx,R} + F_{g,x}$$
$$F_z = F_{d,z} - F_{sz,F} - F_{sz,R} + F_{g,z}$$
$$M_y = aF_{sz,F} - bF_{sz,R} + h(F_{wF} + F_{wR} + F_{sx,F} + F_{sx,R}) - M_{d,y}$$
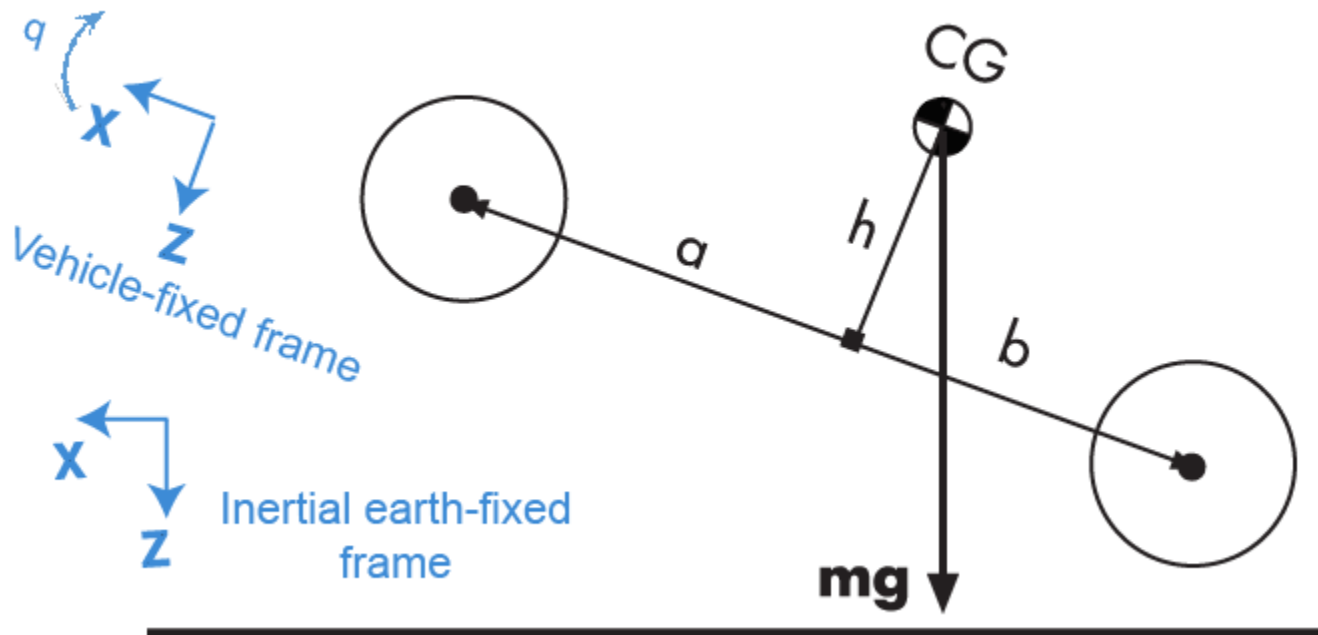
Rigid-Body Vehicle Motion

The vehicle axles are parallel and form a plane. The longitudinal direction lies in this plane and is perpendicular to the axles. If the vehicle is traveling on an inclined slope, the normal direction is not parallel to gravity but is always perpendicular to the axle-longitudinal plane.

The block uses the net effect of all the forces and torques acting on it to determine the vehicle motion. The longitudinal tire forces push the vehicle forward or backward. The weight of the vehicle acts through its center of gravity (CG). Depending on the inclined angle, the weight pulls the vehicle to the ground and either forward or backward. Whether the vehicle travels forward or backward, aerodynamic drag slows it down. For simplicity, the drag is assumed to act through the CG.

The Vehicle Body 3DOF Longitudinal implements these equations.

$$\ddot{x} = \frac{F_x}{m} - qz$$

$$\ddot{z} = \frac{F_z}{m} - qx$$

$$\dot{q} = \frac{M_y}{I_{yy}}$$

$$\dot{\theta} = q$$

Suspension System Forces

If you configure the block with the **Ground interaction type** parameter `Grade angle` or `Axle displacement, velocity`, the block uses nonlinear stiffness and damping parameters to model the suspension system.

The front and rear axle suspension forces are given by:

$$Fs_F = N_F[Fk_F + Fb_F]$$
$$Fs_R = N_R[Fk_R + Fb_R]$$

The block uses lookup tables to implement the front and rear suspension stiffness. To account for kinematic and material nonlinearities, including collisions with end-stops, the tables are functions of the stroke.

$$Fk_F = f(dZ_F)$$
$$Fk_R = f(dZ_R)$$

The block uses lookup tables to implement the front and rear suspension damping. To account for nonlinearities, compression, and rebound, the tables are functions of the stroke rate.

$$Fb_F = f(d\dot{Z}_F)$$
$$Fb_R = f(d\dot{Z}_R)$$

The stroke is the difference in the vehicle vertical and axle positions. The stroke rate is the difference in the vertical and axle velocities.

$$dZ_F = Z_F - \bar{Z}_F$$
$$dZ_R = Z_R - \bar{Z}_R$$
$$d\dot{Z}_F = \dot{Z}_F - \dot{\bar{Z}}_F$$
$$d\dot{Z}_R = \dot{Z}_R - \dot{\bar{Z}}_R$$

When the **Ground interaction type** parameter is `Grade angle`, the axle vertical positions ($\bar{Z}_F, \bar{Z}_R$) and velocities ($\dot{\bar{Z}}_F, \dot{\bar{Z}}_R$) are set to `0`.

Wind and Drag Forces

The block subtracts the wind speeds from the vehicle velocity components to obtain a net relative airspeed. To calculate the drag force and moments acting on the vehicle, the block uses the net relative airspeed:

$$F_{d,x} = \frac{1}{2TR}C_dA_fP_{abs}(^{\dot{x}}$$

$$F_{d,z} = \frac{1}{2TR}C_lA_fP_{abs}(^{\dot{x}}$$

$$M_{d,y} = \frac{1}{2TR}C_{pm}A_fP_{abs}(^{\dot{x}}(a + b)$$

Power Accounting

For the power accounting, the block implements these equations.

Bus Signal			Description	Equations
PwrInfo	PwrTrnsfrd — Power transferred between blocks • Positive signals indicate flow into block • Negative signals indicate flow out of block	PwrFxExt	Externally applied longitudinal force power	$P_{FxExt} = F_{xExt}\dot{x}$
		PwrFzExt	Externally applied longitudinal force power	$P_{FzExt} = F_{zExt}\dot{z}$
		PwrMyExt	Externally applied pitch moment power	$P_{MzExt} = M_{zExt}\dot{\theta}$
		PwrFwFx	Longitudinal force applied at the front axle	$P_{FwFx} = F_{wF}\dot{x}$
		PwrFwRx	Longitudinal force applied at the rear axle	$P_{FwRx} = F_{wR}\dot{x}$
	PwrNotTrnsfrd — Power crossing the block boundary, but not transferred • Positive signals indicate an input • Negative signals indicate a loss	PwrFsF	Internal power transferred between suspension and vehicle body at the front axle	$P_{Fs,F} = -P_{FwFx} + P_{FsbF} + P_{Fsk,F} + F_{xF}\dot{x}_F + F_{zF}\dot{z}_F$
		PwrFsR	Internal power transferred between suspension and vehicle body at the rear axle	$P_{Fs,R} = -P_{FwRx} + P_{Fsb,R} + P_{Fsk,R} + F_{xF}\dot{x}_F + F_{zF}\dot{z}_F$
		PwrFxDrag	Longitudinal drag force power	$P_{d,x} = F_{d,x}\dot{x}$
		PwrFzDrag	Vertical drag force power	$P_{d,z} = F_{d,z}\dot{z}$
		PwrMyDrag	Drag pitch moment power	$P_{d,My} = M_{d,y}\dot{\theta}$
		PwrFsb	Total suspension damping power	$P_{Fsb} = \sum_{i=F,R} F_{sb,i}\dot{z}_i$
	PwrStored — Stored energy rate of change • Positive signals indicate an increase • Negative signals indicate a decrease	PwrStoredGrvty	Rate change in gravitational potential energy	$P_g = -mg\dot{Z}$
		PwrStoredxdot	Rate of change of longitudinal kinetic energy	$P_{\dot{x}} = m\ddot{x}\dot{x}$
		PwrStoredzdot	Rate of change of longitudinal kinetic energy	$P_{\dot{z}} = m\ddot{z}\dot{z}$

Bus Signal		Description	Equations
	PwrStoredq	Rate of change of rotational pitch kinetic energy	$P_{\dot{\theta}} = I_{yy}\ddot{\theta}\dot{\theta}$
	PwrStoredFsFzSprng	Stored spring energy from front suspension	$P_{FskF} = F_{sk,F}\dot{z}_F$
	PwrStoredFsRzSprng	Stored spring energy from rear suspension	$P_{FskF} = F_{sk,R}\dot{z}_R$

The equations use these variables.

F_x	Longitudinal force on vehicle
F_z	Normal force on vehicle
M_y	Torque on vehicle about the vehicle-fixed y-axis
F_{wF}, F_{wR}	Longitudinal force on front and rear axles along vehicle-fixed x-axis
$F_{d,x}$, $F_{d,z}$	Longitudinal and normal drag force on vehicle CG
$F_{sx,F}$, $F_{sx,R}$	Longitudinal suspension force on front and rear axles
$F_{sz,F}$, $F_{sz,R}$	Normal suspension force on front and rear axles
$F_{g,x}$, $F_{g,z}$	Longitudinal and normal gravitational force on vehicle along the vehicle-fixed frame
$M_{d,y}$	Torque due to drag on vehicle about the vehicle-fixed y-axis
a, b	Distance of front and rear axles, respectively, from the normal projection point of vehicle CG onto the common axle plane
h	Height of vehicle CG above the axle plane along vehicle-fixed z-axis
Fs_F, Fs_R	Front and rear axle suspension force along vehicle-fixed z-axis
Z_{wF}, Z_{wR}	Front and rear vehicle normal position along earth-fixed z-axis
Θ	Vehicle pitch angle about the vehicle-fixed y-axis
m	Vehicle body mass
N_F, N_R	Number of front and rear wheels
I_{yy}	Vehicle body moment of inertia about the vehicle-fixed y-axis
$x, \dot{x}, \ddot{x}$	Vehicle longitudinal position, velocity, and acceleration along the vehicle-fixed x-axis
$z, \dot{z}, \ddot{z}$	Vehicle normal position, velocity, and acceleration along the vehicle-fixed z-axis
Fk_F, Fk_R	Front and rear wheel suspension stiffness force along vehicle-fixed z-axis
Fb_F, Fb_R	Front and rear wheel suspension damping force along vehicle-fixed z-axis
Z_F, Z_R	Front and rear vehicle vertical position along earth-fixed Z-axis
$\dot{Z}_F$, $\dot{Z}_R$	Front and rear vehicle vertical velocity along vehicle-fixed z-axis
$\bar{Z}_F$, $\bar{Z}_R$	Front and rear wheel axle vertical position along vehicle-fixed z-axis
$\dot{\bar{Z}}_F$, $\dot{\bar{Z}}_R$	Front and rear wheel axle vertical velocity along earth-fixed z-axis

dZ_F, dZ_R	Front and rear axle suspension deflection along vehicle-fixed z-axis
$d\dot{Z}_F, d\dot{Z}_R$	Front and rear axle suspension deflection rate along vehicle-fixed z-axis
C_d	Frontal air drag coefficient acting along the vehicle-fixed x-axis
C_l	Lateral air drag coefficient acting along the vehicle-fixed z-axis
C_{pm}	Air drag pitch moment acting about the vehicle-fixed y-axis
A_f	Frontal area
P_{abs}	Environmental absolute pressure
R	Atmospheric specific gas constant
T	Environmental air temperature
w_x	Wind speed along the vehicle-fixed x-axis

Ports

Input

FExt — External force on vehicle CG
`array`

External forces applied to vehicle CG, F_{xext}, F_{yext}, F_{zext}, in vehicle-fixed frame, in N. Signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, select **External forces**.

MExt — External moment about vehicle CG
`array`

External moment about vehicle CG, M_x, M_y, M_z, in the vehicle-fixed frame, in N·m. Signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, select **External moments**.

FwF — Total longitudinal force on the front axle
`scalar`

Longitudinal force on the front axle, Fw_F, along vehicle-fixed x-axis, in N.

FwR — Total longitudinal force on the rear axle
`scalar`

Longitudinal force on the rear axle, Fw_R, along vehicle-fixed x-axis, in N.

Grade — Road grade angle
`scalar`

Road grade angle, γ, in deg.

FsF — Suspension force on front axle per wheel
`vector`

Suspension force on front axle, Fs_F, along the vehicle-fixed z-axis, in N.

Dependencies

To enable this port, for the **Ground interaction type** parameter, select `External suspension`.

FsR — Suspension force on rear axle per wheel
vector

Suspension force on rear axle, Fs_R, along the vehicle-fixed z-axis, in N.

Dependencies

To enable this port, for the **Ground interaction type** parameter, select `External suspension`.

WindXYZ — Wind speed
array

Wind speed, W_X, W_Y, W_Z along earth-fixed X-, Y-, and Z-axes, in m/s. Signal vector dimensions are [1x3] or [3x1].

AirTemp — Ambient air temperature
scalar

Ambient air temperature, T_{air}, in K. Considering this option if you want to vary the temperature during run-time.

Dependencies

To enable this port, select **Air temperature**.

zF,R — Forward and rear axle positions
vector

Forward and rear axle positions along the vehicle-fixed z-axis, $\bar{Z}_F$, $\bar{Z}_R$, in m.

Dependencies

To enable this port, for the **Ground interaction type** parameter, select `Axle displacement, velocity`.

zdotF,R — Forward and rear axle velocities
vector

Forward and rear axle velocities along the vehicle-fixed z-axis, $\dot{\bar{Z}}_F$, $\dot{\bar{Z}}_R$, in m/s.

Dependencies

To enable this port, for the **Ground interaction type** parameter, select `Axle displacement, velocity`.

Output

Info — Bus signal
bus

Bus signal containing these block values.

Signal				Description	Value	Units
InertFrm	Cg	Disp	X	Vehicle CG displacement along earth-fixed *X*-axis	Computed	m
			Y	Vehicle CG displacement along earth-fixed *Y*-axis	0	m
			Z	Vehicle CG displacement along earth-fixed *Z*-axis	Computed	m
		Vel	Xdot	Vehicle CG velocity along earth-fixed *X*-axis	Computed	m/s
			Ydot	Vehicle CG velocity along earth-fixed *Y*-axis	0	m/s
			Zdot	Vehicle CG velocity along earth-fixed *Z*-axis	Computed	m/s
		Ang	phi	Rotation of vehicle-fixed frame about the earth-fixed *X*-axis (roll)	0	rad
			theta	Rotation of vehicle-fixed frame about the earth-fixed *Y*-axis (pitch)	Computed	rad
			psi	Rotation of vehicle-fixed frame about the earth-fixed *Z*-axis (yaw)	0	rad
	FrntAxl	Disp	X	Front axle displacement along the earth-fixed *X*-axis	Computed	m
			Y	Front axle displacement along the earth-fixed *Y*-axis	0	m
			Z	Front axle displacement along the earth-fixed *Z*-axis	Computed	m
		Vel	Xdot	Front axle velocity along the earth-fixed *X*-axis	Computed	m/s
			Ydot	Front axle velocity along the earth-fixed *Y*-axis	0	m/s
			Zdot	Front axle velocity along the earth-fixed *Z*-axis	Computed	m/s
	RearAxl	Disp	X	Rear axle displacement along the earth-fixed *X*-axis	Computed	m
			Y	Rear axle displacement along the earth-fixed *Y*-axis	0	m
			Z	Rear axle displacement along the earth-fixed *Z*-axis	Computed	m
		Vel	Xdot	Rear axle velocity along the earth-fixed *X*-axis	Computed	m/s
			Ydot	Rear axle velocity along the earth-fixed *Y*-axis	0	m/s

Signal				Description	Value	Units
			Zdot	Rear axle velocity along the earth-fixed *Z*-axis	Computed	m/s
BdyFrm	Cg	Disp	x	Vehicle CG displacement along the vehicle-fixed *x*-axis	Computed	m
			y	Vehicle CG displacement along the vehicle-fixed *y*-axis	0	m
			z	Vehicle CG displacement along the vehicle-fixed *z*-axis	Computed	m
		Vel	xdot	Vehicle CG velocity along the vehicle-fixed *x*-axis	Computed	m/s
			ydot	Vehicle CG velocity along the vehicle-fixed *y*-axis	0	m/s
			zdot	Vehicle CG velocity along the vehicle-fixed *z*-axis	Computed	m/s
		AngVel	p	Vehicle angular velocity about the vehicle-fixed *x*-axis (roll rate)	0	rad/s
			q	Vehicle angular velocity about the vehicle-fixed *y*-axis (pitch rate)	Computed	rad/s
			r	Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate)	0	rad/s
		Accel	ax	Vehicle CG acceleration along the vehicle-fixed *x*-axis	Computed	gn
			ay	Vehicle CG acceleration along the vehicle-fixed *y*-axis	0	gn
			az	Vehicle CG acceleration along the vehicle-fixed *z*-axis	Computed	gn
	Forces	Body	Fx	Net force on vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy	Net force on vehicle CG along the vehicle-fixed *y*-axis	0	N
			Fz	Net force on vehicle CG along the vehicle-fixed *z*-axis	Computed	N

Signal					Description	Value	Units
		Ext	Fx		External force on vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy		External force on vehicle CG along the vehicle-fixed *y*-axis	Computed	N
			Fz		External force on vehicle CG along the vehicle-fixed *z*-axis	Computed	N
		FrntAxl	Fx		Longitudinal force on front axle, along the vehicle-fixed *x*-axis	Computed	N
			Fy		Lateral force on front axle, along the vehicle-fixed *y*-axis	0	N
			Fz		Normal force on front axle, along the vehicle-fixed *z*-axis	Computed	N
		RearAxl	Fx		Longitudinal force on rear axle, along the vehicle-fixed *x*-axis	Computed	N
			Fy		Lateral force on rear axle, along the vehicle-fixed *y*-axis	0	N
			Fz		Normal force on rear axle, along the vehicle-fixed *z*-axis	Computed	N
		Tires	FrntTire	Fx	Front tire force, along the vehicle-fixed *x*-axis	0	N
				Fy	Front tire force, along the vehicle-fixed *y*-axis	0	N
				Fz	Front tire force, along the vehicle-fixed *z*-axis	Computed	N
			RearTire	Fx	Rear tire force, along the vehicle-fixed *x*-axis	0	N
				Fy	Rear tire force, along the vehicle-fixed *y*-axis	0	N
				Fz	Rear tire force, along the vehicle-fixed *z*-axis	Computed	N
		Drag	Fx		Drag force on vehicle CG along the vehicle-fixed *x*-axis	Computed	N

Signal				Description	Value	Units
			Fy	Drag force on vehicle CG along the vehicle-fixed *y*-axis	Computed	N
			Fz	Drag force on vehicle CG along the vehicle-fixed *z*-axis	Computed	N
		Grvty	Fx	Gravity force on vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy	Gravity force on vehicle CG along the vehicle-fixed *y*-axis	0	N
			Fz	Gravity force on vehicle CG along the vehicle-fixed *z*-axis	Computed	N
	Moments	Body	Mx	Body moment on vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My	Body moment on vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz	Body moment on vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Drag	Mx	Drag moment on vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My	Drag moment on vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz	Drag moment on vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Ext	Fx	External moment on vehicle CG about the vehicle-fixed *x*-axis	Computed	N·m
			Fy	External moment on vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Fz	External moment on vehicle CG about the vehicle-fixed *z*-axis	Computed	N·m
	FrntAxl	Disp	x	Front axle displacement along the vehicle-fixed *x*-axis	Computed	m

Signal				Description	Value	Units
			y	Front axle displacement along the vehicle-fixed *y*-axis	0	m
			z	Front axle displacement along the vehicle-fixed *z*-axis	Computed	m
		Vel	xdot	Front axle velocity along the vehicle-fixed *x*-axis	Computed	m/s
			ydot	Front axle velocity along the vehicle-fixed *y*-axis	0	m/s
			zdot	Front axle velocity along the vehicle-fixed *z*-axis	Computed	m/s
		Steer	WhlAngFL	Front left wheel steering angle	Computed	rad
			WhlAngFR	Front right wheel steering angle	Computed	rad
	RearAxl	Disp	x	Rear axle displacement along the vehicle-fixed *x*-axis	Computed	m
			y	Rear axle displacement along the vehicle-fixed *y*-axis	0	m
			z	Rear axle displacement along the vehicle-fixed *z*-axis	Computed	m
		Vel	xdot	Rear axle velocity along the vehicle-fixed *x*-axis	Computed	m/s
			ydot	Rear axle velocity along the vehicle-fixed *y*-axis	0	m/s
			zdot	Rear axle velocity along the vehicle-fixed *z*-axis	Computed	m/s
		Steer	WhlAngRL	Rear left wheel steering angle	Computed	rad
			WhlAngRR	Rear right wheel steering angle	Computed	rad
	Pwr	PwrExt		Applied external power	Computed	W
		Drag		Power loss due to drag	Computed	W
PwrInfo	PwrTrnsfrd	PwrFxExt		Externally applied longitudinal force power	Computed	W
		PwrFzExt		Externally applied longitudinal force power	Computed	W
		PwrMyExt		Externally applied pitch moment power	Computed	W

Signal			Description	Value	Units
		PwrFwFx	Longitudinal force applied at the front axle	Computed	W
		PwrFwRx	Longitudinal force applied at the rear axle	Computed	W
	PwrNotTrnsfrd	PwrFsF	Internal power transferred between suspension and vehicle body at the front axle	Computed	W
		PwrFsR	Internal power transferred between suspension and vehicle body at the rear axle	Computed	W
		PwrFxDrag	Longitudinal drag force power	Computed	W
		PwrFzDrag	Vertical drag force power	Computed	W
		PwrMyDrag	Drag pitch moment power	Computed	W
		PwrFsb	Total suspension damping power	Computed	W
	PwrStored	PwrStoredGrvty	Rate change in gravitational potential energy	Computed	W
		PwrStoredxdot	Rate of change of longitudinal kinetic energy	Computed	W
		PwrStoredzdot	Rate of change of longitudinal kinetic energy	Computed	W
		PwrStoredq	Rate of change of rotational pitch kinetic energy	Computed	W
		PwrStoredFsFzSprng	Stored spring energy from front suspension	Computed	W
		PwrStoredFsRzSprng	Stored spring energy from rear suspension	Computed	W

xdot — Vehicle longitudinal velocity
scalar

Vehicle CG velocity along the vehicle-fixed x-axis, in m/s.

FzF — Front axle normal force
scalar

Normal force on front axle, Fz_F, along the vehicle-fixed z-axis, in N.

FzR — Rear axle normal force
scalar

Normal force on rear axle, Fz_R, along the vehicle-fixed z-axis, in N.

Parameters

Options

External forces — FExt input port
off (default) | on

Specify to create input port FExt.

External moments — MExt input port
off (default) | on

Specify to create input port MExt.

Air temperature — AirTemp input port
off (default) | on

Specify to create input port AirTemp.

Longitudinal

Number of wheels on front axle, NF — Front wheel count
2 (default) | scalar

Number of wheels on front axle, N_F. The value is dimensionless.

Number of wheels on rear axle, NR — Rear wheel count
2 (default) | scalar

Number of wheels on rear axle, N_R. The value is dimensionless.

Mass, m — Vehicle mass
1200 (default) | scalar

Vehicle mass, m, in kg.

Horizontal distance from CG to front axle, a — Front axle distance
1.4 (default) | scalar

Horizontal distance a from the vehicle CG to the front wheel axle, in m.

Horizontal distance from CG to rear axle, b — Rear axle distance
1.8 (default) | scalar

Horizontal distance b from the vehicle CG to the rear wheel axle, in m.

CG height above axles, h — Height
0.35 (default) | scalar

Height of vehicle CG above the axles, h, in m.

Drag coefficient, Cd — Drag
.3 (default) | scalar

Air drag coefficient, C_d. The value is dimensionless.

Frontal area, Af — Area
2 (default) | scalar

Effective vehicle cross-sectional area, A_f to calculate the aerodynamic drag force on the vehicle, in m^2.

Initial position, x_o — Position
0 (default) | scalar

Vehicle body longitudinal initial position along earth-fixed x-axis, x_o, in m.

Initial velocity, xdot_o — Velocity
0 (default) | scalar

Vehicle body longitudinal initial velocity along earth-fixed x-axis, $\dot{x}_0$, in m/s.

Vertical

Lift coefficient, Cl — Lift
.1 (default) | scalar

Lift coefficient, C_l. The value is dimensionless.

Initial vertical position, z_o — Position
-.35 (default) | scalar

Initial vertical CG position, z_o, along the vehicle-fixed z-axis, in m.

Initial vertical velocity, zdot_o — Velocity
0 (default) | scalar

Initial vertical CG velocity, $zdot_o$, along the vehicle-fixed z-axis, in m.

Pitch

Inertia, Iyy — About body y-axis
3500 (default) | scalar

Vehicle body moment of inertia about body z-axis.

Pitch drag moment coefficient, Cpm — Drag coefficient
.1 (default) | scalar

Pitch drag moment coefficient. The value is dimensionless.

Initial pitch angle, theta_o — Pitch
0 (default) | scalar

Initial pitch angle about body z-axis, in rad.

Initial angular velocity, q_o — Pitch velocity
0 (default) | scalar

Initial vehicle body angular velocity about body z-axis, in rad/s.

Suspension

Front axle stiffness force data, FskF — Force
[-50, -1, 0, 2, 3, 52].*1.5e4 (default) | vector

Front axle stiffness force data, Fk_F, in N.

Dependencies

To enable this parameter, for the **Ground interaction type** parameter, select Grade angle or Axle displacement, velocity.

Front axle displacement data, dzsF — Displacement
[-5e-3, -1e-4, 0, .2, .2001, .2051] (default) | vector

Front axle displacement data, in m.

Dependencies

To enable this parameter, for the **Ground interaction type** parameter, select Grade angle or Axle displacement, velocity.

Front axle damping force data, FsbF — Damping force
[-10000 -100 -10 0 10 100 10000] (default) | vector

Front axle damping force, in N.

Dependencies

To enable this parameter, for the **Ground interaction type** parameter, select Grade angle or Axle displacement, velocity.

Front axle velocity data, dzdotsF — Velocity
[-10 -1 -.1 0 .1 1 10] (default) | vector

Front axle velocity data, in m/s.

Dependencies

To enable this parameter, for the **Ground interaction type** parameter, select Grade angle or Axle displacement, velocity.

Rear axle stiffness force data, FskR — Force
[-50, -1, 0, 2, 3, 52].*1e4 (default) | vector

Rear axle stiffness force data, in N.

Dependencies

To enable this parameter, for the **Ground interaction type** parameter, select Grade angle or Axle displacement, velocity.

Rear axle displacement data, dzsR — Displacement
[-5e-3, -1e-4, 0, .2, .2001, .2051] (default) | vector

Rear axle displacement data, in m.

Dependencies

To enable this parameter, for the **Ground interaction type** parameter, select `Grade angle` or `Axle displacement, velocity`.

Rear axle damping force data, FsbR — Damping force
`[-10000 -100 -10 0 10 100 10000]` (default) | `vector`

Rear axle damping force, in N.

Dependencies

To enable this parameter, for the **Ground interaction type** parameter, select `Grade angle` or `Axle displacement, velocity`.

Rear axle velocity data, dzdotsR — Velocity
`[-10 -1 -.1 0 .1 1 10]` (default) | `vector`

Rear axle velocity data, in m/s.

Dependencies

To enable this parameter, for the **Ground interaction type** parameter, select `Grade angle` or `Axle displacement, velocity`.

Environment

Absolute air pressure, Pabs — Pressure
`101325` (default) | `scalar`

Environmental air absolute pressure, P_{abs}, in Pa.

Air temperature, Tair — Ambient air temperature
`273` (default) | `scalar`

Ambient air temperature, T_{air}, in K.

Dependencies

To enable this parameter, clear **Air temperature**.

Gravitational acceleration, g — Gravity
`9.81` (default)

Gravitational acceleration, g, in m/s^2.

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers, 1992.

[2] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[3] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary*. ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Vehicle Body 1DOF Longitudinal | Vehicle Body Total Road Load

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2017a

Vehicle Body 3DOF

3DOF rigid vehicle body to calculate longitudinal, lateral, and yaw motion
Library: Vehicle Dynamics Blockset / Vehicle Body
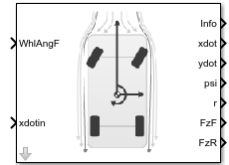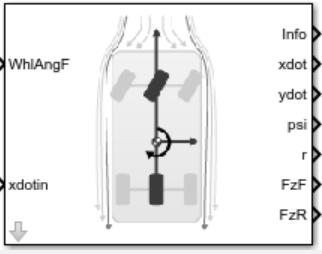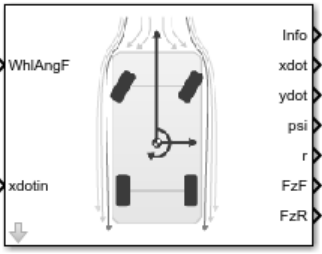
Description

The Vehicle Body 3DOF block implements a rigid two-axle vehicle body model to calculate longitudinal, lateral, and yaw motion. The block accounts for body mass and aerodynamic drag between the axles due to acceleration and steering.

Use this block in vehicle dynamics and automated driving studies to model nonholonomic vehicle motion when vehicle pitch, roll, and vertical motion are not significant.

In the Vehicle Dynamics Blockset™ library, there are two types of Vehicle Body 3DOF blocks that model longitudinal, lateral, and yaw motion.

Block	Vehicle Track Setting	Implementation
Vehicle Body 3DOF Single Track	Single (bicycle)	• Forces act along the center line at the front and rear axles. • No lateral load transfer.
Vehicle Body 3DOF Dual Track	Dual	Forces act at the four vehicle corners or *hard points*.

Use the **Axle forces** parameter to specify the type of force.

Axle Forces Setting	Implementation

`External longitudinal velocity`	• The block assumes that the external longitudinal velocity is in a quasi-steady state, so the longitudinal acceleration is approximately zero.
	
	• Because the motion is quasi-steady, the block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	
	• Consider this setting when you want to:
	
	   • Generate virtual sensor signal data.
	
	   • Conduct high-level software studies that are not impacted by driveline or nonlinear tire responses.

`External longitudinal forces`	• The block uses the external longitudinal force to accelerate or brake the vehicle.
	
	• The block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	
	• Consider this setting when you want to:
	
	   • Account for changes in the longitudinal velocity on the lateral and yaw motion.
	
	   • Specify the external longitudinal motion through a force instead of an external longitudinal velocity.
	
	   • Connect the block to tractive actuators, wheels, brakes, and hitches.

`External forces`	• The block uses the external lateral and longitudinal forces to steer, accelerate, or brake the vehicle. • The block does not use the steering input to calculate vehicle motion. • Consider this setting when you need tire models with more accurate nonlinear combined lateral and longitudinal slip.

You can use these block parameters to create additional input ports. This table summarizes the settings.

Input Signals Pane Parameter	Input Port	Description
Front wheel steering	`WhlAngF`	Front wheel angle, δ_F
External wind	`WindXYZ`	Wind speed, W_X, W_Y, W_Z, in the inertial reference frame
External forces	`FExt`	External force on vehicle center of gravity (CG), F_x, F_y, F_z, in the vehicle-fixed frame
Rear wheel steering	`WhlAngR`	Rear wheel angle, δ_R
External friction	`Mu`	Friction coefficient
External moments	`MExt`	External moment about vehicle CG, M_x, M_y, M_z, in vehicle-fixed frame
Initial longitudinal position	`X_o`	Initial vehicle CG displacement along the earth-fixed X-axis, in m

Input Signals Pane Parameter	Input Port	Description
Initial lateral position	`Y_o`	Initial vehicle CG displacement along the earth-fixed *Y*-axis, in m
Initial longitudinal velocity	`xdot_o`	Initial vehicle CG velocity along the vehicle-fixed *x*-axis, in m/s
Initial lateral velocity	`ydot_o`	Initial vehicle CG velocity along the vehicle-fixed *y*-axis, in m/s
Initial yaw angle	`psi_o`	Initial rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw), in rad
Initial yaw rate	`r_o`	Initial vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate), in rad/s
Air temperature	`AirTemp`	Ambient air temperature. Considering this option if you want to vary the temperature during run-time.

Theory

The Vehicle Body 3DOF block implements a rigid two-axle vehicle body model to calculate longitudinal, lateral, and yaw motion. The block accounts for body mass, aerodynamic drag, and weight distribution between the axles due to acceleration and steering. To determine the vehicle motion, the block implements these equations for the single track, dual track, and drag calculations.

Single Track

Calculation	Description
Dynamics	The block uses these equations to calculate the rigid body planar dynamics. $$\ddot{y} = -\dot{x}r + \frac{F_{yf} + F_{yr} + F_{yext}}{m}$$ $$\dot{r} = \frac{aF_{yf} - bF_{yr} + M_{zext}}{I_{zz}}$$ $$r = \dot{\psi}$$ If you set **Axle forces** to either `External longitudinal forces` or `External forces`, the block uses this equation for the longitudinal acceleration. $$\ddot{x} = \dot{y}r + \frac{F_{xf} + F_{xr} + F_{xext}}{m}$$ If you set **Axle forces** to `External longitudinal velocity`, the block assumes a quasi-steady state for the longitudinal acceleration. $$\ddot{x} = 0$$

Calculation	Description
External forces	External forces include both drag and external force inputs. The forces act on the vehicle CG. $$F_{x,y,z\,ext} = F_{d\,x,y,z} + F_{x,y,z\,input}$$ $$M_{x,y,z\,ext} = M_{d\,x,y,z} + M_{x,y,z\,input}$$ If you set **Axle forces** to `External longitudinal forces`, the block uses these equations. $$F_{xft} = F_{xfinput}$$ $$F_{yft} = -C_{yf}\alpha_f\mu_f\frac{F_{zf}}{F_{znom}}$$ $$F_{xrt} = F_{xrinput}$$ $$F_{yrt} = -C_{yr}\alpha_r\mu_r\frac{F_{zr}}{F_{znom}}$$ If you set **Axle forces** to `External longitudinal velocity`, the block uses these equations. $$F_{xft} = 0$$ $$F_{yft} = -C_{yf}\alpha_f\mu_f\frac{F_{zf}}{F_{znom}}$$ $$F_{xft} = 0$$ $$F_{yrt} = -C_{yr}\alpha_r\mu_r\frac{F_{zr}}{F_{znom}}$$ The block divides the normal forces by the nominal normal load to vary the effective friction parameters during weight and load transfer. The block uses these equations to maintain pitch and roll equilibrium. $$F_{zf} = \frac{bmg - (\ddot{x} - \dot{y}r)mh + hF_{xext} + bF_{zext} - M_{yext}}{a+b}$$ $$F_{zr} = \frac{amg + (\ddot{x} - \dot{y}r)mh - hF_{xext} + aF_{zext} + M_{yext}}{a+b}$$

Calculation	Description
Tire forces	The block uses the ratio of the local and longitudinal and lateral velocities to determine the slip angles. $$\alpha_f = atan\left(\frac{\dot{y} + ar}{\dot{x}}\right) - \delta_f$$ $$\alpha_r = atan\left(\frac{\dot{y} - br}{\dot{x}}\right) - \delta_r$$ To determine the tire forces, the block uses the slip angles. $$F_{xf} = F_{xft}\cos(\delta_f) - F_{yft}\sin(\delta_f)$$ $$F_{yf} = -F_{xft}\sin(\delta_f) + F_{yft}\cos(\delta_f)$$ $$F_{xr} = F_{xrt}\cos(\delta_r) - F_{yrt}\sin(\delta_r)$$ $$F_{yr} = -F_{xrt}\sin(\delta_r) + F_{yft}\cos(\delta_r)$$ If you set **Axle forces** to `External forces`, the block sets the tire forces equal to the external input force. $$F_{xf} = F_{xft} = F_{xfinput}$$ $$F_{yf} = F_{yft} = F_{yfinput}$$ $$F_{xr} = F_{xrt} = F_{xrinput}$$ $$F_{yr} = F_{yrt} = F_{yrinput}$$

Dual Track

Calculation	Description
Dynamics	The block uses these equations to calculate the rigid body planar dynamics. $$\ddot{x} = \dot{y}r + \frac{F_{xfl} + F_{xfr} + F_{xrl} + F_{xrr} + F_{xext}}{m}$$ $$\ddot{y} = -\dot{x}r + \frac{F_{yfl} + F_{yfr} + F_{yrl} + F_{yrr} + F_{yext}}{m}$$ $$\dot{r} = \frac{a(F_{yfl} + F_{yfr}) - b(F_{yrl} + F_{yrr}) + \frac{w_f(F_{xfl} - F_{xfr})}{2} + \frac{w_r(F_{xrl} - F_{xrr})}{2} + M_{zext}}{I_{zz}}$$ $$r = \dot{\psi}$$ If you set **Axle forces** to `External longitudinal velocity`, the block assumes a quasi-steady state for the longitudinal acceleration. $$\ddot{x} = 0$$

Calculation	Description
External forces	External forces include both drag and external force inputs. The forces act on the vehicle CG. $$F_{x, y, z\ ext} = F_{d\ x, y, z} + F_{x, y, z\ input}$$ $$M_{x, y, z\ ext} = M_{d\ x, y, z} + M_{x, y, z\ input}$$ If you set **Axle forces** to `External longitudinal forces`, the block uses these equations. $$F_{xflt} = F_{xflinput}$$ $$F_{yflt} = -C_{yfl}\alpha_{fl}\mu_{fl}\frac{F_{zfl}}{2F_{znom}}$$ $$F_{xfrt} = F_{xlrinput}$$ $$F_{yfrt} = -C_{yfr}\alpha_{fr}\mu_{fr}\frac{F_{zfr}}{2F_{znom}}$$ $$F_{xrlt} = F_{xrlinput}$$ $$F_{yrlt} = -C_{yrl}\alpha_{rl}\mu_{rl}\frac{F_{zrl}}{2F_{znom}}$$ $$F_{xrrt} = F_{xrrinput}$$ $$F_{yrrt} = -C_{yrr}\alpha_{rr}\mu_{rr}\frac{F_{zrr}}{2F_{znom}}$$ If you set **Axle forces** to `External longitudinal velocity`, the block uses these equations. $$F_{xflt} = 0$$ $$F_{yflt} = -C_{yfl}\alpha_{fl}\mu_{fl}\frac{F_{zfl}}{2F_{znom}}$$ $$F_{xfrt} = 0$$ $$F_{yfrt} = -C_{yfr}\alpha_{fr}\mu_{fr}\frac{F_{zfr}}{2F_{znom}}$$ $$F_{xrlt} = 0$$ $$F_{yrlt} = -C_{yrl}\alpha_{rl}\mu_{rl}\frac{F_{zrl}}{2F_{znom}}$$ $$F_{xrrt} = 0$$ $$F_{yrrt} = -C_{yrr}\alpha_{rr}\mu_{rr}\frac{F_{zrr}}{2F_{znom}}$$ The block divides the normal forces by the nominal normal load to vary the effective friction parameters during weight and load transfer. The block uses these equations to maintain pitch and roll equilibrium.

Calculation	Description
	$$F_{zf} = \frac{bmg - (\ddot{x} - \dot{y}r)mh + hF_{xext} + bF_{zext} - M_{yext}}{a + b}$$ $$F_{zr} = \frac{amg + (\ddot{x} - \dot{y}r)mh - hF_{xext} + aF_{zext} + M_{yext}}{(a + b)}$$ $$F_{zfl} = F_{zf} + (mh(\ddot{y} + \dot{x}r) - hF_{yext} - M_{xext})\frac{2}{w_f}$$ $$F_{zfr} = F_{zf} + (-mh(\ddot{y} + \dot{x}r) + hF_{yext} + M_{xext})\frac{2}{w_f}$$ $$F_{zrl} = F_{zr} + (mh(\ddot{y} + \dot{x}r) - hF_{yext} - M_{xext})\frac{2}{w_r}$$ $$F_{zrr} = F_{zr} + (-mh(\ddot{y} + \dot{x}r) + hF_{yext} + M_{xext})\frac{2}{w_r}$$
Tire forces	The block uses the ratio of the local and longitudinal and lateral velocities to determine the slip angles. $$\alpha_{fl} = atan\left(\frac{\dot{y} + ar}{\dot{x} + r\frac{w_f}{2}}\right) - \delta_{fl}$$ $$\alpha_{fr} = atan\left(\frac{\dot{y} + ar}{\dot{x} - r\frac{w_f}{2}}\right) - \delta_{fr}$$ $$\alpha_{rl} = atan\left(\frac{\dot{y} - ar}{\dot{x} + r\frac{w_r}{2}}\right) - \delta_{rl}$$ $$\alpha_{rr} = atan\left(\frac{\dot{y} - ar}{\dot{x} - r\frac{w_r}{2}}\right) - \delta_{rr}$$ The block uses the steering angles to transform the tire forces to the vehicle-fixed frame. $$F_{xf} = F_{xft}\cos(\delta_f) - F_{yft}\sin(\delta_f)$$ $$F_{yf} = -F_{xft}\sin(\delta_f) + F_{yft}\cos(\delta_f)$$ $$F_{xr} = F_{xrt}\cos(\delta_r) - F_{yrt}\sin(\delta_r)$$ $$F_{yr} = -F_{xrt}\sin(\delta_r) + F_{yft}\cos(\delta_r)$$ If you set **Axle forces** to `External forces`, the block uses these equations. The blocks assumes that the externally provided forces are in the vehicle-fixed frame at the axle-wheel location. $$F_{xf} = F_{xft} = F_{xfinput}$$ $$F_{yf} = F_{yft} = F_{yfinput}$$ $$F_{xr} = F_{xrt} = F_{xrinput}$$ $$F_{yr} = F_{yrt} = F_{yrinput}$$

Drag

Calculation	Description
Coordinate transformation	The block transforms the wind speeds from the inertial frame to the vehicle-fixed frame. $$w_x = W_x\cos(\psi) + W_y\sin(\psi)$$ $$w_y = W_y\cos(\psi) - W_x\sin(\psi)$$ $$w_z = W_z$$
Drag forces	To determine a relative airspeed, the block subtracts the wind speed from the CG vehicle velocity. Using the relative airspeed, the block determines the drag forces. $$\overline{w} = \sqrt{(\dot{x}_b - w_x)^2 + (\dot{x}_y - w_x)^2 + (w_z)^2}$$ $$F_{dx} = -\frac{1}{2TR}C_d A_f P_{abs}(\overline{w}$$ $$F_{dy} = -\frac{1}{2TR}C_s A_f P_{abs}(\overline{w}$$ $$F_{dz} = -\frac{1}{2TR}C_l A_f P_{abs}(\overline{w}$$
Drag moments	Using the relative airspeed, the block determines the drag moments. $$M_{dr} = -\frac{1}{2TR}C_{rm} A_f P_{abs}(\overline{w}(a+b)$$ $$M_{dp} = -\frac{1}{2TR}C_{pm} A_f P_{abs}(\overline{w}(a+b)$$ $$M_{dy} = -\frac{1}{2TR}C_{ym} A_f P_{abs}(\overline{w}(a+b)$$

Lateral Corner Stiffness and Relaxation Dynamics

Description	Implementation
Constant values.	The block uses constant stiffness values for Cy_f and Cy_r.
Lookup tables as a function of corner stiffness data and slip angles.	The block uses lookup tables that are functions of the corner stiffness data and slip angles. $$Cy_f = f(\alpha_f, Cy_{fdata})$$ $$Cy_r = f(\alpha_r, Cy_{rdata})$$

Description	Implementation
Lookup tables as a function of corner stiffness data and slip angles. *Slip angles include the relaxation length dynamic settings.*	The block uses lookup tables that are functions of the corner stiffness data and slip angles. The slip angles include the relaxation length dynamic settings. The relaxation length approximates an effective corner stiffness force that is a function of wheel travel. $$Cy_f = f(\alpha_{f\sigma}, Cy_{fdata})$$ $$Cy_r = f(\alpha_{r\sigma}, Cy_{rdata})$$ $$\alpha_{f\sigma} = \frac{1}{s}\left[\frac{(\alpha_f - \alpha_{f\sigma})v_{wf}}{\alpha_f}\right]$$ $$\alpha_{r\sigma} = \frac{1}{s}\left[\frac{(\alpha_r - \alpha_{r\sigma})v_{wr}}{\alpha_r}\right]$$

The equations use these variables.

$x, \dot{x}, \ddot{x}$	Vehicle CG displacement, velocity, and acceleration, along the vehicle-fixed *x*-axis
$y, \dot{y}, \ddot{y}$	Vehicle CG displacement, velocity, and acceleration, along the vehicle-fixed *y*-axis
ψ	Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw)
$r, \dot{\psi}$	Vehicle angular velocity, about the vehicle-fixed *z*-axis (yaw rate)
F_{xf}, F_{xr}	Longitudinal forces applied to front and rear wheels, along the vehicle-fixed *x*-axis
F_{yf}, F_{yr}	Lateral forces applied to front and rear wheels, along vehicle-fixed *y*-axis
$F_{xext}, F_{yext}, F_{zext}$	External forces applied to vehicle CG, along the vehicle-fixed *x*-, *y*-, and *z*-axes
F_{dx}, F_{dy}, F_{dz}	Drag forces applied to vehicle CG, along the vehicle-fixed *x*-, *y*-, and *z*-axes
$F_{xinput}, F_{yinput}, F_{zinput}$	Input forces applied to vehicle CG, along the vehicle-fixed *x*-, *y*-, and *z*-axes
$M_{xext}, M_{yext}, M_{zext}$	External moment about vehicle CG, about the vehicle-fixed *x*-, *y*-, and *z*-axes
M_{dx}, M_{dy}, M_{dz}	Drag moment about vehicle CG, about the vehicle-fixed *x*-, *y*-, and *z*-axes
$M_{xinput}, M_{yinput}, M_{zinput}$	Input moment about vehicle CG, about the vehicle-fixed *x*-, *y*-, and *z*-axes
I_{zz}	Vehicle body moment of inertia about the vehicle-fixed *z*-axis
F_{xft}, F_{xrt}	Longitudinal tire force applied to front and rear wheels, along the vehicle-fixed *x*-axis
F_{yft}, F_{yft}	Lateral tire force applied to front and rear wheels, along vehicle-fixed *y*-axis
F_{xfl}, F_{xfr}	Longitudinal force applied to front left and front right wheels, along the vehicle-fixed *x*-axis
F_{yfl}, F_{yfr}	Lateral force applied to front left and front right wheels, along the vehicle-fixed *y*-axis
F_{xrl}, F_{xrr}	Longitudinal force applied to rear left and rear right wheels, along the vehicle-fixed *x*-axis

F_{yrl}, F_{yrr}	Lateral force applied to rear left and rear right wheels, along the vehicle-fixed y-axis
F_{xflt}, F_{xfrt}	Longitudinal tire force applied to front left and front right wheels, along the vehicle-fixed x-axis
F_{yflt}, F_{yfrt}	Lateral force tire applied to front left and front right wheels, along the vehicle-fixed y-axis
F_{xrlt}, F_{xrrt}	Longitudinal tire force applied to rear left and rear right wheels, along the vehicle-fixed x-axis
F_{yrlt}, F_{yrrt}	Lateral force applied to rear left and rear right wheels, along the vehicle-fixed y-axis
F_{zf}, F_{zr}	Normal force applied to front and rear wheels, along vehicle-fixed z-axis
F_{znom}	Nominal normal force applied to axles, along the vehicle-fixed z-axis
F_{zfl}, F_{zfr}	Normal force applied to front left and right wheels, along vehicle-fixed z-axis
F_{zrl}, F_{zrr}	Normal force applied to rear left and right wheels, along vehicle-fixed z-axis
m	Vehicle body mass
a, b	Distance of front and rear wheels, respectively, from the normal projection point of vehicle CG onto the common axle plane
h	Height of vehicle CG above the axle plane
α_f, α_r	Front and rear wheel slip angles
α_{fl}, α_{fr}	Front left and right wheel slip angles
α_{rl}, α_{rr}	Rear left and right wheel slip angles
δ_f, δ_r	Front and rear wheel steering angles
δ_{rl}, δ_{rr}	Rear left and right wheel steering angles
δ_{fl}, δ_{fr}	Front left and right wheel steering angles
w_f, w_r	Front and rear track widths
Cy_f, Cy_r	Front and rear wheel cornering stiffness
Cy_{fdata}, Cy_{rdata}	Front and rear wheel cornering stiffness data
σ_f, σ_r	Front and rear wheel relaxation length
$\alpha_{f\sigma}$, $\alpha_{r\sigma}$	Front and rear wheel slip angles that include relaxation length
v_{wf}, v_{wr}	Magnitude of front and rear wheel hardpoint velocity
μ_f, μ_r	Front and rear wheel friction coefficient
μ_{fl}, μ_{fr}	Front left and right wheel friction coefficient
μ_{rl}, μ_{rr}	Rear left and right wheel friction coefficient
C_d	Air drag coefficient acting along vehicle-fixed x-axis
C_s	Air drag coefficient acting along vehicle-fixed y-axis
C_l	Air drag coefficient acting along vehicle-fixed z-axis
C_{rm}	Air drag roll moment acting about the vehicle-fixed x-axis
C_{pm}	Air drag pitch moment acting about the vehicle-fixed y-axis
C_{ym}	Air drag yaw moment acting about the vehicle-fixed z-axis

A_f	Frontal area
R	Atmospheric specific gas constant
T	Environmental air temperature
P_{abs}	Environmental absolute pressure
w_x, w_y, w_z	Wind speed, along the vehicle-fixed x-, y-, and z-axes
W_x, W_y, W_z	Wind speed, along inertial X-, Y-, and Z-axes

Ports

Input

WhlAngF — Front wheel steering angles
scalar | array

Front wheel steering angles, δ_F, in rad.

Vehicle Track Setting	Variable	Signal Dimension
Single (bicycle)	δ_F	Scalar – 1
Dual	$\delta_F = [\delta_{fl}\ \delta_{fr}]$ or $\begin{bmatrix} \delta_{fl} \\ \delta_{fr} \end{bmatrix}$	Array – [1x2] or [2x1]

Dependencies

To enable this port, on the **Input signals** pane, select **Front wheel steering**.

WhlAngR — Rear wheel steering angles
scalar | array

Rear wheel steering angles, δ_R, in rad.

Vehicle Track Setting	Variable	Signal Dimension
Single (bicycle)	δ_R	Scalar – 1
Dual	$\delta_R = [\delta_{rl}\ \delta_{rr}]$ or $\begin{bmatrix} \delta_{rl} \\ \delta_{rr} \end{bmatrix}$	Array – [1x2] or [2x1]

Dependencies

To enable this port, on the **Input signals** pane, select **Rear wheel steering**.

xdotin — Longitudinal velocity
scalar

Vehicle CG velocity along the vehicle-fixed x-axis, in m/s.

Dependencies

To enable this port, set **Axle forces** to External longitudinal velocity.

FwF — Total force on the front wheels
scalar | array

Force on the front wheels, Fw_F, along the vehicle-fixed axis, in N.

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Single (bicycle)	External longitudinal forces	Longitudinal force on the front wheel	$FwF = Fx_f$	Scalar – 1
	External forces	Longitudinal and lateral forces on the front wheel	$FwF = [Fx_f\ Fy_f]$ or $\begin{bmatrix} Fx_f \\ Fy_f \end{bmatrix}$	Array – [1x2] or [2x1]
Dual	External longitudinal forces	Longitudinal force on the front wheels	$FwF = [F_{xfl}\ F_{xfr}]$ or $\begin{bmatrix} F_{xfl} \\ F_{xfr} \end{bmatrix}$	Array – [1x2] or [2x1]
	External forces	Longitudinal and lateral forces on the front wheels	$FwF = \begin{bmatrix} F_{xfl}\ F_{xfr} \\ F_{yfl}\ F_{yfr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, set **Axle forces** to one of these options:

* External longitudinal forces
* External forces

FwR — Total force on the rear wheels
scalar | array

Force on the rear wheels, Fw_R, along the vehicle-fixed axis, in N.

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Single (bicycle)	External longitudinal forces	Longitudinal force on the rear wheel	$FwR = Fx_r$	Scalar – 1
	External forces	Longitudinal and lateral forces on the rear wheel	$FwR = [Fx_r\ Fy_r]$ or $\begin{bmatrix} Fx_r \\ Fy_r \end{bmatrix}$	Array – [1x2] or [2x1]
Dual	External longitudinal forces	Longitudinal force on the rear wheels	$FwR = [F_{xrl}\ F_{xrr}]$ or $\begin{bmatrix} F_{xrl} \\ F_{xrr} \end{bmatrix}$	Array – [1x2] or [2x1]
	External forces	Longitudinal and lateral forces on the rear wheels	$FwR = \begin{bmatrix} F_{xrl}\ F_{xrr} \\ F_{yrl}\ F_{yrr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, set **Axle forces** to one of these options:

- `External longitudinal forces`
- `External forces`

FExt — External force on vehicle CG
`array`

External forces applied to vehicle CG, F_{xext}, F_{yext}, F_{zext}, in vehicle-fixed frame, in N. Signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, on the **Input signals** pane, select **External forces**.

MExt — External moment about vehicle CG
`array`

External moment about vehicle CG, M_x, M_y, M_z, in the vehicle-fixed frame, in N·m. Signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, on the **Input signals** pane, select **External moments**.

WindXYZ — Wind speed
`array`

Wind speed, W_x, W_y, W_z along inertial X-, Y-, and Z-axes, in m/s. Signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, on the **Input signals** pane, select **External wind**.

Mu — Tire friction coefficient
`scalar`

Tire friction coefficient, μ. The value is dimensionless.

Vehicle Track Setting	Description	Variable	Signal Dimension
`Single (bicycle)`	Longitudinal force on the front wheel	$Mu = [\mu_f \ \mu_r]$ or $\begin{bmatrix} \mu_f \\ \mu_r \end{bmatrix}$	Array – [1x2] or [2x1]
`Dual`	Longitudinal force on the front wheels	$Mu = \begin{bmatrix} \mu_{fl} & \mu_{fr} \\ \mu_{rl} & \mu_{rr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, on the **Input signals** pane, select **External friction**.

AirTemp — Ambient air temperature
`scalar`

Ambient air temperature, in K.

Dependencies

To enable this port, on the **Input signals** pane, select **Air temperature**.

X_o — Initial longitudinal position
scalar

Initial vehicle CG displacement along the earth-fixed *X*-axis, in m.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial longitudinal position**.

Y_o — Initial lateral position
scalar

Initial vehicle CG displacement along the earth-fixed *Y*-axis, in m.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial lateral position**.

xdot_o — Initial longitudinal position
scalar

Initial vehicle CG velocity along the vehicle-fixed *x*-axis, in m/s.

Dependencies

To enable this port:

1   Set **Axle forces** to one of these options:

   • External longitudinal forces
   • External forces

2   On the **Input signals** pane, select **Initial longitudinal velocity**

ydot_o — Initial lateral position
scalar

Initial vehicle CG velocity along the vehicle-fixed *y*-axis, in m/s.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial lateral velocity**.

psi_o — Initial yaw angle
scalar

Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw), in rad.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial yaw angle**.

r_o — Initial yaw rate
scalar

Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate), in rad/s.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial yaw rate**.

Output

Info — Bus signal
bus

Bus signal containing these block values.

Signal				Description	Value	Units
InertFrm	Cg	Disp	X	Vehicle CG displacement along the earth-fixed *X*-axis	Computed	m
			Y	Vehicle CG displacement along the earth-fixed *Y*-axis	Computed	m
			Z	Vehicle CG displacement along the earth-fixed *Z*-axis	0	m
		Vel	Xdot	Vehicle CG velocity along the earth-fixed *X*-axis	Computed	m/s
			Ydot	Vehicle CG velocity along the earth-fixed *Y*-axis	Computed	m/s
			Zdot	Vehicle CG velocity along the earth-fixed *Z*-axis	0	m/s
		Ang	phi	Rotation of the vehicle-fixed frame about the earth-fixed *X*-axis (roll)	0	rad
			theta	Rotation of the vehicle-fixed frame about the earth-fixed *Y*-axis (pitch)	0	rad
			psi	Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw)	Computed	rad
	FrntAxl	Lft	Disp	X	Front left wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Front left wheel displacement along the earth-fixed *Y*-axis	Computed	m

Signal					Description	Value	Units
				Z	Front left wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Front left wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Front left wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Front left wheel velocity along the earth-fixed *Z*-axis	0	m/s
		Rght	Disp	X	Front right wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Front right wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Front right wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Front right wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Front right wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Front right wheel velocity along the earth-fixed *Z*-axis	0	m/s
	RearAxl	Lft	Disp	X	Rear left wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Rear left wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Rear left wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Rear left wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Rear left wheel velocity along the earth-fixed *Y*-axis	Computed	m/s

Signal					Description	Value	Units
				Zdot	Rear left wheel velocity along the earth-fixed *Z*-axis	0	m/s
		Rght	Disp	X	Rear right wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Rear right wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Rear right wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Rear right wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Rear right wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Rear right wheel velocity along the earth-fixed *Z*-axis	0	m/s
		Geom	Disp	X	Vehicle chassis offset from axle plane along the earth-fixed *X*-axis	Computed	m
				Y	Vehicle chassis offset from center plane along the earth-fixed *Y*-axis	Computed	m
				Z	Vehicle chassis offset from axle plane along the earth-fixed *Z*-axis	Computed	m
			Vel	Xdot	Vehicle chassis offset velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Vehicle chassis offset velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Vehicle chassis offset velocity along the earth-fixed *Z*-axis	Computed	m/s
BdyFrm	Cg	Vel	xdot		Vehicle CG velocity along the vehicle-fixed *x*-axis	Computed	m/s
			ydot		Vehicle CG velocity along the vehicle-fixed *y*-axis	Computed	m/s

Signal				Description	Value	Units
			zdot	Vehicle CG velocity along the vehicle-fixed z-axis	0	m/s
		Ang	Beta	Body slip angle, β $$\beta = \frac{V_y}{V_x}$$	Computed	rad
		AngVel	p	Vehicle angular velocity about the vehicle-fixed x-axis (roll rate)	0	rad/s
			q	Vehicle angular velocity about the vehicle-fixed y-axis (pitch rate)	0	rad/s
			r	Vehicle angular velocity about the vehicle-fixed z-axis (yaw rate)	Computed	rad/s
		Acc	ax	Vehicle CG acceleration along the vehicle-fixed x-axis	Computed	gn
			ay	Vehicle CG acceleration along the vehicle-fixed y-axis	Computed	gn
			az	Vehicle CG acceleration along the vehicle-fixed z-axis	0	gn
			xddot	Vehicle CG acceleration along the vehicle-fixed x-axis	Computed	m/s^2
			yddot	Vehicle CG acceleration along the vehicle-fixed y-axis	Computed	m/s^2
			zddot	Vehicle CG acceleration along the vehicle-fixed z-axis	0	m/s^2
		AngAcc	pdot	Vehicle angular acceleration about the vehicle-fixed x-axis	0	rad/s
			qdot	Vehicle angular acceleration about the vehicle-fixed y-axis	0	rad/s
			rdot	Vehicle angular acceleration about the vehicle-fixed z-axis	Computed	rad/s
	Forces	Body	Fx	Net force on vehicle CG along the vehicle-fixed x-axis	Computed	N

Signal				Description	Value	Units
			Fy	Net force on vehicle CG along the vehicle-fixed y-axis	Computed	N
			Fz	Net force on vehicle CG along the vehicle-fixed z-axis	0	N
		Ext	Fx	External force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy	External force on vehicle CG along the vehicle-fixed y-axis	Computed	N
			Fz	External force on vehicle CG along the vehicle-fixed z-axis	0	N
		FrntAxl	Lft	Fx	Longitudinal force on left front wheel, along the vehicle-fixed x-axis	Computed	N
				Fy	Lateral force on left front wheel along the vehicle-fixed y-axis	Computed	N
				Fz	Normal force on left front wheel, along the vehicle-fixed z-axis	Computed	N
			Rght	Fx	Longitudinal force on right front wheel, along the vehicle-fixed x-axis	Computed	N
				Fy	Lateral force on right front wheel along the vehicle-fixed y-axis	Computed	N
				Fz	Normal force on right front wheel, along the vehicle-fixed z-axis	Computed	N
		RearAxl	Lft	Fx	Longitudinal force on left rear wheel, along the vehicle-fixed x-axis	Computed	N
				Fy	Lateral force on left rear wheel along the vehicle-fixed y-axis	Computed	N
				Fz	Normal force on left rear wheel, along the vehicle-fixed z-axis	Computed	N
			Rght	Fx	Longitudinal force on right rear wheel, along the vehicle-fixed x-axis	Computed	N

Signal						Description	Value	Units
				Fy		Lateral force on right rear wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz		Normal force on right rear wheel, along the vehicle-fixed *z*-axis	Computed	N
		Tires	FrntTires	Lft	Fx	Front left tire force, along the vehicle-fixed *x*-axis	Computed	N
					Fy	Front left tire force, along the vehicle-fixed *y*-axis	Computed	N
					Fz	Front left tire force, along the vehicle-fixed *z*-axis	Computed	N
				Rght	Fx	Front right tire force, along the vehicle-fixed *x*-axis	Computed	N
					Fy	Front right tire force, along the vehicle-fixed *y*-axis	Computed	N
					Fz	Front right tire force, along the vehicle-fixed *z*-axis	Computed	N
			RearTires	Lft	Fx	Rear left tire force, along the vehicle-fixed *x*-axis	Computed	N
					Fy	Rear left tire force, along the vehicle-fixed *y*-axis	Computed	N
					Fz	Rear left tire force, along the vehicle-fixed *z*-axis	Computed	N
				Rght	Fx	Rear right tire force, along the vehicle-fixed *x*-axis	Computed	N
					Fy	Rear right tire force, along the vehicle-fixed *y*-axis	Computed	N
					Fz	Rear right tire force, along the vehicle-fixed *z*-axis	Computed	
		Drag	Fx			Drag force on vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy			Drag force on vehicle CG along the vehicle-fixed *y*-axis	Computed	N

Signal					Description	Value	Units
			Fz		Drag force on vehicle CG along the vehicle-fixed *z*-axis	Computed	N
		Grvty	Fx		Gravity force on vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy		Gravity force on vehicle CG along the vehicle-fixed *y*-axis	Computed	N
			Fz		Gravity force on vehicle CG along the vehicle-fixed *z*-axis	Computed	N
	Moments	Body	Mx		Body moment on vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My		Body moment on vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz		Body moment on vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Drag	Mx		Drag moment on vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My		Drag moment on vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz		Drag moment on vehicle CG about the vehicle-fixed *z*-axis	0	N·m
	FrntAxl	Lft	Disp	x	Front left wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Front left wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Front left wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Front left wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Front left wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s

Signal					Description	Value	Units
				zdot	Front left wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Rght	Disp	x	Front right wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Front right wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Front right wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Front right wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Front right wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Front right wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Steer	WhlAngFL		Front left wheel steering angle	Computed	rad
			WhlAngFR		Front right wheel steering angle	Computed	rad
	RearAxl	Lft	Disp	x	Rear left wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Rear left wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Rear left wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Rear left wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Rear left wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Rear left wheel velocity along the vehicle-fixed *z*-axis	0	m/s

Signal					Description	Value	Units
		Rght	Disp	x	Rear right wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Rear right wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Rear right wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Rear right wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Rear right wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Rear right wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Steer	WhlAngRL		Rear left wheel steering angle	Computed	rad
			WhlAngRR		Rear right wheel steering angle	Computed	rad
	Pwr	PwrExt			Applied external power	Computed	W
		Drag			Power loss due to drag	Computed	W
	Geom	Disp		x	Vehicle chassis offset from axle plane along the vehicle-fixed *x*-axis	Input	m
				y	Vehicle chassis offset from center plane along the vehicle-fixed *y*-axis	Input	m
				z	Vehicle chassis offset from axle plane along the earth-fixed *z*-axis	Input	m
		Vel		xdot	Vehicle chassis offset velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Vehicle chassis offset velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Vehicle chassis offset velocity along the vehicle-fixed *z*-axis	0	m/s

Signal				Description	Value	Units
		Beta	Bet a	Body slip angle, β $$\beta = \frac{V_y}{V_x}$$	Computed	rad

xdot — Vehicle longitudinal velocity
scalar

Vehicle CG velocity along the vehicle-fixed x-axis, in m/s.

ydot — Vehicle lateral velocity
scalar

Vehicle CG velocity along the vehicle-fixed y-axis, in m/s.

psi — Yaw
scalar

Rotation of the vehicle-fixed frame about the earth-fixed Z-axis (yaw), in rad.

r — Yaw rate
scalar

Vehicle angular velocity, r, about the vehicle-fixed z-axis (yaw rate), in rad/s.

FzF — Normal force on front wheels
scalar | array

Normal force on front wheels, Fz_F, along the vehicle-fixed z-axis, in N.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single (bicycle)	Normal force on front axle	$FzF = Fz_f$	Scalar – 1
Dual	Normal force on the front wheels	$FzF = [Fz_{fl}\ Fz_{fr}]$	Array – [1x2]

FzR — Normal force on rear wheels
scalar | array

Normal force on rear wheels, Fz_R, along the vehicle-fixed z-axis, in N.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single (bicycle)	Normal force on rear wheel	$FzR = Fz_r$	Scalar – 1
Dual	Normal force on the rear wheels	$FzR = [Fz_{rl}\ Fz_{rr}]$	Array – [1x2]
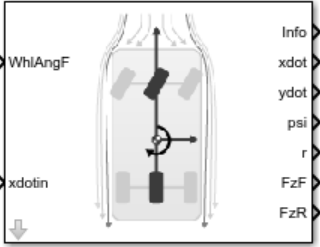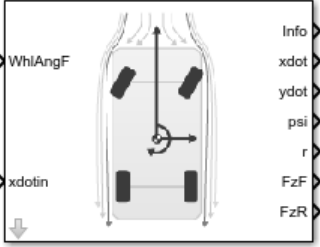
Parameters

Options

Vehicle track — Number of tracks
`Single (bicycle)|Dual`

In the Vehicle Dynamics Blockset library, there are two types of Vehicle Body 3DOF blocks that model longitudinal, lateral, and yaw motion.

Block	Vehicle Track Setting	Implementation
Vehicle Body 3DOF Single Track	`Single (bicycle)`	• Forces act along the center line at the front and rear axles. • No lateral load transfer.
Vehicle Body 3DOF Dual Track	`Dual`	Forces act at the four vehicle corners or *hard points*.

Axle forces — Type of axle force
`External longitudinal velocity|External longitudinal forces|External forces`

Use the **Axle forces** parameter to specify the type of force.

Axle Forces Setting	Implementation

`External longitudinal velocity`	• The block assumes that the external longitudinal velocity is in a quasi-steady state, so the longitudinal acceleration is approximately zero.
	• Because the motion is quasi-steady, the block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	• Consider this setting when you want to:
	  • Generate virtual sensor signal data.
	  • Conduct high-level software studies that are not impacted by driveline or nonlinear tire responses.

Axle Forces Setting	Implementation

`External longitudinal forces`	• The block uses the external longitudinal force to accelerate or brake the vehicle.
	• The block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	• Consider this setting when you want to:
	
	  • Account for changes in the longitudinal velocity on the lateral and yaw motion.
	  • Specify the external longitudinal motion through a force instead of an external longitudinal velocity.
	  • Connect the block to tractive actuators, wheels, brakes, and hitches.

`External forces`	• The block uses the external lateral and longitudinal forces to steer, accelerate, or brake the vehicle. • The block does not use the steering input to calculate vehicle motion. • Consider this setting when you need tire models with more accurate nonlinear combined lateral and longitudinal slip.

Input Signals

Front wheel steering — WhlAngF input port
on (default) | off

Specify to create input port WhlAngF.

External wind — WindXYZ input port
off (default) | on

Specify to create input port WindXYZ.

External forces — FExt input port
off (default) | on

Specify to create input port FExt.

External moments — MExt input port
off (default) | on

Specify to create input port MExt.

Rear wheel steering — WhlAngR input port
off (default) | on

Specify to create input port WhlAngR.

External friction — Mu input port
off (default) | on

Specify to create input port Mu.

Initial longitudinal position — X_o input port
off (default) | on

Specify to create input port X_o.

Initial lateral position — Y_o input port
off (default) | on

Specify to create input port Y_o.

Initial longitudinal velocity — xdot_o input port
off (default) | on

Specify to create input port xdot_o.

Initial lateral velocity — ydot_o input port
off (default) | on

Specify to create input port ydot_o.

Initial yaw angle — psi_o input port
off (default) | on

Specify to create input port psi_o.

Initial yaw rate — r_o input port
off (default) | on

Specify to create input port r_o.

Air temperature — AirTemp input port
off (default) | on

Specify to create input port AirTemp.

Longitudinal

Number of wheels on front axle, NF — Front wheel count
2 (default) | scalar

Number of wheels on front axle, N_F. The value is dimensionless.

Number of wheels on rear axle, NR — Rear wheel count
2 (default) | scalar

Number of wheels on rear axle, N_R. The value is dimensionless.

Vehicle mass, m — Vehicle mass
2000 (default) | scalar

Vehicle mass, m, in kg.

Longitudinal distance from center of mass to front axle, a — Front axle distance
1.4 (default) | scalar

Horizontal distance a from the vehicle CG to the front wheel axle, in m.

Longitudinal distance from center of mass to rear axle, b — Rear axle distance
1.6 (default) | scalar

Horizontal distance *b* from the vehicle CG to the rear wheel axle, in m.

Vertical distance from center of mass to axle plane, h — Height
0.35 (default) | scalar

Height of vehicle CG above the axles, *h*, in m.

Initial inertial frame longitudinal position, X_o — Position
0 (default) | scalar

Initial vehicle CG displacement along earth-fixed *X*-axis, in m.

Initial longitudinal velocity, xdot_o — Velocity
0 (default) | scalar

Initial vehicle CG velocity along vehicle-fixed *x*-axis, in m/s.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter, set **Axle forces** to one of these options:

- External longitudinal forces
- External forces

Lateral

Front tire corner stiffness, Cy_f — Stiffness
12e3 (default) | scalar

Front tire corner stiffness, Cy_f, in N/rad.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Clear **Mapped corner stiffness**.

Rear tire corner stiffness, Cy_r — Stiffness
11e3 (default) | scalar

Rear tire corner stiffness, Cy_r, in N/rad.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter:

1 Set **Axle forces** to one of these options:

- `External longitudinal velocity`
- `External longitudinal forces`

2 Clear **Mapped corner stiffness**.

Initial inertial frame lateral displacement, Y_o — Position
`0` (default) | `scalar`

Initial vehicle CG displacement along earth-fixed *Y*-axis, in m.

Initial lateral velocity, ydot_o — Velocity
`0` (default) | `scalar`

Initial vehicle CG velocity along vehicle-fixed *y*-axis, in m/s.

Mapped corner stiffness — Selection
`off` (default) | `on`

Enables mapped corner stiffness calculation.

Dependencies

To enable this parameter, set **Axle forces** to one of these options:

- `External longitudinal velocity`
- `External longitudinal forces`

Include relaxation length dynamics — Enable relaxation length dynamics
`on` (default) | `off`

Enables relaxation length dynamics.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- `External longitudinal velocity`
- `External longitudinal forces`

2 Clear **Mapped corner stiffness**.

Lateral distance from geometric centerline to center of mass, d — Distance
`0` (default) | `scalar`

Lateral distance from geometric centerline to center of mass, *d*, in m, along the vehicle-fixed *y*. Positive values indicate that the vehicle CM is to the right of the geometric centerline. Negative values indicate that the vehicle CM is to the left of the geometric centerline.

Track width — Width
`[1.4,1.4]` (default) | 1-by-2 `vector`

Track width, *w*, in m.

Dependencies

To enable this parameter, set **Vehicle track** to Dual.

Front tire(s) relaxation length, sigma_f — Relaxation length
.1 (default) | scalar

Front tire relaxation length, σ_f, in m.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Do either of these:

- Select **Mapped corner stiffness**.
- Clear **Mapped corner stiffness** and select **Include relaxation length dynamics**.

Rear tire(s) relaxation length, sigma_r — Relaxation length
.1 (default) | scalar

Rear tire relaxation length, σ_r, in m.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Do either of these:

- Select **Mapped corner stiffness**.
- Clear **Mapped corner stiffness** and select **Include relaxation length dynamics**.

Front axle slip angle breakpoints, alpha_f_brk — Breakpoints
[-.1 .1] (default) | vector

Front axle slip angle breakpoints, α_{fbrk}, in rad.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Select **Mapped corner stiffness**.

Front axle corner data, Cy_f_data — Breakpoints
[-9e3 9e3] (default) | vector

Front axle corner data, Cy_{fdata}, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Select **Mapped corner stiffness**.

Rear axle slip angle breakpoints, alpha_r_brk — Breakpoints
[-.1 .1] (default) | vector

Rear axle slip angle breakpoints, α_{rbrk}, in rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Select **Mapped corner stiffness**.

Rear axle corner data, Cy_r_data — Data
[-9e3 9e3] (default) | vector

Rear axle corner data, Cy_{rdata}, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Select **Mapped corner stiffness**.

Yaw

Yaw polar inertia, Izz — Inertia
4000 (default) | scalar

Yaw polar inertia, in kg*m^2.

Initial yaw angle, psi_o — Psi rotation
0 (default) | scalar

Rotation of the vehicle-fixed frame about earth-fixed *Z*-axis (yaw), in rad.

`Initial yaw rate, r_o` — Yaw rate
0 (default) | `scalar`

Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate), in rad/s.

Aerodynamic

`Longitudinal drag area, Af` — Effective vehicle cross-sectional area
2 (default) | `scalar`

Effective vehicle cross-sectional area, A_f, to calculate the aerodynamic drag force on the vehicle, in m^2.

`Longitudinal drag coefficient, Cd` — Air drag coefficient
.3 (default) | `scalar`

Air drag coefficient, C_d. The value is dimensionless.

`Longitudinal lift coefficient, Cl` — Air lift coefficient
.1 (default) | `scalar`

Air lift coefficient, C_l. The value is dimensionless.

`Longitudinal drag pitch moment, Cpm` — Pitch drag
.1 (default) | `scalar`

Longitudinal drag pitch moment coefficient, C_{pm}. The value is dimensionless.

`Relative wind angle vector, beta_w` — Wind angle
[0:0.01:0.3] (default) | `vector`

Relative wind angle vector, β_w, in rad.

`Side force coefficient vector, Cs` — Side force coefficient
[0:0.03:0.9] (default) | `vector`

Side force coefficient vector coefficient, C_s. The value is dimensionless.

`Yaw moment coefficient vector, Cym` — Yaw moment drag
[0:0.01:0.3] (default) | `vector`

Yaw moment coefficient vector coefficient, C_{ym}. The value is dimensionless.

Environment

`Absolute air pressure, Pabs` — Pressure
101325 (default) | `scalar` | `scalar`

Environmental absolute pressure, P_{abs}, in Pa.

`Air temperature, Tair` — Temperature
273 (default) | `scalar`

Environmental absolute temperature, *T*, in K.

Dependencies

To enable this parameter, clear **Air temperature**.

Gravitational acceleration, g — Gravity
9.81 (default) | scalar

Gravitational acceleration, g, in m/s^2.

Nominal friction scaling factor, mu — Friction scale factor
1 (default) | scalar

Nominal friction scale factor, μ. The value is dimensionless.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Clear **External Friction**.

Simulation

Longitudinal velocity tolerance, xdot_tol — Tolerance
.01 (default) | scalar

Longitudinal velocity tolerance, in m/s.

Nominal normal force, Fznom — Normal force
5000 (default) | scalar

Nominal normal force, in N.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter, set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

Geometric longitudinal offset from axle plane, longOff — Longitudinal offset
0 (default) | scalar

Vehicle chassis offset from axle plane along body-fixed *x*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

Geometric lateral offset from center plane, latOff — Lateral offset
0 (default) | scalar

Vehicle chassis offset from center plane along body-fixed *y*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

`Geometric vertical offset from axle plane, vertOff` **— Vertical offset**
0 (default) | `scalar`

Vehicle chassis offset from axle plane along body-fixed *z*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

`Wrap Euler angles, wrapAng` **— Selection**
off (default) | on

Wrap the Euler angles to the interval `[-pi, pi]`. For vehicle maneuvers that might undergo vehicle yaw rotations that are outside of the interval, consider deselecting the parameter if you want to:

- Track the total vehicle yaw rotation.
- Avoid discontinuities in the vehicle state estimators.

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers (SAE), 1992.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Vector Concatenate, Matrix Concatenate | Vehicle Body 6DOF | Vehicle Body 3DOF Longitudinal

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2018a

Vehicle Body 6DOF

6DOF rigid vehicle body to calculate translational and rotational motion
Library: Vehicle Dynamics Blockset / Vehicle Body
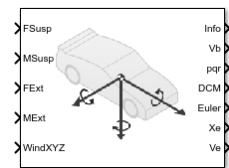
Description

The Vehicle Body 6DOF block implements a six degrees-of-freedom (DOF) rigid two-axle vehicle body model to calculate longitudinal, lateral, vertical, pitch, roll, and yaw motion. The block accounts for body mass, inertia, aerodynamic drag, road incline, and weight distribution between the axles due to suspension and external forces and moments. Use the **Inertial Loads** parameters to analyze the vehicle dynamics under different loading conditions.

You can connect the block to virtual sensors, suspension system, or external systems like body control actuators. Use the Vehicle Body 6DOF block in ride and handling studies to model the effects of drag forces, passenger loading, and suspension hardpoint locations.

Inertial Loads

To analyze the vehicle dynamics under different loading conditions, use the **Inertial Loads** parameters. Specifically, you can specify these loads:

- Front powertrain
- Front and rear row passengers
- Overhead cargo
- Rear cargo

For each of the loads, you can specify the mass, location, and inertia.

The dots in this illustration indicate example load locations. The table provides the corresponding location parameter sign settings.
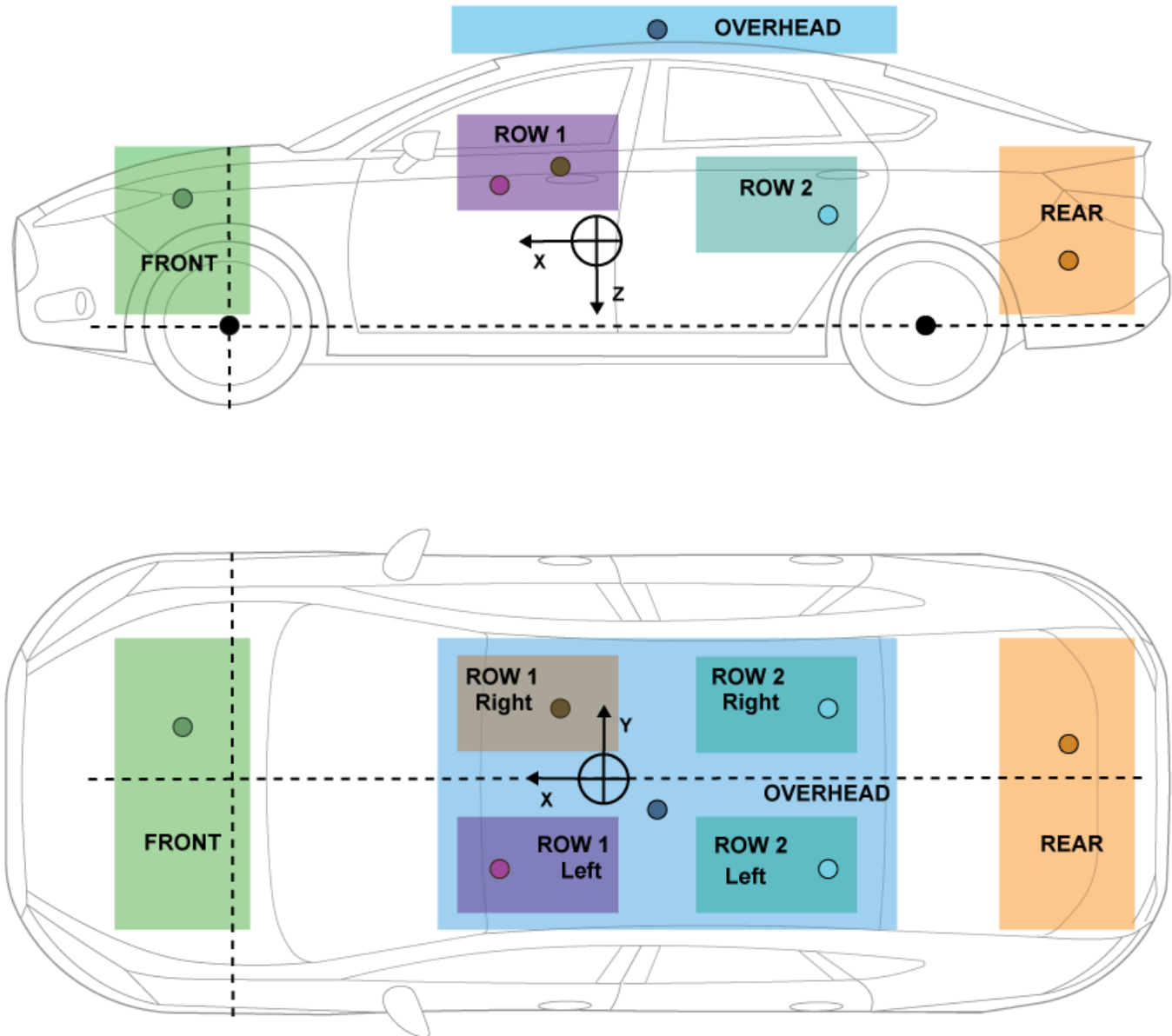
This table summarizes the parameter settings that specify the load locations indicated by the dots. For the location, the block uses this distance vector:

- Front suspension hardpoint to load, along the vehicle-fixed *x*-axis
- Vehicle centerline to load, along the vehicle-fixed *y*-axis
- Front suspension hardpoint to load, along the vehicle-fixed *z*-axis

Load	Parameter	Example Location
Front	**Distance vector from front axle, z1R**	• z1R(1,1)<0 — Forward of the front axle • z1R(1,2)>0 — Right of the vehicle centerline • z1R(1,3)>0 — Above the front axle suspension hardpoint
Overhead	**Distance vector from front axle, z2R**	• z2R(1,1)>0 — Rear of the front axle • z2R(1,2)<0 — Left of the vehicle centerline • z2R(1,3)>0 — Above the front axle suspension hardpoint
Row 1, left side	**Distance vector from front axle, z3R**	• z3R(1,1)>0 — Rear of the front axle • z3R(1,2)<0 — Left of the vehicle centerline • z3R(1,3)>0 — Above the front axle suspension hardpoint
Row 1, right side	**Distance vector from front axle, z4R**	• z4R(1,1)>0 — Rear of the front axle • z4R(1,2)>0 — Right of the vehicle centerline • z4R(1,3)>0 — Above the front axle suspension hardpoint
Row 2, left side	**Distance vector from front axle, z5R**	• z5R(1,1)>0 — Rear of the front axle • z5R(1,2)<0 — Left of the vehicle centerline • z5R(1,3)>0 — Above the front axle suspension hardpoint
Row 2, right side	**Distance vector from front axle, z6R**	• z6R(1,1)>0 — Rear of the front axle • z6R(1,2)>0 — Right of the vehicle centerline • z6R(1,3)>0 — Above the front axle suspension hardpoint
Rear	**Distance vector from front axle, z7R**	• z7R(1,1)>0 — Rear of the front axle • z7R(1,2)>0 — Right of the vehicle centerline • z7R(1,3)>0 — Above the front axle suspension hardpoint

Equations of Motion

To determine the vehicle motion, the block implements calculations for the rigid body vehicle dynamics, wind drag, inertial loads, and coordinate transformations. The body-fixed and the vehicle-fixed are the same coordinate systems.

The Vehicle Body 6DOF block considers the rotation of a body-fixed coordinate frame about a flat earth-fixed inertial reference frame. The origin of the body-fixed coordinate frame is the vehicle center of gravity of the body.
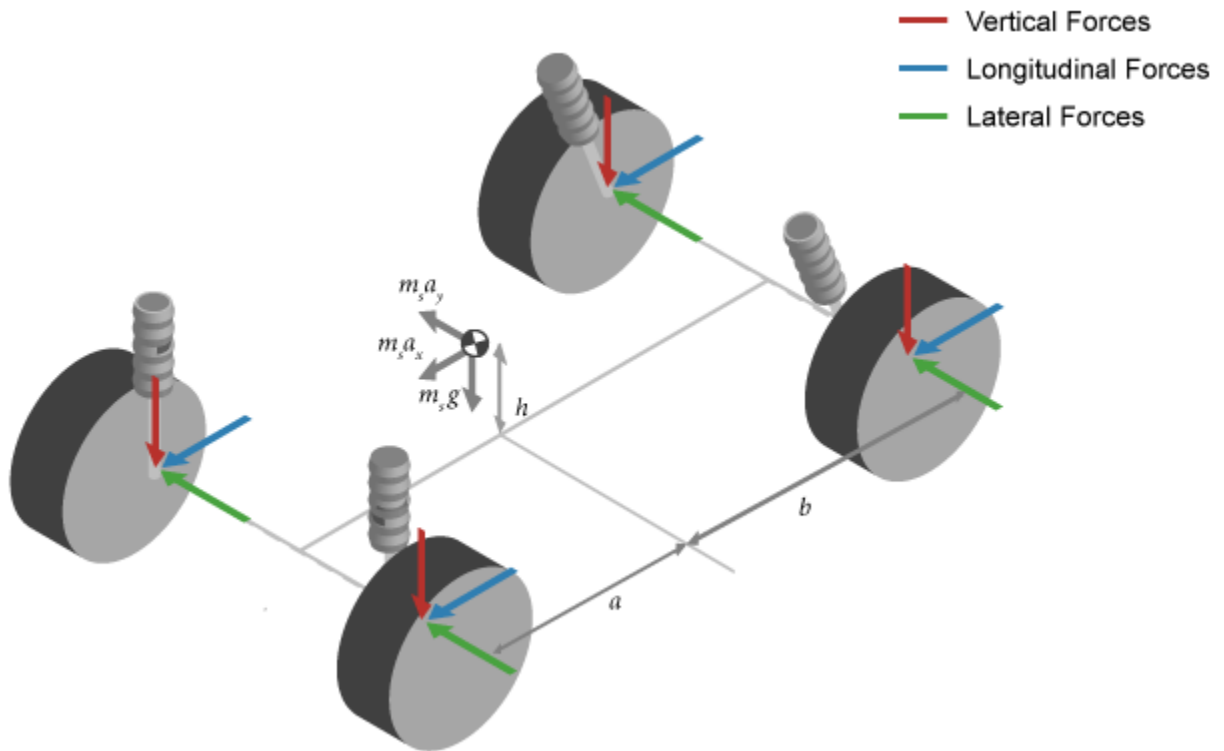
The block uses this equation to calculate the translational motion of the body-fixed coordinate frame, where the applied forces $[F_x\ F_y\ F_z]^T$ are in the body-fixed frame, and the mass of the body, m, is assumed constant.

$$\bar{F}_b = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = m\left(\dot{\bar{V}}_b + \bar{\omega} \times \bar{V}_b\right)$$

$$\bar{M}_b = \begin{bmatrix} L \\ M \\ N \end{bmatrix} = I\dot{\bar{\omega}} + \bar{\omega} \times (I\bar{\omega})$$

$$I = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

To determine the relationship between the body-fixed angular velocity vector, $[p\ q\ r]^T$, and the rate of change of the Euler angles, $\begin{bmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T$, the block resolves the Euler rates into the body-fixed frame.

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}\begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}\begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}\begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \equiv J^{-1}\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}$$

Inverting J gives the required relationship to determine the Euler rate vector.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = J\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & (\sin\phi\tan\theta) & (\cos\phi\tan\theta) \\ 0 & \cos\phi & -\sin\phi \\ 0 & \dfrac{\sin\phi}{\cos\theta} & \dfrac{\cos\phi}{\cos\theta} \end{bmatrix}\begin{bmatrix} p \\ q \\ r \end{bmatrix}$$

The applied forces and moments are the sum of the drag, gravitational, external, and suspension forces.

$$\bar{F}_b = \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = \begin{bmatrix} F_{d_x} \\ F_{d_y} \\ F_{d_z} \end{bmatrix} + \begin{bmatrix} F_{g_x} \\ F_{g_y} \\ F_{g_z} \end{bmatrix} + \begin{bmatrix} F_{ext_x} \\ F_{ext_y} \\ F_{ext_z} \end{bmatrix} + \begin{bmatrix} F_{FL_x} \\ F_{FL_y} \\ F_{FL_z} \end{bmatrix} + \begin{bmatrix} F_{FR_x} \\ F_{FR_y} \\ F_{FR_z} \end{bmatrix} + \begin{bmatrix} F_{RL_x} \\ F_{RL_y} \\ F_{RL_z} \end{bmatrix} + \begin{bmatrix} F_{RR_x} \\ F_{RR_y} \\ F_{RR_z} \end{bmatrix}$$

$$\bar{M}_b = \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} = \begin{bmatrix} M_{d_x} \\ M_{d_y} \\ M_{d_z} \end{bmatrix} + \begin{bmatrix} M_{ext_x} \\ M_{ext_y} \\ M_{ext_z} \end{bmatrix} + \begin{bmatrix} M_{FL_x} \\ M_{FL_y} \\ M_{FL_z} \end{bmatrix} + \begin{bmatrix} M_{FR_x} \\ M_{FR_y} \\ M_{FR_z} \end{bmatrix} + \begin{bmatrix} M_{RL_x} \\ M_{RL_y} \\ M_{RL_z} \end{bmatrix} + \begin{bmatrix} M_{RR_x} \\ M_{RR_y} \\ M_{RR_z} \end{bmatrix} + \bar{M}_F$$

Calculation	Implementation
Load masses and inertias	Block uses parallel axis theorem to resolve the individual load masses and inertias with the vehicle mass and inertia. $$J_{ij} = I_{ij} + m(|R|^2 \delta_{ij} - R_i R_j)$$
Gravitational forces, F_g	Block uses direction cosine matrix (DCM) to transform the gravitational vector in the inertial-fixed frame to the body-fixed frame.
Drag forces, F_d*, and moments,* M_d	To determine a relative airspeed, the block subtracts the wind speed from the vehicle center of mass (CM) velocity. Using the relative airspeed, the block determines the drag forces. $$\bar{w} = \sqrt{(\dot{x}_b - w_x)^2 + (\dot{x}_y - w_x)^2 + (w_z)^2}$$ $$F_{dx} = -\frac{1}{2TR}C_d A_f P_{abs}(\bar{w}$$ $$F_{dy} = -\frac{1}{2TR}C_s A_f P_{abs}(\bar{w}$$ $$F_{dz} = -\frac{1}{2TR}C_l A_f P_{abs}(\bar{w}$$ Using the relative airspeed, the block determines the drag moments. $$M_{dr} = -\frac{1}{2TR}C_{rm} A_f P_{abs}(\bar{w}(a+b)$$ $$M_{dp} = -\frac{1}{2TR}C_{pm} A_f P_{abs}(\bar{w}(a+b)$$ $$M_{dy} = -\frac{1}{2TR}C_{ym} A_f P_{abs}(\bar{w}(a+b)$$
External forces, F_{in}*, and moments,* M_{in}	External forces and moments are input via ports FExt and MExt.

Calculation	Implementation
Suspension forces and moments	Block assumes that the suspension forces and moments act on these hardpoint locations: • F_{FL}, M_{FL} — Front left • F_{FR}, M_{FR} — Front right • F_{RL}, M_{RL} — Rear left • F_{RR}, M_{RR} — Rear right

The equations use these variables.

$x, \dot{x}, \ddot{x}$	Vehicle CM displacement, velocity, and acceleration along the vehicle-fixed *x*-axis
$y, \dot{y}, \ddot{y}$	Vehicle CM displacement, velocity, and acceleration along the vehicle-fixed *y*-axis
$z, \dot{z}, \ddot{z}$	Vehicle CM displacement, velocity, and acceleration along the vehicle-fixed *z*-axis
φ	Rotation of the vehicle-fixed frame about the earth-fixed *X*-axis (roll)
θ	Rotation of the vehicle-fixed frame about the earth-fixed *Y*-axis (pitch)
ψ	Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw)
F_{FLx}, F_{FLy}, F_{FLz}	Suspension forces applied to front left hardpoint along the vehicle-fixed *x*-, *y*-, and *z*-axes
F_{FRx}, F_{FRy}, F_{FRz}	Suspension forces applied to front right hardpoint along the vehicle-fixed *x*-, *y*-, and *z*-axes
F_{RLx}, F_{RLy}, F_{RLz}	Suspension forces applied to rear left hardpoint along the vehicle-fixed *x*-, *y*-, and *z*-axes
F_{RRx}, F_{RRy}, F_{RRz}	Suspension forces applied to rear right hardpoint along the vehicle-fixed *x*-, *y*-, and *z*-axes
M_{Fx}, F_{Fy}, F_{Fz}	Suspension moments applied to vehicle CM about the vehicle-fixed *x*-, *y*-, and *z*-axes
F_{extx}, F_{exty}, F_{extz}	External forces applied to vehicle CM along the vehicle-fixed *x*-, *y*-, and *z*-axes
F_{dx}, F_{dy}, F_{dz}	Drag forces applied to vehicle CM along the vehicle-fixed *x*-, *y*-, and *z*-axes
M_{extx}, M_{exty}, M_{extz}	External moment about vehicle CM about the vehicle-fixed *x*-, *y*-, and *z*-axes
M_{dx}, M_{dy}, M_{dz}	Drag moment about vehicle CM about the vehicle-fixed *x*-, *y*-, and *z*-axes
I	Vehicle body moments of inertia
a, b	Distance of front and rear wheels, respectively, from the normal projection point of vehicle CM onto the common axle plane
h	Height of vehicle CM above the axle plane
w_F, w_R	Front and rear track widths
γ	Road grade angle
C_d	Air drag coefficient acting along vehicle-fixed *x*-axis
C_s	Air drag coefficient acting along vehicle-fixed *y*-axis

C_l	Air drag coefficient acting along vehicle-fixed z-axis
C_{rm}	Air drag roll moment acting about vehicle-fixed x-axis
C_{pm}	Air drag pitch moment acting about the vehicle-fixed y-axis
C_{ym}	Air drag yaw moment acting about vehicle-fixed z-axis
A_f	Frontal area
R	Atmospheric specific gas constant
T	Environmental air temperature
P_{abs}	Environmental absolute pressure
w_x, w_y, w_z	Wind speed along the vehicle-fixed x-, y-, and z-axes
W_x, W_y, W_z	Wind speed along inertial X-, Y-, and Z-axes

Ports

Input

FSusp — Suspension forces on vehicle
`array`

Suspension longitudinal, lateral, and vertical suspension forces applied to the vehicle at the hardpoint location, in N. Signal dimensions are [3x4].

$$FSusp = \begin{bmatrix} F_{x_{FL}} & F_{x_{FR}} & F_{x_{RL}} & F_{x_{RR}} \\ F_{y_{FL}} & F_{y_{FR}} & F_{y_{RL}} & F_{y_{RR}} \\ F_{z_{FL}} & F_{z_{FR}} & F_{z_{RL}} & F_{z_{RR}} \end{bmatrix}$$

Array Element	Axle	Track	Force Axis
FSusp(1,1)	Front	Left	Vehicle-fixed x-axis (longitudinal)
FSusp(1,2)	Front	Right	
FSusp(1,3)	Rear	Left	
FSusp(1,4)	Rear	Right	
FSusp(2,1)	Front	Left	Vehicle-fixed y-axis (lateral)
FSusp(2,2)	Front	Right	
FSusp(2,3)	Rear	Left	
FSusp(2,4)	Rear	Right	
FSusp(3,1)	Front	Left	Vehicle-fixed z-axis (vertical)
FSusp(3,2)	Front	Right	
FSusp(3,3)	Rear	Left	
FSusp(3,4)	Rear	Right	

MSusp — Suspension moment on vehicle
`array`

Suspension longitudinal, lateral, and vertical suspension moments applied about the vehicle at the hardpoint location, in N. Signal dimensions are [3x4].

$$MSusp = \begin{bmatrix} M_{x_{FL}} & M_{x_{FR}} & M_{x_{RL}} & M_{x_{RR}} \\ M_{y_{FL}} & M_{y_{FR}} & M_{y_{RL}} & M_{y_{RR}} \\ M_{z_{FL}} & M_{z_{FR}} & M_{z_{RL}} & M_{z_{RR}} \end{bmatrix}$$

Array Element	Axle	Track	Moment Axis
MSusp(1,1)	Front	Left	Vehicle-fixed x-axis (longitudinal)
MSusp(1,2)	Front	Right	
MSusp(1,3)	Rear	Left	
MSusp(1,4)	Rear	Right	
MSusp(2,1)	Front	Left	Vehicle-fixed y-axis (lateral)
MSusp(2,2)	Front	Right	
MSusp(2,3)	Rear	Left	
MSusp(2,4)	Rear	Right	
MSusp(3,1)	Front	Left	Vehicle-fixed z-axis (vertical)
MSusp(3,2)	Front	Right	
MSusp(3,3)	Rear	Left	
MSusp(3,4)	Rear	Right	

FExt — External forces acting on vehicle
`vector`

External forces on vehicle, in N. Signal vector dimensions are `[1x3]` or `[3x1]`.

$$FExt = F_{ext} = \begin{bmatrix} F_{ext_x} & F_{ext_y} & F_{ext_z} \end{bmatrix} or \begin{bmatrix} F_{ext_x} \\ F_{ext_y} \\ F_{ext_z} \end{bmatrix}$$

Array Element	Force Axis
FExt(1,1)	Vehicle-fixed x-axis (longitudinal)
FExt(1,2) or FExt(2,1)	Vehicle-fixed y-axis (lateral)
FExt(1,3) or FExt(3,1)	Vehicle-fixed z-axis (vertical)

MExt — External moments acting on vehicle
`vector`

External moments acting on vehicle, in N·m. Signal vector dimensions are `[1x3]` or `[3x1]`.

$$MExt = M_{ext} = \begin{bmatrix} M_{ext_x} & M_{ext_y} & M_{ext_z} \end{bmatrix} or \begin{bmatrix} M_{ext_x} \\ M_{ext_y} \\ M_{ext_z} \end{bmatrix}$$

Array Element	Force Axis
MExt(1,1)	Vehicle-fixed *x*-axis (longitudinal)
MExt(1,2) or MExt(2,1)	Vehicle-fixed *y*-axis (lateral)
MExt(1,3) or MExt(3,1)	Vehicle-fixed *z*-axis (vertical)

WindXYZ — Wind speed
array

Wind speed, W_x, W_y, W_z along inertial *X*-, *Y*-, and *Z*-axes, in m/s. Signal vector dimensions are [1x3] or [3x1].

