

SimMechanics™

User's Guide

R2012b

MATLAB®
& SIMULINK®

How to Contact MathWorks



www.mathworks.com
comp.soft-sys.matlab
www.mathworks.com/contact_TS.html

Web
Newsgroup
Technical Support



suggest@mathworks.com
bugs@mathworks.com
doc@mathworks.com
service@mathworks.com
info@mathworks.com

Product enhancement suggestions
Bug reports
Documentation error reports
Order status, license renewals, passcodes
Sales, pricing, and general information



508-647-7000 (Phone)



508-647-7001 (Fax)



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

For contact information about worldwide offices, see the MathWorks Web site.

SimMechanics™ User's Guide

© COPYRIGHT 2008–2012 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

September 2012 Online only

New for Version 4.1 (Release R2012b)

Multibody Modeling

Spatial Relationships

1

Modeling with Frames	1-2
Purpose of Frames	1-2
Frame Visualization	1-2
Rigid Body Modeling	1-3
Multibody Assembly	1-6
Multibody Actuation and Sensing	1-8
Frames	1-11
Origin and Basis Vectors	1-11
Inertial and Non-Inertial Frames	1-12
Adding Frames	1-12
Frame Transformations	1-14
Rigid Transformations Are Constant	1-14
Rigid Transformations Are Directional	1-15
Frame Connection Rules	1-16
Frame Line Identifies a Single Frame	1-16
Frame Node Identifies a Single Frame	1-16
Rigid Transform Block Separates Two Frames	1-17
Frame Definitions Are Local	1-17
Invalid Rigid Transform Uses	1-19
Rigidity Loops are Invalid	1-19
Shorted Rigid Transform Blocks are Invalid	1-20
Add and Transform Frames	1-22
Workflow	1-22
Build Model	1-23
Specify Transformation Parameters	1-24
Visualize Frames	1-25

Save Model	1-26
Connect Solids to Frames	1-27
Example Requirements	1-28
Open and Modify Model	1-28
Add Shape and Color to Solids	1-28
Visualize Model	1-29
Model Frame Tree	1-31
Build Model	1-32
Define Model Parameters	1-33
Add World Frame	1-34
Add Bottom Plane Frames	1-36
Add Top Plane Frames	1-40
Add Arch Frames	1-44
Save Model	1-47
Add Graphics to Frames	1-48
Measure the Motion State of a Frame	1-52
Sense Motion with Joint Blocks	1-52
Sense Motion with the Transform Sensor Block	1-53

Rigid Bodies

2

Properties of Rigid Bodies	2-2
Flexible and Rigid Bodies	2-2
Geometry	2-3
Inertia	2-7
Graphic	2-8
Solid Shapes	2-10
Simple Shapes	2-10
Advanced Shapes	2-12
Mass and Inertia	2-14
Relevant Blocks	2-14

Mass	2-14
Inertia	2-15
Import from CAD Assembly	2-17
About Extrusion and Revolution Geometries	2-18
General Extrusion Examples	2-18
Solid of Revolution	2-21
Extrusion and Revolution Cross-Section Rules	2-23
Coordinates Define Closed Loop	2-23
Counterclockwise Loop Encloses Solid Section	2-24
Clockwise Loop Encloses Hollow Section	2-24
Coordinate Loops Must Not Self-Intersect	2-25
Extrusion Rules	2-26
Revolution Rules	2-28
Model a Simple Binary Link	2-31
Build Model	2-32
Add Joint Frames	2-33
Add Geometry, Inertia, and Color	2-33
Generate and Mask Subsystem	2-35
Specify Subsystem Parameters	2-38
Visualize Model	2-38
Save Subsystem Block	2-41
Model a Hollow Cone	2-43
Parameterize Cross-Section Coordinates	2-44
Build Model	2-46
Add Geometry, Inertia, and Color	2-47
Generate and Mask Subsystem	2-48
Specify Subsystem Parameters	2-51
Visualize Model	2-52
Modify Geometry	2-53
Model a Dome	2-54
Parameterize Cross-Section Coordinates	2-55
Build Model	2-57
Add Geometry	2-58
Generate and Mask Subsystem	2-58
Specify Subsystem Parameters	2-62
Visualize Model	2-63
Modify Geometry	2-64

Model an I-Beam	2-66
Parameterize Cross-Section Coordinates	2-67
Build Model	2-69
Generate an I-Beam Subsystem	2-71
Visualize I-Beam in Mechanics Explorer	2-74
Change the I-Beam Geometry	2-76
Model a Box Beam	2-78
Parameterize Cross-Section Coordinates	2-79
Build Model	2-81
Generate a Box Beam Subsystem	2-83
Visualize Box Beam in Mechanics Explorer	2-86
Change the Box Beam Geometry	2-88
Model a Compound Binary Link	2-90
Build Model	2-92
Add Joint Frames	2-93
Add Geometry, Inertia, and Color	2-94
Generate and Mask Subsystem	2-95
Specify Subsystem Parameters	2-99
Visualize Model	2-100
Save Subsystem Block	2-102

Mechanisms

3

Joints	3-2
Joint Frames	3-2
Joint Primitives	3-3
Joint Assembly	3-4
Joint Degrees of Freedom	3-5
Joint Primitive Composition	3-7
Multibody Assembly	3-8
Workflow	3-8
Identify Joint Requirements	3-8
Connect Rigid Bodies with Joints	3-9
Specify Joint State Targets	3-10

Check Assembly	3-11
Model Report	3-14
Open Model Report	3-14
Model Report Tabs	3-14
Status Icons	3-16
Assemble Double-Pendulum Model	3-17
Summary	3-17
Requirements	3-17
Build Multibody Model	3-18
Specify Binary Link Parameters	3-19
Inspect Frames	3-19
Change Frame Orientation	3-22
Change Gravity Vector	3-24
Specify Joint State Targets	3-25
Save Model	3-27
Refine Double-Pendulum Model	3-28
Add Advanced Binary Link Subsystems	3-28
Update Model	3-29
Adjust Frames	3-30
Save Model	3-32

Actuation & Motion Sensing

4

Model Actuation Workflow	4-2
Actuate Four-Bar Crank Joint	4-3
Open Model	4-4
Expose Actuation Input Port	4-4
Model Actuation Signal	4-5
Expose Motion Sensing Output Ports	4-7
Plot Motion Data	4-8
Simulate Model	4-9
Add Internal Forces to Double-Pendulum Model	4-12

Summary	4-12
Requirements	4-12
Add Spring and Damper Forces	4-12
Add Motion Sensors	4-13
Simulate Damped Double-Pendulum Model	4-15
Actuation	4-18
Forces & Torques Blocks	4-18
Joints Blocks	4-24
Force and Motion Sensing	4-26
Motion Sensing	4-26
Force Sensing	4-27

Simulation and Analysis

Simulation

5

Configure Model for Simulation	5-2
Specify Solver Settings	5-2
Update and Simulate Model in Mechanics Explorer ...	5-4
Update Model	5-4
Simulate Model	5-5
Simulation Limitations	5-7
Limited Visualization with For Each Subsystems	5-7
No Visualization with Referenced Models	5-7
No Support for Simscape Local Solvers	5-7
Simulation Errors	5-8
Frame Position Violation	5-8
Data Validation Error	5-9
Degenerate Mass Distribution	5-10
Gimbal Lock	5-11
No Solver Block	5-11

Configure Mechanics Explorer Display	6-2
Change Background Color	6-2
Change View Point	6-5
Change View Convention	6-7
Display Multiple Screens	6-8
Toggle Visibility of Frames and Mass Centers	6-10
Visualization with Mechanics Explorer	6-11
Mechanics Explorer Window	6-11
Model Report	6-13
Simulation Video	6-13
Rotate, Pan, and Zoom View	6-14
Rotate, Pan, and Zoom Shortcuts	6-14
Rotate View	6-14
Pan View	6-15
Zoom View	6-16
Record and Play Simulation Video	6-18
Record and Save Simulation Video	6-18
Change Video Playback Speed	6-20
Play Simulation Video	6-22
Visualization Troubleshooting	6-24
Mechanics Explorer Fails to Open	6-24
Mechanics Explorer Does not Display Imported Model ...	6-24

CAD Import

CAD Translation	7-2
Software Requirements	7-2

CAD Export	7-3
CAD Import	7-4
Install and Register SimMechanics Link Software	7-8
Installation Requirements	7-8
Download SimMechanics Link Software	7-8
Install SimMechanics Link Software	7-9
Register SimMechanics Link Utility with CAD Platform ..	7-10
Link External Application to SimMechanics Link Software	7-10
Register MATLAB as Automation Server	7-11
Unregister SimMechanics Link Software	7-13
About the SimMechanics Import XML File	7-14
Organization of SimMechanics XML Import File	7-14
Root Assembly	7-15
Organization of Assemblies	7-19
Organization of Parts	7-20
Import Robot Assembly	7-24
Verify Import Files	7-25
Import Robot Assembly	7-27
Visualize and Simulate Robot Assembly	7-27
Import Stewart Platform Model	7-30
CAD Import Issues	7-33
SimMechanics Replaces Unsupported Constraint with Rigid Connection	7-33
Default View Convention is Z-Axis Up	7-34
Invalid STL Files Cause Import Issues	7-35

Deployment

Code Generation

8

About Code Generation	8-2
-----------------------------	-----

Simulation Accelerator Modes	8-2
Model Deployment	8-3
Code Generation Limitations	8-4
Variable-Step Simulink Solver Requires rsim.tlc target	8-4
Simulink Solver Must Be Continuous	8-4
Accelerator Simulation Modes Do Not Support Visualization	8-5
SimMechanics Does Not Support Run-Time Parameters ..	8-6
Configure Four-Bar Model for Code Generation	8-7
Simulate Four-Bar Model in Rapid Accelerator Mode	8-9

Multibody Modeling

Spatial Relationships

- “Modeling with Frames” on page 1-2
- “Frames” on page 1-11
- “Frame Transformations” on page 1-14
- “Frame Connection Rules” on page 1-16
- “Invalid Rigid Transform Uses” on page 1-19
- “Add and Transform Frames” on page 1-22
- “Connect Solids to Frames” on page 1-27
- “Model Frame Tree” on page 1-31
- “Add Graphics to Frames” on page 1-48
- “Measure the Motion State of a Frame” on page 1-52

Modeling with Frames

In this section...
“Purpose of Frames” on page 1-2
“Frame Visualization” on page 1-2
“Rigid Body Modeling” on page 1-3
“Multibody Assembly” on page 1-6
“Multibody Actuation and Sensing” on page 1-8

Purpose of Frames

Frames provide a means to study motion in multibody systems. With frames, you can make two solids coincident in space, separate two solids along an axis, or rotate the two solids about that same axis. In SimMechanics™, frames contain all of the information about rigid body position and orientation.