AirTemp — Ambient air temperature
scalar

Ambient air temperature, T_{air}, in K.

Dependencies

To enable this port, on the **Environment** pane, select **Air temperature**.

Output Arguments

Info — Bus signal
bus

Bus signal containing these block values.

Signal				Description	Value	Units
InertFrm	Cg	Disp	X	Vehicle CM displacement along the earth-fixed *X*-axis	Computed	m
			Y	Vehicle CM displacement along the earth-fixed *Y*-axis	Computed	m
			Z	Vehicle CM displacement along the earth-fixed *Z*-axis	Computed	m
		Vel	Xdot	Vehicle CM velocity along the earth-fixed *X*-axis	Computed	m/s
			Ydot	Vehicle CM velocity along the earth-fixed *Y*-axis	Computed	m/s
			Zdot	Vehicle CM velocity along the earth-fixed *Z*-axis	Computed	m/s
		Ang	phi	Rotation of the vehicle-fixed frame about the earth-fixed *X*-axis (roll)	Computed	rad

Signal					Description	Value	Units
			theta		Rotation of the vehicle-fixed frame about the earth-fixed *Y*-axis (pitch)	Computed	rad
			psi		Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw)	Computed	rad
	FrntAxl	Lft	Disp	X	Front left axle displacement along the earth-fixed *X*-axis	Computed	m
				Y	Front left axle displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Front left axle displacement along the earth-fixed *Z*-axis	Computed	m
			Vel	Xdot	Front left axle velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Front left axle velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Front left axle velocity along the earth-fixed *Z*-axis	Computed	m/s
		Rght	Disp	X	Front right axle displacement along the earth-fixed *X*-axis	Computed	m
				Y	Front right axle displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Front right axle displacement along the earth-fixed *Z*-axis	Computed	m
			Vel	Xdot	Front right axle velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Front right axle velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Front right axle velocity along the earth-fixed *Z*-axis	Computed	m/s
	RearAxl	Lft	Disp	X	Rear left axle displacement along the earth-fixed *X*-axis	Computed	m

Signal					Description	Value	Units
				Y	Rear left axle displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Rear left axle displacement along the earth-fixed *Z*-axis	Computed	m
			Vel	Xdot	Rear left axle velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Rear left axle velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Rear left axle velocity along the earth-fixed *Z*-axis	Computed	m/s
		Rght	Disp	X	Rear right axle displacement along the earth-fixed *X*-axis	Computed	m
				Y	Rear right axle displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Rear right axle displacement along the earth-fixed *Z*-axis	Computed	m
			Vel	Xdot	Rear right axle velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Rear right axle velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Rear right axle velocity along the earth-fixed *Z*-axis	Computed	m/s
	Geom	Disp	X		Vehicle chassis offset from axle plane along the earth-fixed *X*-axis	Computed	m
			Y		Vehicle chassis offset from center plane along the earth-fixed *Y*-axis	Computed	m
			Z		Vehicle chassis offset from axle plane along the earth-fixed *Z*-axis	Computed	m
		Vel	Xdot		Vehicle chassis offset velocity along the earth-fixed *X*-axis	Computed	m/s

Signal				Description	Value	Units
			Ydot	Vehicle chassis offset velocity along the earth-fixed *Y*-axis	Computed	m/s
			Zdot	Vehicle chassis offset velocity along the earth-fixed *Z*-axis	Computed	m/s
BdyFrm	Cg	Vel	xdot	Vehicle CM velocity along the vehicle-fixed *x*-axis	Computed	m/s
			ydot	Vehicle CM velocity along the vehicle-fixed *y*-axis	Computed	m/s
			zdot	Vehicle CM velocity along the vehicle-fixed *z*-axis	Computed	m/s
		AngVel	p	Vehicle angular velocity about the vehicle-fixed *x*-axis (roll rate)	Computed	rad/s
			q	Vehicle angular velocity about the vehicle-fixed *y*-axis (pitch rate)	Computed	rad/s
			r	Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate)	Computed	rad/s
		Acc	ax	Vehicle CM acceleration along the vehicle-fixed *x*-axis	Computed	gn
			ay	Vehicle CM acceleration along the vehicle-fixed *y*-axis	Computed	gn
			az	Vehicle CM acceleration along the vehicle-fixed *z*-axis	Computed	gn
			xddot	Vehicle CM acceleration along the vehicle-fixed *x*-axis	Computed	m/s^2
			yddot	Vehicle CM acceleration along the vehicle-fixed *y*-axis	Computed	m/s^2
			zddot	Vehicle CM acceleration along the vehicle-fixed *z*-axis	Computed	m/s^2
		DCM	Direction cosine matrix		Computed	rad
	Forces	Body	Fx	Net force on vehicle CM along the vehicle-fixed *x*-axis	Computed	N

Signal					Description	Value	Units
			Fy		Net force on vehicle CM along the vehicle-fixed y-axis	Computed	N
			Fz		Net force on vehicle CM along the vehicle-fixed z-axis	Computed	N
		FrntAxl	Lft	Fx	Longitudinal force on front left axle along the vehicle-fixed x-axis	Computed	N
				Fy	Lateral force on front axle left along the vehicle-fixed y-axis	Computed	N
				Fz	Normal force on front axle left along the vehicle-fixed z-axis	Computed	N
			Rght	Fx	Longitudinal force on front right axle along the vehicle-fixed x-axis	Computed	N
				Fy	Lateral force on front axle right along the vehicle-fixed y-axis	Computed	N
				Fz	Normal force on front axle right along the vehicle-fixed z-axis	Computed	N
		RearAxl	Lft	Fx	Longitudinal force on rear left axle along the vehicle-fixed x-axis	Computed	N
				Fy	Lateral force on rear left axle along the vehicle-fixed y-axis	Computed	N
				Fz	Normal force on rear left axle along the vehicle-fixed z-axis	Computed	N
			Rght	Fx	Longitudinal force on rear right axle along the vehicle-fixed x-axis	Computed	N
				Fy	Lateral force on rear right axle along the vehicle-fixed y-axis	Computed	N
				Fz	Normal force on rear right axle along the vehicle-fixed z-axis	Computed	N
		Tires	FrntTires	Lft Fx	Front left tire force along the vehicle-fixed x-axis	Computed	N

Signal						Description	Value	Units
					F y	Front left tire force along the vehicle-fixed *y*-axis	Computed	N
					F z	Front left tire force along the vehicle-fixed *z*-axis	Computed	N
				R g h t	F x	Front right tire force along the vehicle-fixed *x*-axis	Computed	N
					F y	Front right tire force along the vehicle-fixed *y*-axis	Computed	N
					F z	Front right tire force along the vehicle-fixed *z*-axis	Computed	N
		RearTires	L f t		F x	Rear left tire force along the vehicle-fixed *x*-axis	Computed	N
					F y	Rear left tire force along the vehicle-fixed *y*-axis	Computed	N
					F z	Rear left tire force along the vehicle-fixed *z*-axis	Computed	N
			R g h t		F x	Rear right tire force along the vehicle-fixed *x*-axis	Computed	N
					F y	Rear right tire force along the vehicle-fixed *y*-axis	Computed	N
					F z	Rear right tire force along the vehicle-fixed *z*-axis	Computed	N
	Drag	Fx				Drag force on vehicle CM along the vehicle-fixed *x*-axis	Computed	N
		Fy				Drag force on vehicle CM along the vehicle-fixed *y*-axis	Computed	N
		Fz				Drag force on vehicle CM along the vehicle-fixed *z*-axis	Computed	N
	Grvty	Fx				Gravity force on vehicle CM along the vehicle-fixed *x*-axis	Computed	N
		Fy				Gravity force on vehicle CM along the vehicle-fixed *y*-axis	Computed	N
		Fz				Gravity force on vehicle CM along the vehicle-fixed *z*-axis	Computed	N

Signal				Description	Value	Units
	Moments	Body	Mx	Body moment on vehicle CM about the vehicle-fixed *x*-axis	Computed	N·m
			My	Body moment on vehicle CM about the vehicle-fixed *y*-axis	Computed	N·m
			Mz	Body moment on vehicle CM about the vehicle-fixed *z*-axis	Computed	N·m
		Drag	Mx	Drag moment on vehicle CM about the vehicle-fixed *x*-axis	Computed	N·m
			My	Drag moment on vehicle CM about the vehicle-fixed *y*-axis	Computed	N·m
			Mz	Drag moment on vehicle CM about the vehicle-fixed *z*-axis	Computed	N·m
	FrntAxl	Lft	Disp	x	Front left axle displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Front left axle displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Front left axle displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Front left axle velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Front left axle velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Front left axle velocity along the vehicle-fixed *z*-axis	Computed	m/s
		Rght	Disp	x	Front right axle displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Front right axle displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Front right axle displacement along the vehicle-fixed *z*-axis	Computed	m

Signal					Description	Value	Units
			Vel	xdot	Front right axle velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Front right axle velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Front right axle velocity along the vehicle-fixed *z*-axis	Computed	m/s
	RearAxl	Lft	Disp	x	Rear left axle displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Rear left axle displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Rear left axle displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Rear left axle velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Rear left axle velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Rear left axle velocity along the vehicle-fixed *z*-axis	Computed	m/s
		Rght	Disp	x	Rear right axle displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Rear right axle displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Rear right axle displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Rear right axle velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Rear right axle velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Rear right axle velocity along the vehicle-fixed *z*-axis	Computed	m/s

Signal				Description	Value	Units
	Pwr	PwrExt		Applied external power	Computed	W
		Drag		Power loss due to drag	Computed	W
	Geom	Disp	x	Vehicle chassis offset from axle plane along the vehicle-fixed *x*-axis	Input	m
			y	Vehicle chassis offset from center plane along the vehicle-fixed *y*-axis	Input	m
			z	Vehicle chassis offset from axle plane along the earth-fixed *z*-axis	Input	m
		Vel	xdot	Vehicle chassis offset velocity along the vehicle-fixed *x*-axis	Computed	m/s
			ydot	Vehicle chassis offset velocity along the vehicle-fixed *y*-axis	Computed	m/s
			zdot	Vehicle chassis offset velocity along the vehicle-fixed *z*-axis	Computed	m/s
		Ang	Beta	Body slip angle, β $$\beta = \frac{V_y}{V_x}$$	Computed	rad

Vb — Vehicle velocity along the vehicle-fixed frame
vector

Vehicle CM velocity along the vehicle-fixed *x*-, *y*-, *z*- axes, respectively, in m/s.

pqr — Vehicle angular velocity about the vehicle-fixed frame
vector

Vehicle CM angular velocity about the vehicle-fixed *x*(roll rate)-, *y*(pitch rate)-, *z*(yaw rate)- axes, respectively, in rad/s.

DCM — Direction cosine matrix
array

Direction cosine matrix, in rad.

Euler — Euler angles
array

Euler angles, φ, θ, and ψ, respectively, in rad.

Xe — Vehicle position in the inertial reference frame
vector

Vehicle CM position along inertial-fixed *X*-, *Y*-, *Z*- axes, respectively, in m.

Ve — Vehicle velocity in the inertial reference frame
vector

Vehicle CM velocity along inertial-fixed *X*-, *Y*-, *Z*- axes, respectively, in m/s.

Parameters

Chassis

Vehicle mass, m — Mass
2000 (default) | scalar

Vehicle mass, *m*, in kg.

Longitudinal distance from center of mass to front axle, a — Distance
1.4 (default) | scalar

Distance from vehicle CM to front axle, *a*, in m.
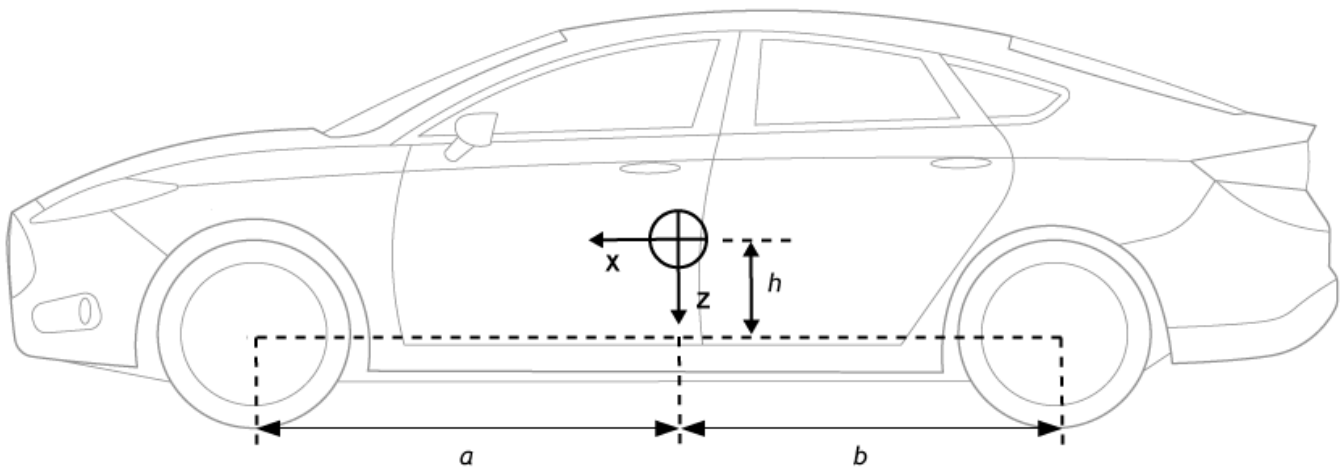
Longitudinal distance from center of mass to rear axle, b — Distance
1.6 (default) | scalar

Distance from vehicle CM to front axle, *b*, in m.
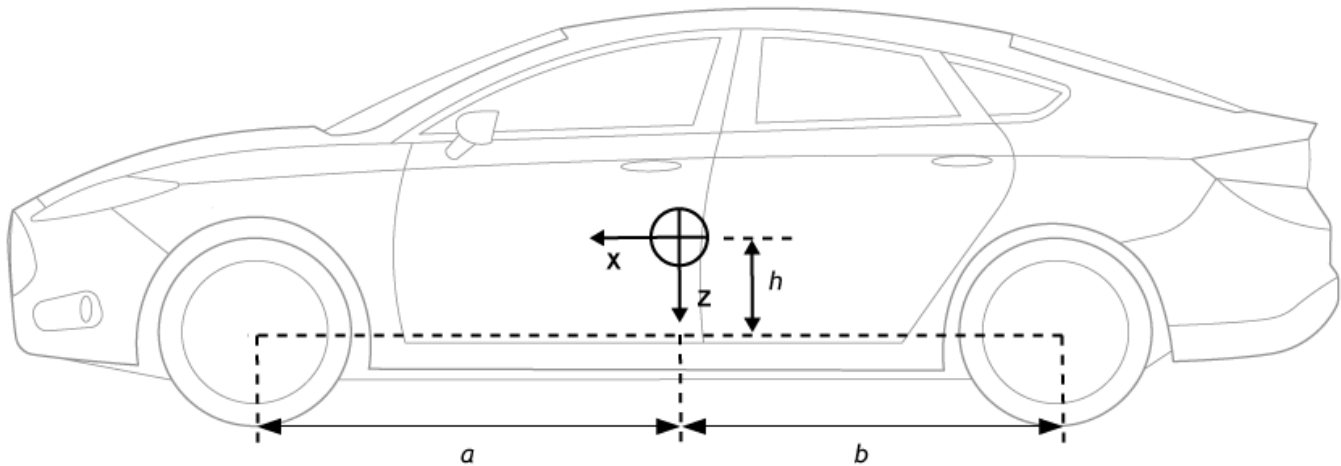
Lateral distance from geometric centerline to center of mass, d — Distance
0 (default) | `scalar`

Lateral distance from geometric centerline to center of mass, d, in m, along the vehicle-fixed y. Positive values indicate that the vehicle CM is to the right of the geometric centerline. Negative values indicate that the vehicle CM is to the left of the geometric centerline.
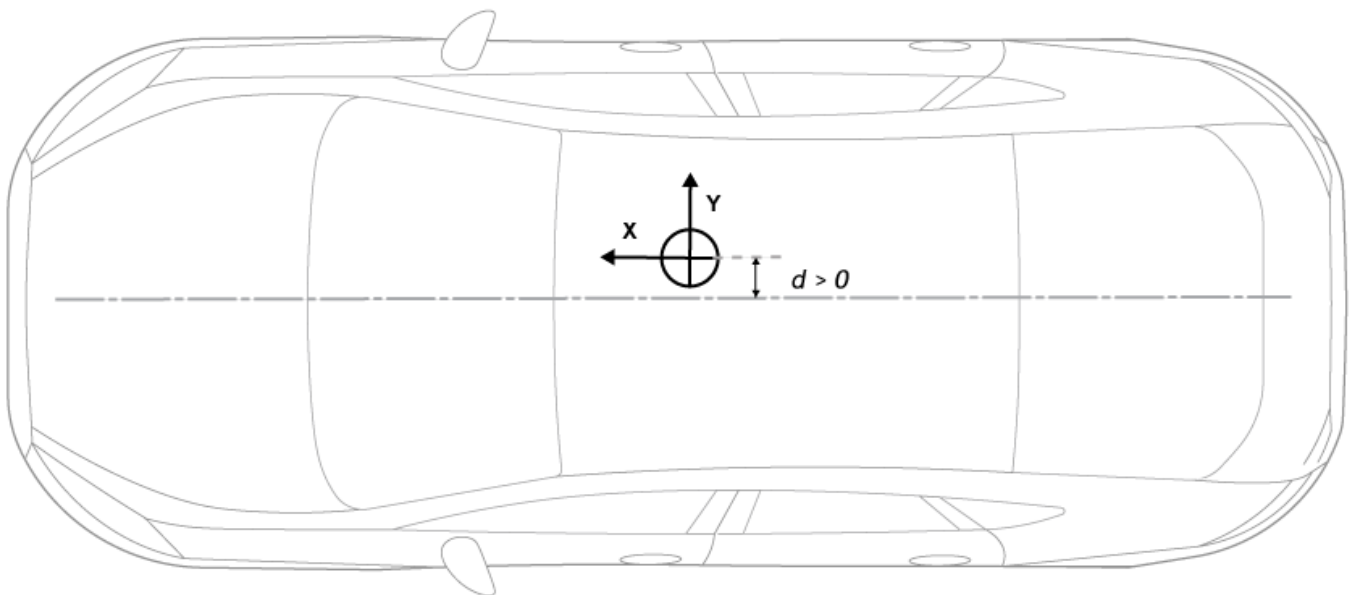
Vertical distance from center of mass to axle plane, h — Distance
.35 (default) | `scalar`

Vertical distance from vehicle CM to axle plane, h, in m.
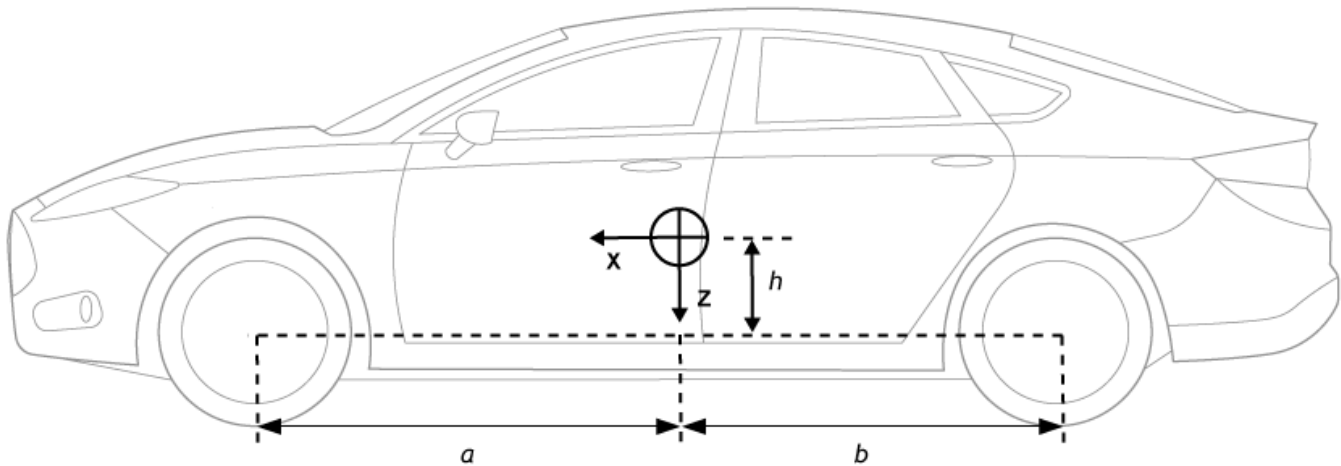
Initial position in the inertial frame [Xeo,Yeo,Zeo], Xe_o — Position
[0,0,0] (default) | vector

Initial position of vehicle in the inertial frame, Xe_o, in m.

Initial velocity in body axes [xdot_o,ydot_o,zdot_o], xbdot_o — Velocity
[0,0,0] (default) | vector

Initial vehicle CM velocity along the vehicle-fixed x, y-, and z- axes, respectively, in m/s.

Initial Euler orientation [roll, pitch, yaw], eul_o — Rotation
[0,0,0] (default) | vector

Initial Euler rotation of the vehicle-fixed frame about the earth-fixed X(roll)-, Y(pitch)-, Z(yaw)- axes, respectively, in rad.

Initial body rotation rates [p,q,r], p_o — Rotation rate
[0,0,0] (default) | vector

Initial vehicle CM angular velocity about the vehicle-fixed x(roll rate)-, y(pitch rate)-, z(yaw rate)- axes, respectively, in rad/s.

Chassis inertia tensor, Iveh — Inertia
[430 0 0; 0 1900 0; 0 0 2100] (default) | array

Vehicle inertia tensor, I_{veh}, in kg*m^2. Dimensions are [3-by-3].

Track widths [front,rear], w — Widths
[1.9,1.9] (default) | vector

Front and rear track width, in m. Dimensions are [1-by-2].

Inertial Loads

Front

Mass, z1m — Mass
0 (default) | scalar

Mass, *z1m*, in kg.

Distance vector from front axle, z1R — Distance
[-.25,.125,.15] (default) | vector

Distance vector from front axle to load, *z1R*, in m. Dimensions are [1-by-3].

Array Element	Description
z1R(1,1)	Front suspension hardpoint to load, along vehicle-fixed *x*-axis
z1R(1,2)	Vehicle centerline to load, along vehicle-fixed *y*-axis
z1R(1,3)	Front suspension hardpoint to load, along vehicle-fixed *z*-axis
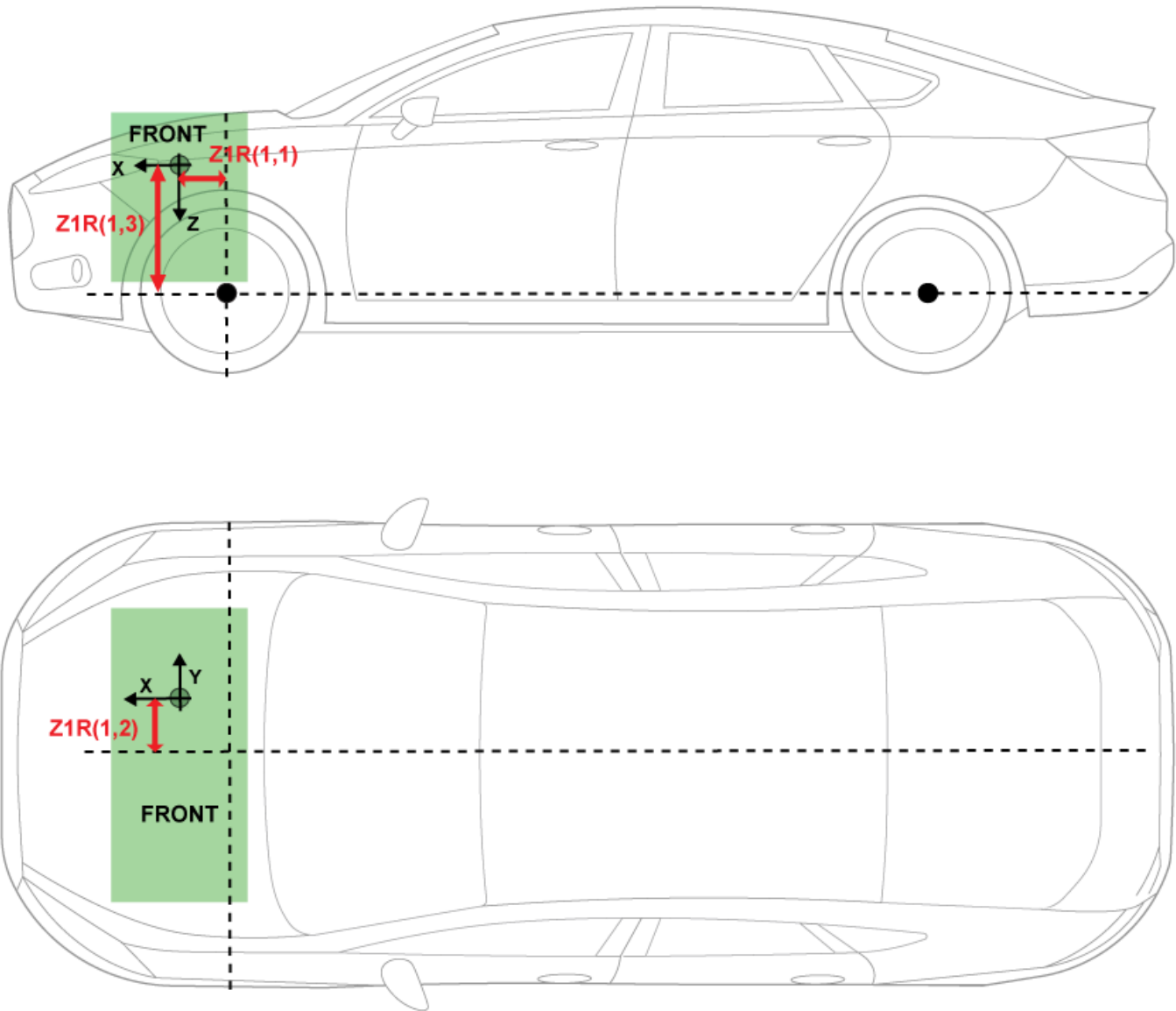
For example, this table summarizes the parameter settings that specify the load location indicated by the dots.

Example Location	Sign
• Forward of the front axle • Right of the vehicle centerline • Above the front axle suspension hardpoint	• z1R(1,1) < 0 • z1R(1,2) > 0 • z1R(1,3) > 0

Inertia tensor, z1I — Inertia
[1.4,-.2,.1;-.2,1.4,.1;.1,.1,2.25].*0 (default) | array

Inertia tensor, *z1I*, in kg·m^2. Dimensions are [3-by-3].

$$z1I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

The tensor uses a coordinate system with an origin at the load CM.

- *x*-axis along the vehicle-fixed *x*-axis
- *y*-axis along the vehicle-fixed *y*-axis
- *z*-axis along the vehicle-fixed *z*-axis
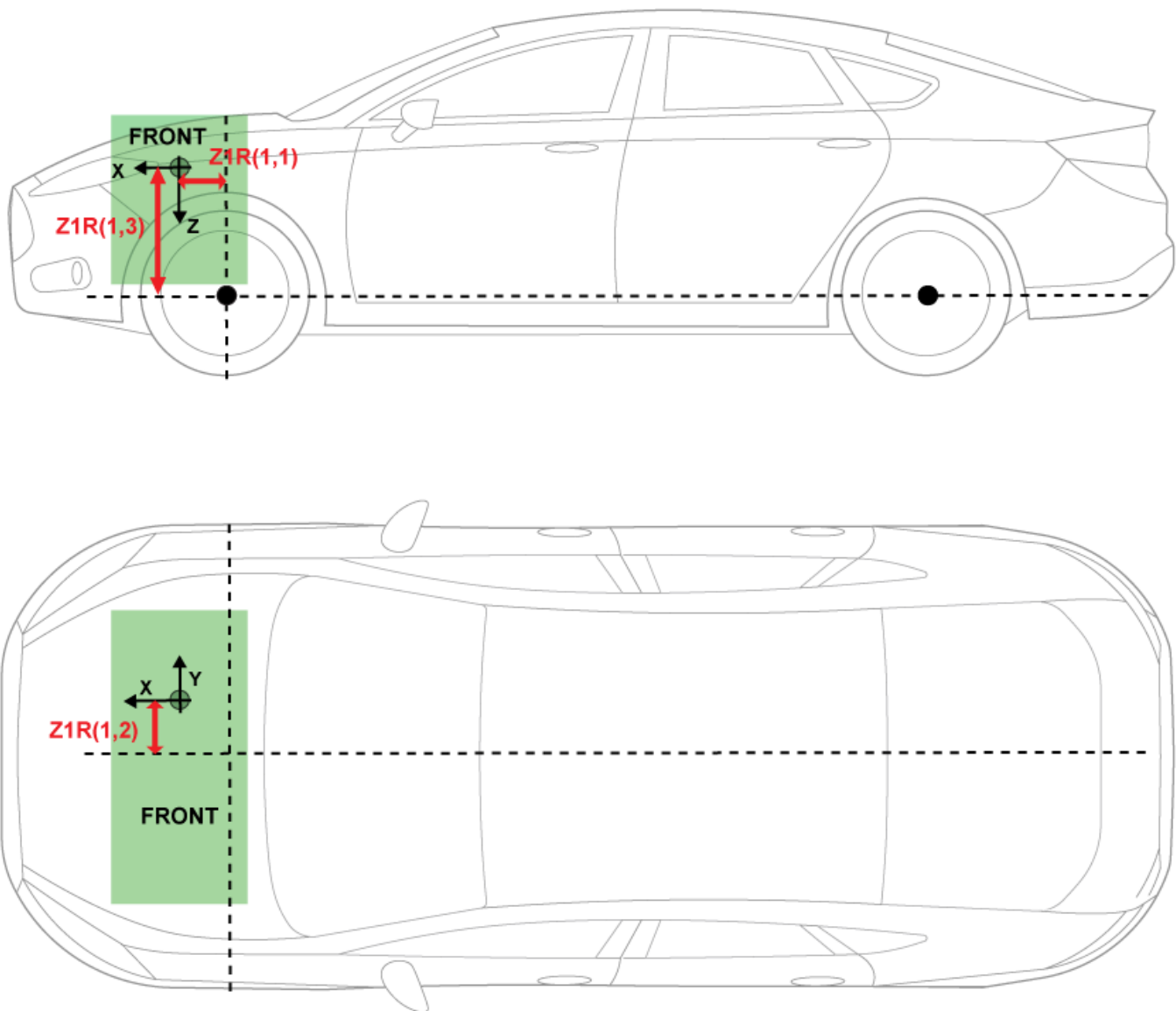
Overhead

`Mass, z2m` — Mass
`0` (default) | `scalar`

Mass, *z2m*, in kg.

`Distance vector from front axle, z2R` — Distance
`[1.4,0,.8]` (default) | `vector`

Distance vector from front axle to load, *z2R*, in m. Dimensions are `[1-by-3]`.

Array Element	Description
z2R(1,1)	Front suspension hardpoint to load, along vehicle-fixed *x*-axis
z2R(1,2)	Vehicle centerline to load, along vehicle-fixed *y*-axis
z2R(1,3)	Front suspension hardpoint to load, along vehicle-fixed *z*-axis
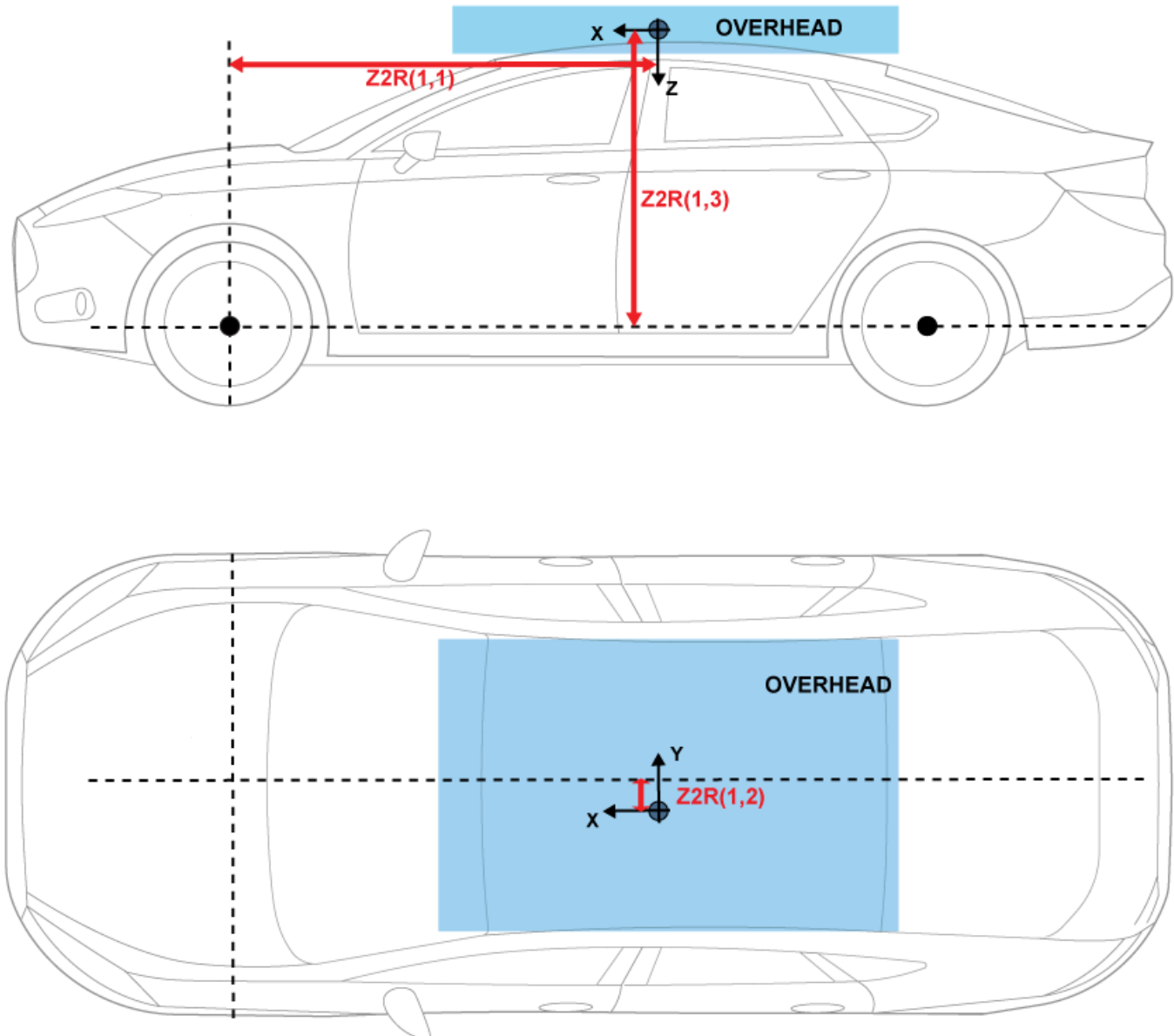
For example, this table summarizes the parameter settings that specify the load location indicated by the dot.

Example Location	Sign
• Rear of the front axle	• z2R(1,1) > 0
• Left of the vehicle centerline	• z2R(1,2) < 0
• Above the front axle suspension hardpoint	• z2R(1,3) > 0

Inertia tensor, z2I — Inertia
[1.4,-.2,.1;-.2,1.4,.1;.1,.1,2.25].*0 (default) | array

Inertia tensor, *z2I*, in kg·m^2. Dimensions are [3-by-3].

$$z2I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

The tensor uses a coordinate system with an origin at the load CM.

- x-axis along the vehicle-fixed x-axis
- y-axis along the vehicle-fixed y-axis
- z-axis along the vehicle-fixed z-axis
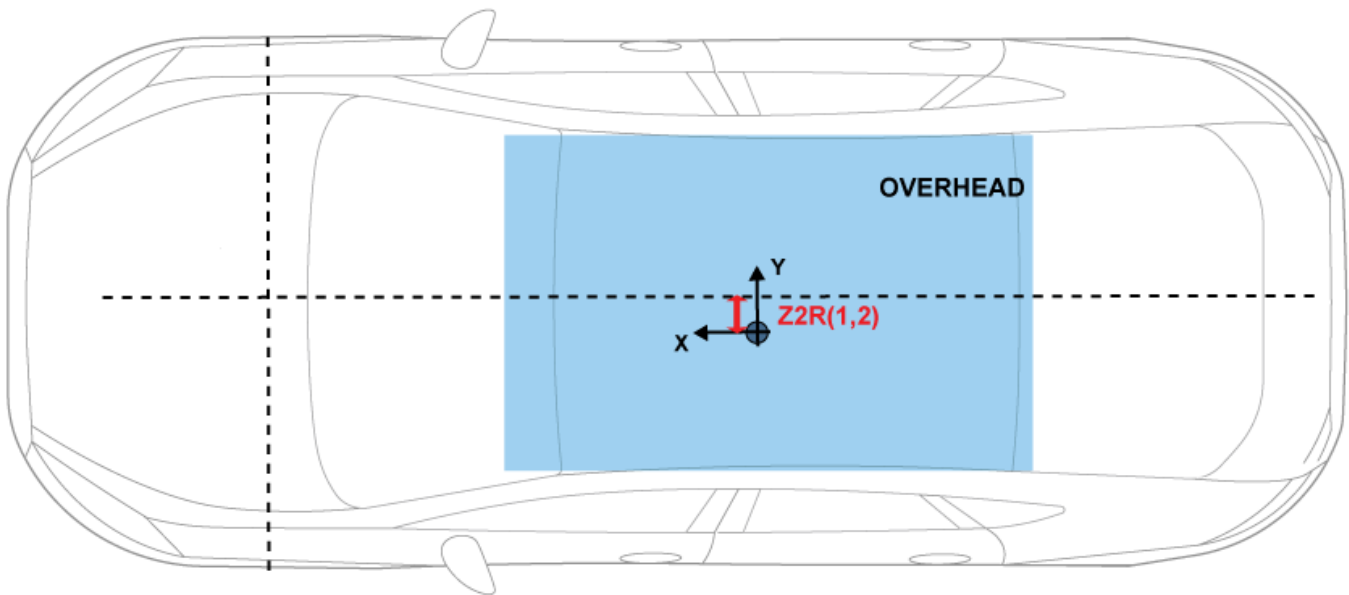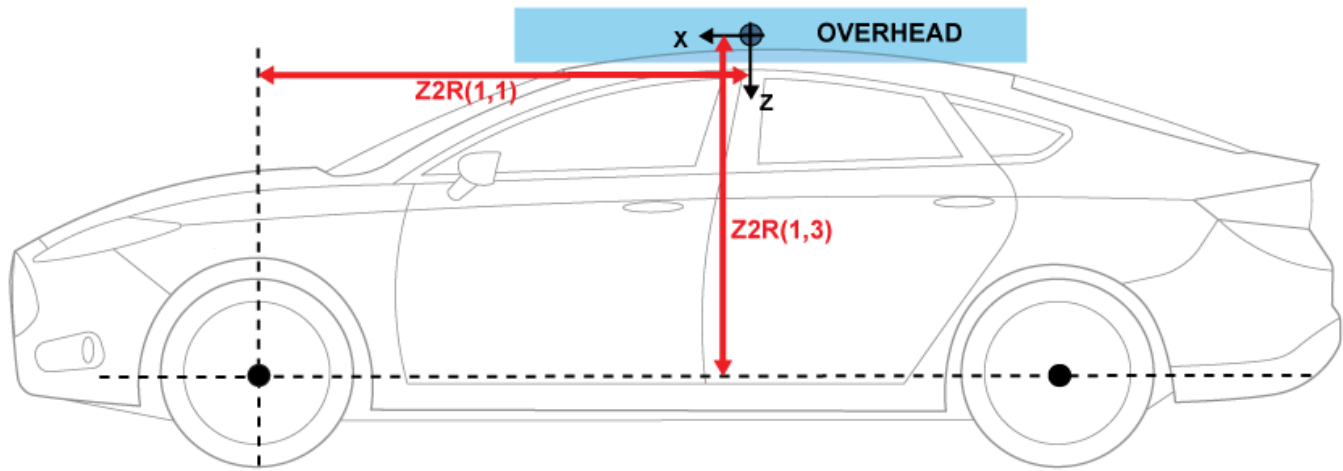
Row 1, left side

Mass, z3m — Mass
0 (default) | scalar

Mass, *z3m*, in kg.

Distance vector from front axle, z3R — Distance
[.75,-.5,.4] (default) | vector

Distance vector from front axle to load, *z3R*, in m. Dimensions are [1-by-3].

Array Element	Description
z3R(1,1)	Front suspension hardpoint to load, along vehicle-fixed *x*-axis
z3R(1,2)	Vehicle centerline to load, along vehicle-fixed *y*-axis
z3R(1,3)	Front suspension hardpoint to load, along vehicle-fixed *z*-axis
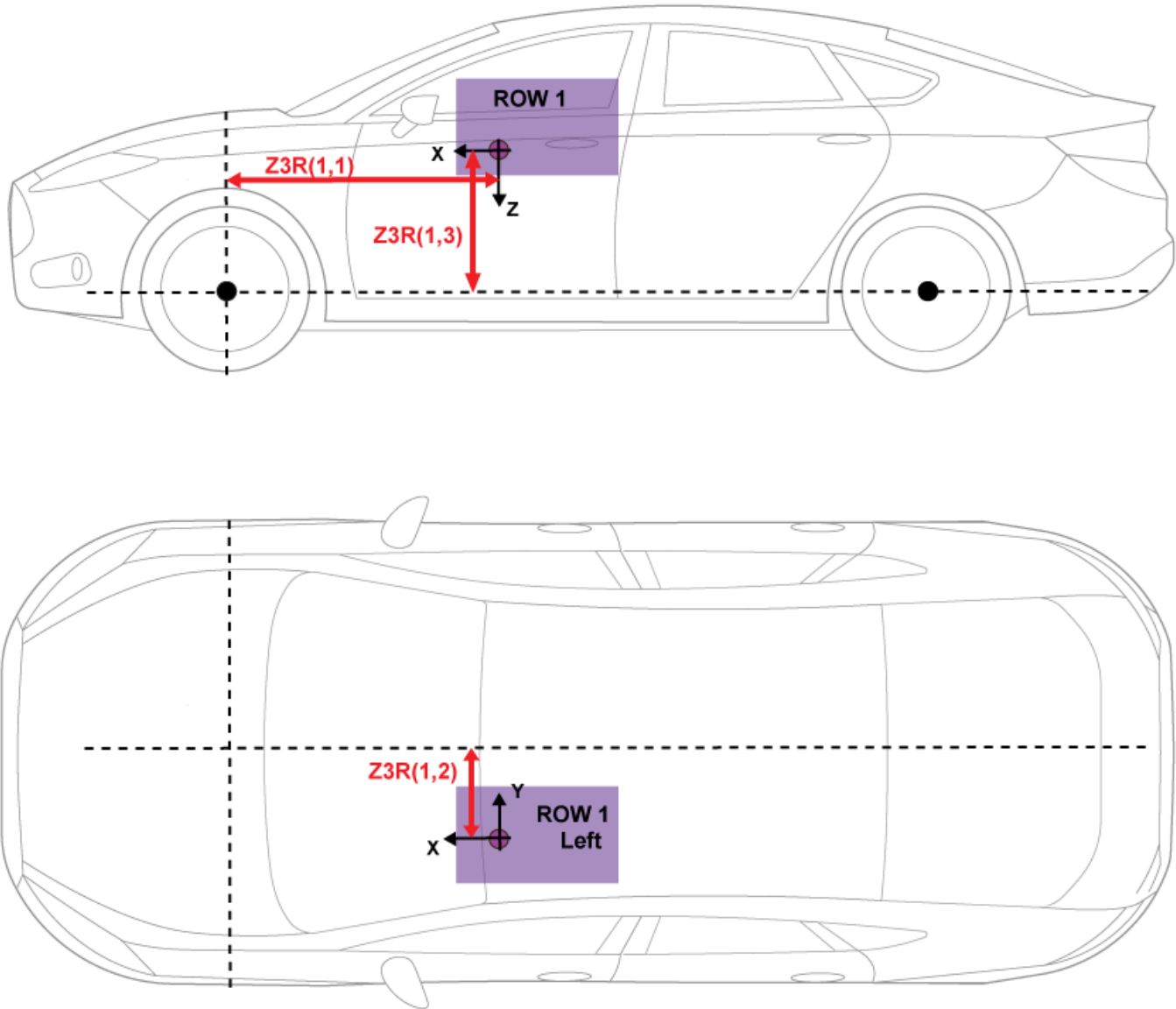
For example, this table summarizes the parameter settings that specify the load location indicated by the dot.

Example Location	Sign
• Rear of the front axle	• z3R(1,1) > 0
• Left of the vehicle centerline	• z3R(1,2) < 0
• Above the front axle suspension hardpoint	• z3R(1,3) > 0

Inertia tensor, z3I — Inertia
[5,-.1,-2;-2,9,.1;-.1,.1,6].*0 (default) | array

Inertia tensor, *z3I*, in kg·m^2. Dimensions are [3-by-3].

$$z3I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

The tensor uses a coordinate system with an origin at the load CM.

- *x*-axis along the vehicle-fixed *x*-axis
- *y*-axis along the vehicle-fixed *y*-axis
- *z*-axis along the vehicle-fixed *z*-axis
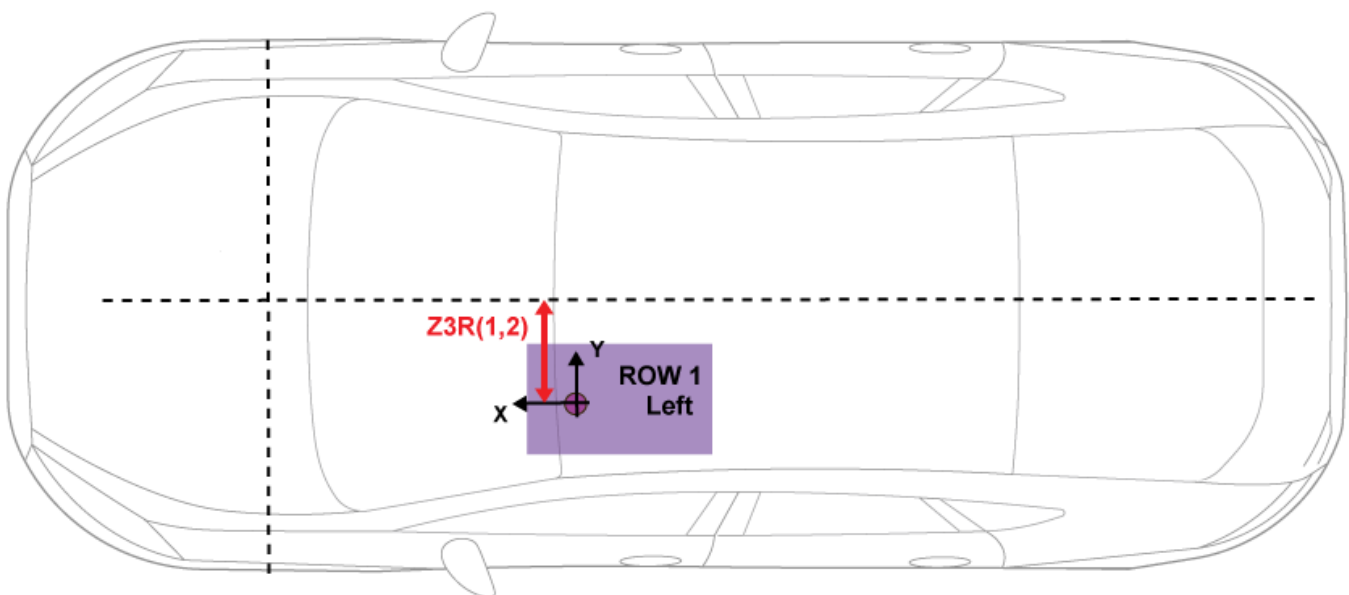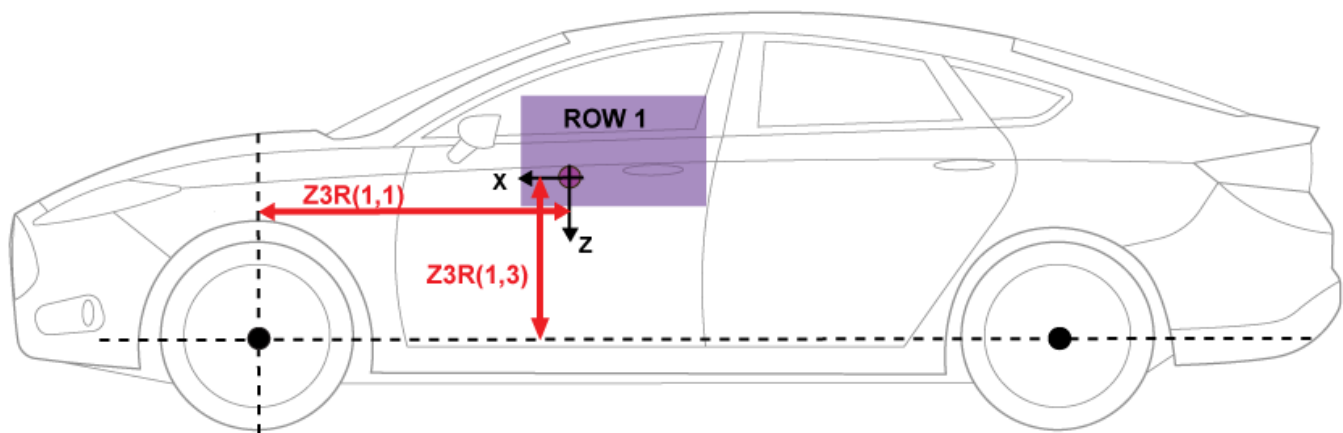
Row 1, right side

Mass, z4m — Mass
0 (default) | scalar

Mass, $z4m$, in kg.

Distance vector from front axle, z4R — Distance
[.75,.5,.4] (default) | vector

Distance vector from front axle to load, $z4R$, in m. Dimensions are [1-by-3].

Array Element	Description
z4R(1,1)	Front suspension hardpoint to load, along vehicle-fixed x-axis
z4R(1,2)	Vehicle centerline to load, along vehicle-fixed y-axis
z4R(1,3)	Front suspension hardpoint to load, along vehicle-fixed z-axis
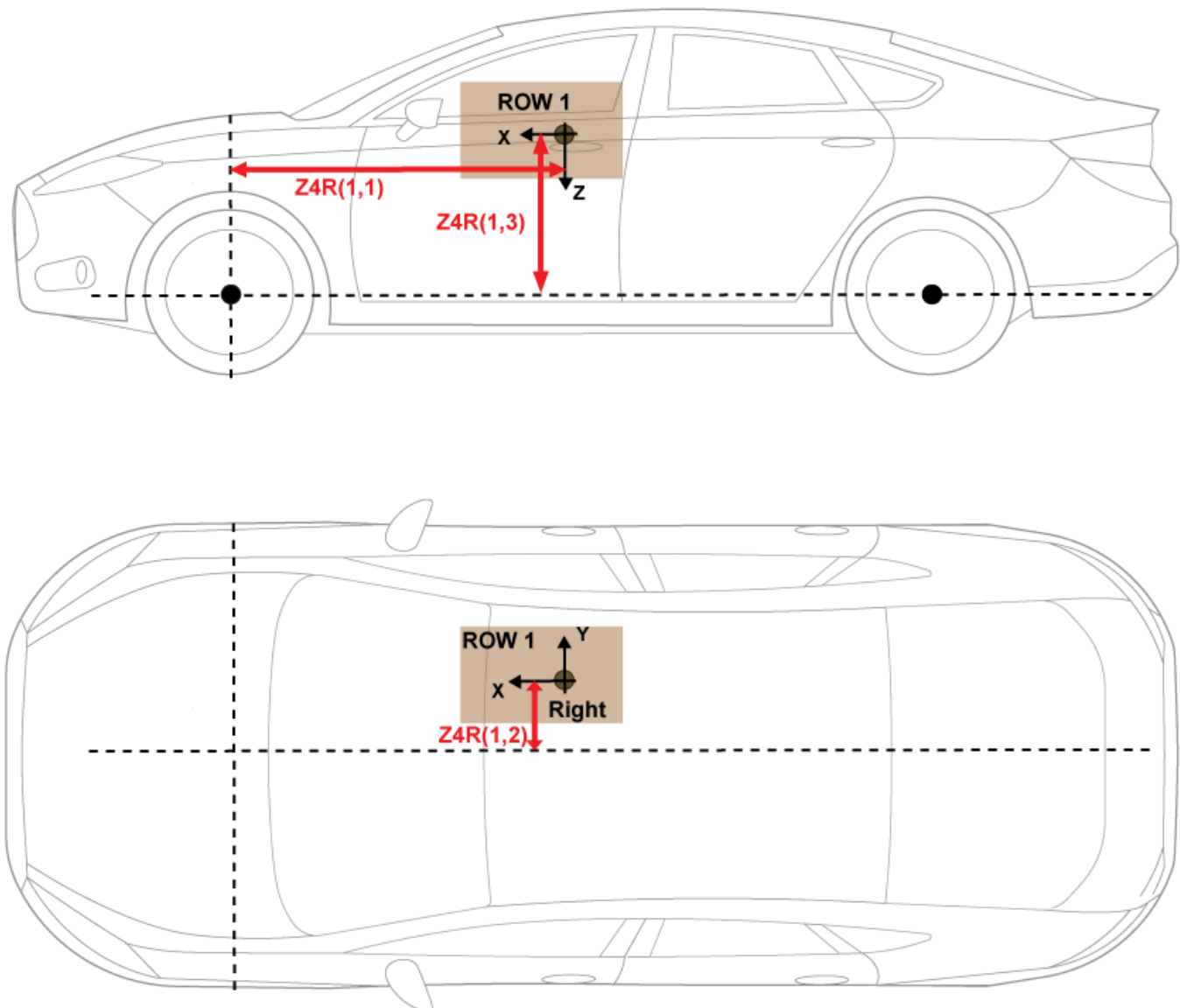
For example, this table summarizes the parameter settings that specify the load location indicated by the dot.

Example Location	Sign
• Rear of the front axle	• z4R(1,1) > 0
• Right of the vehicle centerline	• z4R(1,2) > 0
• Above the front axle suspension hardpoint	• z4R(1,3) > 0

Inertia tensor, z4I — Inertia
[5,-.1,-2;-2,9,.1;-.1,.1,6].*0 (default) | array

Inertia tensor, *z4I*, in kg·m^2. Dimensions are [3-by-3].

$$z4I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

The tensor uses a coordinate system with an origin at the load CM.

- x-axis along the vehicle-fixed x-axis
- y-axis along the vehicle-fixed y-axis
- z-axis along the vehicle-fixed z-axis
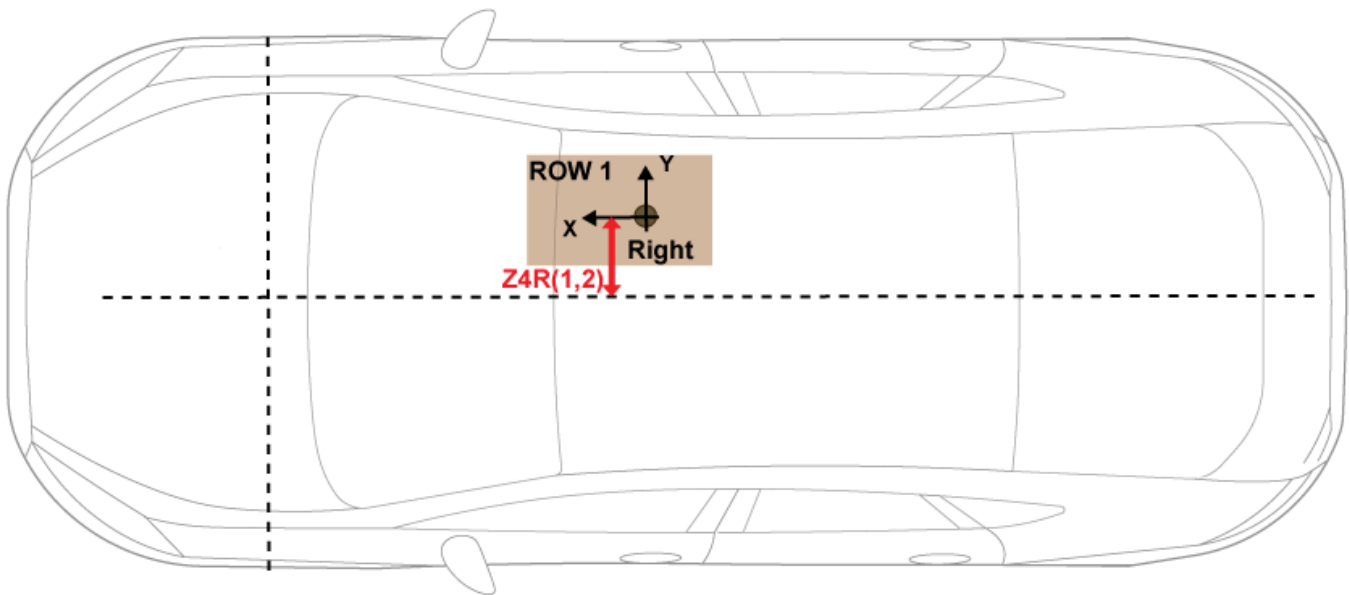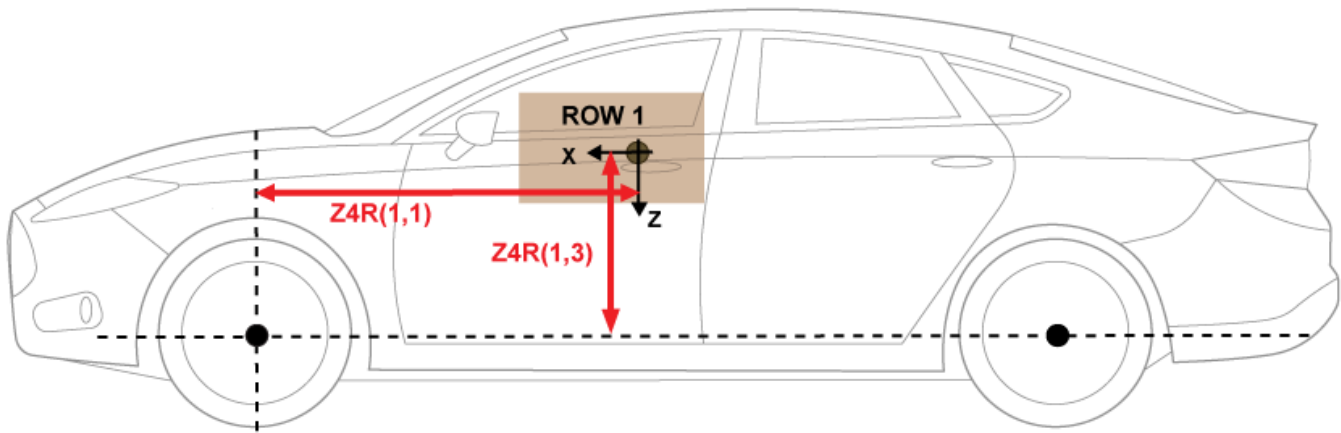
Row 2, left side

Mass, z5m — Mass
0 (default) | scalar

Mass, z5m, in kg.

Distance vector from front axle, z5R — Distance
[1.25,-.5,.4] (default) | vector

Distance vector from front axle to load, *z5R*, in m. Dimensions are [1-by-3].

Array Element	Description
z5R(1,1)	Front suspension hardpoint to load, along vehicle-fixed *x*-axis
z5R(1,2)	Vehicle centerline to load, along vehicle-fixed *y*-axis
z5R(1,3)	Front suspension hardpoint to load, along vehicle-fixed *z*-axis
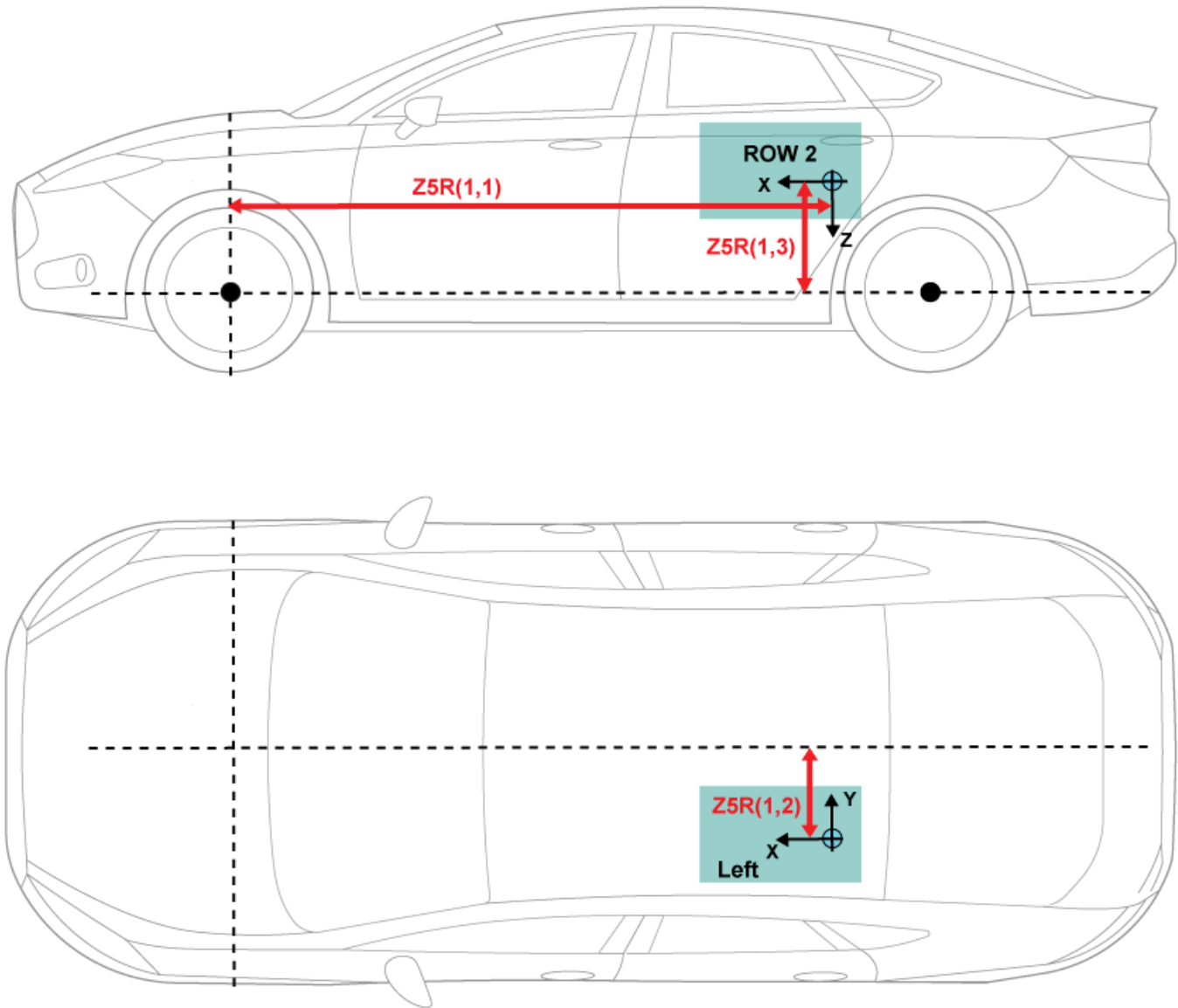
For example, this table summarizes the parameter settings that specify the load location indicated by the dot.

Example Location	Sign
• Rear of the front axle • Left of the vehicle centerline • Above the front axle suspension hardpoint	• `z5R(1,1) > 0` • `z5R(1,2) < 0` • `z5R(1,3) > 0`

Inertia tensor, z5I — Inertia
`[5,-.1,-2;-2,9,.1;-.1,.1,6].*0` (default) | `array`

Inertia tensor, *z5I*, in kg·m^2. Dimensions are `[3-by-3]`.

$$z5I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

The tensor uses a coordinate system with an origin at the load CM.

- x-axis along the vehicle-fixed x-axis
- y-axis along the vehicle-fixed y-axis
- z-axis along the vehicle-fixed z-axis
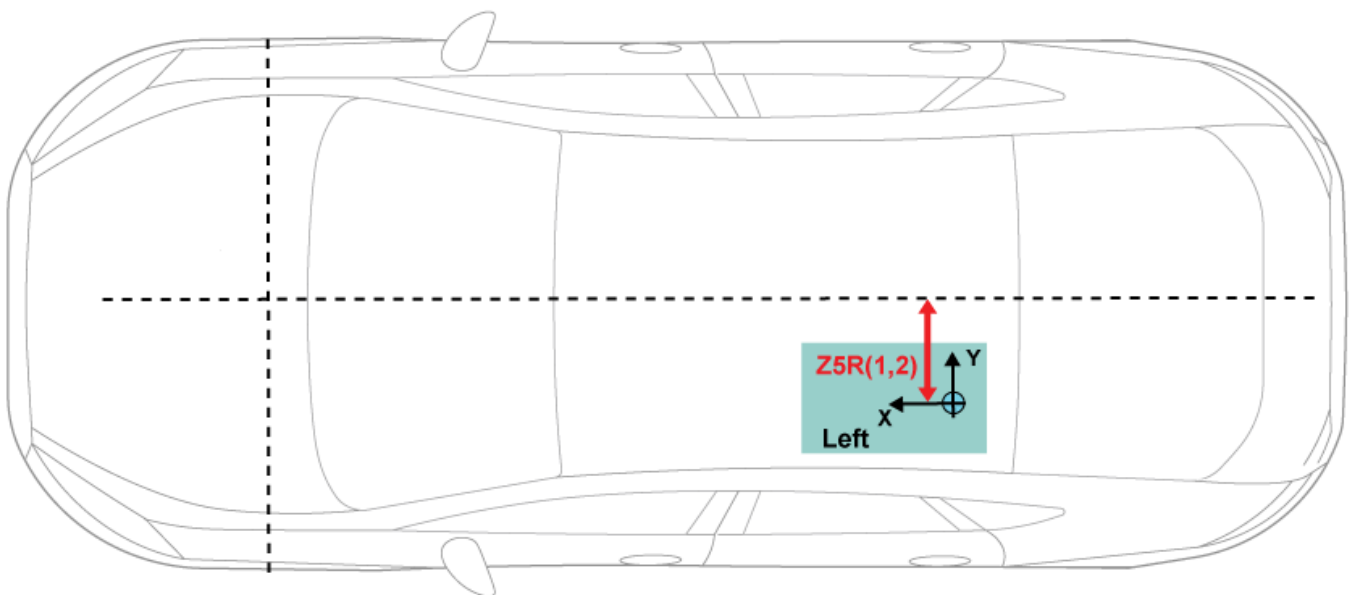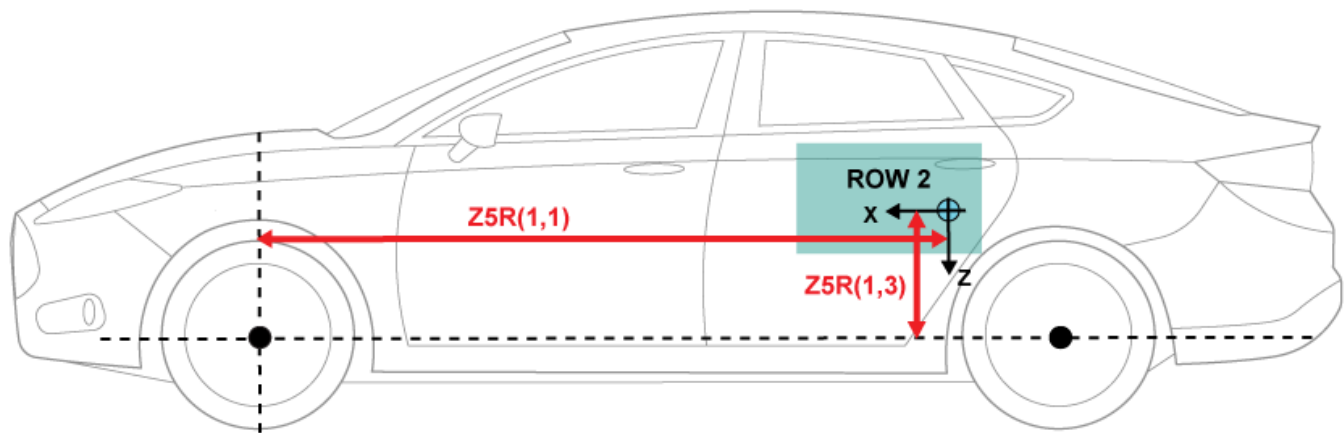
Row 2, right side

Mass, z6m — Mass
0 (default) | scalar

Mass, *z6m*, in kg.

Distance vector from front axle, z6R — Distance
[1.25,-.5,.4] (default) | vector

Distance vector from front axle to load, *z6R*, in m. Dimensions are [1-by-3].

Array Element	Description
z6R(1,1)	Front suspension hardpoint to load, along vehicle-fixed *x*-axis
z6R(1,2)	Vehicle centerline to load, along vehicle-fixed *y*-axis
z6R(1,3)	Front suspension hardpoint to load, along vehicle-fixed *z*-axis
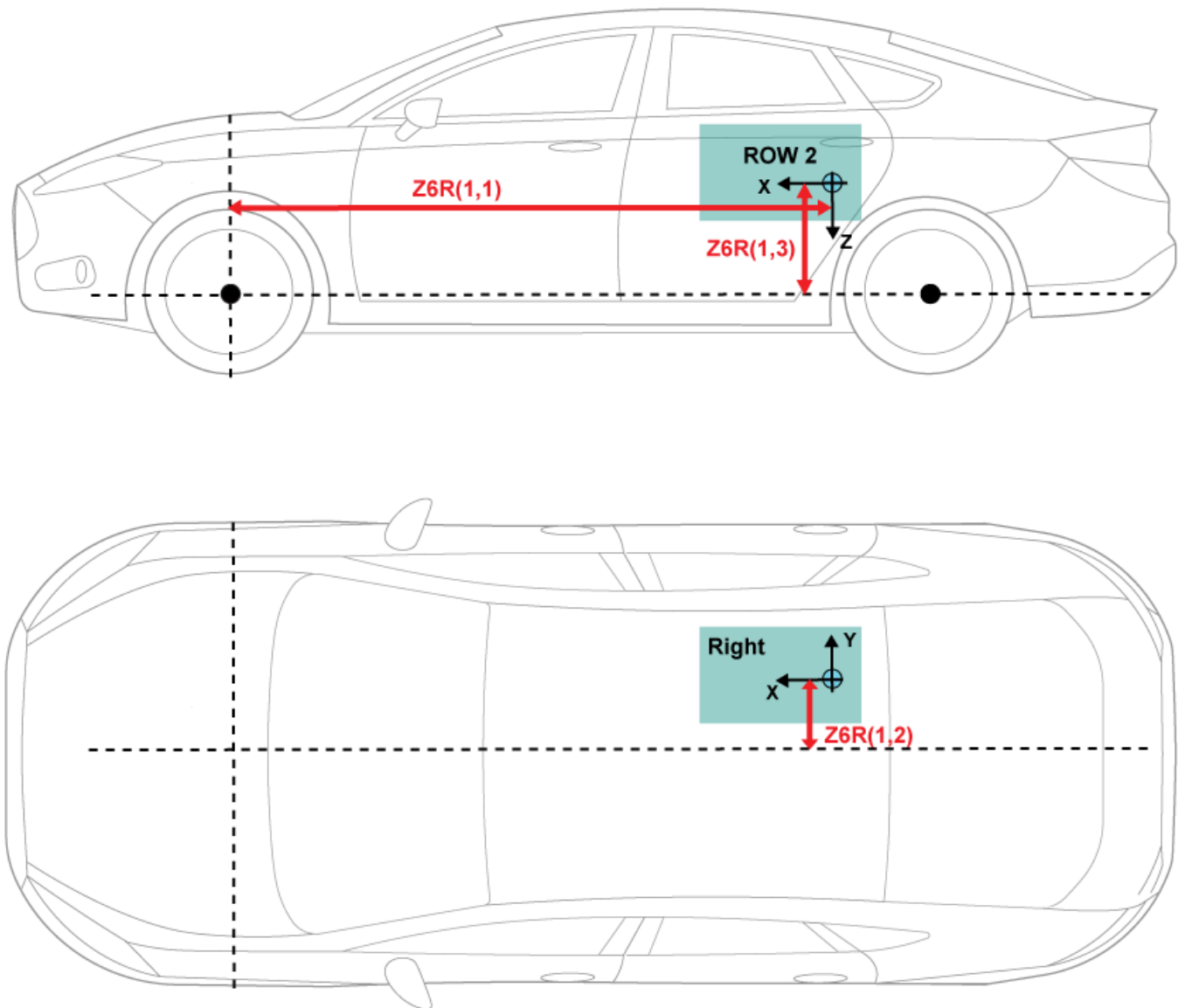
For example, this table summarizes the parameter settings that specify the load location indicated by the dot.

Example Location	Sign
• Rear of the front axle	• z6R(1,1) > 0
• Right of the vehicle centerline	• z6R(1,2) > 0
• Above the front axle suspension hardpoint	• z6R(1,3) > 0

Inertia tensor, z6I — Inertia
[5,-.1,-2;-2,9,.1;-.1,.1,6].*0 (default) | array

Inertia tensor, *z6I*, in kg·m^2. Dimensions are [3-by-3].

$$
z6I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}
$$

The tensor uses a coordinate system with an origin at the load CM.

- x-axis along the vehicle-fixed x-axis
- y-axis along the vehicle-fixed y-axis
- z-axis along the vehicle-fixed z-axis
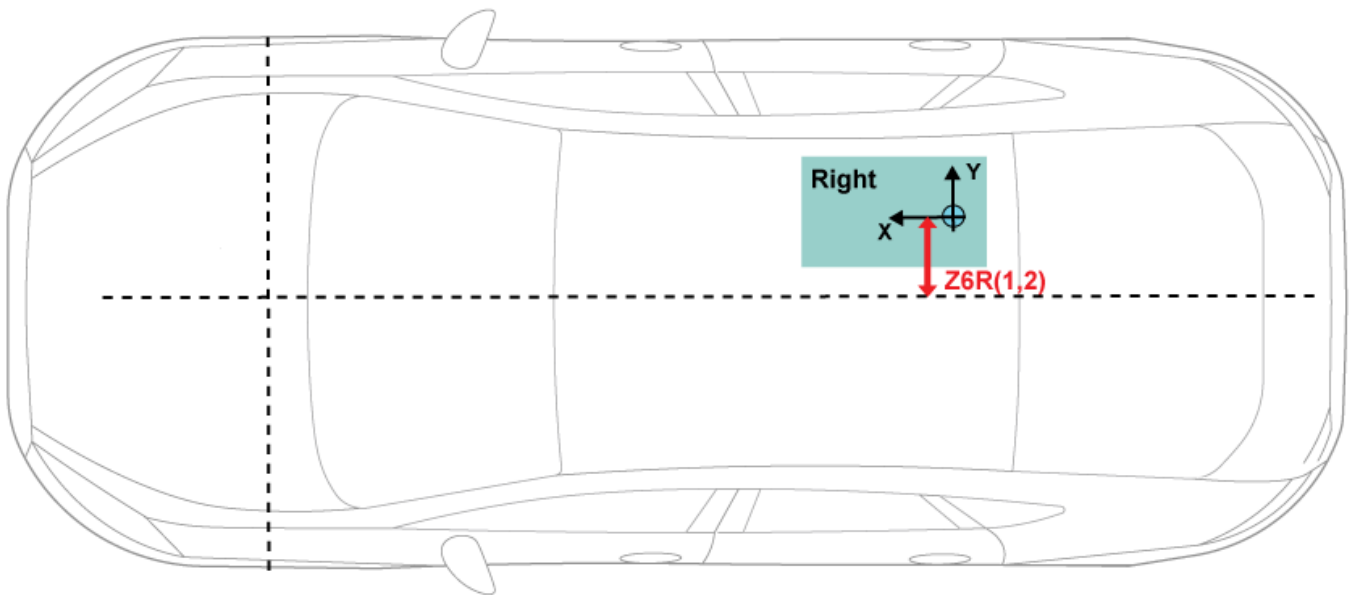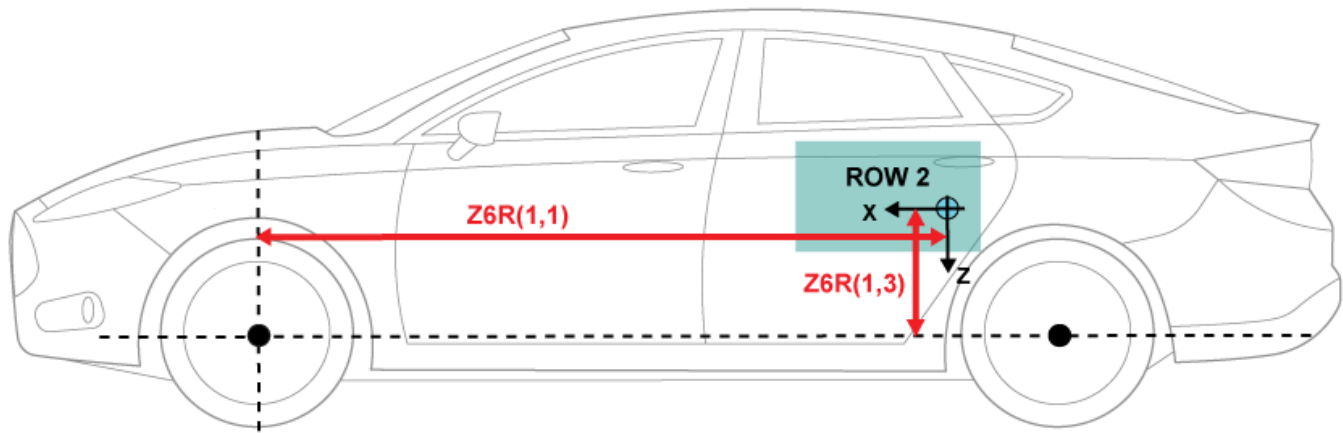
Rear

Mass, z7m — Mass
0 (default) | scalar

Mass, $z7m$, in kg.

Distance vector from front axle, z7R — Distance
[2,0,.25] (default) | vector

Distance vector from front axle to load, $z7R$, in m. Dimensions are [1-by-3].

Array Element	Description
z7R(1,1)	Front suspension hardpoint to load, along vehicle-fixed x-axis
z7R(1,2)	Vehicle centerline to load, along vehicle-fixed y-axis
z7R(1,3)	Front suspension hardpoint to load, along vehicle-fixed z-axis
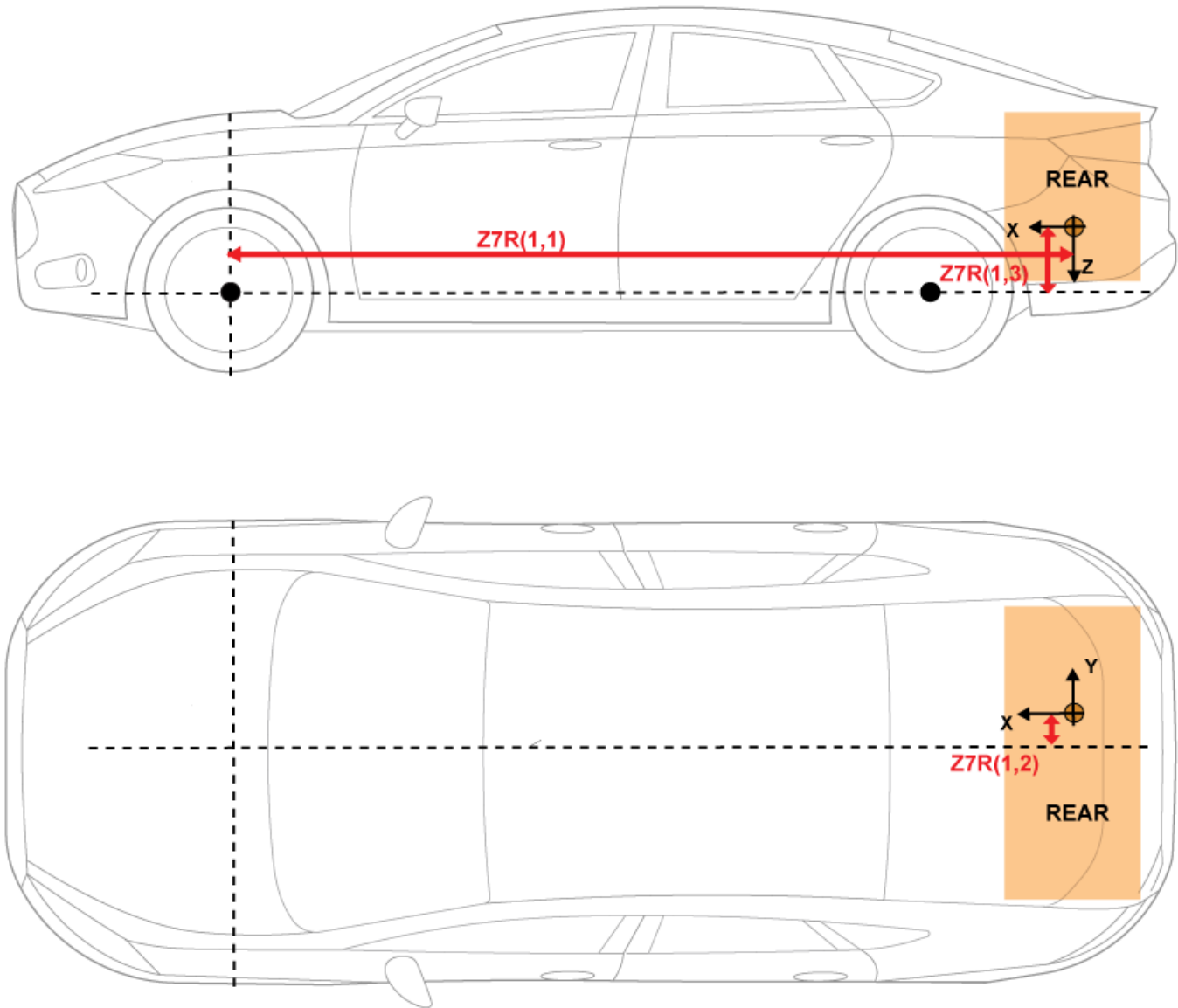
For example, this table summarizes the parameter settings that specify the load location indicated by the dot.

Example Location	Sign
• Rear of the front axle • Right of the vehicle centerline • Above the front axle suspension hardpoint	• z7R(1,1) > 0 • z7R(1,2) > 0 • z7R(1,3) > 0

Inertia tensor, z7I — Inertia
[1.4,-.2,.1;-.2,1.4,.1;.1,.1,2.25].*0 (default) | array

Inertia tensor, *z7I*, in kg·m^2. Dimensions are [3-by-3].

$$z7I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

The tensor uses a coordinate system with an origin at the load CM.

- x-axis along the vehicle-fixed x-axis
- y-axis along the vehicle-fixed y-axis
- z-axis along the vehicle-fixed z-axis
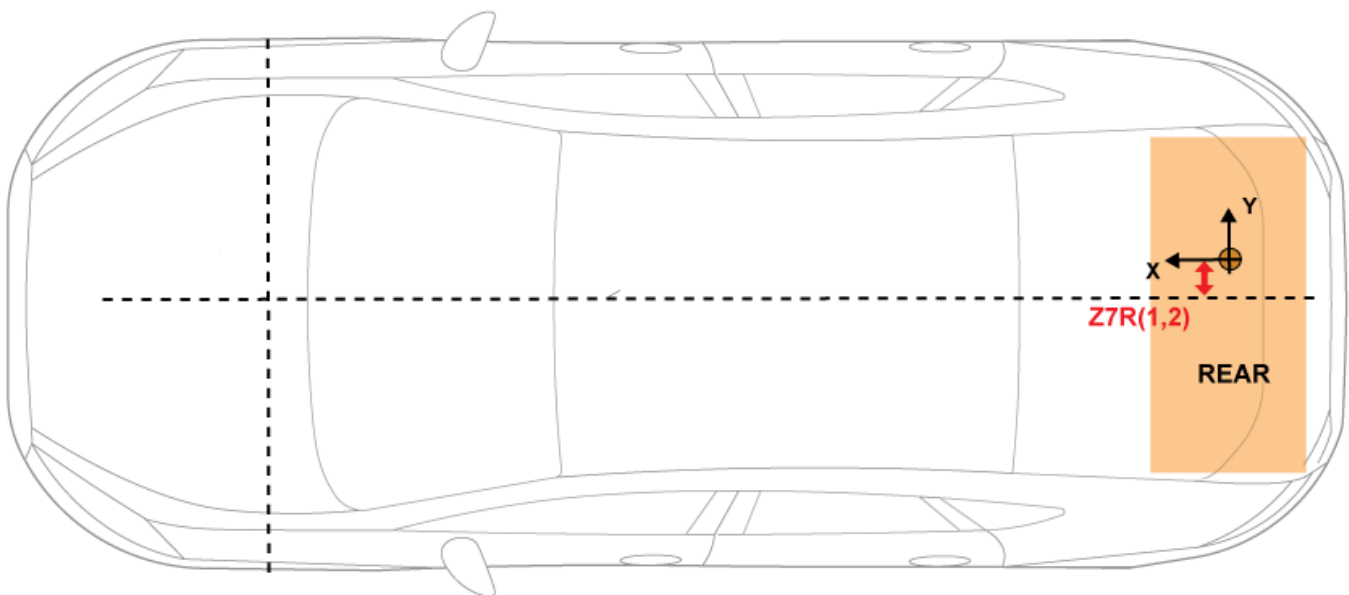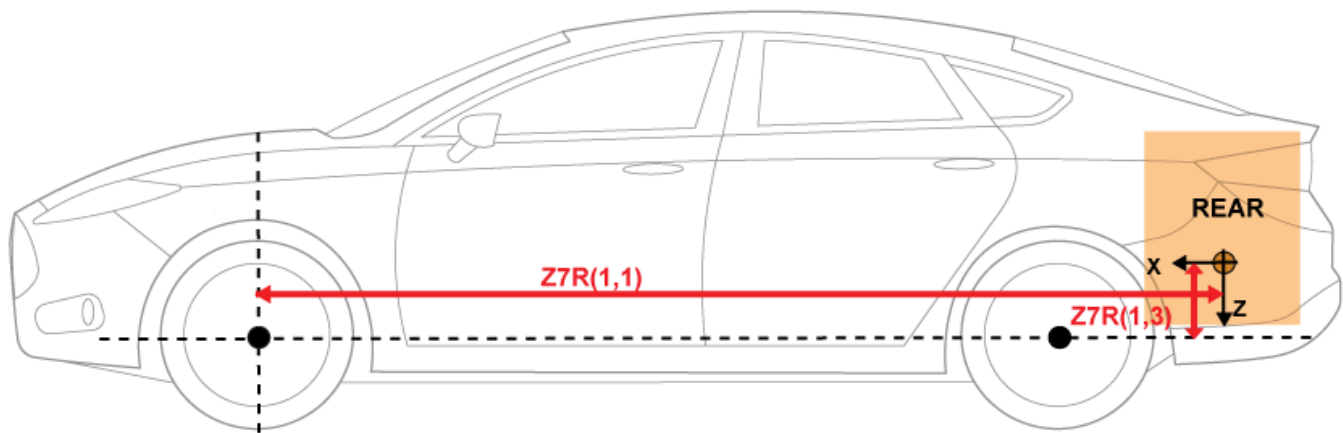
Aerodynamic

Longitudinal drag area, Af — Area
2 (default) | scalar

Effective vehicle cross-sectional area, A_f to calculate the aerodynamic drag force on the vehicle, in m^2.

Longitudinal drag coefficient, Cd — Drag
.3 (default) | scalar

Air drag coefficient, C_d, dimensionless.

Longitudinal lift coefficient, Cl — Lift
.1 (default) | scalar

Air lift coefficient, C_l, dimensionless.

Longitudinal drag pitch moment, Cpm — Pitch drag
.1 (default) | scalar

Longitudinal drag pitch moment coefficient, C_{pm}, dimensionless.

Relative wind angle vector, beta_w — Wind angle
[0:0.001:0.01] (default) | vector

Relative wind angle vector, β_w, in rad.

Side force coefficient vector, Cs — Side force drag
[0:0.01:0.1] (default) | vector

Side force coefficient vector coefficient, C_s, dimensionless.

Yaw moment coefficient vector, Cym — Yaw moment drag
[0:0.001:0.01] (default) | vector

Yaw moment coefficient vector coefficient, C_{ym}, dimensionless.

Environment

Absolute air pressure, Pabs — Pressure
101325 (default) | scalar

Environmental air absolute pressure, P_{abs}, in Pa.

Air temperature, Tair — Ambient air temperature
273 (default) | scalar

Ambient air temperature, T_{air}, in K.

Dependencies

To enable this parameter, clear **Air temperature**.

Gravitational acceleration, g — Gravity
9.81 (default) | scalar

Gravitational acceleration, *g*, in m/s^2.

Simulation

Longitudinal velocity tolerance, xdot_tol — Tolerance
.1 (default) | scalar

Longitudinal velocity tolerance, $xdot_{tol}$, in m/s.

The block uses this parameter to avoid a division by zero when it calculates the body slip angle, *β*.

Geometric longitudinal offset from axle plane, longOff — Longitudinal offset
0 (default) | scalar

Vehicle chassis offset from axle plane along body-fixed *x*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

Geometric lateral offset from center plane, latOff — Lateral offset
0 (default) | scalar

Vehicle chassis offset from center plane along body-fixed *y*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

Geometric vertical offset from axle plane, vertOff — Vertical offset
0 (default) | scalar

Vehicle chassis offset from axle plane along body-fixed *z*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

Wrap Euler angles, wrapAng — Selection
on (default) | off

Wrap the Euler angles to the interval [-pi, pi]. For vehicle maneuvers that might undergo vehicle yaw rotations that are outside of the interval, consider deselecting the parameter if you want to:

- Track the total vehicle yaw rotation.
- Avoid discontinuities in the vehicle state estimators.

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers (SAE), 1992.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

6DOF (Euler Angles) | Vector Concatenate, Matrix Concatenate | Vehicle Body 3DOF

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2018a

Trailer Body 3DOF

Trailer body with longitudinal, lateral, and yaw motion
Library: Vehicle Dynamics Blockset / Vehicle Body
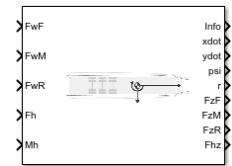
Description

The Trailer Body 3DOF block implements a rigid two-axle or three-axle trailer body model to calculate longitudinal, lateral, and yaw motion. Configure the block for a single or dual track. The block accounts for axle and hitch reaction forces due to the trailer acceleration, aerodynamic drag, and steering.

Use this block in vehicle dynamics and automated driving studies to model nonholonomic vehicle motion when vehicle pitch, roll, and vertical motion are not significant.

Use the **Vehicle track** parameter to specify the number of tracks.

Vehicle Track Setting	Implementation
Single 2-axle	Trailer with a single track and two axles. • Forces act along the center line of the axles. • No lateral load transfer.
Dual 2-axle	Trailer with a dual track and two axles. Forces act at the axle hard-point locations.
Single 3-axle	Trailer with a single track and three axles. • Forces act along the center line of the axles. • No lateral load transfer.
Dual 3-axle (default)	Trailer with a dual track and three axles. Forces act at the axle hard-point locations.

Use the **Axle forces** parameter to specify the type of force.

Axle Forces Setting	Implementation

`External longitudinal velocity`	• The block assumes that the external longitudinal velocity is in a quasi-steady state, and the longitudinal acceleration is approximately zero.
	
	• Because the motion is quasi-steady, the block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	
	• Consider this setting when you want to:
	
	   • Generate virtual sensor signal data.
	
	   • Conduct high-level software studies that are not impacted by driveline or nonlinear tire responses.

`External longitudinal forces`	• The block uses the external longitudinal force to accelerate or brake the vehicle.
	
	• The block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	
	• Consider this setting when you want to:
	
	   • Account for changes in the longitudinal velocity on the lateral and yaw motion.
	
	   • Specify the external longitudinal motion through a force instead of an external longitudinal velocity.
	
	   • Connect the block to tractive actuators, wheels, brakes, and hitches.

`External forces`	• The block uses the external lateral and longitudinal forces to steer, accelerate, or brake the vehicle. • The block does not use the steering input to calculate vehicle motion. • Consider this setting when you need tire models with more accurate nonlinear combined lateral and longitudinal slip.

To create additional input ports, under **Input signals**, select these block parameters.

Input Signals Pane Parameter	Input Port	Description
Front wheel steering	`WhlAngF`	Front wheel angle, δ_F
Middle wheel steering	`WhlAngM`	Middle wheel angle, δ_M
Rear wheel steering	`WhlAngR`	Rear wheel angle, δ_R
External wind	`WindXYZ`	Wind speed, W_X, W_Y, and W_Z, in an inertial reference frame
External friction	`Mu`	Friction coefficient
External forces	`FExt`	External force on the vehicle center of gravity (CG), F_x, F_y, and F_z, in the vehicle-fixed frame
External moments	`MExt`	External moment about the vehicle CG, M_x, M_y, and M_z, in the vehicle-fixed frame
Hitch forces	`Fh`	Hitch force applied to the body at the hitch location, Fh_x, Fh_y, and Fh_z, in the vehicle-fixed frame

Input Signals Pane Parameter	Input Port	Description
Hitch moments	Mh	Hitch moment at the hitch location, Mh_x, Mh_y, and Mh_z, about the vehicle-fixed frame
Initial longitudinal position	X_o	Initial vehicle CG displacement along the earth-fixed X-axis
Initial yaw angle	psi_o	Initial rotation of the vehicle-fixed frame about the earth-fixed Z-axis (yaw)
Initial longitudinal velocity	xdot_o	Initial vehicle CG velocity along the vehicle-fixed x-axis
Initial yaw rate	r_o	Initial vehicle angular velocity about the vehicle-fixed z-axis (yaw rate)
Initial lateral position	Y_o	Initial vehicle CG displacement along the earth-fixed Y-axis
Air temperature	AirTemp	Ambient air temperature. Consider this option if you want to vary the temperature during run time.
Initial lateral velocity	ydot_o	Initial vehicle CG velocity along the vehicle-fixed y-axis

Theory

To determine the vehicle motion, the block solves the rigid body planar dynamics equations of motion.

Calculation	Description
Dynamics	The block solves the rigid-body planar dynamics equations to determine the vehicle longitudinal motion. If you set **Axle forces** to External longitudinal velocity, the block assumes a quasi-steady state for the longitudinal acceleration.
External forces	External forces include both drag and external force inputs. The forces act on the vehicle CG. The block divides the normal forces by the nominal normal load to vary the effective friction parameters during weight and load transfer. The block maintains pitch and roll equilibrium.
Tire forces	The block uses the ratio of the local, longitudinal, and lateral velocities to determine the slip angles. The block uses the steering angles to transform the tire forces to the vehicle-fixed frame. If you set **Axle forces** to External forces, the block assumes that the externally provided forces are in the vehicle-fixed frame at the axle-wheel location.