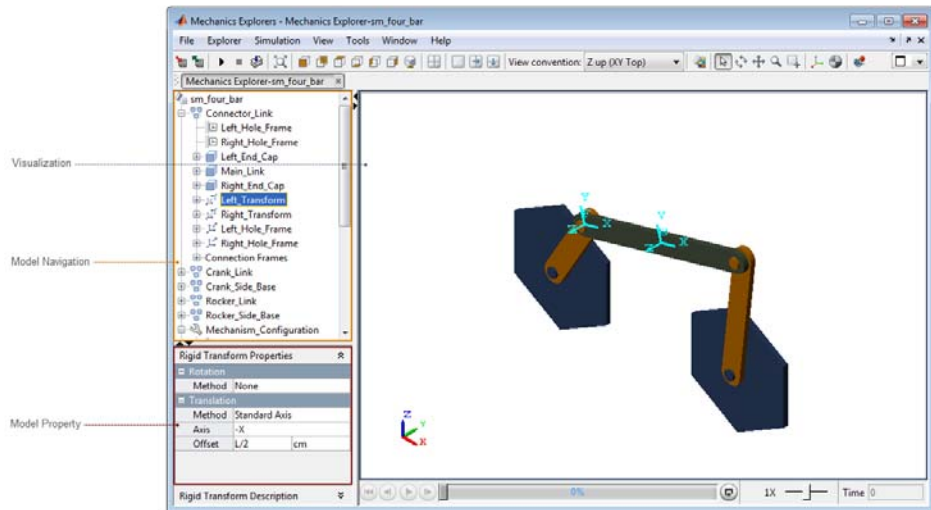
Modeling tasks that require frames include:

- Rigid body modeling
- Multibody assembly
- Multibody actuation and sensing

Frame Visualization

You can visualize the relative position and orientation of frames in a model. By default, when you update or simulate a model, a Mechanics Explorer window opens with a 3-D display of the model. Selecting the name of a subsystem, block, or frame port in the model navigation pane causes Mechanics Explorer to highlight all frames associated with that subsystem, block, or frame port. For more information, see “Visualization with Mechanics Explorer” on page 6-11.

In the following figure, a Mechanics Explorer window highlights the base and follower frames of a block named Left Transform. This Rigid Transform block translates the B frame along the $-X$ axis to obtain the F frame that connects to the revolute joint.



Rigid Body Modeling

Rigid bodies are the basic building blocks of a multibody system. In SimMechanics, the model of a rigid body contains two components, *solid properties* and *frames*. The following table describes each component and identifies the block library you use to specify the component. In the table, the **Important Blocks** column lists commonly used blocks for each component.

Component	Description	Block Library	Important Blocks
Frames	Resolve the position and orientation of solid elements, joints, sensors, forces, and torques	Frames and Transforms	Rigid Transform
Solid properties	Define the geometry, inertia, and graphic	Body Elements	Solid

Component	Description	Block Library	Important Blocks
	properties of a solid element. For more information, see “Properties of Rigid Bodies” on page 2-2		

Solid Properties

SimMechanics rigid bodies accept input for three sets of solid properties:

- Geometry — 3-D shape and size
- Inertia — Mass, density, moments and products of inertia

Note SimMechanics can automatically calculate the moments and products of inertia of a rigid body based on geometry and either mass or density parameters.

- Graphic — Color and opacity

Frames

Frames identify the position and orientation of mechanical components and dynamic influences in a model. SimMechanics provides two reference frames you can use as a starting point in a model:

- World Frame — Inertial reference frame. Each frame network can contain a single World frame. If a model contains multiple World Frame blocks, all blocks identify the same World frame.
- Reference Frame — Non-inertial reference frame. Unless directly connected to each other, Reference Frame blocks identify spatially distinct frames.

Blocks in the Body Elements library also contain a reference (R) frame port. When you add a block from this library to a model, you automatically add a non-inertial reference frame to that model.

To add new frames with a specified position and orientation, you use the Rigid Transform block. The block transforms a base frame into a new, follower, frame. The follower frame can in turn act as a base frame for another Rigid Transform block. Connecting Rigid Transform blocks in a chain structure generates a network of rigidly connected frames. You can connect Solid blocks to different frames in a frame network to represent a compound rigid body.

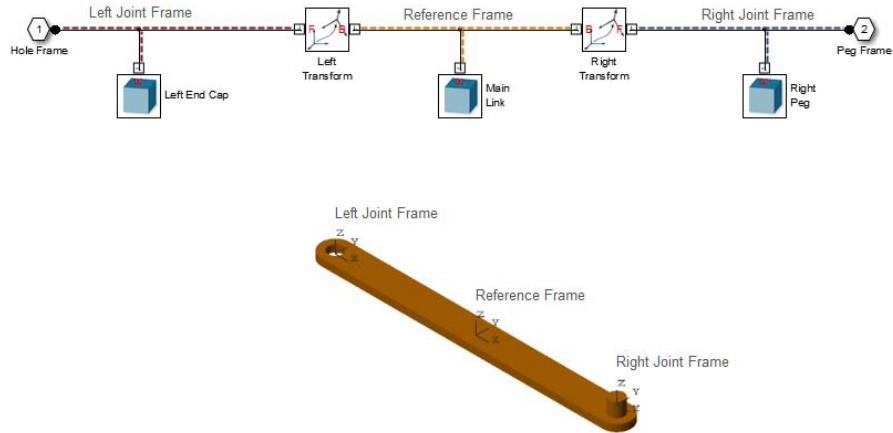
Binary Link Example

The following figure illustrates the use of frames in rigid body modeling. Each binary link contains three frames, identifying a position and orientation of interest. One frame identifies the center of mass, and two frames each identify a joint. The R frame port of the Main Link Solid block provides the parent frame for the binary link model.

The Left Transform Rigid Transform block defines the left joint frame with respect to the R frame of the Main Link Solid block. The R port of the Left End Cap Solid block identifies a frame that coincides with the left joint frame.

The Right Transform Transform Block defines the right joint frame with respect to the R frame of the Main Link Solid block. The R port of the Right Peg Solid block identifies a frame that coincides with the right joint frame.

For more information, see “Frame Connection Rules” on page 1-16.



Multibody Assembly

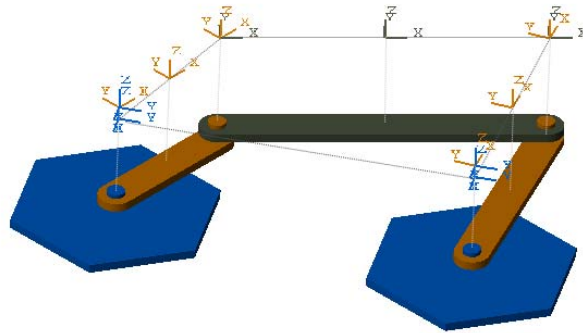
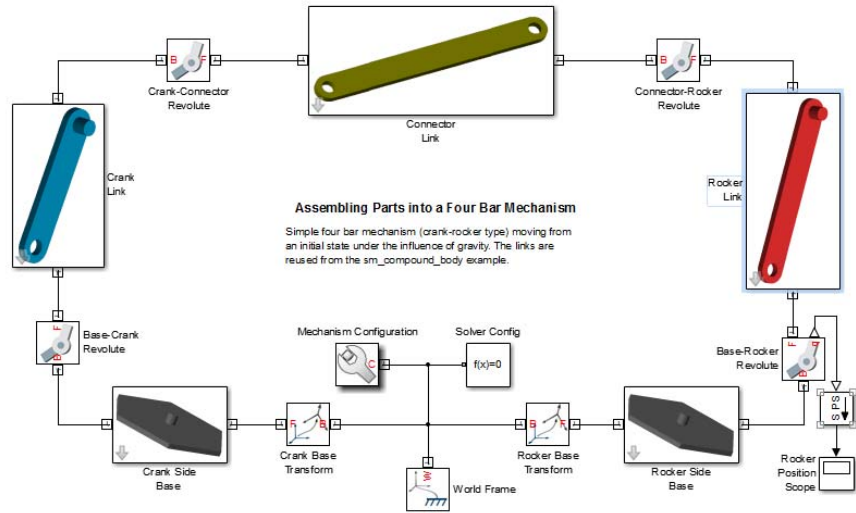
Multibody models contain two or more rigid body subsystems that interconnect through blocks from the Joints or Constraints libraries. Each rigid body subsystem behaves as an encapsulated unit with a separate set of frames and properties that you specify at the block or subsystem mask level. Blocks from the Joints and Constraints libraries provide mechanical degrees of freedom between the interconnected rigid body subsystems. A maximum of three translational and three rotational degrees of freedom are possible.

Joints and Constraints

Like Rigid Transform blocks, blocks from the Joints and Constraints libraries connect *frames*. The blocks contain two frame ports, base (B) and follower (F), each identifying one frame on a separate rigid body. Before you can connect a block from the Joints and Constraints libraries, you must add a frame to position and orient the joint or constraint between the two rigid bodies. Determine the correct position and orientation of the joint or constraint. Then, use the Rigid Transform block to create a frame with an origin at that position and a convenient set of basis vectors that you can use to orient the joint or constraint in space.

Four-Bar Example

The following figure shows the frame network of a four-bar model. Each rigid body contains one local reference frame located at the center of geometry. Rigid Transform blocks add two frames per link at the desired joint locations. Two pegs connected to the hexagonal base plates contain a reference frame that coincides with the desired joint location. No Rigid Transform block is required. Four Revolute Joint blocks connect to the links at the new frame locations. State target parameters in the Revolute Joint block dialog box set the relative position and velocity between joint frames at time zero.



Multibody Actuation and Sensing

You can add force and torque actuation inputs to frames, apply internal forces between joint frames, and sense motion parameters between any two frames.

Actuation

Dynamic models contain forces and torques that alter the rigid body state of motion. You can add forces and torques to joint frames directly with joint blocks. For greater versatility, you can use blocks from the Forces & Torques library. Forces and torques act at the origin of the frames that the block connects to. Before you add a block with force and torque capability, you must add a frame with the correct position and convenient orientation for actuation purposes.

The External Force and Torque block represents a force that originates outside the system. The block contains a single frame port (F) which identifies the frame on which the force and torque act. The Inverse Square Law Force and Spring and Damper Force blocks represent forces between two masses. The blocks contain two frame ports (B and F) that identify the frames on which the forces act. Likewise, joint blocks with actuation capabilities (all joint blocks except Weld Joint) can represent forces and torques between two masses. The blocks contain two ports (B and F) that identify the frames on which the forces and torques act. For more information, see “Actuation” on page 4-18.

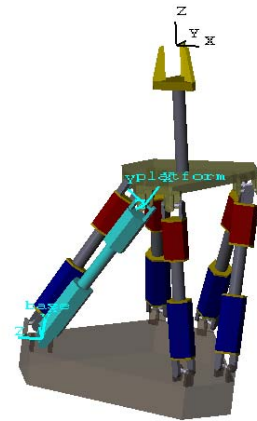
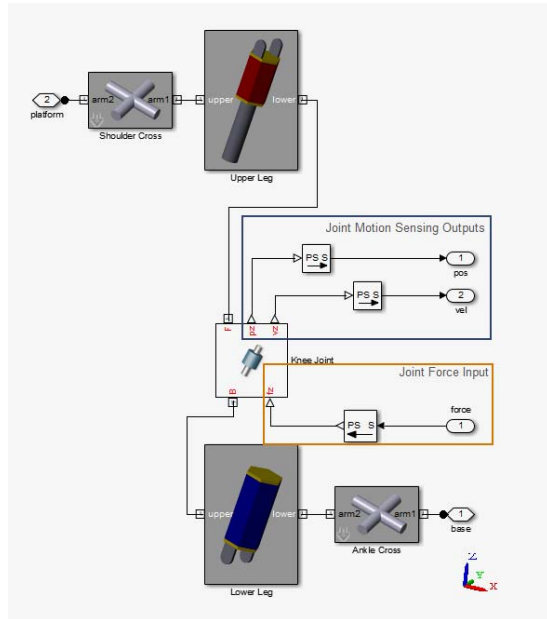
Motion Sensing

SimMechanics provides blocks with motion sensing capability for the purpose of analysis. Blocks in the Joints library can measure motion parameters between joint frames B and F. You can measure the position, velocity, and acceleration of each joint primitive. The Transform Sensor block can measure motion parameters between any two frames in a model. The block includes an extended set of translational and rotational motion parameters that you can measure. Before you can measure motion parameters, you must add frames with the correct position and orientation for the purposes of your application. For more information, see “Force and Motion Sensing” on page 4-26.

Stewart Platform Actuation and Sensing Example

The following figure shows the model of a Stewart platform leg and the corresponding display in Mechanics Explorer. The platform contains six legs, each with a prismatic knee joint that accepts force input. To control the force input, the model extracts position and velocity data from the prismatic joint, feeds the data through a control algorithm, and generates the correct force input for the prismatic joint. The prismatic joint block accepts a force input that acts reciprocally between the base and follower frame ports. The

block outputs the position and velocity parameters of the follower frame with reference to the base frame.



Frames

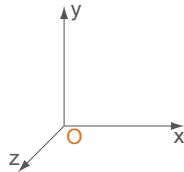
In this section...

“Origin and Basis Vectors” on page 1-11

“Inertial and Non-Inertial Frames” on page 1-12

“Adding Frames” on page 1-12

A frame is a mathematical construct that allows you to parameterize the kinematic state of any other frame in terms of coordinates. You can specify position and orientation, as well as their time derivatives—including linear and angular velocity, acceleration, and jerk. To provide this capability, frames contain a set of *basis vectors*, and an *origin*. The following figure illustrates a frame with three mutually orthogonal basis vectors \hat{e}_x , \hat{e}_y , \hat{e}_z .



Origin and Basis Vectors

The frame origin identifies a zero coordinate point. Each time that you specify or measure the coordinates of a frame, you do so relative to the origin of a chosen reference frame. Every frame can serve as a reference frame. The coordinates of a frame depend directly on the location of your chosen reference frame.

The basis vectors allow you to resolve any vector in space. Each time that you specify a vector quantity, you do so relative to the basis vectors of a chosen reference frame. SimMechanics blocks which rely on the basis vectors of a reference frame include Joints, as well as Forces and Torques. For example, the Revolute Joint block requires a well-defined z-axis, about which rotation can occur.

Inertial and Non-Inertial Frames

Multibody models contain two types of frames:

- Inertial — Frame with zero linear and angular acceleration. The frame defines absolute rest in a model. The World Frame block provides an ultimate inertial reference frame to which all other frames in a model refer.
- Non-inertial — Frame that you rigidly attach to a rigid body to describe its motion in space. The frame is free to accelerate with respect to other frames, according to the multibody dynamics that governs the model.

The optional Reference Frame block provides an ultimate non-inertial reference frame against which you can define other frames in a rigid body subsystem. The block is strictly optional.

The following properties apply to the World Frame and Reference Frame blocks:

- Multiple instances of the World Frame block represent exactly the same frame. The origin and axes of all World Frame blocks are always coincident in space.
- Multiple instances of the Reference Frame block can represent different frames. The frames are coincident in space *only* if a frame line directly connects the port frames of the Reference Frame blocks.

Adding Frames

You can add an arbitrary number of frames to a model. The following table lists the different ways you can add a frame.

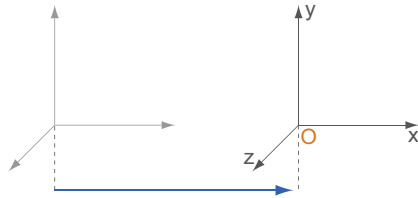
Frame Type	Ways to Add
Inertial	<ul style="list-style-type: none"> • Add World Frame block to model. • Connect Rigid Transform block to World Frame block. The Rigid Transform block defines a frame that is rigidly connected to World, and therefore inertial.
Non-inertial	<ul style="list-style-type: none"> • Add Reference Frame block to model.

Frame Type	Ways to Add
	<ul style="list-style-type: none">• Connect a Rigid Transform block to any frame port or frame line not directly connected to the World Frame block.

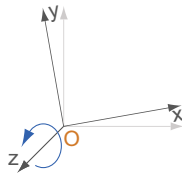
Frame Transformations

A frame transformation is an operation that changes frame position, orientation, or both. You can perform two types of transformations on a frame:

- Rotation — Angular displacement about an axis



- Translation — Linear displacement along an axis



Rotation changes frame orientation. Translation changes frame position.

To perform either transformation, use the Rigid Transform block. The base frame port of the block identifies the starting frame. The follower frame port of the block identifies the resulting frame. Vary the translation and rotation parameters to obtain any new frame in space.

Rigid Transformations Are Constant

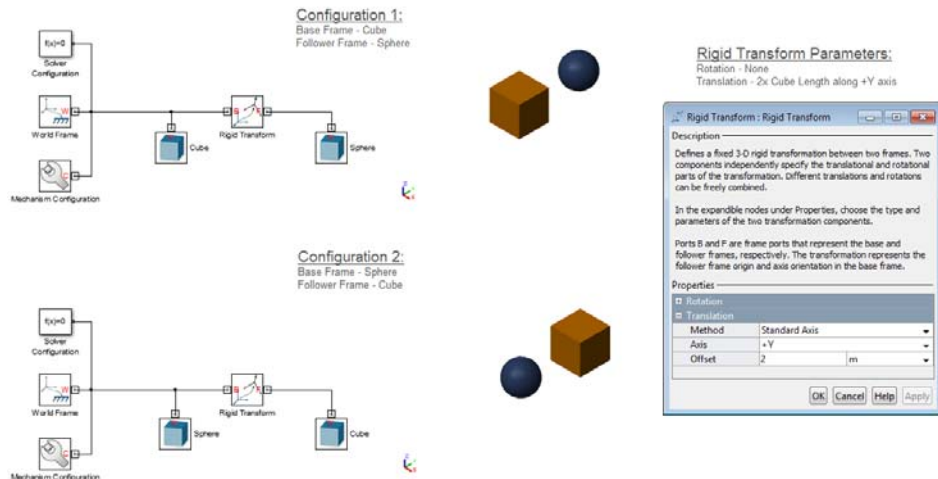
Rigid Transform blocks specify a constant spatial relationship between base and follower frames. The relative position and orientation of the two frames does not change. Connect two Solid blocks to the base and follower frames to represent a compound rigid body. The solid elements that make up the compound rigid body respond to dynamic inputs as a single unit.

Rigid Transformations Are Directional

The Rigid Transform block specifies a fixed spatial relationship between base and follower frames. However, the transformation parameters of the Rigid Transform block are *always* relative to the base frame. If you switch the base and follower port frames, you change the spatial relationship between the frames.

The following figure shows two block diagrams that contain one sphere and one cube. A Rigid Transform block specifies a positive translation parameter along the +Y axis.