Single Track — Three Axles
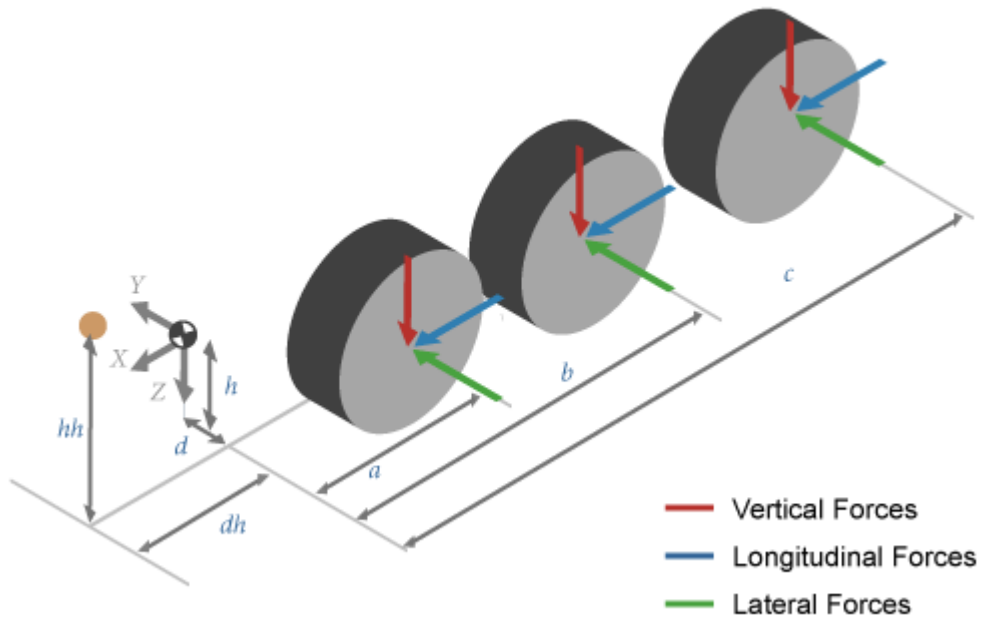
Single Track — Two Axles
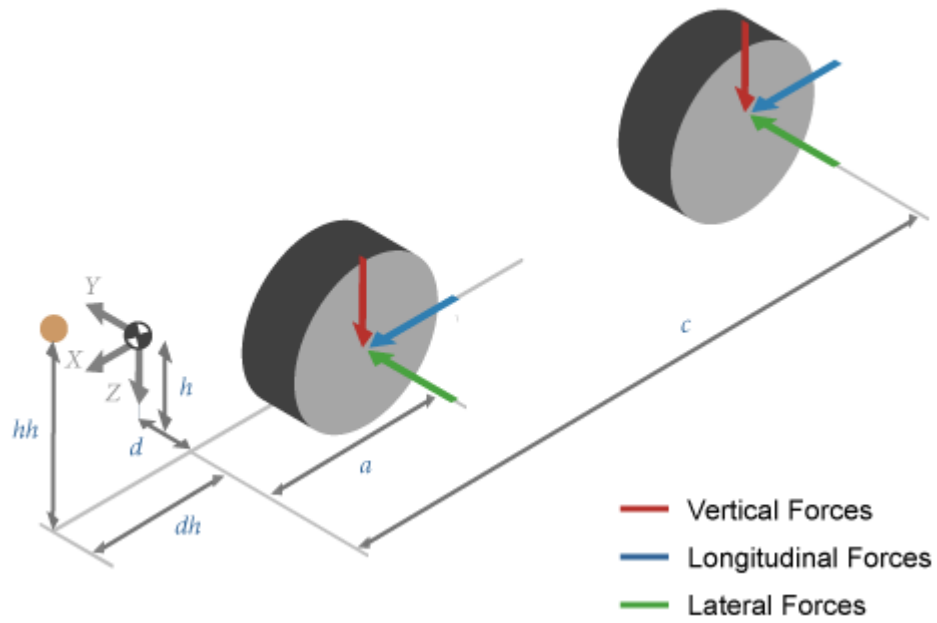
Dual Track — Three Axles
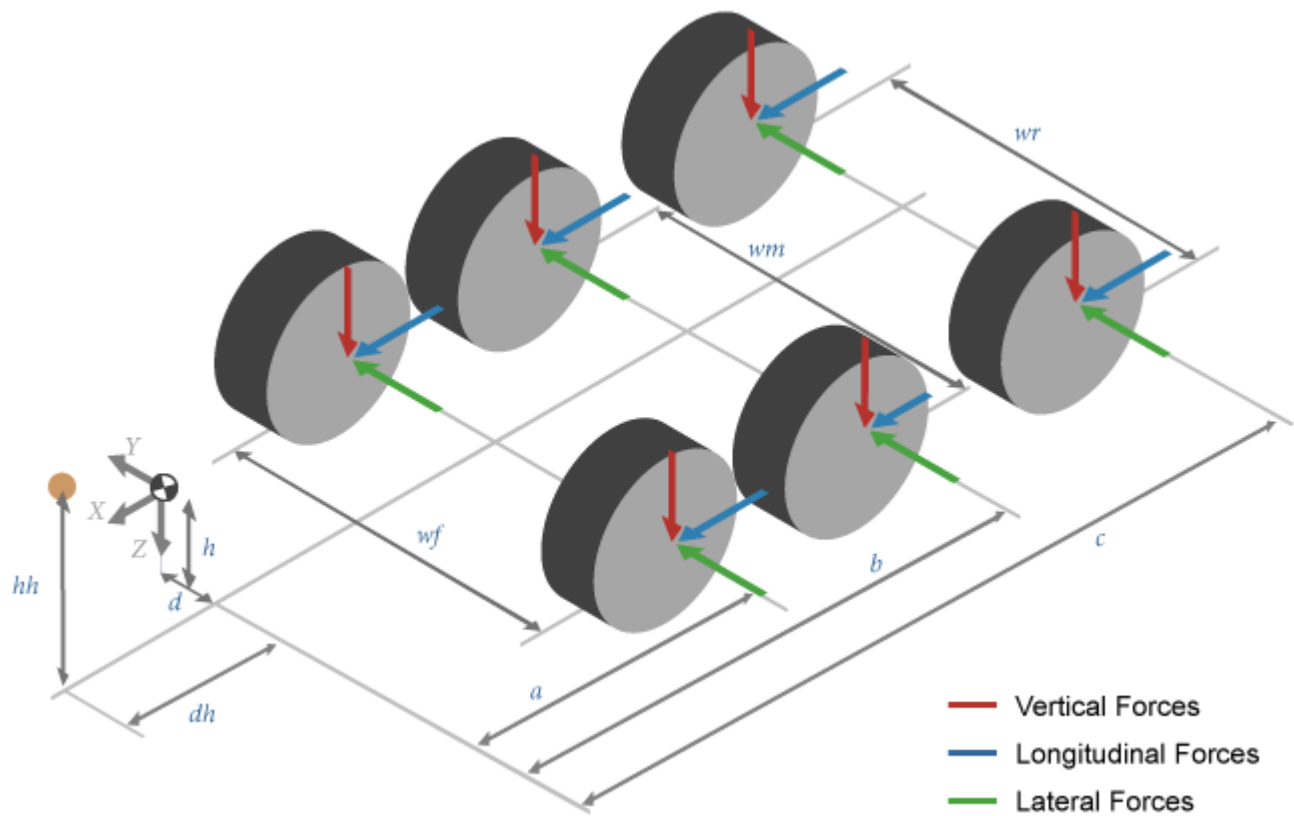
Dual Track — Two Axles
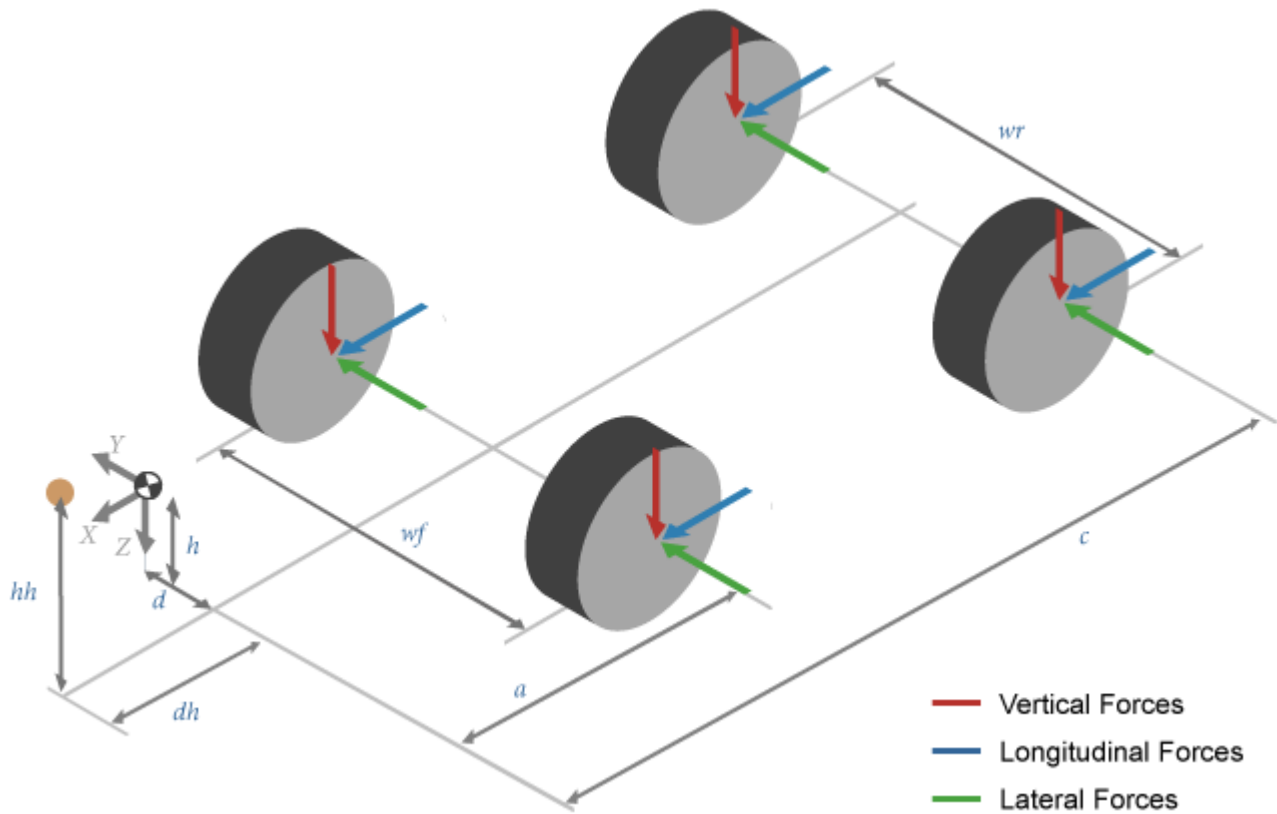
The illustrations use these variables.

a, b, c	Longitudinal distance of the front, middle, and rear axles, respectively, from the normal projection point of the vehicle CG onto the common axle plane
h	Height of the tractor CG above the axle plane along the vehicle-fixed z-axis
d	Lateral distance from the geometric centerline to the center of mass along the vehicle-fixed y-axis
hh	Height of the hitch above the axle plane along the vehicle-fixed z-axis
dh	Longitudinal distance of the hitch from the normal projection point of tractor CG onto the common axle plane
wf, wm, wr	Front, middle, and rear track width, respectively

Drag

This table summarizes the block implementation for the drag calculation.

Calculation	Description
Coordinate transformation	The block transforms the wind speeds from the inertial frame to the vehicle-fixed frame.

Calculation	Description
Drag forces	To determine a relative airspeed, the block subtracts the wind speed from the CG vehicle velocity. Using the relative airspeed, the block determines the drag forces.
Drag moments	Using the relative airspeed, the block determines the drag moments.

Lateral Corner Stiffness and Relaxation Dynamics

To enable the mapped corner stiffness and relaxation length dynamic parameters, set **Axle forces** to `External longitudinal forces` or `External longitudinal velocity`.

Parameter Settings		Description
Mapped Corner Stiffness	**Include Relaxation Length Dynamics**	
`Off` (default)	`On` (default)	The block uses constant corner stiffness values. The slip angles include the relaxation length dynamic settings. The relaxation length approximates an effective corner stiffness force that is a function of wheel travel.
`On`	`On` (default)	The block uses lookup tables that are functions of the corner stiffness data and slip angles. The slip angles include the relaxation length dynamic settings. The relaxation length approximates an effective corner stiffness force that is a function of wheel travel.
`Off` (default)	`Off`	The block uses constant corner stiffness values.

Ports

Input

WhlAngF — Front wheel steering angles
scalar | array

Front wheel steering angles, δ_F, in rad.

Vehicle Track Setting	Variable	Signal Dimension
`Single 2-axle` `Single 3-axle`	δ_F	Scalar – 1
`Dual 2-axle` `Dual 3-axle`	$\delta_F = [\delta_{fl} \ \delta_{fr}]$ or $\begin{bmatrix} \delta_{fl} \\ \delta_{fr} \end{bmatrix}$	Array – [1x2] or [2x1]

Dependencies

To enable this port, under **Input signals**, select **Front wheel steering**.

WhlAngM — Middle wheel steering angles
scalar | array

Middle wheel steering angles, δ_M, in rad.

Vehicle Track Setting	Variable	Signal Dimension
Single 3-axle	δ_M	Scalar – 1
Dual 3-axle	$\delta_M = [\delta_{ml}\ \delta_{mr}]$ or $\begin{bmatrix} \delta_{ml} \\ \delta_{mr} \end{bmatrix}$	Array – [1x2] or [2x1]

Dependencies

To enable this port:

- Set **Vehicle track** to Single 3-axle or Dual 3-axle.
- To enable this port, under **Input signals**, select **Middle wheel steering**.

WhlAngR — Rear wheel steering angles
scalar | array

Rear wheel steering angles, δ_R, in rad.

Vehicle Track Setting	Variable	Signal Dimension
Single 2-axle Single 3-axle	δ_R	Scalar – 1
Dual 2-axle Dual 3-axle	$\delta_R = [\delta_{rl}\ \delta_{rr}]$ or $\begin{bmatrix} \delta_{rl} \\ \delta_{rr} \end{bmatrix}$	Array – [1x2] or [2x1]

Dependencies

To enable this port, under **Input signals**, select **Rear wheel steering**.

xdotin — Longitudinal velocity
scalar

Vehicle CG velocity along the vehicle-fixed *x*-axis, in m/s.

Dependencies

To enable this port, set **Axle forces** to External longitudinal velocity.

FwF — Total force on the front wheels
scalar | array

Force on the front wheels, Fw_F, along the vehicle-fixed axis, in N.

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Single 2-axle Single 3-axle	External longitudinal forces	Longitudinal force on the front wheel	$FwF = Fx_f$	Scalar – 1
	External forces	Longitudinal and lateral forces on the front wheel	$FwF = \begin{bmatrix} Fx_f & Fy_f \end{bmatrix}$ or $\begin{bmatrix} Fx_f \\ Fy_f \end{bmatrix}$	Array – [1x2] or [2x1]
Dual 2-axle Dual 3-axle	External longitudinal forces	Longitudinal force on the front wheels	$FwF = \begin{bmatrix} F_{xfl} & F_{xfr} \end{bmatrix}$ or $\begin{bmatrix} F_{xfl} \\ F_{xfr} \end{bmatrix}$	Array – [1x2] or [2x1]
	External forces	Longitudinal and lateral forces on the front wheels	$FwF = \begin{bmatrix} F_{xfl} & F_{xfr} \\ F_{yfl} & F_{yfr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, set **Axle forces** to one of these options:

- `External longitudinal forces`
- `External forces`

FwM — Total force on the middle wheels
scalar | array

Force on the middle wheels, Fw_M, along the vehicle-fixed axis, in N.

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Single 3-axle	External longitudinal forces	Longitudinal force on the middle wheel	$FwM = Fx_r$	Scalar – 1
	External forces	Longitudinal and lateral forces on the middle wheel	$FwM = \begin{bmatrix} Fx_m & Fy_m \end{bmatrix}$ or $\begin{bmatrix} Fx_m \\ Fy_m \end{bmatrix}$	Array – [1x2] or [2x1]
Dual 3-axle	External longitudinal forces	Longitudinal force on the middle wheels	$FwM = \begin{bmatrix} F_{xml} & F_{xmr} \end{bmatrix}$ or $\begin{bmatrix} F_{xml} \\ F_{xmr} \end{bmatrix}$	Array – [1x2] or [2x1]

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
	External forces	Longitudinal and lateral forces on the middle wheels	$FwM = \begin{bmatrix} F_{xml} & F_{xmr} \\ F_{yml} & F_{ymr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, set:

- **Vehicle track** to `Single 3-axle` or `Dual 3-axle`.
- **Axle forces** to `External longitudinal forces` or `External forces`.

FwR — Total force on the rear wheels
scalar | array

Force on the rear wheels, Fw_R, along the vehicle-fixed axis, in N.

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Single 2-axle Single 3-axle	External longitudinal forces	Longitudinal force on the rear wheel	$FwR = Fx_r$	Scalar – 1
	External forces	Longitudinal and lateral forces on the rear wheel	$FwR = \begin{bmatrix} Fx_r & Fy_r \end{bmatrix}$ or $\begin{bmatrix} Fx_r \\ Fy_r \end{bmatrix}$	Array – [1x2] or [2x1]
Dual 2-axle Dual 3-axle	External longitudinal forces	Longitudinal force on the rear wheels	$FwR = \begin{bmatrix} F_{xrl} & F_{xrr} \end{bmatrix}$ or $\begin{bmatrix} F_{xrl} \\ F_{xrr} \end{bmatrix}$	Array – [1x2] or [2x1]
	External forces	Longitudinal and lateral forces on the rear wheels	$FwR = \begin{bmatrix} F_{xrl} & F_{xrr} \\ F_{yrl} & F_{yrr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, set **Axle forces** to one of these options:

- `External longitudinal forces`
- `External forces`

FExt — External force on the vehicle CG
array

External forces applied to the vehicle CG, F_{xext}, F_{yext}, F_{zext}, in vehicle-fixed frame, in N. The signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, under **Input signals**, select **External forces**.

MExt — External moment about vehicle CG
array

External moment about the vehicle CG, M_x, M_y, M_z, in the vehicle-fixed frame, in N·m. The signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, under **Input signals**, select **External moments**.

Fh — Hitch force on the body
array

Hitch force applied to the body at the hitch location, Fh_x, Fh_y, Fh_z, in the vehicle-fixed frame, in N. The signal array dimensions are [1x3] or [3x1].

Dependencies

To enable this port, under **Input signals**, select **Hitch forces**.

Mh — Hitch moment about body
array

Hitch moment at the hitch location, Mh_x, Mh_y, Mh_z, about the vehicle-fixed frame, in N·m. The signal array dimensions are [1x3] or [3x1].

Dependencies

To enable this port, under **Input signals**, select **Hitch moments**.

WindXYZ — Wind speed
array

Wind speed, W_x, W_y, W_z, along the inertial X-, Y-, and Z-axes, in m/s. The signal vector dimensions are [1x3] or [3x1].

Dependencies

To enable this port, under **Input signals**, select **External wind**.

Mu — Tire friction coefficient
array

Tire friction coefficient, μ. The value is dimensionless.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single 2-axle	Friction coefficient on the wheels	$Mu = \begin{bmatrix} \mu_f & \mu_r \end{bmatrix}$ or $\begin{bmatrix} \mu_f \\ \mu_r \end{bmatrix}$	Array – [1x2] or [2x1]
Dual 2-axle	Friction coefficient on the wheels	$Mu = \begin{bmatrix} \mu_{fl} & \mu_{fr} \\ \mu_{rl} & \mu_{rr} \end{bmatrix}$	Array – [2x2]

Vehicle Track Setting	Description	Variable	Signal Dimension
Single 3-axle	Friction coefficient on the wheels	$Mu = \begin{bmatrix} \mu_f & \mu_m & \mu_r \end{bmatrix}$ or $\begin{bmatrix} \mu_f \\ \mu_m \\ \mu_r \end{bmatrix}$	Array – [1x3] or [3x1]
Dual 3-axle	Friction coefficient on the wheels	$Mu = \begin{bmatrix} \mu_{fl} & \mu_{fr} \\ \mu_{ml} & \mu_{mr} \\ \mu_{rl} & \mu_{rr} \end{bmatrix}$	Array – [3x2]

Dependencies

To enable this port, under **Input signals**, select **External friction**.

AirTemp — Ambient air temperature
scalar

Ambient air temperature, in K.

Dependencies

To enable this port, under **Input signals**, select **Air temperature**.

X_o — Initial longitudinal position
scalar

Initial vehicle CG displacement along the earth-fixed *X*-axis, in m.

Dependencies

To enable this port, under **Input signals**, select **Initial longitudinal position**.

Y_o — Initial lateral position
scalar

Initial vehicle CG displacement along the earth-fixed *Y*-axis, in m.

Dependencies

To enable this port, under **Input signals**, select **Initial lateral position**.

xdot_o — Initial longitudinal position
scalar

Initial vehicle CG velocity along the vehicle-fixed *x*-axis, in m/s.

Dependencies

To enable this port:

1 Set **Axle forces** to one of these options:

- External longitudinal forces
- External forces

2 Under **Input signals**, select **Initial longitudinal velocity**

ydot_o — Initial lateral position
scalar

Initial vehicle CG velocity along the vehicle-fixed *y*-axis, in m/s.

Dependencies

To enable this port, under **Input signals**, select **Initial lateral velocity**.

psi_o — Initial yaw angle
scalar

Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw), in rad.

Dependencies

To enable this port, under **Input signals**, select **Initial yaw angle**.

r_o — Initial yaw rate
scalar

Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate), in rad/s.

Dependencies

To enable this port, under **Input signals**, select **Initial yaw rate**.

Output

Info — Trailer data
bus

Trailer data, returned as a bus signal containing these block values.

Signal				Description	Value	Units
InertFrm	Cg	Disp	X	Vehicle CG displacement along the earth-fixed *X*-axis	Computed	m
			Y	Vehicle CG displacement along the earth-fixed *Y*-axis	Computed	m
			Z	Vehicle CG displacement along the earth-fixed *Z*-axis	0	m
		Vel	Xdot	Vehicle CG velocity along the earth-fixed *X*-axis	Computed	m/s
			Ydot	Vehicle CG velocity along the earth-fixed *Y*-axis	Computed	m/s

Signal				Description	Value	Units
			Zdot	Vehicle CG velocity along the earth-fixed *Z*-axis	0	m/s
		Ang	phi	Rotation of the vehicle-fixed frame about the earth-fixed *X*-axis (roll)	0	rad
			theta	Rotation of the vehicle-fixed frame about the earth-fixed *Y*-axis (pitch)	0	rad
			psi	Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw)	Computed	rad
	FrntAxl	Lft	Disp	X	Front left wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Front left wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Front left wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Front left wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Front left wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Front left wheel velocity along the earth-fixed *Z*-axis	0	m/s
		Rght	Disp	X	Front right wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Front right wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Front right wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Front right wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Front right wheel velocity along the earth-fixed *Y*-axis	Computed	m/s

Signal					Description	Value	Units
				Zdot	Front right wheel velocity along the earth-fixed Z-axis	0	m/s
	MidlAxl	Lft	Disp	X	Middle left wheel displacement along the earth-fixed X-axis	Computed	m
				Y	Middle left wheel displacement along the earth-fixed Y-axis	Computed	m
				Z	Middle left wheel displacement along the earth-fixed Z-axis	0	m
			Vel	Xdot	Middle left wheel velocity along the earth-fixed X-axis	Computed	m/s
				Ydot	Middle left wheel velocity along the earth-fixed Y-axis	Computed	m/s
				Zdot	Middle left wheel velocity along the earth-fixed Z-axis	0	m/s
		Rght	Disp	X	Middle right wheel displacement along the earth-fixed X-axis	Computed	m
				Y	Middle right wheel displacement along the earth-fixed Y-axis	Computed	m
				Z	Middle right wheel displacement along the earth-fixed Z-axis	0	m
			Vel	Xdot	Middle right wheel velocity along the earth-fixed X-axis	Computed	m/s
				Ydot	Middle right wheel velocity along the earth-fixed Y-axis	Computed	m/s
				Zdot	Middle right wheel velocity along the earth-fixed Z-axis	0	m/s
	RearAxl	Lft	Disp	X	Rear left wheel displacement along the earth-fixed X-axis	Computed	m
				Y	Rear left wheel displacement along the earth-fixed Y-axis	Computed	m

Signal					Description	Value	Units
				Z	Rear left wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Rear left wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Rear left wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Rear left wheel velocity along the earth-fixed *Z*-axis	0	m/s
		Rght	Disp	X	Rear right wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Rear right wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Rear right wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Rear right wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Rear right wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Rear right wheel velocity along the earth-fixed *Z*-axis	0	m/s
	Geom	Disp	X		Trailer body offset from the axle plane along the earth-fixed *X*-axis	Computed	m
			Y		Trailer body offset from the center plane along the earth-fixed *Y*-axis	Computed	m
			Z		Trailer body offset from the axle plane along the earth-fixed *Z*-axis	Computed	m
		Vel	Xdot		Trailer body offset velocity along the earth-fixed *X*-axis	Computed	m/s
			Ydot		Trailer body offset velocity along the earth-fixed *Y*-axis	Computed	m/s

Signal				Description	Value	Units
			Zdot	Trailer body offset velocity along the earth-fixed Z-axis	Computed	m/s
BdyFrm	Cg	Vel	xdot	Vehicle CG velocity along the vehicle-fixed x-axis	Computed	m/s
			ydot	Vehicle CG velocity along the vehicle-fixed y-axis	Computed	m/s
			zdot	Vehicle CG velocity along the vehicle-fixed z-axis	0	m/s
		Ang	Beta	Body slip angle, β $$\beta = \frac{V_y}{V_x}$$	Computed	rad
		AngVel	p	Vehicle angular velocity about the vehicle-fixed x-axis (roll rate)	0	rad/s
			q	Vehicle angular velocity about the vehicle-fixed y-axis (pitch rate)	0	rad/s
			r	Vehicle angular velocity about the vehicle-fixed z-axis (yaw rate)	Computed	rad/s
		Acc	ax	Vehicle CG acceleration along the vehicle-fixed x-axis	Computed	gn
			ay	Vehicle CG acceleration along the vehicle-fixed y-axis	Computed	gn
			az	Vehicle CG acceleration along the vehicle-fixed z-axis	0	gn
			xddot	Vehicle CG acceleration along the vehicle-fixed x-axis	Computed	m/s^2
			yddot	Vehicle CG acceleration along the vehicle-fixed y-axis	Computed	m/s^2
			zddot	Vehicle CG acceleration along the vehicle-fixed z-axis	0	m/s^2
		AngAcc	pdot	Vehicle angular acceleration about the vehicle-fixed x-axis	0	rad/s

Signal				Description	Value	Units
			qdot	Vehicle angular acceleration about the vehicle-fixed *y*-axis	0	rad/s
			rdot	Vehicle angular acceleration about the vehicle-fixed *z*-axis	Computed	rad/s
	Forces	Body	Fx	Net force on the vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy	Net force on the vehicle CG along the vehicle-fixed *y*-axis	Computed	N
			Fz	Net force on the vehicle CG along the vehicle-fixed *z*-axis	0	N
		Ext	Fx	External force on the vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy	External force on the vehicle CG along the vehicle-fixed *y*-axis	Computed	N
			Fz	External force on the vehicle CG along the vehicle-fixed *z*-axis	0	N
		Hitch	Fx	Hitch force applied to the body at the hitch location along the vehicle-fixed *x*-axis	Computed	N
			Fy	Hitch force applied to the body at the hitch location along the vehicle-fixed *y*-axis	Computed	N
			Fz	Hitch force applied to the body at the hitch location along the vehicle-fixed *z*-axis	Computed	N
		FrntAxl	Lft	Fx	Longitudinal force on the left front wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on the left front wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on the left front wheel along the vehicle-fixed *z*-axis	Computed	N

Signal					Description	Value	Units
			Rght	Fx	Longitudinal force on the right front wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on the right front wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on the right front wheel along the vehicle-fixed *z*-axis	Computed	N
		MidlAxl	Lft	Fx	Longitudinal force on the left middle wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on the left middle wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on the left middle wheel along the vehicle-fixed *z*-axis	Computed	N
			Rght	Fx	Longitudinal force on the right middle wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on the right middle wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on the right middle wheel along the vehicle-fixed *z*-axis	Computed	N
		RearAxl	Lft	Fx	Longitudinal force on the left rear wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on the left rear wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on the left rear wheel along the vehicle-fixed *z*-axis	Computed	N
			Rght	Fx	Longitudinal force on the right rear wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on the right rear wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on the right rear wheel along the vehicle-fixed *z*-axis	Computed	N

5-137

Signal						Description	Value	Units
		Tires	FrntTires	Lft	Fx	Front left tire force along the vehicle-fixed *x*-axis	Computed	N
					Fy	Front left tire force along the vehicle-fixed *y*-axis	Computed	N
					Fz	Front left tire force along the vehicle-fixed *z*-axis	Computed	N
				Rght	Fx	Front right tire force along the vehicle-fixed *x*-axis	Computed	N
					Fy	Front right tire force along the vehicle-fixed *y*-axis	Computed	N
					Fz	Front right tire force along the vehicle-fixed *z*-axis	Computed	N
			RearTires	Lft	Fx	Rear left tire force along the vehicle-fixed *x*-axis	Computed	N
					Fy	Rear left tire force along the vehicle-fixed *y*-axis	Computed	N
					Fz	Rear left tire force along the vehicle-fixed *z*-axis	Computed	N
				Rght	Fx	Rear right tire force along the vehicle-fixed *x*-axis	Computed	N
					Fy	Rear right tire force along the vehicle-fixed *y*-axis	Computed	N
					Fz	Rear right tire force along the vehicle-fixed *z*-axis	Computed	
		Drag	Fx			Drag force on the vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy			Drag force on the vehicle CG along the vehicle-fixed *y*-axis	Computed	N
			Fz			Drag force on the vehicle CG along the vehicle-fixed *z*-axis	Computed	N
		Grvty	Fx			Gravity force on the vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy			Gravity force on the vehicle CG along the vehicle-fixed *y*-axis	Computed	N

Signal					Description	Value	Units
			Fz		Gravity force on the vehicle CG along the vehicle-fixed *z*-axis	Computed	N
	Moments	Body	Mx		Body moment on the vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My		Body moment on the vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz		Body moment on the vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Drag	Mx		Drag moment on the vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My		Drag moment on the vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz		Drag moment on the vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Ext	Mx		External moment on the vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My		External moment on the vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz		External moment on the vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Hitch	Mx		Hitch moment at the hitch location about vehicle-fixed *x*-axis	0	N·m
			My		Hitch moment at the hitch location about vehicle-fixed *y*-axis	Computed	N·m
			Mz		Hitch moment at the hitch location about vehicle-fixed *z*-axis	0	N·m
	FrntAxl	Lft	Disp	x	Front left wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Front left wheel displacement along the vehicle-fixed *y*-axis	Computed	m

Signal					Description	Value	Units
				z	Front left wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Front left wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Front left wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Front left wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Rght	Disp	x	Front right wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Front right wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Front right wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Front right wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Front right wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Front right wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Steer	WhlAngFL		Front left wheel steering angle	Computed	rad
			WhlAngFR		Front right wheel steering angle	Computed	rad
	MidlAxl	Lft	Disp	x	Middle left wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Middle left wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Middle left wheel displacement along the vehicle-fixed *z*-axis	Computed	m

Signal					Description	Value	Units
			Vel	xdot	Middle left wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Middle left wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Middle left wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Rght	Disp	x	Middle right wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Middle right wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Middle right wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Middle right wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Middle right wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Middle right wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Steer	WhlAngRL		Middle left wheel steering angle	Computed	rad
			WhlAngRR		Middle right wheel steering angle	Computed	rad
	RearAxl	Lft	Disp	x	Rear left wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Rear left wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Rear left wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Rear left wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s

5-141

Signal					Description	Value	Units
				ydot	Rear left wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Rear left wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Rght	Disp	x	Rear right wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Rear right wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Rear right wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Rear right wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Rear right wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Rear right wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Steer	WhlAngRL		Rear left wheel steering angle	Computed	rad
			WhlAngRR		Rear right wheel steering angle	Computed	rad
	Pwr	PwrExt			Applied external power	Computed	W
		PwrHtch			Hitch power	Computed	
		Drag			Power loss due to drag	Computed	W
	Geom	Disp		x	Trailer offset from axle plane along the vehicle-fixed *x*-axis	Input	m
				y	Trailer offset from center plane along the vehicle-fixed *y*-axis	Input	m
				z	Trailer offset from axle plane along the earth-fixed *z*-axis	Input	m
		Vel		xdot	Trailer offset velocity along the vehicle-fixed *x*-axis	Computed	m/s

Signal				Description	Value	Units
			ydot	Trailer offset velocity along the vehicle-fixed y-axis	Computed	m/s
			zdot	Trailer offset velocity along the vehicle-fixed z-axis	0	m/s
		Ang	Beta	Body slip angle, β $$\beta = \frac{V_y}{V_x}$$	Computed	rad

xdot — Trailer longitudinal velocity
scalar

Trailer CG velocity along the vehicle-fixed x-axis, in m/s.

ydot — Trailer lateral velocity
scalar

Trailer CG velocity along the vehicle-fixed y-axis, in m/s.

psi — Yaw
scalar

Rotation of the vehicle-fixed frame about the earth-fixed Z-axis (yaw), in rad.

r — Yaw rate
scalar

Vehicle angular velocity, r, about the vehicle-fixed z-axis (yaw rate), in rad/s.

FzF — Normal force on the front wheels
scalar | array

Normal force on the front wheels, Fz_F, along the vehicle-fixed z-axis, in N.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single 2-axle Single 3-axle	Normal force on the front axle	$FzF = Fz_f$	Scalar – 1
Dual 2-axle Dual 3-axle	Normal force on the front wheels	$FzF = [Fz_{fl} \ Fz_{fr}]$	Array – [1x2]

FzM — Normal force on the middle wheels
scalar | array

Normal force on the middle wheels, Fz_M, along the vehicle-fixed z-axis, in N.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single 3-axle	Normal force on the middle axle	$FzM = Fz_m$	Scalar – 1
Dual 3-axle	Normal force on the right and left middle wheels	$FzM = [Fz_{ml} \; Fz_{rl}]$	Array – [1x2]

Dependencies

To enable this port, set **Vehicle track** to `Single 3-axle` or `Dual 3-axle`.

FzR — Normal force on the rear wheels
scalar | array

Normal force on the rear wheels, Fz_R, along the vehicle-fixed z-axis, in N.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single 2-axle Single 3-axle	Normal force on the rear wheel	$FzR = Fz_r$	Scalar – 1
Dual 2-axle Dual 3-axle	Normal force on the rear wheels	$FzR = [Fz_{rl} \; Fz_{rr}]$	Array – [1x2]

Fhz — Normal component of hitch force on the body
scalar

Normal hitch force applied to the body at the hitch location, Fh_z, in the vehicle-fixed frame z-axis, in N.

If you enable the **Hitch forces** parameter, the block offsets the normal hitch force, Fh_z, with the value of the *Fh* input port component along the vehicle-fixed z-axis.

Parameters

Options

Vehicle track — Type of vehicle track
Dual 3-axle (default) | Single 2-axle | Single 3-axle | Dual 2-axle

Use the **Vehicle track** parameter to specify the number of tracks.

Vehicle Track Setting	Implementation
Single 2-axle	Trailer with a single track and two axles. • Forces act along the center line of the axles. • No lateral load transfer.

Vehicle Track Setting	Implementation
Dual 2-axle	Trailer with a dual track and two axles. Forces act at the axle hard-point locations.
Single 3-axle	Trailer with a single track and three axles. • Forces act along the center line of the axles. • No lateral load transfer.
Dual 3-axle (default)	Trailer with a dual track and three axles. Forces act at the axle hard-point locations.

Axle forces — Type of axle force
External forces (default) | External longitudinal velocity | External longitudinal forces

Use the **Axle forces** parameter to specify the type of force.

Axle Forces Setting	Implementation

External longitudinal velocity	• The block assumes that the external longitudinal velocity is in a quasi-steady state, and the longitudinal acceleration is approximately zero.
	• Because the motion is quasi-steady, the block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	• Consider this setting when you want to:
	
	  • Generate virtual sensor signal data.
	  • Conduct high-level software studies that are not impacted by driveline or nonlinear tire responses.

External longitudinal forces	• The block uses the external longitudinal force to accelerate or brake the vehicle.
	• The block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	• Consider this setting when you want to:
	
	  • Account for changes in the longitudinal velocity on the lateral and yaw motion.
	  • Specify the external longitudinal motion through a force instead of an external longitudinal velocity.
	  • Connect the block to tractive actuators, wheels, brakes, and hitches.

External forces	• The block uses the external lateral and longitudinal forces to steer, accelerate, or brake the vehicle. • The block does not use the steering input to calculate vehicle motion. • Consider this setting when you need tire models with more accurate nonlinear combined lateral and longitudinal slip.

Input Signals

Front wheel steering — WhlAngF input port
off (default) | on

Select to create input port WhlAngF.

Middle wheel steering — WhlAngM input port
off (default) | on

Select to create input port WhlAngM.

Dependencies

To enable this parameter, set **Vehicle track** to Single 3-axle or Dual 3-axle.

Rear wheel steering — WhlAngR input port
off (default) | on

Select to create input port WhlAngR.

External wind — WindXYZ input port
off (default) | on

Select to create input port WindXYZ.

External friction — Mu input port
off (default) | on

Select to create input port Mu.

Dependencies

To enable this parameter, set **Axle forces** to one of these options:

- External longitudinal forces
- External forces

External forces — FExt input port
off (default) | on

Select to create input port FExt.

External moments — MExt input port
off (default) | on

Select to create input port MExt.

Hitch forces — Fh input port
on (default) | off

Select to create input port Fh.

Hitch moments — Mh input port
on (default) | off

Select to create input port Mh.

Initial longitudinal position — X_o input port
off (default) | on

Select to create input port X_o.

Initial yaw angle — psi_o input port
off (default) | on

Select to create input port psi_o.

Initial longitudinal velocity — xdot_o input port
off (default) | on

Select to create input port xdot_o.

Dependencies

To enable this parameter, set **Axle forces** to External longitudinal forces or External forces.

Initial yaw rate — r_o input port
off (default) | on

Select to create input port r_o.

Initial lateral position — Y_o input port
off (default) | on

Select to create input port Y_o.

Air temperature — AirTemp input port
off (default) | on

Select to create input port AirTemp.

Initial lateral velocity — ydot_o input port
off (default) | on

Select to create input port ydot_o.

Longitudinal

Number of wheels on front axle, NF — Front wheel count
2 (default) | scalar

Number of wheels on the front axle, N_F. The value is dimensionless.

Number of wheels on middle axle, NM — Middle wheel count
2 (default) | scalar

Number of wheels on the middle axle, N_M. The value is dimensionless.

Dependencies

To enable this parameter, set **Vehicle track** to Single 3-axle or Dual 3-axle.

Number of wheels on rear axle, NR — Rear wheel count
2 (default) | scalar

Number of wheels on the rear axle, N_R. The value is dimensionless.

Vehicle mass, m — Vehicle mass
26000 (default) | scalar

Vehicle mass, m, in kg.

Longitudinal distance from center of mass to front axle, a — Distance from CM to front axle
4 (default) | scalar

Distance from the vehicle CM to the front axle, a, in m.
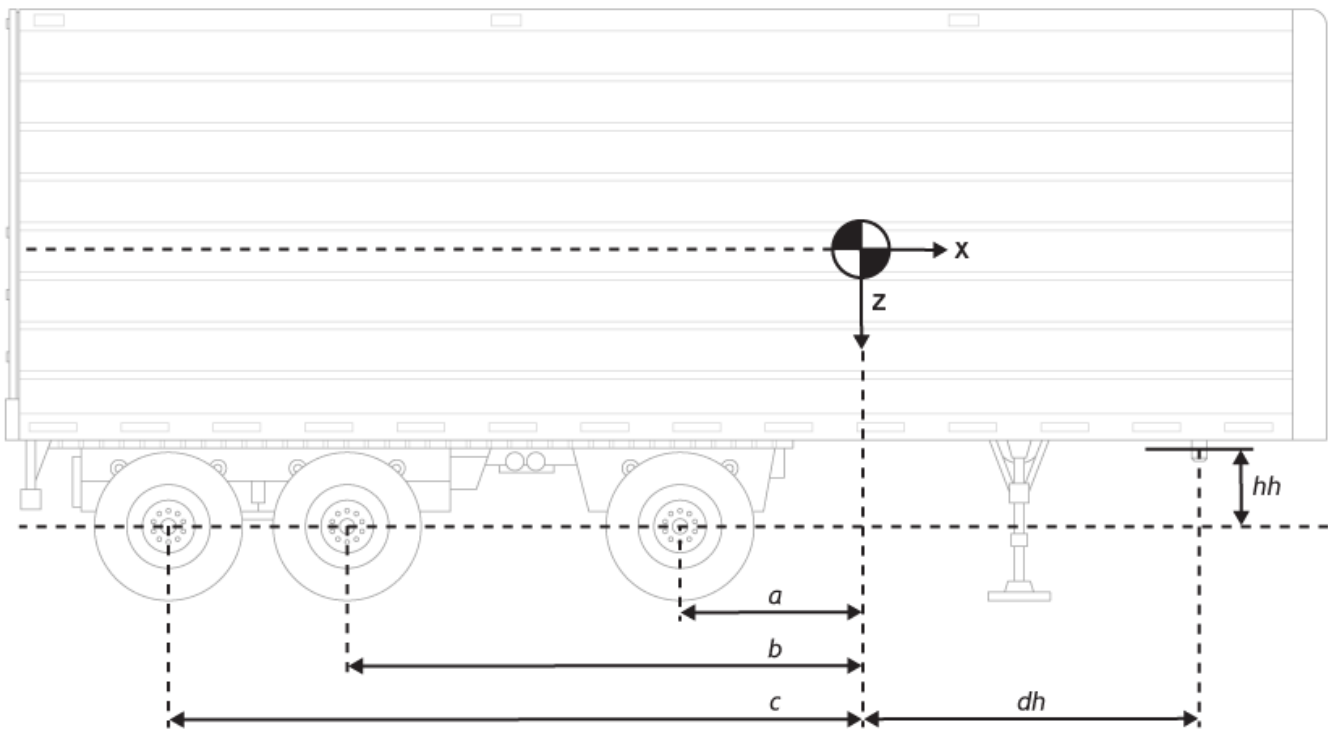
Longitudinal distance from center of mass to middle axle, b — Distance from CM to middle axle
4.5 (default) | scalar

Distance from vehicle CM to middle axle, b, in m.
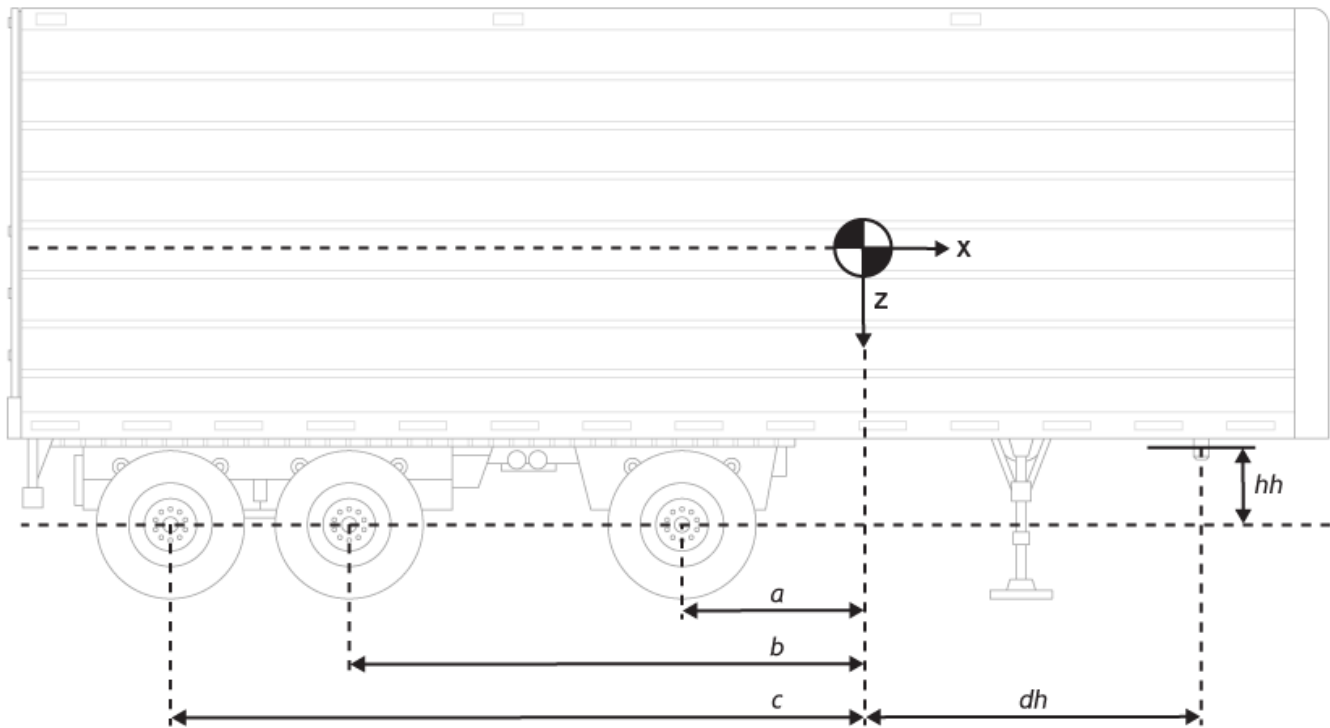
Dependencies

To enable this parameter, set **Vehicle track** to `Single 3-axle` or `Dual 3-axle`.

Longitudinal distance from center of mass to rear axle, c — Distance from CM to rear axle
5 (default) | `scalar`

Distance from vehicle CM to the front axle, *c*, in m.
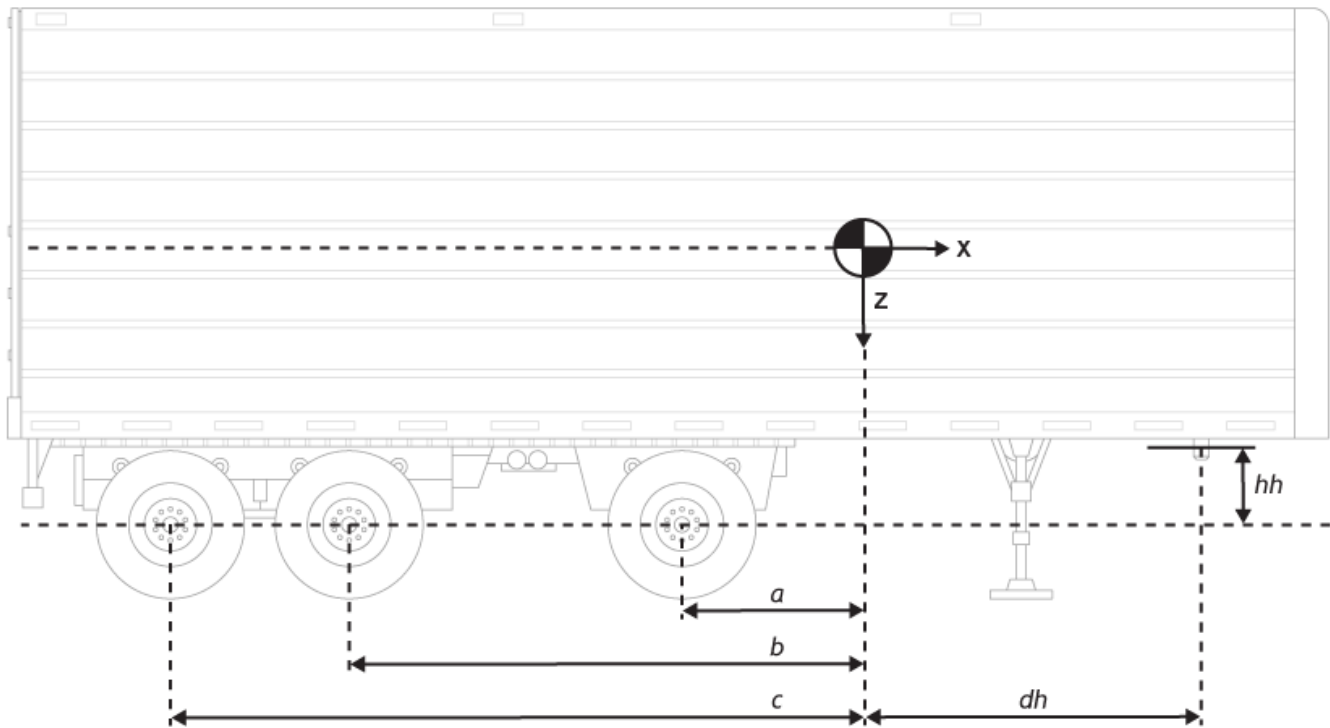
Vertical distance from center of mass to axle plane, h — Distance from CM to axle plane
2 (default) | `scalar`

Vertical distance from vehicle CM to the axle plane, h, in m.
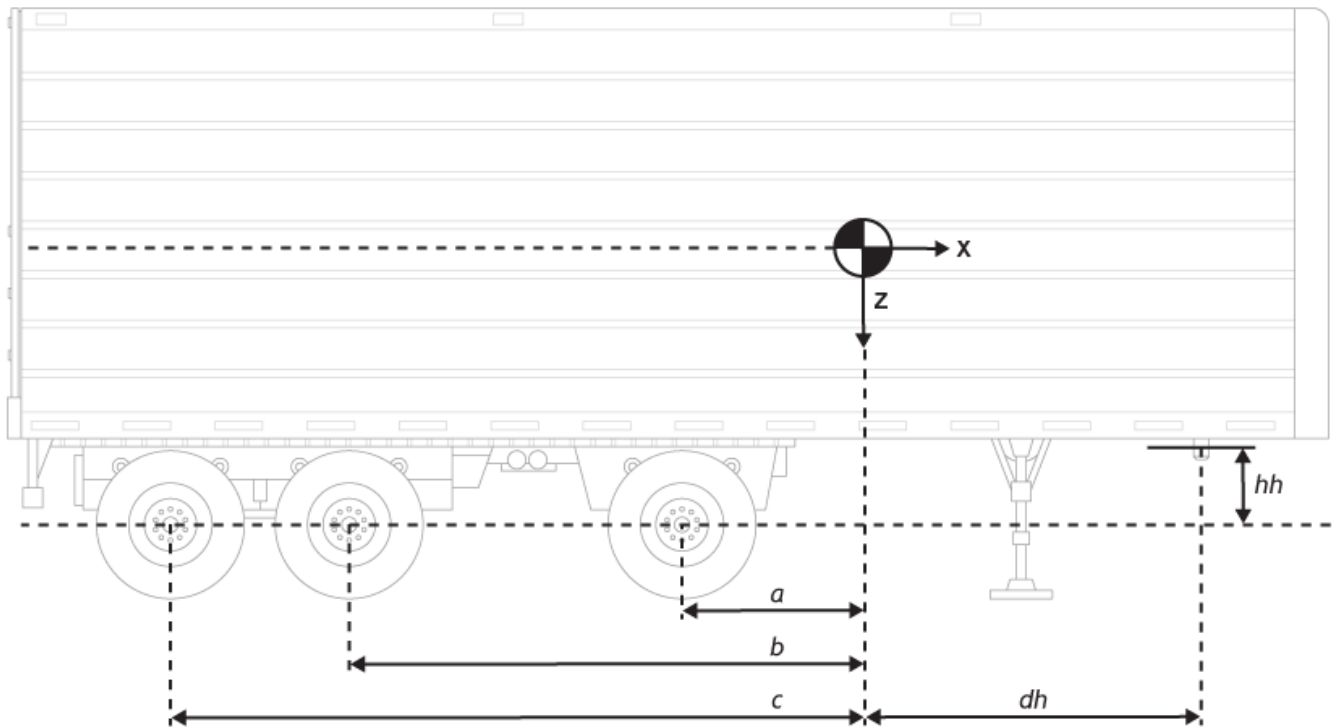
`Vertical distance from hitch to axle plane, hh` — Distance from hitch to axle plane
`0.6` (default) | `scalar`

Vertical distance from the hitch to the axle plane, *hh*, in m.
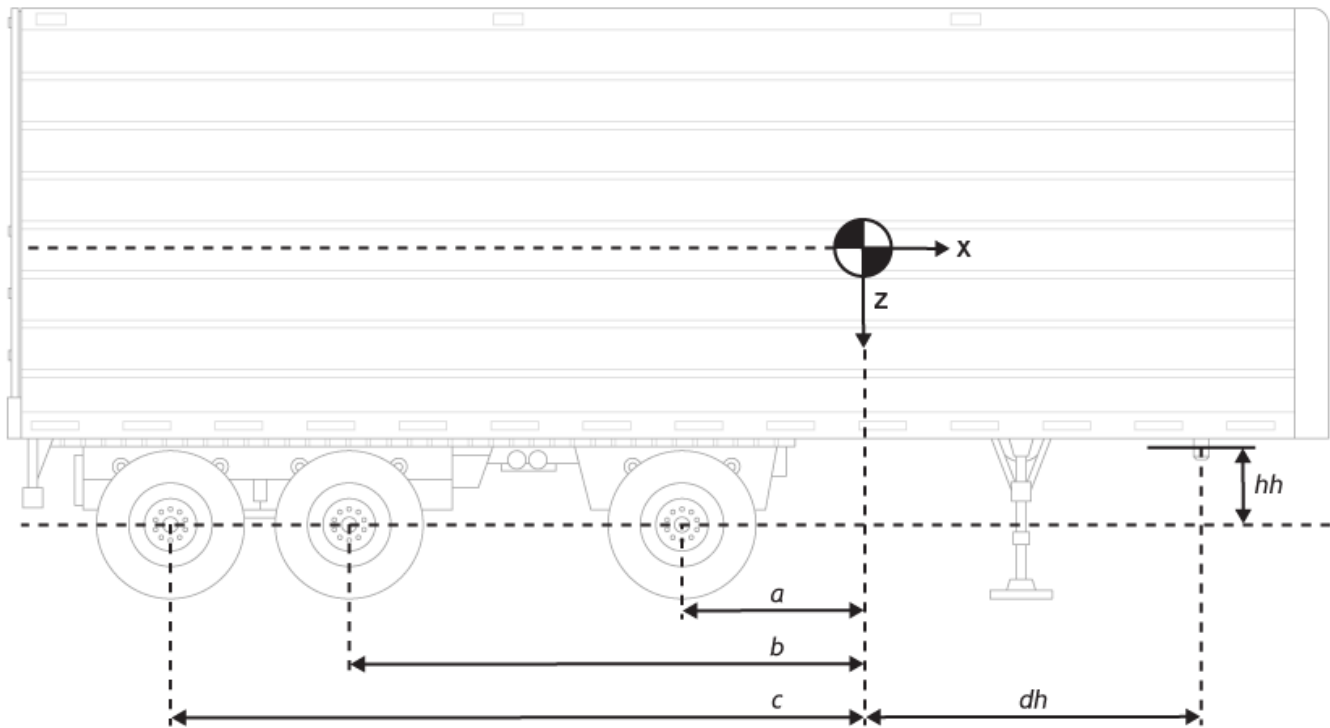
Dependencies

To enable this parameter, on the **Input signals** pane, select **Hitch forces** or **Hitch moments**.

Longitudinal distance from center of mass to hitch, dh — Distance to hitch
7.5 (default) | scalar

Longitudinal distance from the center of mass to the hitch, *dh*, in m.
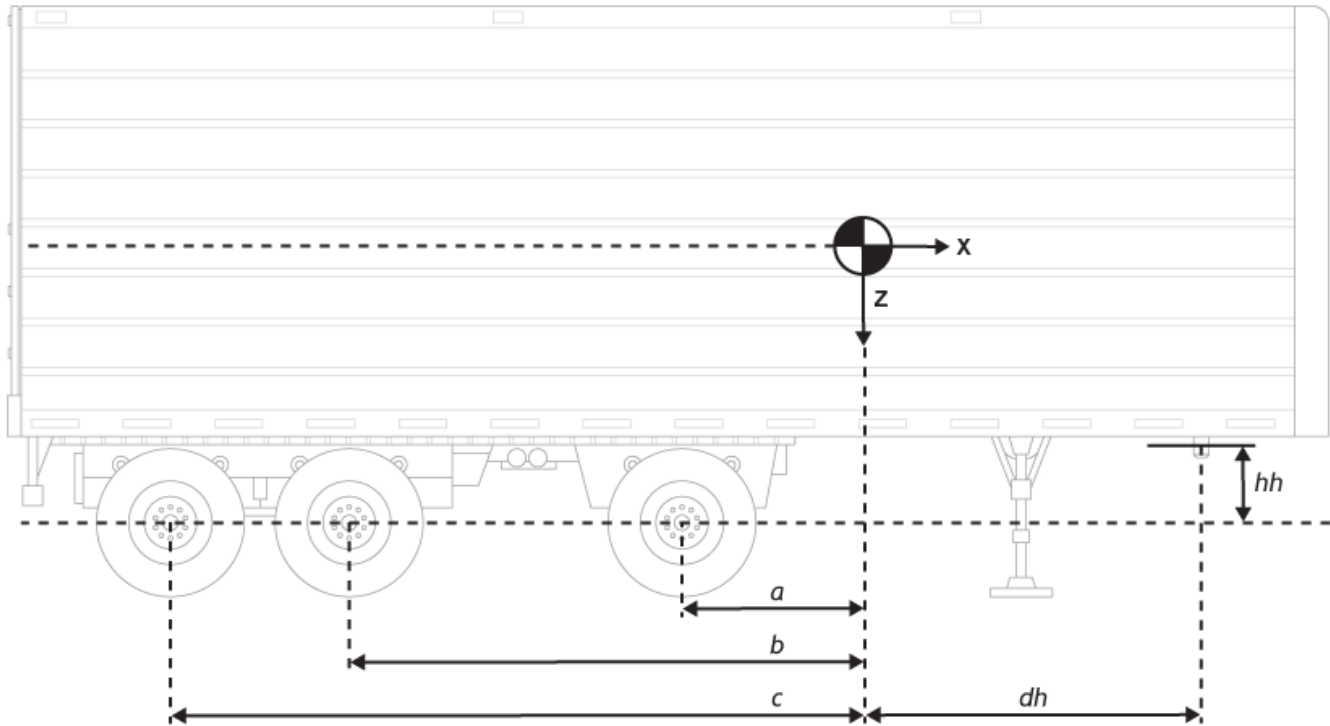
Dependencies

To enable this parameter, on the **Input signals** pane, select **Hitch forces** or **Hitch moments**.

`Initial inertial frame longitudinal position, X_o` — Initial inertial X location
`0` (default) | `scalar`

Initial vehicle CG displacement along the earth-fixed *X*-axis, in m.

`Initial longitudinal velocity, xdot_o` — Initial velocity
`0` (default) | `scalar`

Initial vehicle CG velocity along the vehicle-fixed *x*-axis, in m/s.

Dependencies

To enable this parameter, set **Axle forces** to one of these options:

- `External longitudinal forces`
- `External forces`

Lateral

`Mapped corner stiffness` — Selection
`off` (default) | `on`

Enables mapped corner stiffness calculation.

Dependencies

To enable this parameter, set **Axle forces** to one of these options:

- `External longitudinal velocity`
- `External longitudinal forces`

Include relaxation length dynamics — Enable relaxation length dynamics
`on` (default) | `off`

Enables relaxation length dynamics.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- `External longitudinal velocity`
- `External longitudinal forces`

2   Clear **Mapped corner stiffness**.

Lateral distance from geometric centerline to center of mass, d — Distance from centerline to CM
`0` (default) | `scalar`

Lateral distance from the geometric centerline to the center of mass, *d*, in m, along the vehicle-fixed *y*. Positive values indicate that the trailer CM is to the right of the geometric centerline. Negative values indicate that the trailer CM is to the left of the geometric centerline.
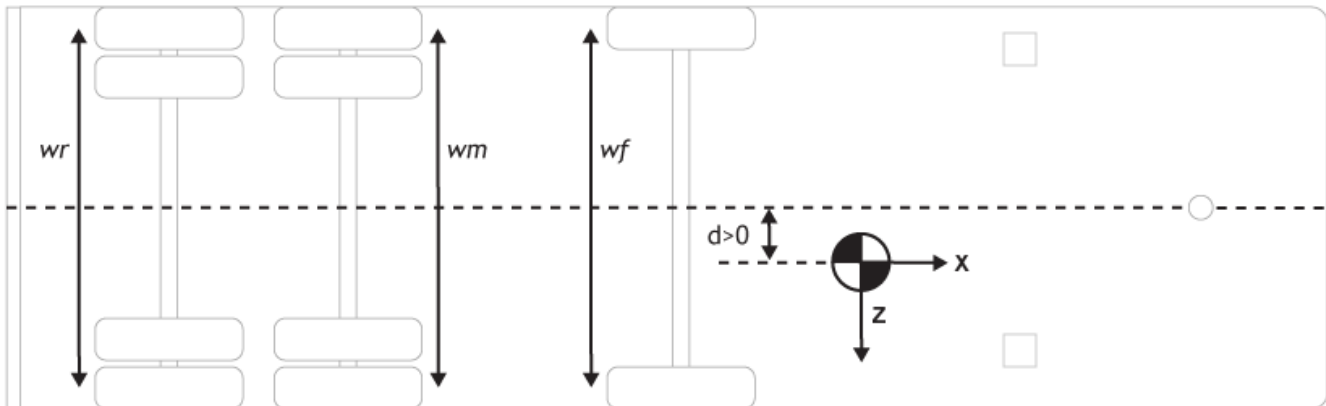
`Front track width, w_f` — Front track width
1.82 (default) | `scalar`

Front track width, *wf*, in m.
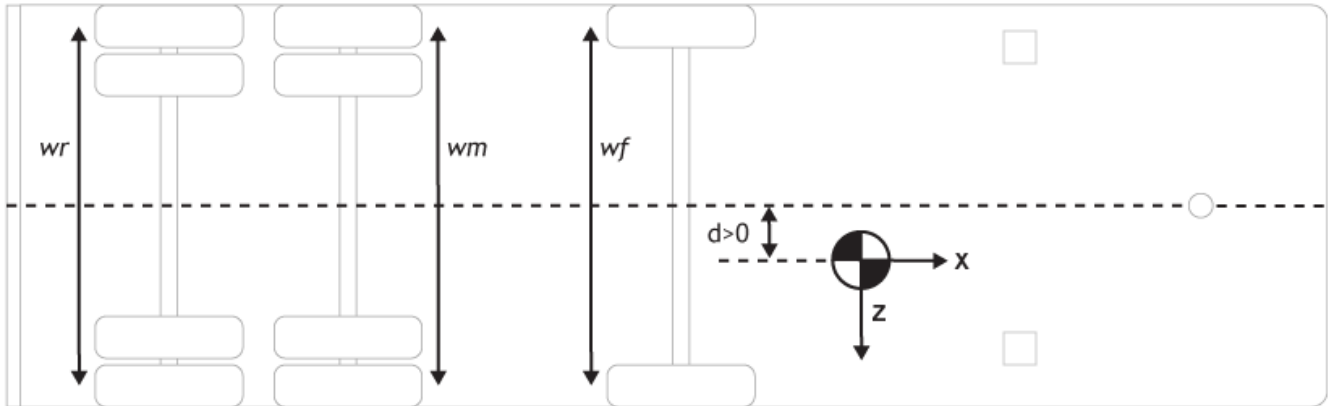
Dependencies

To enable this parameter, set **Vehicle track** to `Dual 2-axle` or `Dual 3-axle`.

Middle track width, `w_m` — Middle track width
`1.82` (default) | `scalar`

Middle track width, *wm*, in m.
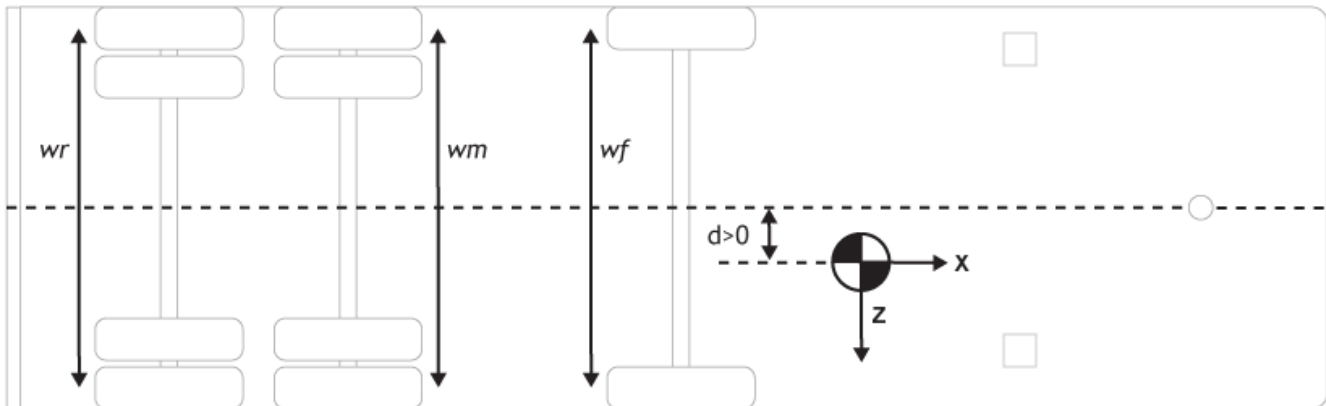
Dependencies

To enable this parameter, set **Vehicle track** to `Dual 3-axle`.

Rear track width, w_r — Rear track width
1.82 (default) | scalar

Rear track width, *wr*, in m.
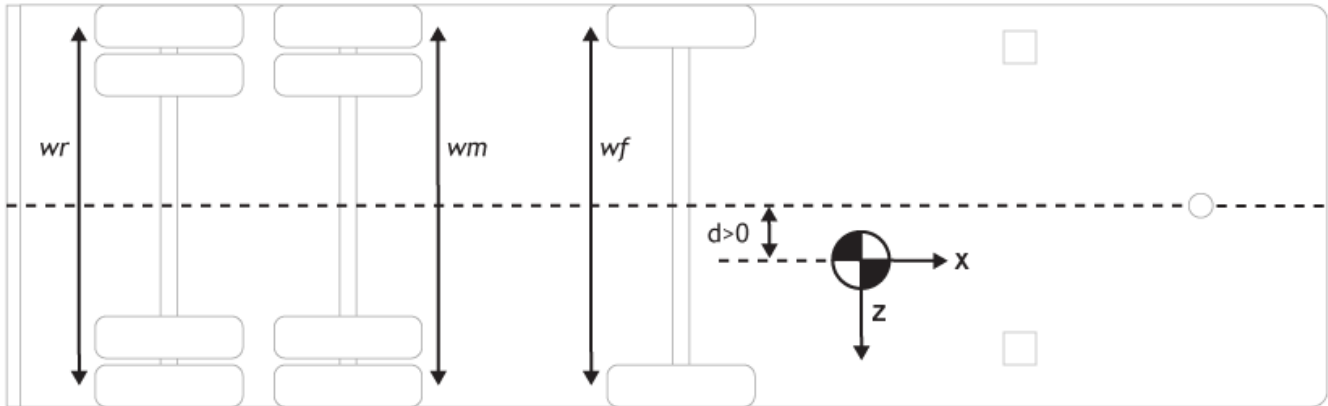
Dependencies

To enable this parameter, set **Vehicle track** to `Dual 2-axle` or `Dual 3-axle`.

`Front axle tire corner stiffness, Cy_f` — **Front axle tire stiffness**
`12.3` (default) | `scalar`

Front tire corner stiffness, Cy_f, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- `External longitudinal velocity`
- `External longitudinal forces`

2 Clear **Mapped corner stiffness**.

`Middle axle tire corner stiffness, Cy_m` — **Middle axle tire stiffness**
`11.3` (default) | `scalar`

Middle tire corner stiffness, Cy_m, in N/rad.

Dependencies

To enable this parameter:

1 Set **Vehicle track** to one of these options:

- Single 3-axle
- Dual 3-axle

2 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

3 Clear **Mapped corner stiffness**.

Rear axle tire corner stiffness, Cy_r — Rear axle tire stiffness
11.3 (default) | scalar

Rear tire corner stiffness, Cy_r, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Clear **Mapped corner stiffness**.

Front tire(s) relaxation length, sigma_f — Relaxation length
.1 (default) | scalar

Front tire relaxation length, σ_f, in m.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Do either of these:

- Select **Mapped corner stiffness**.
- Clear **Mapped corner stiffness** and select **Include relaxation length dynamics**.

Middle tire(s) relaxation length, sigma_m — Relaxation length
.1 (default) | scalar

Middle tire relaxation length, σ_m, in m.

Dependencies

To enable this parameter:

1 Set **Vehicle track** to one of these options:

- Single 3-axle
- Dual 3-axle

2 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

3 Do either of these:

- Select **Mapped corner stiffness**.
- Clear **Mapped corner stiffness** and select **Include relaxation length dynamics**.

Rear tire(s) relaxation length, sigma_r — Relaxation length
.1 (default) | scalar

Rear tire relaxation length, σ_r, in m.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Do either of these:

- Select **Mapped corner stiffness**.
- Clear **Mapped corner stiffness** and select **Include relaxation length dynamics**.

Front axle slip angle breakpoints, alpha_f_brk — Breakpoints
[-.1 .1] (default) | vector

Front axle slip angle breakpoints, α_{fbrk}, in rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Select **Mapped corner stiffness**.

Front axle corner data, Cy_f_data — Breakpoints
[-9e3 9e3] (default) | vector

Front axle corner data, Cy_{fdata}, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Select **Mapped corner stiffness**.

Middle axle slip angle breakpoints, alpha_m_brk — Breakpoints
[-.1 .1] (default) | vector

Middle axle slip angle breakpoints, α_{mbrk}, in rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Select **Mapped corner stiffness**.

Middle axle corner data, Cy_m_data — Breakpoints
[-9e3 9e3] (default) | vector

Middle axle corner data, Cy_{mdata}, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Select **Mapped corner stiffness**.

Rear axle slip angle breakpoints, alpha_r_brk — Breakpoints
[-.1 .1] (default) | vector

Rear axle slip angle breakpoints, α_{rbrk}, in rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Select **Mapped corner stiffness**.

Rear axle corner data, Cy_r_data — Data
[-9e3 9e3] (default) | vector

Rear axle corner data, Cy_{rdata}, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- `External longitudinal velocity`
- `External longitudinal forces`

2 Select **Mapped corner stiffness**.

`Initial inertial frame lateral displacement, Y_o` — **Position**
0 (default) | scalar

Initial vehicle CG displacement along the earth-fixed *Y*-axis, in m.

`Initial lateral velocity, ydot_o` — **Velocity**
0 (default) | scalar

Initial vehicle CG velocity along the vehicle-fixed *y*-axis, in m/s.

Yaw

`Yaw polar inertia, Izz` — **Inertia**
4000 (default) | scalar

Yaw polar inertia, in kg*m^2.

`Initial yaw angle, psi_o` — **Psi rotation**
0 (default) | scalar

Rotation of the vehicle-fixed frame about earth-fixed *Z*-axis (yaw), in rad.

`Initial yaw rate, r_o` — **Yaw rate**
0 (default) | scalar

Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate), in rad/s.

Aerodynamic

`Longitudinal drag area, Af` — **Effective vehicle cross-sectional area**
2 (default) | scalar

Effective vehicle cross-sectional area, A_f, to calculate the aerodynamic drag force on the vehicle, in m^2.

`Longitudinal drag coefficient, Cd` — **Air drag coefficient**
.3 (default) | scalar

Air drag coefficient, C_d. The value is dimensionless.

`Longitudinal lift coefficient, Cl` — **Air lift coefficient**
.1 (default) | scalar

Air lift coefficient, C_l. The value is dimensionless.

Longitudinal drag pitch moment, Cpm — Pitch drag
.1 (default) | scalar

Longitudinal drag pitch moment coefficient, C_{pm}. The value is dimensionless.

Relative wind angle vector, beta_w — Wind angle
[0:0.01:0.3] (default) | vector

Relative wind angle vector, β_w, in rad.

Side force coefficient vector, Cs — Side force coefficient
[0:0.03:0.9] (default) | vector

Side force coefficient vector coefficient, C_s. The value is dimensionless.

Yaw moment coefficient vector, Cym — Yaw moment drag
[0:0.01:0.3] (default) | vector

Yaw moment coefficient vector coefficient, C_{ym}. The value is dimensionless.

Environment

Absolute air pressure, Pabs — Pressure
101325 (default) | scalar

Environmental absolute pressure, P_{abs}, in Pa.

Air temperature, Tair — Temperature
273 (default) | scalar

Environmental absolute temperature, T, in K.

Dependencies

To enable this parameter, clear **Air temperature**.

Gravitational acceleration, g — Gravity
9.81 (default) | scalar

Gravitational acceleration, g, in m/s^2.

Nominal friction scaling factor, mu — Friction scale factor
1 (default) | scalar

Nominal friction scale factor, μ. The value is dimensionless.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

   • External longitudinal velocity
   • External longitudinal forces

2   Clear **External Friction**.

Simulation

Longitudinal velocity tolerance, xdot_tol — Tolerance
.01 (default) | scalar

Longitudinal velocity tolerance, in m/s.

Nominal normal force, Fznom — Normal force
5000 (default) | scalar

Nominal normal force, in N.

Dependencies

To enable this parameter, set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

Geometric longitudinal offset from axle plane, longOff — Longitudinal offset
0 (default) | scalar

Vehicle chassis offset from the axle plane along the vehicle-fixed *x*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independently of the vehicle CG.

Geometric lateral offset from center plane, latOff — Lateral offset
0 (default) | scalar

Vehicle chassis offset from the center plane along the vehicle-fixed *y*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independently of the vehicle CG.

Geometric vertical offset from axle plane, vertOff — Vertical offset
0 (default) | scalar

Vehicle chassis offset from the axle plane along the vehicle-fixed *z*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independently of the vehicle CG.

Wrap Euler angles, wrapAng — Wrap the Euler angles to the interval [-pi, pi]
off (default) | on

Wrap the Euler angles to the interval [-pi, pi]. For vehicle maneuvers that might undergo vehicle yaw rotations that are outside of this interval, consider clearing the parameter if you want to:

- Track the total vehicle yaw rotation.
- Avoid discontinuities in the vehicle state estimators.

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers (SAE), 1992.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Vehicle Body 3DOF Three Axles

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2020a

Vehicle Body 3DOF Three Axles

Three-axle vehicle body with longitudinal, lateral, and yaw motion
Library: Vehicle Dynamics Blockset / Vehicle Body
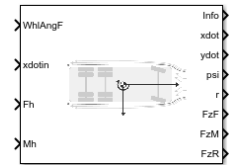
Description

The Vehicle Body 3DOF Three Axles block implements a rigid, three-axle vehicle body model to calculate longitudinal, lateral, and yaw motion. The block accounts for the axle and hitch reaction forces due to the vehicle body mass acceleration, aerodynamic drag, and steering.

Use this block in vehicle dynamics and automated driving studies to model nonholonomic vehicle motion when vehicle pitch, roll, and vertical motion are not significant.

Use the **Vehicle track** parameter to specify the number of tracks.

Vehicle Track Setting	Implementation
Single (bicycle)	• Forces act along the center line of the axles. • No lateral load transfer.
Dual	Forces act at the axle hard-point locations.

Use the **Axle forces** parameter to specify the type of force.

Axle Forces Setting	Implementation

External longitudinal velocity	• The block assumes that the external longitudinal velocity is in a quasi-steady state, so the longitudinal acceleration is approximately zero.
	 • Because the motion is quasi-steady, the block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	 • Consider this setting when you want to:
	  • Generate virtual sensor signal data.
	  • Conduct high-level software studies that are not impacted by driveline or nonlinear tire responses.

Axle Forces Setting	Implementation

External longitudinal forces	• The block uses the external longitudinal force to accelerate or brake the vehicle.
	• The block calculates lateral forces using the tire slip angles and linear cornering stiffness.
	• Consider this setting when you want to:
	  • Account for changes in the longitudinal velocity on the lateral and yaw motion.
	  • Specify the external longitudinal motion through a force instead of an external longitudinal velocity.
	  • Connect the block to tractive actuators, wheels, brakes, and hitches.

External forces	• The block uses the external lateral and longitudinal forces to steer, accelerate, or brake the vehicle. • The block does not use the steering input to calculate vehicle motion. • Consider this setting when you need tire models with more accurate nonlinear combined lateral and longitudinal slip.

To create additional input ports, under **Input signals**, select these block parameters.

Input Signals Pane Parameter	Input Port	Description
Front wheel steering	WhlAngF	Front wheel angle, δ_F
Middle wheel steering	WhlAngM	Middle wheel angle, δ_M
Rear wheel steering	WhlAngR	Rear wheel angle, δ_R
External wind	WindXYZ	Wind speed, W_X, W_Y, and W_Z, in an inertial reference frame
External friction	Mu	Friction coefficient
External forces	FExt	External force on the vehicle center of gravity (CG), F_x, F_y, and F_z, in the vehicle-fixed frame
External moments	MExt	External moment about the vehicle CG, M_x, M_y, and M_z, in the vehicle-fixed frame
Hitch forces	Fh	Hitch force applied to the body at the hitch location, Fh_x, Fh_y, and Fh_z, in the vehicle-fixed frame
Hitch moments	Mh	Hitch moment at the hitch location, Mh_x, Mh_y, and Mh_z, about the vehicle-fixed frame
Initial longitudinal position	X_o	Initial vehicle CG displacement along the earth-fixed X-axis
Initial yaw angle	psi_o	Initial rotation of the vehicle-fixed frame about the earth-fixed Z-axis (yaw)
Initial longitudinal velocity	xdot_o	Initial vehicle CG velocity along the vehicle-fixed x-axis

Input Signals Pane Parameter	Input Port	Description
Initial yaw rate	r_o	Initial vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate)
Initial lateral position	Y_o	Initial vehicle CG displacement along the earth-fixed *Y*-axis
Air temperature	AirTemp	Ambient air temperature. Consider this option if you want to vary the temperature during run time.
Initial lateral velocity	ydot_o	Initial vehicle CG velocity along the vehicle-fixed *y*-axis

Theory

To determine the vehicle motion, the block solves the rigid body planar dynamics equations of motion.

Calculation	Description
Dynamics	The block solves the rigid-body planar dynamics equations to determine the vehicle longitudinal motion. If you set **Axle forces** to `External longitudinal velocity`, the block assumes a quasi-steady state for the longitudinal acceleration.
External forces	External forces include both drag and external force inputs. The forces act on the vehicle CG. The block divides the normal forces by the nominal normal load to vary the effective friction parameters during weight and load transfer. The block maintains pitch and roll equilibrium.
Tire forces	The block uses the ratio of the local, longitudinal, and lateral velocities to determine the slip angles. The block uses the steering angles to transform the tire forces to the vehicle-fixed frame. If you set **Axle forces** to `External forces`, the block assumes that the externally provided forces are in the vehicle-fixed frame at the axle-wheel location.

Single Track

Dual Track

The illustrations use these variables.

a, *b*, *c*	Longitudinal distance of the front, middle, and rear axles, respectively, from the normal projection point of the vehicle CG onto the common axle plane
h	Height of vehicle CG above the axle plane along the vehicle-fixed z-axis
d	Lateral distance from geometric centerline to center of mass along the vehicle-fixed *y*-axis
hh	Height of the hitch above the axle plane along the vehicle-fixed z-axis
dh	Longitudinal distance of the hitch from normal projection point of the vehicle CG onto the common axle plane
wf, *wm*, *wr*	Front, middle, and rear track width, respectively

Drag

This table summarizes the block implementation for the drag calculation.

Calculation	Description
Coordinate transformation	The block transforms the wind speeds from the inertial frame to the vehicle-fixed frame.

Calculation	Description
Drag forces	To determine a relative airspeed, the block subtracts the wind speed from the CG vehicle velocity. Using the relative airspeed, the block determines the drag forces.
Drag moments	Using the relative airspeed, the block determines the drag moments.

Lateral Corner Stiffness and Relaxation Dynamics

To enable the mapped corner stiffness and relaxation length dynamic parameters, set **Axle forces** to `External longitudinal force` or `External longitudinal velocity`.

Parameter Settings		Description
Mapped Corner Stiffness	**Include Relaxation Length Dynamics**	
`Off` (default)	`On` (default)	The block uses constant corner stiffness values. The slip angles include the relaxation length dynamic settings. The relaxation length approximates an effective corner stiffness force that is a function of wheel travel.
`On`	`On` (default)	The block uses lookup tables that are functions of the corner stiffness data and slip angles. The slip angles include the relaxation length dynamic settings. The relaxation length approximates an effective corner stiffness force that is a function of wheel travel.
`Off` (default)	`Off`	The block uses constant corner stiffness values.

Ports

Input

WhlAngF — Front wheel steering angles
scalar | array

Front wheel steering angles, δ_F, in rad.

Vehicle Track Setting	Variable	Signal Dimension
`Single (bicycle)`	δ_F	Scalar – 1
`Dual`	$\delta_F = [\delta_{fl}\ \delta_{fr}]$ or $\begin{bmatrix} \delta_{fl} \\ \delta_{fr} \end{bmatrix}$	Array – [1x2] or [2x1]

Dependencies

To enable this port, on the **Input signals** pane, select **Front wheel steering**.

WhlAngM — Middle wheel steering angles
scalar | array

Middle wheel steering angles, δ_M, in rad.

Vehicle Track Setting	Variable	Signal Dimension
Single (bicycle)	δ_M	Scalar – 1
Dual	$\delta_M = [\delta_{ml} \; \delta_{mr}] \;$ or $\; \begin{bmatrix} \delta_{ml} \\ \delta_{mr} \end{bmatrix}$	Array – [1x2] or [2x1]

Dependencies

To enable this port, on the **Input signals** pane, select **Middle wheel steering**.

WhlAngR — Rear wheel steering angles
scalar | array

Rear wheel steering angles, δ_R, in rad.

Vehicle Track Setting	Variable	Signal Dimension
Single (bicycle)	δ_R	Scalar – 1
Dual	$\delta_R = [\delta_{rl} \; \delta_{rr}] \;$ or $\; \begin{bmatrix} \delta_{rl} \\ \delta_{rr} \end{bmatrix}$	Array – [1x2] or [2x1]

Dependencies

To enable this port, on the **Input signals** pane, select **Rear wheel steering**.

xdotin — Longitudinal velocity
scalar

Vehicle CG velocity along the vehicle-fixed *x*-axis, in m/s.

Dependencies

To enable this port, set **Axle forces** to External longitudinal velocity.

FwF — Total force on the front wheels
scalar | array

Force on the front wheels, Fw_F, along the vehicle-fixed axis, in N.

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Single (bicycle)	External longitudinal forces	Longitudinal force on the front wheel	$FwF = Fx_f$	Scalar – 1
	External forces	Longitudinal and lateral forces on the front wheel	$FwF = [Fx_f \; Fy_f] \;$ or $\; \begin{bmatrix} Fx_f \\ Fy_f \end{bmatrix}$	Array – [1x2] or [2x1]

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Dual	External longitudinal forces	Longitudinal force on the front wheels	$FwF = \begin{bmatrix} F_{xfl} & F_{xfr} \end{bmatrix}$ or $\begin{bmatrix} F_{xfl} \\ F_{xfr} \end{bmatrix}$	Array – [1x2] or [2x1]
	External forces	Longitudinal and lateral forces on the front wheels	$FwF = \begin{bmatrix} F_{xfl} & F_{xfr} \\ F_{yfl} & F_{yfr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, set **Axle forces** to one of these options:

- External longitudinal forces
- External forces

FwM — Total force on the middle wheels
scalar | array

Force on the middle wheels, Fw_M, along the vehicle-fixed axis, in N.

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Single (bicycle)	External longitudinal forces	Longitudinal force on the middle wheel	$FwM = Fx_r$	Scalar – 1
	External forces	Longitudinal and lateral forces on the middle wheel	$FwM = \begin{bmatrix} Fx_m & Fy_m \end{bmatrix}$ or $\begin{bmatrix} Fx_m \\ Fy_m \end{bmatrix}$	Array – [1x2] or [2x1]
Dual	External longitudinal forces	Longitudinal force on the middle wheels	$FwM = \begin{bmatrix} F_{xml} & F_{xmr} \end{bmatrix}$ or $\begin{bmatrix} F_{xml} \\ F_{xmr} \end{bmatrix}$	Array – [1x2] or [2x1]
	External forces	Longitudinal and lateral forces on the middle wheels	$FwM = \begin{bmatrix} F_{xml} & F_{xmr} \\ F_{yml} & F_{ymr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, set **Axle forces** to one of these options:

- External longitudinal forces

- External forces

FwR — Total force on the rear wheels
scalar | array

Force on the rear wheels, Fw_R, along the vehicle-fixed axis, in N.

Vehicle Track Setting	Axle Forces Setting	Description	Variable	Signal Dimension
Single (bicycle)	External longitudinal forces	Longitudinal force on the rear wheel	$FwR = Fx_r$	Scalar – 1
	External forces	Longitudinal and lateral forces on the rear wheel	$FwR = \begin{bmatrix} Fx_r & Fy_r \end{bmatrix}$ or $\begin{bmatrix} Fx_r \\ Fy_r \end{bmatrix}$	Array – [1x2] or [2x1]
Dual	External longitudinal forces	Longitudinal force on the rear wheels	$FwR = \begin{bmatrix} F_{xrl} & F_{xrr} \end{bmatrix}$ or $\begin{bmatrix} F_{xrl} \\ F_{xrr} \end{bmatrix}$	Array – [1x2] or [2x1]
	External forces	Longitudinal and lateral forces on the rear wheels	$FwR = \begin{bmatrix} F_{xrl} & F_{xrr} \\ F_{yrl} & F_{yrr} \end{bmatrix}$	Array – [2x2]

Dependencies

To enable this port, set **Axle forces** to one of these options:

- External longitudinal forces
- External forces

FExt — External force on vehicle CG
array

External forces applied to the vehicle CG, F_{xext}, F_{yext}, F_{zext}, in vehicle-fixed frame, in N. The signal array dimensions are [1x3] or [3x1].

Dependencies

To enable this port, on the **Input signals** pane, select **External forces**.

MExt — External moment about vehicle CG
array

External moment about the vehicle CG, M_x, M_y, M_z, in the vehicle-fixed frame, in N·m. The signal array dimensions are [1x3] or [3x1].

Dependencies

To enable this port, on the **Input signals** pane, select **External moments**.

Fh — Hitch force on the body
array

Hitch force applied to the body at the hitch location, Fh_x, Fh_y, Fh_z, in the vehicle-fixed frame, in N. The signal array dimensions are [1x3] or [3x1].

Dependencies

To enable this port, under **Input signals**, select **Hitch forces**.

Mh — Hitch moment about body
array

Hitch moment at the hitch location, Mh_x, Mh_y, Mh_z, about the vehicle-fixed frame, in N·m. The signal array dimensions are [1x3] or [3x1].

Dependencies

To enable this port, under **Input signals**, select **Hitch moments**.

WindXYZ — Wind speed
array

Wind speed, W_x, W_y, W_z along the inertial X-, Y-, and Z-axes, in m/s. The signal array dimensions are [1x3] or [3x1].

Dependencies

To enable this port, on the **Input signals** pane, select **External wind**.

Mu — Tire friction coefficient
array

Tire friction coefficient, μ, dimensionless.

Vehicle Track Setting	Variable	Signal Dimension
Single (bicycle)	$Mu = \begin{bmatrix} \mu_f & \mu_m & \mu_r \end{bmatrix}$ or $\begin{bmatrix} \mu_f \\ \mu_m \\ \mu_r \end{bmatrix}$	Array – [1x3] or [3x1]
Dual	$Mu = \begin{bmatrix} \mu_{fl} & \mu_{fr} \\ \mu_{ml} & \mu_{mr} \\ \mu_{rl} & \mu_{rr} \end{bmatrix}$	Array – [3x2]

Dependencies

To enable this port, on the **Input signals** pane, select **External friction**.

AirTemp — Ambient air temperature
scalar

Ambient air temperature, in K.

Dependencies

To enable this port, on the **Input signals** pane, select **Air temperature**.

X_o — Initial longitudinal position
scalar

Initial vehicle CG displacement along the earth-fixed *X*-axis, in m.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial longitudinal position**.

Y_o — Initial lateral position
scalar

Initial vehicle CG displacement along the earth-fixed *Y*-axis, in m.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial lateral position**.

xdot_o — Initial longitudinal position
scalar

Initial vehicle CG velocity along the vehicle-fixed x-axis, in m/s.

Dependencies

To enable this port:

1   Set **Axle forces** to one of these options:

- External longitudinal forces
- External forces

2   On the **Input signals** pane, select **Initial longitudinal velocity**

ydot_o — Initial lateral position
scalar

Initial vehicle CG velocity along the vehicle-fixed y-axis, in m/s.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial lateral velocity**.

psi_o — Initial yaw angle
scalar

Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw), in rad.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial yaw angle**.

r_o — Initial yaw rate
scalar

Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate), in rad/s.

Dependencies

To enable this port, on the **Input signals** pane, select **Initial yaw rate**.

Output

`Info` — Vehicle data
bus

Vehicle data, returned as a bus signal containing these block values.

Signal					Description	Value	Units
InertFrm	Cg	Disp	X		Vehicle CG displacement along the earth-fixed *X*-axis	Computed	m
			Y		Vehicle CG displacement along the earth-fixed *Y*-axis	Computed	m
			Z		Vehicle CG displacement along the earth-fixed *Z*-axis	0	m
		Vel	Xdot		Vehicle CG velocity along the earth-fixed *X*-axis	Computed	m/s
			Ydot		Vehicle CG velocity along the earth-fixed *Y*-axis	Computed	m/s
			Zdot		Vehicle CG velocity along the earth-fixed *Z*-axis	0	m/s
		Ang	phi		Rotation of the vehicle-fixed frame about the earth-fixed *X*-axis (roll)	0	rad
			theta		Rotation of the vehicle-fixed frame about the earth-fixed *Y*-axis (pitch)	0	rad
			psi		Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw)	Computed	rad
	FrntAxl	Lft	Disp	X	Front left wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Front left wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Front left wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Front left wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Front left wheel velocity along the earth-fixed *Y*-axis	Computed	m/s

5-177

Signal					Description	Value	Units
				Zdot	Front left wheel velocity along the earth-fixed *Z*-axis	0	m/s
		Rght	Disp	X	Front right wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Front right wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Front right wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Front right wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Front right wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Front right wheel velocity along the earth-fixed *Z*-axis	0	m/s
	MidlAxl	Lft	Disp	X	Middle left wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Middle left wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Middle left wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Middle left wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Middle left wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Middle left wheel velocity along the earth-fixed *Z*-axis	0	m/s
		Rght	Disp	X	Middle right wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Middle right wheel displacement along the earth-fixed *Y*-axis	Computed	m

Signal					Description	Value	Units
				Z	Middle right wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Middle right wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Middle right wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Middle right wheel velocity along the earth-fixed *Z*-axis	0	m/s
	RearAxl	Lft	Disp	X	Rear left wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Rear left wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Rear left wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Rear left wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Rear left wheel velocity along the earth-fixed *Y*-axis	Computed	m/s
				Zdot	Rear left wheel velocity along the earth-fixed *Z*-axis	0	m/s
		Rght	Disp	X	Rear right wheel displacement along the earth-fixed *X*-axis	Computed	m
				Y	Rear right wheel displacement along the earth-fixed *Y*-axis	Computed	m
				Z	Rear right wheel displacement along the earth-fixed *Z*-axis	0	m
			Vel	Xdot	Rear right wheel velocity along the earth-fixed *X*-axis	Computed	m/s
				Ydot	Rear right wheel velocity along the earth-fixed *Y*-axis	Computed	m/s

Signal					Description	Value	Units
				Zdot	Rear right wheel velocity along the earth-fixed Z-axis	0	m/s
	Geom	Disp	X		Vehicle chassis offset from axle plane along the earth-fixed X-axis	Computed	m
			Y		Vehicle chassis offset from center plane along the earth-fixed Y-axis	Computed	m
			Z		Vehicle chassis offset from axle plane along the earth-fixed Z-axis	Computed	m
		Vel	Xdot		Vehicle chassis offset velocity along the earth-fixed X-axis	Computed	m/s
			Ydot		Vehicle chassis offset velocity along the earth-fixed Y-axis	Computed	m/s
			Zdot		Vehicle chassis offset velocity along the earth-fixed Z-axis	Computed	m/s
BdyFrm	Cg	Vel	xdot		Vehicle CG velocity along the vehicle-fixed x-axis	Computed	m/s
			ydot		Vehicle CG velocity along the vehicle-fixed y-axis	Computed	m/s
			zdot		Vehicle CG velocity along the vehicle-fixed z-axis	0	m/s
		Ang	Beta		Body slip angle, β $$\beta = \frac{V_y}{V_x}$$	Computed	rad
		AngVel	p		Vehicle angular velocity about the vehicle-fixed x-axis (roll rate)	0	rad/s
			q		Vehicle angular velocity about the vehicle-fixed y-axis (pitch rate)	0	rad/s
			r		Vehicle angular velocity about the vehicle-fixed z-axis (yaw rate)	Computed	rad/s
		Acc	ax		Vehicle CG acceleration along the vehicle-fixed x-axis	Computed	gn

Signal				Description	Value	Units
			ay	Vehicle CG acceleration along the vehicle-fixed y-axis	Computed	gn
			az	Vehicle CG acceleration along the vehicle-fixed z-axis	0	gn
			xddot	Vehicle CG acceleration along the vehicle-fixed x-axis	Computed	m/s^2
			yddot	Vehicle CG acceleration along the vehicle-fixed y-axis	Computed	m/s^2
			zddot	Vehicle CG acceleration along the vehicle-fixed z-axis	0	m/s^2
		AngAcc	pdot	Vehicle angular acceleration about the vehicle-fixed x-axis	0	rad/s
			qdot	Vehicle angular acceleration about the vehicle-fixed y-axis	0	rad/s
			rdot	Vehicle angular acceleration about the vehicle-fixed z-axis	Computed	rad/s
	Forces	Body	Fx	Net force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy	Net force on vehicle CG along the vehicle-fixed y-axis	Computed	N
			Fz	Net force on vehicle CG along the vehicle-fixed z-axis	0	N
		Ext	Fx	External force on vehicle CG along the vehicle-fixed x-axis	Computed	N
			Fy	External force on vehicle CG along the vehicle-fixed y-axis	Computed	N
			Fz	External force on vehicle CG along the vehicle-fixed z-axis	0	N
		Hitch	Fx	Hitch moment at the hitch location about vehicle-fixed z-axis	Computed	N

Signal				Description	Value	Units
			Fy	Hitch force applied to body at the hitch location along the vehicle-fixed *y*-axis	Computed	N
			Fz	Hitch force applied to body at the hitch location along the vehicle-fixed *z*-axis	Computed	N
		FrntAxl	Lft	Fx	Longitudinal force on left front wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on left front wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on left front wheel along the vehicle-fixed *z*-axis	Computed	N
			Rght	Fx	Longitudinal force on right front wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on right front wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on right front wheel along the vehicle-fixed *z*-axis	Computed	N
		MidlAxl	Lft	Fx	Longitudinal force on left middle wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on left middle wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on left middle wheel along the vehicle-fixed *z*-axis	Computed	N
			Rght	Fx	Longitudinal force on right middle wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy	Lateral force on right middle wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz	Normal force on right middle wheel along the vehicle-fixed *z*-axis	Computed	N

Signal						Description	Value	Units
		RearAxl	Lft	Fx		Longitudinal force on left rear wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy		Lateral force on left rear wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz		Normal force on left rear wheel along the vehicle-fixed *z*-axis	Computed	N
			Rght	Fx		Longitudinal force on right rear wheel along the vehicle-fixed *x*-axis	Computed	N
				Fy		Lateral force on right rear wheel along the vehicle-fixed *y*-axis	Computed	N
				Fz		Normal force on right rear wheel along the vehicle-fixed *z*-axis	Computed	N
		Tires	FrntTires	Lft	Fx	Front left tire force along the vehicle-fixed *x*-axis	Computed	N
					Fy	Front left tire force along the vehicle-fixed *y*-axis	Computed	N
					Fz	Front left tire force along the vehicle-fixed *z*-axis	Computed	N
				Rght	Fx	Front right tire force along the vehicle-fixed *x*-axis	Computed	N
					Fy	Front right tire force along the vehicle-fixed *y*-axis	Computed	N
					Fz	Front right tire force along the vehicle-fixed *z*-axis	Computed	N
			RearTires	Lft	Fx	Rear left tire force along the vehicle-fixed *x*-axis	Computed	N
					Fy	Rear left tire force along the vehicle-fixed *y*-axis	Computed	N
					Fz	Rear left tire force along the vehicle-fixed *z*-axis	Computed	N
				Rght	Fx	Rear right tire force along the vehicle-fixed *x*-axis	Computed	N
					Fy	Rear right tire force along the vehicle-fixed *y*-axis	Computed	N

Signal						Description	Value	Units
					F z	Rear right tire force along the vehicle-fixed *z*-axis	Computed	
		Drag	Fx			Drag force on vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy			Drag force on vehicle CG along the vehicle-fixed *y*-axis	Computed	N
			Fz			Drag force on vehicle CG along the vehicle-fixed *z*-axis	Computed	N
		Grvty	Fx			Gravity force on vehicle CG along the vehicle-fixed *x*-axis	Computed	N
			Fy			Gravity force on vehicle CG along the vehicle-fixed *y*-axis	Computed	N
			Fz			Gravity force on vehicle CG along the vehicle-fixed *z*-axis	Computed	N
	Moments	Body	Mx			Body moment on vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My			Body moment on vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz			Body moment on vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Drag	Mx			Drag moment on vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My			Drag moment on vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m
			Mz			Drag moment on vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Ext	Mx			External moment on vehicle CG about the vehicle-fixed *x*-axis	0	N·m
			My			External moment on vehicle CG about the vehicle-fixed *y*-axis	Computed	N·m

Signal					Description	Value	Units
			Mz		External moment on vehicle CG about the vehicle-fixed *z*-axis	0	N·m
		Hitch	Mx		Hitch moment at the hitch location about vehicle-fixed *x*-axis	0	N·m
			My		Hitch moment at the hitch location about vehicle-fixed *y*-axis	Computed	N·m
			Mz		Hitch moment at the hitch location about vehicle-fixed *z*-axis	0	N·m
	FrntAxl	Lft	Disp	x	Front left wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Front left wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Front left wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Front left wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Front left wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Front left wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Rght	Disp	x	Front right wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Front right wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Front right wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Front right wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Front right wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s

Signal					Description	Value	Units
				zdot	Front right wheel velocity along the vehicle-fixed z-axis	0	m/s
		Steer	WhlAngFL		Front left wheel steering angle	Computed	rad
			WhlAngFR		Front right wheel steering angle	Computed	rad
	MidlAxl	Lft	Disp	x	Middle left wheel displacement along the vehicle-fixed x-axis	Computed	m
				y	Middle left wheel displacement along the vehicle-fixed y-axis	Computed	m
				z	Middle left wheel displacement along the vehicle-fixed z-axis	Computed	m
			Vel	xdot	Middle left wheel velocity along the vehicle-fixed x-axis	Computed	m/s
				ydot	Middle left wheel velocity along the vehicle-fixed y-axis	Computed	m/s
				zdot	Middle left wheel velocity along the vehicle-fixed z-axis	0	m/s
		Rght	Disp	x	Middle right wheel displacement along the vehicle-fixed x-axis	Computed	m
				y	Middle right wheel displacement along the vehicle-fixed y-axis	Computed	m
				z	Middle right wheel displacement along the vehicle-fixed z-axis	Computed	m
			Vel	xdot	Middle right wheel velocity along the vehicle-fixed x-axis	Computed	m/s
				ydot	Middle right wheel velocity along the vehicle-fixed y-axis	Computed	m/s
				zdot	Middle right wheel velocity along the vehicle-fixed z-axis	0	m/s

Signal					Description	Value	Units
		Steer	WhlAngRL		Middle left wheel steering angle	Computed	rad
			WhlAngRR		Middle right wheel steering angle	Computed	rad
	RearAxl	Lft	Disp	x	Rear left wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Rear left wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Rear left wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Rear left wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Rear left wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Rear left wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Rght	Disp	x	Rear right wheel displacement along the vehicle-fixed *x*-axis	Computed	m
				y	Rear right wheel displacement along the vehicle-fixed *y*-axis	Computed	m
				z	Rear right wheel displacement along the vehicle-fixed *z*-axis	Computed	m
			Vel	xdot	Rear right wheel velocity along the vehicle-fixed *x*-axis	Computed	m/s
				ydot	Rear right wheel velocity along the vehicle-fixed *y*-axis	Computed	m/s
				zdot	Rear right wheel velocity along the vehicle-fixed *z*-axis	0	m/s
		Steer	WhlAngRL		Rear left wheel steering angle	Computed	rad
			WhlAngRR		Rear right wheel steering angle	Computed	rad

Signal				Description	Value	Units
	Pwr	PwrExt		Applied external power	Computed	W
		PwrHtch		Hitch power	Computed	
		Drag		Power loss due to drag	Computed	W
	Geom	Disp	x	Vehicle chassis offset from axle plane along the vehicle-fixed *x*-axis	Input	m
			y	Vehicle chassis offset from center plane along the vehicle-fixed *y*-axis	Input	m
			z	Vehicle chassis offset from axle plane along the earth-fixed *z*-axis	Input	m
		Vel	xdot	Vehicle chassis offset velocity along the vehicle-fixed *x*-axis	Computed	m/s
			ydot	Vehicle chassis offset velocity along the vehicle-fixed *y*-axis	Computed	m/s
			zdot	Vehicle chassis offset velocity along the vehicle-fixed *z*-axis	0	m/s
		Ang	Beta	Body slip angle, β $$\beta = \frac{V_y}{V_x}$$	Computed	rad

xdot — Vehicle longitudinal velocity
scalar

Vehicle CG velocity along the vehicle-fixed *x*-axis, in m/s.

ydot — Vehicle lateral velocity
scalar

Vehicle CG velocity along the vehicle-fixed *y*-axis, in m/s.

psi — Yaw
scalar

Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw), in rad.

r — Yaw rate
scalar

Vehicle angular velocity, r, about the vehicle-fixed *z*-axis (yaw rate), in rad/s.

FzF — Normal force on front wheels
scalar | array

Normal force on the front wheels, Fz_F, along the vehicle-fixed z-axis, in N.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single (bicycle)	Normal force on front axle	$FzF = Fz_f$	Scalar – 1
Dual	Normal force on the right and left front wheels	$FzF = [Fz_{fl}\ Fz_{fr}]$	Array – [1x2]

FzM — Normal force on middle wheels
scalar | array

Normal force on the middle wheels, Fz_M, along the vehicle-fixed z-axis, in N.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single (bicycle)	Normal force on middle axle	$FzM = Fz_m$	Scalar – 1
Dual	Normal force on the right and left middle wheels	$FzM = [Fz_{ml}\ Fz_{rl}]$	Array – [1x2]

FzR — Normal force on rear wheels
scalar | array

Normal force on the rear wheels, Fz_R, along the vehicle-fixed z-axis, in N.

Vehicle Track Setting	Description	Variable	Signal Dimension
Single (bicycle)	Normal force on rear wheel	$FzR = Fz_r$	Scalar – 1
Dual	Normal force on the right and left rear wheels	$FzR = [Fz_{rl}\ Fz_{rr}]$	Array – [1x2]

Parameters

Options

Vehicle track — Number of vehicle tracks
Dual (default) | Single (bicycle)

Use the **Vehicle track** parameter to specify the number of tracks.

Vehicle Track Setting	Implementation
Single (bicycle)	• Forces act along the center line of the axles. • No lateral load transfer.
Dual	Forces act at the axle hard-point locations.

Axle forces — Type of axle force
External longitudinal velocity (default) | External longitudinal forces | External forces

Use the **Axle forces** parameter to specify the type of force.

Axle Forces Setting	Implementation
External longitudinal velocity	• The block assumes that the external longitudinal velocity is in a quasi-steady state, so the longitudinal acceleration is approximately zero. • Because the motion is quasi-steady, the block calculates lateral forces using the tire slip angles and linear cornering stiffness. • Consider this setting when you want to: • Generate virtual sensor signal data. • Conduct high-level software studies that are not impacted by driveline or nonlinear tire responses.
External longitudinal forces	• The block uses the external longitudinal force to accelerate or brake the vehicle. • The block calculates lateral forces using the tire slip angles and linear cornering stiffness. • Consider this setting when you want to: • Account for changes in the longitudinal velocity on the lateral and yaw motion. • Specify the external longitudinal motion through a force instead of an external longitudinal velocity. • Connect the block to tractive actuators, wheels, brakes, and hitches.
External forces	• The block uses the external lateral and longitudinal forces to steer, accelerate, or brake the vehicle. • The block does not use the steering input to calculate vehicle motion. • Consider this setting when you need tire models with more accurate nonlinear combined lateral and longitudinal slip.

Input Signals

Front wheel steering — WhlAngF input port
on (default) | off

Select to create input port WhlAngF.

Middle wheel steering — WhlAngM input port
off (default) | on

Select to create input port WhlAngM.

Rear wheel steering — WhlAngR input port
off (default) | on

Select to create input port WhlAngR.

External wind — WindXYZ input port
off (default) | on

Select to create input port WindXYZ.

External friction — Mu input port
off (default) | on

Select to create input port Mu.

Dependencies

To enable this parameter, set **Axle forces** to External longitudinal forces or External forces.

External forces — FExt input port
off (default) | on

Select to create input port FExt.

External moments — MExt input port
off (default) | on

Select to create input port MExt.

Hitch forces — Fh input port
on (default) | off

Select to create input port Fh.

Hitch moments — Mh input port
on (default) | off

Specify to create input port Mh.

Initial longitudinal position — X_o input port
off (default) | on

Specify to create input port X_o.

Initial yaw angle — psi_o input port
off (default) | on

Specify to create input port psi_o.

Initial longitudinal velocity — xdot_o input port
off (default) | on

Specify to create input port xdot_o.

Dependencies

To enable this parameter, set **Axle forces** to External longitudinal forces or External forces.

5-191

Initial yaw rate — r_o input port
off (default) | on

Specify to create input port r_o.

Initial lateral position — Y_o input port
off (default) | on

Specify to create input port Y_o.

Air temperature — AirTemp input port
off (default) | on

Specify to create input port AirTemp.

Initial lateral velocity — ydot_o input port
off (default) | on

Specify to create input port ydot_o.

Longitudinal

Number of wheels on front axle, NF — Front wheel count
2 (default) | scalar

Number of wheels on the front axle, N_F, dimensionless.

Number of wheels on middle axle, NM — Middle wheel count
2 (default) | scalar

Number of wheels on the middle axle, N_M, dimensionless.

Number of wheels on rear axle, NR — Rear wheel count
2 (default) | scalar

Number of wheels on the rear axle, N_R, dimensionless.

Vehicle mass, m — Vehicle mass
47000 (default) | scalar

Vehicle mass, m, in kg.

Longitudinal distance from center of mass to front axle, a — Distance from CM to front axle
0.5 (default) | scalar
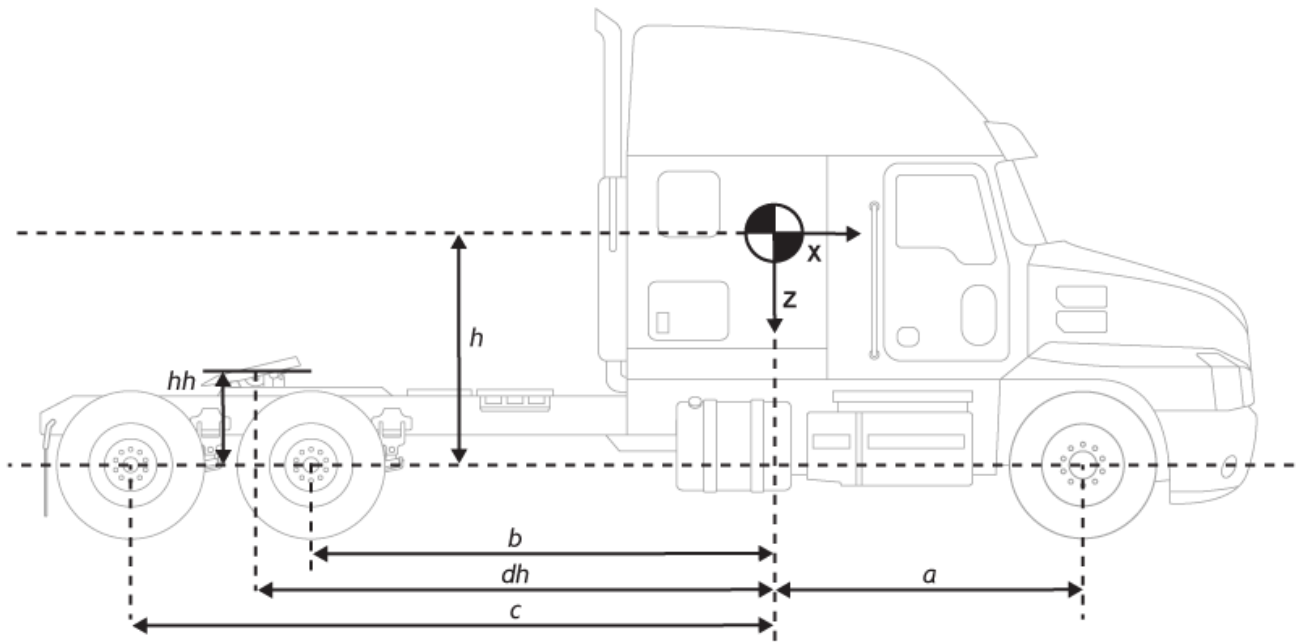
Distance from vehicle CM to front axle, a, in m.

Longitudinal distance from center of mass to middle axle, b — Distance from CM to middle axle
4.5 (default) | scalar

Distance from vehicle CM to middle axle, *b*, in m.
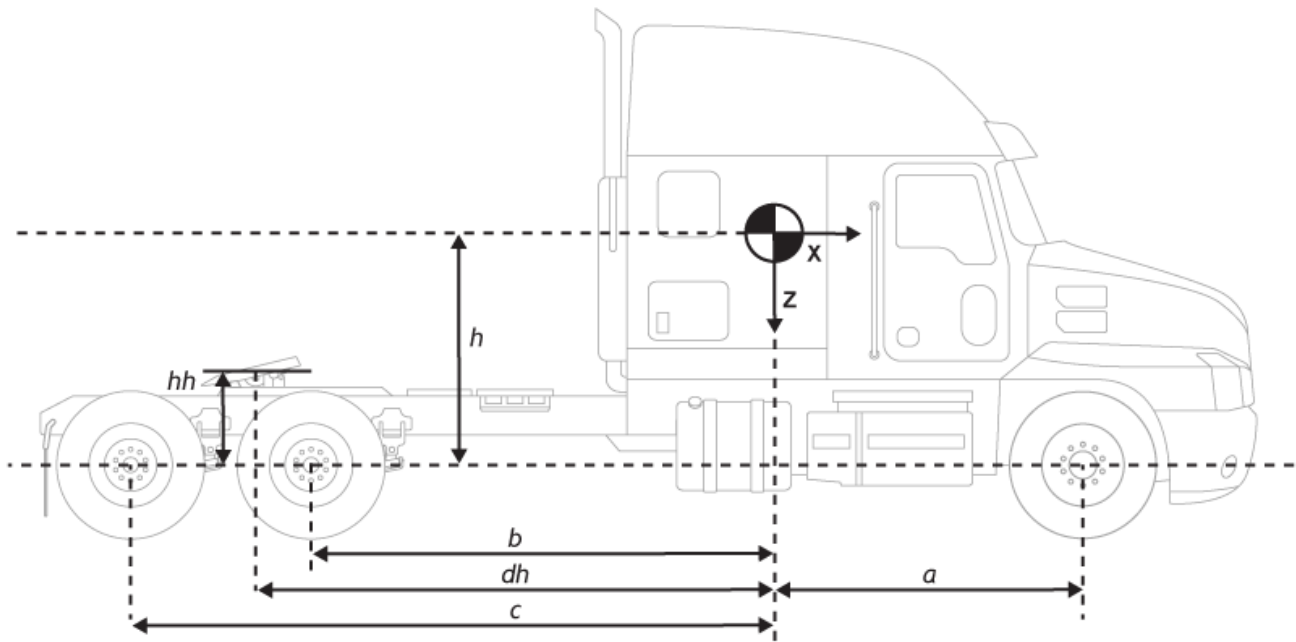
Longitudinal distance from center of mass to rear axle, c — Distance from CM to rear axle
5.7 (default) | scalar

Distance from vehicle CM to rear axle, *c*, in m.
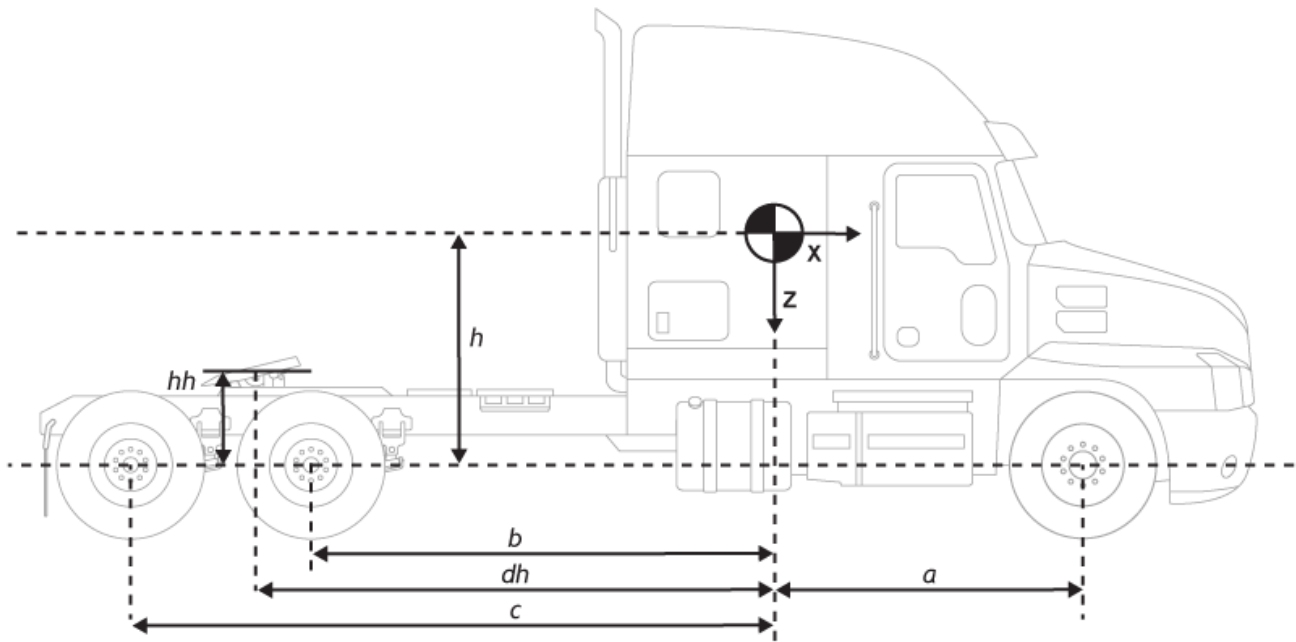
Vertical distance from center of mass to axle plane, h — Distance from CM to axle plane
0.3 (default) | scalar

Vertical distance from vehicle CM to axle plane, h, in m.
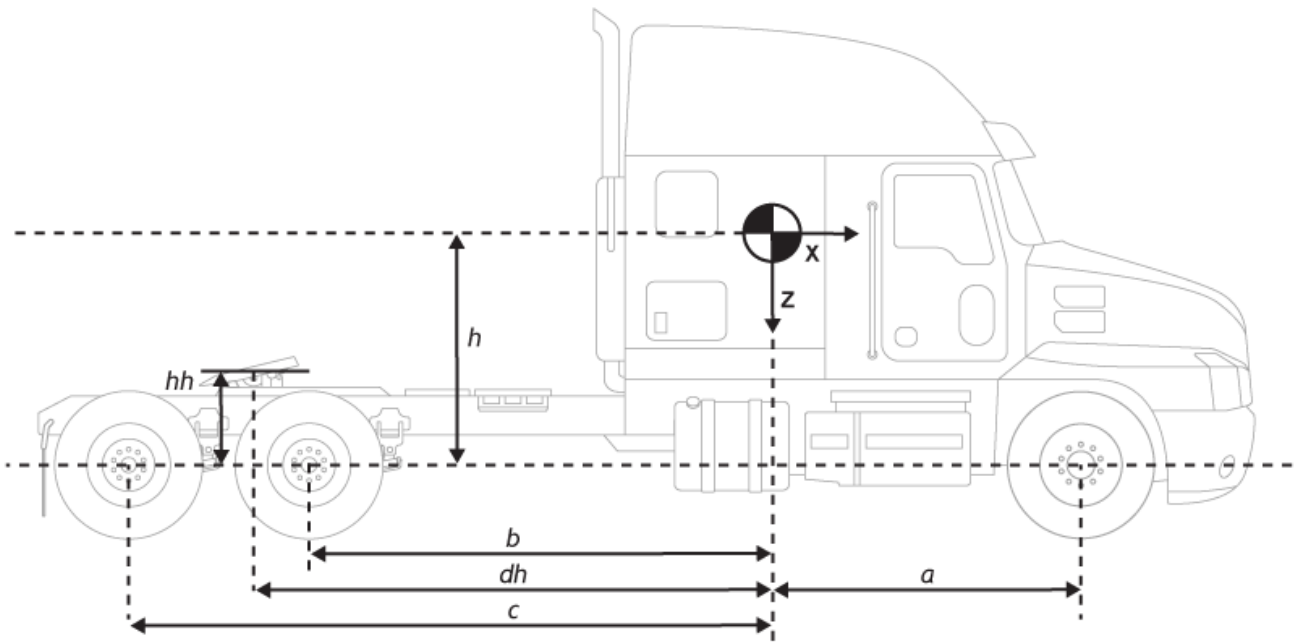
Vertical distance from hitch to axle plane, hh — Distance from hitch to axle plane
0.5 (default) | scalar

Vertical distance from hitch to axle plane, *hh*, in m.
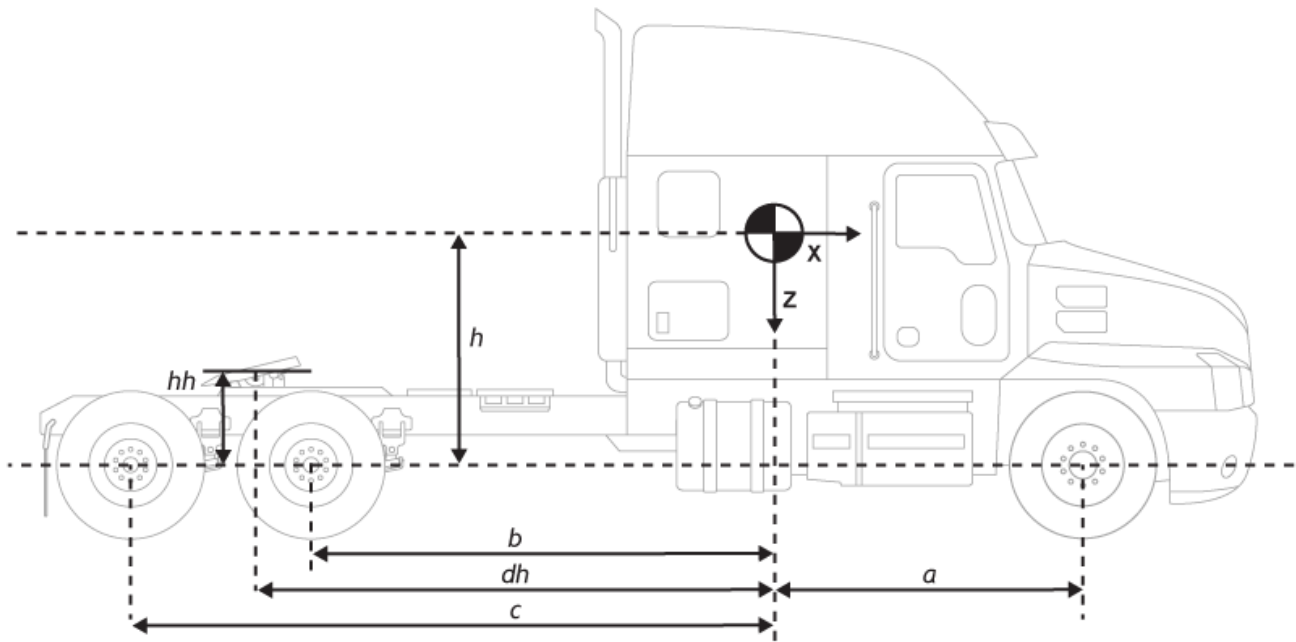
Dependencies

To enable this parameter, on the **Input signals** pane, select **Hitch forces** or **Hitch moments**.

Longitudinal distance from center of mass to hitch, dh — Distance from CM to hitch
5 (default) | `scalar`

Longitudinal distance from center of mass to hitch, *dh*, in m.
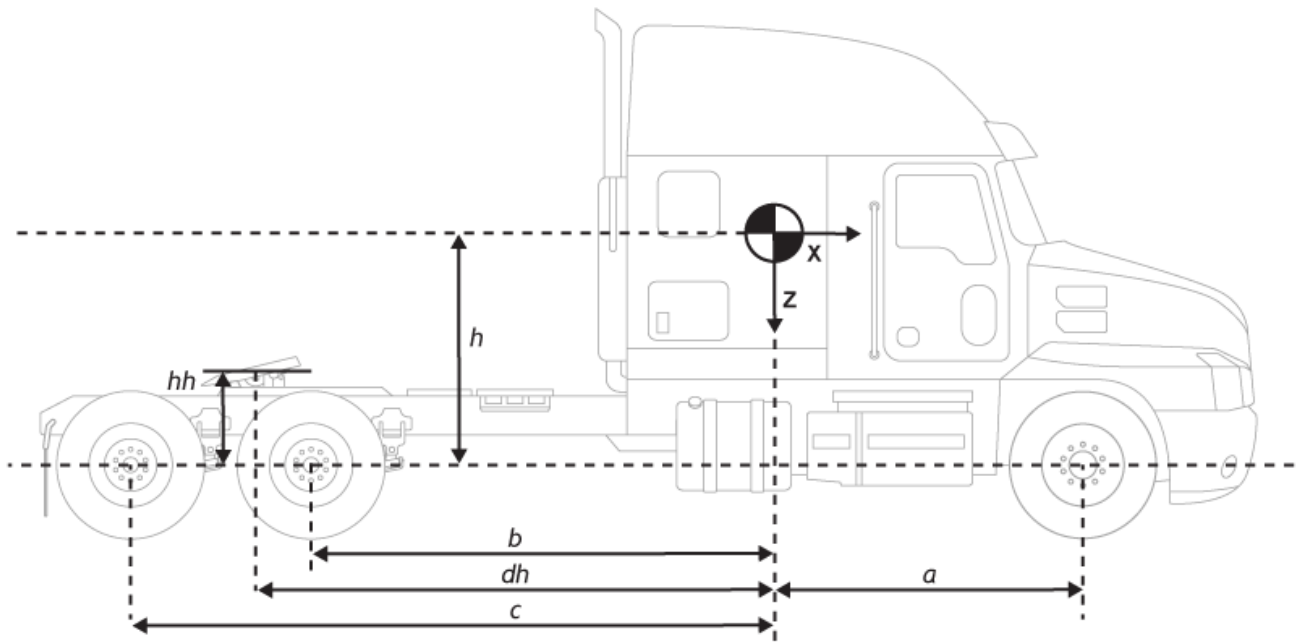
Dependencies

To enable this parameter, on the **Input signals** pane, select **Hitch forces** or **Hitch moments**.

Initial inertial frame longitudinal position, X_o — Initial longitudinal displacement
0 (default) | scalar

Initial vehicle CG displacement along the earth-fixed *X*-axis, in m.

Initial longitudinal velocity, xdot_o — Initial longitudinal velocity
0 (default) | scalar

Initial vehicle CG velocity along the vehicle-fixed *x*-axis, in m/s.

Dependencies

To enable this parameter, set **Axle forces** to one of these options:

* External longitudinal forces
* External forces

Lateral

Mapped corner stiffness — Enable mapped corner stiffness
off (default) | on

Enables mapped corner stiffness calculation.

Dependencies

To enable this parameter, set **Axle forces** to one of these options:

- `External longitudinal velocity`
- `External longitudinal forces`

`Include relaxation length dynamics` — Enable relaxation length dynamics
`on` (default) | `off`

Enables relaxation length dynamics.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

   - `External longitudinal velocity`
   - `External longitudinal forces`

2   Clear **Mapped corner stiffness**.

`Lateral distance from geometric centerline to center of mass, d` — Distance from centerline to CM
`0` (default) | `scalar`

Lateral distance from the geometric centerline to the center of mass, d, in m, along the vehicle-fixed y-axis. Positive values indicate that the vehicle CM is to the right of the geometric centerline. Negative values indicate that the vehicle CM is to the left of the geometric centerline.
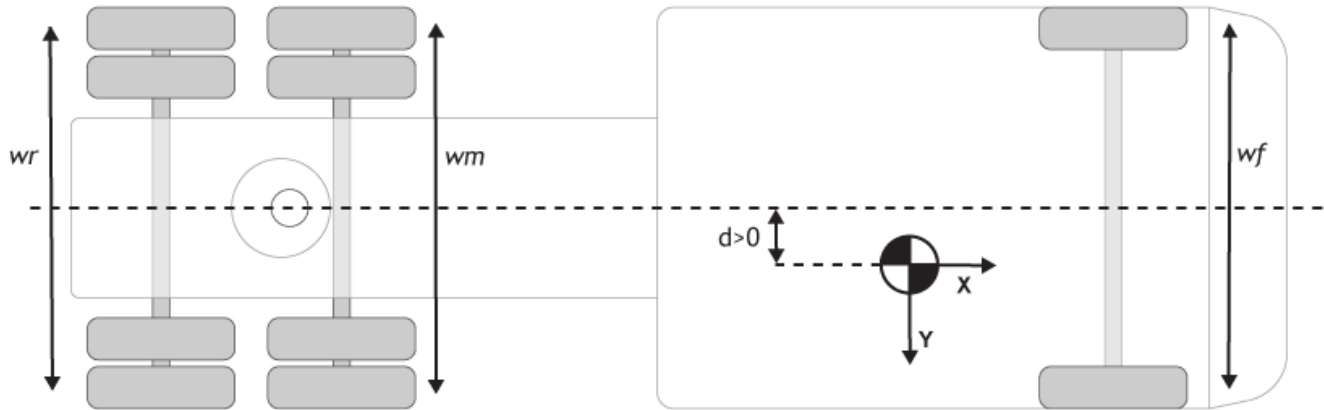
`Track width, w` — Front, middle, and rear track widths
[1.82,1.82,1.82] (default) | `vector`

Front, middle, and rear track widths, *wf*, *wm*, and, *wr*, respectively, in m. Dimensions are [1-by-3].
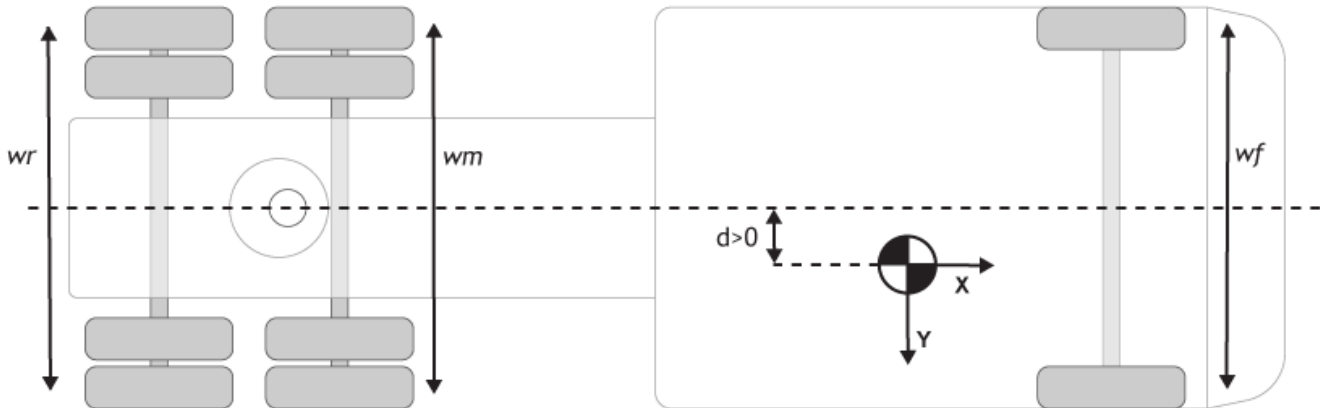
Dependencies

To enable this parameter, set **Vehicle track** to Dual.

Front axle tire corner stiffness, Cy_f — Front tire corner stiffness
12e3 | scalar

Front tire corner stiffness, Cy_f, in N/rad.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

   - External longitudinal velocity
   - External longitudinal forces

2   Clear **Mapped corner stiffness**.

Middle axle tire corner stiffness, Cy_m — Middle tire corner stiffness
11e3 | scalar

Middle axle tire corner stiffness, Cy_m, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Clear **Mapped corner stiffness**.

Rear axle tire corner stiffness, Cy_r — Rear tire corner stiffness
11e3 | scalar

Rear axle tire corner stiffness, Cy_r, in N/rad.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Clear **Mapped corner stiffness**.

Front tire(s) relaxation length, sigma_f — Front tire relaxation length
.1 (default) | scalar

Front tire relaxation length, σ_f, in m.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Do either of these:

- Select **Mapped corner stiffness**.
- Clear **Mapped corner stiffness** and select **Include relaxation length dynamics**.

Middle tire(s) relaxation length, sigma_m — Middle tire relaxation length
.1 (default) | scalar

Middle tire relaxation length, σ_m, in m.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Do either of these:

- Select **Mapped corner stiffness**.
- Clear **Mapped corner stiffness** and select **Include relaxation length dynamics**.

Rear tire(s) relaxation length, sigma_r — Rear tire relaxation length
.1 (default) | scalar

Rear tire relaxation length, σ_r, in m.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Do either of these:

- Select **Mapped corner stiffness**.
- Clear **Mapped corner stiffness** and select **Include relaxation length dynamics**.

Front axle slip angle breakpoints, alpha_f_brk — Breakpoints
[-.1 .1] (default) | vector

Front axle slip angle breakpoints, α_{fbrk}, in rad.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Select **Mapped corner stiffness**.

Front axle tire corner data, Cy_f_data — Front axle tire corner data
[-9e3 9e3] (default) | vector

Front axle tire corner data, Cy_{fdata}, in N/rad.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Select **Mapped corner stiffness**.

Middle axle slip angle breakpoints, alpha_m_brk — Breakpoints
[-.1 .1] (default) | vector

Middle axle slip angle breakpoints, α_{mbrk}, in rad.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Select **Mapped corner stiffness**.

Middle axle tire corner data, Cy_m_data — Middle axle tire corner data
[-9e3 9e3] (default) | vector

Middle axle tire corner data, Cy_{mdata}, in N/rad.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Select **Mapped corner stiffness**.

Rear axle slip angle breakpoints, alpha_r_brk — Breakpoints
[-.1 .1] (default) | vector

Rear axle slip angle breakpoints, α_{rbrk}, in rad.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Select **Mapped corner stiffness**.

Rear axle tire corner data, Cy_r_data — Rear axle tire corner data
[-9e3 9e3] (default) | vector

Rear axle tire corner data, Cy_{rdata}, in N/rad.

Dependencies

To enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Select **Mapped corner stiffness**.

Initial inertial frame lateral displacement, Y_o — Initial lateral displacement
0 (default) | scalar

Initial vehicle CG displacement along the earth-fixed *Y*-axis, in m.

Initial lateral velocity, ydot_o — Initial lateral velocity
0 (default) | scalar

Initial vehicle CG velocity along the vehicle-fixed *y*-axis, in m/s.

Yaw

Yaw polar inertia, Izz — Inertia
4000 (default) | scalar

Yaw polar inertia, in kg*m^2.

Initial yaw angle, psi_o — Psi rotation
0 (default) | scalar

Rotation of the vehicle-fixed frame about earth-fixed *Z*-axis (yaw), in rad.

Initial yaw rate, r_o — Yaw rate
0 (default) | scalar

Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate), in rad/s.

Aerodynamic

Longitudinal drag area, Af — Effective vehicle cross-sectional area
2 (default) | scalar

Effective vehicle cross-sectional area, A_f, to calculate the aerodynamic drag force on the vehicle, in m^2.

Longitudinal drag coefficient, Cd — Air drag coefficient
.3 (default) | scalar

Air drag coefficient, C_d. The value is dimensionless.

Longitudinal lift coefficient, Cl — Air lift coefficient
.1 (default) | scalar

Air lift coefficient, C_l. The value is dimensionless.

Longitudinal drag pitch moment, Cpm — Pitch drag
.1 (default) | scalar

Longitudinal drag pitch moment coefficient, C_{pm}. The value is dimensionless.

Relative wind angle vector, beta_w — Wind angle
[0:0.01:0.3] (default) | vector

Relative wind angle vector, β_w, in rad.

Side force coefficient vector, Cs — Side force coefficient
[0:0.03:0.9] (default) | vector

Side force coefficient vector coefficient, C_s. The value is dimensionless.

Yaw moment coefficient vector, Cym — Yaw moment drag
[0:0.01:0.3] (default) | vector

Yaw moment coefficient vector coefficient, C_{ym}. The value is dimensionless.

Environment

Absolute air pressure, Pabs — Pressure
101325 (default) | scalar

Environmental absolute pressure, P_{abs}, in Pa.

Air temperature, Tair — Temperature
273 (default) | scalar

Environmental absolute temperature, T, in K.

Dependencies

To enable this parameter, clear **Air temperature**.

Gravitational acceleration, g — Gravity
9.81 (default) | scalar

Gravitational acceleration, g, in m/s^2.

Nominal friction scaling factor, mu — Friction scale factor
1 (default) | scalar

Nominal friction scale factor, μ. The value is dimensionless.

Dependencies

To enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2 Clear **External Friction**.

Simulation

Longitudinal velocity tolerance, xdot_tol — Tolerance
.01 (default) | scalar

Longitudinal velocity tolerance, in m/s.

Nominal normal force, Fznom — Normal force
5000 (default) | scalar

Nominal normal force, in N.

Dependencies

To enable this parameter, set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

Geometric longitudinal offset from axle plane, longOff — Longitudinal offset
0 (default) | scalar

Vehicle chassis offset from the axle plane along the vehicle-fixed *x*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independently of the vehicle CG.

Geometric lateral offset from center plane, latOff — Lateral offset
0 (default) | scalar

Vehicle chassis offset from the center plane along the vehicle-fixed *y*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independently of the vehicle CG.

Geometric vertical offset from axle plane, vertOff — Vertical offset
0 (default) | scalar

Vehicle chassis offset from the axle plane along the vehicle-fixed *z*-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independently of the vehicle CG.

Wrap Euler angles, wrapAng — Wrap the Euler angles to the interval [-pi, pi]
off (default) | on

Wrap the Euler angles to the interval [-pi, pi]. For vehicle maneuvers that might undergo vehicle yaw rotations that are outside of this interval, consider clearing the parameter if you want to:

- Track the total vehicle yaw rotation.
- Avoid discontinuities in the vehicle state estimators.

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers (SAE), 1992.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Trailer Body 3DOF

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"

Introduced in R2020a

Three-axis Inertial Measurement Unit

Implement three-axis inertial measurement unit (IMU)
Library: Vehicle Dynamics Blockset / Sensors
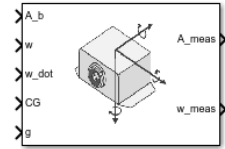
Description

The Three-Axis Inertial Measurement Unit block implements an inertial measurement unit (IMU) containing a three-axis accelerometer and a three-axis gyroscope.

For a description of the equations and application of errors, see Three-axis Accelerometer and Three-axis Gyroscope.

Limitations

*   Vibropendulous error, hysteresis affects, anisoelastic bias and anisoinertial bias are not accounted for in this block.
*   This block is not intended to model the internal dynamics of different forms of the instrument.

Ports

Input

A_b — Actual accelerations
three-element vector

Actual accelerations in body-fixed axes, specified as a three-element vector, in selected units.

Data Types: `double`

w — Angular rates
three-element vector

Angular rates in body-fixed axes, specified as a three-element vector, in radians per second.

Data Types: `double`

w_dot — Angular accelerations
three-element vector

Angular accelerations in body-fixed axes, specified as a three-element vector, in radians per second squared.

Data Types: `double`

CG — Location of center of gravity
three-element vector

Location of the center of gravity, specified as a three-element vector, in selected units.

Data Types: `double`

g — Gravity
three-element vector

Gravity in body axis, specified as a three-element vector, in selected units.

Data Types: `double`

Output

A_meas — Measured accelerations
three-element vector

Measured accelerations from the accelerometer, specified as a three-element vector, in selected units.

Data Types: `double`

w_meas — Measured angular rates
three-element vector

Measured angular rates from the gyroscope, specified as a three-element vector, in radians per second.

Data Types: `double`

Parameters

Main

Units — Units
`Metric (MKS)` (default) | `English`

Input and output units, specified as:

Units	Acceleration	Length
Metric (MKS)	Meters per second squared	Meters
English	Feet per second squared	Feet

Programmatic Use
Block Parameter: units
Type: character vector
Values: 'Metric (MKS)' | 'English'
Default: 'Metric (MKS)'

IMU location — IMU location
`[0 0 0]` (default) | three-element vector

The location of the IMU, which is also the accelerometer group location, from the vehicle center of gravity, along the vehicle-fixed axis. This measurement reference is the same for the center of gravity input. The units are in selected length units.

Programmatic Use
Block Parameter: imu
Type: character vector
Values: three-element vector
Default: '[0 0 0]'

Update rate — Update rate
0 (default) | real, double scalar

Update rate of the accelerometer and gyroscope, specified as a real, double scalar, in seconds. An update rate of 0 creates a continuous accelerometer and continuous gyroscope. If you select the **Noise on** parameter and the update rate is 0, the block updates the noise at a rate of 0.1.

Tip If you:

• Update this parameter value to 0 (continuous)
• Configure a fixed-step solver for the model

you must also select the **Automatically handle rate transition for data transfer** check box in the **Solver** pane. This check box enables the software to handle rate transitions correctly.

Programmatic Use
Block Parameter: a_Ts
Type: character vector
Values: real, double scalar
Default: '0'

Accelerometer

Second order dynamics for accelerometer — Second-order dynamics
on (default) | off

To apply second-order dynamics to acceleration readings, select this check box.

Programmatic Use
Block Parameter: dtype_a
Type: character vector
Values: 'on' | 'off'
Default: 'on'

Accelerometer natural frequency (rad/sec) — Accelerometer natural frequency
190 (default) | real, double scalar

Natural frequency of the accelerometer, specified as a real, double scalar, in radians per second.

Programmatic Use
Block Parameter: w_a
Type: character vector
Values: real, double scalar
Default: '190'

Dependencies

To enable this parameter, select **Second order dynamics for accelerometer**.

Accelerometer damping ratio — Accelerometer damping ratio
`0.707` (default) | real, double scalar

Damping ratio of the accelerometer, specified as a real, double scalar, with no dimensions.

Programmatic Use
Block Parameter: `z_a`
Type: character vector
Values: real, double scalar
Default: `'0.707'`

Dependencies

To enable this parameter, select **Second order dynamics for accelerometer**.

Accelerometer scale factor and cross-coupling — Scale factor and cross coupling
`[1 0 0; 0 1 0; 0 0 1]` (default) | 3-by-3 matrix

Scale factor and cross-coupling, specified as a 3-by-3 matrix, to skew the accelerometer from body axes and to scale accelerations along body axes.

Programmatic Use
Block Parameter: `a_sf_cc`
Type: character vector
Values: 3-by-3 matrix
Default: `'[1 0 0; 0 1 0; 0 0 1]'`

Accelerometer measurement bias — Accelerometer measurement bias
`[0 0 0]` (default) | three-element vector

Long-term biases along the accelerometer axes, specified as a three-element vector, in selected acceleration units.

Programmatic Use
Block Parameter: `a_bias`
Type: character vector
Values: three-element vector
Default: `'[0 0 0]'`

Accelerometer upper and lower limits — Minimum and maximum values of acceleration
`[-inf -inf -inf inf inf inf]` (default) | six-element vector

Three minimum values and three maximum values of acceleration in each of accelerometer axes, specified as a six-element vector, in selected acceleration units.

Programmatic Use
Block Parameter: `a_sat`
Type: character vector
Values: six-element vector
Default: `'[-inf -inf -inf inf inf inf]'`

Gyroscope

Second-order dynamics for gyro — Gyroscope second-order dynamics
`on` (default) | `off`

To apply second-order dynamics to gyroscope readings, select this check box.

Programmatic Use
Block Parameter: dtype_g
Type: character vector
Values: 'on' | 'off'
Default: 'on'

Gyro natural frequency (rad/sec) — Gyroscope natural frequency
190 (default) | real, double scalar

Natural frequency of the gyroscope, specified as a real, double scalar, in radians per second.

Programmatic Use
Block Parameter: w_g
Type: character vector
Values: real, double scalar
Default: '190'

Dependencies

To enable this parameter, select **Second-order dynamics for gyro**.

Gyro damping ratio — Gyroscope damping ratio
0.707 (default) | real, double scalar

Damping ratio of the gyroscope, specified as a real, double scale, with no dimensions.

Programmatic Use
Block Parameter: z_g
Type: character vector
Values: real, double scalar
Default: '0.707'

Dependencies

To enable this parameter, select **Second-order dynamics for gyro**.

Gyro scale factors and cross-coupling — Gyroscope scale factors and cross-coupling
[1 0 0; 0 1 0; 0 0 1] (default) | 3-by-3 matrix

Gyroscope scale factors and cross-coupling, specified as a 3-by-3 matrix, to skew the gyroscope from body axes and to scale angular rates along body axes.

Programmatic Use
Block Parameter: g_sf_cc
Type: character vector
Values: 3-by-3 matrix
Default: '[1 0 0; 0 1 0; 0 0 1]'

Gyro measurement bias — Gyroscope measurement bias
[0 0 0] (default) | three-element vector

Long-term biases along the gyroscope axes, specified a three-element vector, in radians per second.

Programmatic Use
Block Parameter: g_bias

Type: character vector
Values: three-element vector
Default: '[0 0 0]'

G-sensitive bias — Maximum change in rates

[0 0 0] (default) | three-element vector

Maximum change in rates due to linear acceleration, specified as a three-element vector, in radians per second per g-unit.

Programmatic Use
Block Parameter: g_sens
Type: character vector
Values: three-element vector
Default: '[0 0 0]'

Gyro upper and lower limits — Minimum and maximum values of angular rates

[-inf -inf -inf inf inf inf] (default) | six-element vector

Three minimum values and three maximum values of angular rates in each of the gyroscope axes, specified as a six-element vector, in radians per second.

Programmatic Use
Block Parameter: g_sat
Type: character vector
Values: six-element vector
Default: '[-inf -inf -inf inf inf inf]'

Noise

Noise on — White noise

on (default) | off

To apply white noise to acceleration and gyroscope readings, select this check box.

Programmatic Use
Block Parameter: a_rand
Type: character vector
Values: 'on' | 'off'
Default: 'on'

Noise seeds — Noise seeds

[23093 23094 23095 23096 23097 23098] (default) | six-element vector

Scalar seeds for the Gaussian noise generator for each axis of the accelerometer and gyroscope, specified as a six-element vector.

Programmatic Use
Block Parameter: a_seeds
Type: character vector
Values: six-element vector
Default: '[23093 23094 23095 23096 23097 23098]'

Dependencies

To enable this parameter, select **Noise on**.

Noise power — Noise power
[0.001 0.001 0.001 0.0001 0.0001 0.0001] (default) | six-element vector

Height of the power spectral density (PSD) of the white noise for each axis of the accelerometer and gyroscope, specified as a six-element vector, in:

- (m/s^2)/Hz for `Metric (MKS)`
- (ft/s^2)/Hz for `English`

Programmatic Use
Block Parameter: a_pow
Type: character vector
Values: six-element vector
Default: '[0.001 0.001 0.001 0.0001 0.0001 0.0001]'

Dependencies

To enable this parameter, select **Noise on**.

References

[1] Rogers, R. M., *Applied Mathematics in Integrated Navigation Systems*, AIAA Education Series, 2000.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Three-axis Accelerometer | Three-axis Gyroscope

Introduced in R2020a

Vehicle Scenario Blocks

Drive Cycle Source

Standard or specified longitudinal drive cycle

Library: Powertrain Blockset / Vehicle Scenario Builder
         Vehicle Dynamics Blockset / Vehicle Scenarios / Drive Cycle
         and Maneuvers

FTP75 (2474 seconds)
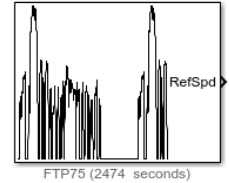
Description

The Drive Cycle Source block generates a standard or user-specified longitudinal drive cycle. The block output is the specified vehicle longitudinal speed, which you can use to:

- Predict the engine torque and fuel consumption that a vehicle requires to achieve desired speed and acceleration for a given gear shift reference.

- Produce realistic velocity and shift references for closed loop acceleration and braking commands for vehicle control and plant models.

- Study, tune, and optimize vehicle control, system performance, and system robustness over multiple drive cycles.

- Identify the faults within tolerances specified by standardized tests, including:

    - EPA dynamometer driving schedules[1]

    - Worldwide Harmonised Light Vehicle Test Procedure (WLTP) laboratory tests[2]

For the drive cycles, you can use:

- Drive cycles from predefined sources. By default, the block includes the FTP–75 drive cycle. To install additional drive cycles from a support package, see "Support Package For Maneuver and Drive Cycle Data". The support package has drive cycles that include the gear shift schedules, for example JC08 and CUEDC.

- Workspace variables.

- .mat, .xls, .xlsx, or .txt files.

- Wide open throttle (WOT) parameters, including initial and nominal reference speed, deceleration start time, and final reference speed.

To achieve the goals listed in the table, use the specified Drive Cycle Source block parameter options.

Goal	Action
Repeat the drive cycle if the simulation run time exceeds the drive cycle length.	Select **Repeat cyclically**.
Output the acceleration, as calculated by Savitzky-Golay differentiation.	Select **Output acceleration**.

Goal	Action
Specify a sample period for discrete applications.	Specify a **Output sample period (0 for continuous), dt** parameter.
Update the simulation run time so that it equals the length of the drive cycle.	Click **Update simulation time**. If a model configuration reference exists, the block does not enable this option.
Plot the drive cycle in a MATLAB® figure.	Click **Plot drive cycle**.
Specify the drive cycle using a workspace variable.	Click **Specify variable**. The block: • Sets the **Drive cycle source** parameter to `Workspace variable`. • Enables the **From workspace** parameter. Specify the workspace variable so that it contains time, velocity, and, optionally, the gear shift schedule.
Specify the drive cycle using a file.	Click **Select file**. The block: • Sets the **Drive cycle source** parameter to `.mat, .xls, .xlsx or .txt file`. • Enables the **Drive cycle source file** parameter. Specify a file that contains time, velocity, and, optionally, the gear shift schedule.
Output drive cycle gear.	Specify a drive cycle that contains a gear shift schedule. You can use: • A support package to install standard drive cycles that include the gear shift schedules, for example `JC08` and `CUEDC`. • Workspace variables. • .mat, .xls, .xlsx, or .txt files. Click **Output gear shift data**.
Install additional drive cycles from a support package.	Click **Install additional drive cycles**. The block enables the parameter if you can install additional drive cycles from a support package.
Identify drive cycle faults within tolerances specified by standardized tests.	On the **Fault Tracking** tab, use the parameters to specify the fault tolerances. If the vehicle speed is not within the allowable speed range, the block sets a fault condition.

Fault and Failure Tracking

On the **Fault Tracking** tab, use the parameters to specify the fault tolerances. If the vehicle speed or time is not within the allowable range, the block sets a fault condition.

Parameter	Description	Setting	
		EPA Standard[1]	**WLTP Tests**[2]
Speed tolerance	Speed tolerance above the highest point and below the lowest point of the drive cycle speed trace within the time tolerance.	2.0 mph	2.0 km/h
Time tolerance	Time that the block uses to determine the speed tolerance.	1.0 s	1.0 s
Maximum number of faults	Maximum number of faults during the drive cycle.	*Not specified*	10
Maximum single fault time	Maximum fault duration.	2.0 s	1.0 s
Maximum total fault time	Maximum accumulated time spent under fault condition.	*Not specified*	*Not specified*

These figures illustrate how the block uses the velocity and time tolerances to determine the allowable speed range.
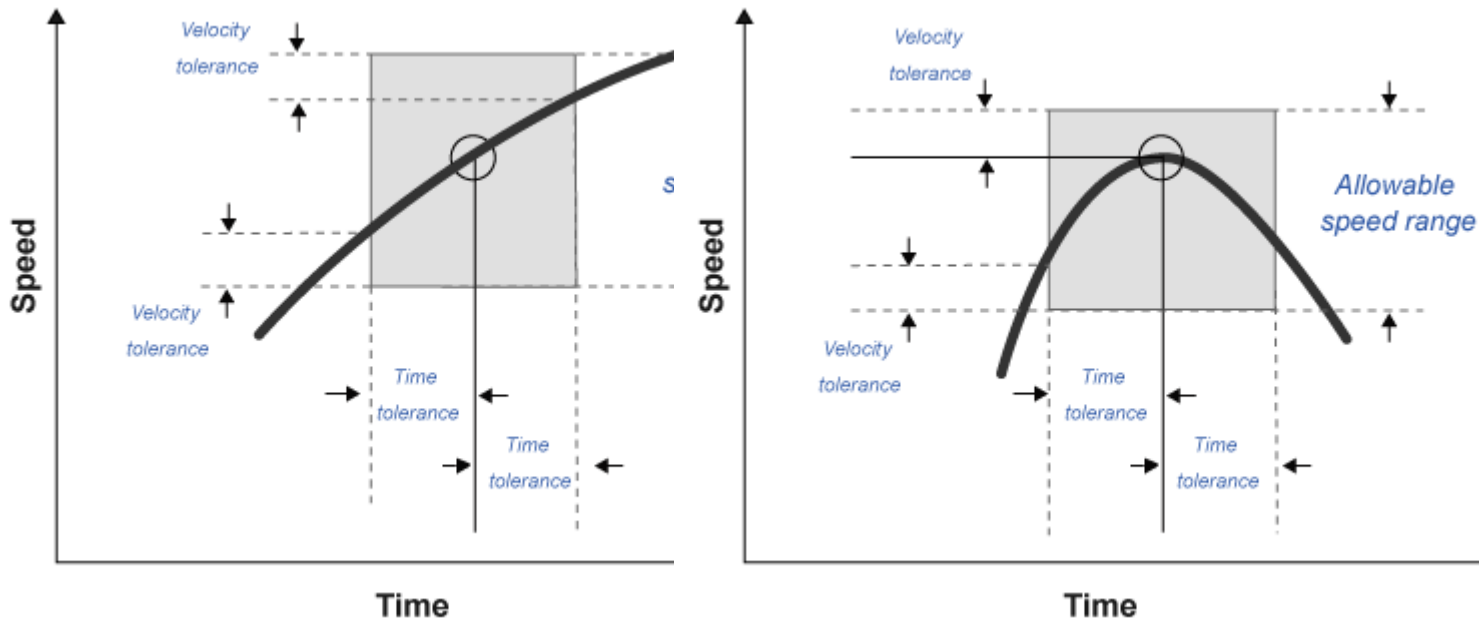
Ports

Input

VelFdbk — Vehicle longitudinal speed
scalar

Longitudinal vehicle speed.

Dependencies

To enable this port, on the **Fault Tracking** tab, select **Enable fault tracking**. Set the **Velocity feedback units, inUnit** parameter to the VelFdbk input port signal units.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal		Description
Reference Speed		Vehicle reference speed
Reference Accel		Vehicle reference acceleration
Gear		Vehicle gear
Fault	UpprBnd	Upper bound of allowable vehicle speed range.
	LowerBnd	Lower bound of allowable vehicle speed range.
	Fault	Boolean value indicating fault condition: • 1 — Fault • 0 — No fault If the vehicle speed is not within the allowable speed range, the block sets a fault condition.
	FaultCnt	Number of faults.
	CumFaultTime	Cumulative time spent in fault condition.
	SnglFaultTime	Tim spent in a single fault.
	Fail	Boolean value indicating fault failure: • 1 — Failure • 0 — No failure If the fault conditions exceed the maximum number of faults, maximum single fault time, or maximum total fault time, the block sets a fault failure.

Dependencies

To enable this port, on the **Fault Tracking** tab, select **Enable fault tracking**.

RefSpd — Vehicle reference speed
scalar

Vehicle reference speed, in units that you specify. To specify the units, use the **Output velocity units** parameter.

RefAcc — Vehicle reference acceleration
scalar

To calculate the acceleration, the block implements Savitzky-Golay differentiation using a second-order polynomial with a three-sample point filter.

Dependencies

To create the output acceleration port, select **Output acceleration**. Selecting **Output acceleration** enables the **Output acceleration units** parameter.

Gear — Vehicle gear
scalar

Dependencies

To enable this port:

1   Specify a drive cycle that contains a gear shift schedule. You can use:

- A support package to install standard drive cycles that include the gear shift schedules, for example JC08 and CUEDC.
- Workspace variables.
- .mat, .xls, .xlsx, or .txt files.

2   Select **Output gear shift** data.

Parameters

Cycle Setup

Setup

Drive cycle source — Select the drive cycle source
FTP75 (default) | Wide Open Throttle (WOT) | Workspace variable | .mat, .xls, .xlsx or .txt file

- FTP75 — Load the FTP75 drive cycle from a .mat file into a 1-D Lookup Table block. The FTP75 represents a city drive cycle that you can use to determine tailpipe emissions and fuel economy of passenger cars. To install additional drive cycles from a support package, see "Support Package For Maneuver and Drive Cycle Data".
- Wide Open Throttle (WOT) — Use WOT parameters to specify a drive cycle for performance testing.
- Workspace variable — Specify time, speed, and, optionally, gear data as a structure, 2-D array, or time series object.
- .mat, .xls, .xlsx or .txt file — Specify a file that contains time, speed and, optionally, gear data in column format.

Once you have installed additional cycles, you can use `set_param` to set the drive cycle. For example, to use drive cycle `US06`:

```
set_param([gcs '/Drive Cycle Source'],'cycleVar','US06')
```

Dependencies

The table summarizes the parameter dependencies.
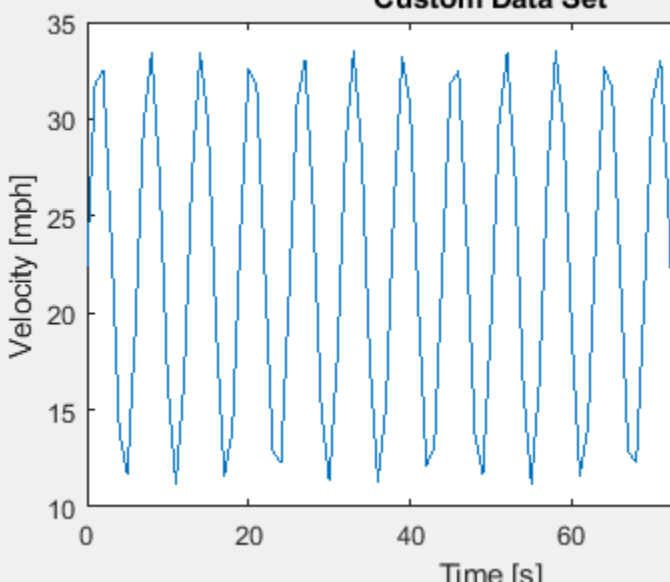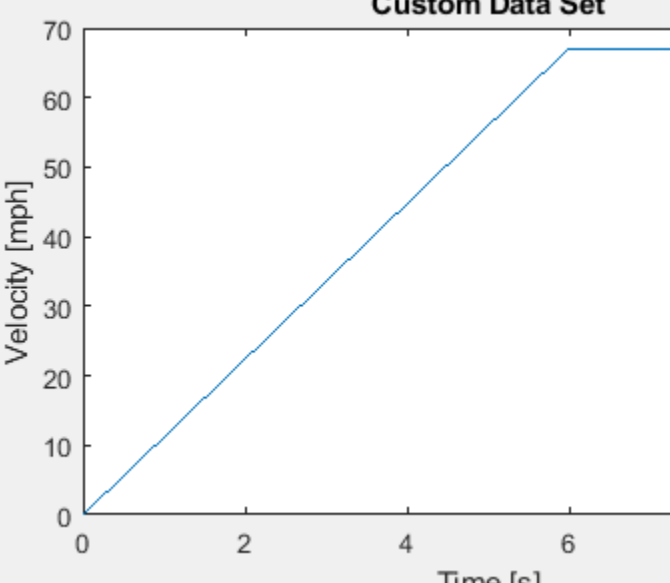
Drive Cycle Source	Enables Parameter
Wide Open Throttle (WOT)	**Start time, t_wot1**
	Initial reference speed, xdot_woto
	Nominal reference speed, xdot_wot1
	Time to start deceleration, wot2
	Final reference speed, xdot_wot2
	WOT simulation time, t_wotend
	Source velocity units
Workspace variable	**From workspace**
	Source velocity units
	Output gear shift data, if drive cycle includes gear shift schedule
.mat, .xls, .xlsx or .txt file	**Drive cycle source file**
	Source velocity units
	Output gear shift data, if drive cycle includes gear shift schedule
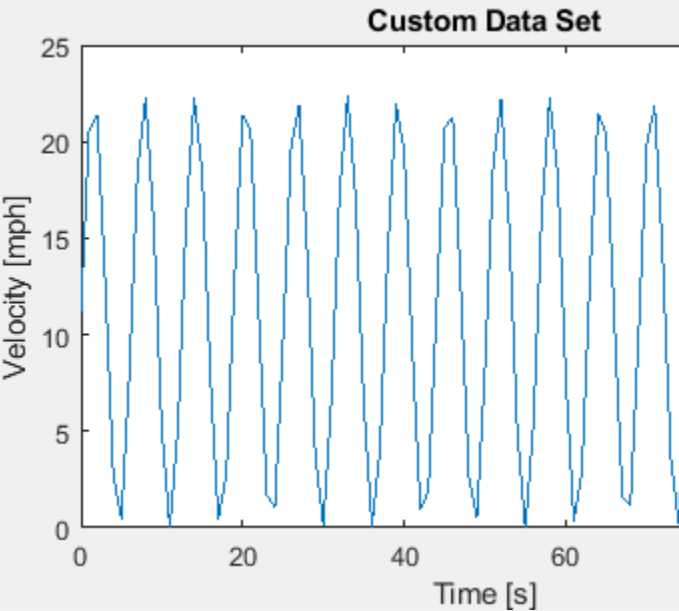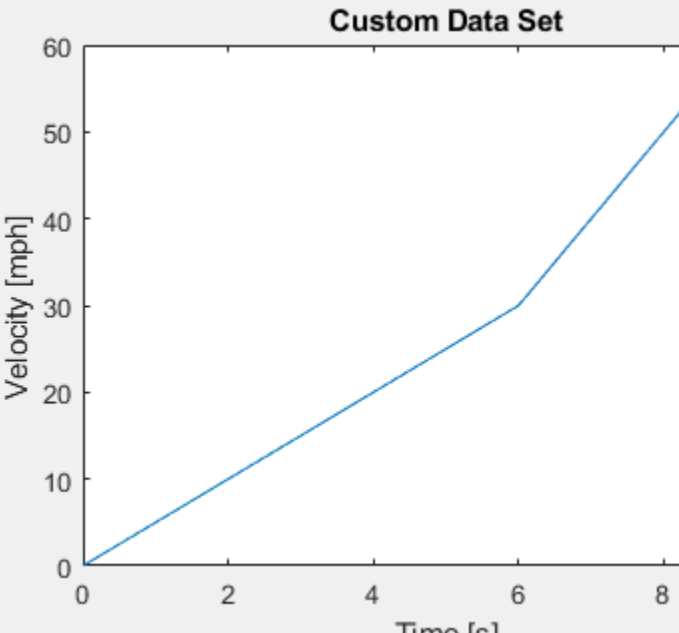
From workspace — Workspace
variable
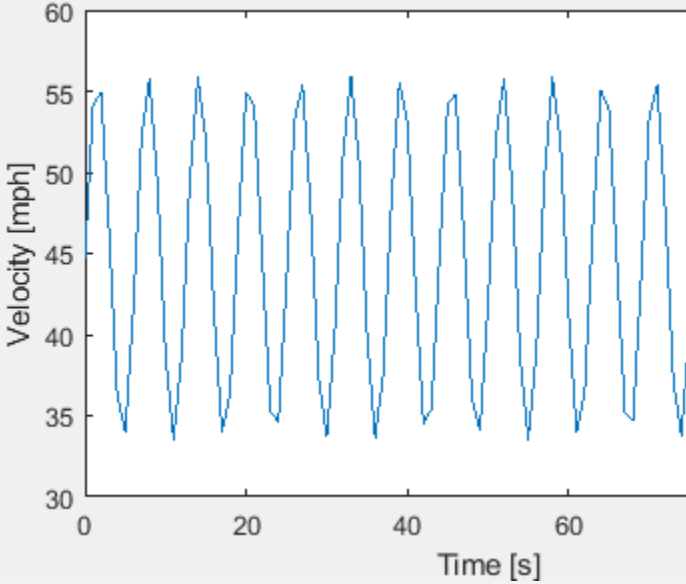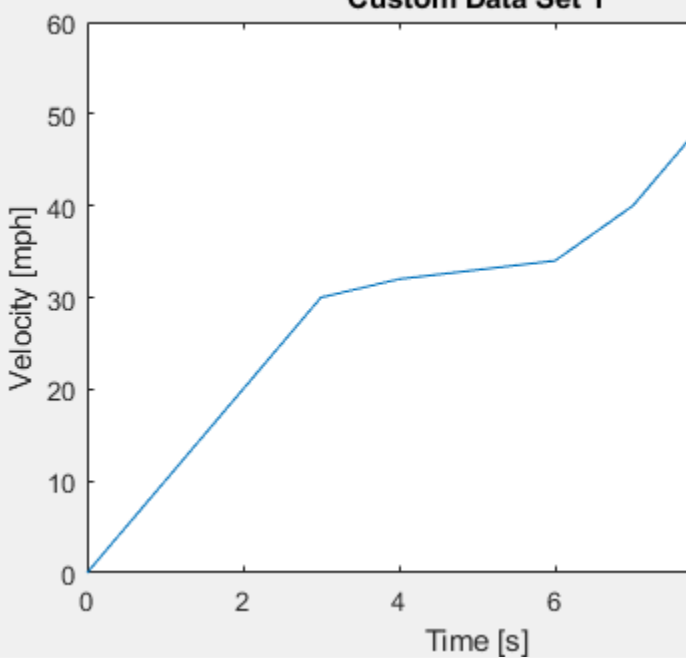
Monotonically increasing time, velocity, and, optionally, gear data, specified by a structure, 2-D array, or time series object. Enter units for velocity in the **Source velocity units** parameter field.

A valid point must exist for each corresponding time value. You cannot specify `inf`, `empty`, or `NaN`.

Workspace Variable	Source Velocity Unit	Output Velocity Unit	Drive Cycle Plot

Structure without a gear shift schedule. **From workspace** set to myCycleS.
 	
`t = 0:1:100;`
`xdot = 5.*sin(t)+10;`
`myCycleS.time = t';`
`myCycleS.signals.values = xdot';`
	m/s	mph	

Structure with a gear shift schedule. **From workspace** set to myCycleS.
 	
`gears=[0, 1, 2, 3, 3, 4, 4, 4, 4, 4, 4];`
`t=0:1:10;`
`xdot=[0,5,10,15,20,25,30,30,30,30,30];`
`myCycleS.time=t';`
`myCycleS.signals.values=[xdot',gears'];`
	m/s	mph	

Workspace Variable	Source Velocity Unit	Output Velocity Unit	Drive Cycle Plot

2-D array without a gear shift schedule. **From workspace** set to `myCycleA`.
 	
`t = 0:1:100;`
`xdot = 5.*sin(t)+5;`
`myCycleA = [t',xdot'];`
	m/s	mph	

2-D array with a gear shift schedule. **From workspace** set to `myCycleA`.
 	
`gears=[0, 1, 2, 3, 4, 4, 4, 5, 5, 5, 5];`
`t=0:1:10;`
`xdot=[0,5,10,15,20,25,30,40,50,60,60];`
`myCycleA=[t',xdot',gears'];`
	mph	mph	

Workspace Variable	Source Velocity Unit	Output Velocity Unit	Drive Cycle Plot

Time series object without a gear shift schedule. **From workspace** set to myCycleT.
 	
`myCycleT = timeseries;`
`t = 0:1:100;`
`xdot = 5.*sin(t)+20;`
`myCycleT.Data = xdot';`
`myCycleT.Time = t;`
	m/s	mph	

Time series object without a gear shift schedule. **From workspace** set to myCycleT.
 	
`myCycleT = timeseries;`
`gears=[0, 1, 2, 3, 4, 4, 4, 5, 5, 5, 5];`
`t=0:1:10;`
`xdot=[0,10,20,30,32,33,34,40,50,60,60];`
`myCycleT.Data = [xdot',gears'];`
`myCycleT.Time = t';`
	mph	mph	
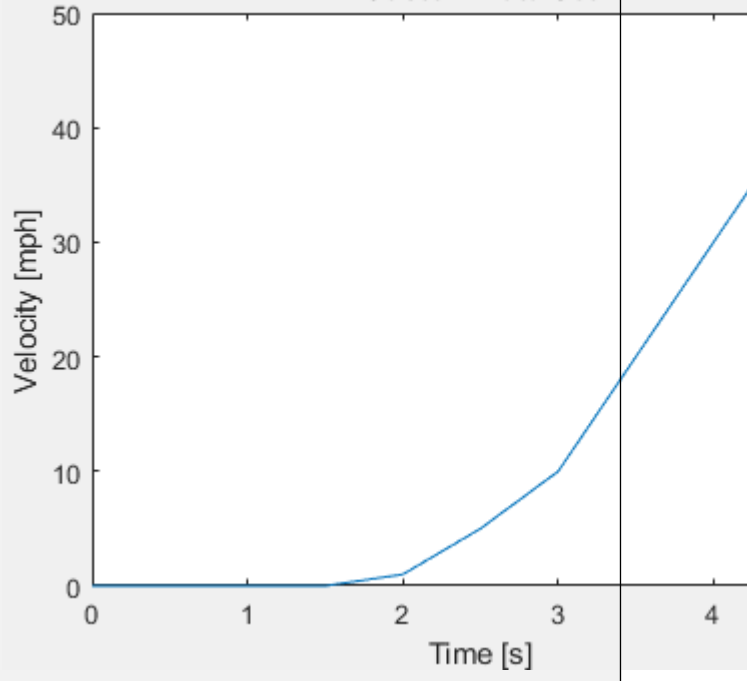
Dependencies
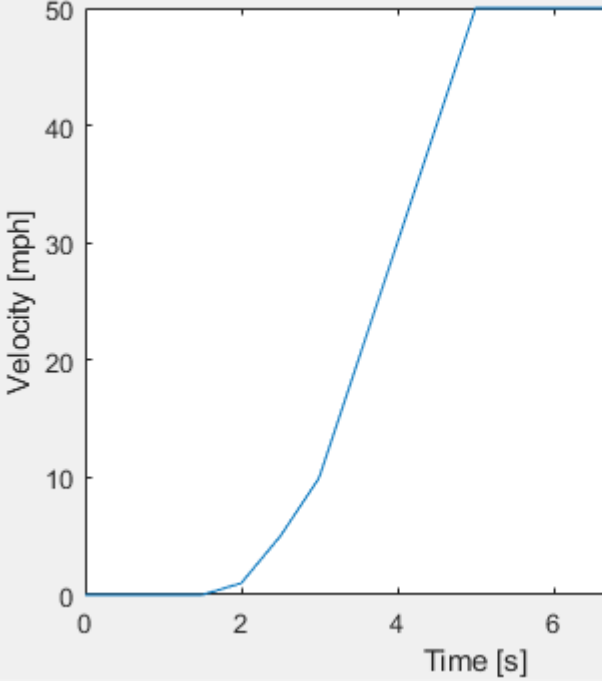
To enable this parameter, select `Workspace variable` from **Drive cycle source**.

Drive cycle source file — File name
`.mat, .xls, .xlsx or .txt`

File containing monotonically increasing time, velocity, and, optionally, gear in column or comma-separated format. The block ignores units in the file. Enter units for velocity in the **Source velocity units** parameter field.

File	Source Velocity Unit	Output Velocity Unit	Drive Cycle Plot
An .xls or .xlsx file with time in column A and velocity in column B. 	mph	mph	

File	Source Velocity Unit	Output Velocity Unit	Drive Cycle Plot
An .xls or .xlsx file with time in column A, velocity in column B, and gear in column C. The block: • Ignores the units in the file. • Converts the gear information to integers: • N to 0 • D to 2	mph	mph	

	A	B	C
1	sec	mph	gear
2	0	0	N
3	0.5	0	N
4	1	0	N
5	1.5	0	N
6	2	1	D
7	2.5	5	D
8	3	10	D
9	3.5	20	D
10	4	30	D
11	4.5	40	D
12	5	50	D

File	Source Velocity Unit	Output Velocity Unit	Drive Cycle Plot
A .txt with time in column 1 and velocity in column 2. The block ignores the header and units information. Time Speed sec mph 0 0 1 0 2 0 3 0 4 0 5 5 6 10 7 15 8 20 9 30 10 35 11 40 12 45 13 50 14 55 15 60 16 60 17 60 18 60 19 60 20 60	mph	mph	

If you provide the gear schedule using **P**, **R**, **N**, **D**, **L**, **OD**, the block maps the gears to integers.

Gear	Integer
P	80
R	-1
N	0
L	1
D	2
OD	Next integer after highest specified gear.

For example, the block converts the gear schedule P P N L D 3 4 5 6 5 4 5 6 7 OD 7 to 80 80 0 1 2 3 4 5 6 5 4 5 6 7 8 7.

Dependencies

To enable this parameter, select `.mat, .xls, .xlsx or .txt file` from **Drive cycle source**.

Repeat cyclically — Repeat drive cycle
off (default) | on

Repeat the drive cycle if the simulation run time exceeds the length of the drive cycle.

`Output acceleration` — Output the acceleration
`off` (default)

To calculate the acceleration, the block implements Savitzky-Golay differentiation using a second-order polynomial with a three-sample point filter.

Dependencies

To create the output acceleration port, select **Output acceleration**. Selecting **Output acceleration** enables the **Output acceleration units** parameter.

`Output gear shift data` — Output the gear
`off` (default) | `on`

Dependencies

- Specify a drive cycle that contains a gear shift schedule. You can use:

  - A support package to install standard drive cycles that include the gear shift schedules, for example `JC08` and `CUEDC`.
  - Workspace variables.
  - `.mat`, `.xls`, `.xlsx`, or `.txt` files.

- Clicking this parameter creates input port **Gear**.

WOT

`Start time, t_wot1` — Drive cycle start time
`5` (default) | `scalar`

Drive cycle start time, in s. For example, this plot shows a drive cycle with a start time of `10` s.
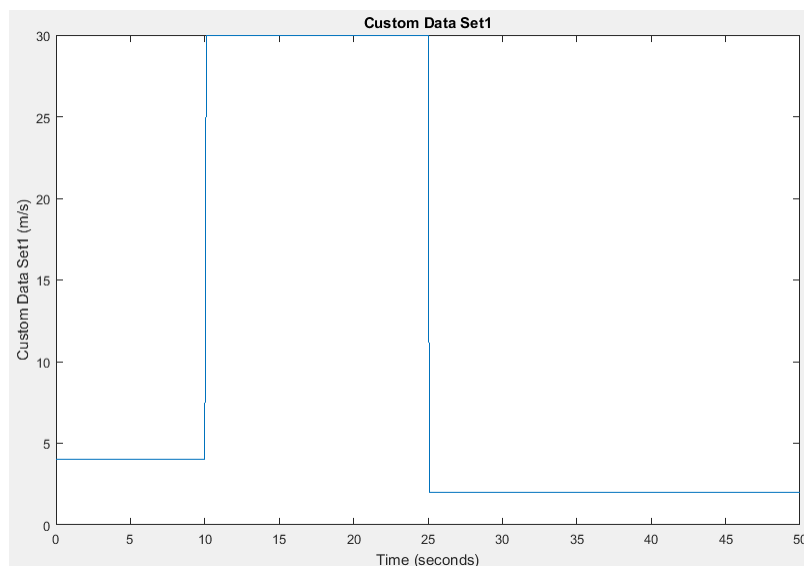
Custom Data Set1

Dependencies

To enable this parameter, select the **Drive cycle source** parameter `Wide Open Throttle (WOT)`.

Initial reference speed, xdot_woto — Speed
0 (default) | `scalar`

Initial reference speed, in units that you specify with the **Source velocity units** parameter. For example, this plot shows a drive cycle with an initial reference speed of 4 m/s.
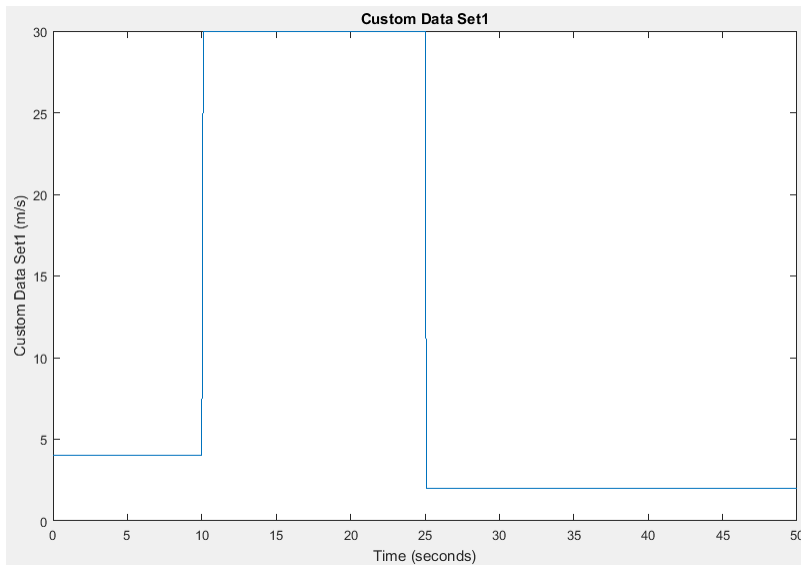
Dependencies

To enable this parameter, select the **Drive cycle source** parameter `Wide Open Throttle (WOT)`.

Nominal reference speed, xdot_wot1 — Speed
30 (default) | `scalar`

Nominal reference speed, in units that you specify with the **Source velocity units** parameter. For example, this plot shows a drive cycle with a nominal reference speed of 30 m/s.
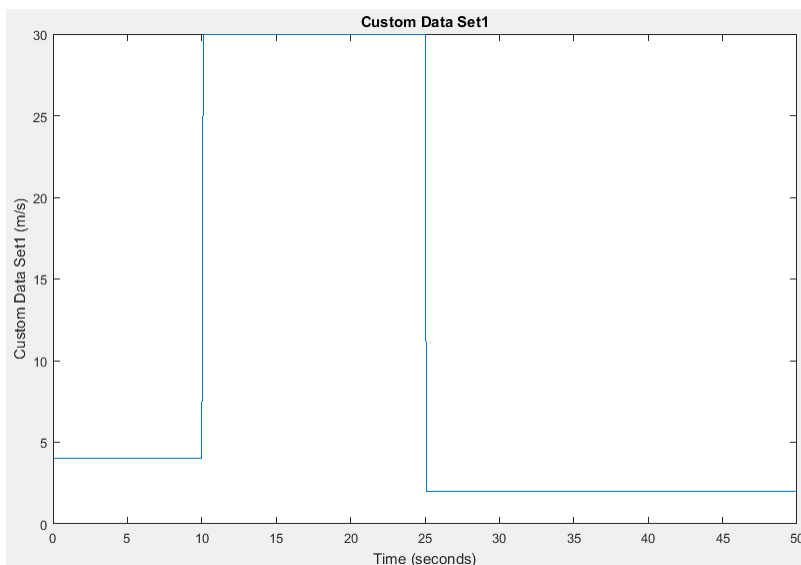
Dependencies

To enable this parameter, select the **Drive cycle source** parameter `Wide Open Throttle (WOT)`.

Time to start deceleration, wot2 — Time
20 (default) | `scalar`

Time to start vehicle deceleration, in s. For example, this plot shows a drive cycle with vehicle deceleration starting at 25 s.
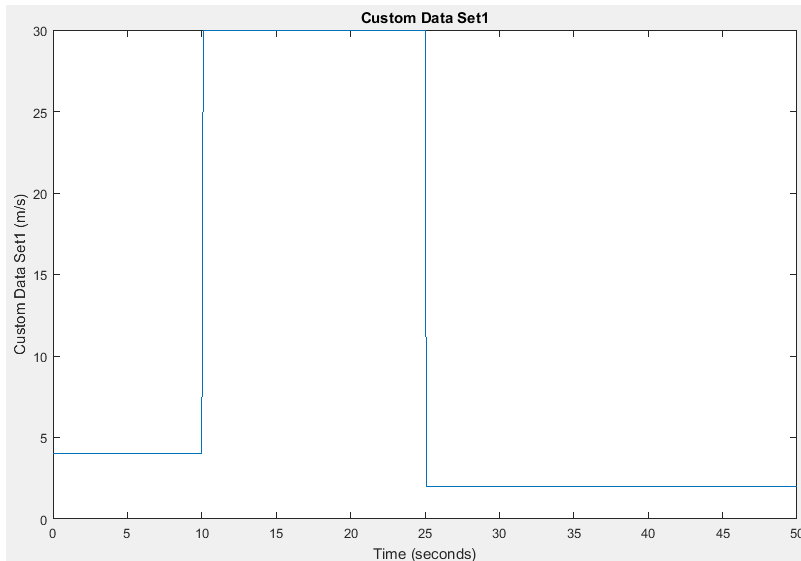
Dependencies

To enable this parameter, select the **Drive cycle source** parameter `Wide Open Throttle (WOT)`.

Final reference speed, xdot_wot2 — Speed
0 (default) | `scalar`

Final reference speed, in units that you specify with the **Source velocity units** parameter. For example, this plot shows a drive cycle with a final reference speed of 2 m/s.
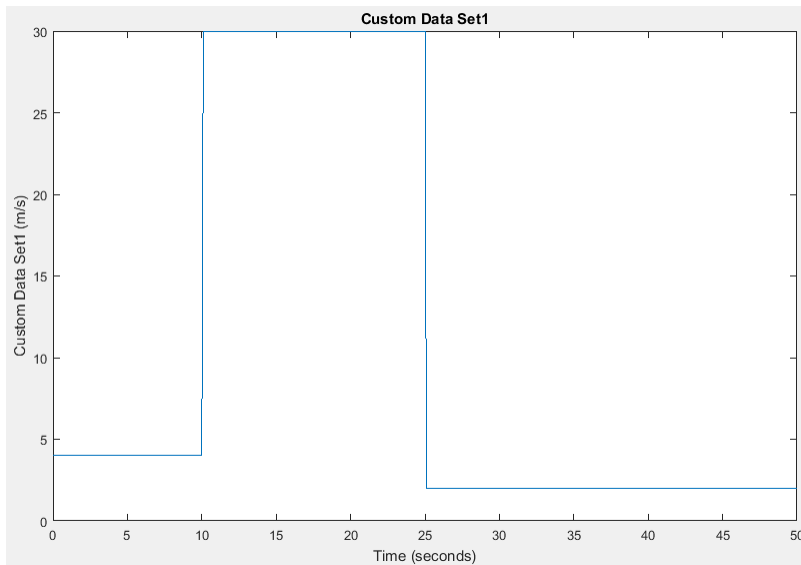
Dependencies

To enable this parameter, select the **Drive cycle source** parameter `Wide Open Throttle (WOT)`.

WOT simulation time, t_wotend — Time
30 (default) | `scalar`

Drive cycle WOT simulation time, in s. For example, this plot shows a drive cycle with a simulation time of 50 s.
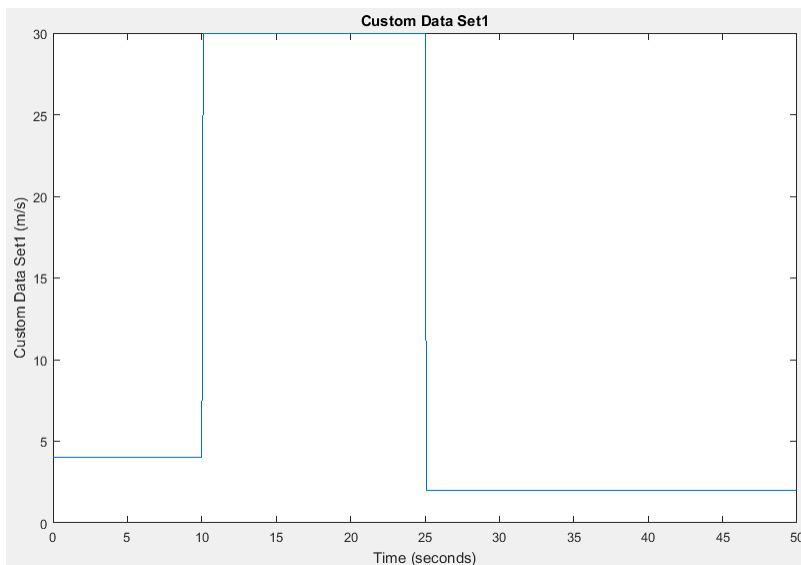
Dependencies

To enable this parameter, select the **Drive cycle source** parameter `Wide Open Throttle (WOT)`.

Units and Sample Period

Source velocity units — Specify velocity units
m/s (default)

Input velocity units.

Dependencies

To enable this parameter, select the **Drive cycle source** parameter Wide Open Throttle (WOT), Workspace variable, or .mat, .xls, .xlsx or .txt file.

Output velocity units — Specify velocity units
m/s (default)

Output velocity units.

Output acceleration units — Specify acceleration units
m/s^2 (default)

Specify the output acceleration units.

Dependencies

To enable this parameter, select **Output acceleration**.

Output sample period (0) for continuous — Sample rate
0 (default) | scalar

Sample rate. Set to 0 for continuous sample period. For a discrete period, specify a non-zero rate.

Fault Tracking

Fault Settings

Enable fault tracking — Enable fault tracking
off (default) | on

Select this parameter to enable drive cycle fault tracking. Use the parameters to specify the fault tolerances. If the vehicle speed is not within the allowable speed range, the block sets a fault condition.

Dependencies

Selecting this parameter enables these parameters:

- **Speed tolerance, velBnd**
- **Speed tolerance units, velBndUnit**
- **Velocity feedback units, inUnit**
- **Time tolerance, timeBnd**

Speed tolerance, velBnd — Drive cycle speed tolerance
2.0 (default) | scalar

The speed tolerance above the highest point and below the lowest point of the drive cycle speed trace within the time tolerance. If the vehicle speed is not within the allowable speed range, the block sets a fault condition. For the tolerances specified by the standardized tests, use these settings:

- EPA dynamometer driving schedules — `2.0`
- WLTP tests — `2.0`

These figures illustrate how the block uses the velocity and time tolerances to determine the allowable speed range.
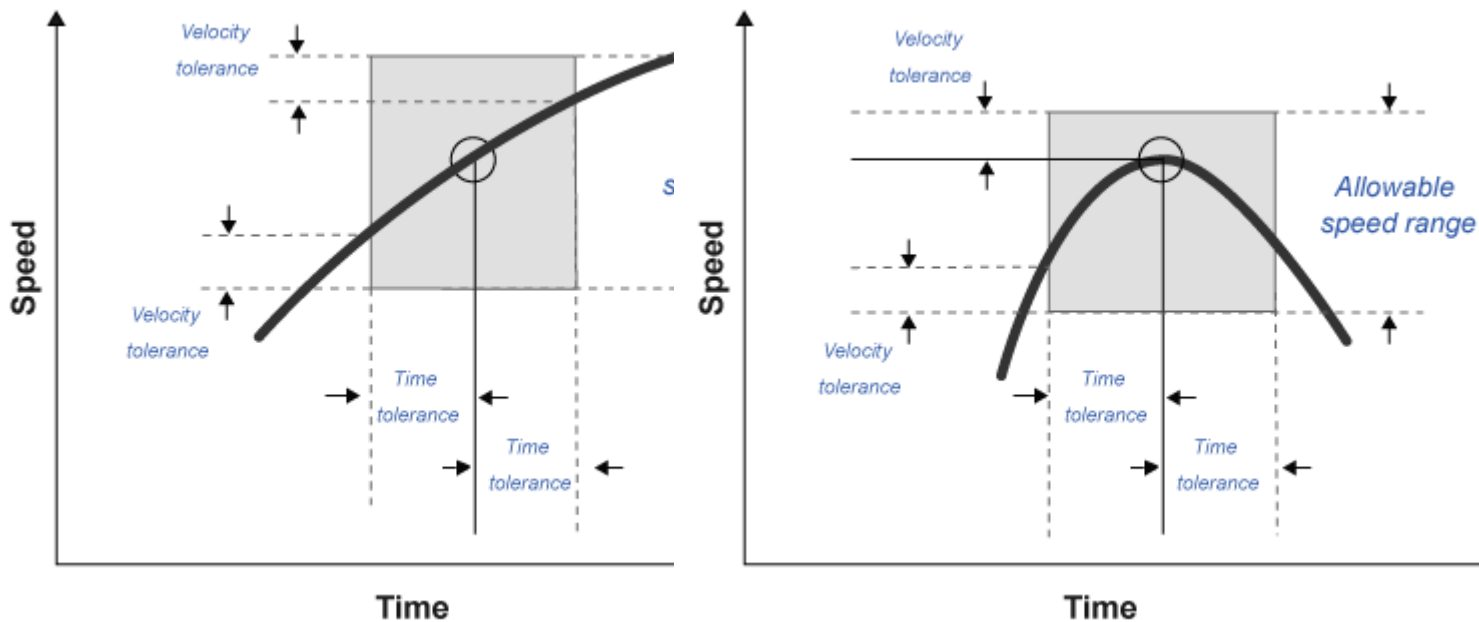
Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Enable fault tracking**.

`Speed tolerance units, velBndUnit` — Set units
mph (default)

Speed tolerance units. For the units specified by the standardized tests, use these units:

- EPA dynamometer driving schedules — `m/s`
- WLTP tests — `km/h`

Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Enable fault tracking**.

`Velocity feedback units, inUnit` — Set velocity feedback units
m/s (default)

Velocity feedback units. Set the value to the `VelFdbk` input port signal units.

Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Enable fault tracking**.

`Time tolerance, timeBnd` — Time tolerance
`1.0` (default) | `scalar`

Time that the block uses to determine the speed tolerance. If the vehicle speed is not within the allowable speed range, the block sets a fault condition. For the time tolerances specified by the standardized tests, use these settings:

- EPA dynamometer driving schedules — `1.0`
- WLTP tests — `1.0`

These figures illustrate how the block uses the velocity and time tolerances to determine the allowable speed range.
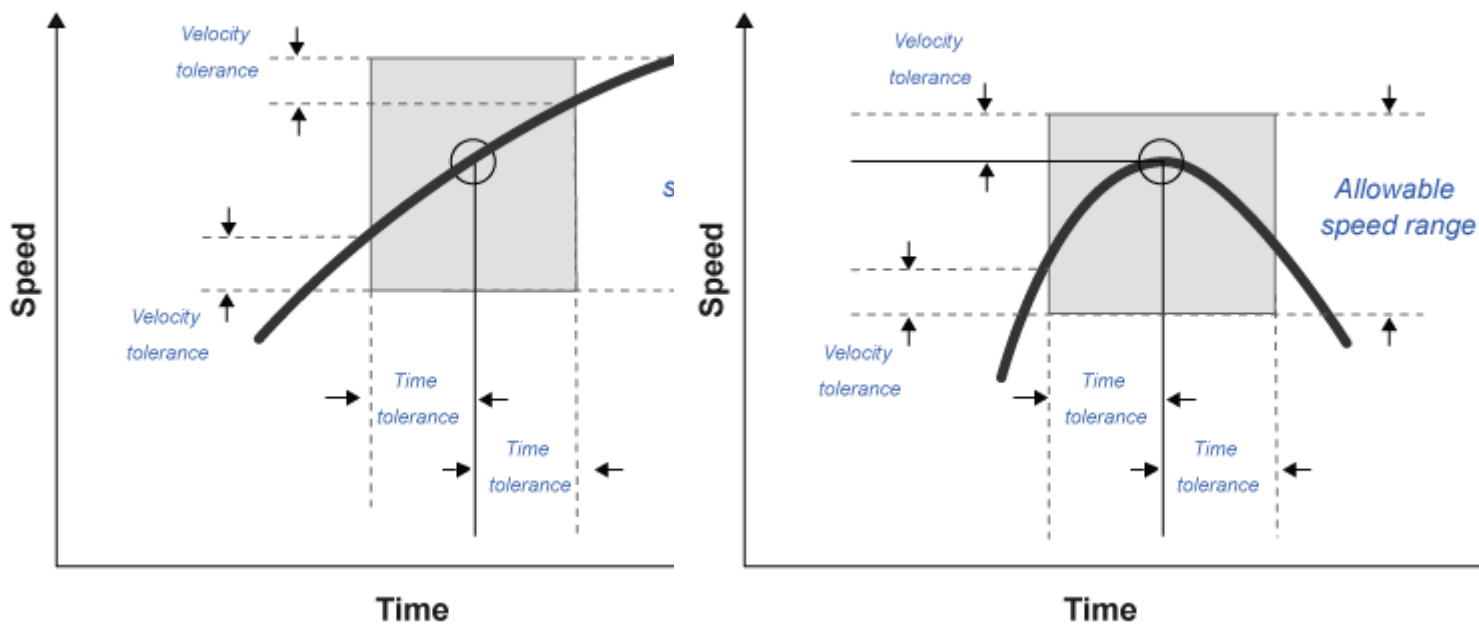
Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Enable fault tracking**.

Failure Settings

`Enable failure tracking` — Enable failure tracking
`off` (default) | `on`

Select this parameter to enable drive cycle failure tracking.

Dependencies

To enable this parameter, select **Enable fault tracking**. Selecting **Enable failure tracking** parameter enables these parameters:

- **Stop simulation when trace fails, stopSim**
- **Maximum number of faults, maxFaultCnt**

- **Maximum single fault time, maxFaultTime**
- **Maximum total fault time, maxTotFaultTime**

Maximum number of faults, maxFaultCnt — Maximum number of faults
10 (default) | scalar

Maximum number of faults during the drive cycle. For the number specified by the standardized tests, use these settings:

- EPA dynamometer driving schedules — *Not specified*
- WLTP tests — 10

If the number of faults exceeds the maximum number of faults, the block sets a fault failure.

Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Enable failure tracking**.

Maximum single fault time, maxFaultTime — Maximum duration of single fault
2.0 (default) | scalar

Maximum duration of single fault, in s. For the time specified by the standardized tests, use these settings:

- EPA dynamometer driving schedules — 2.0
- WLTP tests — 1.0

If the fault duration exceeds the maximum single fault time, the block sets a fault failure.

Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Enable failure tracking**.

Maximum total fault time, maxTotFaultTime — Maximum total fault time
15.0 (default) | scalar

Maximum accumulated time spent under fault condition, in s.

If the accumulated time spent under fault condition exceeds the maximum total fault time, the block sets a fault failure.

Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Enable failure tracking**.

Simulation Trace

Display simulation trace — Display velocity trace
off (default) | on

Select this parameter to display a velocity trace window. Selecting this parameter can slow the simulation time.

Dependencies

Selecting this parameter enables these parameters:

- **Simulation trace update rate, dtTrace**
- **Simulation trace display window, traceWindow**

`Simulation trace update rate, dtTrace` — Trace update rate
1 (default) | scalar

Simulation trace update rate, in s. Set to 0 for continuous sample period. For a discrete period, specify a non-zero rate.

Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Display simulation trace**.

`Simulation trace display window, traceWindow` — Trace window update rate
10 (default) | scalar

Simulation trace window update rate, in s.

Dependencies

To enable this parameter, on the **Fault Tracking** tab, select **Display simulation trace**.

References

[1] Environmental Protection Agency (EPA). *EPA urban dynamometer driving schedule*. 40 CFR 86.115-78, July 1, 2001.

[2] European Union Commission. "Speed trace tolerances". *European Union Commission Regulation*. 32017R1151, Sec 1.2.6.6, June 1, 2017.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Lateral Driver | Longitudinal Driver | Predictive Driver

Topics
"Support Package For Maneuver and Drive Cycle Data"
"Time Series Objects and Collections" (MATLAB)

Introduced in R2017a

Longitudinal Driver

Longitudinal speed-tracking controller
Library: Powertrain Blockset / Vehicle Scenario Builder
                        Vehicle Dynamics Blockset / Vehicle Scenarios / Driver
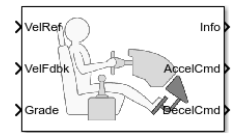
Description

The Longitudinal Driver block implements a longitudinal speed-tracking controller. Based on reference and feedback velocities, the block generates normalized acceleration and braking commands that can vary from 0 through 1. You can use the block to model the dynamic response of a driver or to generate the commands necessary to track a longitudinal drive cycle.

Configurations

External Actions

Use the **External Actions** parameters to create input ports for signals that can disable, hold, or override the closed-loop acceleration or deceleration commands. The block uses this priority order for the input commands: disable (highest), hold, override.

This table summarizes the external action parameters.

Goal	External Action Parameter	Input Ports	Data Type
Override the accelerator command with an input acceleration command.	**Accelerator override**	EnablAccelOvr	Boolean
		AccelOvrCmd	double
Hold the acceleration command at the current value.	**Accelerator hold**	AccelHld	Boolean
Disable the acceleration command.	**Accelerator disable**	AccelZero	Boolean
Override the decelerator command with an input deceleration command.	**Decelerator override**	EnablDecelOvr	Boolean
		DecelOvrCmd	double
Hold the decelerator command at current value.	**Decelerator hold**	DecelHld	Boolean
Disable the decelerator command.	**Decelerator disable**	DecelZero	Boolean

Controller

Use the **Control type, cntrlType** parameter to specify one of these control options.

Setting	Block Implementation
`PI`	Proportional-integral (PI) control with tracking windup and feed-forward gains.
`Scheduled PI`	PI control with tracking windup and feed-forward gains that are a function of vehicle velocity.
`Predictive`	Optimal single-point preview (look ahead) control model developed by C. C. MacAdam[1, 2, 3]. The model represents driver steering control behavior during path-following and obstacle avoidance maneuvers. Drivers preview (look ahead) to follow a predefined path. To implement the MacAdam model, the block: • Represents the dynamics as a linear single track (bicycle) vehicle • Minimizes the previewed error signal at a single point T^* seconds ahead in time • Accounts for the driver lag deriving from perceptual and neuromuscular mechanisms

Shift

Use the **Shift type, shftType** parameter to specify one of these shift options.

Setting	Block Implementation
`None`	No transmission. Block outputs a constant gear of 1. Use this setting to minimize the number of parameters you need to generate acceleration and braking commands to track forward vehicle motion. This setting does not allow reverse vehicle motion.
`Reverse, Neutral, Drive`	Block uses a Stateflow® chart to model reverse, neutral, and drive gear shift scheduling. Use this setting to generate acceleration and braking commands to track forward and reverse vehicle motion using simple reverse, neutral, and drive gear shift scheduling. Depending on the vehicle state and vehicle velocity feedback, the block uses the initial gear and time required to shift to shift the vehicle up into drive or down into reverse or neutral. For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.

Setting	Block Implementation
Scheduled	Block uses a Stateflow chart to model reverse, neutral, park, and N-speed gear shift scheduling. Use this setting to generate acceleration and braking commands to track forward and reverse vehicle motion using reverse, neutral, park, and N-speed gear shift scheduling. Depending on the vehicle state and vehicle velocity feedback, the block uses these parameters to determine the: • Initial gear • Upshift and downshift accelerator pedal positions • Upshift and downshift velocity • Timing for shifting and engaging forward and reverse from neutral For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.
External	Block uses the input gear, vehicle state, and velocity feedback to generate acceleration and braking commands to track forward and reverse vehicle motion. For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.

Gear Signal

Use the **Output gear signal** parameter to create the `GearCmd` output port. The `GearCmd` signal contains the integer value of the commanded vehicle gear.

Gear	Integer
Park	80
Reverse	-1
Neutral	0
Drive	1
Gear	Gear number

Controller: PI Speed-Tracking

If you set the control type to `PI` or `Scheduled PI`, the block implements proportional-integral (PI) control with tracking windup and feed-forward gains. For the `Scheduled PI` configuration, the block uses feed forward gains that are a function of vehicle velocity.

To calculate the speed control output, the block uses these equations.