- If the Cube Solid block connects to the base frame of the Rigid Transform block and the Sphere Solid block connects to the follower frame, the sphere appears on the +Y side of the cube.
- If you switch Cube and Sphere Solid block positions, the cube appears on the +Y side of the sphere.



Frame Connection Rules

Frames provide position and orientation to rigid bodies in space. The spatial relationship between frames depends on the underlying block diagram. Each block diagram must follow a set of SimMechanics frame rules that translate how blocks and edges connect into the spatial relationships between frames.

Frame Line Identifies a Single Frame

Each frame line represents a single frame along the entirety of its length. All frame ports that connect to a frame line represent the same frame, and are therefore coincident in space. The frame relationship is time-invariant, and frames remain coincident for all time.

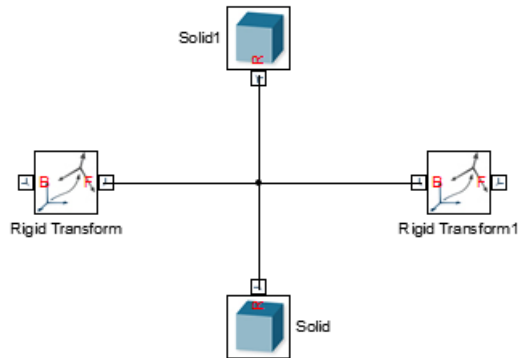
In the following figure, the follower frame port (F) of the Rigid Transform block represents the same frame as the base frame port (B) of the Rigid Transform1 block.



Frame Node Identifies a Single Frame

When two or more frame lines connect at a node, the frame lines represent the same frame. All frame ports that connect to any of the frame lines become coincident in space. The frame relationship is time-invariant and frames remain coincident for all time.

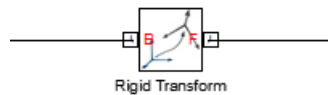
In the following figure, the four frame lines that connect at the center node represent the same frame.



Rigid Transform Block Separates Two Frames

Blocks with base (B) and follower (F) frame ports define a non-coincident spatial relationship between frames. The frame that connects to the base frame port is distinct from the frame that connects to the follower frame port.

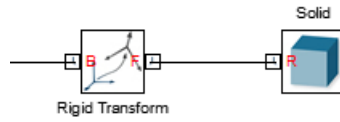
In the following figure, each frame line represents a distinct frame. The Rigid Transform block that connects the two frame lines defines the spatial relationship between the two frames.



Frame Definitions Are Local

Every SimMechanics frame has a definition that depends strictly on a neighboring frame. Frames in a frame network are indirectly related to all other frames in a multibody system, but directly related only to a neighboring frame.

In the following figure, the reference frame (R) of the Solid block is defined relative to the follower frame (F) of the Rigid Transform block.



Invalid Rigid Transform Uses

In this section...
“Rigidity Loops are Invalid” on page 1-19
“Shorted Rigid Transform Blocks are Invalid” on page 1-20

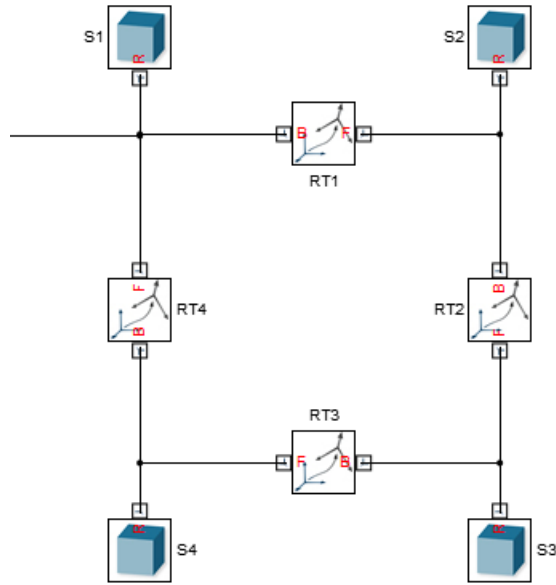
The following limitations apply to the Rigid Transform block.

Rigidity Loops are Invalid

A rigidity loop is a closed loop of Rigid Transform blocks. The loop contains one redundant Rigid Transform block that over-constrains the subsystem. If a rigidity loop is present, SimMechanics issues an error and the model does not simulate.

To remove the simulation error, disconnect one Rigid Transform block. This step removes the redundant constraint, and allows the model to simulate.

The following figure shows a rigidity loop. The loop contains four Rigid Transform blocks directly connected to each other.



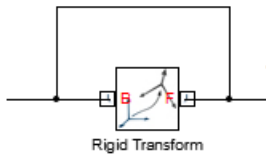
Shorted Rigid Transform Blocks are Invalid

A shorted Rigid Transform block contains a direct connection line between base (B) and follower frames (F). The connection line makes the two port frames coincident in space (see “Frame Line Identifies a Single Frame” on page 1-16). However, the Rigid Transform block enforces a spatial transformation that translates or rotates one port frame relative to the other. The result is a conflict in the frame definition.

If a shorted Rigid Transform block is present, SimMechanics issues an error and the model does not simulate. The error remains even if the Rigid Transform block specifies no rotation and no translation.

To remove the simulation error, delete the direct connection line between base and follower frame ports of the Rigid Transform block.

The following figure shows a shorted Rigid Transform block.



Add and Transform Frames

In this section...
“Workflow” on page 1-22
“Build Model” on page 1-23
“Specify Transformation Parameters” on page 1-24
“Visualize Frames” on page 1-25
“Save Model” on page 1-26

SimMechanics models are based on the concept of frames. Frames provide rigid bodies position and orientation in space. Blocks from the Body Elements library contain a reference frame port that you must tie to the appropriate frame in a model. To add a frame to a model, you use the Rigid Transform block.

This example shows you how to add transform a frame to a model. Rotation and translation parameters determine the position and orientation of the frame with reference to a base frame. The following table provides the rotation and translation parameters for this example.

Transformation	Transformation Axis	Transformation Magnitude
Rotation	X-axis	30 deg
Translation	Y-Axis	5 m

Once you have completed the example, you can rigidly connect solids to the frames. See “Connect Solids to Frames” on page 1-27.

Workflow

Adding a frame to a model involves these steps:

- 1 Identify the desired position and orientation of the new frame.
- 2 Add a Rigid Transform block to the model. The block copies and transforms a base frame to obtain the new frame. The new frame is called *follower* frame.

- 3 Connect the base frame port of the Rigid Transform block to the frame you wish to use as reference. If a model contains no frames, add a Reference Frame or World Frame block. The blocks provide an ultimate reference frame that you can use as a starting point.
- 4 In the dialog box of the Rigid Transform block, specify the rotation and translation parameters required to bring the base frame into coincidence with the follower frame.

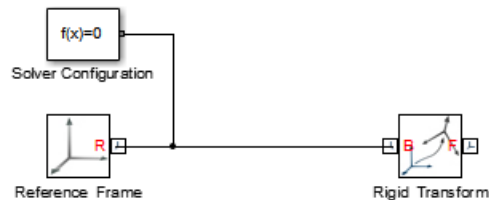
Build Model

To proceed with the example, create a block diagram to represent one reference frame. Then, use the Rigid Transform block to add a second frame. Adjust the rotation and translation parameters of the block to modify the spatial position and orientation of the frame.

- 1 Open a new Simulink[®] model.
- 2 Add the following blocks to the model.

Block	Library	Quantity
Reference Frame	Frames and Transforms	1
Rigid Transform	Frames and Transforms	1
Solver Configuration	Simscape [™] Utilities	1

- 3 Connect the blocks according to the following figure.



The model contains two frames. The Reference Frame block provides a starting frame. The Rigid Transform block transforms the reference frame to create a second frame. The base (B) frame port of the block identifies the frame relative to which you specify the transformation. The follower (F) frame port identifies the frame that you create with the Rigid Transform block.

Because you have not yet specified rotation or translation parameters in the Rigid Transform block, the two frames are *coincident* in space. The frames share the same origin and the frame axes point in the same directions.

Specify Transformation Parameters

Specify rotation and translation parameters in the Rigid Transform dialog box to spatially separate the two frames.

- 1 Double-click the Rigid Transform block.
- 2 In the **Rotation Method** drop-down list, select **Standard Axis**.
- 3 In the **Axis** drop-down list, select **+X**.
- 4 In **Angle**, enter 30.

Rotation		
Method	Standard Axis ▼	
Axis	+X ▼	
Angle	30	deg ▼


- 5 Expand the **Translation** menu.
- 6 In **Method**, select **Standard Axis**.
- 7 In **Axis**, select **+Y**.
- 8 In **Offset**, enter 5 and click **OK**.

Translation		
Method	Standard Axis ▼	
Axis	+Y ▼	
Offset	5	m ▼

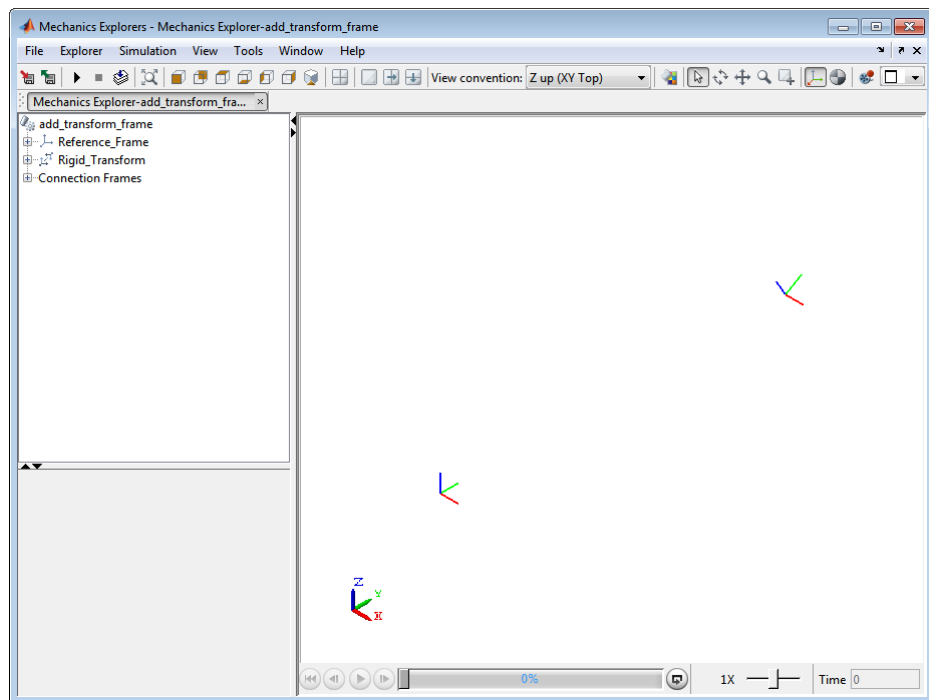
Visualize Frames

Update the model. On update, Mechanics Explorer opens with an empty visualization pane. You can highlight the frames in the visualization pane.