Setting	Equation
PI	$$y = \frac{K_{ff}}{v_{nom}}v_{ref} + \frac{K_p e_{ref}}{v_{nom}} + \int \left(\frac{K_i e_{ref}}{v_{nom}} + K_{aw}e_{out} \right) dt + K_g \theta$$

Setting	Equation
`Scheduled PI`	$$y = \frac{K_{ff}(v)}{v_{nom}}v_{ref} + \frac{K_p(v)e_{ref}}{v_{nom}} + \int\left(\frac{K_i(v)e_{ref}}{v_{nom}} + K_{aw}e_{out}\right)e_{ref}dt + K_g(v)\theta$$

where:

$$e_{ref} = v_{ref} - v$$

$$e_{out} = y_{sat} - y$$

$$y_{sat} = \begin{cases} -1 & y < -1 \\ y & -1 \le y \le 1 \\ 1 & 1 < y \end{cases}$$

The velocity error low-pass filter uses this transfer function.

$$H(s) = \frac{1}{\tau_{err}s + 1} \quad \text{for} \quad \tau_{err} > 0$$

To calculate the acceleration and braking commands, the block uses these equations.

$$y_{acc} = \begin{cases} 0 & y_{sat} < 0 \\ y_{sat} & 0 \le y_{sat} \le 1 \\ 1 & 1 < y_{sat} \end{cases}$$

$$y_{dec} = \begin{cases} 0 & y_{sat} > 0 \\ -y_{sat} & -1 \le y_{sat} \le 0 \\ 1 & y_{sat} < -1 \end{cases}$$

The equations use these variables.

v_{nom}	Nominal vehicle speed
K_p	Proportional gain
K_i	Integral gain
K_{aw}	Anti-windup gain
K_{ff}	Velocity feed-forward gain
K_g	Grade feed-forward gain
θ	Grade angle
τ_{err}	Error filter time constant
y	Nominal control output magnitude
y_{sat}	Saturated control output magnitude
e_{ref}	Velocity error
e_{out}	Difference between saturated and nominal control outputs
y_{acc}	Acceleration signal
y_{dec}	Braking signal
v	Velocity feedback signal

v_{ref} Reference velocity signal

Controller: Predictive Speed-Tracking

If you set the **Control type, cntrlType** parameter to `Predictive`, the block implements an optimal single-point preview (look ahead) control model developed by C. C. MacAdam[1, 2, 3]. The model represents driver steering control behavior during path-following and obstacle avoidance maneuvers. Drivers preview (look ahead) to follow a predefined path. To implement the MacAdam model, the block:

- Represents the dynamics as a linear single track (bicycle) vehicle
- Minimizes the previewed error signal at a single point T* seconds ahead in time
- Accounts for the driver lag deriving from perceptual and neuromuscular mechanisms

Vehicle Dynamics

For longitudinal motion, the block implements these linear dynamics.

$$x_1 = v$$

$$\dot{x}_1 = x_2 = \frac{K_{pt}}{m} - g\sin(\gamma) + F_r x_1$$

In matrix notation:

$$\dot{x} = Fx + g\bar{u}$$

where:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$F = \begin{bmatrix} 0 & 1 \\ \frac{F_r}{m} & 0 \end{bmatrix}$$

$$g = \begin{bmatrix} 0 \\ \frac{K_{pt}}{m} \end{bmatrix}$$

$$\bar{u} = u - \frac{m^2}{K_{pt}} g\sin(\gamma)$$

The block uses this equation for the rolling resistance.

$$F_r = -\left[\tanh(x_1)(\frac{a_r}{x_1} + c_r x_1) + b_r \right]$$

The single-point model assumes a minimum previewed error signal at a single point $T*$ seconds ahead in time. $a*$ is the driver ability to predict the future vehicle response based on the current steering control input. $b*$ is the driver ability to predict the future vehicle response based on the current vehicle state. The block uses these equations.

$$a* = (T*)m^T \left[I + \sum_{n=1}^{\infty} \frac{F^n(T*)^n}{(n+1)!} \right] ge$$

$$b* = m^T \left[I + \sum_{n=1}^{\infty} \frac{F^n(T*)^n}{n!} \right]$$

where:

$$m^T = [1\ 1]$$

The equations use these variables.

a, b	Forward and rearward tire location, respectively
m	Vehicle mass
I	Vehicle rotational inertia
$a*$, $\boldsymbol{b}*$	Driver prediction scalar and vector gain, respectively
$\boldsymbol{x}$	Predicted vehicle state vector
v	Longitudinal velocity
$\boldsymbol{F}$	System matrix
K_{pt}	Tractive force and brake limit
γ	Grade angle
$\boldsymbol{g}$	Control coefficient vector
g	Gravitational constant
$T*$	Preview time window
$f(t+T*)$	Previewed path input T* seconds ahead
U	Forward vehicle velocity
$\boldsymbol{m^T}$	Constant observer vector; provides vehicle lateral position
F_r	Rolling resistance
a_r	Static rolling and driveline resistance
b_r	Linear rolling and driveline resistance
c_r	Aerodynamic rolling and driveline resistance

Optimization

The single-point model implemented by the block finds the steering command that minimizes a local performance index, J, over the current preview interval, $(t, t+T)$.

$$J = \frac{1}{T} \int_{t}^{t+T} [f(\eta) - y(\eta)]^2 d\eta$$

To minimize J with respect to the steering command, this condition must be met.

$$\frac{dJ}{du} = 0$$

You can express the optimal control solution in terms of a current non-optimal and corresponding nonzero preview output error $T*$ seconds ahead[1, 2, 3].

$$u^o(t) = u(t) + \frac{e(t + T^*)}{a^*}$$

The block uses the preview distance and vehicle longitudinal velocity to determine the preview time window.

$$T^* = \frac{L}{U}$$

The equations use these variables.

T^*	Preview time window
$f(t+T^*)$	Previewed path input T^* sec ahead
$y(t+T^*)$	Previewed plant output T^* sec ahead
$e(t+T^*)$	Previewed error signal T^* sec ahead
$u(t), u^o(t)$	Steer angle and optimal steer angle, respectively
L	Preview distance
J	Performance index
U	Forward (longitudinal) vehicle velocity

Driver Lag

The single-point model implemented by the block introduces a driver lag. The driver lag accounts for the delay when the driver is tracking tasks. Specifically, it is the transport delay deriving from perceptual and neuromuscular mechanisms. To calculate the driver transport delay, the block implements this equation.

$$H(s) = e^{-s\tau}$$

The equations use these variables.

τ	Driver transport delay
$y(t+T^*)$	Previewed plant output T^* sec ahead
$e(t+T^*)$	Previewed error signal T^* sec ahead
$u(t), u^o(t)$	Steer angle and optimal steer angle, respectively
J	Performance index

Ports

Input

VelRef — Reference vehicle velocity
scalar

Reference velocity, v_{ref}, in m/s.

EnblAccelOvr — Enable acceleration command override
scalar

Enable acceleration command override.

Dependencies

To enable this port, select **Acceleration override**.

Data Types: `Boolean`

AccelOvrCmd — Acceleration override command
`scalar`

Acceleration override command, normalized from 0 through 1.

Dependencies

To enable this port, select **Acceleration override**.

Data Types: `double`

AccelHld — Acceleration hold
`scalar`

Boolean signal that holds the acceleration command at the current value.

Dependencies

To enable this port, select **Acceleration hold**.

Data Types: `Boolean`

AccelZero — Disable acceleration command
`scalar`

Disable acceleration command.

Dependencies

To enable this port, select **Acceleration disable**.

Data Types: `Boolean`

EnblDecelOvr — Enable deceleration command override
`scalar`

Enable deceleration command override.

Dependencies

To enable this port, select **Deceleration override**.

Data Types: `Boolean`

DecelOvrCmd — Deceleration override command
`scalar`

Deceleration override command, normalized from 0 through 1.

Dependencies

To enable this port, select **Deceleration override**.

Data Types: `double`

DecelHld — Deceleration hold
scalar

Boolean signal that holds the deceleration command at the current value.

Dependencies

To enable this port, select **Deceleration hold**.

Data Types: Boolean

DecelZero — Disable deceleration command
scalar

Disable deceleration command.

Dependencies

To enable this port, select **Deceleration disable**.

Data Types: Boolean

ExtGear — Gear
scalar

Gear	Integer
Park	80
Reverse	-1
Neutral	0
Drive	1
Gear	Gear number

Dependencies

To enable this port, set **Shift type, shftType** to External.

VelFdbk — Longitudinal vehicle velocity
scalar

Longitudinal vehicle velocity, U, in the vehicle-fixed frame, in m/s.

Grade — Road grade angle
scalar

Road grade angle, θ or γ, in deg.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal		Variable	Description
Accel		y_{acc}	Commanded vehicle acceleration, normalized from 0 through 1
Decel		y_{dec}	Commanded vehicle deceleration, normalized from 0 through 1
Gear			Integer value of commanded gear
Clutch			Clutch command
Err		e_{ref}	Difference in reference vehicle speed and vehicle speed
ErrSqrSum		$\int_0^t e_{ref}^2 dt$	Integrated square of error
ErrMax		$\max(e_{ref}(t))$	Maximum error during simulation
ErrMin		$\min(e_{ref}(t))$	Minimum error during simulation
ExtActions	EnblAccelOvr		Override the accelerator command with an input acceleration command
	AccelOvrCmd		Input accelerator override command
	AccelHld		Hold the acceleration command at the current value
	AccelZero		Disable the acceleration command
	EnblDecelOvr		Override the decelerator command with an input deceleration command
	DecelOvrCmd		Input deceleration override command
	DecelHld		Hold the decelerator command at current value
	DecelZero		Disable the decelerator command

AccelCmd — Commanded vehicle acceleration
scalar

Commanded vehicle acceleration, y_{acc}, normalized from 0 through 1.

DecelCmd — Commanded vehicle deceleration
scalar

Commanded vehicle deceleration, y_{dec}, normalized from 0 through 1.

GearCmd — Commanded vehicle gear
scalar

Integer value of commanded vehicle gear.

Gear	Integer
Park	80
Reverse	-1

Gear	Integer
Neutral	`0`
Drive	`1`
Gear	`Gear number`

Dependencies

To enable this port, select **Output gear signal**.

Parameters

External Actions

`Accelerator override` — Override acceleration command
off (default) | on

Select to override the acceleration command with an input acceleration command.

Dependencies

Selecting this parameter creates the `EnblAccelOvr` and `AccelOvrCmd` input ports.

`Accelerator hold` — Hold acceleration command
off (default) | on

Select to hold the acceleration command.

Dependencies

Selecting this parameter creates the `AccelHld` input port.

`Accelerator disable` — Disable acceleration command
off (default) | on

Select to disable the acceleration command.

Dependencies

Selecting this parameter creates the `AccelZero` input port.

`Decelerator override` — Override deceleration command
off (default) | on

Select to override the deceleration command with an input deceleration command.

Dependencies

Selecting this parameter creates the `EnblDecelOvr` and `DecelOvrCmd` input ports.

`Decelerator hold` — Hold deceleration command
off (default) | on

Select to hold the deceleration command.

Dependencies

Selecting this parameter creates the `DecelHld` input port.

Decelerator disable — Disable deceleration command
off (default) | on

Select to disable the deceleration command.

Dependencies

Selecting this parameter creates the `DecelZero` input port.

Configuration

Control type, cntrlType — Longitudinal control
PI (default) | Scheduled PI | Predictive

Type of longitudinal control.

Setting	Block Implementation
PI	Proportional-integral (PI) control with tracking windup and feed-forward gains.
Scheduled PI	PI control with tracking windup and feed-forward gains that are a function of vehicle velocity.
Predictive	Optimal single-point preview (look ahead) control model developed by C. C. MacAdam[1, 2, 3]. The model represents driver steering control behavior during path-following and obstacle avoidance maneuvers. Drivers preview (look ahead) to follow a predefined path. To implement the MacAdam model, the block: • Represents the dynamics as a linear single track (bicycle) vehicle • Minimizes the previewed error signal at a single point $T*$ seconds ahead in time • Accounts for the driver lag deriving from perceptual and neuromuscular mechanisms

Shift type, shftType — Shift type
None (default) | Reverse, Neutral, Drive | Scheduled | External

Shift type.

Setting	Block Implementation
None	No transmission. Block outputs a constant gear of 1. Use this setting to minimize the number of parameters you need to generate acceleration and braking commands to track forward vehicle motion. This setting does not allow reverse vehicle motion.

Setting	Block Implementation
`Reverse, Neutral, Drive`	Block uses a Stateflow chart to model reverse, neutral, and drive gear shift scheduling. Use this setting to generate acceleration and braking commands to track forward and reverse vehicle motion using simple reverse, neutral, and drive gear shift scheduling. Depending on the vehicle state and vehicle velocity feedback, the block uses the initial gear and time required to shift to shift the vehicle up into drive or down into reverse or neutral. For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.
`Scheduled`	Block uses a Stateflow chart to model reverse, neutral, park, and N-speed gear shift scheduling. Use this setting to generate acceleration and braking commands to track forward and reverse vehicle motion using reverse, neutral, park, and N-speed gear shift scheduling. Depending on the vehicle state and vehicle velocity feedback, the block uses these parameters to determine the: • Initial gear • Upshift and downshift accelerator pedal positions • Upshift and downshift velocity • Timing for shifting and engaging forward and reverse from neutral For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.
`External`	Block uses the input gear, vehicle state, and velocity feedback to generate acceleration and braking commands to track forward and reverse vehicle motion. For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.

Reference and feedback units, velUnits — Velocity units
m/s (default)

Vehicle velocity reference and feedback units.

Dependencies

If you set **Control type, cntrlType** control type to `Scheduled` or `Scheduled PI`, the block uses the **Reference and feedback units, velUnits** for the **Nominal speed, vnom** parameter dimension.

If you set **Shift Type, shftType** to `Scheduled`, the block uses the **Longitudinal velocity units, velUnits** for these parameter dimensions:

• **Upshift velocity data table, upShftTbl**

- **Downshift velocity data table, dwnShftTbl**

Output gear signal — Create GearCmd output port
off (default) | on

Specify to create output port GearCmd.

Control

Longitudinal

Proportional gain, Kp — Gain
10 (default) | scalar

Proportional gain, K_p, dimensionless.

Dependencies

To create this parameter, set **Control type** to PI.

Integral gain, Ki — Gain
5 (default) | scalar

Proportional gain, K_i, dimensionless.

Dependencies

To create this parameter, set **Control type** to PI.

Velocity feed-forward, Kff — Gain
.1 (default) | scalar

Velocity feed-forward gain, K_{ff}, dimensionless.

Dependencies

To create this parameter, set **Control type** to PI.

Grade feed-forward, Kg — Gain
0 (default) | scalar

Grade feed-forward gain, K_g, in 1/deg.

Dependencies

To create this parameter, set **Control type** to PI.

Velocity gain breakpoints, VehVelVec — Breakpoints
[0 100] (default) | vector

Velocity gain breakpoints, *VehVelVec*, dimensionless.

Dependencies

To create this parameter, set **Control type** to Scheduled PI.

Velocity feed-forward gain values, KffVec — Gain
[.1 .1] (default) | vector

Velocity feed-forward gain values, *KffVec*, as a function of vehicle velocity, dimensionless.

Dependencies

To create this parameter, set **Control type** to `Scheduled PI`.

Proportional gain values, KpVec — Gain
[10 10] (default) | `vector`

Proportional gain values, *KpVec*, as a function of vehicle velocity, dimensionless.

Dependencies

To create this parameter, set **Control type** to `Scheduled PI`.

Integral gain values, KiVec — Gain
[5 5] (default) | `vector`

Integral gain values, *KiVec*, as a function of vehicle velocity, dimensionless.

Dependencies

To create this parameter, set **Control type** to `Scheduled PI`.

Grade feed-forward values, KgVec — Grade gain
[0 0] (default) | `vector`

Grade feed-forward values, *KgVec*, as a function of vehicle velocity, in 1/deg.

Dependencies

To create this parameter, set **Control type** to `Scheduled PI`.

Nominal speed, vnom — Nominal vehicle speed
5 (default) | `scalar`

Nominal vehicle speed, v_{nom}, in units specified by the **Reference and feedback units, velUnits** parameter. The block uses the nominal speed to normalize the controller gains.

Dependencies

To create this parameter, set **Control type** to `PI` or `Scheduled PI`.

Anti-windup, Kaw — Gain
1 (default) | `scalar`

Anti-windup gain, K_{aw}, dimensionless.

Dependencies

To create this parameter, set **Control type** to `PI` or `Scheduled PI`.

Error filter time constant, tauerr — Filter
.01 (default) | `scalar`

Error filter time constant, τ_{err}, in s. To disable the filter, enter 0.

Dependencies

To create this parameter, set **Control type** to PI or `Scheduled PI`.

Predictive

Vehicle mass, m — Mass
1500 (default) | scalar

Vehicle mass, m, in kg.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to `Predictive`.

Effective vehicle total tractive force, Kp — Tractive force
3000 (default) | scalar

Effective vehicle total tractive force, K_p, in N.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to `Predictive`.

Driver response time, tau — Tau
.1 (default) | scalar

Driver response time, τ, in s.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to `Predictive`.

Preview distance, L — Distance
2 (default) | scalar

Driver preview distance, L, in m.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to `Predictive`.

Rolling resistance coefficient, aR — Resistance
200 (default) | scalar

Static rolling and driveline resistance coefficient, a_R, in N. Block uses the parameter to estimate the constant acceleration or braking effort.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to `Predictive`.

Rolling and driveline resistance coefficient, bR — Resistance
2.5 (default) | scalar

Rolling and driveline resistance coefficient, b_R, in N·s/m. Block uses the parameter to estimate the linear velocity-dependent acceleration or braking effort.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to `Predictive`.

Aerodynamic drag coefficient, cR — Drag
.5 (default) | scalar

Aerodynamic drag coefficient, c_R, in N·s^2/m^2. Block uses the parameter to estimate the quadratic velocity-dependent acceleration or braking effort.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to `Predictive`.

Gravitational constant, g — Gravitational constant
9.81 (default) | scalar

Gravitational constant, g, in m/s^2.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to `Predictive`.

Shift

Reverse, Neutral, Drive

Initial gear, GearInit — Initial gear
0 (default) | scalar

Integer value of the initial gear. The block uses the initial gear to generate acceleration and braking commands to track forward and reverse vehicle motion.

Gear	Integer
Park	80
Reverse	-1
Neutral	0
Drive	1
Gear	Gear number

Dependencies

To create this parameter, set **Shift type, shftType** to `Reverse, Neutral, Drive` or `Scheduled`. If you specify `Reverse, Neutral, Drive`, the **Initial Gear, GearInit** parameter value can be only `-1`, `0`, or `1`.

Time required to shift, tShift — Time
.1 (default) | scalar

Time required to shift, *tShift*, in s. The block uses the time required to shift to generate acceleration and braking commands to track forward and reverse vehicle motion using reverse, neutral, and drive gear shift scheduling.

Dependencies

To create this parameter, set **Shift type, shftType** to `Reverse, Neutral, Drive`.

Scheduled

Initial gear, GearInit — Initial gear
0 (default) | scalar

Integer value of the initial gear. The block uses the initial gear to generate acceleration and braking commands to track forward and reverse vehicle motion.

Gear	Integer
Park	80
Reverse	-1
Neutral	0
Drive	1
Gear	Gear number

Dependencies

To create this parameter, set **Shift type, shftType** to `Reverse, Neutral, Drive` or `Scheduled`. If you specify `Reverse, Neutral, Drive`, the **Initial Gear, GearInit** parameter value can be only -1, 0, or 1.

Up and down shift accelerator pedal positions, pdlVec — Pedal position breakpoints
[0.1 0.4 0.5 0.9] (default) | [1-by-m] vector

Pedal position breakpoints for lookup tables when calculating upshift and downshift velocities, dimensionless. Vector dimensions are 1 by the number of pedal position breakpoints, m.

Dependencies

To create this parameter, set **Shift type, shftType** to `Scheduled`.

Upshift velocity data table, upShftTbl — Table
[m-by-n] array

Upshift velocity data as a function of pedal position and gear, in units specified by the **Reference and feedback units, velUnits** parameter. Upshift velocities indicate the vehicle velocity at which the gear should increase by 1.

The array dimensions are m pedal positions by n gears. The first column of data, when n equals 1, is the upshift velocity for the neutral gear.

Dependencies

To create this parameter, set **Shift type, shftType** to `Scheduled`.

Downshift velocity data table, dwnShftTbl — Table
[m-by-n] array

Downshift velocity data as a function of pedal position and gear, in units specified by the **Reference and feedback units, velUnits** parameter. Downshift velocities indicate the vehicle velocity at which the gear should decrease by 1.

The array dimensions are m pedal positions by n gears. The first column of data, when n equals 1, is the downshift velocity for the neutral gear.

Dependencies

To create this parameter, set **Shift type, shftType** to Scheduled.

Time required to shift, tClutch — Time
.5 (default) | scalar

Time required to shift, t_{Clutch}, in s.

Dependencies

To create this parameter, set **Shift type, shftType** to Scheduled.

Time required to engage reverse from neutral, tRev — Time
.5 (default) | scalar

Time required to engage reverse from neutral, t_{Rev}, in s.

Dependencies

To create this parameter, set **Shift type, shftType** to Scheduled.

Time required to engage park from neutral, tPark — Time
120 (default) | scalar

Time required to engage park from neutral, t_{Park}, in s.

Dependencies

To create this parameter, set **Shift type, shftType** to Scheduled.

References

[1] MacAdam, C. C. "An Optimal Preview Control for Linear Systems". *Journal of Dynamic Systems, Measurement, and Control*. Vol. 102, Number 3, Sept. 1980.

[2] MacAdam, C. C. "Application of an Optimal Preview Control for Simulation of Closed-Loop Automobile Driving ". *IEEE Transactions on Systems, Man, and Cybernetics*. Vol. 11, Issue 6, June 1981.

[3] MacAdam, C. C. *Development of Driver/Vehicle Steering Interaction Models for Dynamic Analysis*. Final Technical Report UMTRI-88-53. Ann Arbor, Michigan: The University of Michigan Transportation Research Institute, Dec. 1988.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Drive Cycle Source | Lateral Driver | Predictive Driver

Introduced in R2017a

Lateral Driver

Lateral path-tracking controller
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Driver
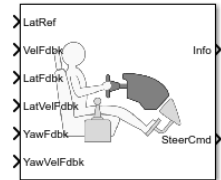
Description

The Lateral Driver block implements an optimal single point preview (look ahead) control model to generate normalized steering commands that track a lateral reference displacement. The normalized steering commands can vary between -1 to 1. To model the dynamics, the block uses a linear single track (bicycle) model. Use the Lateral Driver block to:

- Close the loop between a predefined path and actual vehicle motion.
- Generate steering commands that track predefined paths. You can connect the Predictive Driver block output to steering block inputs.

Also, you can specify a tire wheel angle saturation limit using the **Tire wheel angle limit, theta** parameter.

External Actions

Use the **External Actions** parameters to create input ports for signals that can disable, hold, or override the closed-loop steering command. The block uses this priority order for the input commands: disable (highest), hold, override. The block uses this priority order for the input commands: disable (highest), hold, override.

This table summarizes the external action parameters.

Goal	External Action Parameter	Input Ports	Data Type
Override the steering command with an input steering command.	**Steering override**	EnblSteerOvr	Boolean
		SteerOvrCmd	double
Hold the steering command at the current value.	**Steering hold**	SteerHld	Boolean
Disable the steering command.	**Steering disable**	SteerZero	Boolean

Use the **Output handwheel angle** parameter to specify the units for the steering ports.

Setting	Block Implementation	Port
off (default)	Commanded steer angle, normalized from -1 through 1. The block uses the tire wheel angle saturation limit **Tire wheel angle limit, theta** parameter to normalize the command.	SteerCmd — Output
	Overrides the steering command with an input steering command normalized from -1 through 1.	SteerOvrCmd — Input
on	Commanded steer angle in rad.	SteerCmd — Output
	Overrides the steering command with an input steering command in rad.	SteerOvrCmd — Input

Lateral Path-Tracking Controller

The Lateral Driver block implements an optimal single-point preview (look ahead) control model developed by C. C. MacAdam[1, 2, 3]. The model represents driver steering control behavior during path-following and obstacle avoidance maneuvers. Drivers preview (look ahead) to follow a predefined path. To implement the MacAdam model, the block:

- Represents the dynamics as a linear single track (bicycle) vehicle
- Minimizes the previewed error signal at a single point T* seconds ahead in time
- Accounts for the driver lag deriving from perceptual and neuromuscular mechanisms

This figure illustrates the block implementation of the single-point version of the driver model.
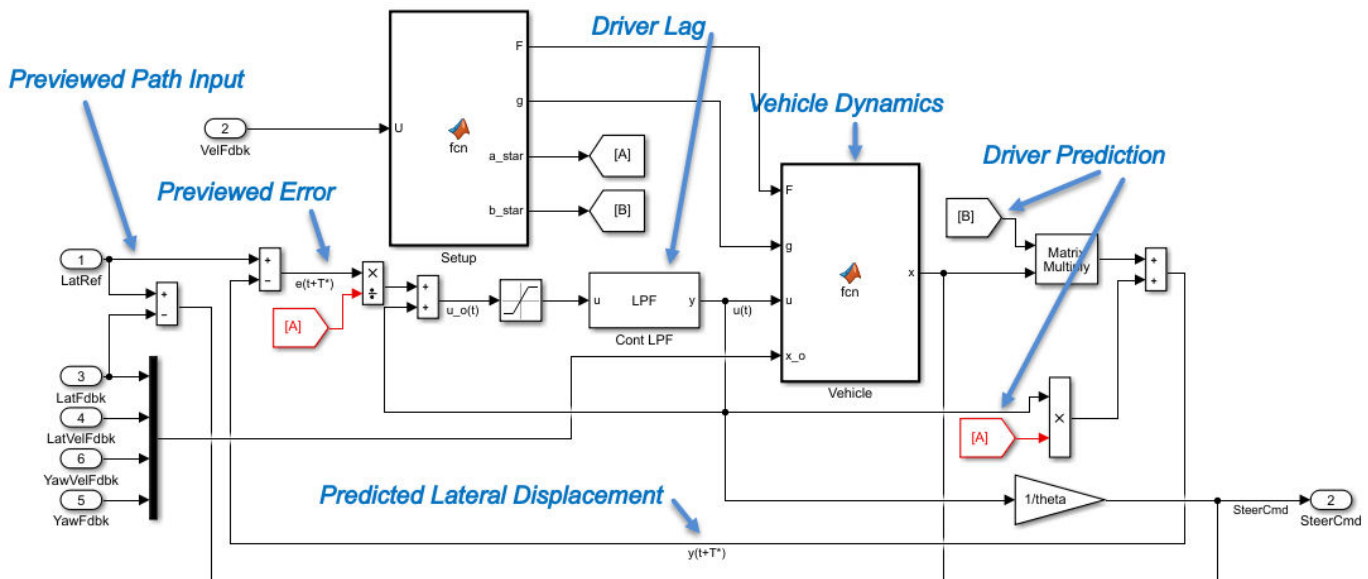
Vehicle Dynamics

For lateral and yaw motion, the block implements these linear dynamic equations.

$$\dot{y} = v + U\psi$$

$$\dot{v} = \left[-\frac{2(C_{\alpha F} + C_{\alpha R})}{mU}\right]v + \left[\frac{2(bC_{\alpha R} - aC_{\alpha F})}{mU} - U\right]r + \left(\frac{2C_{\alpha F}}{m}\right)\delta_F$$

$$\dot{r} = \left[\frac{2(bC_{\alpha R} - aC_{\alpha F})}{IU}\right]v + \left[-\frac{2\left(a^2 C_{\alpha F} + b^2 C_{\alpha R}\right)}{IU}\right]r + \left(\frac{2aC_{\alpha F}}{I}\right)\delta_F$$

$$\dot{\psi} = r$$

In matrix notation:

$$\dot{x} = Fx + g\delta_F$$

where:

$$x = \begin{bmatrix} y \\ v \\ r \\ \psi \end{bmatrix}$$

$$F = \begin{bmatrix} 0 & 1 & 0 & U \\ 0 & -2\dfrac{C_{\alpha F} + C_{\alpha R}}{mU} & 2\dfrac{bC_{\alpha R} - aC_{\alpha F}}{mU} - U & 0 \\ 0 & 2\dfrac{bC_{\alpha R} - aC_{\alpha F}}{IU} & -2\dfrac{a^2 C_{\alpha F} + b^2 C_{\alpha R}}{IU} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$x = \begin{bmatrix} 0 \\ \dfrac{2C_{\alpha F}}{m} \\ \dfrac{2aC_{\alpha F}}{I} \\ 0 \end{bmatrix}$$

The single-point model assumes a minimum previewed error signal at a single point T* seconds ahead in time. a* is the driver ability to predict the future vehicle response based on the current steering control input. b* is the driver ability to predict the future vehicle response based on the current vehicle state. The block uses these equations.

$$a* = T*m^T\left[I + \sum_{n=1}^{\infty} \frac{F^n(T*)^n}{(n+1)!}\right]g$$

$$b* = m^T\left[I + \sum_{n=1}^{\infty} \frac{F^n(T*)^n}{n!}\right]$$

where:

$$m^T = [1\ 0\ 0\ 0]$$

The equations use these variables.

a, b	Forward and rearward tire location, respectively
m	Vehicle mass
I	Vehicle rotational inertia
$C_{\alpha F}$	Front tire cornering coefficient
$C_{\alpha R}$	Rear tire cornering coefficient
a*, ***b****	Driver prediction scalar and vector gain, respectively
x	Predicted vehicle state vector
v	Lateral velocity
r	Yaw rate
Ψ	Front wheel heading angle
y	Lateral displacement
F	System matrix
δ, δ_F	Steer angle and front axle steer angle, respectively
g	Control coefficient vector
U	Forward (longitudinal) vehicle velocity
T*	Preview time window
f(t+T*)	Previewed path input T* seconds ahead
U	Forward vehicle velocity
m^T	Constant observer vector; provides vehicle lateral position

Optimization

The single-point model implemented by the block finds the steering command that minimizes a local performance index, *J*, over the current preview interval, (*t, t+T*).

$$J = \frac{1}{T}\int_{t}^{t+T}[f(\eta) - y(\eta)]^2 d\eta$$

To minimize *J* with respect to the steering command, this condition must be met.

$$\frac{dJ}{du} = 0$$

You can express the optimal control solution in terms of a current non-optimal and corresponding nonzero preview output error *T** seconds ahead[1, 2, 3].

$$u^o(t) = u(t) + \frac{e(t + T^*)}{a^*}$$

The block uses the preview distance and vehicle longitudinal velocity to determine the preview time window.

$$T^* = \frac{L}{U}$$

The equations use these variables.

T^*	Preview time window
$f(t+T^*)$	Previewed path input T^* sec ahead
$y(t+T^*)$	Previewed plant output T^* sec ahead
$e(t+T^*)$	Previewed error signal T^* sec ahead
$u(t)$, $u^o(t)$	Steer angle and optimal steer angle, respectively
L	Preview distance
J	Performance index
U	Forward (longitudinal) vehicle velocity

Driver Lag

The single-point model implemented by the block introduces a driver lag. The driver lag accounts for the delay when the driver is tracking tasks. Specifically, it is the transport delay deriving from perceptual and neuromuscular mechanisms. To calculate the driver transport delay, the block implements this equation.

$$H(s) = e^{-s\tau}$$

The equations use these variables.

τ	Driver transport delay
$y(t+T^*)$	Previewed plant output T^* sec ahead
$e(t+T^*)$	Previewed error signal T^* sec ahead
$u(t)$, $u^o(t)$	Steer angle and optimal steer angle, respectively
J	Performance index

Ports

Input

LatRef — Lateral displacement reference
`scalar`

Lateral center of mass (CM) displacement reference, in the inertial reference frame, in m.

EnblSteerOvr — Enable steering command override
`scalar`

Enable steering command override.

Dependencies

To enable this port, select **Steering override**.

Data Types: `Boolean`

SteerOvrCmd — Steering override command
`scalar`

Steering override command.

Use the **Output handwheel angle** parameter to specify the units for the steering ports.

Setting	Block Implementation	Port
off (default)	Commanded steer angle, normalized from -1 through 1. The block uses the tire wheel angle saturation limit **Tire wheel angle limit, theta** parameter to normalize the command.	SteerCmd — Output
	Overrides the steering command with an input steering command normalized from -1 through 1.	SteerOvrCmd — Input
on	Commanded steer angle in rad.	SteerCmd — Output
	Overrides the steering command with an input steering command in rad.	SteerOvrCmd — Input

Dependencies

To enable this port, select **Steering override**.

Data Types: double

SteerHld — Steering hold
scalar

Boolean signal that holds the steering command at the current value.

Dependencies

To enable this port, select **Steering hold**.

Data Types: Boolean

SteerZero — Disable steering command
scalar

Disable steering command.

Dependencies

To enable this port, select **Steering disable**.

Data Types: Boolean

VelFdbk — Longitudinal vehicle velocity
scalar

Longitudinal vehicle velocity, U, in the vehicle-fixed frame, in m/s.

LatFdbk — Lateral displacement
scalar

Lateral CM displacement, y_o, in the inertial reference frame, in m.

LatVelFdbk — Lateral vehicle velocity
scalar

Lateral vehicle velocity, v_o, in the vehicle-fixed frame, in m/s.

YawFdbk — Vehicle yaw angle
scalar

Vehicle yaw angle, Ψ_o, in the inertial reference frame, in rad.

YawVelFdbk — Yaw rate
scalar

Yaw rate, r_o, in the vehicle-fixed frame, in rad/s.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal		Variable	Description
Predicted	y	y	Predicted lateral displacement, in the vehicle-fixed frame.
	ydot	v	Predicted lateral velocity, in the vehicle-fixed frame.
	psi	Ψ	Predicted front wheel heading angle.
	r	r	Predicted yaw rate, in the vehicle-fixed frame.
SteerCmd		δ_F	Commanded steer angle.
Err		e_{ref}	Difference in reference vehicle position and vehicle position.
ErrSqrSum		$\int\limits_0^t e_{ref}{}^2 dt$	Integrated square of error.
ErrMax		$\max(e_{ref}(t))$	Maximum error during simulation.
ErrMin		$\min(e_{ref}(t))$	Minimum error during simulation.
ExtActions	EnblSteerOvr		Override the steering command with an input deceleration command.
	SteerOvrCmd		Input steering override command
	SteerHld		Hold the steering command at the current value
	SteerZero		Disable the steering command

SteerCmd — Steer angle command
scalar

Commanded steer angle, δ_F.

Use the **Output handwheel angle** parameter to specify the units for the steering ports.

Setting	Block Implementation	Port
off (default)	Commanded steer angle, normalized from -1 through 1. The block uses the tire wheel angle saturation limit **Tire wheel angle limit, theta** parameter to normalize the command.	SteerCmd — Output
	Overrides the steering command with an input steering command normalized from -1 through 1.	SteerOvrCmd — Input
on	Commanded steer angle in rad.	SteerCmd — Output
	Overrides the steering command with an input steering command in rad.	SteerOvrCmd — Input

Parameters

Configuration

Steering override — Override steering command
off (default) | on

Select to override the steering command with an input steering command.

Dependencies

Selecting this parameter creates the EnblSteerOvr and SteerOvrCmd input ports.

Steering hold — Hold steering command
off (default) | on

Select to hold the steering command.

Dependencies

Selecting this parameter creates the SteerHld input port.

Steering disable — Disable steering command
off (default) | on

Select to disable the steering command.

Dependencies

Selecting this parameter creates the SteerZero input port.

Output handwheel angle — Steering port units in rad
off (default) | on

Use the **Output handwheel angle** parameter to specify the units for the steering ports.

Setting	Block Implementation	Port
off (default)	Commanded steer angle, normalized from -1 through 1. The block uses the tire wheel angle saturation limit **Tire wheel angle limit, theta** parameter to normalize the command.	SteerCmd — Output
	Overrides the steering command with an input steering command normalized from -1 through 1.	SteerOvrCmd — Input
on	Commanded steer angle in rad.	SteerCmd — Output
	Overrides the steering command with an input steering command in rad.	SteerOvrCmd — Input

Dependencies

To create the SteerOvrCmd input port, select **Steering override**.

Reference Control

Predictive

Forward location of tire, a — Along vehicle longitudinal axis
1.41 (default) | scalar

Forward location of tire, *a*, in m. Distance from vehicle cg to forward tire location, along vehicle longitudinal axis.

Rearward location of tire, b — Along vehicle longitudinal axis
1.41 (default) | scalar

Rearward location of tire, *b*, in m. Absolute value of distance from vehicle cg to rearward tire location, along vehicle longitudinal axis.

Vehicle mass, m — Mass
2016 (default) | scalar

Vehicle mass, *m*, in kg.

Vehicle rotational inertia, I — Inertia about yaw axis
4013 (default) | scalar

Vehicle rotational inertia, *I*, about the vehicle yaw axis, in N·m·s^2.

Front tire cornering coefficient, Cy_f — Coefficient
25266 (default) | scalar

Cornering stiffness coefficient, $C_{\alpha F}$, in N/rad.

Rear tire cornering coefficient, Cy_r — Coefficient
70933 (default) | scalar

Cornering stiffness coefficient, $C_{\alpha R}$, in N/rad.

Tire wheel angle limit, theta — Angle limit
45*pi/180 (default) | scalar

Tire wheel angle limit, θ, in rad.

Nominal steering ratio, Ksteer — Steering ratio
18 (default) | scalar

Steering ratio, K_{steer}. The value has no dimension.

Dependencies

To enable this parameter, select **Output handwheel angle**.

Driver response time, tau — Response time
0.1 (default) | scalar

Driver response time, τ, in s.

Preview distance, L — Distance
3 (default) | scalar

Driver preview distance, L, in m. Used to determine the preview time window, T^*.

References

[1] MacAdam, C. C. "An Optimal Preview Control for Linear Systems". *Journal of Dynamic Systems, Measurement, and Control*. 102(3), Sept 1980.

[2] MacAdam, C. C. "Application of an Optimal Preview Control for Simulation of Closed-Loop Automobile Driving ". *IEEE Transactions on Systems, Man, and Cybernetics*. Volume 11, Issue 6, June 1981.

[3] MacAdam, C. C. *Development of Driver/Vehicle Steering Interaction Models for Dynamic Analysis*. Final Technical Report UMTRI-88-53. The University of Michigan Transportation Research Institute. December, 1988.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also
Longitudinal Driver | Predictive Driver

Introduced in R2018a

Predictive Driver

Predictive driver controller to track longitudinal speed and lateral path
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Driver
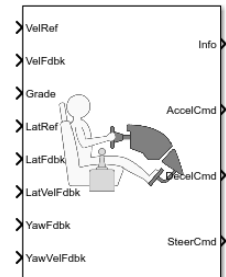
Description

The Predictive Driver block implements a controller that generates normalized steering, acceleration, and braking commands to track longitudinal velocity and a lateral reference displacement. The normalized commands can vary between -1 to 1. The controller uses a single-track (bicycle) model for optimal single-point preview control.

Configurations

External Actions

Use the **External Actions** parameters to create input ports for signals that you can use to simulate standard test maneuvers. The block uses this priority order for the input commands: disable (highest), hold, override.

This table summarizes the external action parameters.

Goal	External Action Parameter	Input Ports	Data Type
Override the accelerator command with an input acceleration command.	**Accelerator override**	EnablAccelOvr	Boolean
		AccelOvrCmd	double
Hold the acceleration command at the current value.	**Accelerator hold**	AccelHld	Boolean
Disable the acceleration command.	**Accelerator disable**	AccelZero	Boolean
Override the decelerator command with an input deceleration command.	**Decelerator override**	EnablDecelOvr	Boolean
		DecelOvrCmd	double
Hold the decelerator command at current value.	**Decelerator hold**	DecelHld	Boolean
Disable the decelerator command.	**Decelerator disable**	DecelZero	Boolean
Override the steering command with an input steering command.	**Steering override**	EnblSteerOvr	Boolean

Goal	External Action Parameter	Input Ports	Data Type
		`SteerOvrCmd`	`double`
Hold the steering command at the current value.	**Steering hold**	`SteerHld`	`Boolean`
Disable the steering command.	**Steering disable**	`SteerZero`	`Boolean`

Controller

Use the **Longitudinal control type, cntrlType** parameter to specify one of these control options.

Setting	Block Implementation
`PI`	Proportional-integral (PI) control with tracking windup and feed-forward gains.
`Scheduled PI`	PI control with tracking windup and feed-forward gains that are a function of vehicle velocity.
`Predictive`	Optimal single-point preview (look ahead) control model developed by C. C. MacAdam[1, 2, 3]. The model represents driver steering control behavior during path-following and obstacle avoidance maneuvers. Drivers preview (look ahead) to follow a predefined path. To implement the MacAdam model, the block: • Represents the dynamics as a linear single track (bicycle) vehicle • Minimizes the previewed error signal at a single point T^* seconds ahead in time • Accounts for the driver lag deriving from perceptual and neuromuscular mechanisms

Shift

Use the **Shift type, ShftType** parameter to specify one of these shift options.

Setting	Block Implementation
`None`	No transmission. Block outputs a constant gear of 1. Use this setting to minimize the number of parameters you need to generate acceleration and braking commands to track forward vehicle motion. This setting does not allow reverse vehicle motion.

Setting	Block Implementation
Reverse, Neutral, Drive	Block uses a Stateflow chart to model reverse, neutral, and drive gear shift scheduling. Use this setting to generate acceleration and braking commands to track forward and reverse vehicle motion using simple reverse, neutral, and drive gear shift scheduling. Depending on the vehicle state and vehicle velocity feedback, the block uses the initial gear and time required to shift to shift the vehicle up into drive or down into reverse or neutral. For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.
Scheduled	Block uses a Stateflow chart to model reverse, neutral, park, and N-speed gear shift scheduling. Use this setting to generate acceleration and braking commands to track forward and reverse vehicle motion using reverse, neutral, park, and N-speed gear shift scheduling. Depending on the vehicle state and vehicle velocity feedback, the block uses these parameters to determine the: • Initial gear • Upshift and downshift accelerator pedal positions • Upshift and downshift velocity • Timing for shifting and engaging forward and reverse from neutral For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.
External	Block uses the input gear, vehicle state, and velocity feedback to generate acceleration and braking commands to track forward and reverse vehicle motion. For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.

Gear Signal

Use the **Output gear signal** parameter to create the `GearCmd` output port. The `GearCmd` signal contains the integer value of the commanded vehicle gear.

Gear	Integer
Park	80
Reverse	-1
Neutral	0
Drive	1
Gear	Gear number

Output Handwheel Angle

Use the **Output handwheel angle** parameter to specify the units for the steering ports.

Setting	Block Implementation	Port
off (default)	Commanded steer angle, normalized from -1 through 1. The block uses the tire wheel angle saturation limit **Tire wheel angle limit, theta** parameter to normalize the command.	SteerCmd — Output
	Overrides the steering command with an input steering command normalized from -1 through 1.	SteerOvrCmd — Input
on	Commanded steer angle in rad.	SteerCmd — Output
	Overrides the steering command with an input steering command in rad.	SteerOvrCmd — Input

Controller: PI Speed-Tracking

If you set the control type to PI or Scheduled PI, the block implements proportional-integral (PI) control with tracking windup and feed-forward gains. For the Scheduled PI configuration, the block uses feed forward gains that are a function of vehicle velocity.

To calculate the speed control output, the block uses these equations.

Setting	Equation
PI	$$y = \frac{K_{ff}}{v_{nom}}v_{ref} + \frac{K_p e_{ref}}{v_{nom}} + \int\left(\frac{K_i e_{ref}}{v_{nom}} + K_{aw}e_{out}\right)dt + K_g\theta$$
Scheduled PI	$$y = \frac{K_{ff}(v)}{v_{nom}}v_{ref} + \frac{K_p(v)e_{ref}}{v_{nom}} + \int\left(\frac{K_i(v)e_{ref}}{v_{nom}} + K_{aw}e_{out}\right)e_{ref}dt + K_g(v)\theta$$

where:

$$e_{ref} = v_{ref} - v$$

$$e_{out} = y_{sat} - y$$

$$y_{sat} = \begin{cases} -1 & y < -1 \\ y & -1 \leq y \leq 1 \\ 1 & 1 < y \end{cases}$$

The velocity error low-pass filter uses this transfer function.

$$H(s) = \frac{1}{\tau_{err}s + 1} \quad \text{for} \quad \tau_{err} > 0$$

To calculate the acceleration and braking commands, the block uses these equations.

$$y_{acc} = \begin{cases} 0 & y_{sat} < 0 \\ y_{sat} & 0 \le y_{sat} \le 1 \\ 1 & 1 < y_{sat} \end{cases}$$

$$y_{dec} = \begin{cases} 0 & y_{sat} > 0 \\ -y_{sat} & -1 \le y_{sat} \le 0 \\ 1 & y_{sat} < -1 \end{cases}$$

The equations use these variables.

v_{nom}	Nominal vehicle speed
K_p	Proportional gain
K_i	Integral gain
K_{aw}	Anti-windup gain
K_{ff}	Velocity feed-forward gain
K_g	Grade feed-forward gain
θ	Grade angle
τ_{err}	Error filter time constant
y	Nominal control output magnitude
y_{sat}	Saturated control output magnitude
e_{ref}	Velocity error
e_{out}	Difference between saturated and nominal control outputs
y_{acc}	Acceleration signal
y_{dec}	Braking signal
v	Velocity feedback signal
v_{ref}	Reference velocity signal

Controller: Predictive Speed-Tracking

If you set the control type to `Predictive`, the block implements an optimal single-point preview (look ahead) control model developed by C. C. MacAdam[1, 2, 3]. The model represents driver steering control behavior during path-following and obstacle avoidance maneuvers. Drivers preview (look ahead) to follow a predefined path. To implement the MacAdam model, the block:

- Represents the dynamics as a linear single track (bicycle) vehicle
- Minimizes the previewed error signal at a single point T^* seconds ahead in time
- Accounts for the driver lag deriving from perceptual and neuromuscular mechanisms

Vehicle Dynamics

For lateral and yaw motion, the block implements these linear dynamic equations.

$$x_1 = U$$

$$\dot{x}_1 = x_2 = \frac{K_{pt}}{m} + \quad vr - g\sin(\gamma) + F_r x_1$$

$$\dot{y} = v + U\psi$$

$$\dot{v} = \left[-\frac{2(C_{\alpha F} + C_{\alpha R})}{mU}\right]v + \left[\frac{2(bC_{\alpha R} - aC_{\alpha F})}{mU} - U\right]r + \left(\frac{2C_{\alpha F}}{m}\right)\delta_F$$

$$\dot{r} = \left[\frac{2(bC_{\alpha R} - aC_{\alpha F})}{IU}\right]v + \left[-\frac{2\left(a^2C_{\alpha F} + b^2C_{\alpha R}\right)}{IU}\right]r + \left(\frac{2aC_{\alpha F}}{I}\right)\delta_F$$

$$\dot{\psi} = r$$

In matrix notation:

$$\dot{x} = Fx + gu$$

where:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ y \\ v \\ r \\ \psi \end{bmatrix}$$

$$F = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{F_r}{m} & 0 & 0 & 0 & v & 0 \\ 0 & 0 & 0 & 1 & 0 & U \\ 0 & 0 & 0 & -\frac{2(C_{\alpha F} + C_{\alpha R})}{mU} & \frac{2(bC_{\alpha R} - aC_{\alpha F})}{mU} - U & 0 \\ 0 & 0 & 0 & \frac{2(bC_{\alpha R} - aC_{\alpha F})}{IU} & -\frac{2\left(a^2C_{\alpha F} + b^2C_{\alpha R}\right)}{IU} & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$g = \begin{bmatrix} 0 & 0 \\ \frac{K_{pt}}{m} & 0 \\ 0 & 0 \\ 0 & \frac{2C_{\alpha F}}{m} \\ 0 & \frac{2aC_{\alpha F}}{I} \\ 0 & 0 \end{bmatrix}$$

$$u = \begin{bmatrix} \bar{u} \\ \delta_F \end{bmatrix}$$

$$\bar{u} = u - \frac{m^2}{K_{pt}}g\sin(\gamma)$$

The single-point model assumes a minimum previewed error signal at a single point $T*$ seconds ahead in time. $a*$ is the driver ability to predict the future vehicle response based on the current steering control input. $b*$ is the driver ability to predict the future vehicle response based on the current vehicle state. The block uses these equations.

$$a* = (T*)m^T[I + \sum_{n = 1}^{\infty} \frac{F^n(T*)^n}{(n + 1)!}]g$$

$$b* = m^T[I + \sum_{n = 1}^{\infty} \frac{F^n(T*)^n}{n!}]$$

$$m^T = [1 \ 1 \ 1 \ 0 \ 0 \ 0]$$

The equations use these variables.

a, b	Forward and rearward tire location, respectively
m	Vehicle mass
I	Vehicle rotational inertia
$C_{\alpha F}$	Front tire cornering coefficient
$C_{\alpha R}$	Rear tire cornering coefficient
$a*$, $\boldsymbol{b}*$	Driver prediction scalar and vector gain, respectively
$\boldsymbol{x}$	Predicted vehicle state vector
v	Lateral velocity
r	Yaw rate
Ψ	Front wheel heading angle
y	Lateral displacement
$\boldsymbol{F}$	System matrix
δ, δ_F	Steer angle and front axle steer angle, respectively
γ	Grade angle
$\boldsymbol{g}$	Control coefficient vector
U	Forward (longitudinal) vehicle velocity
$T*$	Preview time window
$f(t+T*)$	Previewed path input T* seconds ahead
$\boldsymbol{u}$	Tractive force
$\boldsymbol{m^T}$	Constant observer vector; provides vehicle lateral position
a_r	Static rolling and driveline resistance
b_r	Linear rolling and driveline resistance
c_r	Aerodynamic rolling and driveline resistance
F_r	Rolling resistance

Optimization

The single-point model implemented by the block finds the steering command that minimizes a local performance index, J, over the current preview interval, $(t, t+T)$.

$$J = \frac{1}{T} \int_t^{t+T} [f(\eta) - y(\eta)]^2 d\eta$$

To minimize J with respect to the steering command, this condition must be met.

$$\frac{dJ}{du} = 0$$

You can express the optimal control solution in terms of a current non-optimal and corresponding nonzero preview output error T^* seconds ahead[1, 2, 3].

$$u^o(t) = u(t) + \frac{e(t + T^*)}{a^*}$$

The block uses the preview distance and vehicle longitudinal velocity to determine the preview time window.

$$T^* = \frac{L}{U}$$

The equations use these variables.

T^*	Preview time window
$f(t+T^*)$	Previewed path input T^* sec ahead
$y(t+T^*)$	Previewed plant output T^* sec ahead
$e(t+T^*)$	Previewed error signal T^* sec ahead
$u(t), u^o(t)$	Steer angle and optimal steer angle, respectively
L	Preview distance
J	Performance index
U	Forward (longitudinal) vehicle velocity

Driver Lag

The single-point model implemented by the block introduces a driver lag. The driver lag accounts for the delay when the driver is tracking tasks. Specifically, it is the transport delay deriving from perceptual and neuromuscular mechanisms. To calculate the driver transport delay, the block implements this equation.

$$H(s) = e^{-s\tau}$$

The equations use these variables.

τ	Driver transport delay
$y(t+T^*)$	Previewed plant output T^* sec ahead
$e(t+T^*)$	Previewed error signal T^* sec ahead
$u(t), u^o(t)$	Steer angle and optimal steer angle, respectively
J	Performance index

Ports

Input

VelRef — Reference vehicle velocity
scalar

Reference velocity, v_{ref}, in m/s.

LatRef — Lateral displacement reference
scalar

Lateral center of mass (CM) displacement reference, in the inertial reference frame, in m.

EnblSteerOvr — Enable steering command override
scalar

Enable steering command override.

Dependencies

To enable this port, select **Steering override**.

Data Types: Boolean

SteerOvrCmd — Steering override command
scalar

Steering override command.

Use the **Output handwheel angle** parameter to specify the units for the steering ports.

Setting	Block Implementation	Port
off (default)	Commanded steer angle, normalized from -1 through 1. The block uses the tire wheel angle saturation limit **Tire wheel angle limit, theta** parameter to normalize the command.	SteerCmd — Output
	Overrides the steering command with an input steering command normalized from -1 through 1.	SteerOvrCmd — Input
on	Commanded steer angle in rad.	SteerCmd — Output
	Overrides the steering command with an input steering command in rad.	SteerOvrCmd — Input

Dependencies

To enable this port, select **Steering override**.

Data Types: double

SteerHld — Steering hold
scalar

Boolean signal that holds the steering command at the current value.

Dependencies

To enable this port, select **Steering hold**.

Data Types: `Boolean`

SteerZero — Disable steering command
`scalar`

Disable steering command.

Dependencies

To enable this port, select **Steering disable**.

Data Types: `Boolean`

EnblAccelOvr — Enable acceleration command override
`scalar`

Enable acceleration command override.

Dependencies

To enable this port, select **Acceleration override**.

Data Types: `Boolean`

AccelOvrCmd — Acceleration override command
`scalar`

Acceleration override command, normalized from 0 through 1.

Dependencies

To enable this port, select **Acceleration override**.

Data Types: `double`

AccelHld — Acceleration hold
`scalar`

Boolean signal that holds the acceleration command at the current value.

Dependencies

To enable this port, select **Acceleration hold**.

Data Types: `Boolean`

AccelZero — Disable acceleration command
`scalar`

Disable acceleration command.

Dependencies

To enable this port, select **Acceleration disable**.

Data Types: `Boolean`

EnblDecelOvr — Enable deceleration command override
scalar

Enable deceleration command override.

Dependencies

To enable this port, select **Deceleration override**.

Data Types: Boolean

DecelOvrCmd — Deceleration override command
scalar

Deceleration override command, normalized from 0 through 1.

Dependencies

To enable this port, select **Deceleration override**.

Data Types: double

DecelHld — Deceleration hold
scalar

Boolean signal that holds the deceleration command at the current value.

Dependencies

To enable this port, select **Deceleration hold**.

Data Types: Boolean

DecelZero — Disable deceleration command
scalar

Disable deceleration command.

Dependencies

To enable this port, select **Deceleration disable**.

Data Types: Boolean

ExtGear — Gear
scalar

Gear	Integer
Park	80
Reverse	-1
Neutral	0
Drive	1
Gear	Gear number

Dependencies

To enable this port, set **Shift type, shftType** to External.

Grade — Road grade angle
scalar

Road grade angle, γ, in deg.

VelFdbk — Longitudinal vehicle velocity
scalar

Longitudinal vehicle velocity, U, in the vehicle-fixed frame, in m/s.

LatFdbk — Lateral displacement
scalar

Lateral CM displacement, y_o, in the inertial reference frame, in m.

LatVelFdbk — Lateral vehicle velocity
scalar

Lateral vehicle velocity, v_o , in the vehicle-fixed frame, in m/s.

YawFdbk — Vehicle yaw angle
scalar

Vehicle yaw angle, Ψ_o, in the inertial reference frame, in rad.

YawVelFdbk — Yaw rate
scalar

Yaw rate, r_o, in the vehicle-fixed frame, in rad/s.

Output

Info — Bus signal
bus

Bus signal containing these block calculations.

Signal			Variable	Description
Steer			δ_F	Commanded steer angle, normalized from 0 through 1
Accel			y_{acc}	Commanded vehicle acceleration, normalized from 0 through 1
Decel			y_{dec}	Commanded vehicle deceleration, normalized from 0 through 1
Gear				Integer value of commanded gear
Clutch				Clutch command
Err	LatErr	Err	e_{ref}	Difference in reference vehicle position and vehicle position.
		ErrSqrSum	$\int_0^t e_{ref}{}^2 dt$	Integrated square of error.

Signal			Variable	Description
		ErrMax	$\max(e_{ref}(t))$	Maximum error during simulation.
		ErrMin	$\min(e_{ref}(t))$	Minimum error during simulation.
	LngErr	Err	e_{ref}	Difference in reference vehicle speed and vehicle speed
		ErrSqrSum	$\int\limits_{0}^{t} e_{ref}{}^2 dt$	Integrated square of error
		ErrMax	$\max(e_{ref}(t))$	Maximum error during simulation
		ErrMin	$\min(e_{ref}(t))$	Minimum error during simulation
ExtActions	EnblSteerOvr			Override the steering command with an input deceleration command
	SteerOvrCmd			Input steering override command
	SteerHld			Hold the steering command at the current value
	SteerZero			Disable the steering command
	EnblAccelOvr			Override the accelerator command with an input acceleration command
	AccelOvrCmd			Input accelerator override command
	AccelHld			Hold the acceleration command at the current value
	AccelZero			Disable the acceleration command
	EnblDecelOvr			Override the decelerator command with an input deceleration command
	DecelOvrCmd			Input deceleration override command
	DecelHld			Hold the decelerator command at current value
	DecelZero			Disable the decelerator command

SteerCmd — Steer angle command
scalar

Commanded steer angle, δ_F.

Use the **Output handwheel angle** parameter to specify the units for the steering ports.

Setting	Block Implementation	Port
off (default)	Commanded steer angle, normalized from -1 through 1. The block uses the tire wheel angle saturation limit **Tire wheel angle limit, theta** parameter to normalize the command.	SteerCmd — Output
	Overrides the steering command with an input steering command normalized from -1 through 1.	SteerOvrCmd — Input

Setting	Block Implementation	Port
on	Commanded steer angle in rad.	`SteerCmd` — Output
	Overrides the steering command with an input steering command in rad.	`SteerOvrCmd` — Input

`AccelCmd` — Commanded vehicle acceleration
scalar

Commanded vehicle acceleration, y_{acc}, normalized from 0 through 1.

`DecelCmd` — Commanded vehicle deceleration
scalar

Commanded vehicle deceleration, y_{dec}, normalized from 0 through 1.

`GearCmd` — Commanded vehicle gear
scalar

Integer value of commanded vehicle gear.

Gear	Integer
Park	`80`
Reverse	`-1`
Neutral	`0`
Drive	`1`
Gear	`Gear number`

Dependencies

To enable this port, select **Output gear signal**.

Parameters

Configuration

External Actions

`Accelerator override` — Override acceleration command
off (default) | on

Select to override the acceleration command with an input acceleration command.

Dependencies

Selecting this parameter creates the `EnblAccelOvr` and `AccelOvrCmd` input ports.

`Accelerator hold` — Hold acceleration command
off (default) | on

Select to hold the acceleration command.

Dependencies

Selecting this parameter creates the `AccelHld` input port.

Accelerator disable — Disable acceleration command
off (default) | on

Select to disable the acceleration command.

Dependencies

Selecting this parameter creates the `AccelZero` input port.

Decelerator override — Override deceleration command
off (default) | on

Select to override the deceleration command with an input deceleration command.

Dependencies

Selecting this parameter creates the `EnblDecelOvr` and `DecelOvrCmd` input ports.

Decelerator hold — Hold deceleration command
off (default) | on

Select to hold the deceleration command.

Dependencies

Selecting this parameter creates the `DecelHld` input port.

Decelerator disable — Disable deceleration command
off (default) | on

Select to disable the deceleration command.

Dependencies

Selecting this parameter creates the `DecelZero` input port.

Steering override — Override steering command
off (default) | on

Select to override the steering command with an input steering command.

Dependencies

Selecting this parameter creates the `EnblSteerOvr` and `SteerOvrCmd` input ports.

Steering hold — Hold steering command
off (default) | on

Select to hold the steering command.

Dependencies

Selecting this parameter creates the `SteerHld` input port.

Steering disable — Disable steering command
off (default) | on

Select to disable the steering command.

Dependencies

Selecting this parameter creates the SteerZero input port.

Control and Shift

Longitudinal control type, cntrlType — Longitudinal control
PI (default) | Scheduled PI | Predictive

Type of longitudinal control.

Setting	Block Implementation
PI	Proportional-integral (PI) control with tracking windup and feed-forward gains.
Scheduled PI	PI control with tracking windup and feed-forward gains that are a function of vehicle velocity.
Predictive	Optimal single-point preview (look ahead) control model developed by C. C. MacAdam[1, 2, 3]. The model represents driver steering control behavior during path-following and obstacle avoidance maneuvers. Drivers preview (look ahead) to follow a predefined path. To implement the MacAdam model, the block: • Represents the dynamics as a linear single track (bicycle) vehicle • Minimizes the previewed error signal at a single point $T*$ seconds ahead in time • Accounts for the driver lag deriving from perceptual and neuromuscular mechanisms

Shift type, shftType — Shift type
None (default) | Reverse, Neutral, Drive | Scheduled | External

Shift type.

Setting	Block Implementation
None	No transmission. Block outputs a constant gear of 1. Use this setting to minimize the number of parameters you need to generate acceleration and braking commands to track forward vehicle motion. This setting does not allow reverse vehicle motion.

Setting	Block Implementation
Reverse, Neutral, Drive	Block uses a Stateflow chart to model reverse, neutral, and drive gear shift scheduling. Use this setting to generate acceleration and braking commands to track forward and reverse vehicle motion using simple reverse, neutral, and drive gear shift scheduling. Depending on the vehicle state and vehicle velocity feedback, the block uses the initial gear and time required to shift to shift the vehicle up into drive or down into reverse or neutral. For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.
Scheduled	Block uses a Stateflow chart to model reverse, neutral, park, and N-speed gear shift scheduling. Use this setting to generate acceleration and braking commands to track forward and reverse vehicle motion using reverse, neutral, park, and N-speed gear shift scheduling. Depending on the vehicle state and vehicle velocity feedback, the block uses these parameters to determine the: • Initial gear • Upshift and downshift accelerator pedal positions • Upshift and downshift velocity • Timing for shifting and engaging forward and reverse from neutral For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.
External	Block uses the input gear, vehicle state, and velocity feedback to generate acceleration and braking commands to track forward and reverse vehicle motion. For neutral gears, the block uses braking commands to control the vehicle speed. For reverse gears, the block uses an acceleration command to generate torque and a brake command to reduce vehicle speed.

Longitudinal velocity units, velUnits — Velocity units
m/s (default)

Vehicle velocity reference and feedback units.

Dependencies

If you set **Longitudinal control type, CntrlType** control type to Scheduled or Scheduled PI, the block uses the **Longitudinal velocity units, velUnits** for the **Nominal speed, vnom** parameter dimension.

If you set **Shift Type, shftType** to Scheduled, the block uses the **Longitudinal velocity units, velUnits** for these parameter dimensions:

• **Upshift velocity data table, upShftTbl**

- **Downshift velocity data table, dwnShftTbl**

Output gear signal — Create GearCmd output port
off (default) | on

Specify to create output port GearCmd.

Output handwheel angle — Steering port units in rad
off (default) | on

Use the **Output handwheel angle** parameter to specify the units for the steering ports.

Setting	Block Implementation	Port
off (default)	Commanded steer angle, normalized from -1 through 1. The block uses the tire wheel angle saturation limit **Tire wheel angle limit, theta** parameter to normalize the command.	SteerCmd — Output
	Overrides the steering command with an input steering command normalized from -1 through 1.	SteerOvrCmd — Input
on	Commanded steer angle in rad.	SteerCmd — Output
	Overrides the steering command with an input steering command in rad.	SteerOvrCmd — Input

Dependencies

To create the SteerOvrCmd input port, select **Steering override**.

Reference Control

Longitudinal

Proportional gain, Kp — Gain
10 (default) | scalar

Proportional gain, K_p, dimensionless.

Dependencies

To create this parameter, set **Control type** to PI.

Integral gain, Ki — Gain
5 (default) | scalar

Proportional gain, K_i, dimensionless.

Dependencies

To create this parameter, set **Control type** to PI.

Velocity feed-forward, Kff — Gain
.1 (default) | scalar

Velocity feed-forward gain, K_{ff}, dimensionless.

Dependencies

To create this parameter, set **Control type** to PI.

Grade feed-forward, Kg — Gain
0 (default) | scalar

Grade feed-forward gain, K_g, in 1/deg.

Dependencies

To create this parameter, set **Control type** to PI.

Velocity gain breakpoints, VehVelVec — Breakpoints
[0 100] (default) | vector

Velocity gain breakpoints, *VehVelVec*, dimensionless.

Dependencies

To create this parameter, set **Control type** to Scheduled PI.

Velocity feed-forward gain values, KffVec — Gain
[.1 .1] (default) | vector

Velocity feed-forward gain values, *KffVec*, as a function of vehicle velocity, dimensionless.

Dependencies

To create this parameter, set **Control type** to Scheduled PI.

Proportional gain values, KpVec — Gain
[10 10] (default) | vector

Proportional gain values, *KpVec*, as a function of vehicle velocity, dimensionless.

Dependencies

To create this parameter, set **Control type** to Scheduled PI.

Integral gain values, KiVec — Gain
[5 5] (default) | vector

Integral gain values, *KiVec*, as a function of vehicle velocity, dimensionless.

Dependencies

To create this parameter, set **Control type** to Scheduled PI.

Grade feed-forward values, KgVec — Grade gain
[0 0] (default) | vector

Grade feed-forward values, *KgVec*, as a function of vehicle velocity, in 1/deg.

Dependencies

To create this parameter, set **Control type** to Scheduled PI.

Nominal speed, vnom — Nominal vehicle speed
5 (default) | scalar

Nominal vehicle speed, v_{nom}, in units specified by the **Reference and feedback units, velUnits** parameter. The block uses the nominal speed to normalize the controller gains.

Dependencies

To create this parameter, set **Control type** to PI or Scheduled PI.

Anti-windup, Kaw — Gain
1 (default) | scalar

Anti-windup gain, K_{aw}, dimensionless.

Dependencies

To create this parameter, set **Control type** to PI or Scheduled PI.

Error filter time constant, tauerr — Filter
.01 (default) | scalar

Error filter time constant, τ_{err}, in s. To disable the filter, enter 0.

Dependencies

To create this parameter, set **Control type** to PI or Scheduled PI.

Predictive

Forward location of tire, a — Along vehicle longitudinal axis
1.41 (default) | scalar

Forward location of tire, a, in m. Distance from vehicle cg to forward tire location, along vehicle longitudinal axis.

Rearward location of tire, b — Along vehicle longitudinal axis
1.41 (default) | scalar

Rearward location of tire, b, in m. Absolute value of distance from vehicle cg to rearward tire location, along vehicle longitudinal axis.

Vehicle mass, m — Mass
2016 (default) | scalar

Vehicle mass, m, in kg.

Vehicle rotational inertia, I — Inertia about yaw axis
4013 (default) | scalar

Vehicle rotational inertia, I, about the vehicle yaw axis, in N·m·s^2.

Front tire cornering coefficient, Cy_f — Coefficient
25266 (default) | scalar

Cornering stiffness coefficient, $C_{\alpha F}$, in N/rad.

Rear tire cornering coefficient, Cy_r — Coefficient
70933 (default) | scalar

Cornering stiffness coefficient, $C_{\alpha R}$, in N/rad.

Tire wheel angle limit, theta — Angle limit
45*pi/180 (default) | scalar

Tire wheel angle limit, θ, in rad.

Nominal steering ratio, Ksteer — Steering ratio
18 (default) | scalar

Steering ratio, K_{steer}. The value has no dimension.

Dependencies

To enable this parameter, select **Output handwheel angle**.

Driver response time, tau — Response time
0.1 (default) | scalar

Driver response time, τ, in s.

Preview distance, L — Distance
3 (default) | scalar

Driver preview distance, L, in m. Used to determine the preview time window, T^*.

Effective vehicle total tractive force, Kp — Tractive force
3000 (default) | scalar

Effective vehicle total tractive force, K_p, in N.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to Predictive.

Rolling resistance coefficient, aR — Resistance
200 (default) | scalar

Static rolling and driveline resistance coefficient, a_R, in N. Block uses the parameter to estimate the constant acceleration or braking effort.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to Predictive.

Rolling and driveline resistance coefficient, bR — Resistance
2.5 (default) | scalar

Rolling and driveline resistance coefficient, b_R, in N·s/m. Block uses the parameter to estimate the linear velocity-dependent acceleration or braking effort.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to Predictive.

Aerodynamic drag coefficient, cR — Drag
.5 (default) | scalar

Aerodynamic drag coefficient, c_R, in N·s^2/m^2. Block uses the parameter to estimate the quadratic velocity-dependent acceleration or braking effort.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to Predictive.

Gravitational constant, g — Gravitational constant
9.81 (default) | scalar

Gravitational constant, g, in m/s^2.

Dependencies

To create this parameter, set **Longitudinal control type, cntrlType** to Predictive.

Shift

Reverse, Neutral, Drive

Initial gear, GearInit — Initial gear
0 (default) | scalar

Integer value of the initial gear. The block uses the initial gear to generate acceleration and braking commands to track forward and reverse vehicle motion.

Gear	Integer
Park	80
Reverse	-1
Neutral	0
Drive	1
Gear	Gear number

Dependencies

To create this parameter, set **Shift type, shftType** to Reverse, Neutral, Drive or Scheduled. If you specify Reverse, Neutral, Drive, the **Initial Gear, GearInit** parameter value can be only -1, 0, or 1.

Time required to shift, tShift — Time
.1 (default) | scalar

Time required to shift, *tShift*, in s. The block uses the time required to shift to generate acceleration and braking commands to track forward and reverse vehicle motion using reverse, neutral, and drive gear shift scheduling.

Dependencies

To create this parameter, set **Shift type, shftType** to Reverse, Neutral, Drive.

Scheduled

`Initial gear, GearInit` — Initial gear
`0` (default) | `scalar`

Integer value of the initial gear. The block uses the initial gear to generate acceleration and braking commands to track forward and reverse vehicle motion.

Gear	Integer
Park	`80`
Reverse	`-1`
Neutral	`0`
Drive	`1`
Gear	`Gear number`

Dependencies

To create this parameter, set **Shift type, shftType** to `Reverse, Neutral, Drive` or `Scheduled`. If you specify `Reverse, Neutral, Drive`, the **Initial Gear, GearInit** parameter value can be only `-1`, `0`, or `1`.

`Up and down shift accelerator pedal positions, pdlVec` — Pedal position breakpoints
`[0.1 0.4 0.5 0.9]` (default) | `[1-by-m] vector`

Pedal position breakpoints for lookup tables when calculating upshift and downshift velocities, dimensionless. Vector dimensions are 1 by the number of pedal position breakpoints, m.

Dependencies

To create this parameter, set **Shift type, shftType** to `Scheduled`.

`Upshift velocity data table, upShftTbl` — Table
`[m-by-n] array`

Upshift velocity data as a function of pedal position and gear, in units specified by the **Reference and feedback units, velUnits** parameter. Upshift velocities indicate the vehicle velocity at which the gear should increase by 1.

The array dimensions are m pedal positions by n gears. The first column of data, when n equals 1, is the upshift velocity for the neutral gear.

Dependencies

To create this parameter, set **Shift type, shftType** to `Scheduled`.

`Downshift velocity data table, dwnShftTbl` — Table
`[m-by-n] array`

Downshift velocity data as a function of pedal position and gear, in units specified by the **Reference and feedback units, velUnits** parameter. Downshift velocities indicate the vehicle velocity at which the gear should decrease by 1.

The array dimensions are m pedal positions by n gears. The first column of data, when n equals 1, is the downshift velocity for the neutral gear.

Dependencies

To create this parameter, set **Shift type, shftType** to Scheduled.

Time required to shift, tClutch — Time
.5 (default) | scalar

Time required to shift, t_{Clutch}, in s.

Dependencies

To create this parameter, set **Shift type, shftType** to Scheduled.

Time required to engage reverse from neutral, tRev — Time
.5 (default) | scalar

Time required to engage reverse from neutral, t_{Rev}, in s.

Dependencies

To create this parameter, set **Shift type, shftType** to Scheduled.

Time required to engage park from neutral, tPark — Time
120 (default) | scalar

Time required to engage park from neutral, t_{Park}, in s.

Dependencies

To create this parameter, set **Shift type, shftType** to Scheduled.

References

[1] MacAdam, C. C. "An Optimal Preview Control for Linear Systems". *Journal of Dynamic Systems, Measurement, and Control*. Vol. 102, Number 3, Sept. 1980.

[2] MacAdam, C. C. "Application of an Optimal Preview Control for Simulation of Closed-Loop Automobile Driving ". *IEEE Transactions on Systems, Man, and Cybernetics*. Vol. 11, Issue 6, June 1981.

[3] MacAdam, C. C. *Development of Driver/Vehicle Steering Interaction Models for Dynamic Analysis*. Final Technical Report UMTRI-88-53. Ann Arbor, Michigan: The University of Michigan Transportation Research Institute, Dec. 1988.

Extended Capabilities

C/C++ Code Generation
Generate C and C++ code using Simulink® Coder™.

See Also

Lateral Driver | Longitudinal Driver

Introduced in R2018a

3D Simulation Blocks

Simulation 3D Actor Transform Get

Get actor translation, rotation, scale
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D / Sim3D Core
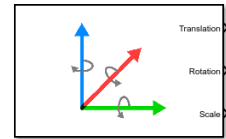
Description

The Simulation 3D Actor Transform Get block provides the actor translation, rotation, and scale for the Simulink simulation environment.

The block uses a vehicle-fixed coordinate system that is initially aligned with the inertial world coordinate system.

Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about *X*-axis
Y	Extends to the right of the vehicle, initially parallel to the ground plane Pitch — Right-handed rotation about *Y*-axis
Z	Extends upwards Yaw — Left-handed rotation about *Z*-axis

Actors are scene objects that support 3D translation, rotation, and scale. Parts are actor components. Components do not exist by themselves; they are associated with an actor.

Tip Verify that the Simulation 3D Scene Configuration block executes before the Simulation 3D Actor Transform Get block. That way, the Unreal Engine® 3D visualization environment prepares the data before the Simulation 3D Actor Transform Get block receives it. To check the block execution order, right-click the blocks and select **Properties**. On the **General** tab, confirm these **Priority** settings:

- Simulation 3D Scene Configuration — `0`
- Simulation 3D Actor Transform Get — `1`

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Ports

Output

`Translation` — Actor translation
`array`

Actor translation, in m. Array dimensions are number of parts per actor-by-3.

- `Translation(1,1)`, `Translation(1,2)`, and `Translation(1,3)` — Vehicle displacement along world X-, Y, and Z- axes, respectively.
- `Translation(...,1)`, `Translation(...,2)`, and `Translation(...,3)` — Actor displacement relative to vehicle, in vehicle-fixed coordinate system initially aligned with world X-, Y, and Z- axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The `Translation` signal:

- Dimensions are `[5x3]`.
- Contains translation information according to the axle and wheel locations, relative to vehicle.

$$Translation = \begin{bmatrix} X_v & Y_v & Z_v \\ X_{FL} & Y_{FL} & Z_{FL} \\ X_{FR} & Y_{FR} & Z_{FR} \\ X_{RL} & Y_{RL} & Z_{RL} \\ X_{RR} & Y_{RR} & Z_{RR} \end{bmatrix}$$

Translation	Array Element
Vehicle, X_v	`Translation(1,1)`
Vehicle, Y_v	`Translation(1,2)`
Vehicle, Z_v	`Translation(1,3)`
Front left wheel, X_{FL}	`Translation(2,1)`
Front left wheel, Y_{FL}	`Translation(2,2)`
Front left wheel, Z_{FL}	`Translation(2,3)`
Front right wheel, X_{FR}	`Translation(3,1)`
Front right wheel, Y_{FR}	`Translation(3,2)`
Front right wheel, Z_{FR}	`Translation(3,3)`
Rear left wheel, X_{RL}	`Translation(4,1)`
Rear left wheel, Y_{RL}	`Translation(4,2)`
Rear left wheel, Z_{RL}	`Translation(4,3)`
Rear right wheel, X_{RR}	`Translation(5,1)`

Translation	Array Element
Rear right wheel, Y_{RR}	`Translation(5,2)`
Rear right wheel, Z_{RR}	`Translation(5,3)`

`Rotation` — Actor rotation
`array`

Actor rotation across a [-pi/2, pi/2] range, in rad. Array dimensions are number of parts per actor-by-3.

- `Rotation(1,1)`, `Rotation(1,2)`, and `Rotation(1,3)` — Vehicle rotation about vehicle-fixed pitch, roll, and yaw Y-, Z-, and X- axes, respectively.
- `Rotation(...,1)`, `Rotation(...,2)`, and `Rotation(...,3)` — Actor rotation about vehicle-fixed pitch, roll, and yaw Y-, Z-, and X- axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The `Rotation` signal::

- Dimensions are [5x3].
- Contains rotation information according to the axle and wheel locations.

$$Rotation = \begin{bmatrix} Pitch_v & Roll_v & Yaw_v \\ Pitch_{FL} & Roll_{FL} & Yaw_{FL} \\ Pitch_{FR} & Roll_{FR} & Yaw_{FR} \\ Pitch_{RL} & Roll_{RL} & Yaw_{RL} \\ Pitch_{RR} & Roll_{RR} & Yaw_{RR} \end{bmatrix}$$

Rotation	Array Element
Vehicle, $Pitch_v$	`Rotation(1,1)`
Vehicle, $Roll_v$	`Rotation(1,2)`
Vehicle, Yaw_v	`Rotation(1,3)`
Front left wheel, $Pitch_{FL}$	`Rotation(2,1)`
Front left wheel, $Roll_{FL}$	`Rotation(2,2)`
Front left wheel, Yaw_{FL}	`Rotation(2,3)`
Front right wheel, $Pitch_{FR}$	`Rotation(3,1)`
Front right wheel, $Roll_{FR}$	`Rotation(3,2)`
Front right wheel, Yaw_{FR}	`Rotation(3,3)`
Rear left wheel, $Pitch_{RL}$	`Rotation(4,1)`
Rear left wheel, $Roll_{RL}$	`Rotation(4,2)`
Rear left wheel, Yaw_{RL}	`Rotation(4,3)`
Rear right wheel, $Pitch_{RR}$	`Rotation(5,1)`
Rear right wheel, $Roll_{RR}$	`Rotation(5,2)`
Rear right wheel, Yaw_{RR}	`Rotation(5,3)`

`Scale` — Actor scale
`array`

Actor scale. Array dimensions are number of number of parts per actor-by-3.

- $Scale(1,1)$, $Scale(1,2)$, and $Scale(1,3)$ — Vehicle scale along world X-, Y-, and Z- axes, respectively.
- $Scale(...,1)$, $Scale(...,2)$, and $Scale(...,3)$ — Actor scale along world X-, Y-, and Z- axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The $Scale$ signal:

- Dimensions are [5x3].
- Contains scale information according to the axle and wheel locations.

$$Scale = \begin{bmatrix} X_{V_{scale}} & Y_{V_{scale}} & Z_{V_{scale}} \\ X_{FL_{scale}} & Y_{FL_{scale}} & Z_{FL_{scale}} \\ X_{FR_{scale}} & Y_{FR_{scale}} & Z_{FR_{scale}} \\ X_{RL_{scale}} & Y_{RL_{scale}} & Z_{RL_{scale}} \\ X_{RR_{scale}} & Y_{RR_{scale}} & Z_{RR_{scale}} \end{bmatrix}$$

Scale	Array Element
Vehicle, $X_{v_{scale}}$	Scale(1,1)
Vehicle, $Y_{v_{scale}}$	Scale(1,2)
Vehicle, $Z_{v_{scale}}$	Scale(1,3)
Front left wheel, $X_{FL_{scale}}$	Scale(2,1)
Front left wheel, $Y_{FL_{scale}}$	Scale(2,2)
Front left wheel, $Z_{FL_{scale}}$	Scale(2,3)
Front right wheel, $X_{FR_{scale}}$	Scale(3,1)
Front right wheel, $Y_{FR_{scale}}$	Scale(3,2)
Front right wheel, $Z_{FR_{scale}}$	Scale(3,3)
Rear left wheel, $X_{RL_{scale}}$	Scale(4,1)
Rear left wheel, $Y_{RL_{scale}}$	Scale(4,2)
Rear left wheel, $Z_{RL_{scale}}$	Scale(4,3)
Rear right wheel, $X_{RR_{scale}}$	Scale(5,1)
Rear right wheel, $Y_{RR_{scale}}$	Scale(5,2)
Rear right wheel, $Z_{RR_{scale}}$	Scale(5,3)

Parameters

Tag for actor in 3D scene, ActorTag — Name
SimulinkActor1 (default) | character vector

Actor name.

Actors are scene objects that support 3D translation, rotation, and scale. Parts are actor components. Components do not exist by themselves; they are associated with an actor.

The block does not support multiple instances of the same actor tag. To refer to the same scene actor when you use the 3D block pairs (e.g. Simulation 3D Actor Transform Get and Simulation 3D Actor Transform Set), specify the same **Tag for actor in 3D scene, ActorTag** parameter.

Number of parts per actor to get, NumberOfParts — Name
1 (default) | scalar

Number of parts per actor. Actors are scene objects that support 3D translation, rotation, and scale. Parts are actor components. Components do not exist by themselves; they are associated with an actor. Typically, a vehicle actor with a body and four wheels has 5 parts.

The block does not support multiple instances of the same actor tag. To refer to the same scene actor when you use the 3D block pairs (e.g. Simulation 3D Actor Transform Get and Simulation 3D Actor Transform Set), specify the same **Tag for actor in 3D scene, ActorTag** parameter.

Sample time — Sample time
-1 (default) | scalar

Sample time, T_s. The graphics frame rate is the inverse of the sample time.

See Also

Simulation 3D Actor Transform Set | Simulation 3D Camera Get | Simulation 3D Scene Configuration | Vehicle Terrain Sensor

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"3D Visualization Engine Requirements"

Introduced in R2018a

Simulation 3D Actor Transform Set

Set actor translation, rotation, scale
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D / Sim3D Core
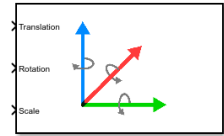
Description

The Simulation 3D Actor Transform Set block sets the actor translation, rotation, and scale in the 3D visualization environment.

The block uses a vehicle-fixed coordinate system that is initially aligned with the inertial world coordinate system.

Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about *X*-axis
Y	Extends to the right of the vehicle, initially parallel to the ground plane Pitch — Right-handed rotation about *Y*-axis
Z	Extends upwards Yaw — Left-handed rotation about *Z*-axis

Actors are scene objects that support 3D translation, rotation, and scale. Parts are actor components. Components do not exist by themselves; they are associated with an actor.

Tip Verify that the Simulation 3D Actor Transform Set block executes before the Simulation 3D Scene Configuration block. That way, Simulation 3D Actor Transform Set prepares the signal data before the Unreal Engine 3D visualization environment receives it. To check the block execution

order, right-click the blocks and select **Properties**. On the **General** tab, confirm these **Priority** settings:

- Simulation 3D Scene Configuration — `0`
- Simulation 3D Actor Transform Set — `-1`

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Ports

Input

`Translation` — Actor translation
`array`

Actor translation, in m. Array dimensions are number of parts per actor-by-3.

- `Translation(1,1)`, `Translation(1,2)`, and `Translation(1,3)` — Vehicle displacement along world *X*-, *Y*, and *Z*- axes, respectively.
- `Translation(...,1)`, `Translation(...,2)`, and `Translation(...,3)` — Actor displacement relative to vehicle, in vehicle-fixed coordinate system initially aligned with world *X*-, *Y*, and *Z*- axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The `Translation` signal:

- Dimensions are `[5x3]`.
- Contains translation information according to the axle and wheel locations, relative to vehicle.

$$Translation = \begin{bmatrix} X_v & Y_v & Z_v \\ X_{FL} & Y_{FL} & Z_{FL} \\ X_{FR} & Y_{FR} & Z_{FR} \\ X_{RL} & Y_{RL} & Z_{RL} \\ X_{RR} & Y_{RR} & Z_{RR} \end{bmatrix}$$

Translation	Array Element
Vehicle, X_v	`Translation(1,1)`
Vehicle, Y_v	`Translation(1,2)`
Vehicle, Z_v	`Translation(1,3)`
Front left wheel, X_{FL}	`Translation(2,1)`
Front left wheel, Y_{FL}	`Translation(2,2)`
Front left wheel, Z_{FL}	`Translation(2,3)`
Front right wheel, X_{FR}	`Translation(3,1)`
Front right wheel, Y_{FR}	`Translation(3,2)`
Front right wheel, Z_{FR}	`Translation(3,3)`
Rear left wheel, X_{RL}	`Translation(4,1)`
Rear left wheel, Y_{RL}	`Translation(4,2)`

Translation	Array Element
Rear left wheel, Z_{RL}	Translation(4,3)
Rear right wheel, X_{RR}	Translation(5,1)
Rear right wheel, Y_{RR}	Translation(5,2)
Rear right wheel, Z_{RR}	Translation(5,3)

Rotation — Actor rotation
array

Actor rotation across a [-pi/2, pi/2] range, in rad. Array dimensions are number of parts per actor-by-3.

- `Rotation(1,1)`, `Rotation(1,2)`, and `Rotation(1,3)` — Vehicle rotation about vehicle-fixed pitch, roll, and yaw Y-, Z-, and X- axes, respectively.

- `Rotation(...,1)`, `Rotation(...,2)`, and `Rotation(...,3)` — Actor rotation about vehicle-fixed pitch, roll, and yaw Y-, Z-, and X- axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The `Rotation` signal::

- Dimensions are [5x3].

- Contains rotation information according to the axle and wheel locations.

$$Rotation = \begin{bmatrix} Pitch_v & Roll_v & Yaw_v \\ Pitch_{FL} & Roll_{FL} & Yaw_{FL} \\ Pitch_{FR} & Roll_{FR} & Yaw_{FR} \\ Pitch_{RL} & Roll_{RL} & Yaw_{RL} \\ Pitch_{RR} & Roll_{RR} & Yaw_{RR} \end{bmatrix}$$

Rotation	Array Element
Vehicle, $Pitch_v$	Rotation(1,1)
Vehicle, $Roll_v$	Rotation(1,2)
Vehicle, Yaw_v	Rotation(1,3)
Front left wheel, $Pitch_{FL}$	Rotation(2,1)
Front left wheel, $Roll_{FL}$	Rotation(2,2)
Front left wheel, Yaw_{FL}	Rotation(2,3)
Front right wheel, $Pitch_{FR}$	Rotation(3,1)
Front right wheel, $Roll_{FR}$	Rotation(3,2)
Front right wheel, Yaw_{FR}	Rotation(3,3)
Rear left wheel, $Pitch_{RL}$	Rotation(4,1)
Rear left wheel, $Roll_{RL}$	Rotation(4,2)
Rear left wheel, Yaw_{RL}	Rotation(4,3)
Rear right wheel, $Pitch_{RR}$	Rotation(5,1)
Rear right wheel, $Roll_{RR}$	Rotation(5,2)

Rotation	Array Element
Rear right wheel, Yaw_{RR}	Rotation(5,3)

Scale — Actor scale
array

Actor scale. Array dimensions are number of number of parts per actor-by-**3**.

- Scale(1,1), Scale(1,2), and Scale(1,3) — Vehicle scale along world *X*-, *Y*-, and *Z*- axes, respectively.
- Scale(...,1), Scale(...,2), and Scale(...,3) — Actor scale along world *X*-, *Y*-, and *Z*-axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The Scale signal:

- Dimensions are [5x3].
- Contains scale information according to the axle and wheel locations.

$$Scale = \begin{bmatrix} X_{V_{scale}} & Y_{V_{scale}} & Z_{V_{scale}} \\ X_{FL_{scale}} & Y_{FL_{scale}} & Z_{FL_{scale}} \\ X_{FR_{scale}} & Y_{FR_{scale}} & Z_{FR_{scale}} \\ X_{RL_{scale}} & Y_{RL_{scale}} & Z_{RL_{scale}} \\ X_{RR_{scale}} & Y_{RR_{scale}} & Z_{RR_{scale}} \end{bmatrix}$$

Scale	Array Element
Vehicle, $X_{v_{scale}}$	Scale(1,1)
Vehicle, $Y_{v_{scale}}$	Scale(1,2)
Vehicle, $Z_{v_{scale}}$	Scale(1,3)
Front left wheel, $X_{FL_{scale}}$	Scale(2,1)
Front left wheel, $Y_{FL_{scale}}$	Scale(2,2)
Front left wheel, $Z_{FL_{scale}}$	Scale(2,3)
Front right wheel, $X_{FR_{scale}}$	Scale(3,1)
Front right wheel, $Y_{FR_{scale}}$	Scale(3,2)
Front right wheel, $Z_{FR_{scale}}$	Scale(3,3)
Rear left wheel, $X_{RL_{scale}}$	Scale(4,1)
Rear left wheel, $Y_{RL_{scale}}$	Scale(4,2)
Rear left wheel, $Z_{RL_{scale}}$	Scale(4,3)
Rear right wheel, $X_{RR_{scale}}$	Scale(5,1)
Rear right wheel, $Y_{RR_{scale}}$	Scale(5,2)
Rear right wheel, $Z_{RR_{scale}}$	Scale(5,3)

Parameters

Actor Setup

Tag for actor in 3D scene, ActorTag — Name
SimulinkActor1 (default) | character vector

Actor name.

Actors are scene objects that support 3D translation, rotation, and scale. Parts are actor components. Components do not exist by themselves; they are associated with an actor.

The block does not support multiple instances of the same actor tag. To refer to the same scene actor when you use the 3D block pairs (e.g. Simulation 3D Actor Transform Get and Simulation 3D Actor Transform Set), specify the same **Tag for actor in 3D scene, ActorTag** parameter.

Number of parts per actor to set, NumberOfParts — Name
1 (default) | scalar

Number of parts per actor. Actors are scene objects that support 3D translation, rotation, and scale. Parts are actor components. Components do not exist by themselves; they are associated with an actor. Typically, a vehicle actor with a body and four wheels has 5 parts.

The block does not support multiple instances of the same actor tag. To refer to the same scene actor when you use the 3D block pairs (e.g. Simulation 3D Actor Transform Get and Simulation 3D Actor Transform Set), specify the same **Tag for actor in 3D scene, ActorTag** parameter.

Initial Values

Initial array values to translate actor per part, Translation — Actor initial position
[0 0 0] (default) | array

Actor initial position, along world *X*-, *Y*-, and *Z*- axes, in m.

Array dimensions are number of parts per actor-by-3.

- Translation(1,1), Translation(1,2), and Translation(1,3) — Vehicle displacement along world *X*-, *Y*, and *Z*- axes, respectively.
- Translation(...,1), Translation(...,2), and Translation(...,3) — Actor displacement relative to vehicle, in vehicle-fixed coordinate system initially aligned with world *X*-, *Y*, and *Z*- axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The parameter:

- Dimensions are [5x3].
- Contains translation information according to the axle and wheel locations, relative to vehicle.

$$
Translation = \begin{bmatrix} X_v & Y_v & Z_v \\ X_{FL} & Y_{FL} & Z_{FL} \\ X_{FR} & Y_{FR} & Z_{FR} \\ X_{RL} & Y_{RL} & Z_{RL} \\ X_{RR} & Y_{RR} & Z_{RR} \end{bmatrix}
$$

Translation	Array Element
Vehicle, X_v	Translation(1,1)
Vehicle, Y_v	Translation(1,2)
Vehicle, Z_v	Translation(1,3)
Front left wheel, X_{FL}	Translation(2,1)
Front left wheel, Y_{FL}	Translation(2,2)
Front left wheel, Z_{FL}	Translation(2,3)
Front right wheel, X_{FR}	Translation(3,1)
Front right wheel, Y_{FR}	Translation(3,2)
Front right wheel, Z_{FR}	Translation(3,3)
Rear left wheel, X_{RL}	Translation(4,1)
Rear left wheel, Y_{RL}	Translation(4,2)
Rear left wheel, Z_{RL}	Translation(4,3)
Rear right wheel, X_{RR}	Translation(5,1)
Rear right wheel, Y_{RR}	Translation(5,2)
Rear right wheel, Z_{RR}	Translation(5,3)

Initial array values to rotate actor per part, Rotation — Actor initial rotation
[0 0 0] (default) | array

Actor initial rotation about world *X*-, *Y*-, and *Z*- axes across a [-pi/2, pi/2] range, in rad.

Array dimensions are number of parts per actor-by-3.

- Rotation(1,1), Rotation(1,2), and Rotation(1,3) — Vehicle rotation about vehicle-fixed pitch, roll, and yaw *Y*-, *Z*-, and *X*- axes, respectively.
- Rotation(...,1), Rotation(...,2), and Rotation(...,3) — Actor rotation about vehicle-fixed pitch, roll, and yaw *Y*-, *Z*-, and *X*- axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The parameter:

- Dimensions are [5x3].
- Contains rotation information according to the axle and wheel locations.

$$
Rotation = \begin{bmatrix} Pitch_v & Roll_v & Yaw_v \\ Pitch_{FL} & Roll_{FL} & Yaw_{FL} \\ Pitch_{FR} & Roll_{FR} & Yaw_{FR} \\ Pitch_{RL} & Roll_{RL} & Yaw_{RL} \\ Pitch_{RR} & Roll_{RR} & Yaw_{RR} \end{bmatrix}
$$

Rotation	Array Element
Vehicle, $Pitch_v$	Rotation(1,1)
Vehicle, $Roll_v$	Rotation(1,2)
Vehicle, Yaw_v	Rotation(1,3)

Rotation	Array Element
Front left wheel, $Pitch_{FL}$	Rotation(2,1)
Front left wheel, $Roll_{FL}$	Rotation(2,2)
Front left wheel, Yaw_{FL}	Rotation(2,3)
Front right wheel, $Pitch_{FR}$	Rotation(3,1)
Front right wheel, $Roll_{FR}$	Rotation(3,2)
Front right wheel, Yaw_{FR}	Rotation(3,3)
Rear left wheel, $Pitch_{RL}$	Rotation(4,1)
Rear left wheel, $Roll_{RL}$	Rotation(4,2)
Rear left wheel, Yaw_{RL}	Rotation(4,3)
Rear right wheel, $Pitch_{RR}$	Rotation(5,1)
Rear right wheel, $Roll_{RR}$	Rotation(5,2)
Rear right wheel, Yaw_{RR}	Rotation(5,3)

Initial array values to scale actor per part, Scale — Actor initial scale
[1 1 1] (default) | array

Actor initial scale.

Array dimensions are number of number of parts per actor-by-3.

- Scale(1,1), Scale(1,2), and Scale(1,3) — Vehicle scale along world *X*-, *Y*, and *Z*- axes, respectively.

- Scale(...,1), Scale(...,2), and Scale(...,3) — Actor scale along world *X*-, *Y*, and *Z*- axes, respectively.

For example, consider a vehicle actor with a vehicle body and four wheels. The parameter:

- Dimensions are [5x3].

- Contains scale information according to the axle and wheel locations.

$$Scale = \begin{bmatrix} X_{V_{scale}} & Y_{V_{scale}} & Z_{V_{scale}} \\ X_{FL_{scale}} & Y_{FL_{scale}} & Z_{FL_{scale}} \\ X_{FR_{scale}} & Y_{FR_{scale}} & Z_{FR_{scale}} \\ X_{RL_{scale}} & Y_{RL_{scale}} & Z_{RL_{scale}} \\ X_{RR_{scale}} & Y_{RR_{scale}} & Z_{RR_{scale}} \end{bmatrix}$$

Scale	Array Element	Scale Axis
Vehicle, $X_{v_{scale}}$	Scale(1,1)	World *X*-axis
Vehicle, $Y_{v_{scale}}$	Scale(1,2)	World *Y*-axis
Vehicle, $Z_{v_{scale}}$	Scale(1,3)	World *Z*-axis
Front left wheel, $X_{FL_{scale}}$	Scale(2,1)	World *X*-axis
Front left wheel, $Y_{FL_{scale}}$	Scale(2,2)	World *Y*-axis

Scale	Array Element	Scale Axis
Front left wheel, $Z_{FL_{scale}}$	Scale(2,3)	World Z-axis
Front right wheel, $X_{FR_{scale}}$	Scale(3,1)	World X-axis
Front right wheel, $Y_{FR_{scale}}$	Scale(3,2)	World Y-axis
Front right wheel, $Z_{FR_{scale}}$	Scale(3,3)	World Z-axis
Rear left wheel, $X_{RL_{scale}}$	Scale(4,1)	World X-axis
Rear left wheel, $Y_{RL_{scale}}$	Scale(4,2)	World Y-axis
Rear left wheel, $Z_{RL_{scale}}$	Scale(4,3)	World Z-axis
Rear right wheel, $X_{RR_{scale}}$	Scale(5,1)	World X-axis
Rear right wheel, $Y_{RR_{scale}}$	Scale(5,2)	World Y-axis
Rear right wheel, $Z_{RR_{scale}}$	Scale(5,3)	World Z-axis

Sample time — Sample time
-1 (default) | scalar

Sample time, T_s. The graphics frame rate is the inverse of the sample time.

See Also
Simulation 3D Actor Transform Get | Simulation 3D Camera Get | Simulation 3D Scene Configuration | Vehicle Terrain Sensor

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"3D Visualization Engine Requirements"

Introduced in R2018a

Simulation 3D Camera Get

Camera image
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D /
Sim3D Core
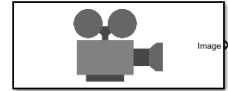
Description

The Simulation 3D Camera Get block provides an interface to an ideal camera in the 3D visualization environment. The image output is a red, green, and blue (RGB) array.

If you set the sample time to -1, the block uses the sample time specified in the Simulation 3D Scene Configuration block. To use this sensor, ensure that the Simulation 3D Scene Configuration block is in your model.

Tip Verify that the Simulation 3D Scene Configuration block executes before the Simulation 3D Camera Get block. That way, the Unreal Engine 3D visualization environment prepares the data before the Simulation 3D Camera Get block receives it. To check the block execution order, right-click the blocks and select **Properties**. On the **General** tab, confirm these **Priority** settings:

*   Simulation 3D Scene Configuration — 0
*   Simulation 3D Camera Get — 1

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Ports

Output

Image — 3D output camera image
m-by-*n*-by-3 array of RGB triplet values

3D output camera image, returned as an *m*-by-*n*-by-3 array of RGB triplet values. *m* is the vertical resolution of the image, and *n* is the horizontal resolution of the image.

Data Types: int8 | uint8

Parameters

Mounting

Sensor identifier — Number to identify unique sensor
0 (default) | positive integer

Unique sensor identifier, specified as a positive integer. This number is used to identify a specific sensor. The sensor identifier distinguishes between sensors in a multi-sensor system.

Example: 2

Vehicle name — Name of a vehicle
Scene Origin (default) | character vector

Vehicle name. Block provides a list of vehicles in the model. If you select Scene Origin, the block places a sensor at the scene origin.
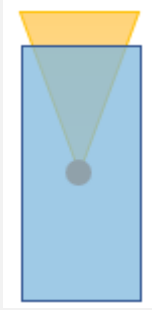
Example: SimulinkVehicle1
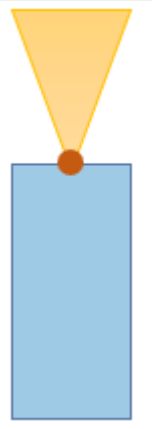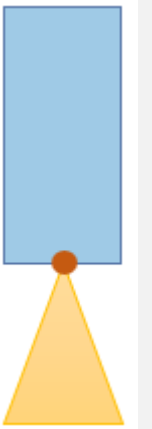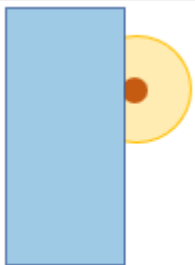
Vehicle mounting location — Sensor mounting location
Origin (default) | Front bumper | Rear bumper | Right mirror | Left mirror | Rearview mirror | Hood center | Roof center
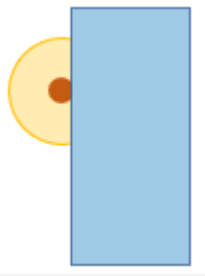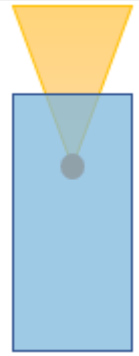
Sensor mounting location.