1 On the Simulink toolbar, click **Simulation > Update Diagram**.

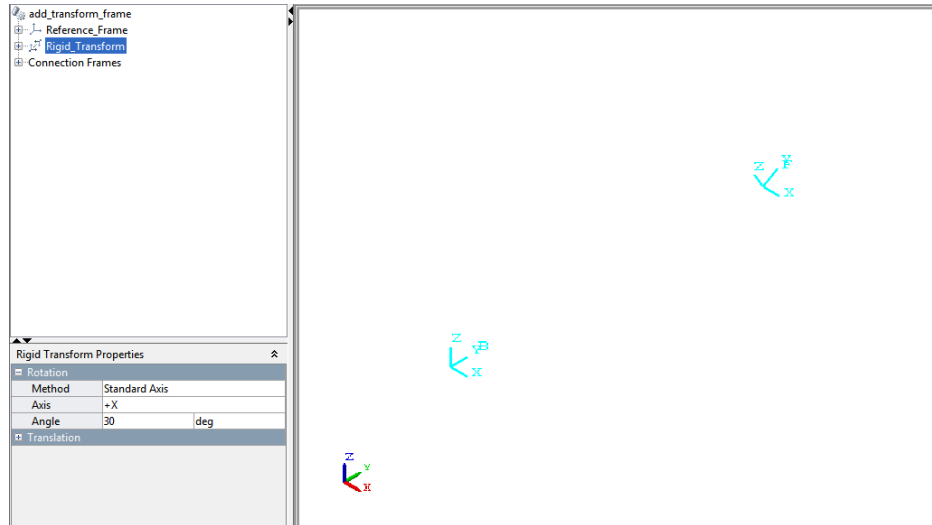
2 On the Mechanics Explorer toolbar, click the frame icon .

Mechanics Explorer displays the two frames in the model.



3 In the tree view pane, click **Rigid_Transform**.

Mechanics Explorer highlights the base and follower frames of the Rigid Transform block. The frames coincide with the two previous frames.



Confirm that translation and rotation parameters are correct:

- Is rotation about the correct axis (+X)?
- Does the magnitude of the rotation appear correct (+30 deg relative to base frame)?
- Is translation along the correct axis (+Y)?
- Does the magnitude of the translation appear correct (+5 m relative to base frame origin)?

Save Model

Save the model in a convenient folder for use in the example, “Connect Solids to Frames” on page 1-27. Name the model `add_transform_frame`.

Connect Solids to Frames

In this section...

“Example Requirements” on page 1-28

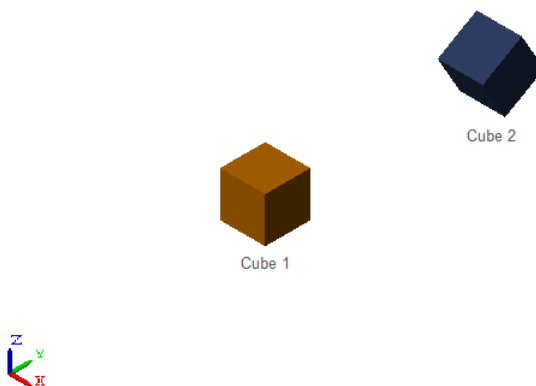
“Open and Modify Model” on page 1-28

“Add Shape and Color to Solids” on page 1-28

“Visualize Model” on page 1-29

Frames provide solids with a position and orientation in space. To add a solid to a model, simply connect the reference frame port of a Solid block to the desired frame in a model. The two frames become coincident in space. See “Frame Connection Rules” on page 1-16.

In this example, you rigidly connect two cubes to two spatially separated frames. A Rigid Transform block defines the relative spatial configuration of the two frames. Two Solid blocks define the geometry and color of the two cubes.



Example Requirements

This example requires completion of example “Add and Transform Frames” on page 1-22. The following rotation and translation parameters spatially separate the two cubes.

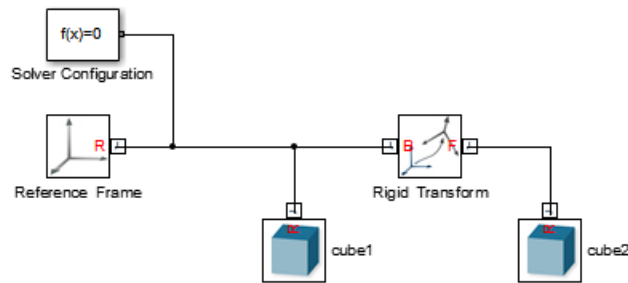
Open and Modify Model

To proceed with the example, open and modify the model `add_transform_frame` that you saved in example “Add and Transform Frames” on page 1-22. Modify the model as follows:

- 1 Add the following block to the model:

Block	Library	Quantity
Solid	Body Elements	2

- 2 Connect and name the blocks according to the following figure.



Add Shape and Color to Solids

Specify solid geometry and color to visually distinguish the two solids during visualization with Mechanics Explorer. Use the Solid block to specify geometry and color parameters. The example uses a simple cube shape for each solid. Different colors help identify cube1 and cube2 shapes.


- 1 Double-click each **Solid** block.
- 2 Expand **Geometry**.

- 3 In the **Shape** drop-down list, select **Brick**.
- 4 Verify that **Dimensions** contains the vector $[1 \ 1 \ 1]$.

Geometry		
Shape	Brick	
Dimensions	$[1 \ 1 \ 1]$	m

- 5 Expand **Graphic**.
- 6 Expand **Visual Properties**.
- 7 Enter the appropriate $[R \ G \ B]$ vector for each solid and click **OK**.

Solid	$[R \ G \ B]$	Color
cube1	$[0.8 \ 0.45 \ 0]$	Orange
cube2	$[0 \ 0.4 \ 0.9]$	Blue

Graphic	
Type	From Geometry
Visual Properties	Simple
Color	$[0.8 \ 0.45 \ 0]$ 
Opacity	1.0

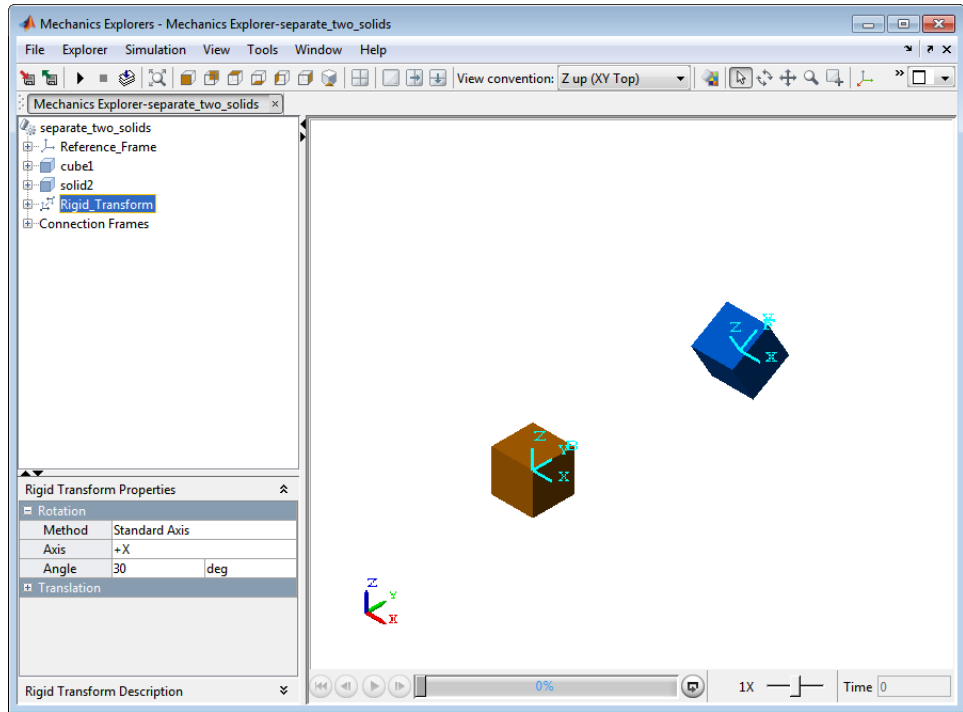
Visualize Model

Update the model. On update, Mechanics Explorer opens with a 3-D display of the two spatially separated solids.

To update and visualize the model:

- 1 On the Simulink Editor menu bar, select **Simulation > Update Diagram**.
- 2 On the Mechanics Explorer tree view pane, click **Rigid_Transform**.

The visualization pane highlights the base and follower frames of the Rigid Transform block. The base frame coincides with the reference frame of cube1. The follower frame coincides with the reference frame of cube2.



Model Frame Tree

In this section...