- When **Vehicle name** is Scene Origin, the block mounts the sensor to the origin of the scene, and **Mounting location** can be set to Origin only. During simulation, the sensor remains stationary.

- When **Vehicle name** is the name of a vehicle (for example, SimulinkVehicle1) the block mounts the sensor to one of the predefined mounting locations described in the table. During simulation, the sensor travels with the vehicle.

Vehicle Mounting Location	Description	Orientation Relative to Vehicle Origin [Roll, Pitch, Yaw] (deg)
Origin	Forward-facing sensor mounted to the vehicle origin, which is on the ground, at the geometric center of the vehicle	[0, 0, 0]

Vehicle Mounting Location	Description	Orientation Relative to Vehicle Origin [Roll, Pitch, Yaw] (deg)
Front bumper	Forward-facing sensor mounted to the front bumper 	[0, 0, 0]
Rear bumper	Backward-facing sensor mounted to the rear bumper 	[0, 0, 180]
Right mirror	Downward-facing sensor mounted to the right side-view mirror 	[0, –90, 0]

Vehicle Mounting Location	Description	Orientation Relative to Vehicle Origin [Roll, Pitch, Yaw] (deg)
Left mirror	Downward-facing sensor mounted to the left side-view mirror	[0, –90, 0]
Rearview mirror	Forward-facing sensor mounted to the rearview mirror, inside the vehicle	[0, 0, 0]
Hood center	Forward-facing sensor mounted to the center of the hood	[0, 0, 0]

Vehicle Mounting Location	Description	Orientation Relative to Vehicle Origin [Roll, Pitch, Yaw] (deg)
Roof center	Forward-facing sensor mounted to the center of the roof	[0, 0, 0]

The (*X*, *Y*, *Z*) location of the sensor relative to the vehicle depends on the vehicle type. To specify the vehicle type, use the **Type** parameter of the Simulation 3D Scene Configuration block to which you are mounting. The tables show the *X*, *Y*, and *Z* locations of sensors in the vehicle coordinate system. In this coordinate system:

- The *X*-axis points forward from the vehicle.
- The *Y*-axis points to the left of the vehicle, as viewed when facing forward.
- The *Z*-axis points up from the ground.
- Roll, pitch, and yaw are clockwise-positive when looking in the positive direction of the *X*-axis, *Y*-axis, and *Z*-axis, respectively. When looking at a vehicle from the top down, then the yaw angle (that is, the orientation angle) is counterclockwise-positive, because you are looking in the negative direction of the axis.

Box Truck — Sensor Locations Relative to Vehicle Origin

Mounting Location	X (m)	Y (m)	Z (m)
Front bumper	5.10	0	0.60
Rear bumper	–5	0	0.60
Right mirror	2.90	1.60	2.10
Left mirror	2.90	–1.60	2.10
Rearview mirror	2.60	0.20	2.60
Hood center	3.80	0	2.10
Roof center	1.30	0	4.20

Hatchback — Sensor Locations Relative to Vehicle Origin

Mounting Location	X (m)	Y (m)	Z (m)
Front bumper	1.93	0	0.51
Rear bumper	–1.93	0	0.51
Right mirror	0.43	–0.84	1.01
Left mirror	0.43	0.84	1.01
Rearview mirror	0.32	0	1.27
Hood center	1.44	0	1.01
Roof center	0	0	1.57

Muscle Car — Sensor Locations Relative to Vehicle Origin

Mounting Location	X (m)	Y (m)	Z (m)
Front bumper	2.47	0	0.45
Rear bumper	–2.47	0	0.45
Right mirror	0.43	–1.08	1.01
Left mirror	0.43	1.08	1.01
Rearview mirror	0.32	0	1.20
Hood center	1.28	0	1.14
Roof center	–0.25	0	1.58

Sedan — Sensor Locations Relative to Vehicle Origin

Mounting Location	X (m)	Y (m)	Z (m)
Front bumper	2.42	0	0.51
Rear bumper	–2.42	0	0.51
Right mirror	0.59	–0.94	1.09
Left mirror	0.59	0.94	1.09
Rearview mirror	0.43	0	1.31
Hood center	1.46	0	1.11
Roof center	–0.45	0	1.69

Small Pickup Truck — Sensor Locations Relative to Vehicle Origin

Mounting Location	X (m)	Y (m)	Z (m)
Front bumper	3.07	0	0.51
Rear bumper	–3.07	0	0.51
Right mirror	1.10	–1.13	1.52
Left mirror	1.10	1.13	1.52
Rearview mirror	0.85	0	1.77
Hood center	2.22	0	1.59
Roof center	0	0	2.27

Sport Utility Vehicle — Sensor Locations Relative to Vehicle Origin

Mounting Location	X (m)	Y (m)	Z (m)
Front bumper	2.42	0	0.51
Rear bumper	–2.42	0	0.51
Right mirror	0.60	–1	1.35
Left mirror	0.60	1	1.35
Rearview mirror	0.39	0	1.55
Hood center	1.58	0	1.39
Roof center	-0.56	0	2

Example: `Origin`

`Specify offset` — Specify offset from mounting location
`off` (default) | `on`

Select this parameter to specify an offset from the mounting location.

`Relative translation [X, Y, Z]` — Translation offset from mounting location
`[0,0,0]` (default) | real-valued 1-by-3 vector

Specify a translation offset from the mount location, about the vehicle coordinate system *X*, *Y*, and *Z* axes. Units are in meters.

- The *X*-axis points forward from the vehicle.
- The *Y*-axis points to the left of the vehicle, as viewed when facing forward.
- The *Z*-axis points up.

Example: `[0,0,0.01]`

Dependencies

To enable this parameter, select **Specify offset**.

`Relative rotation [Roll, Pitch, Yaw]` — Rotational offset from mounting location
`[0,0,0]` (default) | real-valued 1-by-3 vector

Specify a rotational offset from the mounting location, about the vehicle coordinate system *X*, *Y*, and *Z* axes. Units are in degrees.

- Roll angle is the angle of rotation about the *X*-axis of the vehicle coordinate system. A positive roll angle corresponds to a clockwise rotation when looking in the positive direction of the *X*-axis.

- Pitch angle is the angle of rotation about the *Y*-axis of the vehicle coordinate system. A positive pitch angle corresponds to a clockwise rotation when looking in the positive direction of the *Y*-axis.

- Yaw angle is the angle of rotation about the *Z* of the vehicle coordinate system. A positive yaw angle corresponds to a clockwise rotation when looking in the positive direction of the *Z*-axis.

Example: [0,0,10]

Dependencies

To enable this parameter, select **Specify offset**.

Sample time — Sample time
-1 (default) | positive scalar

Sample time of the block in seconds. The 3D simulation environment frame rate is the inverse of the sample time.

If you set the sample time to -1, the block uses the sample time specified in the Simulation 3D Scene Configuration block.

Parameter

Horizontal resolution — Pixels
uint32(1280) (default) | scalar

Horizontal image resolution, in pixels.

Vertical resolution — Pixels
uint32(720) (default) | scalar

Vertical image resolution, in pixels.

Horizontal field of view — Field of view
single(60) (default) | scalar

Horizontal field of view (FOV), in deg.

See Also
Simulation 3D Actor Transform Get | Simulation 3D Actor Transform Set | Simulation 3D Scene Configuration | Vehicle Terrain Sensor

Topics
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Scene Interrogation in 3D Environment"
"3D Visualization Engine Requirements"

Introduced in R2018a

Simulation 3D Scene Configuration

Scene configuration for 3D simulation environment

Library: Automated Driving Toolbox / Simulation 3D
Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D /
Sim3D Core
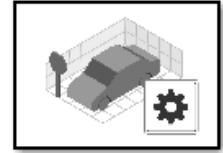
Description

The Simulation 3D Scene Configuration block implements a 3D simulation environment that is rendered by using the Unreal Engine from Epic Games®. Vehicle Dynamics Blockset integrates the 3D simulation environment with Simulink so that you can query the world around the vehicle and virtually test perception, control, and planning algorithms.

You can simulate from a set of prebuilt scenes or from your own custom scenes. Scene customization requires the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package. For more details, see "Support Package for Customizing Scenes".

Note The Simulation 3D Scene Configuration block must execute after blocks that send data to the 3D environment and before blocks that receive data from the 3D environment. To verify the execution order of such blocks, right-click the blocks and select **Properties**. Then, on the **General** tab, confirm these **Priority** settings:

- For blocks that send data to the 3D environment, such as Simulation 3D Vehicle with Ground Following blocks, **Priority** must be set to -1. That way, these blocks prepare their data before the 3D environment receives it.

- For the Simulation 3D Scene Configuration block in your model, **Priority** must be set to 0.

- For blocks that receive data from the 3D environment, such as Simulation 3D Message Get blocks, **Priority** must be set to 1. That way, the 3D environment can prepare the data before these blocks receive it.

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Parameters

Scene Selection

Scene source — Source of scene
Default Scenes (default) | Unreal Executable | Unreal Editor

Source of the scene in which to simulate, specified as one of the options in the table.

Option	Description
Default Scenes	Simulate in one of the default, prebuilt scenes specified in the **Scene name** parameter.

Option	Description
`Unreal Executable`	Simulate in a scene that is part of an Unreal Engine executable file. Specify the executable file in the **File name** parameter. Specify the scene in the **Scene** parameter. Select this option to simulate in custom scenes that have been packaged into an executable for faster simulation.
`Unreal Editor`	Simulate in a scene that is part of an Unreal Engine project (`.uproject`) file and is open in the Unreal® Editor. Specify the project file in the **Project** parameter. Select this option when developing custom scenes. By clicking **Open Unreal Editor**, you can co-simulate within Simulink and the Unreal Editor and modify your scenes based on the simulation results.

Scene name — Name of prebuilt 3D scene
`Straight road` (default) | `Curved road` | `Parking lot` | `Double lane change` | `Open surface` | `US city block` | `US highway` | `Virtual Mcity` | `Large parking lot`

Name of the prebuilt 3D scene in which to simulate, specified as one of these options. For details about a scene, see its listed corresponding reference page.

- `Straight road` — **Straight Road**
- `Curved road` — **Curved Road**
- `Parking lot` — **Parking Lot**
- `Double lane change` — **Double Lane Change**
- `Open surface` — **Open Surface**
- `US city block` — **US City Block**
- `US highway` — **US Highway**
- `Virtual Mcity` — **Virtual Mcity**
- `Large parking lot` — **Large Parking Lot**

The Vehicle Dynamics Blockset Support Package for Unreal Engine 4 Projects contains customizable versions of these scenes. For details about customizing scenes, see "Support Package for Customizing Scenes".

Dependencies

To enable this parameter, set **Scene source** to `Default Scenes`.

File name — Name of Unreal Engine executable file
`VehicleSimulation.exe` (default) | valid executable file name

Name of the Unreal Engine executable file, specified as a valid executable file name. You can either browse for the file or specify the full path to the file, using backslashes. To specify a scene from this file to simulate in, use the **Scene** parameter.

By default, **File name** is set to `VehicleSimulation.exe`, which is on the MATLAB search path.

Example: `C:\Local\WindowsNoEditor\AutoVrtlEnv.exe`

Dependencies

To enable this parameter, set **Scene source** to `Unreal Executable`.

Scene — Name of scene from executable file
`/Game/Maps/HwStrght` (default) | path to valid scene name

Name of a scene from the executable file specified by the **File name** parameter, specified as a path to a valid scene name.

When you package scenes from an Unreal Engine project into an executable file, the Unreal Editor saves the scenes to an internal folder within the executable file. This folder is located at the path `/Game/Maps`. Therefore, you must prepend `/Game/Maps` to the scene name. You must specify this path using forward slashes. For the file name, do not specify the `.umap` extension. For example, if the scene from the executable in which you want to simulate is named `myScene.umap`, specify **Scene** as `/Game/Maps/myScene`.

Alternatively, you can browse for the scene in the corresponding Unreal Engine project. These scenes are typically saved to the `Content/Maps` subfolder of the project. This subfolder contains all the scenes in your project. The scenes have the extension `.umap`. Select one of the scenes that you packaged into the executable file specified by the **File name** parameter. Use backward slashes and specify the `.umap` extension for the scene.

By default, **Scene** is set to `/Game/Maps/HwStrght`, which is a scene from the default `VehicleSimulation.exe` executable file specified by the **File name** parameter. This scene corresponds to the prebuilt **Straight Road** scene.

Example: `/Game/Maps/scene1`

Example: `C:\Local\myProject\Content\Maps\scene1.umap`

Dependencies

To enable this parameter, set **Scene source** to `Unreal Executable`.

Project — Name of Unreal Engine project file
valid project file name

Name of the Unreal Engine project file, specified as a valid project file name. You can either browse for the file or specify the full path to the file, using backslashes. The file must contain no spaces. To simulate scenes from this project in the Unreal Editor, click **Open Unreal Editor**. If you have an Unreal Editor session open already, then this button is disabled.

To run the simulation, in Simulink, click **Run**. Before you click **Play** in the Unreal Editor, wait until the Diagnostic Viewer window displays this confirmation message:

```
In the Simulation 3D Scene Configuration block, you set the scene source to 'Unreal Editor'.
In Unreal Editor, select 'Play' to view the scene.
```

This message confirms that Simulink has instantiated the scene actors, including the vehicles and cameras, in the Unreal Engine 3D environment. If you click **Play** before the Diagnostic Viewer window displays this confirmation message, Simulink might not instantiate the actors in the Unreal Editor.

Dependencies

To enable this parameter, set **Scene source** to `Unreal Editor`.

Scene Parameters

`Scene view` — Configure placement of virtual camera that displays scene
`Scene Origin` (default) | vehicle name

Configure the placement of the virtual camera that displays the scene during simulation.

- If your model contains no Simulation 3D Vehicle or Simulation 3D Vehicle with Ground Following blocks, then during simulation, you view the scene from a camera positioned at the scene origin.
- If your model contains at least one vehicle block, then by default, you view the scene from behind the first vehicle that was placed in your model. To change the view to a different vehicle, set **Scene view** to the name of that vehicle. The **Scene view** parameter list is populated with all the **Name** parameter values of the vehicle blocks contained in your model.

If you add a Simulation 3D Scene Configuration block to your model before adding any vehicle blocks, the virtual camera remains positioned at the scene. To reposition the camera to follow a vehicle, update this parameter.

When **Scene view** is set to a vehicle name, during simulation, you can change the location of the camera around the vehicle.

To change the camera views, use these key commands.

Key	Camera View	
1	Back left	
2	Back	
3	Back right	
4	Left	
5	Internal	
6	Right	
7	Front left	
8	Front	
9	Front right	
0	Overhead	

`Sample time` — Sample time of visualization engine
`.02` (default) | scalar greater than or equal to 0.01

Sample time, T_s, of the visualization engine, specified as a scalar greater than or equal to 0.01. Units are in seconds.

The graphics frame rate of the visualization engine is the inverse of the sample time. For example, if **Sample time** is `1/60`, then the visualization engine solver tries to achieve a frame rate of 60 frames per second. However, the real-time graphics frame rate is often lower due to factors such as graphics card performance and model complexity.

By default, blocks that receive data from the visualization engine, such as Simulation 3D Message blocks, inherit this sample rate.

Display 3D simulation window — Unreal Engine visualization
`on` (default) | `off`

Select whether to run simulations in the 3D visualization environment without visualizing the results, that is, in headless mode.

Consider running in headless mode in these cases:

- You want to run multiple 3D simulations in parallel to test models in different Unreal Engine scenarios.
- You want to optimize model parameters without visualizing the results. For example, consider using headless mode if you want to tune vehicle suspension parameters over a terrain scenario defined in Unreal Engine.

Dependencies

To enable this parameter, set **Scene source** to `Default Scenes` or `Unreal Executable`.

See Also
Simulation 3D Vehicle | Simulation 3D Vehicle with Ground Following

Topics
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Scene Interrogation in 3D Environment"
"3D Visualization Engine Requirements"
"Support Package for Customizing Scenes"

Introduced in R2018a

Vehicle Terrain Sensor

Vehicle and tire distances to objects
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D /
Sim3D Vehicle / Components
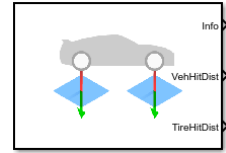
Description

The Vehicle Terrain Sensor block implements ray tracing to detect the terrain below the tires and objects in front of the vehicle. Specifically, for these actor components, the block returns the hit location (in the world coordinate system) and the distance to an object.

- Vehicle body
- Left front wheel
- Right front wheel
- Left rear wheel
- Right rear wheel

Tip Verify that the Vehicle Terrain Sensor block executes before the Simulation 3D Fisheye Camera block. That way, the Unreal Engine 3D visualization environment prepares the data before the Vehicle Terrain Sensor block receives it. To check the block execution order, right-click the blocks and select **Properties**. On the **General** tab, confirm these **Priority** settings:

- Simulation 3D Scene Configuration — `0`
- Vehicle Terrain Sensor — `1`

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Actors are scene objects that support 3D translation, rotation, and scale. Parts are actor components. Components do not exist by themselves; they are associated with an actor.

Hit Distance

To calculate the hit distances shown in the illustration, the block implements these equations.

Front of vehicle to object, *DistToHitVhAdjust*	`DistToHitVh = GetLength(CntrLocVh,HitLocVh)`
	`DistToHitVhAdjust = DistToHitVh - VehCntrLngthVal`
	
	`EndLocVh = CntrLocVh + VehRayLngth - VehRayOffset`
	`VehRayOffset = CntrLocVh - StartLocVh`
	`VehRayLngth = StartLocVh - EndLocVh`

Tires to terrain, *DistToHitTrAdjust*	`DistToHitTr = GetLength(CntrLocTr, HitLocTr)`
	`DistToHitTrAdjust = DistToHitTr - TireRadiiVal`
	
	`EndLocTr = CntrLocTr + LengthTr - OffsetTr`
	`OffsetTr = CntrLocTr - StartLocTr`
	`LengthTr = StartLocTr - EndLocTr`
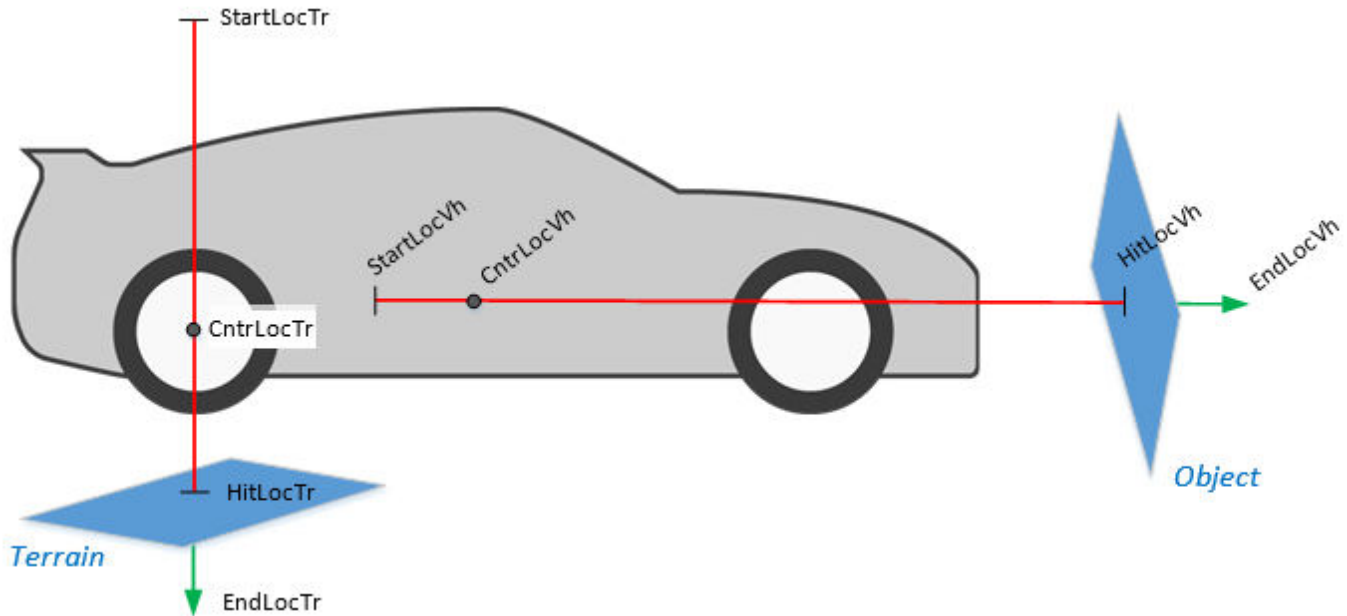
This illustration and equations use these variables.

CntrLocVh	Vehicle center location
DistToHitVh	Distance from vehicle center location to object
DistToHitVhAdjust	Distance from the front of the vehicle to object
EndLocVh	Vehicle ray trace end
HitLocVh	Vehicle hit location
OffsetVh	Vehicle trace offset
StartLocVh	Vehicle ray trace start
VehRayLngth	Vehicle trace length
VehCntrLngthVal	Distance from vehicle center to front
CntrLocTr	Tire center location
DistToHitTr	Distance from tire center location to terrain
DistToHitTrAdjust	Distance from tire to terrain
HitLocTr	Tire hit location
EndLocTr	Tire ray trace end
OffsetTr	Tire trace offset
StartLocTr	Tire ray trace start

LengthTr	Tire trace length
TireRadiiVal	Tire radius

Hit Event

To determine a hit event, the block uses the ray trace. The block provides the hit location in the world coordinate system.
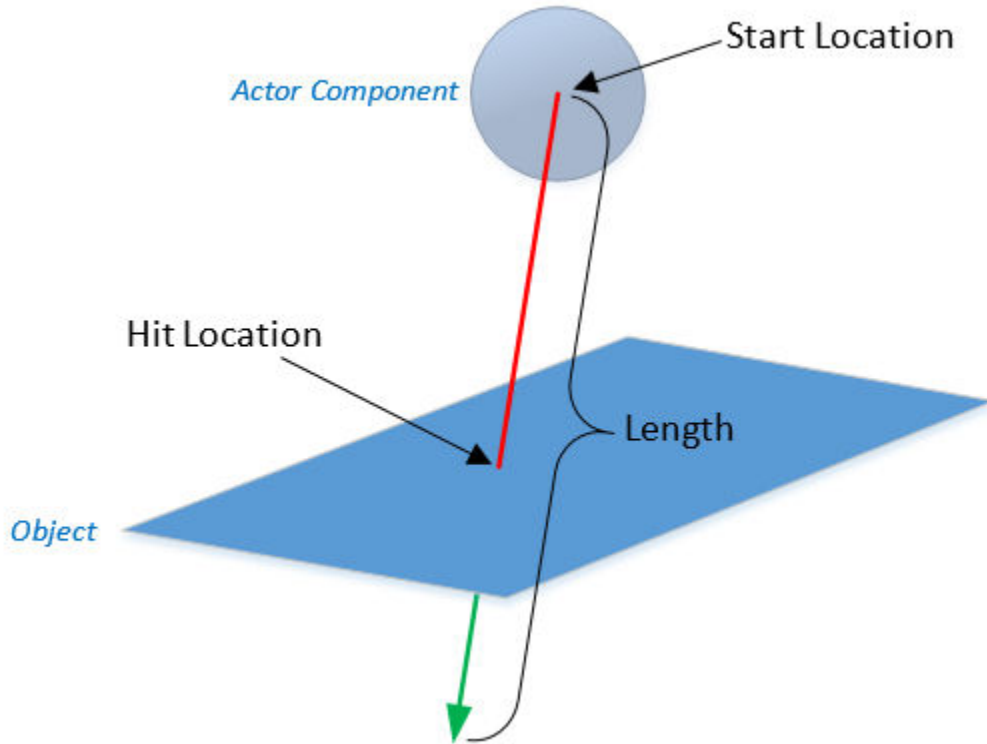
Miss Event

To determine a miss event, the block uses the ray trace.
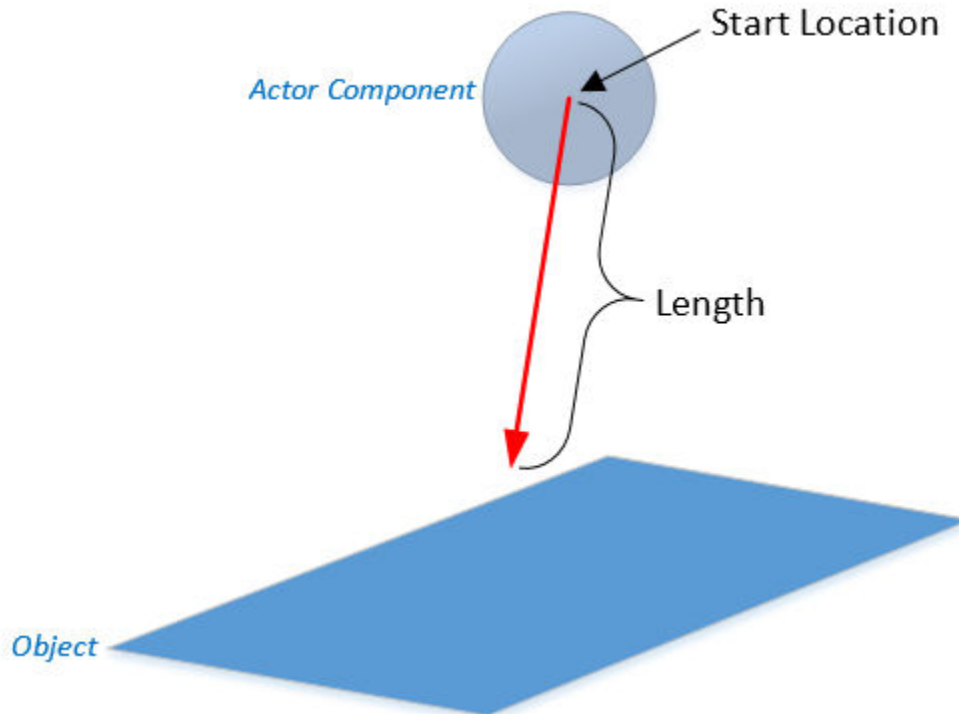
Ports

Input

VehCntr — Vehicle distance from center to front
`scalar`

Distance from the vehicle center to front, *VehCntrLngthVal*, in m.

Dependencies

Distance to vehicle center	Creates Port	Creates Parameter
Constant	None	**Distance from vehicle center to front, VehCntrLngthVal**
External input	VehCntr	*None*

TireRadii — Tire radii
`array`

Tire radii, *TireRadiiVal*, in m.

Dependencies

Distance to tire center Setting	Creates Port	Creates Parameter
Constant	None	**Distance from tire center to ground, TireRadiiVal**
External input	TireRadii	None

Output

Info — Bus signal
bus

Bus signal containing block values. The signals are arrays that depend on the wheel location.

Signal	Description	Variable	Units
HitFlg	Vehicle and wheel hit flag: • Hit an object – 1 • Miss an object – 0	$\begin{bmatrix} Vehicle \\ FrontLeft \\ FrontRight \\ RearLeft \\ RearRight \end{bmatrix}$	NA
HitLoc	Vehicle, *HitLocVh*, and tire, *HitLocTr*, hit locations, in world coordinate system *X*-, *Y*, and *Z*- axes, respectively	$\begin{bmatrix} Vehicle_X & Vehicle_Y & Vehicle_Z \\ FrontLeft_X & FrontLeft_Y & FrontLeft_Z \\ FrontRight_X & FrontRight_Y & FrontRight_Z \\ RearLeft_X & RearLeft_Y & RearLeft_Z \\ RearRear_X & RearRear_Y & RearRear_Z \end{bmatrix}$	m
StartLoc	Vehicle, *StartLocVh*, and tire, *StartLocTr*, ray trace start locations, in world coordinate system *X*-, *Y*, and *Z*- axes, respectively		m

VehHitDist — Front of vehicle distance to object
scalar

Distance from the front of the vehicle to object, *DistToHitVhAdjust*, in m.

TireHitDist — Tire distance to terrain
vector

Distance from tire to terrain, *DistToHitTrAdjust*, in m.

DistToHitTrAdjust = [*FrontLeft FrontRight RearLef RearRight*]

Parameters

Actor Setup

Tag for actor in 3D scene, ActorTag — Name
SimulinVehicle1 (default) | character vector

Actor name.

Actors are scene objects that support 3D translation, rotation, and scale. Parts are actor components. Components do not exist by themselves; they are associated with an actor.

The block does not support multiple instances of the same actor tag. To refer to the same scene actor when you use the 3D block pairs (e.g. Simulation 3D Actor Transform Get and Simulation 3D Actor Transform Set), specify the same **Tag for actor in 3D scene, ActorTag** parameter.

Distance to vehicle center — Selection
Constant (default) | External input

Configure how to provide the distance to the vehicle center.

Dependencies

Distance to vehicle center	Creates Port	Creates Parameter
Constant	None	**Distance from vehicle center to front, VehCntrLngthVal**
External input	VehCntr	*None*

Distance to tire center — Selection
Constant (default) | External input

Configure how to provide the distance to the tire center.

Dependencies

Distance to tire center Setting	Creates Port	Creates Parameter
Constant	None	**Distance from tire center to ground, TireRadiiVal**
External input	TireRadii	None

Distance from vehicle center to front, VehCntrLngthVal — Vehicle center
0 (default) | scalar

Distance from the vehicle center to front, *VehCntrLngthVal*, in m.

Dependencies

Distance to vehicle center	Creates Port	Creates Parameter
Constant	None	**Distance from vehicle center to front, VehCntrLngthVal**
External input	VehCntr	*None*

Distance from tire center to ground, TireRadiiVal — Tire radii
0 (default) | scalar

Tire radius, *TireRadiiVal*, in m.

Dependencies

Distance to tire center Setting	Creates Port	Creates Parameter
Constant	None	**Distance from tire center to ground, TireRadiiVal**
External input	TireRadii	None

Trace Lengths

Vehicle body x-axis trace length, VehRayLngth — Trace length
5 (default) | scalar

Vehicle body trace length, *VehRayLngth*, in m.

Left front wheel z-axis trace length, LfRayLngth — Trace length
5 (default) | scalar

Left front wheel trace length, *LfRayLngth* and *LengthTr*, in m.

Right front wheel z-axis trace length, RfRayLngth — Trace length
5 (default) | scalar

Right front wheel trace length, *RfRayLngth* and *LengthTr*, in m.

Left rear wheel z-axis trace length, LrRayLngth — Trace length
5 (default) | scalar

Left rear wheel trace length, *LrRayLngth* and *LengthTr*, in m.

Right rear wheel z-axis trace length, RrRayLngth — Trace length
5 (default) | scalar

Right rear wheel trace length, *RrRayLngth* and *LengthTr*, in m.

Starting Point Offsets

Vehicle body x-axis trace offset, VehRayOffset — Offset the vehicle ray trace
0 (default) | scalar

Vehicle body trace offset, *OffsetVh*, in m.

Left front wheel z-axis trace offset, LfRayOffset — Offset the left front wheel ray trace
0 (default) | scalar

Left front wheel trace offset, *LfRayOffset* and *OffsetTr*, in m.

Right front wheel z-axis trace offset, RfRayOffset — Offset the right front wheel ray trace
0 (default) | scalar

Right front wheel trace offset, *RfRayOffset* and *OffsetTr*, in m.

`Left rear wheel z-axis trace offset, LrRayOffset` — Offset the left rear wheel ray trace
0 (default) | `scalar`

Left rear wheel trace offset, *LrRayOffset* and *OffsetTr,* in m.

`Right rear wheel z-axis trace offset, RrRayOffset` — Offset the right rear wheel ray trace
0 (default) | `scalar`

Right rear wheel trace offset, *RrRayOffset* and *OffsetTr,* in m.

Enable Traces

`Vehicle body` — Enable vehicle body ray tracing
on (default) | off

Enable vehicle body ray tracing.

`Left front tire` — Enable left front tire ray tracing
on (default) | off

Enable left front tire ray tracing.

`Right front tire` — Enable right front tire ray tracing
on (default) | off

Enable right front tire ray tracing.

`Left rear tire` — Enable left rear tire ray tracing
on (default) | off

Enable left rear tire ray tracing.

`Right rear tire` — Enable right rear tire ray tracing
on (default) | off

Enable right rear tire ray tracing.

`Trace line visualization` — Visualize ray traces
on (default) | off

Enable trace line visualization.

`Sample time` — Sample time
-1 (default) | `scalar`

Sample time, T_s. The graphics frame rate is the inverse of the sample time.

See Also
Simulation 3D Vehicle | Simulation 3D Camera Get | Simulation 3D Scene Configuration | Simulation 3D Vehicle with Ground Following

Topics
"Scene Interrogation in 3D Environment"

External Websites
Unreal Engine

Introduced in R2018a

Simulation 3D Vehicle with Ground Following

Implement vehicle that follows ground in 3D environment
Library: Automated Driving Toolbox / Simulation 3D
       Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D /
       Sim3D Vehicle / Components
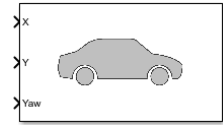
Description

The Simulation 3D Vehicle with Ground Following block implements a vehicle with four wheels in a 3D simulation environment. This environment is rendered using the Unreal Engine from Epic Games. The block uses the input (*X*, *Y*) position and yaw angle of the vehicle to adjust the elevation, roll angle, and pitch angle of the vehicle so that it follows the ground terrain. The block determines the vehicle velocity and heading and adjusts the steering angle and rotation for each wheel. Use this block for automated driving applications.

To use this block, ensure that the Simulation 3D Scene Configuration block is in your model. If you set the **Sample time** parameter of the Simulation 3D Vehicle with Ground Following block to `-1`, the block inherits the sample time specified in the Simulation 3D Scene Configuration block.

The block input uses the vehicle Z-down *right-handed* (RH) *Cartesian* coordinate system defined in SAE J670[1]. The coordinate system is inertial and initially aligned with the vehicle geometric center:

- *X*-axis — Along vehicle longitudinal axis, points forward
- *Y*-axis — Along vehicle lateral axis, points to the right
- *Z*-axis — Points downward

Note The Simulation 3D Vehicle with Ground Following block must execute before the Simulation 3D Scene Configuration block. That way, the Simulation 3D Vehicle with Ground Following block prepares the signal data before the Unreal Engine 3D visualization environment receives it. To check the block execution order, right-click the blocks and select **Properties**. On the **General** tab, confirm these **Priority** settings:

- Simulation 3D Scene Configuration — `0`
- Simulation 3D Vehicle with Ground Following — `-1`

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Ports

Input

X — Longitudinal position of vehicle
scalar

Longitudinal position of the vehicle along the *X*-axis of the scene. **X** is in the inertial *Z*-down coordinate system. Units are in meters.

Y — Lateral position of vehicle
scalar

Lateral position of the vehicle along the *Y*-axis of the scene. **Y** is in the inertial *Z*-down coordinate system. Units are in meters.

Yaw — Yaw orientation angle of vehicle
scalar

Yaw orientation angle of the vehicle along the *Z*-axis of the scene. **Yaw** is in the *Z*-down coordinate system. Units are in radians.

Parameters

Vehicle Parameters

Type — Type of vehicle
Muscle car (default) | Sedan | Sport utility vehicle | Small pickup truck | Hatchback | Box truck

Select the type of vehicle. To obtain the dimensions of each vehicle type, see these reference pages:

- Muscle car — **Muscle Car**
- Sedan — **Sedan**
- Sport utility vehicle — **Sport Utility Vehicle**
- Small pickup truck — **Small Pickup Truck**
- Hatchback — **Hatchback**
- Box truck — **Box Truck**

Color — Color of vehicle
Red (default) | Orange | Yellow | Green | Blue | Black | White | Silver

Select the color of the vehicle.

Initial position [X, Y, Z], InitialPos (m) — Initial vehicle position
[0, 0, 0] (default) | real-valued 1-by-3 vector

Initial vehicle position along the *X*-axis, *Y*-axis, and *Z*-axis in the inertial *Z*-down coordinate system, in m.

Initial rotation [Roll, Pitch, Yaw], InitialRot (rad) — Initial angle of vehicle rotation
[0, 0, 0] (default) | real-valued 1-by-3 vector

Initial angle of vehicle rotation, in rad. The angle of rotation is defined by the roll, pitch, and yaw of the vehicle.

Name, ActorName — Name of vehicle
SimulinkVehicle1 (default) | vehicle name

Name of vehicle. By default, when you use the block in your model, the block sets the **Name** parameter to SimulinkVehicle*X*. The value of *X* depends on the number of Simulation 3D Vehicle with Ground Following blocks that you have in your model.

Sample time, SampleTime — Sample time
-1 (default) | positive scalar

Sample time, T_s. The graphics frame rate is the inverse of the sample time.

If you set the sample time to -1, the block uses the sample time specified in the Simulation 3D Scene Configuration block.

References

[1] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology*. SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[2] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary*. ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

See Also

Simulation 3D Scene Configuration | Simulation 3D Vehicle

Topics
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Scene Interrogation in 3D Environment"
"3D Visualization Engine Requirements"

Introduced in R2019b

Simulation 3D Vehicle

Implement vehicle in 3D environment
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D /
                      Sim3D Vehicle / Components
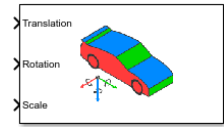
Description

The Simulation 3D Vehicle block implements a vehicle with four wheels in the 3D simulation environment.

To use this block, ensure that the Simulation 3D Scene Configuration block is in your model. If you set the **Sample time** parameter of this block to `-1`, the block uses the sample time specified in the Simulation 3D Scene Configuration block.

The block input uses the vehicle Z-down *right-handed* (RH) *Cartesian* coordinate system defined in SAE J670[1]. The coordinate system is inertial and initially aligned with the vehicle geometric center:

- *X*-axis — Along vehicle longitudinal axis, points forward
- *Y*-axis — Along vehicle lateral axis, points to the right
- *Z*-axis — Points downward

Tip Verify that the Simulation 3D Vehicle block executes before the Simulation 3D Scene Configuration block. That way, Simulation 3D Vehicle prepares the signal data before the Unreal Engine 3D visualization environment receives it. To check the block execution order, right-click the blocks and select **Properties**. On the **General** tab, confirm these **Priority** settings:

- Simulation 3D Scene Configuration — `0`
- Simulation 3D Vehicle — `-1`

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Ports

Input

`Translation` — Vehicle translation
5-by-3 array

Vehicle and wheel translation, in m. Array dimensions are 5-by-3.

- `Translation(1,1)`, `Translation(1,2)`, and `Translation(1,3)` — Vehicle translation along the inertial vehicle Z-down *X*-, *Y*-, and *Z*- axes, respectively.
- `Translation(...,1)`, `Translation(...,2)`, and `Translation(...,3)` — Wheel translation relative to vehicle, along the vehicle Z-down *X*-, *Y*-, and *Z*- axes, respectively.

The signal contains translation information according to the axle and wheel locations.

$$Translation = \begin{bmatrix} X_v & Y_v & Z_v \\ X_{FL} & Y_{FL} & Z_{FL} \\ X_{FR} & Y_{FR} & Z_{FR} \\ X_{RL} & Y_{RL} & Z_{RL} \\ X_{RR} & Y_{RR} & Z_{RR} \end{bmatrix}$$

Translation	Array Element	Translation Axis
Vehicle, X_v	Translation(1,1)	Inertial vehicle Z-down X-axis
Vehicle, Y_v	Translation(1,2)	Inertial vehicle Z-down Y-axis
Vehicle, Z_v	Translation(1,3)	Inertial vehicle Z-down Z-axis
Front left wheel, X_{FL}	Translation(2,1)	Vehicle Z-down X-axis
Front left wheel, Y_{FL}	Translation(2,2)	Vehicle Z-down Y-axis
Front left wheel, Z_{FL}	Translation(2,3)	Vehicle Z-down Z-axis
Front right wheel, X_{FR}	Translation(3,1)	Vehicle Z-down X-axis
Front right wheel, Y_{FR}	Translation(3,2)	Vehicle Z-down Y-axis
Front right wheel, Z_{FR}	Translation(3,3)	Vehicle Z-down Z-axis
Rear left wheel, X_{RL}	Translation(4,1)	Vehicle Z-down X-axis
Rear left wheel, Y_{RL}	Translation(4,2)	Vehicle Z-down Y-axis
Rear left wheel, Z_{RL}	Translation(4,3)	Vehicle Z-down Z-axis
Rear right wheel, X_{RR}	Translation(5,1)	Vehicle Z-down X-axis
Rear right wheel, Y_{RR}	Translation(5,2)	Vehicle Z-down Y-axis
Rear right wheel, Z_{RR}	Translation(5,3)	Vehicle Z-down Z-axis

`Rotation` — Vehicle rotation
5-by-3 array

Vehicle and wheel rotation, in rad. Array dimensions are 5-by-3.

- `Rotation(1,1)`, `Rotation(1,2)`, and `Rotation(1,3)` — Vehicle rotation about the inertial vehicle Z-down X-, Y-, and Z- axes, respectively.
- `Rotation(...,1)`, `Rotation(...,2)`, and `Rotation(...,3)` — Wheel rotation relative to vehicle, about the vehicle Z-down X-, Y-, and Z- axes, respectively.

The signal contains rotation information according to the axle and wheel locations.

$$Rotation = \begin{bmatrix} Roll_v & Pitch_v & Yaw_v \\ Roll_{FL} & Pitch_{FL} & Yaw_{FL} \\ Roll_{FR} & Pitch_{FR} & Yaw_{FR} \\ Roll_{RL} & Pitch_{RL} & Yaw_{RL} \\ Roll_{RR} & Pitch_{RR} & Yaw_{RR} \end{bmatrix}$$

Rotation	Array Element	Rotation Axis
Vehicle, $Roll_v$	Rotation(1,1)	Inertial vehicle Z-down X-axis

Rotation	Array Element	Rotation Axis
Vehicle, $Pitch_v$	Rotation(1,2)	Inertial vehicle Z-down *Y*-axis
Vehicle, Yaw_v	Rotation(1,3)	Inertial vehicle Z-down *Z*-axis
Front left wheel, $Roll_{FL}$	Rotation(2,1)	Vehicle Z-down *X*-axis
Front left wheel, $Pitch_{FL}$	Rotation(2,2)	Vehicle Z-down *Y*-axis
Front left wheel, Yaw_{FL}	Rotation(2,3)	Vehicle Z-down *Z*-axis
Front right wheel, $Roll_{FR}$	Rotation(3,1)	Vehicle Z-down *X*-axis
Front right wheel, $Pitch_{FR}$	Rotation(3,2)	Vehicle Z-down *Y*-axis
Front right wheel, Yaw_{FR}	Rotation(3,3)	Vehicle Z-down *Z*-axis
Rear left wheel, $Roll_{RL}$	Rotation(4,1)	Vehicle Z-down *X*-axis
Rear left wheel, $Pitch_{RL}$	Rotation(4,2)	Vehicle Z-down *Y*-axis
Rear left wheel, Yaw_{RL}	Rotation(4,3)	Vehicle Z-down *Z*-axis
Rear right wheel, $Roll_{RR}$	Rotation(5,1)	Vehicle Z-down *X*-axis
Rear right wheel, $Pitch_{RR}$	Rotation(5,2)	Vehicle Z-down *Y*-axis
Rear right wheel, Yaw_{RR}	Rotation(5,3)	Vehicle Z-down *Z*-axis

Scale — Vehicle scale
5-by-3

Vehicle and wheel scale, dimensionless. Array dimensions are 5-by-3.

- `Scale(1,1)`, `Scale(1,2)`, and `Scale(1,3)` — Vehicle scale along the inertial vehicle Z-down *X*-, *Y*-, and *Z*- axes, respectively.

- `Scale(...,1)`, `Scale(...,2)`, and `Scale(...,3)` — Wheel scale relative to vehicle, along vehicle Z-down *X*-, *Y*-, and *Z*- axes, respectively.

The signal contains scale information according to the axle and wheel locations.

$$Scale = \begin{bmatrix} X_{V_{scale}} & Y_{V_{scale}} & Z_{V_{scale}} \\ X_{FL_{scale}} & Y_{FL_{scale}} & Z_{FL_{scale}} \\ X_{FR_{scale}} & Y_{FR_{scale}} & Z_{FR_{scale}} \\ X_{RL_{scale}} & Y_{RL_{scale}} & Z_{RL_{scale}} \\ X_{RR_{scale}} & Y_{RR_{scale}} & Z_{RR_{scale}} \end{bmatrix}$$

Scale	Array Element	Scale Axis
Vehicle, $X_{v_{scale}}$	Scale(1,1)	Vehicle Z-down *X*-axis
Vehicle, $Y_{v_{scale}}$	Scale(1,2)	Vehicle Z-down *Y*-axis
Vehicle, $Z_{v_{scale}}$	Scale(1,3)	Vehicle Z-down *Z*-axis
Front left wheel, $X_{FL_{scale}}$	Scale(2,1)	Vehicle Z-down *X*-axis
Front left wheel, $Y_{FL_{scale}}$	Scale(2,2)	Vehicle Z-down *Y*-axis
Front left wheel, $Z_{FL_{scale}}$	Scale(2,3)	Vehicle Z-down *Z*-axis

Scale	Array Element	Scale Axis
Front right wheel, $X_{FR_{scale}}$	Scale(3,1)	Vehicle Z-down X-axis
Front right wheel, $Y_{FR_{scale}}$	Scale(3,2)	Vehicle Z-down Y-axis
Front right wheel, $Z_{FR_{scale}}$	Scale(3,3)	Vehicle Z-down Z-axis
Rear left wheel, $X_{RL_{scale}}$	Scale(4,1)	Vehicle Z-down X-axis
Rear left wheel, $Y_{RL_{scale}}$	Scale(4,2)	Vehicle Z-down Y-axis
Rear left wheel, $Z_{RL_{scale}}$	Scale(4,3)	Vehicle Z-down Z-axis
Rear right wheel, $X_{RR_{scale}}$	Scale(5,1)	Vehicle Z-down X-axis
Rear right wheel, $Y_{RR_{scale}}$	Scale(5,2)	Vehicle Z-down Y-axis
Rear right wheel, $Z_{RR_{scale}}$	Scale(5,3)	Vehicle Z-down Z-axis

Parameters

Vehicle Parameters

Type — Type
Muscle car (default) | Sedan | Sport utility vehicle | Small pickup truck | Hatchback | Box truck

If you set **Actor type** to Passenger vehicle, use the **Vehicle type** parameter to specify the vehicle. This table provides links to the vehicle dimensions.

Vehicle type Setting	Vehicle Dimensions
Muscle car	**Muscle Car**
Sedan	**Sedan**
Sport utility vehicle	**Sport Utility Vehicle**
Small pickup truck	**Small Pickup Truck**
Hatchback	**Hatchback**
Box truck	**Box Truck**

Color — Vehicle
Red (default) | Orange | Yellow | Green | Blue | Black | White | Silver

Specify the vehicle color.

Name — Name of vehicle
SimulinkVehicle1 (default) | character vector

Name of vehicle. By default, when you use the block in your model, the block sets the **Name** parameter to SimulinkVehicle*X*. The value of *X* depends on the number of Simulation 3D Vehicle with Ground Following and Simulation 3D Vehicle blocks that you have in your model.

Initial Values

Initial array values to translate vehicle per part, Translation — Vehicle initial translation
zeros(5, 3) (default) | 5-by-3 array

Initial vehicle and wheel translation, in m. Array dimensions are 5-by-3.

- `Translation(1,1)`, `Translation(1,2)`, and `Translation(1,3)` — Initial vehicle translation along the inertial vehicle Z-down coordinate system *X*-, *Y*-, and *Z*- axes, respectively.
- `Translation(...,1)`, `Translation(...,2)`, and `Translation(...,3)` — Initial wheel translation relative to vehicle, along the vehicle Z-down *X*-, *Y*-, and *Z*- axes, respectively.

The parameter contains translation information according to the axle and wheel locations.

$$
Translation = \begin{bmatrix} X_v & Y_v & Z_v \\ X_{FL} & Y_{FL} & Z_{FL} \\ X_{FR} & Y_{FR} & Z_{FR} \\ X_{RL} & Y_{RL} & Z_{RL} \\ X_{RR} & Y_{RR} & Z_{RR} \end{bmatrix}
$$

Translation	Array Element	Translation Axis
Vehicle, X_v	`Translation(1,1)`	Inertial vehicle Z-down *X*-axis
Vehicle, Y_v	`Translation(1,2)`	Inertial vehicle Z-down *Y*-axis
Vehicle, Z_v	`Translation(1,3)`	Inertial vehicle Z-down *Z*-axis
Front left wheel, X_{FL}	`Translation(2,1)`	Vehicle Z-down *X*-axis
Front left wheel, Y_{FL}	`Translation(2,2)`	Vehicle Z-down *Y*-axis
Front left wheel, Z_{FL}	`Translation(2,3)`	Vehicle Z-down *Z*-axis
Front right wheel, X_{FR}	`Translation(3,1)`	Vehicle Z-down *X*-axis
Front right wheel, Y_{FR}	`Translation(3,2)`	Vehicle Z-down *Y*-axis
Front right wheel, Z_{FR}	`Translation(3,3)`	Vehicle Z-down *Z*-axis
Rear left wheel, X_{RL}	`Translation(4,1)`	Vehicle Z-down *X*-axis
Rear left wheel, Y_{RL}	`Translation(4,2)`	Vehicle Z-down *Y*-axis
Rear left wheel, Z_{RL}	`Translation(4,3)`	Vehicle Z-down *Z*-axis
Rear right wheel, X_{RR}	`Translation(5,1)`	Vehicle Z-down *X*-axis
Rear right wheel, Y_{RR}	`Translation(5,2)`	Vehicle Z-down *Y*-axis
Rear right wheel, Z_{RR}	`Translation(5,3)`	Vehicle Z-down *Z*-axis

Initial array values to rotate vehicle per part, Rotation — Vehicle initial rotation
zeros(5, 3) (default) | 5-by-3 array

Initial vehicle and wheel rotation, about the vehicle Z-down *X*-, *Y*-, and *Z*- axes.

Array dimensions are 5-by-3.

- `Rotation(1,1)`, `Rotation(1,2)`, and `Rotation(1,3)` — Initial vehicle rotation about the inertial vehicle Z-down coordinate system*X*-, *Y*-, and *Z*- axes, respectively.
- `Rotation(...,1)`, `Rotation(...,2)`, and `Rotation(...,3)` — Initial wheel rotation relative to vehicle, about the vehicle Z-down *X*-, *Y*-, and *Z*- axes, respectively.

The parameter contains rotation information according to the axle and wheel locations.

$$Rotation = \begin{bmatrix} Roll_v & Pitch_v & Yaw_v \\ Roll_{FL} & Pitch_{FL} & Yaw_{FL} \\ Roll_{FR} & Pitch_{FR} & Yaw_{FR} \\ Roll_{RL} & Pitch_{RL} & Yaw_{RL} \\ Roll_{RR} & Pitch_{RR} & Yaw_{RR} \end{bmatrix}$$

Rotation	Array Element	Rotation Axis
Vehicle, $Roll_v$	Rotation(1,1)	Inertial vehicle Z-down X-axis
Vehicle, $Pitch_v$	Rotation(1,2)	Inertial vehicle Z-down Y-axis
Vehicle, Yaw_v	Rotation(1,3)	Inertial vehicle Z-down Z-axis
Front left wheel, $Roll_{FL}$	Rotation(2,1)	Vehicle Z-down X-axis
Front left wheel, $Pitch_{FL}$	Rotation(2,2)	Vehicle Z-down Y-axis
Front left wheel, Yaw_{FL}	Rotation(2,3)	Vehicle Z-down Z-axis
Front right wheel, $Roll_{FR}$	Rotation(3,1)	Vehicle Z-down X-axis
Front right wheel, $Pitch_{FR}$	Rotation(3,2)	Vehicle Z-down Y-axis
Front right wheel, Yaw_{FR}	Rotation(3,3)	Vehicle Z-down Z-axis
Rear left wheel, $Roll_{RL}$	Rotation(4,1)	Vehicle Z-down X-axis
Rear left wheel, $Pitch_{RL}$	Rotation(4,2)	Vehicle Z-down Y-axis
Rear left wheel, Yaw_{RL}	Rotation(4,3)	Vehicle Z-down Z-axis
Rear right wheel, $Roll_{RR}$	Rotation(5,1)	Vehicle Z-down X-axis
Rear right wheel, $Pitch_{RR}$	Rotation(5,2)	Vehicle Z-down Y-axis
Rear right wheel, Yaw_{RR}	Rotation(5,3)	Vehicle Z-down Z-axis

Initial array values to scale vehicle per part, Scale — Vehicle initial scale
ones(5, 3) (default) | 5-by-3 array

Initial vehicle and wheel scale, dimensionless. Array dimensions are 5-by-3.

- Scale(1,1), Scale(1,2), and Scale(1,3) — Initial vehicle scale along the inertial vehicle Z-down X-, Y-, and Z- axes, respectively.
- Scale(...,1), Scale(...,2), and Scale(...,3) — Initial wheel scale relative to vehicle, along vehicle Z-down X-, Y-, and Z- axes, respectively.

The parameter contains scale information according to the axle and wheel locations.

$$Scale = \begin{bmatrix} X_{V_{scale}} & Y_{V_{scale}} & Z_{V_{scale}} \\ X_{FL_{scale}} & Y_{FL_{scale}} & Z_{FL_{scale}} \\ X_{FR_{scale}} & Y_{FR_{scale}} & Z_{FR_{scale}} \\ X_{RL_{scale}} & Y_{RL_{scale}} & Z_{RL_{scale}} \\ X_{RR_{scale}} & Y_{RR_{scale}} & Z_{RR_{scale}} \end{bmatrix}$$

Scale	Array Element	Scale Axis
Vehicle, $X_{v_{scale}}$	Scale(1,1)	Vehicle Z-down X-axis
Vehicle, $Y_{v_{scale}}$	Scale(1,2)	Vehicle Z-down Y-axis
Vehicle, $Z_{v_{scale}}$	Scale(1,3)	Vehicle Z-down Z-axis
Front left wheel, $X_{FL_{scale}}$	Scale(2,1)	Vehicle Z-down X-axis
Front left wheel, $Y_{FL_{scale}}$	Scale(2,2)	Vehicle Z-down Y-axis
Front left wheel, $Z_{FL_{scale}}$	Scale(2,3)	Vehicle Z-down Z-axis
Front right wheel, $X_{FR_{scale}}$	Scale(3,1)	Vehicle Z-down X-axis
Front right wheel, $Y_{FR_{scale}}$	Scale(3,2)	Vehicle Z-down Y-axis
Front right wheel, $Z_{FR_{scale}}$	Scale(3,3)	Vehicle Z-down Z-axis
Rear left wheel, $X_{RL_{scale}}$	Scale(4,1)	Vehicle Z-down X-axis
Rear left wheel, $Y_{RL_{scale}}$	Scale(4,2)	Vehicle Z-down Y-axis
Rear left wheel, $Z_{RL_{scale}}$	Scale(4,3)	Vehicle Z-down Z-axis
Rear right wheel, $X_{RR_{scale}}$	Scale(5,1)	Vehicle Z-down X-axis
Rear right wheel, $Y_{RR_{scale}}$	Scale(5,2)	Vehicle Z-down Y-axis
Rear right wheel, $Z_{RR_{scale}}$	Scale(5,3)	Vehicle Z-down Z-axis

Sample time — Sample time
-1 (default) | scalar

Sample time, T_s. The graphics frame rate is the inverse of the sample time.

References

[1] Vehicle Dynamics Standards Committee. *Vehicle Dynamics Terminology.* SAE J670. Warrendale, PA: Society of Automotive Engineers, 2008.

[2] Technical Committee. *Road vehicles — Vehicle dynamics and road-holding ability — Vocabulary.* ISO 8855:2011. Geneva, Switzerland: International Organization for Standardization, 2011.

See Also
Simulation 3D Scene Configuration | Simulation 3D Vehicle with Ground Following

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"3D Visualization Engine Requirements"

Introduced in R2019b

Simulation 3D Message Get

Retrieve data from Unreal Engine visualization environment
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D / Sim3D Core
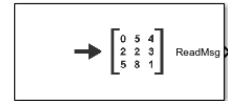
Description

The Simulation 3D Message Get block retrieves data from the Unreal Engine 3D visualization environment. In your model, ensure that the Simulation 3D Scene Configuration block is at the same level as the Simulation 3D Message Get block.

Tip Verify that the Simulation 3D Scene Configuration block executes before the Simulation 3D Message Get block. That way, the Unreal Engine 3D visualization environment prepares the data before the Simulation 3D Message Get block receives it. To check the block execution order, right-click the blocks and select **Properties**. On the **General** tab, confirm these **Priority** settings:
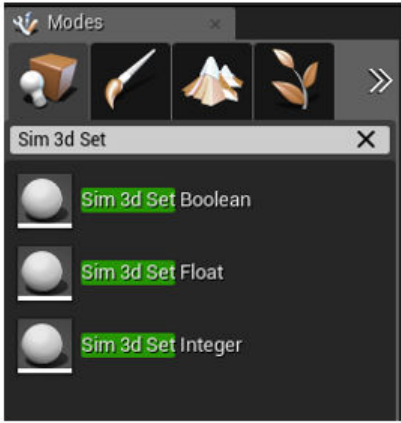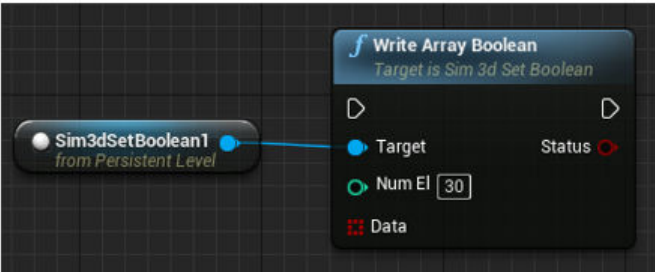
- Simulation 3D Scene Configuration — `0`
- Simulation 3D Message Get — `1`

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Configure Scenes to Send Data

To use the block, you must configure scenes in the Unreal Engine environment to send data to the Simulink model:

1   Install the "Support Package for Customizing Scenes".
2   In the Unreal Editor, follow these general workflows to customize scenes.

Unreal Engine User	Workflow
Blueprint	**a** Instantiate the `Sim3DSet` actor that corresponds to the data type you want to send to the Simulink model. This example shows the Unreal Editor 4.19 `Sim3DSet` data types. **b** Specify an actor tag name that matches the Simulation 3D Message Get block **Signal name** parameter. **c** Navigate to the Level Blueprint. **d** Find the blueprint method for the `Sim3DSet` actor class based on the data type and size specified by the Simulation 3D Message Get block **Data type** and **Message size** parameters. For example, in Unreal Editor 4.19, this diagram shows that `Write Array Boolean` is the method for the `Sim3DSetBoolean` actor class that sends Boolean data type of array size 30. **e** Compile and save the scene. **Note** By default, the Double Lane Change scene has a `Sim3DSetBoolean` actor with tag name `NumOfConesHit`.

Unreal Engine User	Workflow	
C++ class	**a**	Create a new actor class for the mesh or asset that you want the Simulink model to interact with. Derive it from `ASim3dActor`.
	b	In the new actor class:
		• Declare a pointer to the signal name as a class field.
		• Get the class tag.
		• Create a signal writer and assign the pointer in the method `Sim3dSetup`.
		• In the method `Sim3dStep`, invoke the `WriteSimulation3DMessage{DataType}` function to write the data to the Simulink model.
		• Delete the signal writer in the method `Sim3dRelease` of the actor.

For more information about the Unreal Editor, see the Unreal Engine 4 Documentation.

Ports

Output

ReadMsg — Data retrieved from scene
`scalar | array`

Data retrieved from the 3D visualization environment scene data. In the Unreal Engine environment, you can use the `Sim3DSet` class to configure scene actors to send data to the Simulink model.

For example, in the Unreal Editor, the Double Lane Change scene has a `Sim3DSetBoolean` actor with tag name `NumOfConesHit`. Use it to retrieve the number of cones the vehicle hits during a double-lane change maneuver.

This table provides the Double Lane Change scene cone name that corresponds to the `ReadMsg` array element.

Simulation 3D Message Get Block ReadMsg Value	Unreal Editor Cone Name	Simulation 3D Message Get Block Array Element	Unreal Editor Cone Name
`ReadMsg(1,1)`	SM_Cone5	`ReadMsg(2,1)`	SM_Cone10
`ReadMsg(1,2)`	SM_Cone4	`ReadMsg(2,2)`	SM_Cone09
`ReadMsg(1,3)`	SM_Cone3	`ReadMsg(2,3)`	SM_Cone08
`ReadMsg(1,4)`	SM_Cone2	`ReadMsg(2,4)`	SM_Cone07
`ReadMsg(1,5)`	SM_Cone01	`ReadMsg(2,5)`	SM_Cone06
`ReadMsg(1,6)`	SM_Cone15	`ReadMsg(2,6)`	SM_Cone20
`ReadMsg(1,7)`	SM_Cone14	`ReadMsg(2,7)`	SM_Cone19
`ReadMsg(1,8)`	SM_Cone13	`ReadMsg(2,8)`	SM_Cone18

Simulation 3D Message Get Block ReadMsg Value	Unreal Editor Cone Name	Simulation 3D Message Get Block Array Element	Unreal Editor Cone Name
ReadMsg(1,9)	SM_Cone12	ReadMsg(2,9)	SM_Cone17
ReadMsg(1,10)	SM_Cone11	ReadMsg(2,10)	SM_Cone16
ReadMsg(1,11)	SM_Cone25	ReadMsg(2,11)	SM_Cone30
ReadMsg(1,12)	SM_Cone24	ReadMsg(2,12)	SM_Cone29
ReadMsg(1,13)	SM_Cone23	ReadMsg(2,13)	SM_Cone28
ReadMsg(1,14)	SM_Cone22	ReadMsg(2,14)	SM_Cone27
ReadMsg(1,15)	SM_Cone21	ReadMsg(2,15)	SM_Cone26

Parameters

Signal name, SigName — Message signal name
mySignal (default)

Specifies the signal name in the 3D visualization environment. In the Unreal Engine environment, use the Sim3DSet actor class 'Tags' property located in the 'Details' pane.

For example, you can retrieve data from the double-lane change scene that indicates if cones are hit during a double-lane change maneuver. To retrieve cone hit data from the double-lane change scene, set this parameter to NumOfConesHit. In the double-lane change scene, the Sim3DSet actor class 'Tags' property is set to NumOfConesHit.

Data type, DataType — Message data type
uint8 (default) | double | single | int8 | int16 | uint16 | int32 | uint32 | boolean

3D visualization environment signal data type. In the Unreal Engine environment, instantiate the Sim3DSet actor class for the data type that you want to send to the Simulink model.

For example, you can retrieve data from the double-lane change scene that indicates if cones are hit during a double-lane change maneuver. To retrieve cone hit data from the double-lane change scene, set this parameter to boolean. In the double-lane change scene, the Sim3DSetBoolean actor class is instantiated to send the cone hit or miss boolean data.

Message size, MsgSize — Message dimension
[1 1] (default) | scalar | array

3D visualization environment signal dimension. In the Unreal Engine environment blueprint, set the input to the node of the Sim3DSet actor class to specify the dimensions of data that you want to send to the Simulink model.

For example, you can retrieve data from the double-lane change scene that indicates if cones are hit during a double-lane change maneuver. To retrieve cone hit data from the double-lane change scene, set this parameter to [2 15]. In the double-lane change scene, the input to the blueprint node for the Sim3DSetBoolean actor class is set to 30, the number of cones in the scene.

Sample time — Sample time
0.02 (default) | -1 | scalar

Sample time, in s. The graphics frame rate is the inverse of the sample time. If you set the sample time to -1, the block uses the sample time specified in the Simulation 3D Scene Configuration block.

See Also

Double Lane Change | Simulation 3D Message Set | Simulation 3D Scene Configuration

Topics
"Send and Receive Double-Lane Change Scene Data"
"Support Package for Customizing Scenes"
"3D Visualization Engine Requirements"

External Websites
Unreal Engine

Introduced in R2019b

Simulation 3D Message Set

Send data to Unreal Engine visualization environment
Library: Vehicle Dynamics Blockset / Vehicle Scenarios / Sim3D / Sim3D Core
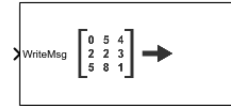
Description

The Simulation 3D Message Set block sends data to the Unreal Engine 3D visualization environment. In your model, ensure that the Simulation 3D Scene Configuration block is at the same level as the Simulation 3D Message Set block.

Tip Verify that the Simulation 3D Message Set block executes before the Simulation 3D Scene Configuration block. That way, Simulation 3D Message Set prepares the signal data before the Unreal Engine 3D visualization environment receives it. To check the block execution order, right-click the blocks and select **Properties**. On the **General** tab, confirm these **Priority** settings:
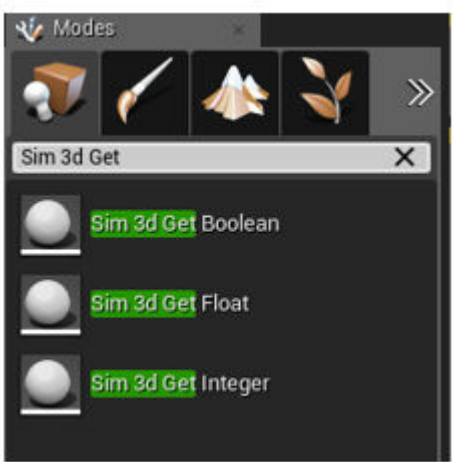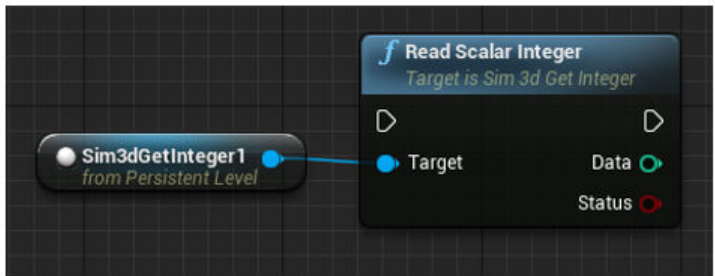
- Simulation 3D Scene Configuration — `0`
- Simulation 3D Message Set — `-1`

For more information about execution order, see "Control and Display Execution Order" (Simulink).

Configure Scenes to Receive Data

To use the block, you must configure scenes in the Unreal Engine environment to receive data from the Simulink model:

1. Install the "Support Package for Customizing Scenes".
2. In the Unreal Editor, follow these general workflows to customize scenes.

Unreal Engine User	Workflow
Blueprint	**a** Instantiate the `Sim3DGet` actor that corresponds to the data type you want to receive from the Simulink model. This example shows the Unreal Editor 4.19 `Sim3DGet` data types. **b** Specify an actor tag name that matches the Simulation 3D Message Set block **Signal name** parameter. **c** Navigate to the Level Blueprint. **d** Find the blueprint method for the `Sim3DGet` actor class based on the data type and size that you want to receive from the Simulink model. For example, in Unreal Editor 4.19, this diagram shows that `Read Scalar Integer` is the method for `Sim3DGetInteger` actor class to receive `int32` data type of size scalar. **e** Compile and save the scene. **Note** By default, the Double Lane Change scene has a `Sim3DGetInteger` actor with tag name `TrafficLight1`.

Unreal Engine User	Workflow	
C++ class	**a**	Create a new actor class for the mesh or asset that you want the Simulink model to interact with. Derive it from `ASim3dActor`.
	b	In the new actor class: • Declare a pointer to the signal name as a class field. • Get the class tag. • Create a signal reader and assign the pointer in the method `Sim3dSetup`. • In the method `Sim3dStep`, invoke the `ReadSimulation3DMessage{DataType}` function to read the data from a Simulink model. • Delete the signal reader in the method `Sim3dRelease` of the actor.

For more information about the Unreal Editor, see the Unreal Engine 4 Documentation.

Ports

Input

`WriteMsg` — Data sent to scene
scalar | array

Data sent to the 3D visualization environment scene. In the Unreal Engine environment, you can configure the `Sim3DGet` class to receive the data from the Simulink model.

For example, in the Unreal Editor, the Double Lane Change scene has a `Sim3DGetInteger` integer actor with tag name `TrafficLight1`. The integer actor reads `int32` data type from the Simulink model. You can use it to control the traffic signal light color.

This table provides the scene traffic signal light color that corresponds to the `WriteMsg` value in the Double Lane Change scene.

Simulation 3D Message Set Block `WriteMsg` Value	`TrafficLight1` Color
`0`	`Red`
`1`	`Yellow`
`2`	`Green`

Parameters

`Signal name, SigName` — Message signal name
`mySignal` (default)

Specifies the signal name in the 3D visualization environment. In the Unreal Engine environment, use the `Sim3Get` actor class 'Tags' property located in the 'Details' pane.

For example, you can send data to the double lane change scene that changes the traffic signal light color to red, yellow, or green. To send data to the traffic signal light, set this parameter to `TrafficLight1`. In the double lane change scene, the 'Tags' property value for `Sim3dGetInteger` actor class is set to TrafficLight1.

Sample time — Sample time
`0.02` (default) | `-1` | `scalar`

Sample time, in s. The graphics frame rate is the inverse of the sample time. If you set the sample time to -1, the block uses the sample time specified in the Simulation 3D Scene Configuration block.

See Also

Double Lane Change | Simulation 3D Message Get | Simulation 3D Scene Configuration

Topics
"Send and Receive Double-Lane Change Scene Data"
"Support Package for Customizing Scenes"
"3D Visualization Engine Requirements"

Introduced in R2019b

Scenes

Straight Road

Straight road 3D environment

Description

The **Straight Road** scene is a 3D environment of a straight four-lane divided highway. The scene is rendered using the Unreal Engine from Epic Games.


To simulate a driving maneuver in this scene:

1   Add a Simulation 3D Scene Configuration block to your Simulink model.
2   In this block, set the **Scene source** parameter to `Default Scenes`.
3   Set the enabled **Scene name** parameter to `Straight road`.

Scene Layout
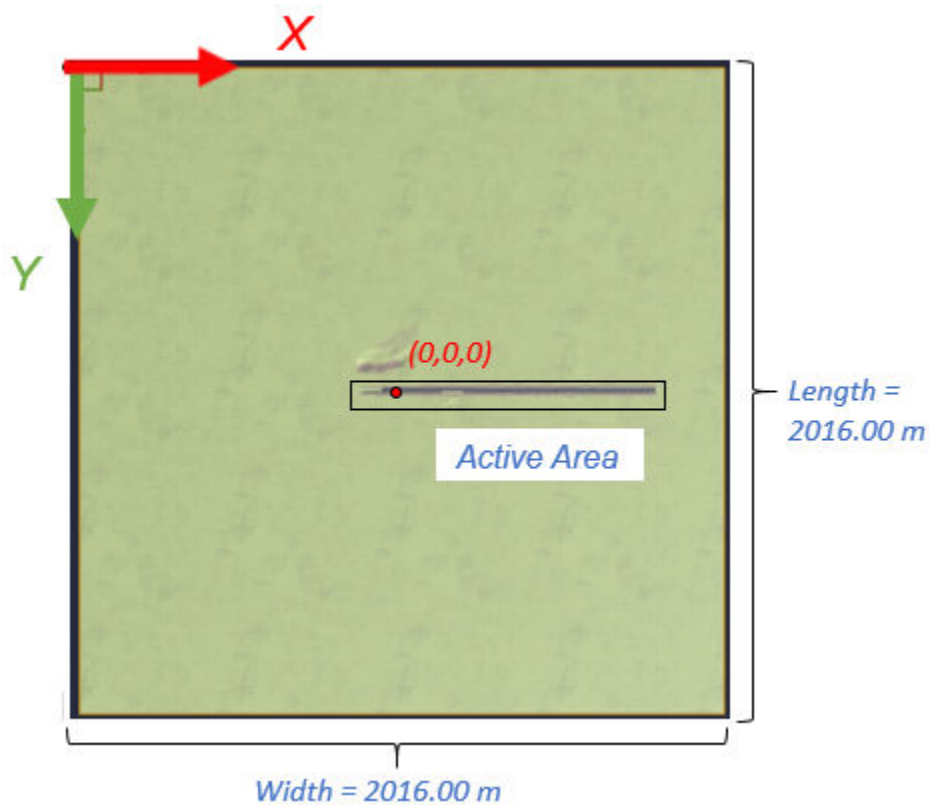
The scene uses the world coordinate system to locate objects.

The active area of the scene contains the road.
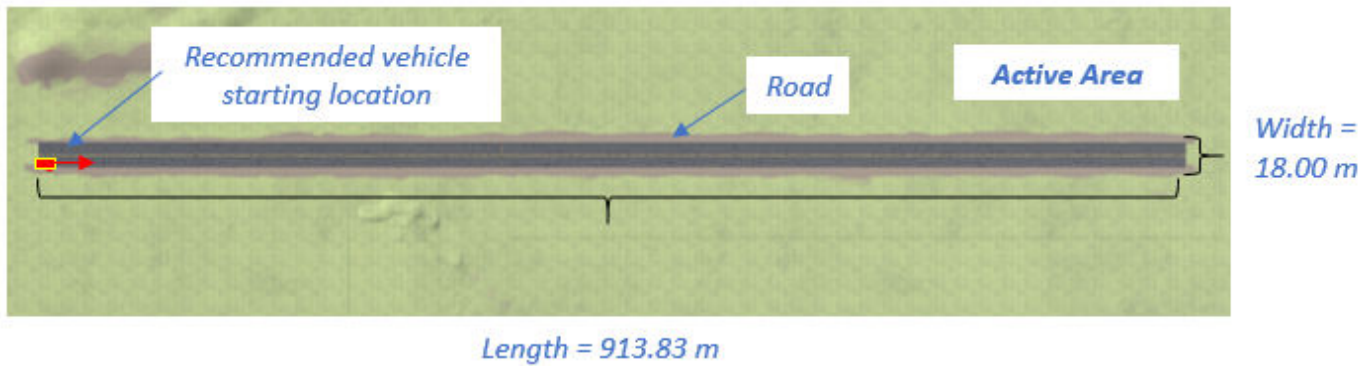
This table provides the scene area corner locations in the world coordinate system. Dimensions are in m.

| Locations	X	Y	Z
	(m)	(m)	(m)
Scene — Top left	-1080	-1080	0
Scene — Bottom right	1080	1080	0
Active area — Bottom left	-122.19	6.00	0

Recommended Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X	Y	Z	Roll	Pitch	Yaw
(m)	(m)	(m)	(deg)	(deg)	(deg)
-118	3.125	0	0	0	0

Lane Dimensions

This figure and table provides the lane dimensions, in m.
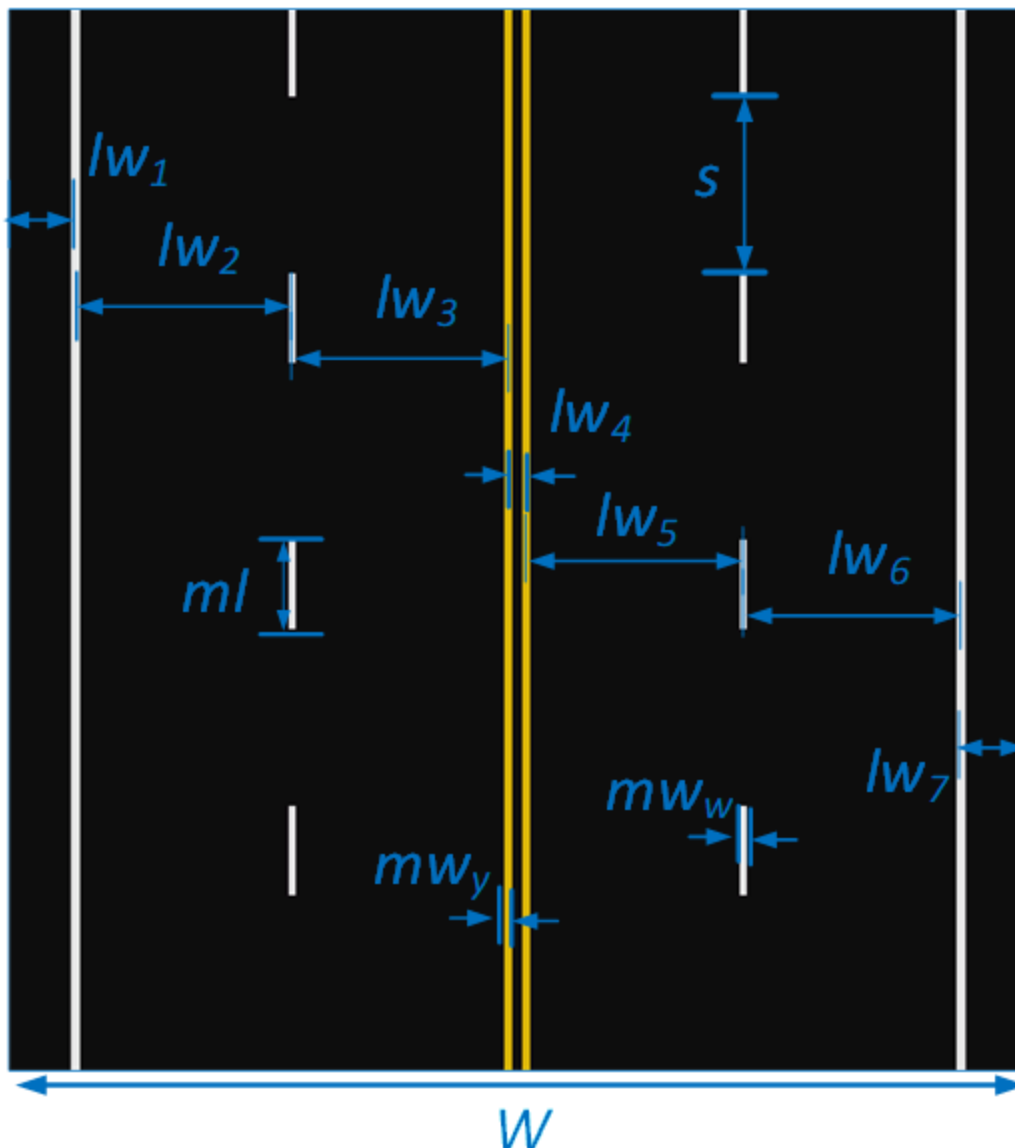
Variable	Dimension (m)
lw_1	1.2
lw_2	3.84
lw_3	3.8
lw_4	0.36
lw_5	3.8
lw_6	3.8
lw_7	1.2
ml	1.5
s	3.0
mw_w	0.13
mw_y	0.14
W	18.0

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.

Axis	Description
X	Forward direction of the vehicle
	Roll — Right-handed rotation about X-axis
Y	Extends to the right of the vehicle, parallel to the ground plane
	Pitch — Right-handed rotation about Y-axis
Z	Extends upwards
	Yaw — Left-handed rotation about Z-axis

Tips

- If you have the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, then you can modify this scene. In the Unreal Engine project file that comes with the support package, this scene is named `HwStrght`.

  For more details on customizing scenes, see "Support Package for Customizing Scenes".

See Also

Curved Road | **Double Lane Change** | **Large Parking Lot** | **Open Surface** | **Parking Lot** | Simulation 3D Scene Configuration | **US City Block** | **US Highway** | **Virtual Mcity**

Topics
"3D Visualization Engine Requirements"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

Curved Road

Curved road 3D environment

Description

The **Curved Road** scene is a 3D environment of a curved highway loop. The scene is rendered using the Unreal Engine from Epic Games.



To simulate a driving maneuver in this scene:

1   Add a Simulation 3D Scene Configuration block to your Simulink model.

2   In this block, set the **Scene source** parameter to `Default Scenes`.

3   Set the enabled **Scene name** parameter to `Curved road`.

Scene Layout
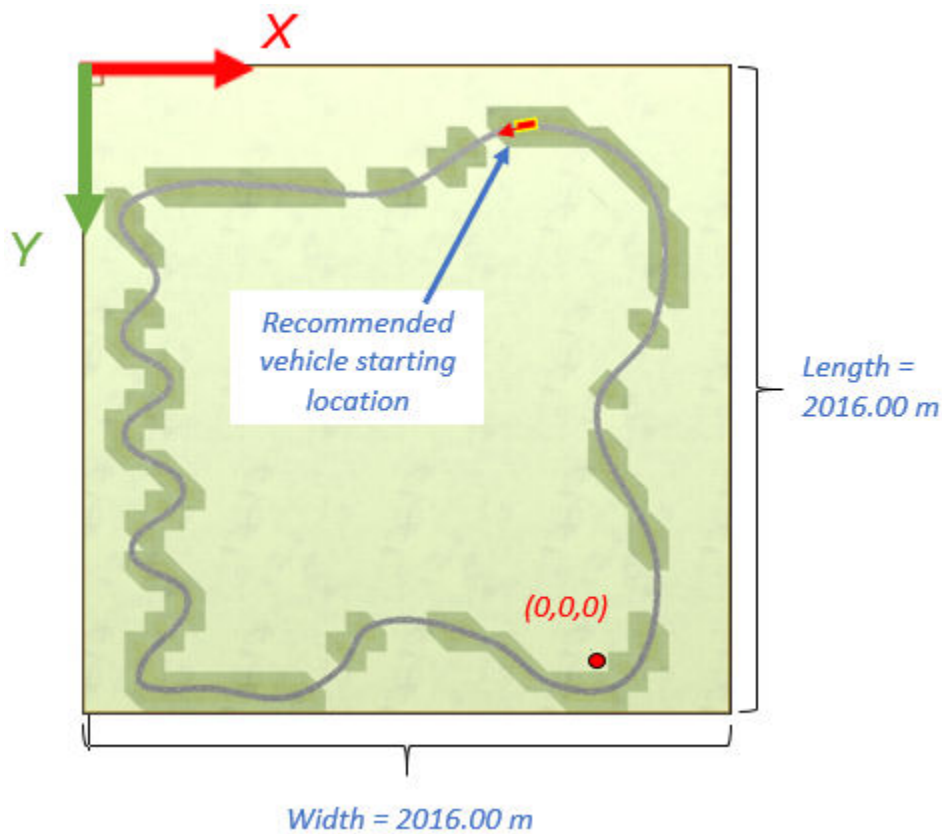
The scene uses the world coordinate system to locate objects.

This table provides the scene corner locations in the world coordinate system. Dimensions are in m.

Location	X (m)	Y (m)	Z (m)
Scene — Bottom left	-1587.75	195.39	0
Scene — Top right	428.26	-1820.60	0

Recommended Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
0.2	-1605.00	0	0	0	−156°

Lane Dimensions

This figure and table provides the lane dimensions, in m.
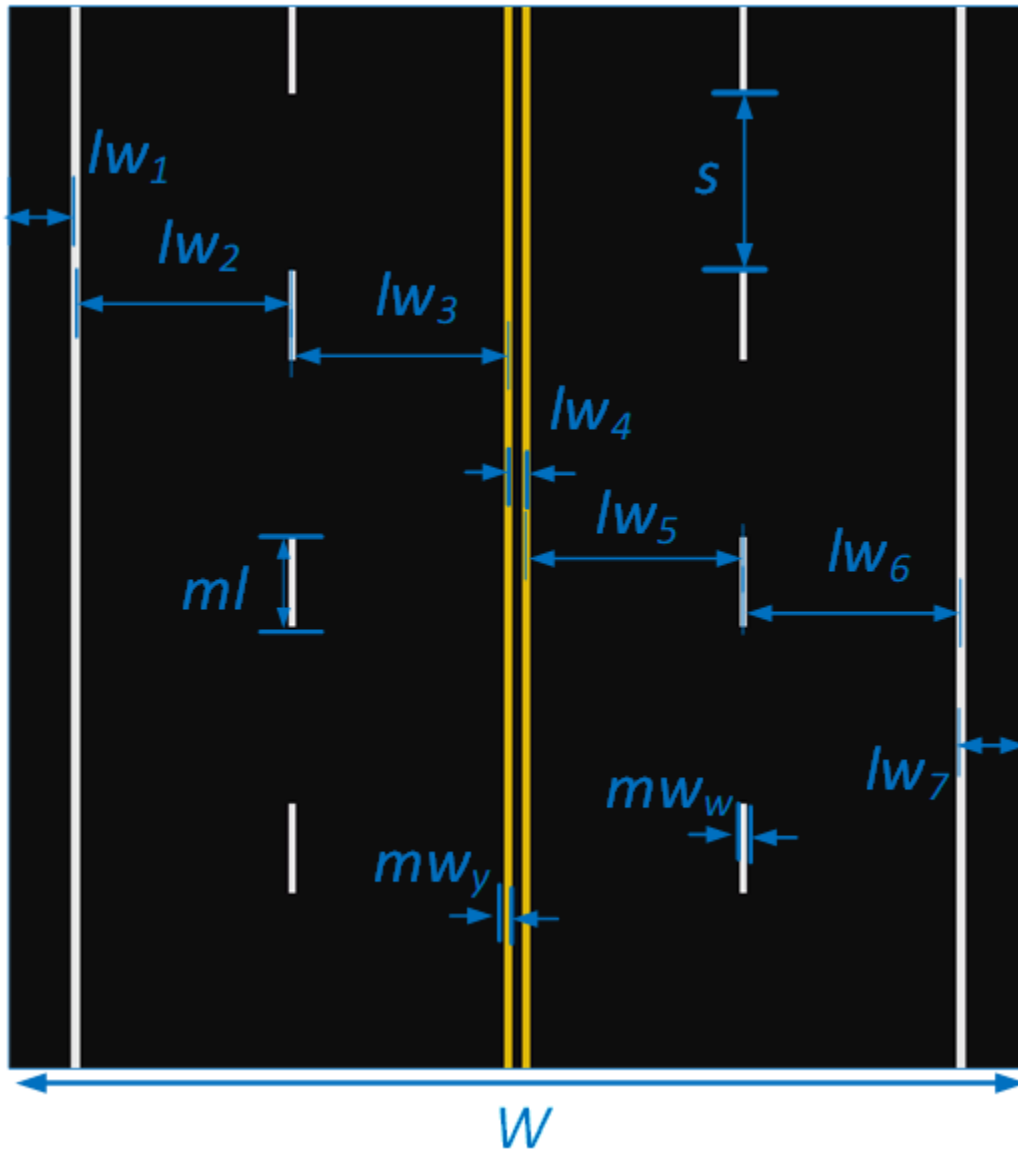
Variable	Dimension (m)
lw_1	1.2
lw_2	3.82
lw_3	3.82
lw_4	0.3
lw_5	3.83
lw_6	3.83
lw_7	1.2
ml	1.5
s	3.0

Variable	Dimension (m)
mw_w	0.13
mw_y	0.15
W	18.0

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.

Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about X-axis
Y	Extends to the right of the vehicle, parallel to the ground plane Pitch — Right-handed rotation about Y-axis
Z	Extends upwards Yaw — Left-handed rotation about Z-axis

Tips

• If you have the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, then you can modify this scene. In the Unreal Engine project file that comes with the support package, this scene is named `HwCurve`.

  For more details on customizing scenes, see "Support Package for Customizing Scenes".

See Also
Double Lane Change | **Large Parking Lot** | **Open Surface** | **Parking Lot** | Simulation 3D Scene Configuration | **Straight Road** | **US City Block** | **US Highway** | **Virtual Mcity**

Topics
"3D Visualization Engine Requirements"

"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

Parking Lot

Parking lot 3D environment

Description

The **Parking Lot** scene is a 3D environment of a parking lot. The scene is rendered using the Unreal Engine from Epic Games.

To simulate a driving maneuver in this scene:

1   Add a Simulation 3D Scene Configuration block to your Simulink model.
2   In this block, set the **Scene source** parameter to `Default Scenes`.
3   Set the enabled **Scene name** parameter to `Parking lot`.

Scene Layout

The scene uses the world coordinate system to locate objects.
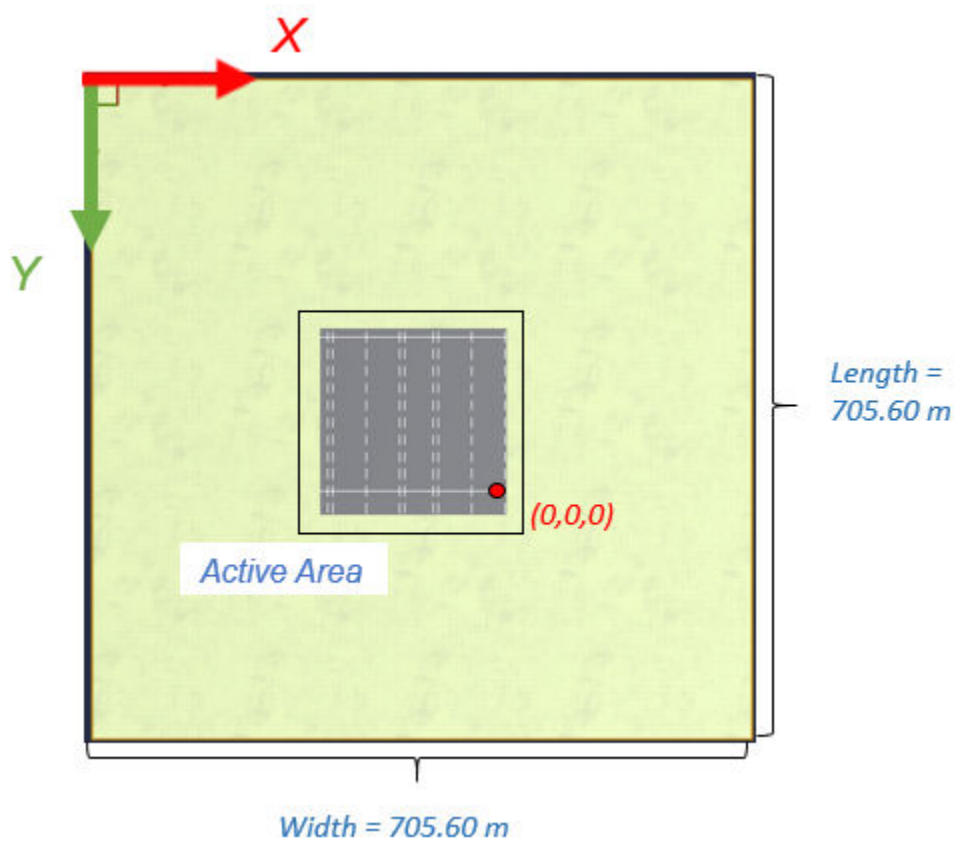
The active area of the scene contains the parking lot.
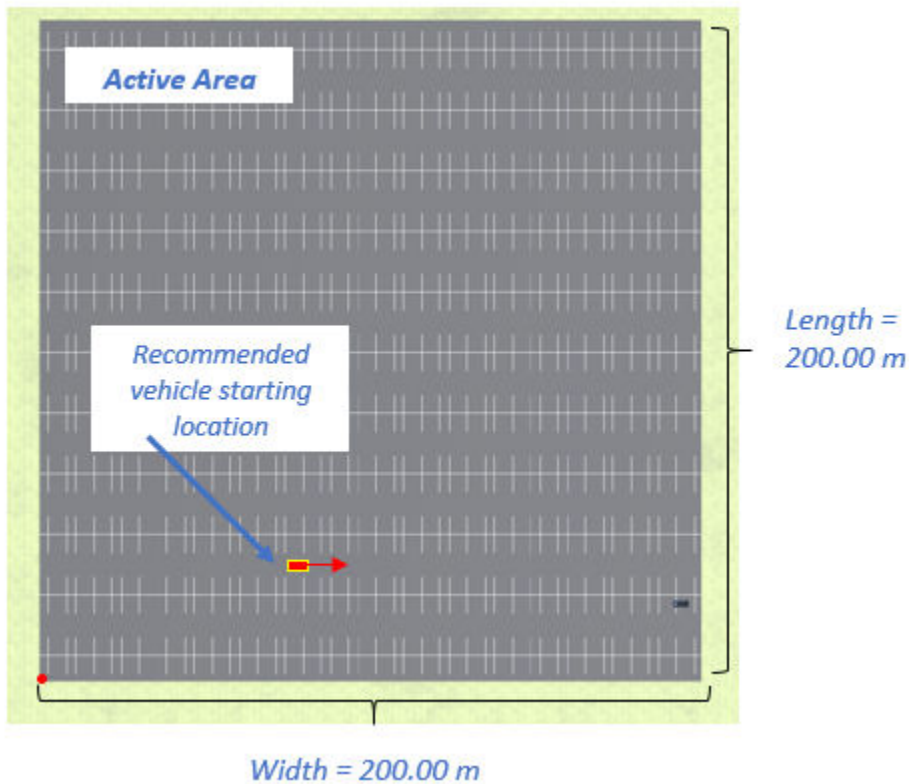
This table provides the scene and active area corner locations in the world coordinate system. Dimensions are in m.

Locations	X (m)	Y (m)	Z (m)
Scene — Bottom left	-437.32	262.79	0
Scene — Top right	268.28	-442.81	0
Active area — Bottom left	-193.86	23.43	0

Recommended Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
-104.0	-9.7	0	0	0	0

Parking Space Dimensions

This figure shows the parking space dimensions, in m.

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.

Axis	Description
X	Forward direction of the vehicle
	Roll — Right-handed rotation about *X*-axis
Y	Extends to the right of the vehicle, parallel to the ground plane
	Pitch — Right-handed rotation about *Y*-axis
Z	Extends upwards
	Yaw — Left-handed rotation about *Z*-axis

Tips

- If you have the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, then you can modify this scene. In the Unreal Engine project file that comes with the support package, this scene is named `SimpleLot`.

  For more details on customizing scenes, see "Support Package for Customizing Scenes".

See Also

Curved Road | **Double Lane Change** | **Large Parking Lot** | **Open Surface** | Simulation 3D Scene Configuration | **Straight Road** | **US City Block** | **US Highway** | **Virtual Mcity**

Topics
"3D Visualization Engine Requirements"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

Large Parking Lot

Large parking lot 3D environment

Description

The **Large Parking Lot** scene is a 3D environment of a large parking lot that contains cones, curbs, traffic signs, and parked vehicles. The scene is rendered using the Unreal Engine from Epic Games.

To simulate a driving maneuver in this scene:

1   Add a Simulation 3D Scene Configuration block to your Simulink model.
2   In this block, set the **Scene source** parameter to `Default Scenes`.
3   Set the enabled **Scene name** parameter to `Large parking lot`.

Scene Layout

The scene uses the world coordinate system to locate objects.

This table provides the scene area corner locations in the world coordinate system. Dimensions are in m and deg.

Locations	X (m)	Y (m)	Z (m)
Scene — Top left	-78.6	-73.5	0
Scene — Bottom right	72.6	77.7	0

Recommended Vehicle Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
45.0	54.7	0	0	0	-90

Parking Space Dimensions

This figure shows the parking space dimensions, in m.

Other Vehicles

This table provides the vehicle tag names and initial locations for other vehicles in the scene, in the world coordinate system. Dimensions are in m and deg.

Object	Unreal Engine Editor Name	Locations					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Vehicle	SM_Hatchback	5.68	-14.25	0.05	0	0	-90
	SM_PickupTruck2	5.59	-40.40	0.05	0	0	90
	SM_PickupTruck	-5.35	-34.87	0.05	0	0	-90
	SM_MuscleCar	29.70	-13.80	0.05	0	0	-90
	SM_PickupTruck3	11.10	-0.9	0.05	0	0	90
	SM_SUVCar	11.10	4.80	0.05	0	0	-90
	SM_SedanCar4	5.60	4.80	0.05	0	0	-90
	SM_Hatchback2	-16.10	4.40	0.05	0	0	-90
	SM_SedanCar	-21.50	18.40	0.05	0	0	90
	SM_SUVCar2	-24.20	18.40	0.05	0	0	90
	SM_SedanCar2	-26.90	18.40	0.05	0	0	90
	SM_SedanCar3	-40.40	18.40	0.05	0	0	90
	SM_MuscleCar2	11.10	25.90	0.05	0	0	-90
	SM_Hatchback3	-21.50	38.90	0.05	0	0	90
	SM_PickupTruck4	-27.00	46.40	0.05	0	0	-90

Cones

This table provides the object names and locations in the world coordinate system. Dimensions are in m and deg.

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Cone	SM_Cone1	-21.43	-23.41	0.05	0	0	0
	SM_Cone2	-24.20	36.19	0.05	0	0	0
	SM_Cone3	-37.74	48.02	0.05	0	0	0
	SM_Cone4	-26.97	-2.68	0.05	0	0	0
	SM_Cone5	13.95	28.21	0.05	0	0	0

Traffic Signs
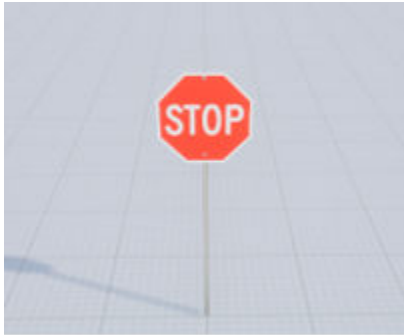
This table provides the object names and locations in the world coordinate system. Dimensions are in m and deg.

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Traffic signs	SM_StopSign1	34.77	57.38	0.11	0	0	0
	SM_StopSign2	35.29	36.38	0.11	0	0	0
	SM_StopSign3	35.28	15.955	0.11	0	0	180
	SM_StopSign4	35.35	-2.92	0.11	0	0	180
	SM_StopSign5	35.69	-23.64	0.11	0	0	0
	SM_DisabilitySign1	23.81	42.71	0.11	0	0	0
	SM_DisabilitySign2	29.21	42.71	0.11	0	0	0
	SM_DisabilitySign3	23.81	41.71	0.11	0	0	180

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
	SM_DisabilitySign 4	29.21	41.71	0.11	0	0	180
	SM_DisabilitySign 5	24.25	-17.01	0.11	0	0	0
	SM_DisabilitySign 6	29.69	-17.01	0.11	0	0	0
	SM_DisabilitySign 7	25.25	-18.31	0.11	0	0	180
	SM_DisabilitySign 8	29.69	-18.31	0.11	0	0	180

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.

Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about *X*-axis
Y	Extends to the right of the vehicle, parallel to the ground plane Pitch — Right-handed rotation about *Y*-axis
Z	Extends upwards Yaw — Left-handed rotation about *Z*-axis

Tips

- If you have the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, then you can modify this scene. In the Unreal Engine project file that comes with the support package, this scene is named `LargeParkingLot`.

  For more details on customizing scenes, see "Support Package for Customizing Scenes".

See Also

Curved Road | **Double Lane Change** | **Open Surface** | **Parking Lot** | Simulation 3D Scene Configuration | **Straight Road** | **US City Block** | **US Highway** | **Virtual Mcity**

Topics
"3D Visualization Engine Requirements"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

Open Surface

Open surface 3D environment

Description

The **Open Surface** scene contains a 3D environment of an open, black road surface. The scene is rendered using the Unreal Engine from Epic Games.


To simulate a driving maneuver in this scene:

1 Add a Simulation 3D Scene Configuration block to your Simulink model.
2 In this block, set the **Scene source** parameter to `Default Scenes`.
3 Set the enabled **Scene name** parameter to `Open surface`.

Scene Layout

The scene uses the world coordinate system to locate objects.
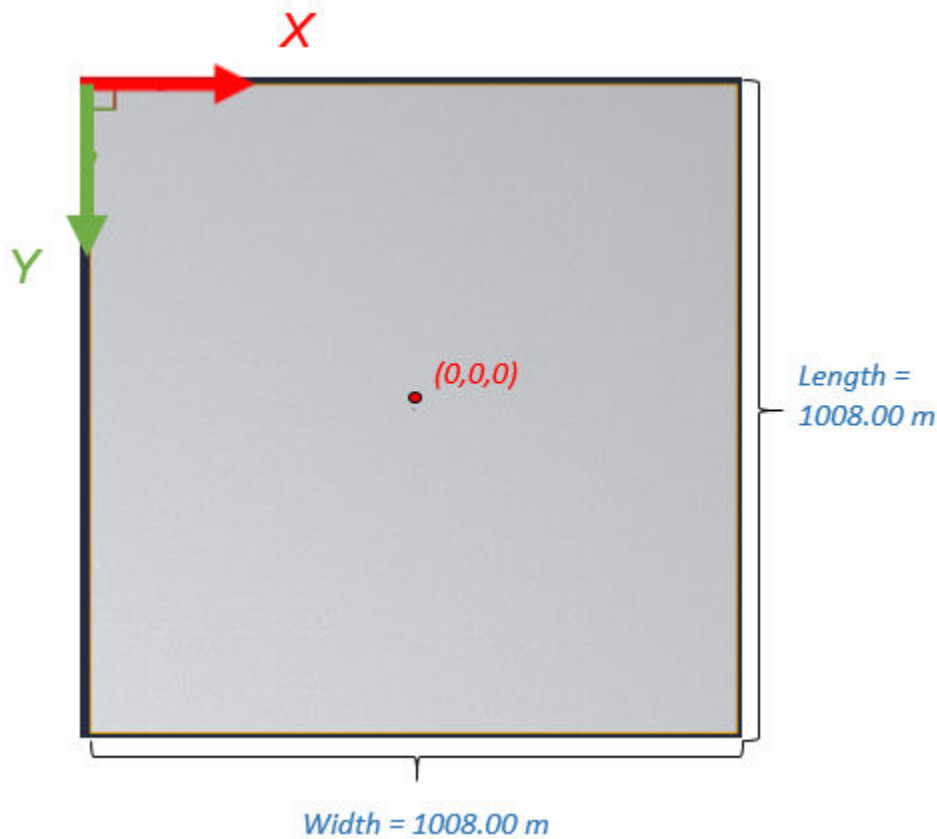
This table provides the scene corner locations in the world coordinate system. Dimensions are in m.

Location	X	Y	Z
Scene — Bottom left	-504.00	504.00	0
Scene — Top right	504.00	-504.00	0

Recommended Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
0	0	0	0	0	0

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.

Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about *X*-axis
Y	Extends to the right of the vehicle, parallel to the ground plane Pitch — Right-handed rotation about *Y*-axis
Z	Extends upwards Yaw — Left-handed rotation about *Z*-axis

Tips

• If you have the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, then you can modify this scene. In the Unreal Engine project file that comes with the support package, this scene is named `BlackLake`.

For more details on customizing scenes, see "Support Package for Customizing Scenes".

See Also
Curved Road | **Double Lane Change** | **Large Parking Lot** | **Parking Lot** | Simulation 3D Scene Configuration | **Straight Road** | **US City Block** | **US Highway** | **Virtual Mcity**

Topics
"3D Visualization Engine Requirements"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

Double Lane Change

Double lane change 3D environment

Description

The **Double Lane Change** scene is a 3D environment of a straight road containing cones, traffic signs, and barrels. The cones are set up for a vehicle to perform a double lane change maneuver. The scene is rendered using the Unreal Engine from Epic Games.

To simulate a driving maneuver in this scene:

1. Add a Simulation 3D Scene Configuration block to your Simulink model.
2. In this block, set the **Scene source** parameter to `Default Scenes`.
3. Set the enabled **Scene name** parameter to `Double lane change`.

Scene Layout

The scene uses the world coordinate system to locate objects.
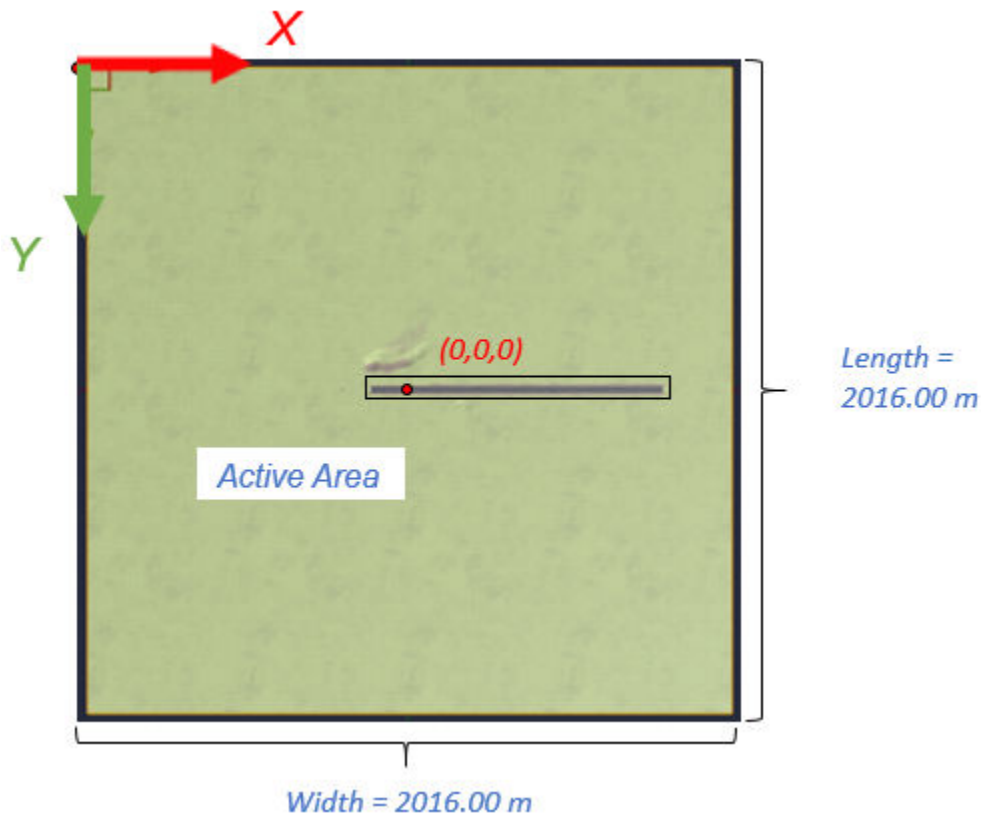
The active area of the scene contains the road.
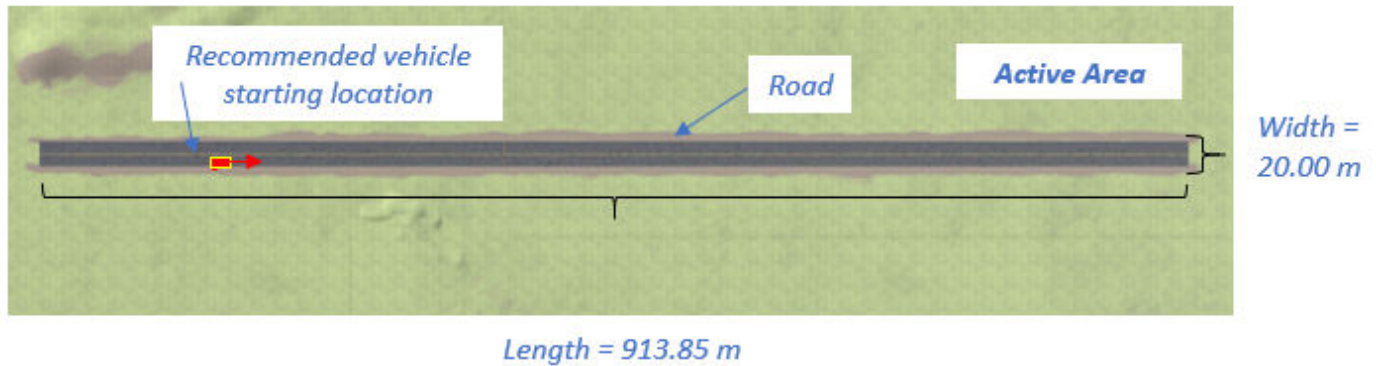
This table provides the scene area corner locations in the world coordinate system. Dimensions are in m.

Locations	X	Y	Z
Scene — Top left	-1008	-1008	0
Scene — Bottom right	1008	1008	0
Active area — Bottom left	-122.19	6.99	0

Recommended Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
0	3.125	0	0	0	0

Cones

This table provides the object names and locations in the world coordinate system. Dimensions are in m.

Object	Unreal Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Cone	SM_Cone01	187.90	1.70	.009	0	0	0
	SM_Cone02	184.90	1.70	.009			
	SM_Cone03	181.90	1.70	.009			
	SM_Cone04	178.90	1.70	.009			
	SM_Cone05	175.90	1.70	.009			
	SM_Cone06	187.90	4.73	.009			
	SM_Cone07	184.90	4.73	.009			
	SM_Cone08	181.90	4.73	.009			
	SM_Cone09	178.90	4.73	.009			
	SM_Cone10	175.90	4.73	.009			
	SM_Cone11	212.40	-2.86	.009			
	SM_Cone12	209.65	-2.86	.009			
	SM_Cone13	206.90	-2.86	.009			
	SM_Cone14	204.15	-2.86	.009			

Object	Unreal Editor Name	Location			Roll	Pitch	Yaw
		X (m)	Y (m)	Z (m)	(deg)	(deg)	(deg)
	SM_Cone15	201.40	-2.86	.009			
	SM_Cone16	212.40	0.69	.009			
	SM_Cone17	209.65	0.69	.009			
	SM_Cone18	206.90	0.69	.009			
	SM_Cone19	204.15	0.69	.009			
	SM_Cone20	201.40	0.69	.009			
	SM_Cone21	236.90	1.70	.009			
	SM_Cone22	236.90	1.70	.009			
	SM_Cone23	230.90	1.70	.009			
	SM_Cone24	227.90	1.70	.009			
	SM_Cone25	224.90	1.70	.009			
	SM_Cone26	236.90	4.95	.009			
	SM_Cone27	233.90	4.95	.009			
	SM_Cone28	230.90	4.95	.009			
	SM_Cone29	227.90	4.95	.009			
	SM_Cone30	224.90	4.95	.009			
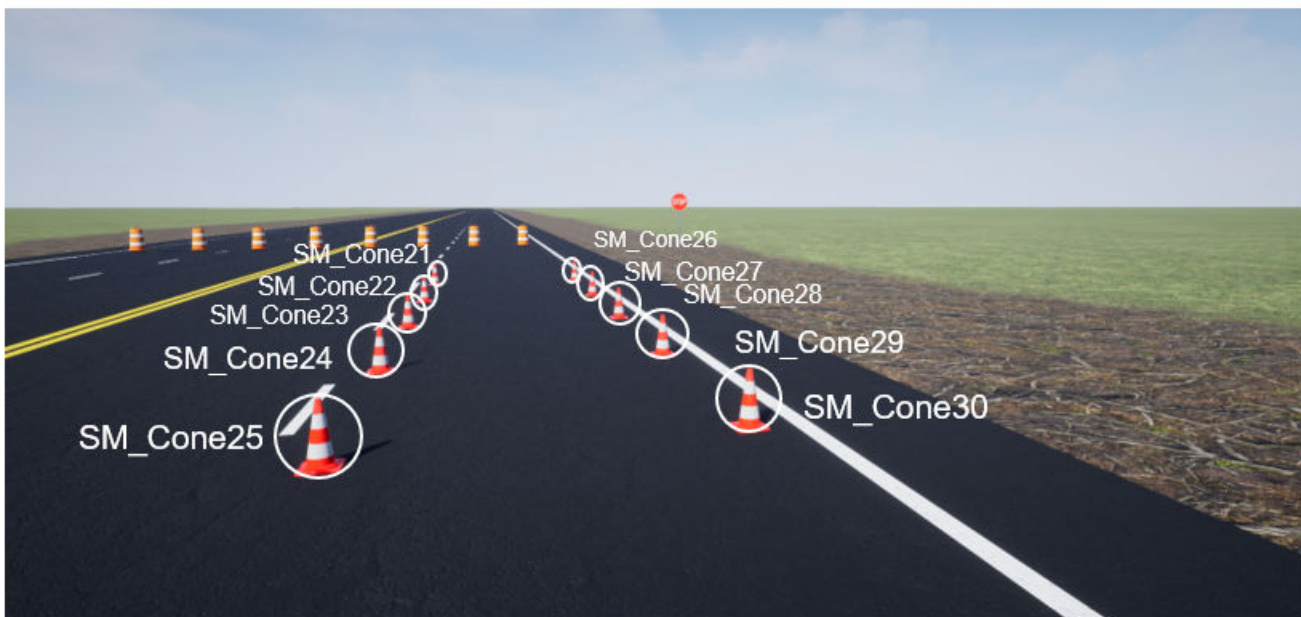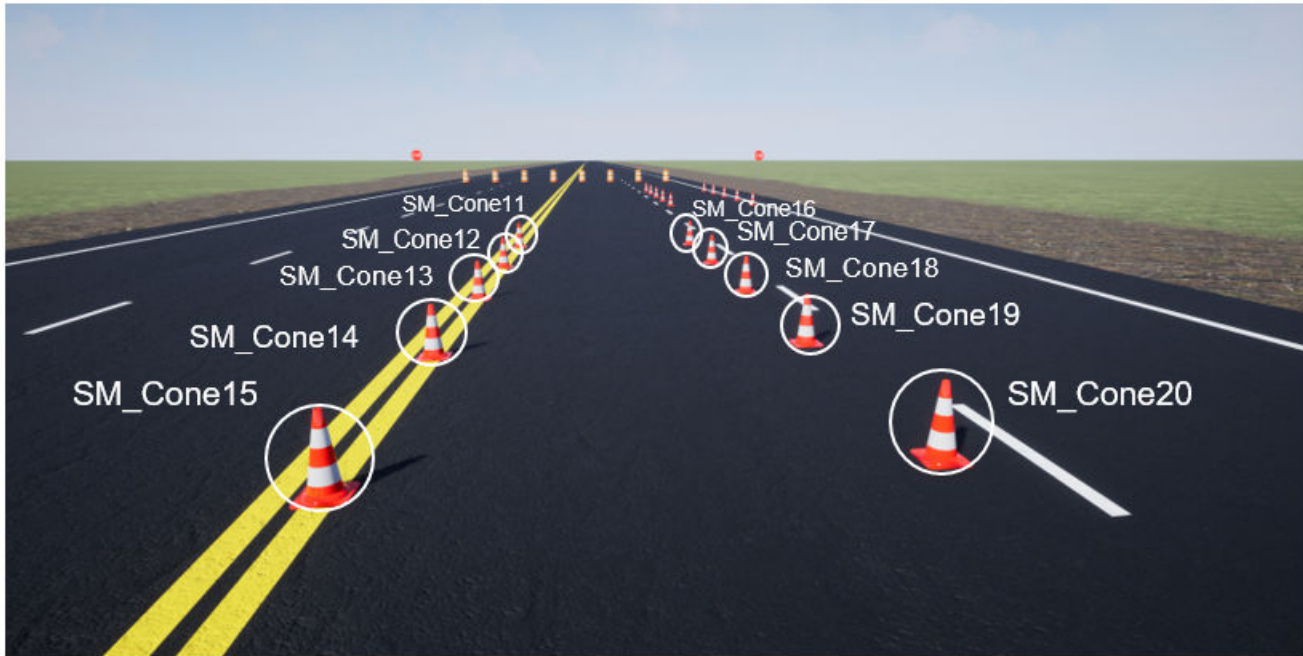
In the Unreal Editor, the scene has a `Sim3DSetBoolean` actor with signal name `NumOfConesHit`. You can use it with the Simulation 3D Message Get block to retrieve how many cones the vehicle hits during a double-lane change maneuver.

Traffic Signs
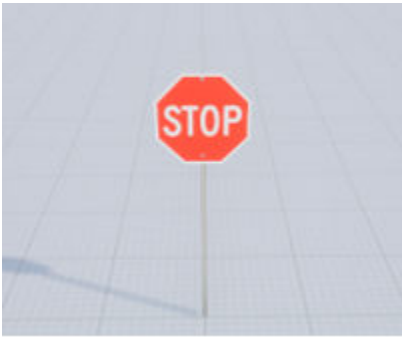
This table provides the object names and locations in the world coordinate system. Dimensions are in m.

Object	Unreal Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Traffic sign	SM_StopSign	248.80	-13.10	0	0	0	0
	SM_StopSign2	248.80	10.90	0			

Traffic Signal Light

This table provides the object name and location in the world coordinate system. Dimensions are in m.

Object	Unreal Editor Name	Location					
		X	Y	Z	Roll	Pitch	Yaw
		(m)	(m)	(m)	(deg)	(deg)	(deg)
Traffic signal light	SM_Traffic LightsSide Only	5.43	6.00	0	0	0	180.00°

In the Unreal Editor, the Double Lane Change scene has a `Sim3DGetInteger` actor with signal name `TrafficLight1`. You can use it with the Simulation 3D Message Set block to control the traffic signal light color.

Barrels

This table provides the object names and locations in the world coordinate system. Dimensions are in m.

Object	Unreal Editor Name	Location					
		X	Y	Z	Roll	Pitch	Yaw
		(m)	(m)	(m)	(deg)	(deg)	(deg)
Barrels	SM_Traffic Barrel1	252.70	4.50	0	0	0	180.00°
	SM_Traffic Barrel2	252.70	2.35	0	0	0	0
	SM_Traffic Barrel3	252.70	.20	0	0	0	0
	SM_Traffic Barrel4	252.70	-1.95	0	0	0	0
	SM_Traffic Barrel5	252.70	-4.10	0	0	0	0
	SM_Traffic Barrel6	252.70	-6.25	0	0	0	0
	SM_Traffic Barrel7	252.70	-8.40	0	0	0	0
	SM_Traffic Barrel8	252.70	-10.55	0	0	0	0

Lane Dimensions

This figure and table provides the lane dimensions, in m.
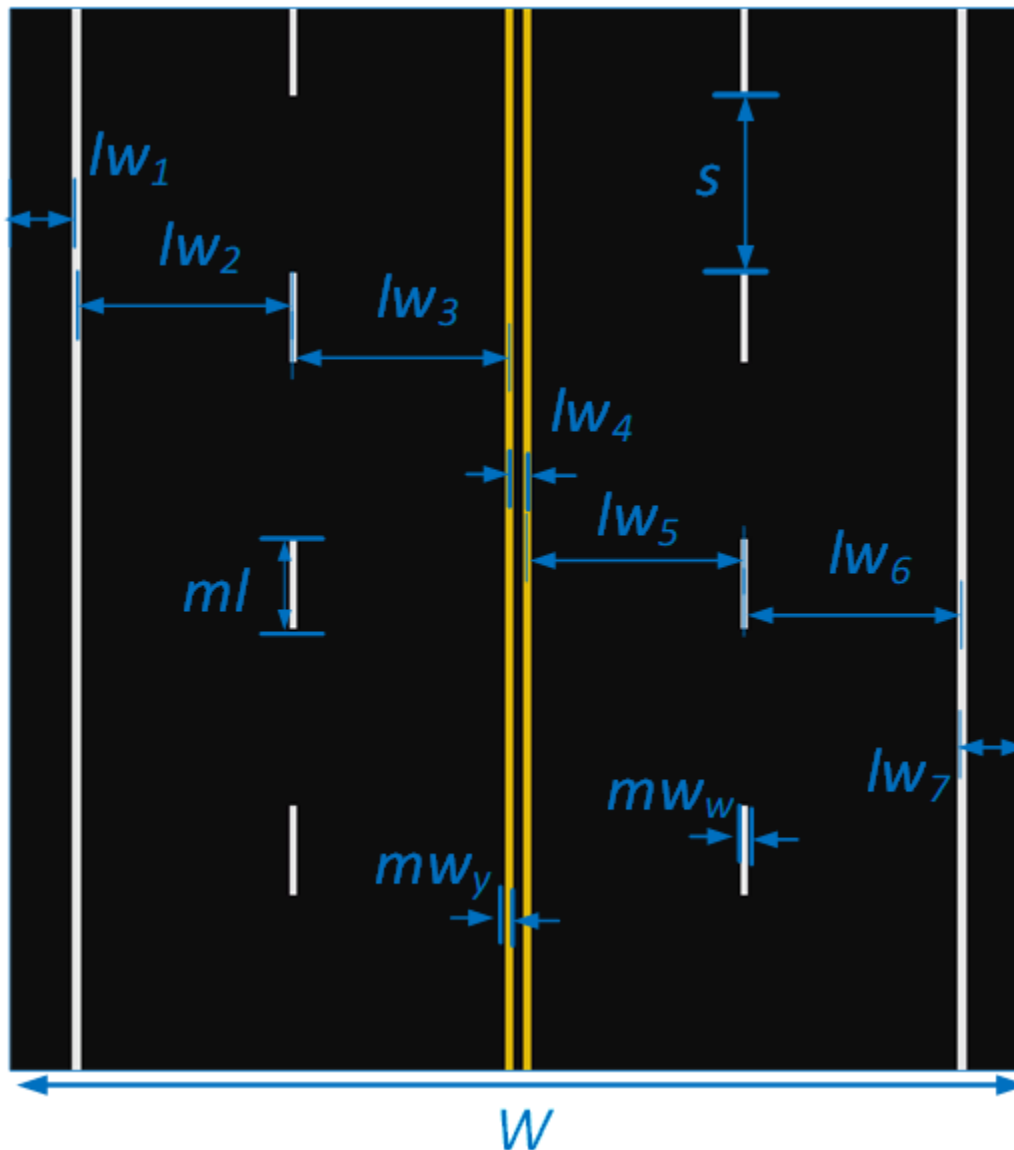
Variable	Dimension (m)
lw_1	1.33
lw_2	4.25
lw_3	4.25
lw_4	0.34
lw_5	4.25
lw_6	4.25
lw_7	1.33

Variable	Dimension (m)
ml	1.5
s	2.0
mw_w	0.13
mw_y	0.17
W	20.0

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.

Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about X-axis
Y	Extends to the right of the vehicle, parallel to the ground plane Pitch — Right-handed rotation about Y-axis
Z	Extends upwards Yaw — Left-handed rotation about Z-axis

Tips

- If you have the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, then you can modify this scene. In the Unreal Engine project file that comes with the support package, this scene is named `DblLnChng`.

  For more details on customizing scenes, see "Support Package for Customizing Scenes".

See Also
Curved Road | **Large Parking Lot** | **Open Surface** | **Parking Lot** | Simulation 3D Scene Configuration | **Straight Road** | **US City Block** | **US Highway** | **Virtual Mcity**

Topics
"Send and Receive Double-Lane Change Scene Data"
"3D Visualization Engine Requirements"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

US City Block

US city block 3D environment

Description

The **US City Block** scene is a 3D environment of a US city block that contains 15 intersections and 30 traffic lights. The scene is rendered using the Unreal Engine from Epic Games.

To simulate a driving maneuver in this scene:

1. Add a Simulation 3D Scene Configuration block to your Simulink model.
2. In this block, set the **Scene source** parameter to `Default Scenes`.
3. Set the enabled **Scene name** parameter to `US city block`.

Scene Layout

The scene uses the world coordinate system to locate objects.
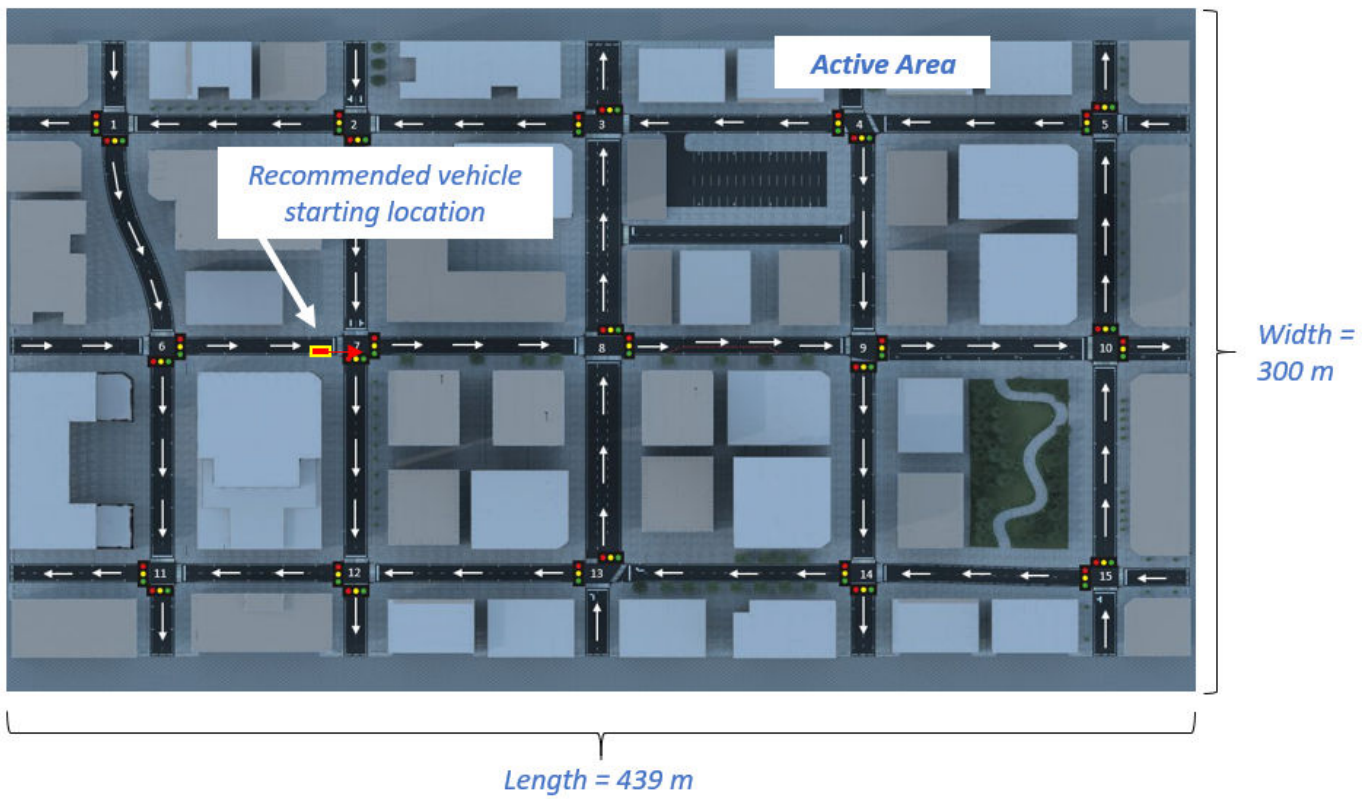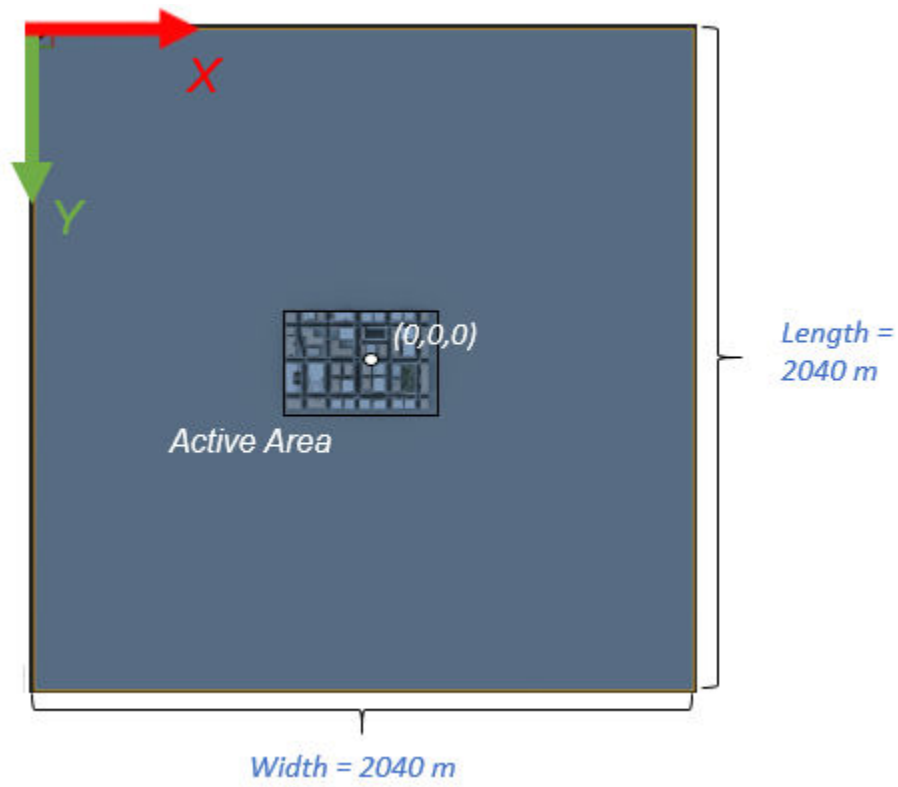
Length =
2040 m

Width = 2040 m

Active Area

Recommended vehicle
starting location

Width =
300 m

Length = 439 m

This table provides the scene area corner locations in the world coordinate system. Dimensions are in m.

Locations	X (m)	Y (m)	Z (m)
Scene — Top left	-1020	-1020	0
Scene — Bottom right	1020	1020	0
Active area — Bottom left	-240.77	151.67	0

Recommended Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
-125.19	1.65	0.04 *-0.04 in vehicle Z-down coordinate system*	0	0	0

Intersections

The US city block scene has 15 intersections, as indicated in this diagram.
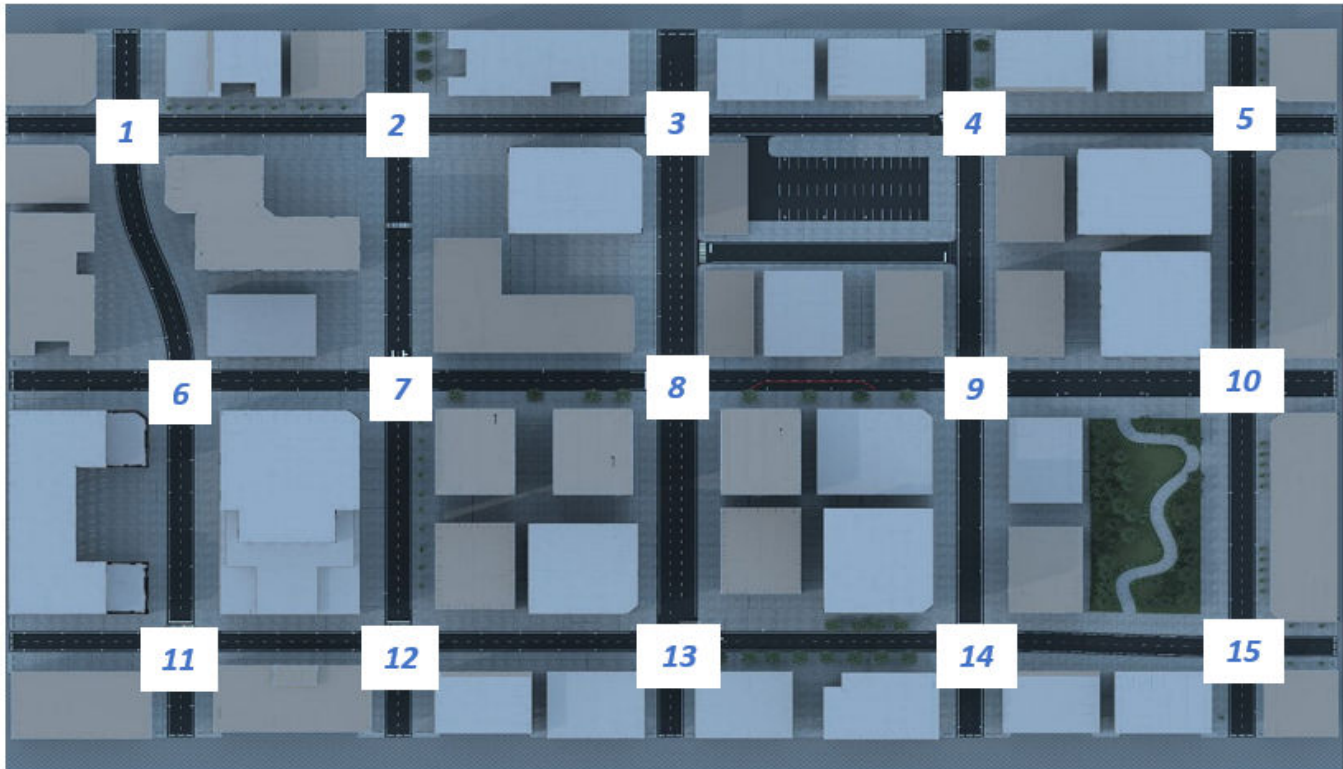
This table provides the intersection locations in the world coordinate system. Dimensions are in m.

Intersection	Center Location		
	X	Y	Z
	(m)	(m)	(m)
1	-202.60	-108	.01
2	-112.60	-108	.01
3	-20.38	-108	.01
4	74.58	-108	.01
5	166.40	-108	.01
6	-184.60	0	.01
7	-112.60	0	.01
8	-20.34	0	.01
9	76.40	0	.01
10	166.46	0	.01
11	-184.60	110.50	.01
12	-112.60	110.50	.01
13	-22.60	110.50	.01
14	76.40	110.50	.01

Intersection	Center Location		
	X	Y	Z
	(m)	(m)	(m)
15	166.40	112.50	.01

Barrier
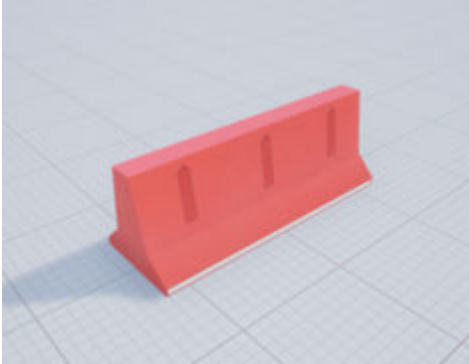
This table provides the object names and locations in the world coordinate system. Dimensions are in m.

Object	Unreal Engine Editor Name	Location					
		X	Y	Z	Roll	Pitch	Yaw
Barrier	SM_Barrier	163.5	-146.95	0	0	0	90°
	SM_Barrier 2	166.35	-146.95	0	0	0	90°
	SM_Barrier 3	169.2	-146.95	0	0	0	90°
	SM_Barrier 7	163.5	150.15	0	0	0	90°
	SM_Barrier 8	166.35	150.15	0	0	0	90°
	SM_Barrier 9	169.2	150.15	0	0	0	90°
	SM_Barrier 11	197.05	109.65	0	0	0	-180°
	SM_Barrier 13	197.05	112.5	0	0	0	-180°
	SM_Barrier 14	197.05	115.34	0	0	0	-180°
	SM_Barrier 18	197.05	-2.9	0	0	0	-180°

Object	Unreal Engine Editor Name	Location					
		X	Y	Z	Roll	Pitch	Yaw
	SM_Barrier 19	197.05	-0.05	0	0	0	-180°
	SM_Barrier 20	197.05	2.8	0	0	0	-180°
	SM_Barrier 21	-240.5	107.65	0	0	0	-180°
	SM_Barrier 22	197.05	-110.9	0	0	0	-180°
	SM_Barrier 24	197.05	5.6	0	0	0	-180°
	SM_Barrier 27	197.05	-108.05	0	0	0	-180°
	SM_Barrier 28	197.05	-105.25	0	0	0	-180°
	SM_Barrier 31	-240.5	110.5	0	0	0	-180°
	SM_Barrier 32	-240.5	113.35	0	0	0	-180°
	SM_Barrier 36	-240.1	-2.9	0	0	0	-180°
	SM_Barrier 37	-240.1	-0.05	0	0	0	-180°
	SM_Barrier 38	-240.1	2.8	0	0	0	-180°
	SM_Barrier 43	-242.15	110.9	0	0	0	-180°
	SM_Barrier 44	-242.15	-108.05	0	0	0	-180°
	SM_Barrier 45	-242.15	-105.25	0	0	0	-180°
	SM_Barrier 48	73.4	150.15	0	0	0	90°
	SM_Barrier 49	76.25	150.15	0	0	0	90°
	SM_Barrier 50	79.1	150.15	0	0	0	90°
	SM_Barrier 54	-25.55	150.15	0	0	0	90°
	SM_Barrier 55	-22.7	150.15	0	0	0	90°

Object	Unreal Engine Editor Name	Location					
		X	Y	Z	Roll	Pitch	Yaw
	SM_Barrier 56	-19.85	150.15	0	0	0	90°
	SM_Barrier 59	-115.3	150.15	0	0	0	90°
	SM_Barrier 60	-112.45	150.15	0	0	0	90°
	SM_Barrier 61	-109.6	150.15	0	0	0	90°
	SM_Barrier 66	69.25	-147.35	0	0	0	90°
	SM_Barrier 68	75.45	-147.5	0.15	0	0	90°
	SM_Barrier 69	72.45	-147.5	0.15	0	0	90°
	SM_Barrier 70	-25.55	-146.45	0	0	0	90°
	SM_Barrier 71	-22.15	-146.45	0	0	0	90°
	SM_Barrier 72	-18.65	-146.45	0	0	0	90°
	SM_Barrier 75	-115.3	-147.6	0	0	0	90°
	SM_Barrier 76	-112.45	-147.6	0	0	0	90°
	SM_Barrier 77	-109.6	-147.6	0	0	0	90°
	SM_Barrier 84	-15.45	-146.45	0	0	0	90°
	SM_Barrier 88	-187.5	150.15	0	0	0	90°
	SM_Barrier 89	-184.65	150.15	0	0	0	90°
	SM_Barrier 90	-181.8	150.15	0	0	0	90°
	SM_Barrier 94	-205.6	-147.4	0	0	0	90°
	SM_Barrier 95	-202.75	-147.4	0	0	0	90°
	SM_Barrier 96	-199.9	-147.4	0	0	0	90°

Object	Unreal Engine Editor Name	Location					
		X	Y	Z	Roll	Pitch	Yaw
	SM_Barrier 101	44.15	3.05	0	0	0	-50°
	SM_Barrier 102	39.15	0.55	0	0	0	-90°
	SM_Barrier 103	41.95	1.3	0	0	0	-50°
	SM_Barrier 104	36.5	.55	0	0	0	-90°
	SM_Barrier 105	33.85	.55	0	0	0	-90°
	SM_Barrier 106	31.2	.55	0	0	0	-90°
	SM_Barrier 107	28.45	.55	0	0	0	-90°
	SM_Barrier 108	25.8	.55	0	0	0	-90°
	SM_Barrier 109	23.15	.55	0	0	0	-90°
	SM_Barrier 110	20.5	.55	0	0	0	-90°
	SM_Barrier 111	17.95	.55	0	0	0	-90°
	SM_Barrier 112	15.3	.55	0	0	0	-90°
	SM_Barrier 113	12.65	.55	0	0	0	-90°
	SM_Barrier 114	10.0	.55	0	0	0	-90°
	SM_Barrier 115	7.01	1.38	0	0	0	-125°
	SM_Barrier 116	4.75	3.05	0	0	0	-125°

Traffic Lights

The US City Scene contains 30 traffic lights, two at each of the 15 intersections. Each intersection has a traffic light group. If you have the "Support Package for Customizing Scenes" for customizing scenes, you can control the timing of the traffic lights.

This table provides the traffic light names and locations in the world coordinate system. Dimensions are in m. Only one of the traffic lights in the group can be green at a time. The traffic lights are green for 10 s and yellow for 3 s. At the start of the simulation, the first traffic lights in the group are green (for example, SM_TrafficLights1_3 and SM_TrafficLights2_4). The second lights in the group are red (for example, SM_TrafficLights1_4 and SM_TrafficLights2_3).

Intersection	Unreal Engine Editor Name		Location					
	Traffic Light Group	Traffic Light	X	Y	Z	Roll	Pitch	Yaw
1	TrafficLightGroup	SM_TrafficLights1_3	-196.55	-100.65	0	0	0	-90°
		SM_TrafficLights1_4	-210.20	-113.40	0	0	0	0
2	TrafficLightGroup2	SM_TrafficLights2_4	-120.40	-113.50	0	0	0	0
		SM_TrafficLights2_3	-106.35	98.35	0	0	0	-90°
3	TrafficLightGroup3	SM_TrafficLights3_1	-13.10	-116.20	0.2	0	0	90°

Intersection	Unreal Engine Editor Name		Location					
	Traffic Light Group	Traffic Light	X	Y	Z	Roll	Pitch	Yaw
		SM_TrafficLights3_4	-30.60	-113.80	0	0	0	0
4	TrafficLightGroup4	SM_TrafficLights4_4	64.80	-113.0	0	0	0	0
		SM_TrafficLights4_3	71.40	-100.30	0	0	0	-100°
5	TrafficLightGroup5	SM_TrafficLights5_1	171.50	-115.70	0	0	0	90°
		SM_TrafficLights5_4	157.40	-113.50	0	0	0	0
6	TrafficLightGroup6	SM_TrafficLights6_3	-189.60	7.40	0	0	0	-90°
		SM_TrafficLights6_2	-177.30	5.70	0	0	0	180°
7	TrafficLightGroup7	SM_TrafficLights7_3	-117.80	7.70	0.2	0	0	-90°
		SM_TrafficLights7_2	-105.20	5.50	0	0	0	180°
8	TrafficLightGroup8	SM_TrafficLights8_2	-10.90	5.60	0	0	0	180°

Intersection	Unreal Engine Editor Name		Location					
	Traffic Light Group	**Traffic Light**	**X**	**Y**	**Z**	**Roll**	**Pitch**	**Yaw**
		SM_Traffic Lights8_1	-13.10	-7.60	0.1	0	0	90°
9	TrafficLightGroup9	SM_Traffic Lights9_3	70.90	9.20	0	0	0	-90°
		SM_Traffic Lights9_2	85.90	7.60	0.2	0	0	180°
10	TrafficLightGroup10	SM_Traffic Lights10_2	173.70	7.50	0	0	0	180°
		SM_Traffic Lights10_1	172.10	-7.70	0	0	0	90°
11	TrafficLightGroup11	SM_Traffic Lights11_3	-189.80	118.45	0	0	0	-90°
		SM_Traffic Lights11_4	-191.05	104.55	0	0	0	0
12	TrafficLightGroup12	SM_Traffic Lights12_4	-120.50	105.40	0	0	0	0
		SM_Traffic Lights12_3	-117.60	117.60	0	0	0	-90°
13	TrafficLightGroup13	SM_Traffic Lights13_1	-12.80	102.50	0	0	0	90°

Intersection	Unreal Engine Editor Name		Location					
	Traffic Light Group	Traffic Light	X	Y	Z	Roll	Pitch	Yaw
		SM_Traffic Lights13_4	-30.50	105.30	0	0	0	0
14	TrafficLightGroup14	SM_Traffic Lights14_4	69.30	105.30	0	0	0	0
		SM_Traffic Lights14_3	70.90	118.70	0	0	0	-90°
15	TrafficLightGroup15	SM_Traffic Lights15_1	171.40	105.20	0	0	0	90°
		SM_Traffic Lights15_4	158.40	107.20	0	0	0	0

Lane Dimensions

The scene contains three types of roads.

Road Type 1

This figure and table provides the road type 1 lane dimensions, in m.
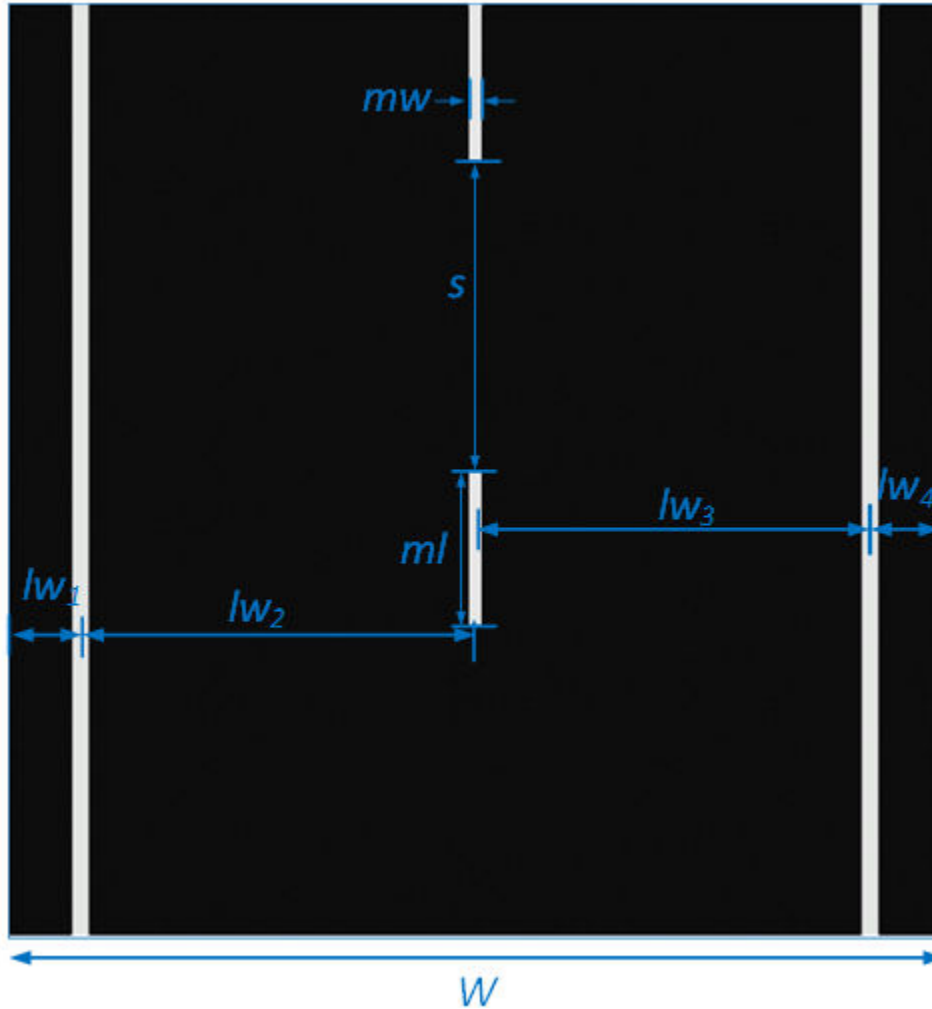
Variable	Dimension (m)
lw_1	0.65
lw_2	3.85
lw_3	3.85
lw_4	0.65
ml	1.5
s	3.0
mw	0.13
W	9

Road Type 2

This figure and table provides the road type 2 lane dimensions, in m.
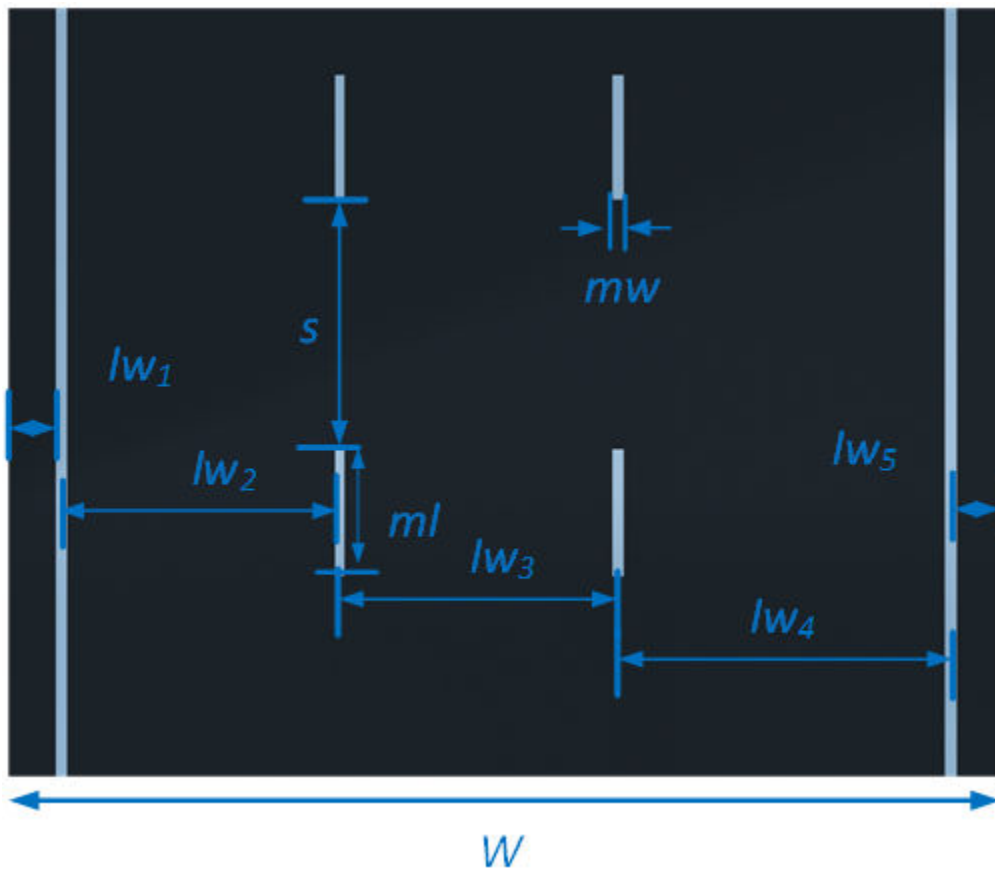
Variable	Dimension (m)
lw_1	0.73
lw_2	3.77
lw_3	3.77
lw_4	4.5
lw_5	0.73
ml	1.5
s	3.0
mw	0.13
W	13.5

Road Type 3

This figure and table provides the road type 3 lane dimensions, in m.
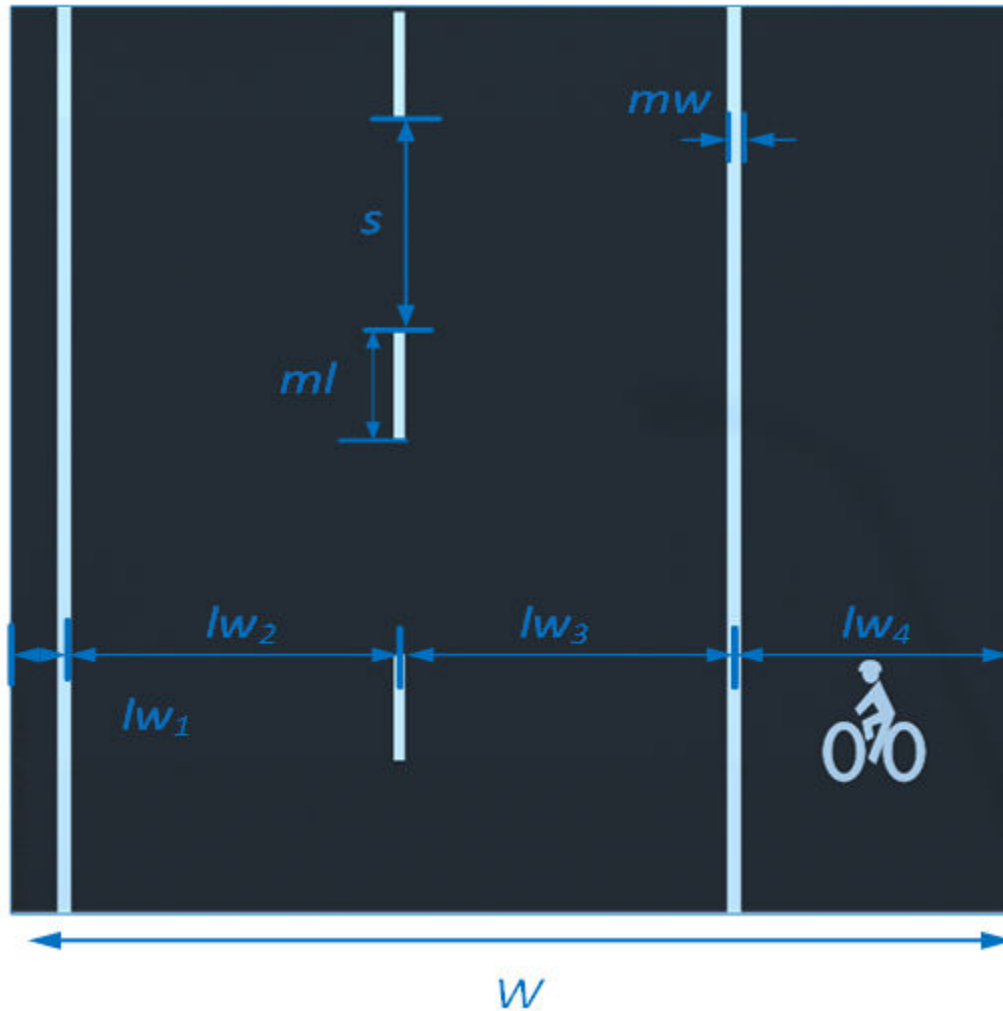
Variable	Dimension (m)
lw_1	0.65
lw_2	3.85
lw_3	3.85
lw_4	3.15
ml	1.5
s	3.0
mw	0.13
W	11.5

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.

Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about *X*-axis
Y	Extends to the right of the vehicle, parallel to the ground plane Pitch — Right-handed rotation about *Y*-axis
Z	Extends upwards Yaw — Left-handed rotation about *Z*-axis

Tips

- If you have the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, then you can modify this scene. In the Unreal Engine project file that comes with the support package, this scene is named `USCityBlock`.

  For more details on customizing scenes, see "Support Package for Customizing Scenes".

See Also
Curved Road | **Double Lane Change** | **Large Parking Lot** | **Open Surface** | **Parking Lot** | Simulation 3D Scene Configuration | **Straight Road** | **US Highway** | **Virtual Mcity**

Topics
"3D Visualization Engine Requirements"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

US Highway

US highway 3D environment

Description

The **US Highway** scene is a 3D environment of a US highway that contains barriers, cones, and traffic signs. The scene is rendered using the Unreal Engine from Epic Games.

To simulate a driving maneuver in this scene:

1   Add a Simulation 3D Scene Configuration block to your Simulink model.
2   In this block, set the **Scene source** parameter to `Default Scenes`.
3   Set the enabled **Scene name** parameter to `US highway`.

Scene Layout

The scene uses the world coordinate system to locate objects.
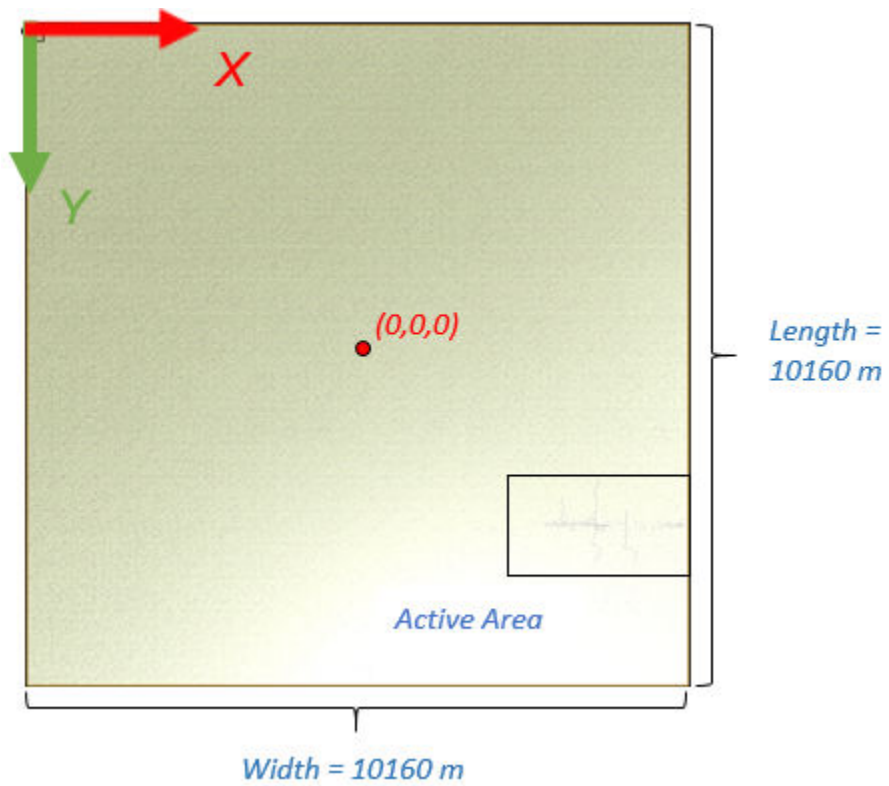
The active area of the scene contains the road.
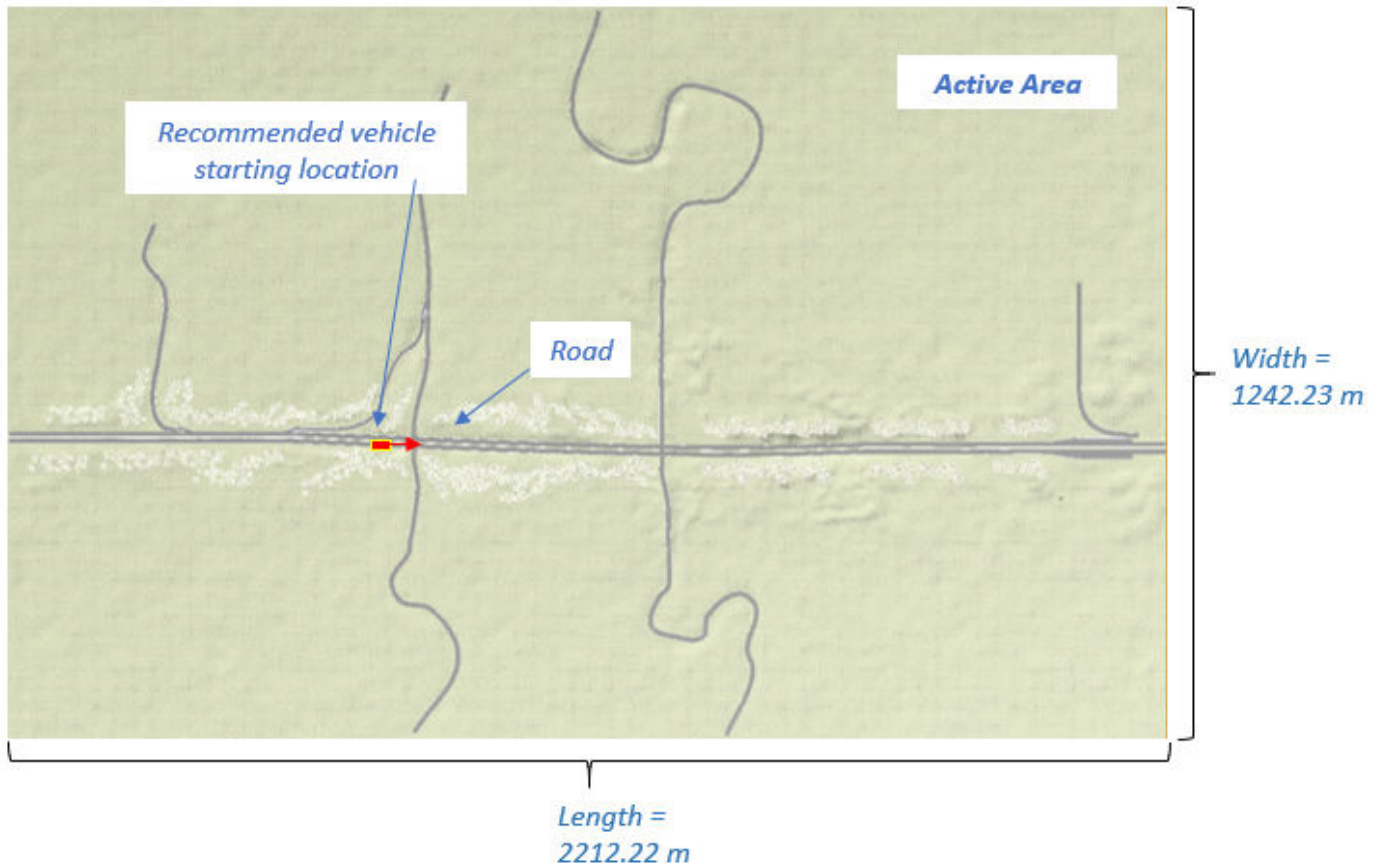
This table provides the scene area corner locations in the world coordinate system. Dimensions are in m.

Locations	X (m)	Y (m)	Z (m)
Scene — Top left	-5080	-5080	1
Scene — Bottom right	5080	5080	1
Active area — Bottom left	2867.41	3169.93	1

Recommended Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
3592.00	2617.00	1.00 *-1.00* in *vehicle Z-down coordinate system*	0	0	0

Barrier
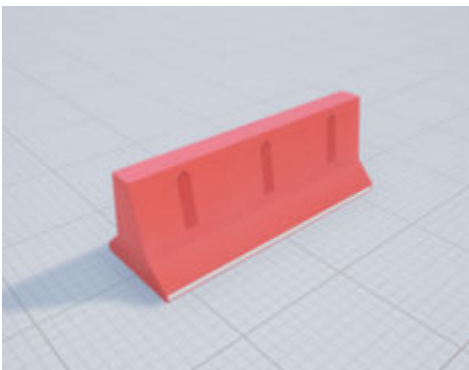
This table provides the object names and locations in the world coordinate system. Dimensions are in m.

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Barrier	SM_Barrier 26	2866.45	2612.1	1	0	0	0
	SM_Barrier 27	2866.45	2609.35	1			
	SM_Barrier 28	2866.45	2606.6	1			
	SM_Barrier 29	2866.45	2603.95	1			
	SM_Barrier 30	2866.45	2599.7	1			
	SM_Barrier 31	2866.45	2596.95	1			
	SM_Barrier 32	2866.45	2594.2	1			

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
	SM_Barrier 33	2866.45	2591.55	1			

Cones

This table provides the cone tag names and locations in the world coordinate system. Dimensions are in m.

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Cone	SM_Cone	3022.85	2599.90	1	0	0	0
	SM_Cone2	3022.85	2599.10	1			
	SM_Cone3	3022.85	2598.25	1			
	SM_Cone4	3022.85	2597.30	1			
	SM_Cone5	3022.85	2596.50	1			
	SM_Cone6	3022.85	2595.65	1			
	SM_Cone7	3022.85	2594.70	1			
	SM_Cone8	3022.85	2593.90	1			
	SM_Cone9	3022.85	2593.05	1			
	SM_Cone10	3022.85	2592.20	1			
	SM_Cone11	3022.85	2591.40	1			

Traffic Signs

This table provides the traffic sign tag names and locations in the world coordinate system. Dimensions are in m.

Object	Unreal Engine Editor Name	Location					
		X	Y	Z	Roll	Pitch	Yaw
		(m)	(m)	(m)	(deg)	(deg)	(deg)
Traffic Sign	SM_Large_Exit_Sign	3368.15	2588.20	1	0	0	-90°
	SM_Small_Exit_Sign	3232.70	2588.40	1	0	0	-90°
	ChevronAlignmentSign	3154.80	2584.50	1	0	0	-85°
	ChevronAlignmentSign2	3149.10	2579.45	1	0	0	-85°
	ChevronAlignmentSign3	3144.15	2571.95	1	0	0	-85°
	ChevronAlignmentSign4	3139.45	2562.60	1	0	0	-85°

Lane Dimensions

This figure and table provides the lane dimensions, in m.
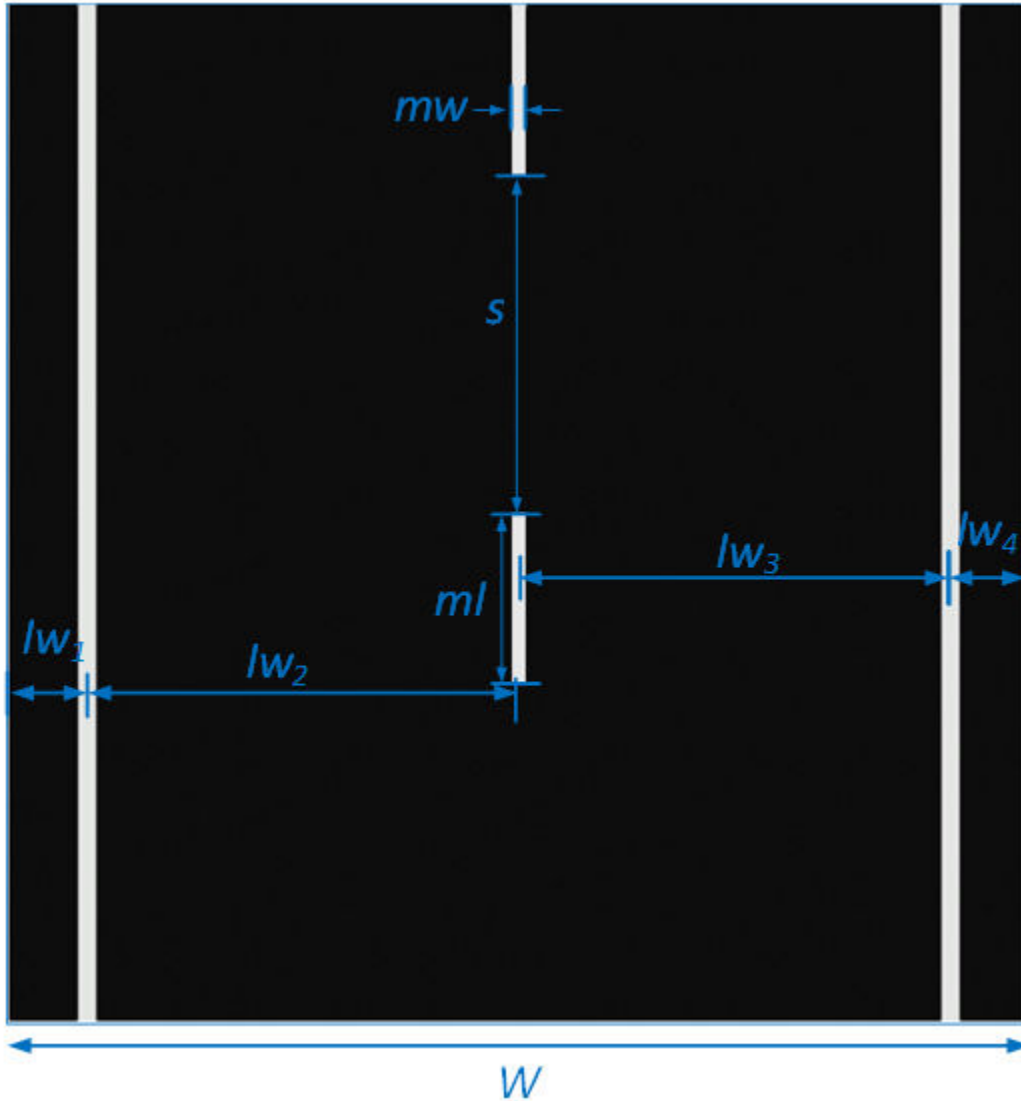
Variable	Dimension (m)
lw_1	0.65
lw_2	3.85
lw_3	3.85
lw_4	0.65
ml	1.5
s	3.0
mw	0.13
W	9.0

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.

Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about *X*-axis
Y	Extends to the right of the vehicle, parallel to the ground plane Pitch — Right-handed rotation about *Y*-axis
Z	Extends upwards Yaw — Left-handed rotation about *Z*-axis

Tips

*   If you have the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, then you can modify this scene. In the Unreal Engine project file that comes with the support package, this scene is named `USHighway`.

    For more details on customizing scenes, see "Support Package for Customizing Scenes".

See Also
Curved Road | **Double Lane Change** | **Large Parking Lot** | **Open Surface** | **Parking Lot** | Simulation 3D Scene Configuration | **Straight Road** | **US City Block** | **Virtual Mcity**

Topics
"3D Visualization Engine Requirements"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

Virtual Mcity

Virtual Mcity 3D environment

Description

The **Virtual Mcity** scene is a 3D environment containing a virtual representation of Mcity®, which is a testing ground belonging to the University of Michigan. For more details, see Mcity Test Facility.

The scene is rendered using the Unreal Engine from Epic Games.

To simulate a driving maneuver in this scene:

1 Add a Simulation 3D Scene Configuration block to your Simulink model.
2 In this block, set the **Scene source** parameter to `Default Scenes`.
3 Set the enabled **Scene name** parameter to `Virtual Mcity`.

Scene Layout

The scene uses the world coordinate system to locate objects.
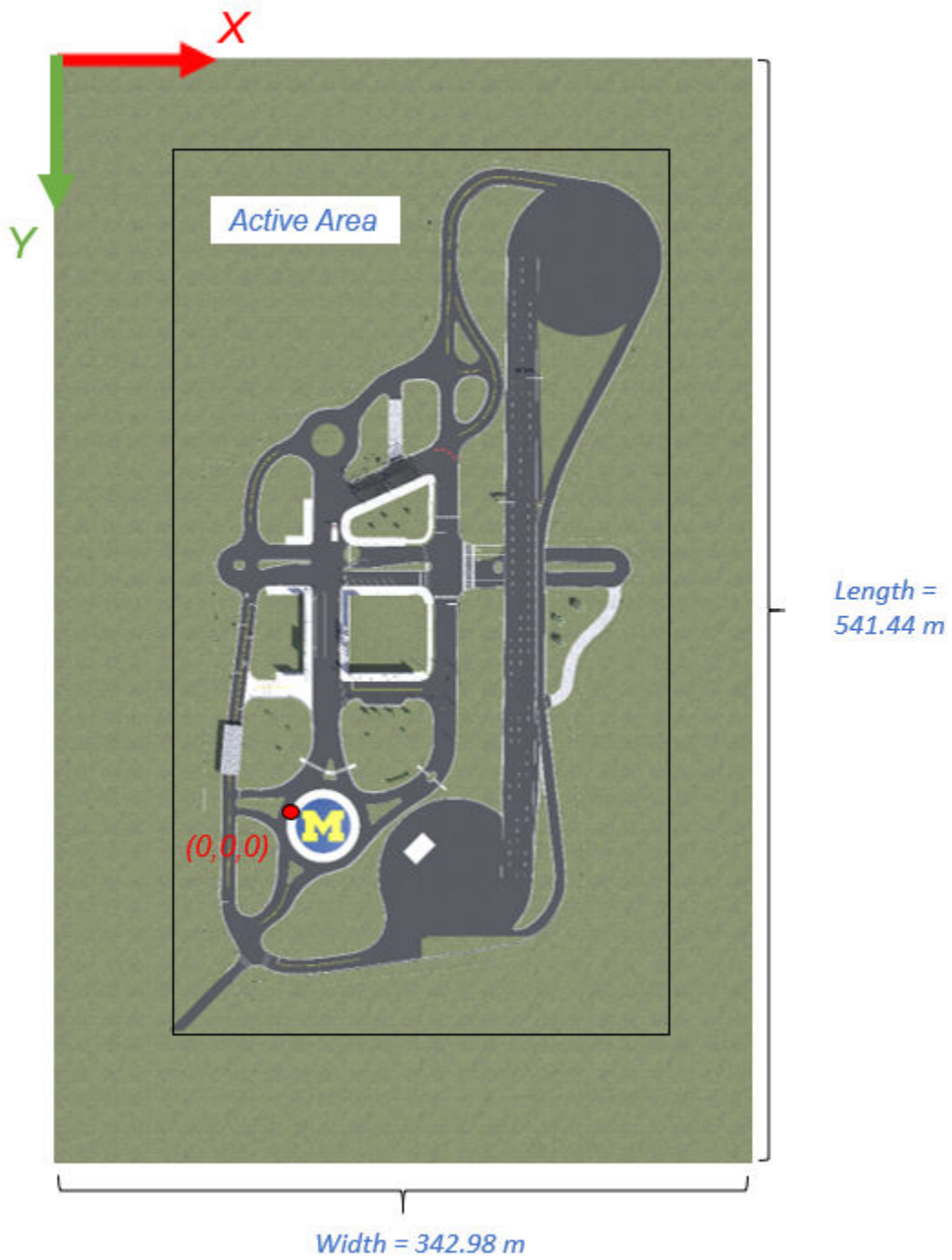
The active area of the scene contains the road.
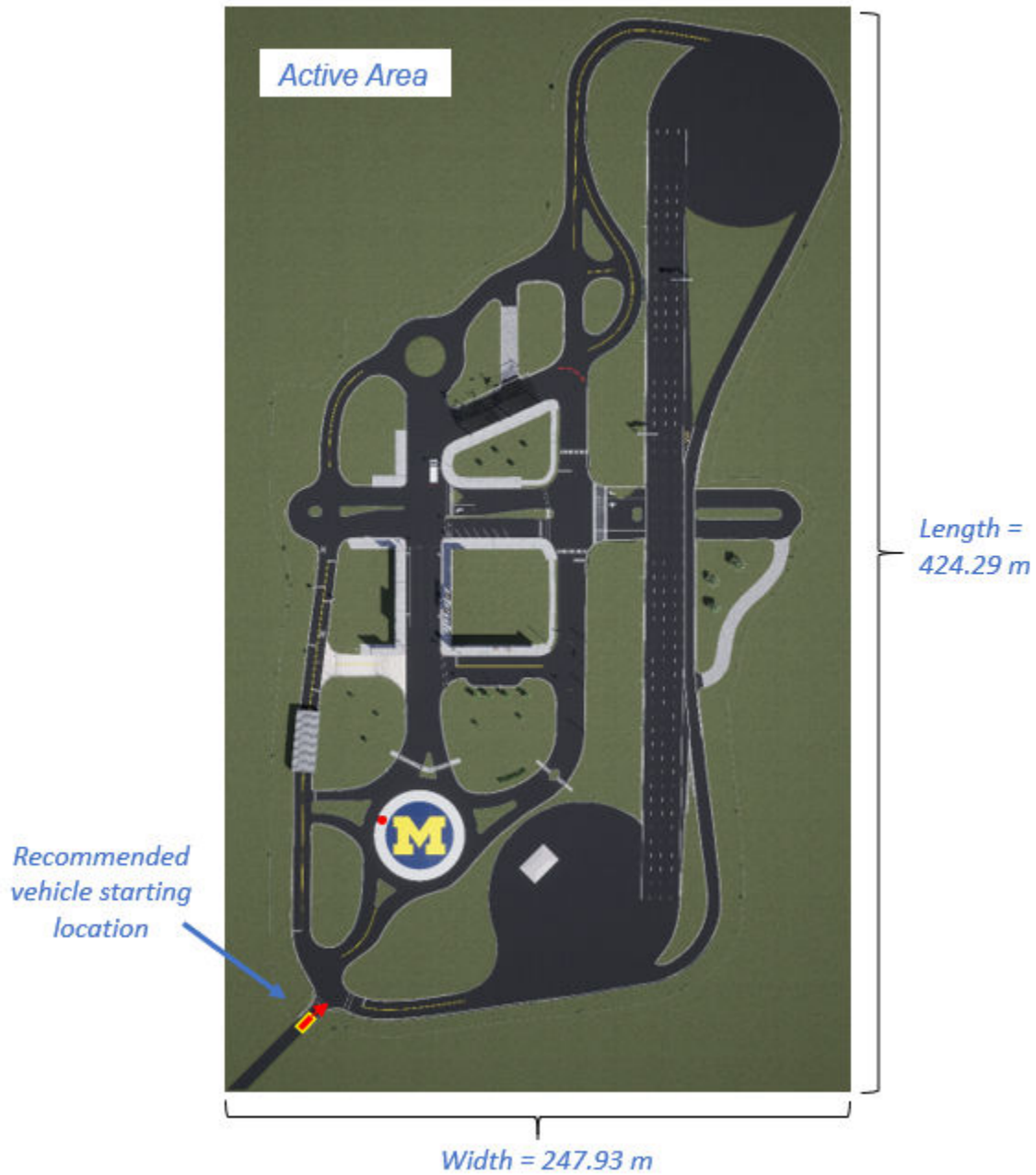
This table provides the scene area corner locations in the world coordinate system. Dimensions are in m.

Locations	X	Y	Z
	(m)	(m)	(m)
Scene — Top left	-116.85	-369.18	-.02
Scene — Bottom right	226.13	172.26	-.02
Active area — Bottom left	-60.61	106.75	-.02

Recommended Vehicle Starting Location

This table provides the recommended starting location for the vehicle in the world coordinate system. Dimensions are in m and deg.

Recommended Starting Location					
X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
-26.00	76.0	0	0	0	-40

Other Vehicles

This table provides the vehicle tag names and initial locations for other vehicles in the scene, in the world coordinate system. Dimensions are in m and deg.

Object	Unreal Engine Editor Name	Locations					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Vehicle	SK_BoxTruck	20.96	-136.90	0	0	0	-90
	SM_Motorcycle	42.50	-157.60	0	0	0	-20
	SK_SedanCar	5.83	-117.91	0	0	0	0
	SM_Bicycle	10.88	-84.42	0	0	0	90

Cones

This table provides the object names and locations in the world coordinate system. Dimensions are in m and deg.

| Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Cone	SM_Cone	22.33	-131.51	0	0	0	0
	SM_Cone2	21.23	-131.51	0	0	0	0
	SM_Cone3	20.03	-131.51	0	0	0	0
	SM_Cone4	18.93	-131.51	0	0	0	0

Barrier
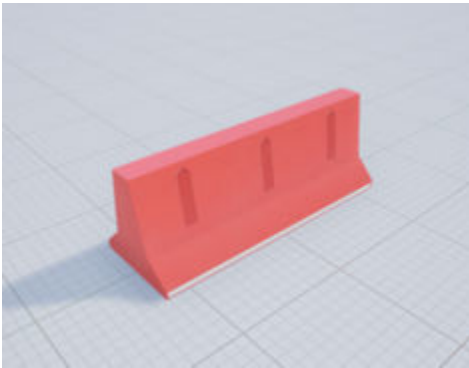
This table provides the object names and locations in the world coordinate system. Dimensions are in m and deg.

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Barrier	SM_Barrier13	79.65	-173.39	0	0	0	-35
	SM_Barrier14	77.31	-175.94	0	0	0	-55
	SM_Barrier15	74.42	-177.49	0	0	0	-80
	SM_Barrier16	71.18	-177.64	0	0	0	-95

Animals
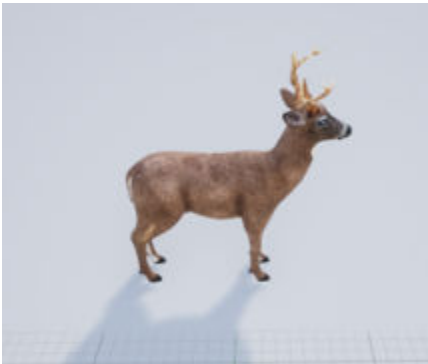
This table provides the object name and location in the world coordinate system. Dimensions are in m and deg.

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Animals	Deer	36.84	-122.15	0	0	0	0

Traffic Signs

This table provides the object names and locations in the world coordinate system. Dimensions are in m and deg.

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Traffic signs	SM_StopSign	-35.21	44.19	0	0	0	95
	SM_YellowRoadSign	-38.75	18.14	-.02	0	0	-170

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
	LargeDoubleArrowSign4	-35.19	-4.39	0	0	0	-90
	LargeDoubleArrowSign	-31.01	-60.55	0	0	0	-80
	RailroadSign2	-27.06	-88.67	0	0	0	5
	RailroadSign	-17.79	-89.77	0	0	0	-170
	SM_YieldSign	26.80	-165.14	0	0	0	0
	SM_StopSign7	54.84	-200.43	0	0	0	-90
	LargeDoubleArrowSign3	47.54	-218.00	0	0	0	-15
	SM_StopSign9	70.32	-195.66	0	0	0	0
	SM_YellowRoadSign3	82.66	-285.75	-.02	0	0	15
	SM_SpeedLimitSign2	80.89	-226.85	-.06	0	0	0
	LargeDoubleArrowSign5	104.10	-212.80	0	0	0	80
	ChevronAlignmentSign	98.45	-191.22	0	0	0	101
	ChevronAlignmentSign2	102.05	-197.62	0	0	0	76.5
	ChevronAlignmentSign3	103.98	-206.06	0	0	0	85
	SM_Large_Exit_Sign	122.45	-212.50	0	0	0	0
	SM_Large_Exit_Sign2	101.79	-151.66	0	0	0	180

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
	SM_StopSign3	32.01	-163.68	0	0	0	160
	SM_StopSign2	54.98	-177.12	0	0	0	90
	LargeSingleArrowSign	121.01	-148.56	0	0	0	0
	SM_YieldSign2	162.22	-109.64	0	0	0	25
	SM_WindingRoadSign	127.11	-50.21	.01	0	0	50
	SchoolBusOnlySign	44.03	-51.11	0	0	0	90
	SM_YellowRoadSign5	68.05	-47.03	.01	0	0	-175
	SM_CrossSignal8	74.37	-14.11	0	0	0	-165
	SM_CrossSignal7	64.69	-22.69	0	0	0	-150
	SM_CrossSignal6	62.51	-20.34	0	0	0	40
	SM_CrossSignal5	72.42	-12.06	0	0	0	40
	SM_YellowRoadSign2	60.01	-2.69	-.01	0	0	50
	SM_CrossSignal2	28.53	-20.58	0	0	0	-20
	SM_CrossSignal	21.19	-17.95	0	0	0	-20
	SM_CrossSignal3	17.55	-21.53	0	0	0	-170
	SM_CrossSignal4	6.59	-27.66	0	0	0	-145
	SM_YieldSign4	4.89	-23.42	0	0	0	-140
	SM_YellowRoadSign4	9.23	-45.63	0	0	0	-175
	SM_BikeLaneSign	24.13	-92.03	.15	0	0	0

Traffic Lights

This table provides the object names and locations in the world coordinate system. Dimensions are in m and deg.

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
Traffic lights	SM_TrafficLights	27.40	-138.55	.16	0	0	90
	SM_TrafficLights2	9.38	-106.90	.16	0	0	-90
	SM_TrafficLightsSideOnly3	8.44	-47.95	-.03	0	0	-92.2
	SM_TrafficLightsSideOnly4	1.64	-55.10	.16	0	0	-5
	SM_TrafficLightsSideOnly5	9.24	-67.70	.16	0	0	85
	SM_TrafficLightsSideOnly6	24.50	-67.82	.16	0	0	85
	SM_TrafficLights3	27.89	-109.86	.16	0	0	180
	SM_HangingTrafficLightSingle	74.43	-69.25	7.37	0	0	0
	SM_HangingTrafficLightSingle2	76.13	-69.10	7.34	0	0	0

Object	Unreal Engine Editor Name	Location					
		X (m)	Y (m)	Z (m)	Roll (deg)	Pitch (deg)	Yaw (deg)
	SM_HangingTrafficLightSingle3	82.58	-60.10	7.57	0	0	-90
	SM_HangingTrafficLightSingle4	82.65	-61.48	7.54	0	0	-90
	SM_HangingTrafficLightSingle6	73.67	-51.25	7.97	0	0	-180
	SM_HangingTrafficLightSingle7	75.07	-51.25	7.95	0	0	-180
	SM_HangingTrafficLight	-24.78	-61.49	6.71	0	0	100
	SM_RailroadCrossing4	-18.21	-86.63	.01	0	0	8
	SM_RailroadCrossing5	-26.73	-90.78	.01	0	0	-172

World Coordinate System

The 3D visualization environment uses a world coordinate system with axes that are fixed in the inertial reference frame.
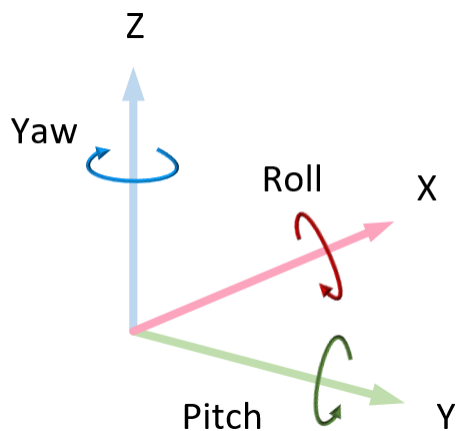
Axis	Description
X	Forward direction of the vehicle Roll — Right-handed rotation about *X*-axis
Y	Extends to the right of the vehicle, parallel to the ground plane Pitch — Right-handed rotation about *Y*-axis
Z	Extends upwards Yaw — Left-handed rotation about *Z*-axis

Limitations

* In the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package, this scene is not available for customization.

  For details on which scenes you can customize, see "Support Package for Customizing Scenes".

See Also

Curved Road | **Double Lane Change** | **Large Parking Lot** | **Open Surface** | **Parking Lot** | Simulation 3D Scene Configuration | **Straight Road** | **US City Block** | **US Highway**

Topics
"3D Visualization Engine Requirements"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"Support Package for Customizing Scenes"

External Websites
Mcity Test Facility

9

Vehicle Dimensions

Hatchback

Hatchback vehicle dimensions

Description

Hatchback is one of the vehicles that you can use within the 3D simulation environment. This environment is rendered using the Unreal Engine from Epic Games. The diagram provides the dimensions of this vehicle. The height dimensions are with respect to the vertical ground plane. The length and width dimensions are with respect to the origin of the vehicle in the vehicle coordinate system. The origin is on the ground, at the geometric center of the vehicle. For more detailed views of these diagrams, see the **Dimensions** section.

To add this type of vehicle to the 3D simulation environment:

1 Add a Simulation 3D Vehicle or Simulation 3D Vehicle with Ground Following block to your Simulink model.

2 In the block, set the **Type** parameter to `Hatchback`.

Dimensions

`Top-down view` — Vehicle width dimensions
diagram
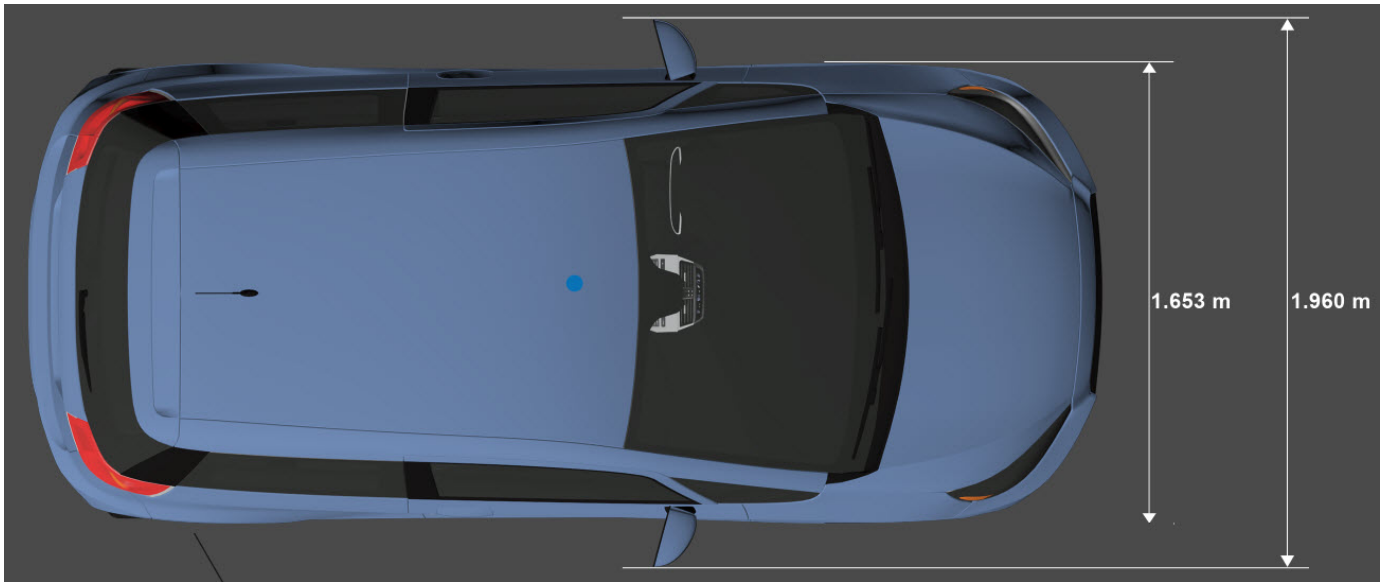
Side view — Vehicle length, front overhang, and rear overhang dimensions
diagram
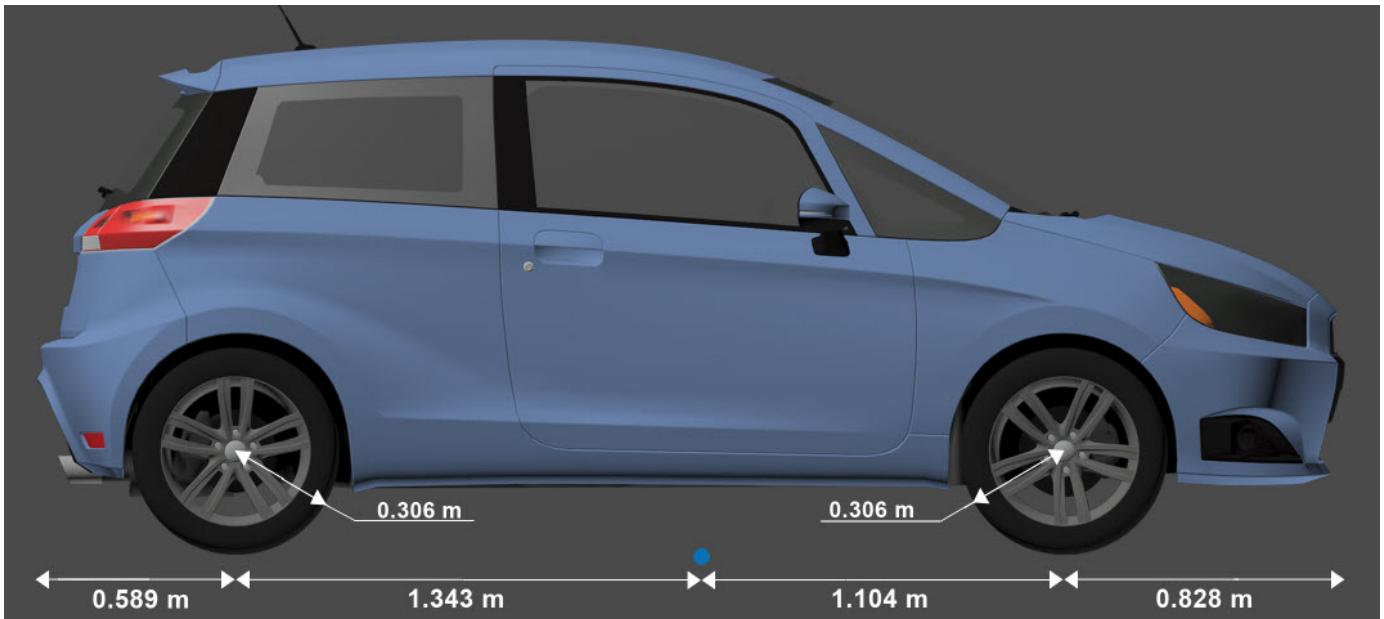
Front view — Tire width and front axle dimensions
diagram
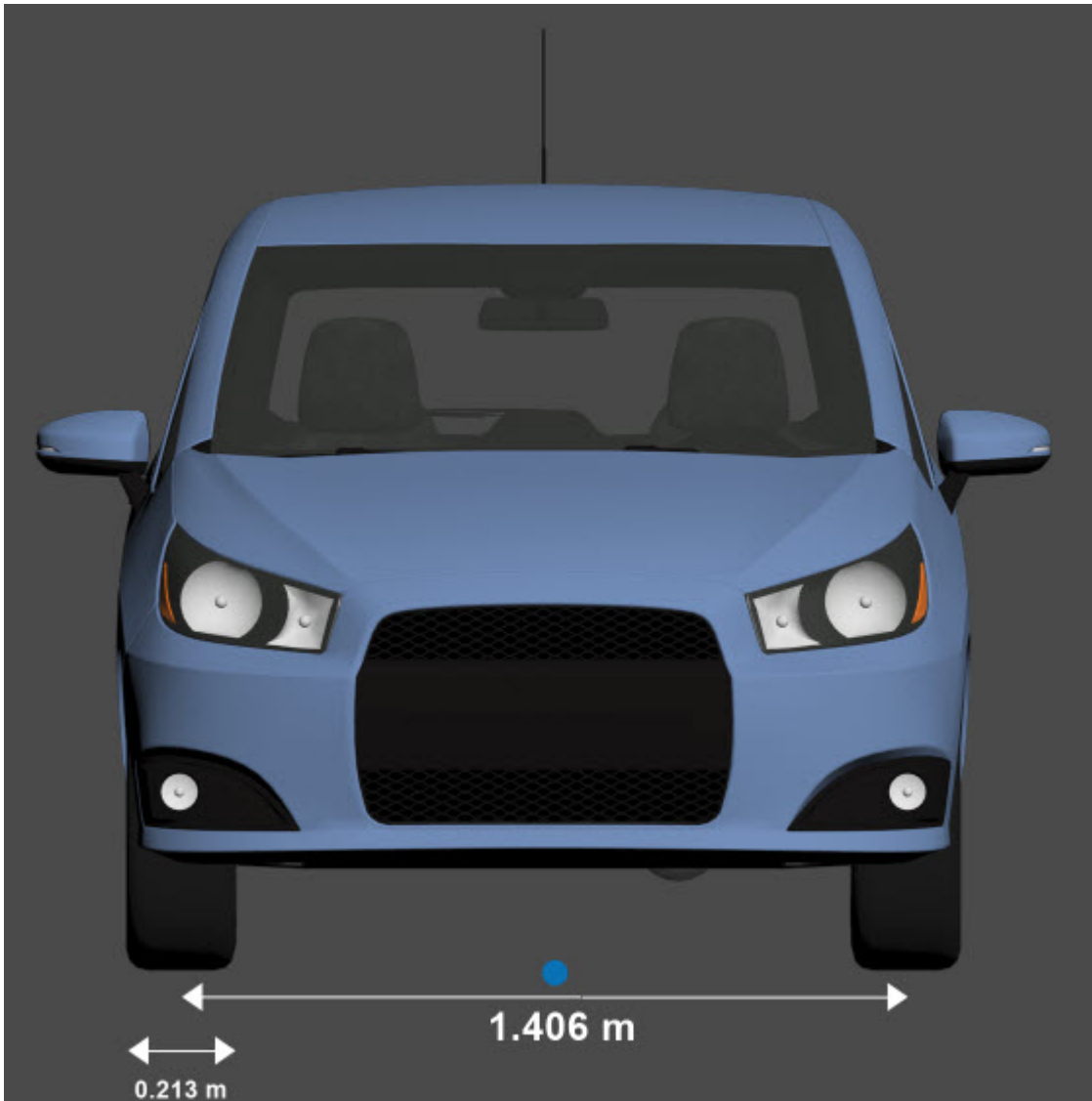
Rear view — Vehicle height and rear axle dimensions
diagram
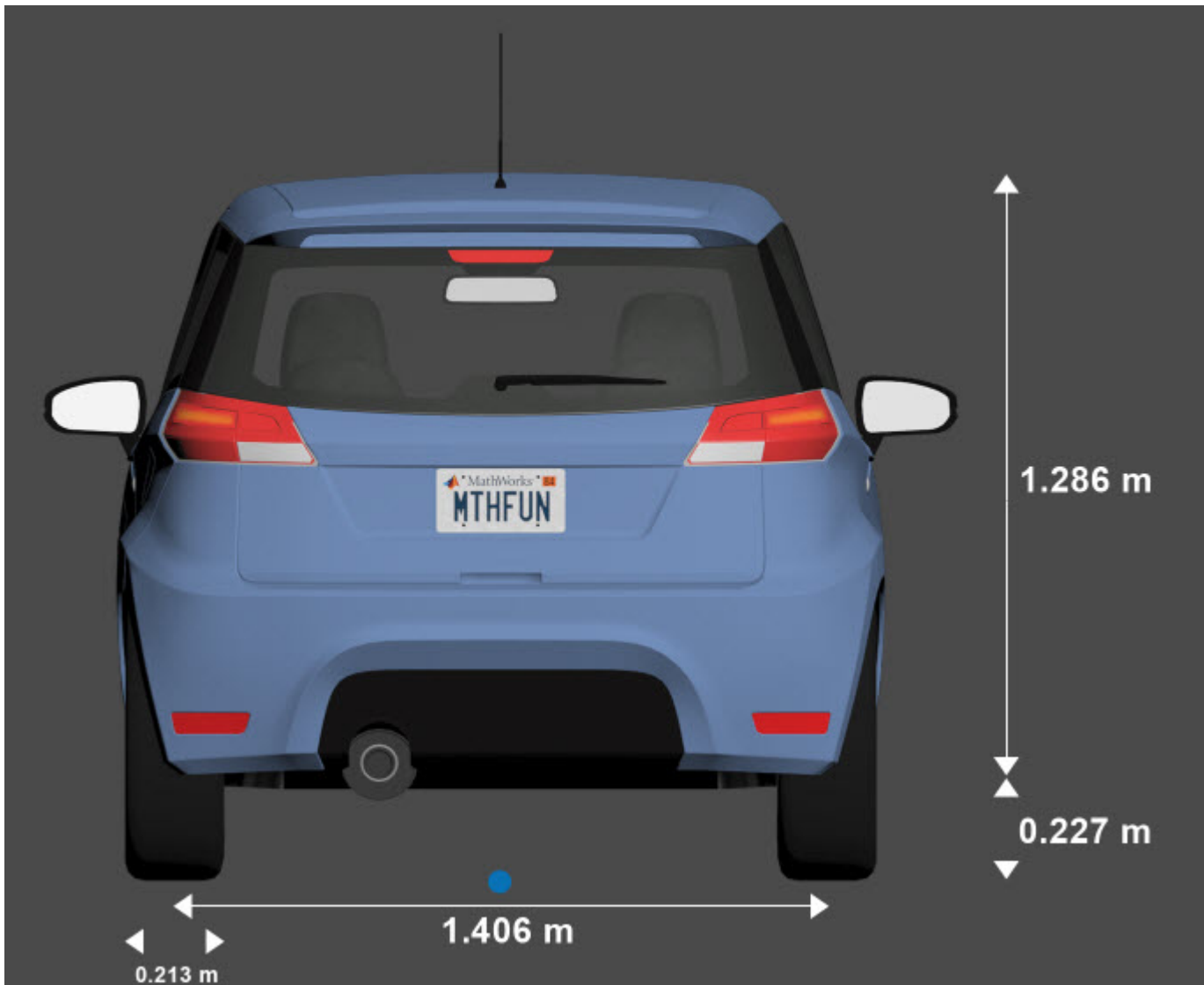
See Also

Simulation 3D Scene Configuration | Simulation 3D Vehicle | Simulation 3D Vehicle with Ground Following

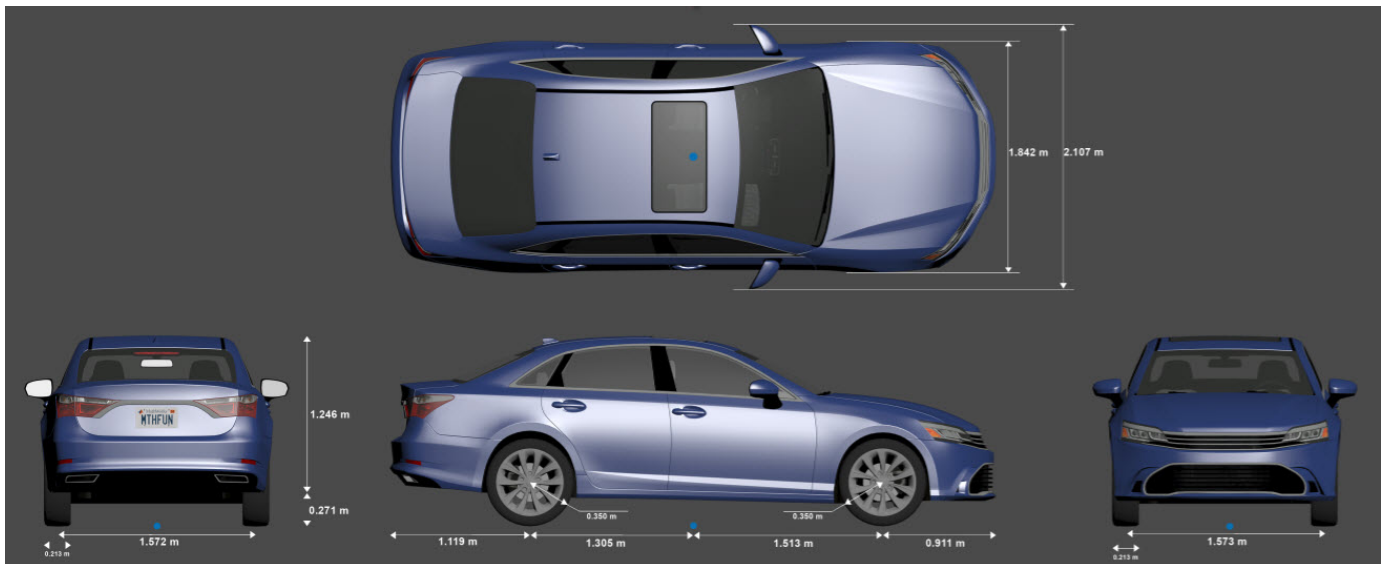**Topics**

"Coordinate Systems in Vehicle Dynamics Blockset"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"

Muscle Car

Muscle car vehicle dimensions

Description

Muscle Car is one of the vehicles that you can use within the 3D simulation environment. This environment is rendered using the Unreal Engine from Epic Games. The following diagram provides the dimensions of this vehicle. The height dimensions are with respect to the vertical ground plane. The length and width dimensions are with respect to the origin of the vehicle in the vehicle coordinate system. The origin is on the ground, at the geometric center of the vehicle. For more detailed views of these diagrams, see the **Dimensions** section.