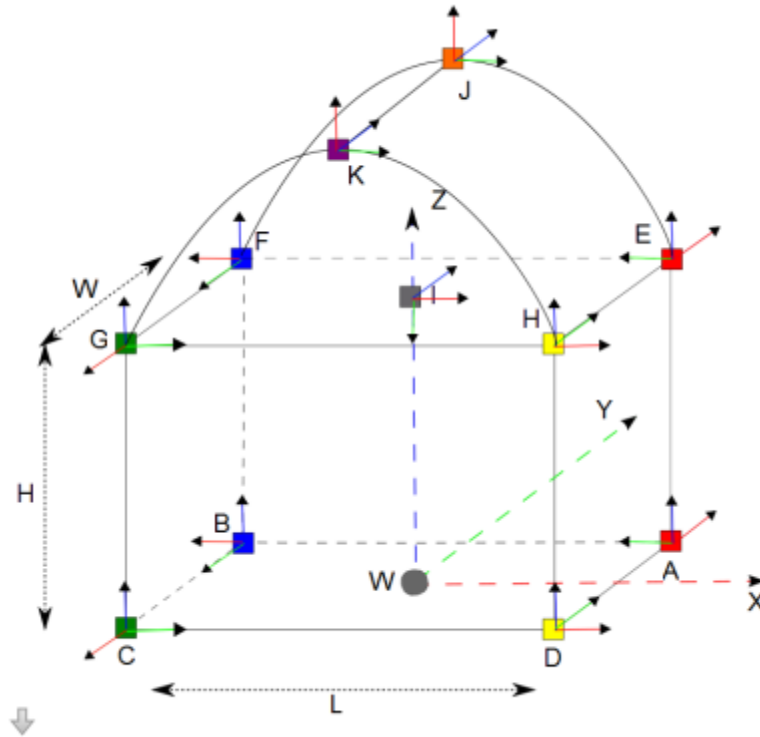
- “Build Model” on page 1-32
- “Define Model Parameters” on page 1-33
- “Add World Frame” on page 1-34
- “Add Bottom Plane Frames” on page 1-36
- “Add Top Plane Frames” on page 1-40
- “Add Arch Frames” on page 1-44
- “Save Model” on page 1-47

Complex rigid bodies can contain a large number of frames that specify the position and orientation of joints, forces and torques, and motion sensors. The frames form a tree with a structure that can range from flat to hierarchical. The block diagram of the frame tree consists of a chain of Rigid Transform blocks. Each block adds one new frame to the model.

In this example, you use Rigid Transform blocks to build a box-shaped frame tree. The World frame provides an ultimate reference frame for the model. Eight frames are located at the cube vertices, one frame at the center of the top face, and two frames at the top of the two arches.

The model proceeds in four stages:

- 1** Add World Frame (W).
- 2** Define vertex frames of bottom cube face (A-D) with reference to the World frame (W).
- 3** Define vertex and center frames of top cube face (E-I) with reference to the bottom face frames (A-D, W).
- 4** Define upper arch frames (J-K) with reference to the center frame of the top cube plane (I)



Build Model

To proceed with the example:

- 1 Start a new Simulink model.
- 2 Add the following block to the model.

Block	Library	Quantity	Purpose
Solver Configuration	Simscape Utilities	1	Enable model assembly, simulation, and visualization

- 3 Save the model in a readily accessible folder. Name the model `frame_tree`.

Define Model Parameters

The frame tree consists of a cube with an arch, with twelve frames at either plane vertices or centers. It is convenient to parameterize the translation quantities of the Rigid Transform blocks in terms of frame cube dimensions. The following table lists the relevant dimensions in this example.

Parameter	Frame Cube Dimension
L	Length
W	Width
H	Height

In the **Translation** fields of the Rigid Transform block, enter the appropriate parameter. You can define the numerical value of each parameter in the model workspace using the Simulink Model Explorer utility:

- 1 On the Simulink menu bar, click **Tools > Model Explorer**.
- 2 In the Model Hierarchy pane, double-click `frame_tree`.
- 3 Click **Model Workspace**.
- 4 In the **Workspace data** pane, enter the following definitions:

```
% Size of Cube
L = 12;
W = 10;
H = 8;
```

- 5 Click **Apply**.

Note The following sections rely on the preceding parameter definitions to provide the numerical values of translation quantities in the Rigid Transform block.

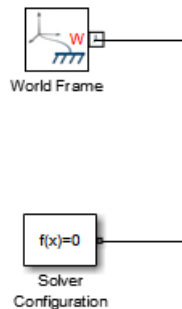
Add World Frame

The World frame provides an ultimate reference frame in the model. You can use this frame as a reference to specify the vertex frames of the bottom cube plane. To add the World frame:


- 1 Add the following block to the model.

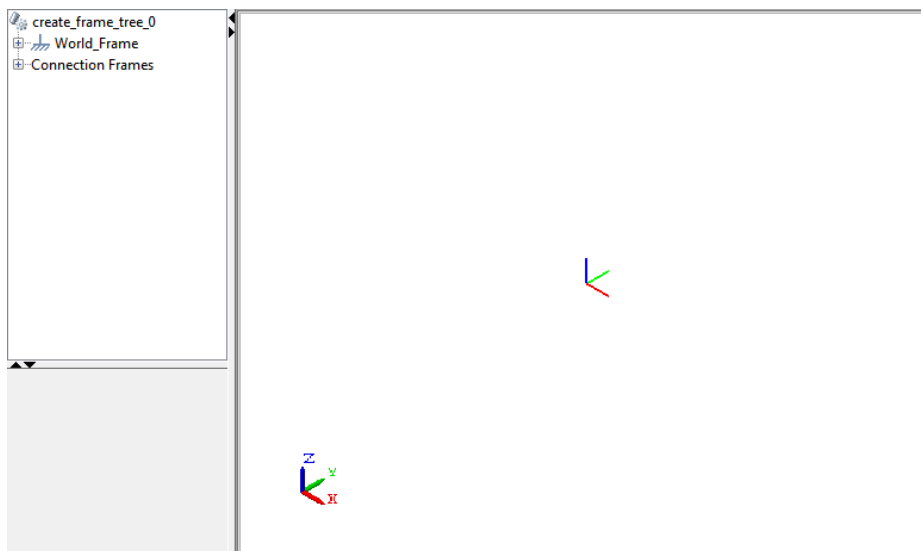
Block	Library	Quantity	Purpose
World Frame	Frames and Transforms	1	Provide inertial reference frame to the model. The frame defines the top level of the frame tree.

- 2 Connect the blocks according to the following figure.



- 3 On the Simulink menu bar, click **Simulation > Update Diagram**.

- 4 On the Mechanics Explorer toolbar, click the frame visibility icon .



5 In the Mechanics Explorer tree view pane, click **World_Frame**.



Add Bottom Plane Frames

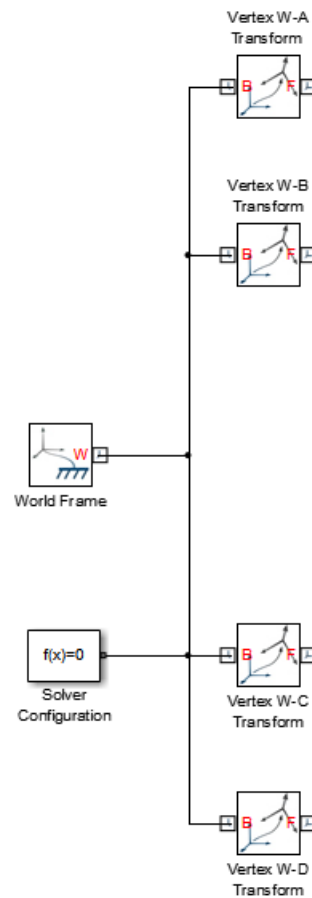
The vertex frames of the bottom cube face form the second level of the frame tree. Use Rigid Transform blocks to define the frames relative to World. Later, you can use the bottom face frames to define the top face frames.

To add the bottom face frames:

- 1 Add the following block to the model.

Block	Library	Quantity	Purpose
Rigid Transform	Frames and Transforms	4	Add new frames

- 2 Connect and name the blocks according to the following figure.



3 Double-click each Rigid Transform block.

4 In the dialog box, specify the parameters according to the following list.

Vertex W-A Transform

Rotation

- **Method** — Standard Axis

- **Axis** — +Z
- **Angle** — 90
- **Units** — deg

Translation

- **Method** — Cartesian
- **Offset** — [L/2 W/2 0]
- **Units** — cm

Vertex W-B Transform

Rotation

- **Method** — Aligned Axes
- **Pair 1:**
 - **Follower** — +X
 - **Base** — -X
- **Pair 2:**
 - **Follower** — +Y
 - **Base** — -Y

Translation

- **Method** — Cartesian
- **Offset** — [-L/2 W/2 0]
- **Units** — cm

Vertex W-C Transform

Rotation

- **Method** — Standard Axis
- **Axis** — +Z
- **Angle** — 270

- **Units** — deg

Translation

- **Method** — Cartesian
- **Offset** — $[-L/2 \ -W/2 \ 0]$
- **Units** — cm

Vertex W-D Transform


Rotation

- **Method** — None

Translation

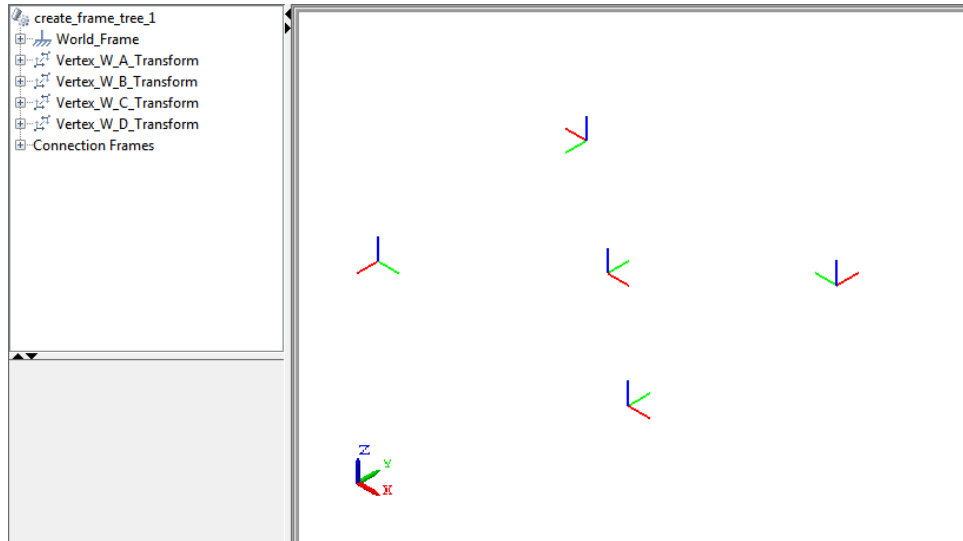
- **Method** — Cartesian
- **Offset** — $[L/2 \ W/2 \ 0]$
- **Units** — cm

5 On the Simulink menu bar, click **Simulation > Update Diagram**.

6 On the Mechanics Explorer toolbar, click the frame visibility icon .

7 On the Mechanics Explorer toolbar, click the isometric view icon .

Mechanics Explorer displays the vertex frames of the bottom cube plane.