To add this type of vehicle to the 3D simulation environment:

1   Add a Simulation 3D Vehicle or Simulation 3D Vehicle with Ground Following block to your Simulink model.

2   In the block, set the **Type** parameter to `Muscle car`.

Dimensions

`Top-down view` — Vehicle width dimensions
diagram
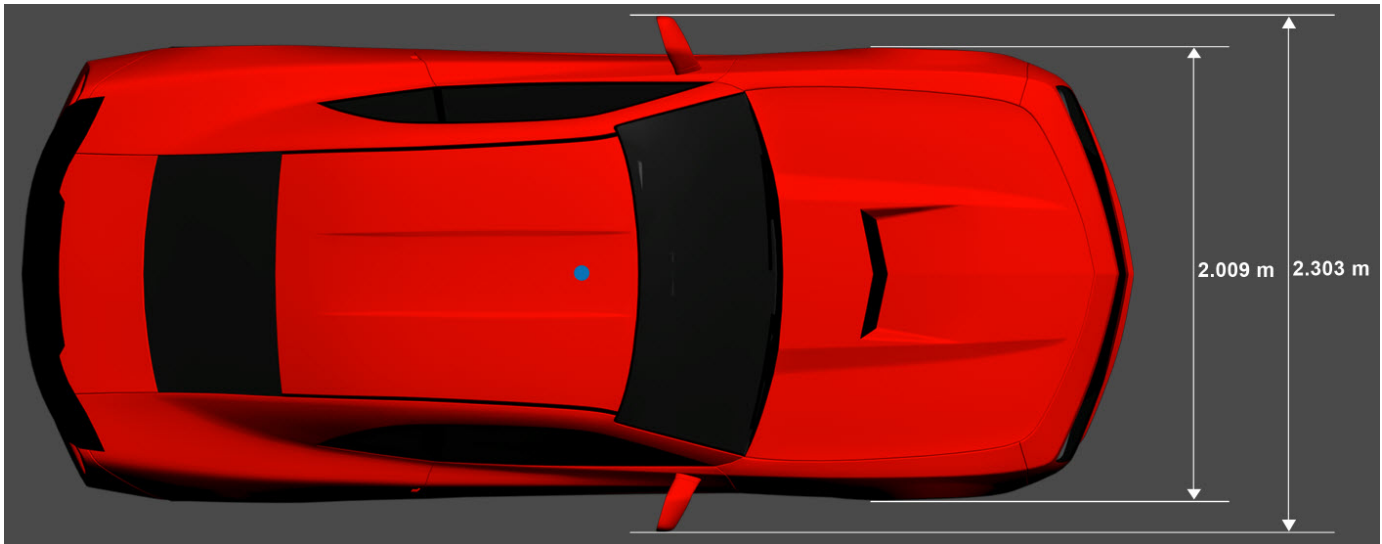
2.009 m **2.303 m**

`Side view` **— Vehicle length, front overhang, and rear overhang dimensions**
diagram

0.369 m **0.369 m**

0.945 m **1.529 m** **1.491 m** **0.983 m**
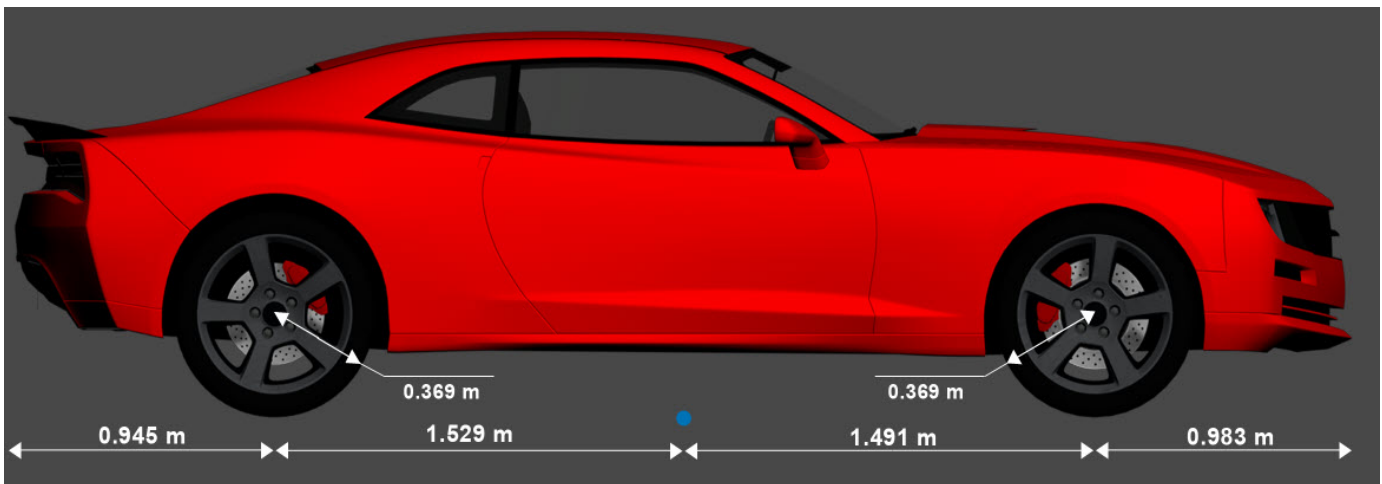
`Front view` **— Tire width and front axle dimensions**
diagram
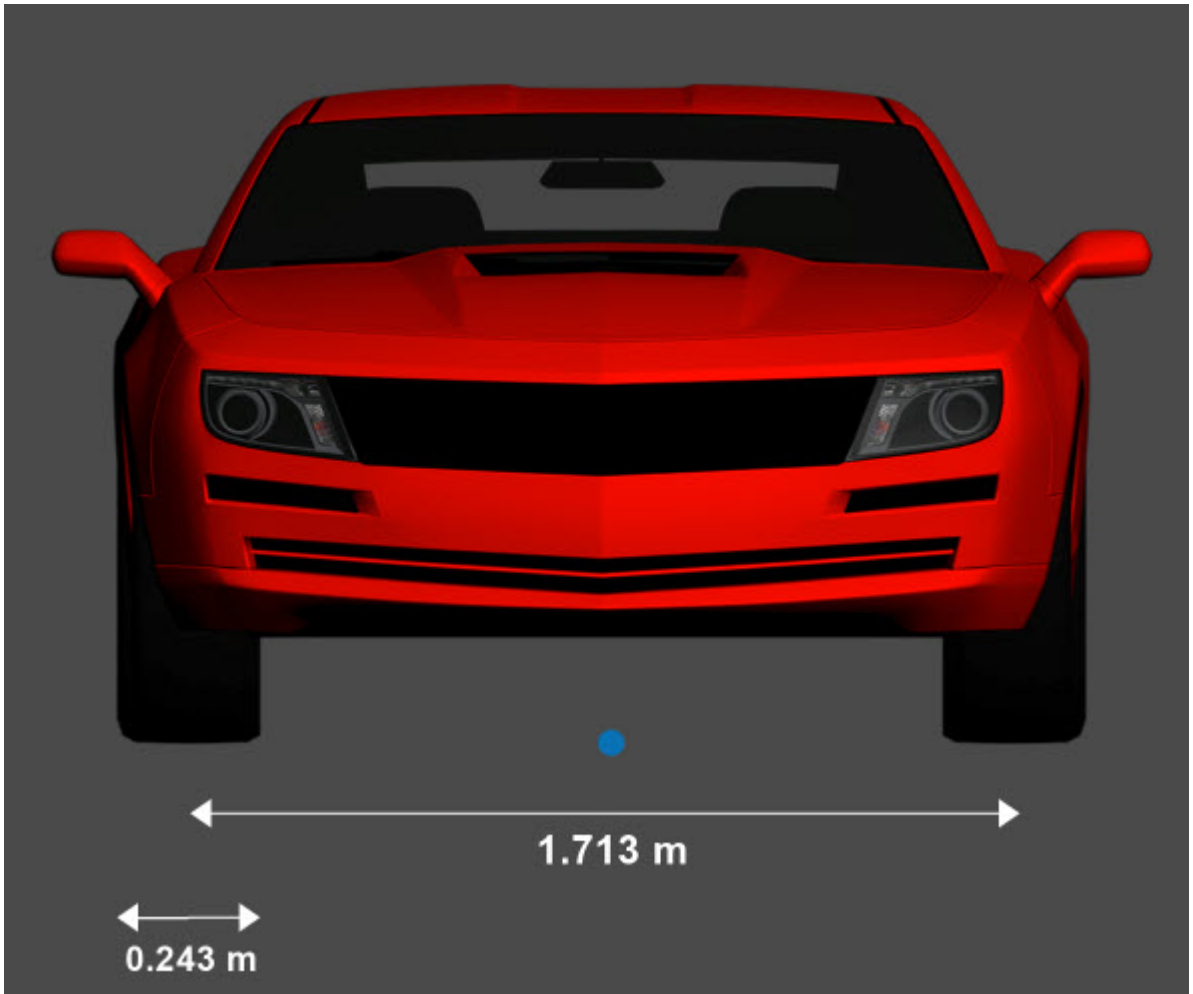
Rear view — Vehicle height and rear axle dimensions
diagram
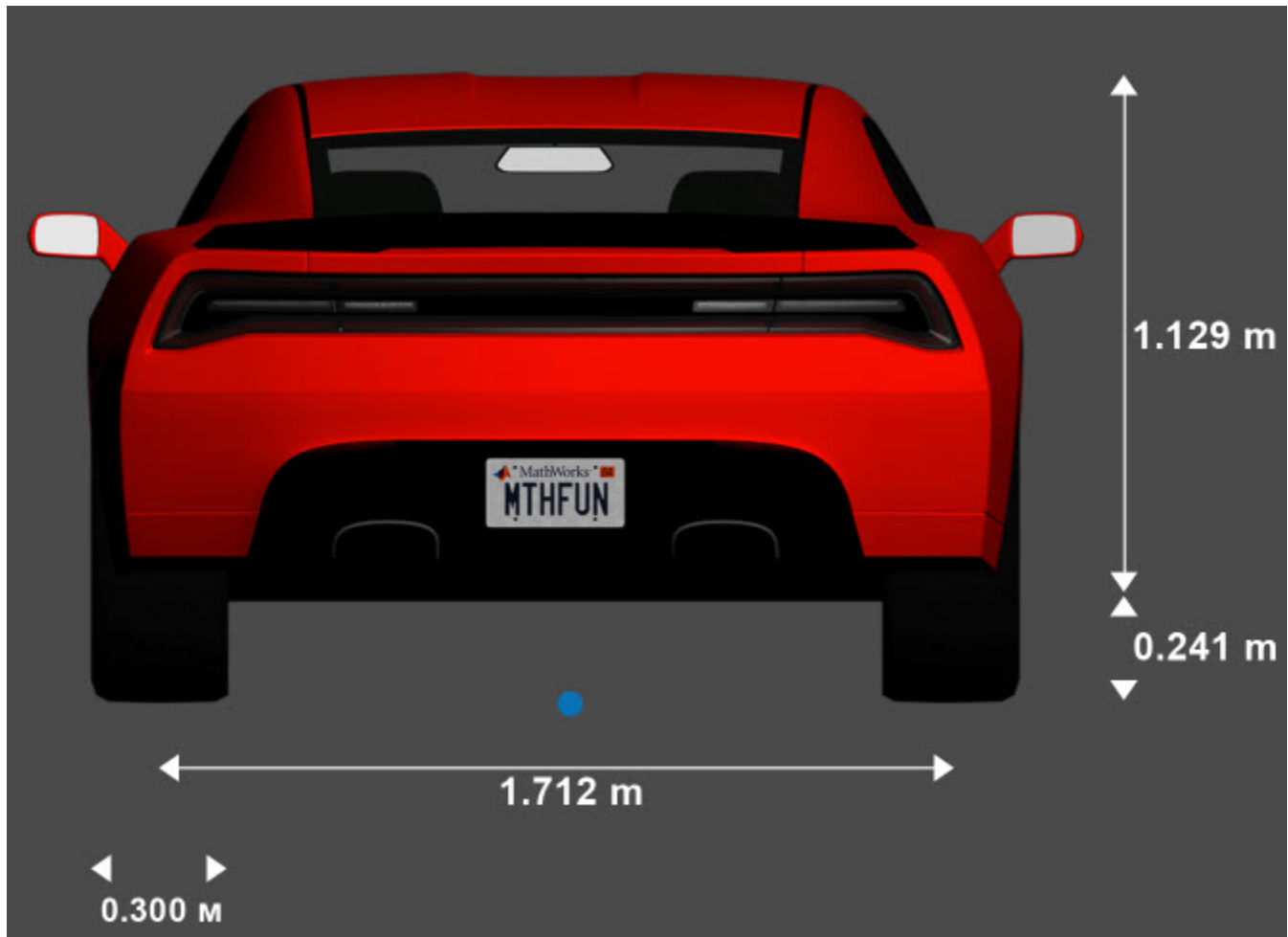
See Also

Simulation 3D Scene Configuration | Simulation 3D Vehicle | Simulation 3D Vehicle with Ground
Following

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"

Sedan

Sedan vehicle dimensions

Description

Sedan is one of the vehicles that you can use within the 3D simulation environment. This environment is rendered using the Unreal Engine from Epic Games. The diagram provides the dimensions of this vehicle. The height dimensions are with respect to the vertical ground plane. The length and width dimensions are with respect to the origin of the vehicle in the vehicle coordinate system. The origin is on the ground, at the geometric center of the vehicle. For more detailed views of these diagrams, see the **Dimensions** section.


To add this type of vehicle to the 3D simulation environment:
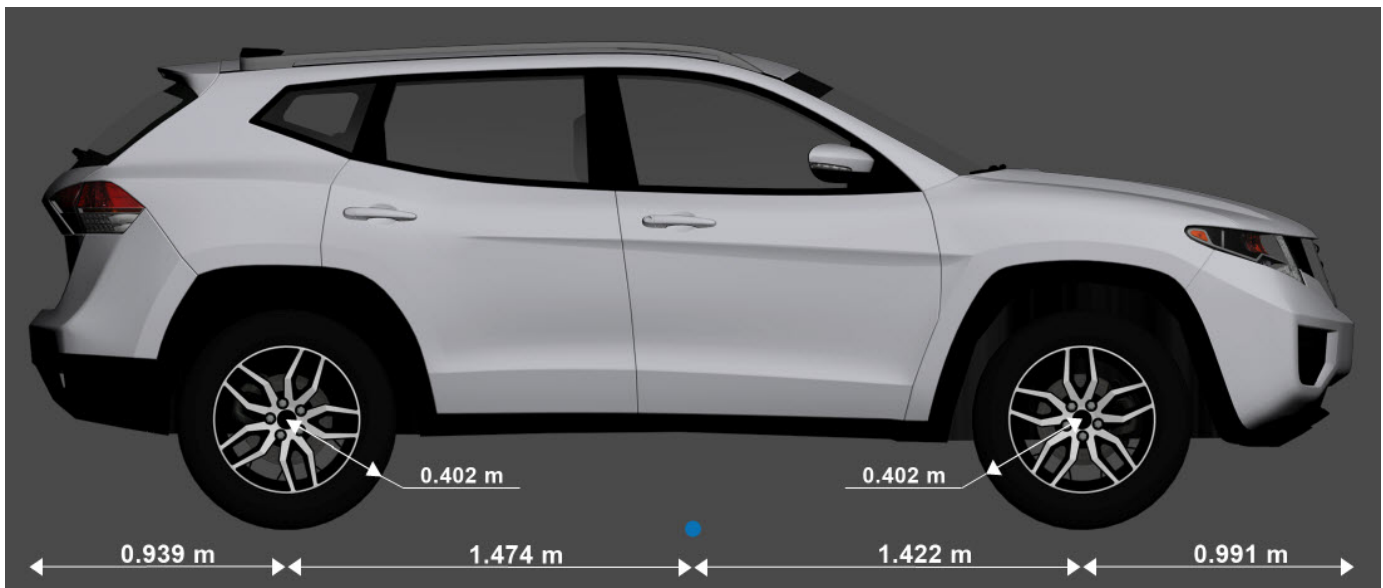
1   Add a Simulation 3D Vehicle or Simulation 3D Vehicle with Ground Following block to your Simulink model.

2   In the block, set the **Type** parameter to `Sedan`.
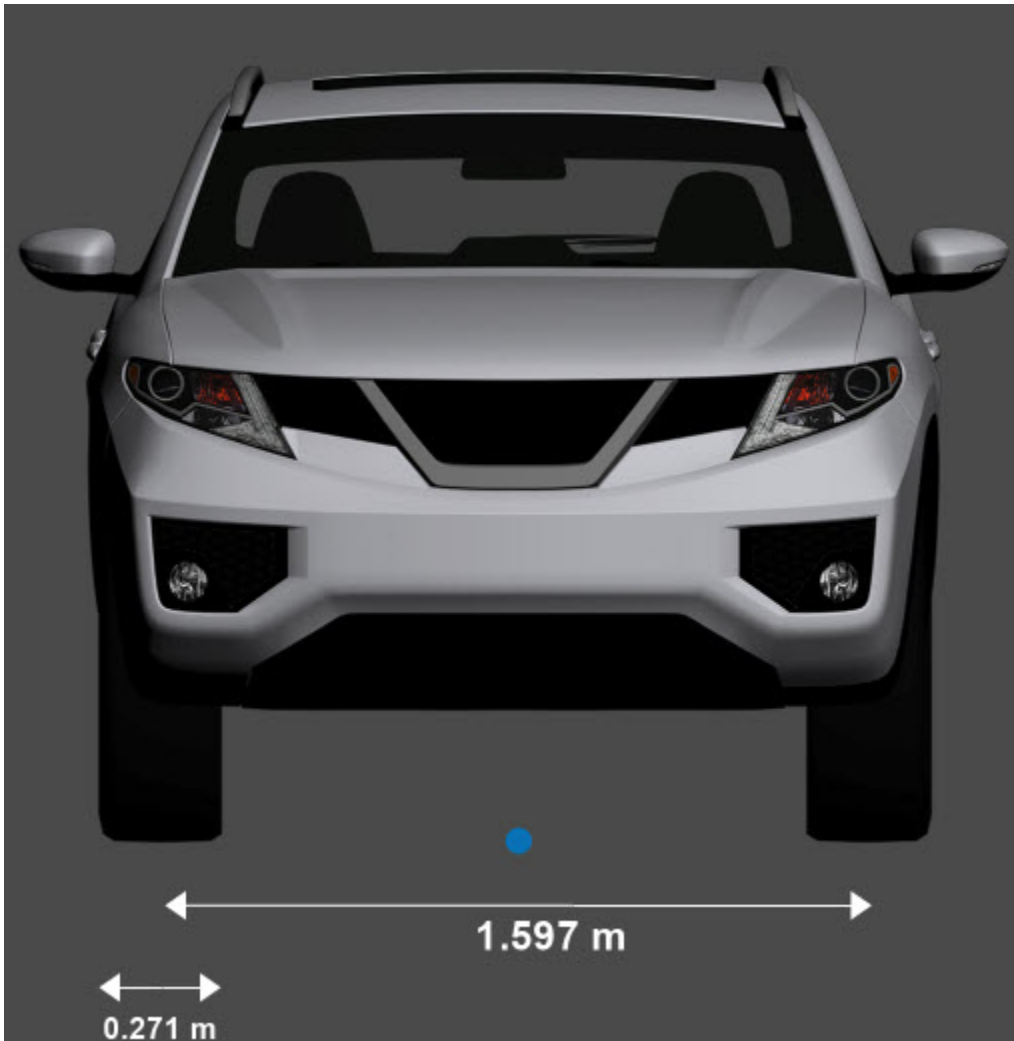
Dimensions

**`Top-down view — Vehicle width dimensions`**
diagram
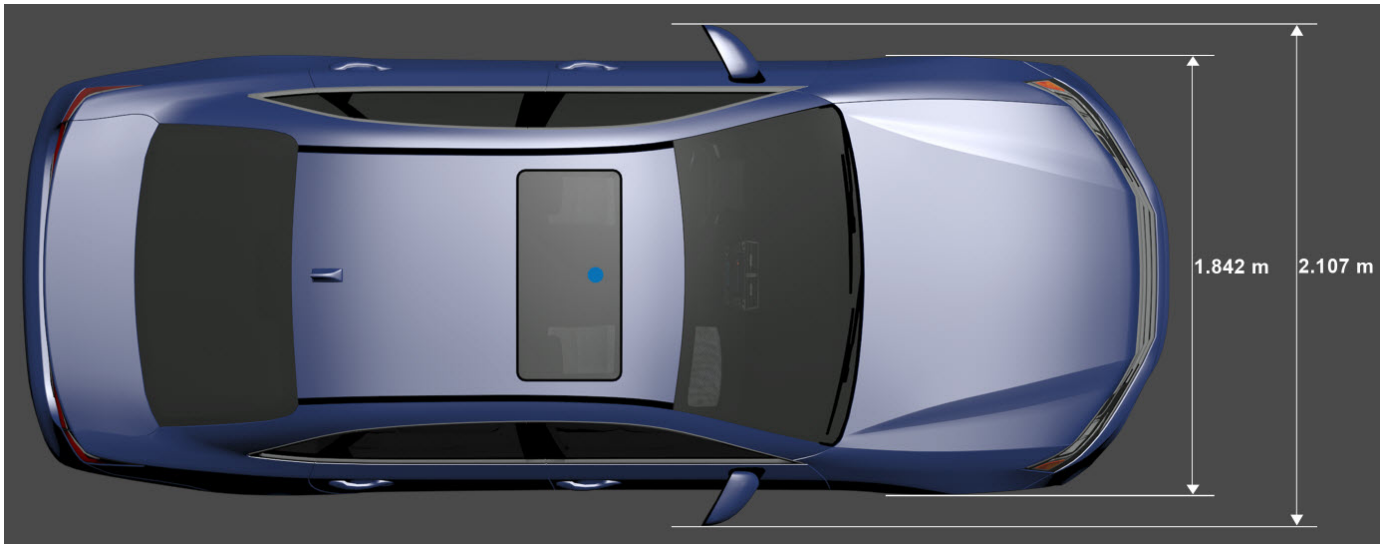
Side view — Vehicle length, front overhang, and rear overhang dimensions
diagram
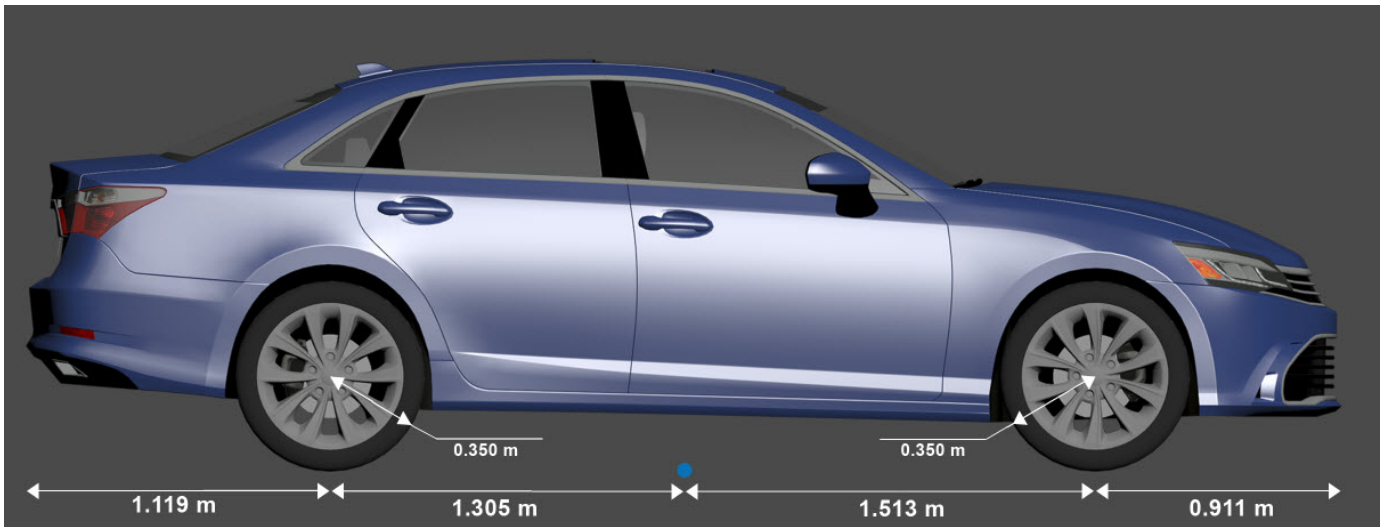
Front view — Tire width and front axle dimensions
diagram
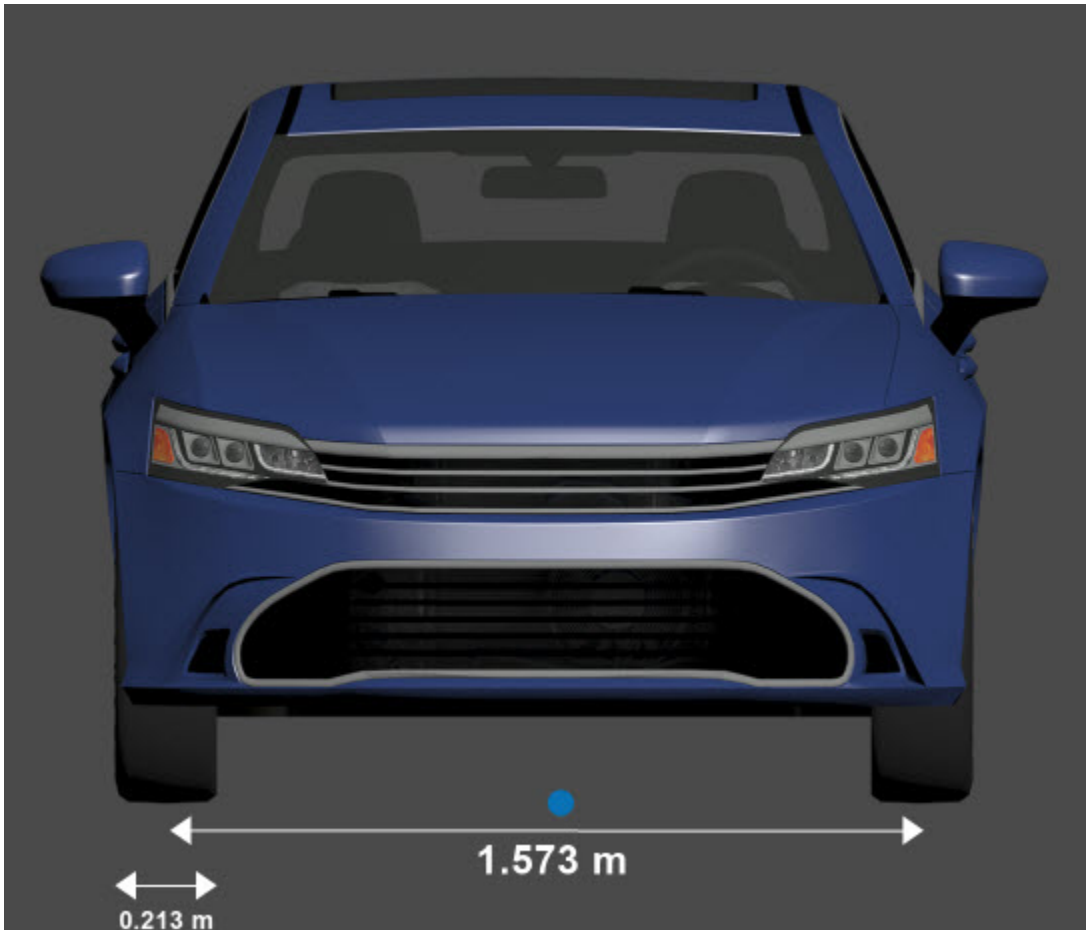
Rear view — Vehicle height and rear axle dimensions
diagram
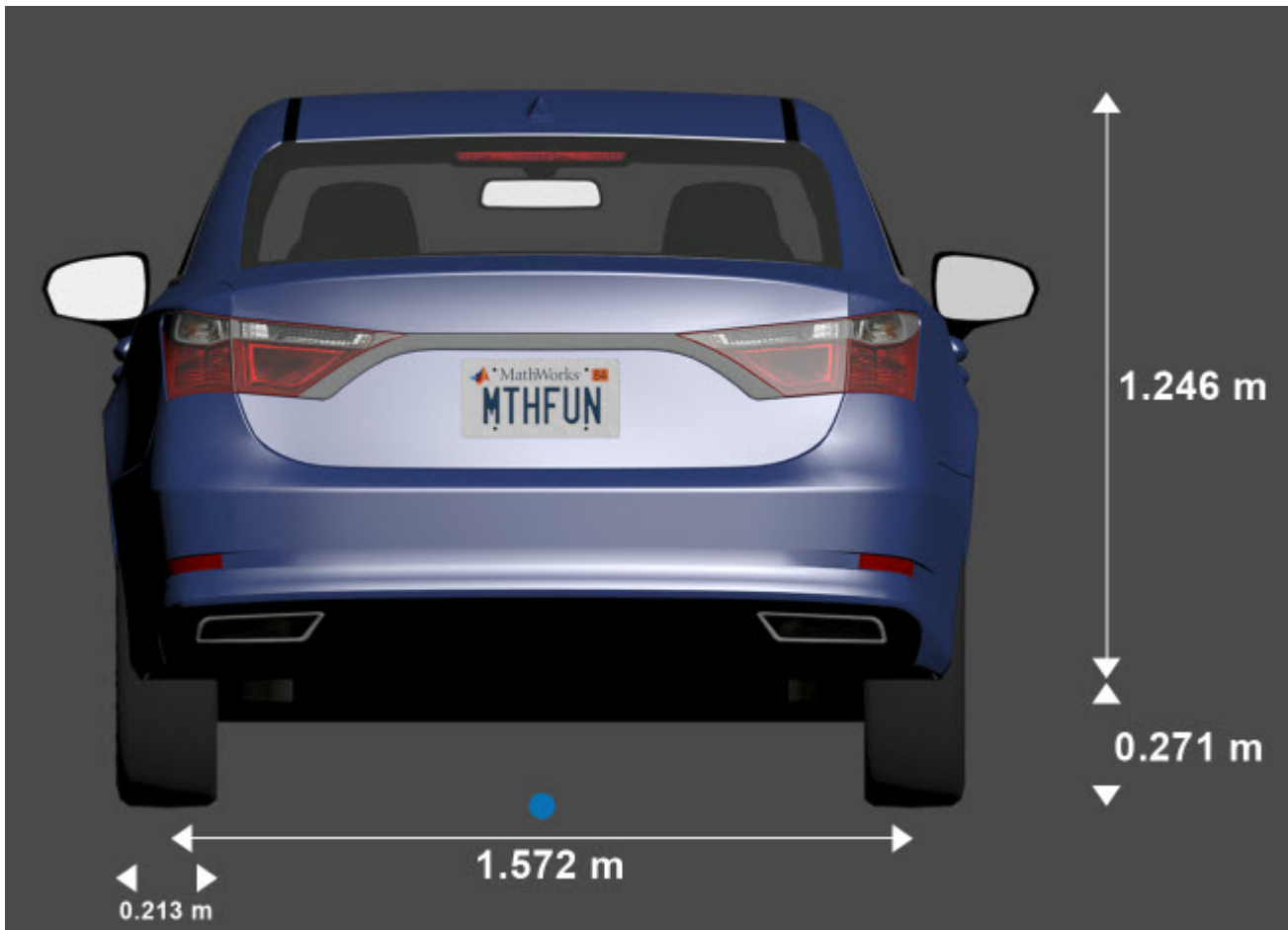
See Also

Simulation 3D Scene Configuration | Simulation 3D Vehicle | Simulation 3D Vehicle with Ground Following

Topics
"Coordinate Systems in Vehicle Dynamics Blockset"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"

Sport Utility Vehicle

Sport utility vehicle dimensions

Description

Sport Utility Vehicle is one of the vehicles that you can use within the 3D simulation environment. This environment is rendered using the Unreal Engine from Epic Games. The following diagram provides the dimensions of this vehicle. The height dimensions are with respect to the vertical ground plane. The length and width dimensions are with respect to the origin of the vehicle in the vehicle coordinate system. The origin is on the ground, at the geometric center of the vehicle. For more detailed views of these diagrams, see the **Dimensions** section.

To add this type of vehicle to the 3D simulation environment:

1 Add a Simulation 3D Vehicle or Simulation 3D Vehicle with Ground Following block to your Simulink model.

2 In the block, set the **Type** parameter to `Sport utility vehicle`.

Dimensions

`Top-down view — Vehicle width dimensions`
diagram
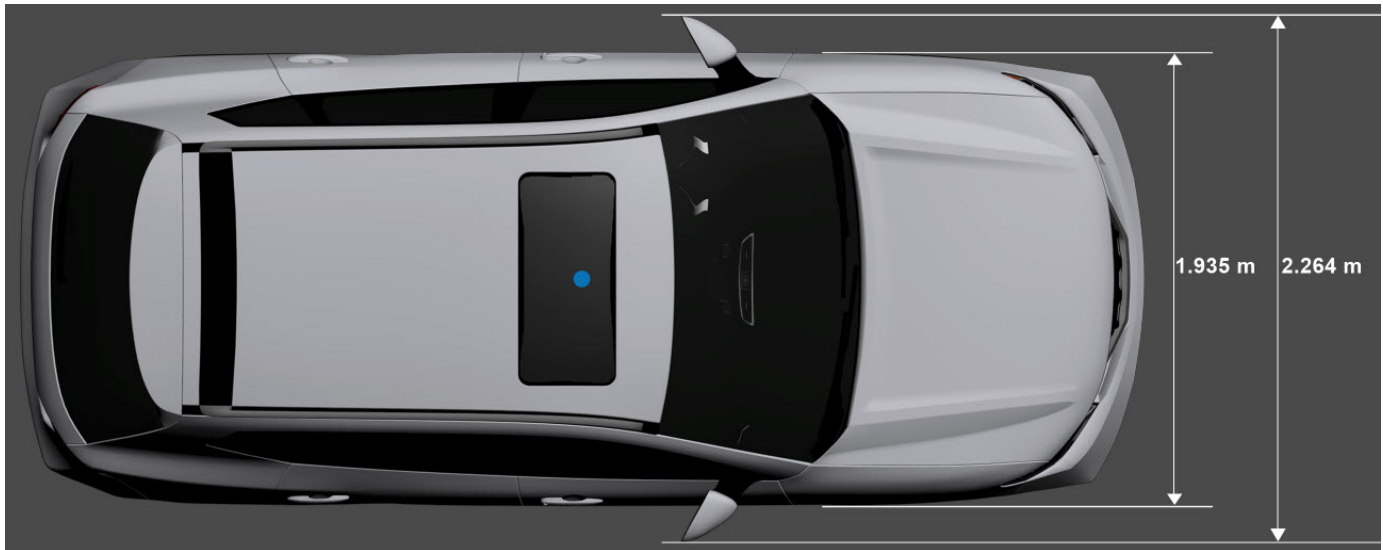
Side view — Vehicle length, front overhang, and rear overhang dimensions
diagram

Front view — Tire width and front axle dimensions
diagram

Rear view — Vehicle height and rear axle dimensions
diagram
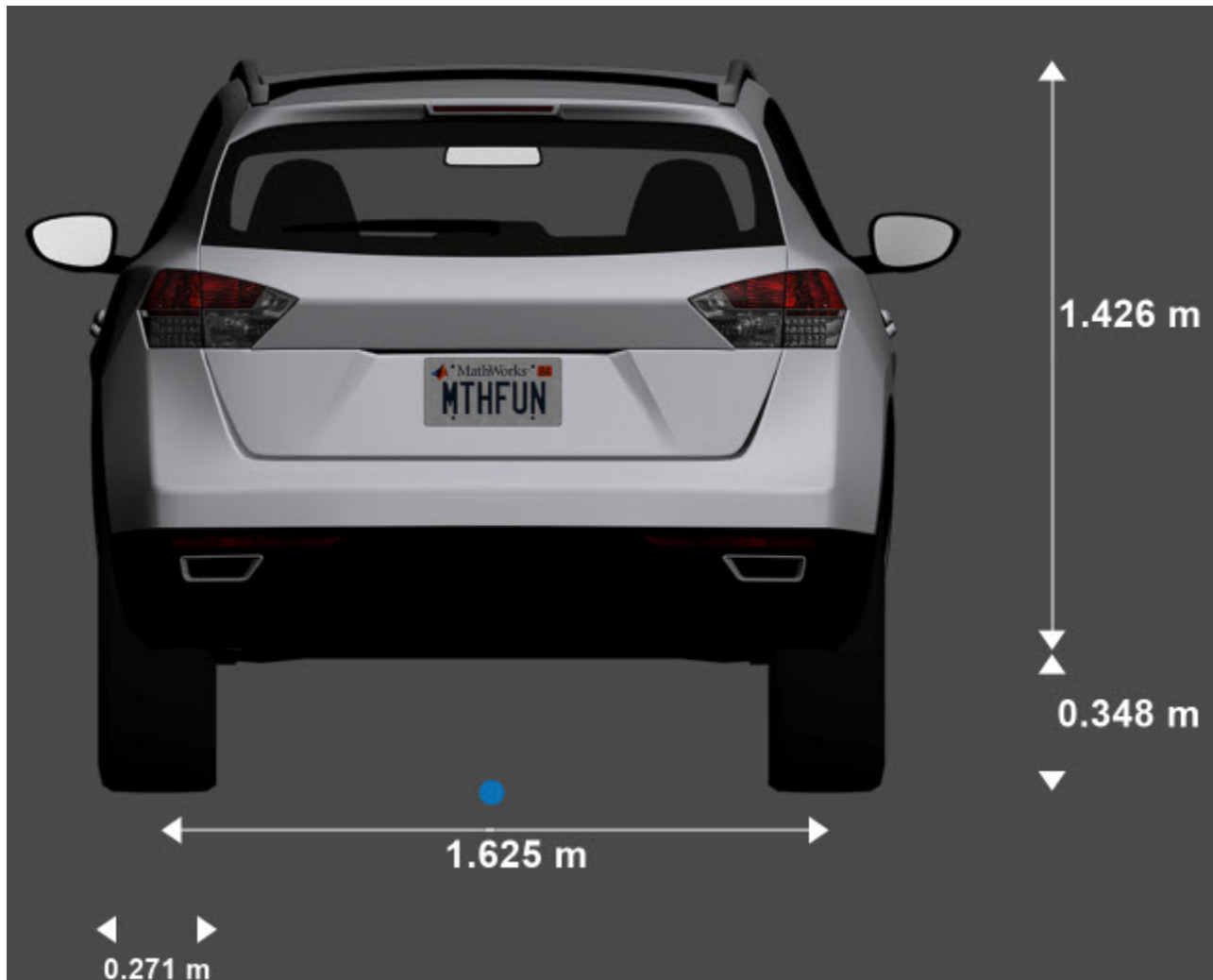
See Also

Simulation 3D Scene Configuration | Simulation 3D Vehicle | Simulation 3D Vehicle with Ground Following

Topics

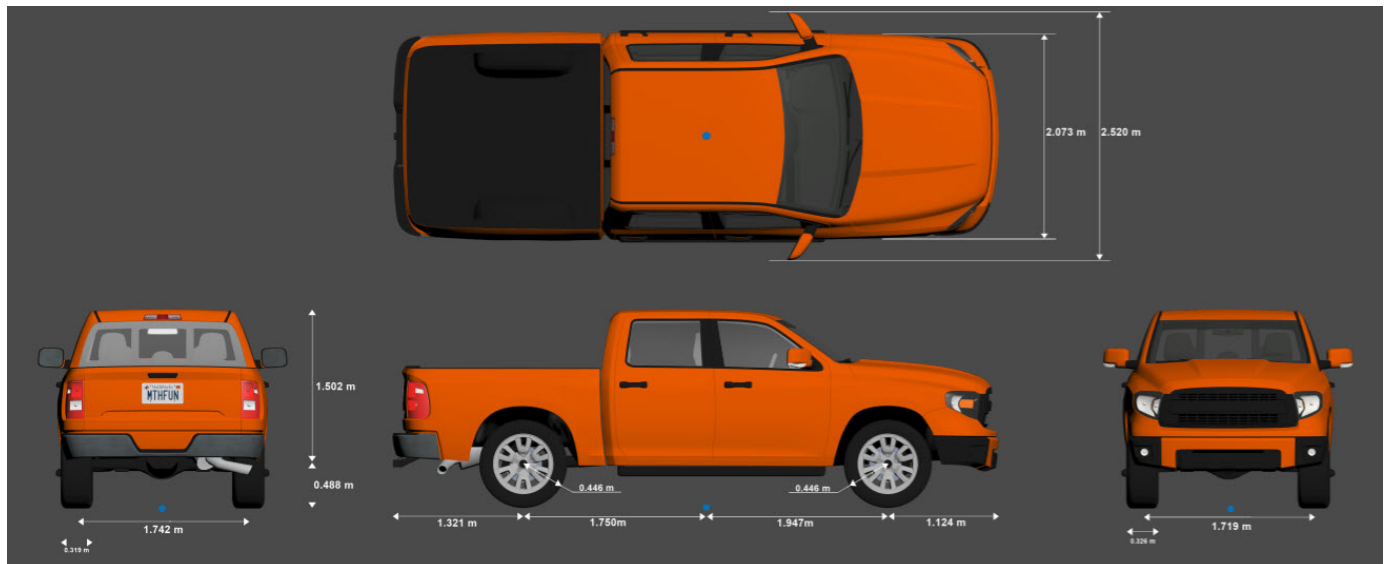"Coordinate Systems in Vehicle Dynamics Blockset"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"

Small Pickup Truck

Small pickup truck vehicle dimensions

Description

Small Pickup Truck is one of the vehicles that you can use within the 3D simulation environment. This environment is rendered using the Unreal Engine from Epic Games. The following diagram provides the dimensions of this vehicle. The height dimensions are with respect to the vertical ground plane. The length and width dimensions are with respect to the origin of the vehicle in the vehicle coordinate system. The origin is on the ground, at the geometric center of the vehicle. For more detailed views of these diagrams, see the **Dimensions** section.

To add this type of vehicle to the 3D simulation environment:

1   Add a Simulation 3D Vehicle or Simulation 3D Vehicle with Ground Following block to your Simulink model.

2   In the block, set the **Type** parameter to `Small pickup truck`.
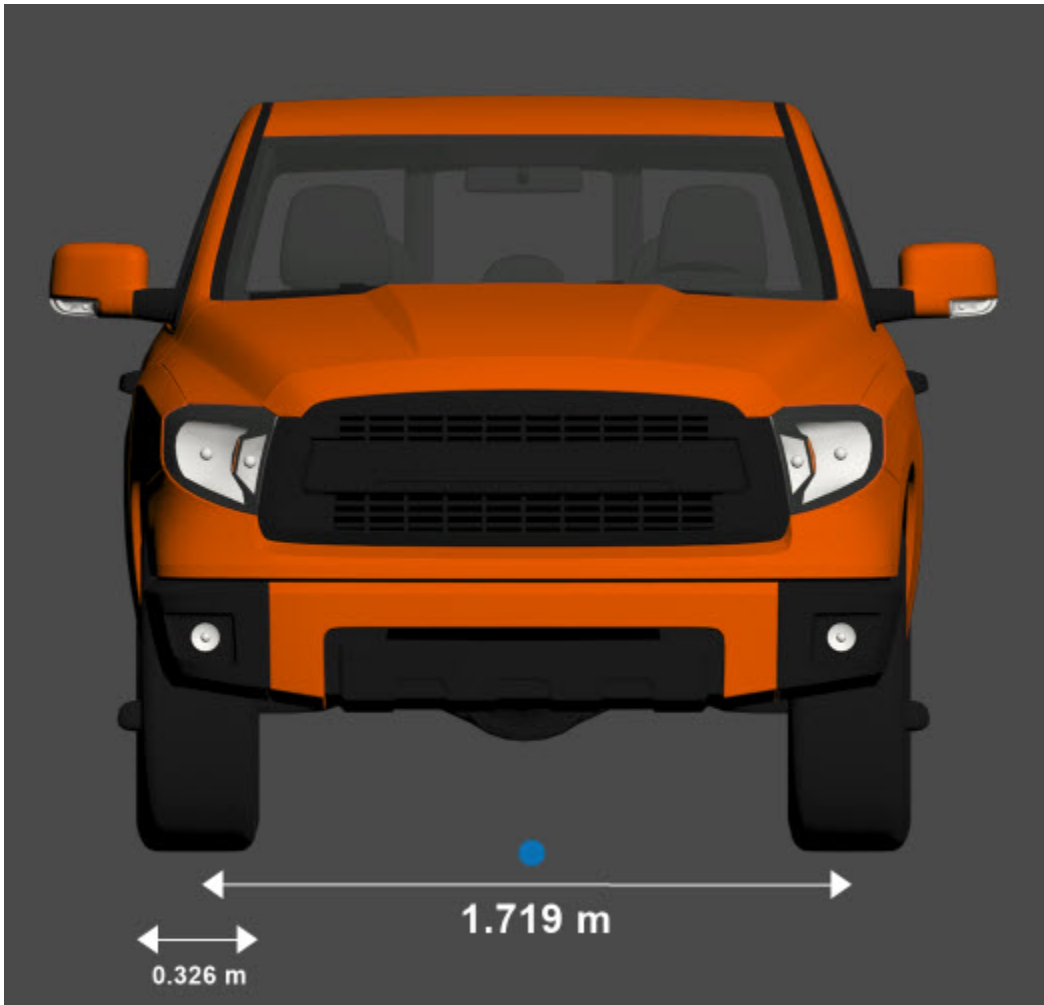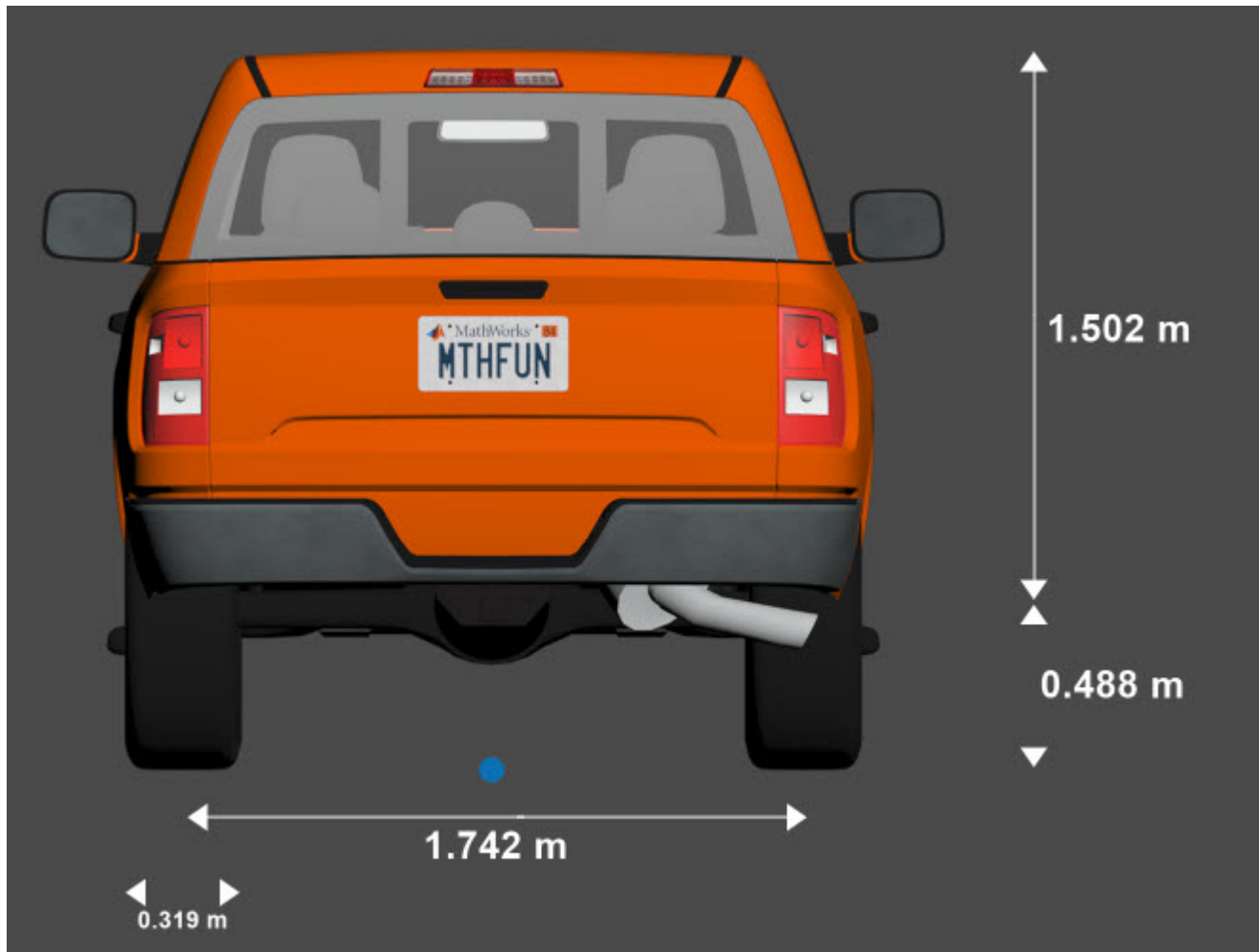
Dimensions
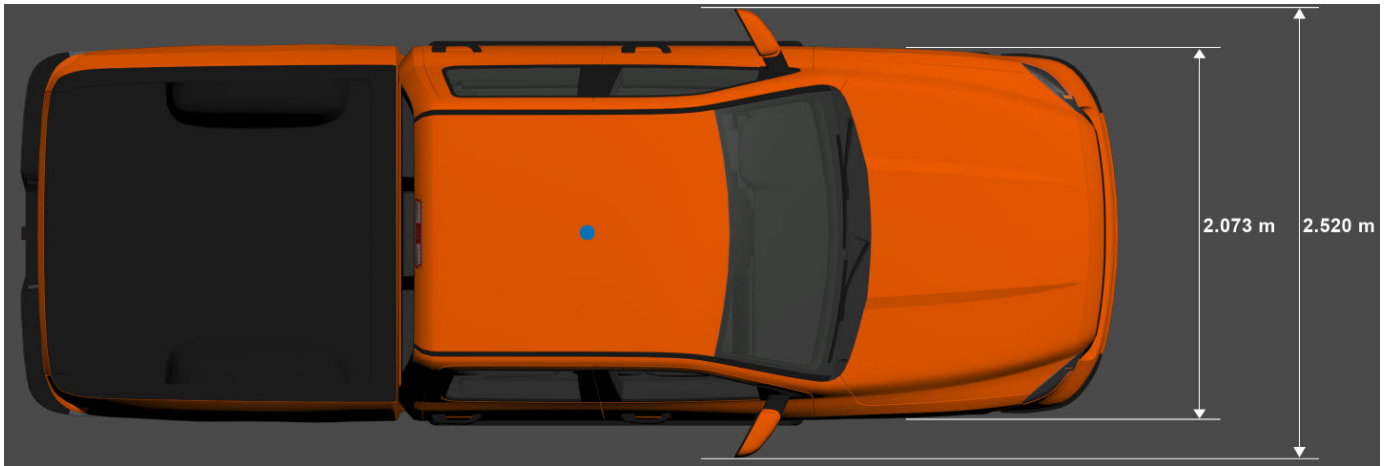
Top-down view — Vehicle width dimensions
diagram

Side view — Vehicle length, front overhang, and rear overhang dimensions
diagram
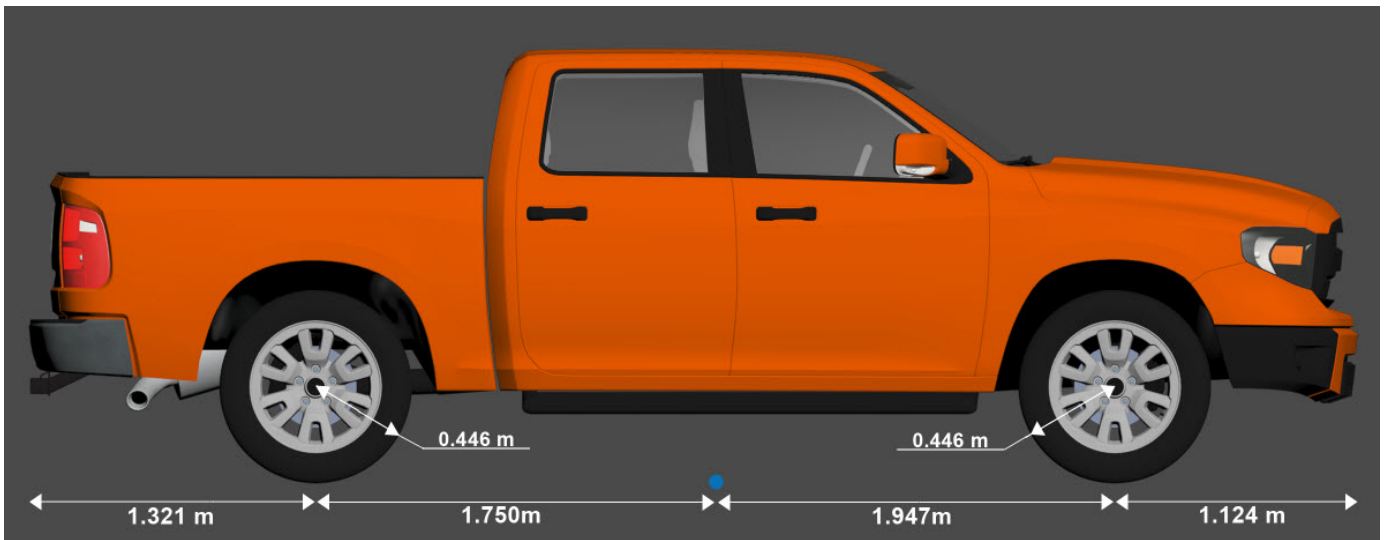
Front view — Tire width and front axle dimensions
diagram

`Rear view` — Vehicle height and rear axle dimensions
diagram

See Also

Simulation 3D Scene Configuration | Simulation 3D Vehicle | Simulation 3D Vehicle with Ground Following

Topics

"Coordinate Systems in Vehicle Dynamics Blockset"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"

Box Truck

Box truck vehicle dimensions

Description

Box truck is one of the vehicles that you can use within the 3D simulation environment. This environment is rendered using the Unreal Engine from Epic Games. The following diagram provides the dimensions of this vehicle. The height dimensions are with respect to the vertical ground plane. The length and width dimensions are with respect to the origin of the vehicle in the vehicle coordinate system. The origin is on the ground, at the geometric center of the vehicle. For more detailed views of these diagrams, see the **Dimensions** section.

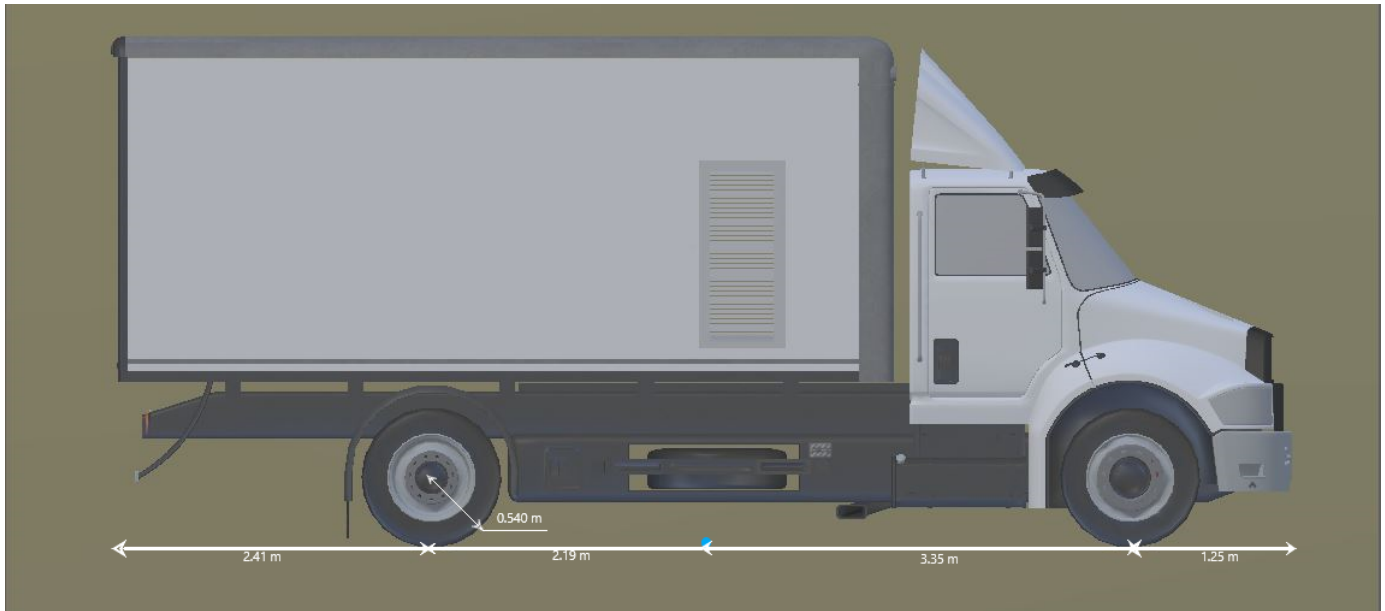To add this type of vehicle to the 3D simulation environment:

1   Add a Simulation 3D Vehicle or Simulation 3D Vehicle with Ground Following block to your Simulink model.

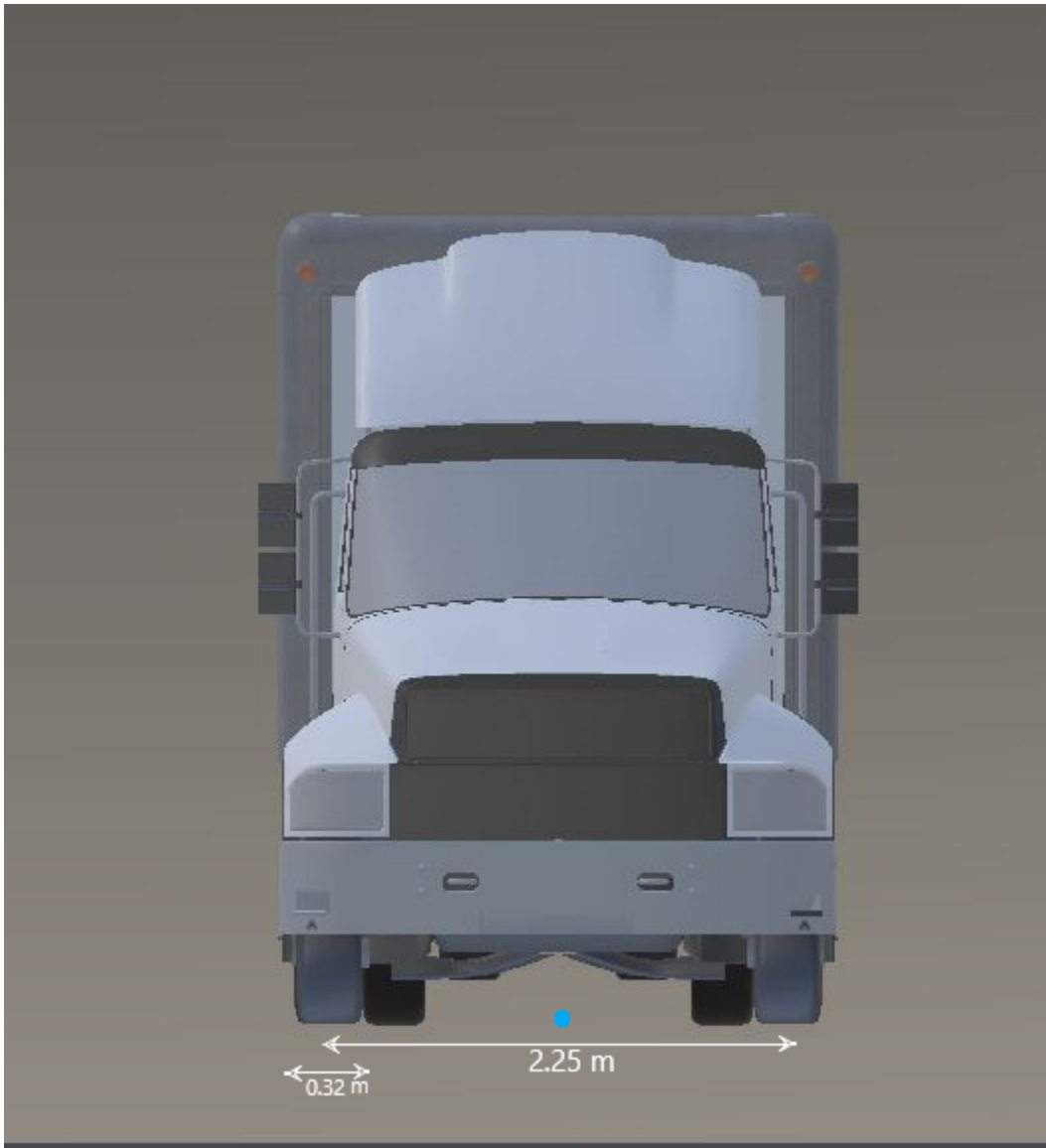2   In the block, set the **Type** parameter to `Box truck`.

Dimensions
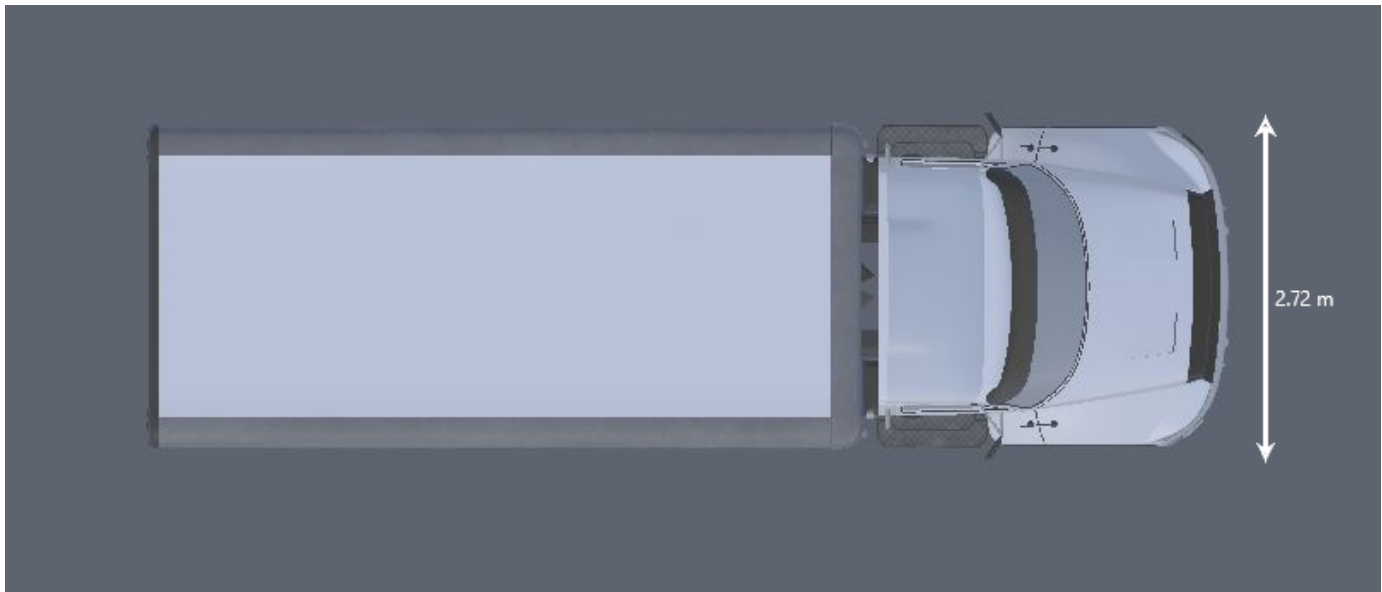
Top-down view — Vehicle width dimensions
diagram

Side view — Vehicle length, front overhang, and rear overhang dimensions
diagram

`Front view` — Tire width and front axle dimensions
diagram

`Rear view` **— Vehicle height and rear axle dimensions**
diagram

See Also

Simulation 3D Scene Configuration | Simulation 3D Vehicle | Simulation 3D Vehicle with Ground Following

Topics

"Coordinate Systems in Vehicle Dynamics Blockset"

"Vehicle Dynamics Blockset Communication with 3D Visualization Software"

Blocks in Reference Applications

3D Engine

Configure scenes in reference applications

Description

The 3D Engine block implements the 3D simulation environment. Vehicle Dynamics Blockset integrates the 3D simulation environment with Simulink so that you can query the world around the vehicle for virtually testing perception, control, and planning algorithms.

To position the vehicle in the scene:

1   Select the position initialization method:

- **Recommended for scene** — Set the initial vehicle position to values recommended for the scene
- **User-specified** — Set your own initial vehicle position

2   Select **Apply** to modify the initial vehicle position parameters.

3   Click **Update the model workspaces with the initial values** to overwrite the initial vehicle position in the model workspaces with the applied values.

Ports

Input

VehFdbk — Vehicle feedback
Bus

Bus containing vehicle feedback signals, including velocity, acceleration, and steering wheel torque.

Parameters

3D Engine

3D Engine — Enable 3D visualization
off (default) | on

Enable 3D visualization.

Scene — 3D scene
Straight road | Curved road | Parking lot | Double lane change | Open surface | US city block | US highway | Virtual Mcity | Large parking lot | Custom

Specify the name of the 3D scene.

Engine frame rate, dt3D — Graphics
.03 (default)

Graphics frame rate, in s. The graphics frame rate is the inverse of the sample time.

Recommended for scene — Initial vehicle position
on (default) | off

Use vehicle positions that are recommended for the scene.

User-specified — Initial vehicle position
off (default) | on

Specify to set your own initial vehicle position values.

Initial longitudinal position, X_o — Initial longitudinal position
off (default) | on

Initial vehicle CG position along the earth-fixed *X*-axis, in m.

Initial lateral position, Y_o — Initial lateral position
off (default) | on

Initial vehicle CG position along the earth-fixed *Y*-axis, in m.

Initial vertical position, Z_o — Initial vertical position
off (default) | on

Initial vehicle CG position along the earth-fixed *Z*-axis, in m.

Initial roll angle, phi_o — Roll
off (default) | on

Rotation of the vehicle-fixed frame about the earth-fixed *X*-axis (roll), in rad.

Initial pitch angle, theta_o — Pitch
off (default) | on

Rotation of the vehicle-fixed frame about the earth-fixed *Y*-axis (pitch), in rad.

Initial yaw angle, psi_o — Yaw
off (default) | on

Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw), in rad.

See Also
Curved Road | **Double Lane Change** | **Large Parking Lot** | **Open Surface** | **Parking Lot** | **Straight Road** | **US City Block** | **US Highway** | **Virtual Mcity**

Topics
"Double-Lane Change Maneuver"
"Slowly Increasing Steering Maneuver"
"Swept-Sine Steering Maneuver"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"3D Visualization Engine Requirements"
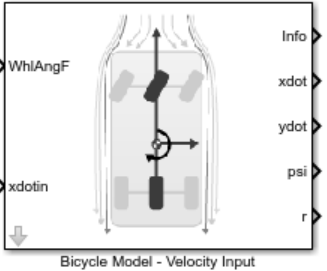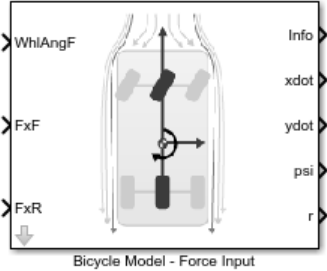
External Websites
Unreal Engine

Introduced in R2019a

Bicycle Model

Implement a single track 3DOF rigid vehicle body to calculate longitudinal, lateral, and yaw motion

Description

The Bicycle Model block implements a rigid two-axle single track vehicle body model to calculate longitudinal, lateral, and yaw motion. The block accounts for body mass, aerodynamic drag, and weight distribution between the axles due to acceleration and steering. There are two types of Bicycle Model blocks.

Block	Implementation
Bicycle Model - Velocity Input Bicycle Model - Velocity Input	• Block assumes that the external longitudinal velocity is quasi-steady state so the longitudinal acceleration is approximately zero. • Since the motion is quasi-steady, the block calculates only lateral forces using the tire slip angles and linear cornering stiffness.
Bicycle Model - Force Input Bicycle Model - Force Input	• Block uses the external longitudinal force to accelerate or brake the vehicle. • Block calculates lateral forces using the tire slip angles and linear cornering stiffness.

To calculate the normal forces on the front and rear axles, the block uses rigid-body vehicle motion, suspension system forces, and wind and drag forces. The block resolves the force and moment components on the rigid vehicle body frame.

Ports

Input

WhlAngF — Wheel angle
scalar

Front wheel angle, in rad.

FxF — Force Input: Total longitudinal force on the front axle
scalar

Longitudinal force on the front axle, Fx_F, along vehicle-fixed x-axis, in N.

Bicycle Model - Force Input block input port.

FxR — Force Input: Total longitudinal force on the rear axle
scalar

Longitudinal force on the rear axle, Fx_R, along vehicle-fixed x-axis, in N.

Bicycle Model - Force Input block input port.

xdotin — Velocity Input: Longitudinal velocity
scalar

Vehicle CG velocity along vehicle-fixed *x*-axis, in m/s.

Bicycle Model - Velocity Input block input port.

Output

Info — Bus signal
bus

Bus signal containing these block values.

Signal				Description	Value	Units
InertFrm	Cg	Disp	X	Vehicle CG displacement along earth-fixed X-axis	Computed	m
			Y	Vehicle CG displacement along earth-fixed Y-axis	0	m
			Z	Vehicle CG displacement along earth-fixed Z-axis	Computed	m
		Vel	Xdot	Vehicle CG velocity along earth-fixed X-axis	Computed	m/s
			Ydot	Vehicle CG velocity along earth-fixed Y-axis	0	m/s
			Zdot	Vehicle CG velocity along earth-fixed Z-axis	Computed	m/s
		Ang	phi	Rotation of the vehicle-fixed frame about earth-fixed X-axis (roll)	0	rad
			theta	Rotation of the vehicle-fixed frame about earth-fixed Y-axis (pitch)	Computed	rad
			psi	Rotation of the vehicle-fixed frame about earth-fixed Z-axis (yaw)	0	rad
	FrntAxl	Disp	X	Front axle displacement along the earth-fixed X-axis	Computed	m
			Y	Front axle displacement along the earth-fixed Y-axis	0	m

Signal				Description	Value	Units
			Z	Front axle displacement along the earth-fixed Z-axis	Computed	m
		Vel	Xdot	Front axle velocity along the earth-fixed X-axis	Computed	m/s
			Ydot	Front axle velocity along the earth-fixed Y-axis	0	m/s
			Zdot	Front axle velocity along the earth-fixed Z-axis	Computed	m/s
	RearAxl	Disp	X	Rear axle displacement along the earth-fixed X-axis	Computed	m
			Y	Rear axle displacement along the earth-fixed Y-axis	0	m
			Z	Rear axle displacement along the earth-fixed Z-axis	Computed	m
		Vel	Xdot	Rear axle velocity along the earth-fixed X-axis	Computed	m/s
			Ydot	Rear axle velocity along the earth-fixed Y-axis	0	m/s
			Zdot	Rear axle velocity along the earth-fixed Z-axis	Computed	m/s
BdyFrm	Cg	Vel	xdot	Vehicle CG velocity along vehicle-fixed x-axis	Computed	m/s
			ydot	Vehicle CG velocity along vehicle-fixed y-axis	0	m/s
			zdot	Vehicle CG velocity along vehicle-fixed z-axis	Computed	m/s
		AngVel	p	Vehicle angular velocity about the vehicle-fixed x-axis (roll rate)	0	rad/s
			q	Vehicle angular velocity about the vehicle-fixed y-axis (pitch rate)	Computed	rad/s
			r	Vehicle angular velocity about the vehicle-fixed z-axis (yaw rate)	0	rad/s
		Acc	ax	Vehicle CG acceleration along vehicle-fixed x-axis	Computed	gn
			ay	Vehicle CG acceleration along vehicle-fixed y-axis	0	gn
			az	Vehicle CG acceleration along vehicle-fixed z-axis	Computed	gn
			xddot	Vehicle CG acceleration along vehicle-fixed x-axis	Computed	gn

Signal				Description	Value	Units
			yddot	Vehicle CG acceleration along vehicle-fixed y-axis	0	gn
			zddot	Vehicle CG acceleration along vehicle-fixed z-axis	Computed	gn
	Forces	Body	Fx	Net force on vehicle CG along vehicle-fixed x-axis	Computed	N
			Fy	Net force on vehicle CG along vehicle-fixed y-axis	0	N
			Fz	Net force on vehicle CG along vehicle-fixed z-axis	Computed	N
		FrntAxl	Fx	Longitudinal force on front axle, along the vehicle-fixed x-axis	Computed	N
			Fy	Lateral force on front axle, along the vehicle-fixed y-axis	0	N
			Fz	Normal force on front axle, along the vehicle-fixed z-axis	Computed	N
		RearAxl	Fx	Longitudinal force on rear axle, along the vehicle-fixed x-axis	Computed	N
			Fy	Lateral force on rear axle, along the vehicle-fixed y-axis	0	N
			Fz	Normal force on rear axle, along the vehicle-fixed z-axis	Computed	N
		Tires	FrntTire	Fx	Front tire force, along vehicle-fixed x-axis	0	N
				Fy	Front tire force, along vehicle-fixed y-axis	0	N
				Fz	Front tire force, along vehicle-fixed z-axis	Computed	N
			RearTire	Fx	Rear tire force, along vehicle-fixed x-axis	0	N
				Fy	Rear tire force, along vehicle-fixed y-axis	0	N
				Fz	Rear tire force, along vehicle-fixed z-axis	Computed	N
		Drag	Fx	Drag force on vehicle CG along vehicle-fixed x-axis	Computed	N

Signal				Description	Value	Units
			Fy	Drag force on vehicle CG along vehicle-fixed y-axis	Computed	N
			Fz	Drag force on vehicle CG along vehicle-fixed z-axis	Computed	N
		Grvty	Fx	Gravity force on vehicle CG along vehicle-fixed x-axis	Computed	N
			Fy	Gravity force on vehicle CG along vehicle-fixed y-axis	0	N
			Fz	Gravity force on vehicle CG along vehicle-fixed z-axis	Computed	N
	Moments	Body	Mx	Body moment on vehicle CG about vehicle-fixed x-axis	0	N·m
			My	Body moment on vehicle CG about vehicle-fixed y-axis	Computed	N·m
			Mz	Body moment on vehicle CG about vehicle-fixed z-axis	0	N·m
		Drag	Mx	Drag moment on vehicle CG about vehicle-fixed x-axis	0	N·m
			My	Drag moment on vehicle CG about vehicle-fixed y-axis	Computed	N·m
			Mz	Drag moment on vehicle CG about vehicle-fixed z-axis	0	N·m
	FrntAxl	Disp	x	Front axle displacement along the vehicle-fixed x-axis	Computed	m
			y	Front axle displacement along the vehicle-fixed y-axis	0	m
			z	Front axle displacement along the vehicle-fixed z-axis	Computed	m
		Vel	xdot	Front axle velocity along the vehicle-fixed x-axis	Computed	m/s
			ydot	Front axle velocity along the vehicle-fixed y-axis	0	m/s
			zdot	Front axle velocity along the vehicle-fixed z-axis	Computed	m/s

10-9

Signal				Description	Value	Units
	RearAxl	Disp	x	Rear axle displacement along the vehicle-fixed x-axis	Computed	m
			y	Rear axle displacement along the vehicle-fixed y-axis	0	m
			z	Rear axle displacement along the vehicle-fixed z-axis	Computed	m
		Vel	xdot	Rear axle velocity along the vehicle-fixed x-axis	Computed	m/s
			ydot	Rear axle velocity along the vehicle-fixed y-axis	0	m/s
			zdot	Rear axle velocity along the vehicle-fixed z-axis	Computed	m/s
	Pwr	PwrExt		Applied external power	Computed	W
		Drag		Power loss due to drag	Computed	W

xdot — Vehicle body longitudinal velocity
scalar

Vehicle CG velocity along vehicle-fixed x-axis, in m/s.

ydot — Vehicle body lateral velocity
scalar

Vehicle CG velocity along vehicle-fixed y-axis, in m/s.

psi — Yaw
scalar

Rotation of the vehicle-fixed frame about earth-fixed Z-axis (yaw), in rad..

r — Yaw rate
scalar

Vehicle angular velocity, r, about the vehicle-fixed z-axis (yaw rate), in rad/s.

Parameters

Longitudinal

Number of wheels on front axle, NF — Front wheel count
2 (default) | scalar

Number of wheels on front axle, N_F. The value is dimensionless.

Number of wheels on rear axle, NR — Rear wheel count
2 (default) | scalar

Number of wheels on rear axle, N_R. The value is dimensionless.

Vehicle mass, m — Vehicle mass
2000 (default) | scalar

Vehicle mass, m, in kg.

Longitudinal distance from center of mass to front axle, a — Front axle distance
1.4 (default) | scalar

Horizontal distance a from the vehicle CG to the front wheel axle, in m.

Longitudinal distance from center of mass to rear axle, b — Rear axle distance
1.6 (default) | scalar

Horizontal distance b from the vehicle CG to the rear wheel axle, in m.

Vertical distance from center of mass to axle plane, h — Height
0.35 (default) | scalar

Height of vehicle CG above the axles, h, in m.

Initial inertial frame longitudinal position, X_o — Position
0 (default) | scalar

Initial vehicle CG displacement along earth-fixed X-axis, in m.

Initial longitudinal velocity, xdot_o — Velocity
0 (default) | scalar

Initial vehicle CG velocity along vehicle-fixed x-axis, in m/s.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter, set **Axle forces** to one of these options:

- External longitudinal forces
- External forces

Lateral

Front tire corner stiffness, Cy_f — Stiffness
12e3 (default) | scalar

Front tire corner stiffness, Cy_f, in N/rad.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Clear **Mapped corner stiffness**.

Rear tire corner stiffness, Cy_r — Stiffness
11e3 (default) | scalar

Rear tire corner stiffness, Cy_r, in N/rad.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter:

1   Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Clear **Mapped corner stiffness**.

Initial inertial frame lateral displacement, Y_o — Position
0 (default) | scalar

Initial vehicle CG displacement along earth-fixed *Y*-axis, in m.

Initial lateral velocity, ydot_o — Velocity
0 (default) | scalar

Initial vehicle CG velocity along vehicle-fixed *y*-axis, in m/s.

Yaw

Yaw polar inertia, Izz — Inertia
4000 (default) | scalar

Yaw polar inertia, in kg*m^2.

Initial yaw angle, psi_o — Psi rotation
0 (default) | scalar

Rotation of the vehicle-fixed frame about earth-fixed *Z*-axis (yaw), in rad.

Initial yaw rate, r_o — Yaw rate
0 (default) | scalar

Vehicle angular velocity about the vehicle-fixed *z*-axis (yaw rate), in rad/s.

Aerodynamic

Longitudinal drag area, Af — Effective vehicle cross-sectional area
2 (default) | scalar

Effective vehicle cross-sectional area, A_f, to calculate the aerodynamic drag force on the vehicle, in m^2.

Longitudinal drag coefficient, Cd — Air drag coefficient
.3 (default) | scalar

Air drag coefficient, C_d. The value is dimensionless.

Longitudinal lift coefficient, Cl — Air lift coefficient
.1 (default) | scalar

Air lift coefficient, C_l. The value is dimensionless.

Longitudinal drag pitch moment, Cpm — Pitch drag
.1 (default) | scalar

Longitudinal drag pitch moment coefficient, C_{pm}. The value is dimensionless.

Relative wind angle vector, beta_w — Wind angle
[0:0.01:0.3] (default) | vector

Relative wind angle vector, β_w, in rad.

Side force coefficient vector, Cs — Side force coefficient
[0:0.03:0.9] (default) | vector

Side force coefficient vector coefficient, C_s. The value is dimensionless.

Yaw moment coefficient vector, Cym — Yaw moment drag
[0:0.01:0.3] (default) | vector

Yaw moment coefficient vector coefficient, C_{ym}. The value is dimensionless.

Environment

Absolute air pressure, Pabs — Pressure
101325 (default) | scalar | scalar

Environmental absolute pressure, P_{abs}, in Pa.

Air temperature, Tair — Temperature
273 (default) | scalar

Environmental absolute temperature, T, in K.

Dependencies

To enable this parameter, clear **Air temperature**.

Gravitational acceleration, g — Gravity
9.81 (default) | scalar

Gravitational acceleration, g, in m/s^2.

Nominal friction scaling factor, mu — Friction scale factor
1 (default) | scalar

Nominal friction scale factor, μ. The value is dimensionless.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter:

1 Set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

2   Clear **External Friction**.

Simulation

Longitudinal velocity tolerance, xdot_tol — Tolerance
.01 (default) | scalar

Longitudinal velocity tolerance, in m/s.

Nominal normal force, Fznom — Normal force
5000 (default) | scalar

Nominal normal force, in N.

Dependencies

For the Vehicle Body 3DOF Single Track or Vehicle Body 3DOF Dual Track blocks, to enable this parameter, set **Axle forces** to one of these options:

- External longitudinal velocity
- External longitudinal forces

Geometric longitudinal offset from axle plane, longOff — Longitudinal offset
0 (default) | scalar

Vehicle chassis offset from axle plane along body-fixed x-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

Geometric lateral offset from center plane, latOff — Lateral offset
0 (default) | scalar

Vehicle chassis offset from center plane along body-fixed y-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

Geometric vertical offset from axle plane, vertOff — Vertical offset
0 (default) | scalar

Vehicle chassis offset from axle plane along body-fixed z-axis, in m. When you use the 3D visualization engine, consider using the offset to locate the chassis independent of the vehicle CG.

Wrap Euler angles, wrapAng — Selection
off (default) | on

Wrap the Euler angles to the interval [-pi, pi]. For vehicle maneuvers that might undergo vehicle yaw rotations that are outside of the interval, consider deselecting the parameter if you want to:

- Track the total vehicle yaw rotation.
- Avoid discontinuities in the vehicle state estimators.

References

[1] Gillespie, Thomas. *Fundamentals of Vehicle Dynamics*. Warrendale, PA: Society of Automotive Engineers (SAE), 1992.

Introduced in R2018a

Driver Commands

Configure driver

Description

The Driver Commands block implements the driver model that the reference application uses to generate acceleration, braking, gear, and steering commands. By default, if you select the Reference Generator block parameter **Use maneuver-specific driver, initial position, and scene**, the reference application selects the driver for the maneuver that you specified.

Vehicle Command Mode Setting	Implementation
Longitudinal Driver	Longitudinal Driver block — Longitudinal speed-tracking controller. Based on reference and feedback velocities, the block generates normalized acceleration and braking commands that can vary from 0 through 1. Use the block to model the dynamic response of a driver or to generate the commands necessary to track a longitudinal drive cycle.
Predictive Driver	Predictive Driver block — Controller that generates normalized steering, acceleration, and braking commands to track longitudinal velocity and a lateral reference displacement. The normalized commands can vary between -1 to 1. The controller uses a single-track (bicycle) model for optimal single-point preview control.
Open Loop	Implements an open-loop system so that you can configure the reference application for constant or signal-based steering, acceleration, braking, and gear command input.

Ports

Input

VehRef — Vehicle reference signals
Bus

Bus containing the vehicle reference signals, including longitudinal and lateral displacement, and steering.

VehFdbk — Vehicle feedback signals
Bus

Bus containing vehicle displacement feedback signals.

Output

Driver — Command signals
Bus

Bus containing the commands, including steering, acceleration, braking, and gear commands.

Parameters

Vehicle command mode — Enable 3D visualization
Predictive Driver (default) | Longitudinal Driver | Open Loop

Specify driver model.

See Also
Longitudinal Driver | Predictive Driver

Introduced in R2019a

Reference Generator

Generate maneuver reference signals

Description

The Reference Generator block sets the parameters that configure the maneuver and 3D simulation environment. By default, the block is set for the constant radius maneuver with the 3D simulation engine environment disabled.

Model

Use the **Maneuver** parameter to specify the type of maneuver. After you select the maneuver, use the parameters to specify the maneuver settings. By default:

- **Use maneuver-specific driver, initial position, and scene** — Set to on
- **Maneuver start time** — Set to 3s
- **Longitudinal velocity reference** — Set to 30s
- **Longitudinal entrance velocity setpoint units** — Set to mph

Maneuver Setting	Implementation
Double Lane Change	"Double-Lane Change Maneuver" • **Vehicle width** — Lane signals for the Visualization subsystem; used for the left and right lane boundaries • **Lateral reference data** — Lateral reference trajectory as a function of the longitudinal distance • **Distance after target speed to begin reference** — Start the maneuver at specified distance after the vehicle reaches the target speed
Increasing Steer	"Slowly Increasing Steering Maneuver" • **Handwheel rate** — Linear rate to increase steering wheel angle • **Maximum handwheel angle** — Maximum steering wheel angle
Swept Sine	"Swept-Sine Steering Maneuver" • **Steering amplitude** — Sinusoidal wave amplitude • **Final frequency** — Cut off frequency to stop the maneuver

Maneuver Setting	Implementation
Sine with Dwell	In the test, the vehicle: • Accelerates until it hits a target velocity. • Maintains the target velocity. • Responds to a sinusoidal with dwell steering command. • **Steer frequency** — Sinusoidal wave frequency • **Steer amplitude** — Sinusoidal wave amplitude • **Dwell time** — Dwell time
Constant Radius	"Constant Radius Maneuver" • **Radius value** — Turn radius

3D Engine

The 3D engine implements the 3D simulation environment. Vehicle Dynamics Blockset integrates the 3D simulation environment with Simulink so that you can query the world around the vehicle for virtually testing perception, control, and planning algorithms. For 3D engine requirements, see "3D Visualization Engine Requirements". To enable the 3D engine, on the **3D Engine** tab, select **Enabled**.

To position the vehicle in the scene:

1 Select the position initialization method:

   • **Recommended for scene** — Set the initial vehicle position to values recommended for the scene
   • **User-specified** — Set your own initial vehicle position

2 Select **Apply** to modify the initial vehicle position parameters.

3 Click **Update the model workspaces with the initial values** to overwrite the initial vehicle position in the model workspaces with the applied values.

Ports

Input

VehFdbk — Vehicle feedback
Bus

Bus containing vehicle feedback signals, including velocity, acceleration, and steering wheel torque.

Output

Vis — Visualization reference signals
Bus

Bus containing the visualization reference signals, including longitudinal and lateral displacement, and steering.

Ref — Vehicle reference signals
Bus

Bus containing the vehicle reference signals, including longitudinal and lateral displacement, and steering.

Fdbk — Vehicle location feedback signals
Bus

Bus containing vehicle location feedback signals, including position.

Parameters

Configuration

Maneuver — Select maneuver
Constant Radius (default) | Double Lane Change | Increasing Steer | Swept Sine | Sine with Dwell

Specify the scene type.

Maneuver start time — Start time
on (default) | off

Specify the name of the 3D scene.

Longitudinal velocity reference — Target velocity
scalar

Target velocity.

Longitudinal entrance velocity setpoint units — Units
mph (default)

Units for target velocity.

Simulation time — Simulation time
scalar

Time, in s.

Constant Radius

Radius value — Radius
scalar

Radius value, in m.

Turn direction — Turn direction
Right (default) | Left

Turn direction.

Lateral acceleration threshold — Lateral acceleration
scalar

Lateral acceleration threshold, in g.

Stop simulation at lateral acceleration threshold — Selection
off (default) | on

Stop simulation if vehicle exceeds lateral acceleration threshold.

Double Lane Change

Inertial longitudinal position of gate entrance — Position
scalar

Inertial longitudinal position of gate entrance, in m.

Distance after target speed to begin reference — Start distance
scalar

Distance after target speed to begin reference, in m.

Vehicle width — Vehicle width
scalar

Vehicle width, in m.

The left and right lane boundaries are a function of the **Vehicle width** parameter.

Lateral offset — Lateral offset
scalar

Lateral offset, in m.

Lateral reference position breakpoints — Breakpoints
scalar

Lateral reference position breakpoints, in m.

Use the **Lateral reference position breakpoints** and **Lateral reference data** parameters to specify the lateral reference trajectory as a function of the longitudinal distance.

Lateral reference data — Lateral data
scalar

Use the **Lateral reference position breakpoints** and **Lateral reference data** parameters to specify the lateral reference trajectory as a function of the longitudinal distance.

Increasing Steer

Handwheel rate — Handwheel rate
scalar

Handwheel rate, in deg/s.

Maximum handwheel angle — Maximum handwheel
scalar

Maximum handwheel angle, in deg.

Steering hold time after max angle reached — Steering hold
scalar

Steering hold, in s.

Lateral acceleration threshold — Lateral acceleration
scalar

Lateral acceleration threshold, in g.

Stop simulation at lateral acceleration threshold — Selection
off (default) | on

Stop simulation if vehicle exceeds lateral acceleration threshold.

Swept Sign

Swept time — Sweep time
scalar

Sweep time, in s.

Steering amplitude — Steering amplitude
scalar

Sinusoidal steering amplitude, in deg.

Final frequency — Final frequency
scalar

Cut off frequency to stop the maneuver, in Hz.

Sign with Dwell

Steer frequency — Steer frequency
scalar

Steer frequency, in Hz.

Steer amplitude — Steer amplitude
scalar

Sinusoidal steering amplitude, in deg.

Dwell time — Dwell time
scalar

Dwell time, in s.

3D Engine

3D Engine — Enable 3D visualization
off (default) | on

Enable 3D visualization.

Scene — 3D scene
Straight road | Curved road | Parking lot | Double lane change | Open surface | US city block | US highway | Virtual Mcity | Large parking lot | Custom

Specify the name of the 3D scene.

Engine frame rate, dt3D — Graphics
.03 (default)

Graphics frame rate, in s. The graphics frame rate is the inverse of the sample time.

Recommended for scene — Initial vehicle position
on (default) | off

Use vehicle positions that are recommended for the scene.

User-specified — Initial vehicle position
off (default) | on

Specify to set your own initial vehicle position values.

Initial longitudinal position, X_o — Initial longitudinal position
off (default) | on

Initial vehicle CG position along the earth-fixed *X*-axis, in m.

Initial lateral position, Y_o — Initial lateral position
off (default) | on

Initial vehicle CG position along the earth-fixed *Y*-axis, in m.

Initial vertical position, Z_o — Initial vertical position
off (default) | on

Initial vehicle CG position along the earth-fixed *Z*-axis, in m.

Initial roll angle, phi_o — Roll
off (default) | on

Rotation of the vehicle-fixed frame about the earth-fixed *X*-axis (roll), in rad.

Initial pitch angle, theta_o — Pitch
off (default) | on

Rotation of the vehicle-fixed frame about the earth-fixed *Y*-axis (pitch), in rad.

Initial yaw angle, psi_o — Yaw
off (default) | on

Rotation of the vehicle-fixed frame about the earth-fixed *Z*-axis (yaw), in rad.

See Also

3D Engine | Driver Commands

Topics
"Constant Radius Maneuver"
"Double-Lane Change Maneuver"
"Slowly Increasing Steering Maneuver"
"Swept-Sine Steering Maneuver"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"3D Visualization Engine Requirements"

External Websites
Unreal Engine

Introduced in R2019a

Classes

sim3d.Editor

Interface to the Unreal Engine project

Description

Use the `sim3d.Editor` class to interface with the Unreal Editor.

To develop scenes with the Unreal Editor and co-simulate with Simulink, you need the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package. The support package contains an Unreal Engine project that allows you to customize the Vehicle Dynamics Blockset scenes. For information about the support package, see "Support Package for Customizing Scenes".

Creation

Syntax

`sim3d.Editor(project)`

Description

MATLAB creates an `sim3d.Editor` object for the Unreal Editor project specified in `sim3d.Editor(project)`.

Input Arguments

`project` — Project path and name
string array

Project path and name.

Example: `"C:\Local\AutoVrtlEnv\AutoVrtlEnv.uproject"`

Data Types: `string`

Properties

`Uproject` — Project path and name
string array

This property is read-only.

Project path and name with Unreal Engine project file extension.

Example: `"C:\Local\AutoVrtlEnv\AutoVrtlEnv.uproject"`

Data Types: `string`

Object Methods

open   Open the Unreal Editor

Examples

Open Project in Unreal Editor

Open an Unreal Engine project in the Unreal Editor.

1   Create an instance of the `sim3d.Editor` class for the Unreal Engine project located in `C:\Local\AutoVrtlEnv\AutoVrtlEnv.uproject`.

    ```
    editor=sim3d.Editor(fullfile("C:\Local\AutoVrtlEnv\AutoVrtlEnv.uproject"))
    ```
2   Open the project in the Unreal Editor.

    ```
    editor.open();
    ```

See Also

Topics
"Support Package for Customizing Scenes"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"
"3D Visualization Engine Requirements"

Introduced in R2019b

open

Open the Unreal Editor

Syntax

```
[status,result]=open(sim3dEditorObj)
```

Description

`[status,result]=open(sim3dEditorObj)` opens the Unreal Engine project in the Unreal Editor.

To develop scenes with the Unreal Editor and co-simulate with Simulink, you need the Vehicle Dynamics Blockset Interface for Unreal Engine 4 Projects support package. The support package contains an Unreal Engine project that allows you to customize the Vehicle Dynamics Blockset scenes. For information about the support package, see "Support Package for Customizing Scenes".

Input Arguments

sim3dEditorObj — sim3d.Editor object
`sim3d.Editor` object

`sim3d.Editor` object for the Unreal Engine project.

Output Arguments

status — Command exit status
0 | nonzero integer

Command exit status, returned as either `0` or a nonzero integer. When the command is successful, `status` is `0`. Otherwise, `status` is a nonzero integer.

- If `command` includes the ampersand character (&), then `status` is the exit status when `command` starts
- If `command` does not include the ampersand character (&), then `status` is the exit status upon `command` completion.

result — Output of operating system command
character vector

Output of the operating system command, returned as a character vector. The system shell might not properly represent non-Unicode® characters.

See Also

`sim3d.Editor`

Topics
"Support Package for Customizing Scenes"
"Vehicle Dynamics Blockset Communication with 3D Visualization Software"

"3D Visualization Engine Requirements"

Introduced in R2019b