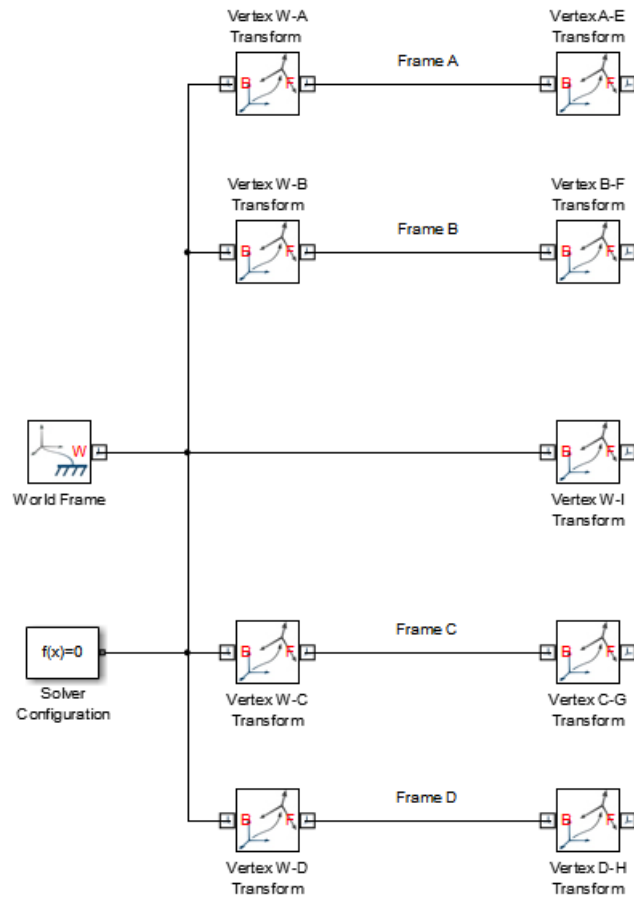
Add Top Plane Frames

You can now define the top plane frames with reference to their bottom plane neighbors. Add another set of Rigid Transform blocks to the model, and specify the appropriate rotation and translation parameters for each block.

- 1 Add the following blocks to the model.

Block	Library	Quantity	Purpose
Rigid Transform	Frames and Transforms	5	Add new frames

- 2 Connect and name the blocks according to this figure.



3 Double-click each of the five new Rigid Transform blocks.

4 In the dialog box, specify the parameters according to the following list.

Vertex A-E Transform

Rotation

- **Method** — None

Translation

- **Method** — Standard Axis
- **Axis** — +Z
- **Offset** — H
- **Units** — cm

Vertex B-F Transform

Rotation

- **Method** — None

Translation

- **Method** — Standard Axis
- **Axis** — +Z
- **Offset** — H
- **Units** — cm

Vertex W-I Transform

Rotation

- **Method** — Aligned Axes
- **Pair 1:**
 - **Follower** — +Y
 - **Base** — -Z
- **Pair 2:**
 - **Follower** — +Z
 - **Base** — +Y

Translation

- **Method** — Standard Axis
- **Axis** — +Z

- **Offset** — H
- **Units** — cm.

Vertex C-G Transform

Rotation

- **Method** — None

Translation

- **Method** — Standard Axis
- **Axis** — +Z
- **Offset** — H
- **Units** — cm

Vertex D-H Transform

Rotation

- **Method** — None

Translation

- **Method** — Standard Axis
- **Axis** — +Z
- **Offset** — H
- **Units** — cm

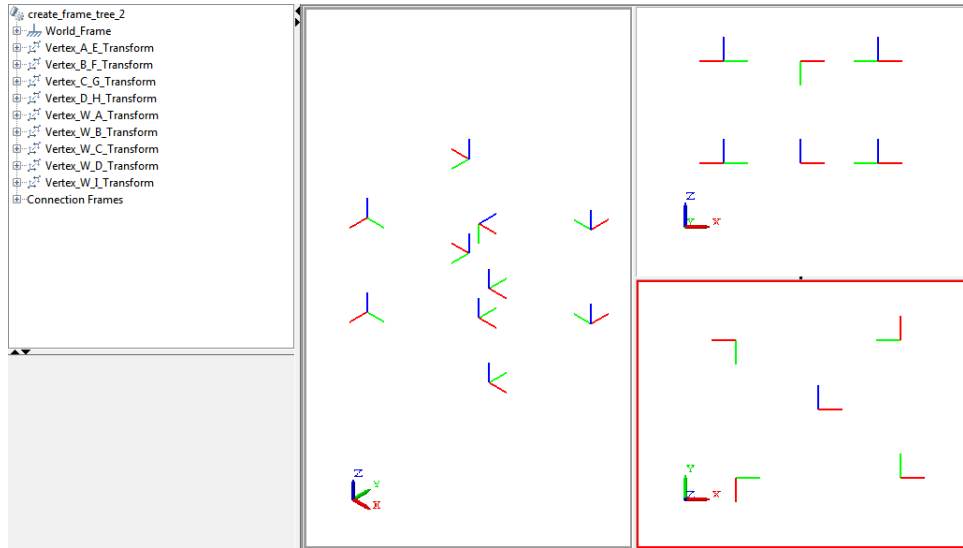
5 On the Simulink menu bar, click **Simulation > Update Diagram**.

6 On the Mechanics Explorer toolbar, verify that the frame visibility icon



is toggled to on.

Mechanics Explorer displays the vertex frames of the bottom and top cube planes.



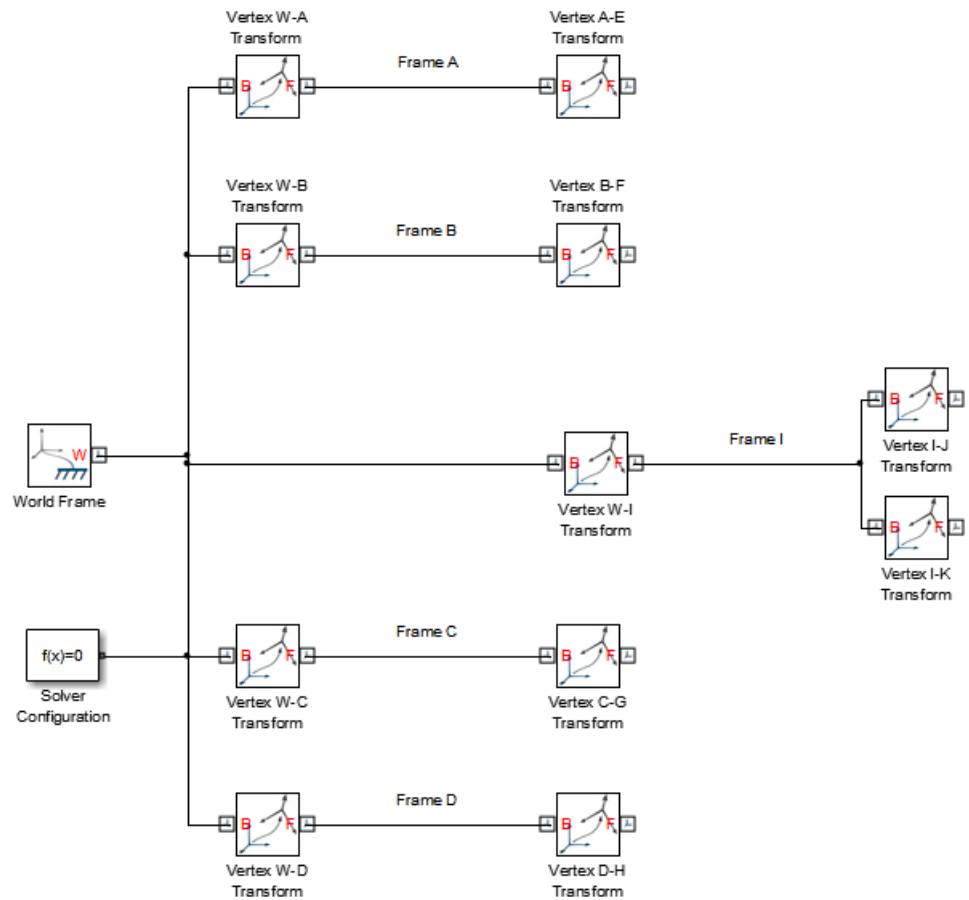
Add Arch Frames

Finally, add the two arch frames to the model. Use the center frame of the top cube plane as the Rigid Transform base frame (frame I).

- 1 Add the following block to the model.

Block	Library	Quantity	Purpose
Rigid Transform	Frames and Transforms	2	Add new frames

- 2 Connect and name the blocks according to the following figure.



3 Double-click each of the two new Rigid Transform blocks.

4 On the dialog box, specify the parameters according to the following list.

Vertex I-J Transform

Rotation

- **Method** — Standard Axis
- **Axis** — +Z

- **Angle** — -90
- **Units** — deg

Translation

- **Method** — Cylindrical
- **Radius** — $L/2$
- **Units** — cm
- **Theta** — -90
- **Units** — deg
- **Z Offset** — $W/2$
- **Units** — cm

Vertex I-K Transform


Rotation

- **Method** — Standard Axis
- **Axis** — +Z
- **Angle** — -90
- **Units** — deg

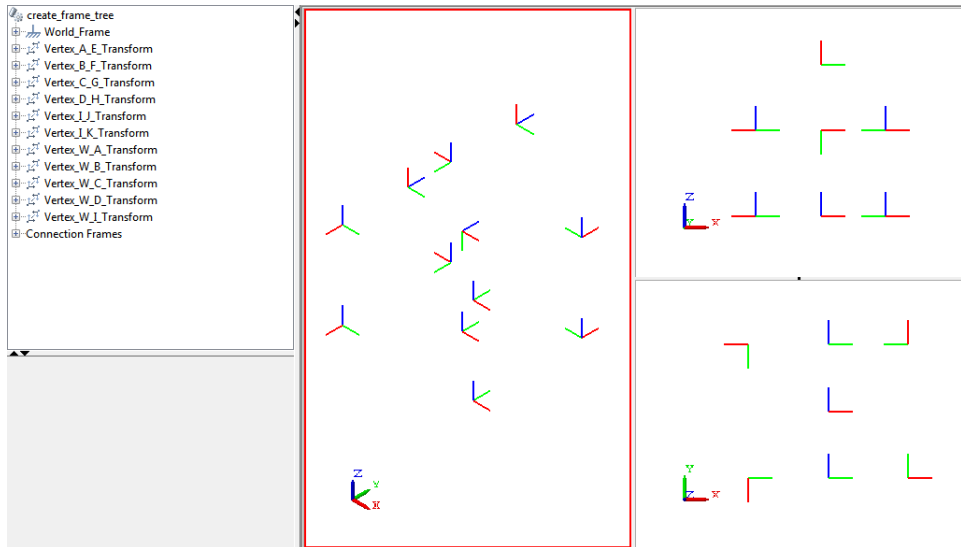
Translation

- **Method** — Cylindrical
- **Radius** — $L/2$
- **Units** — cm
- **Theta** — -90
- **Units** — deg
- **Z Offset** — $-W/2$
- **Units** — cm

5 On the Simulink menu bar, click **Simulation > Update Diagram**.

- 6 On the Mechanics Explorer toolbar, verify that the frame visibility icon  is toggled to on.

Mechanics Explorer displays the vertex frames of the bottom and top cube planes.



Save Model

Save the model in a convenient folder for use in example “Add Graphics to Frames” on page 1-48. Name the file `frame_tree`.

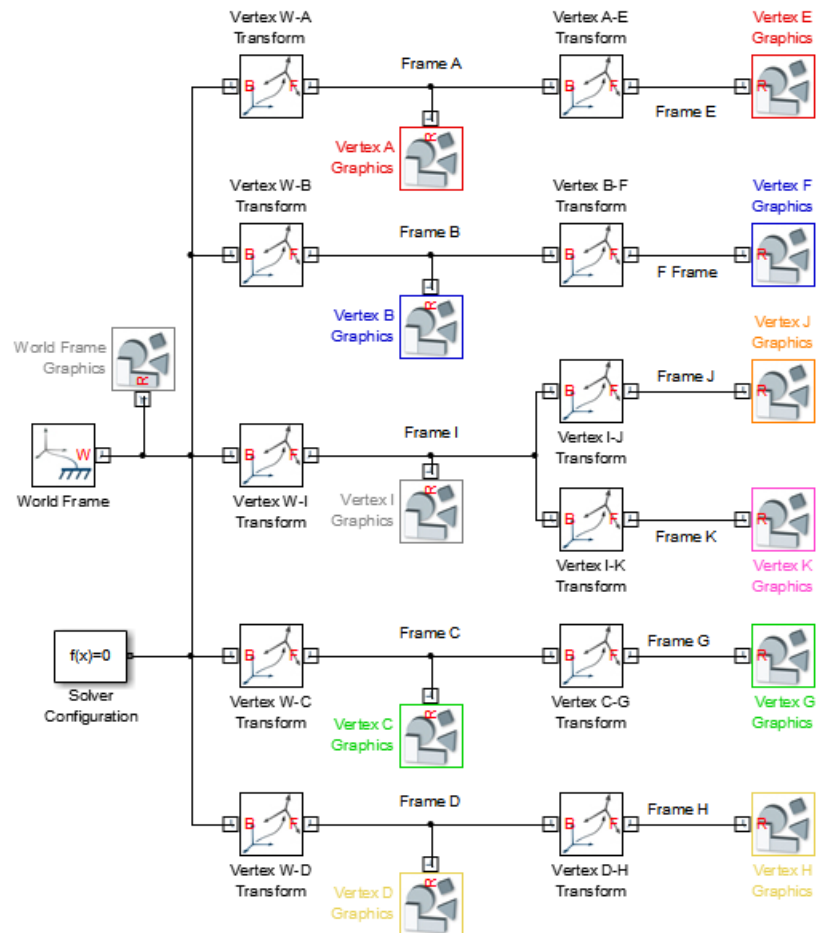
Add Graphics to Frames

To aid frame visualization, you can connect Graphic blocks to frames in a model. The block provides a selection of colored icons that you can use to easily identify frames. To add graphic icons to the `frame_tree` model that you created in example “Model Frame Tree” on page 1-31, follow these steps:

- 1** Open the model `frame_tree` that you created in example “Model Frame Tree” on page 1-31.
- 2** Add the following block to the model.

Block	Library	Quantity	Purpose
Graphic	Body Elements	12	Add graphic icon to frame

- 3** Connect and name the blocks according to the following figure.



4 Double-click each Graphic block.

5 In the dialog box, specify parameters according to the following table.

Graphic Block	Color	Shape	Size
World Frame Graphics	[0.4 0.4 0.4]	Sphere	25
Vertex I Graphics		Cube	
Vertex A Graphics	[1.0 0.0 0.0]		
Vertex E Graphics			
Vertex B Graphics	[0.0 0.0 1.0]		
Vertex F Graphics			
Vertex C Graphics	[0.0 0.6 0.2]		
Vertex G Graphics			
Vertex D Graphics	[1.0 1.0 0.0]		
Vertex H Graphics			
Vertex J Graphics	[1.0 0.4 0.0]		
Vertex K Graphics	[0.6 0.0 0.6]		

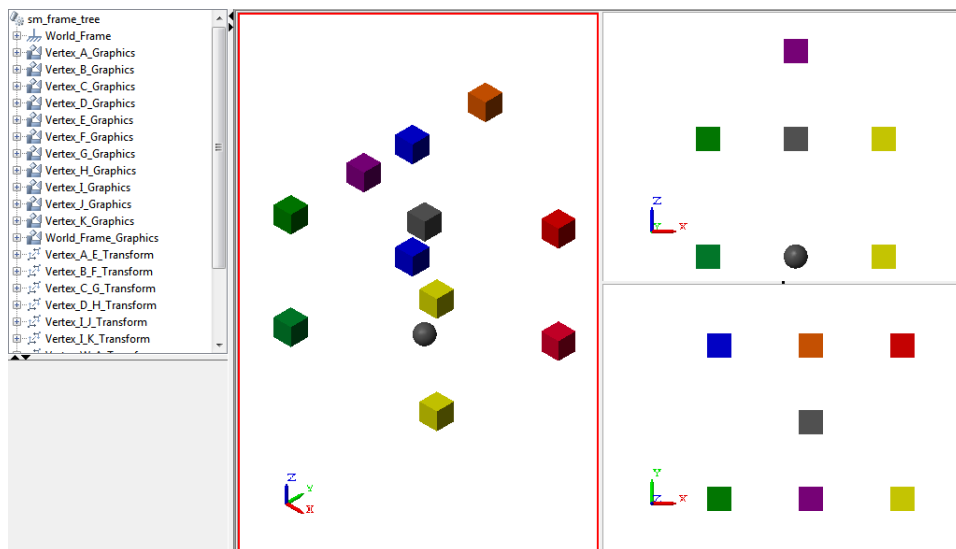
6 On the Simulink menu bar, click **Simulation > Update Diagram**.

7 On the Mechanics Explorer toolbar, verify that the frame visibility icon



is toggled to on.

Mechanics Explorer displays the vertex frames of the bottom and top cube planes.



Measure the Motion State of a Frame

In this section...
“Sense Motion with Joint Blocks” on page 1-52
“Sense Motion with the Transform Sensor Block” on page 1-53

You can measure the relative state of motion between two frames. To measure the frame state, you use one of the following blocks:

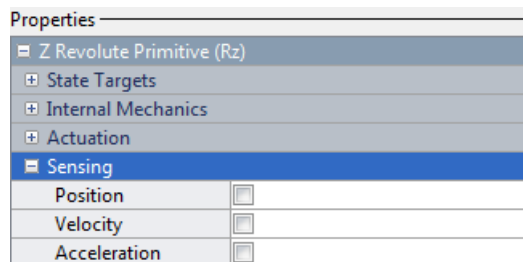
Library	Block
Joints	All except Weld joint
Frames and Transforms	Transform Sensor

Sense Motion with Joint Blocks

Joint blocks measure the relative state of motion between the joint base and follower frames. Motion parameters include the position, velocity and acceleration of each joint primitive.

To select the motion parameters you wish to sense:

- 1 Double-click the Joint block you wish to sense.
- 2 In the dialog box, expand the **Sensing** menu.
- 3 Click the box for each parameter you wish to sense and click **OK**.



- 4 To plot the time-dependence of a parameter, connect the physical signal output of the parameter to a Simulink Scope block through a PS-Simulink Converter block.

Sense Motion with the Transform Sensor Block

The Transform Sensor block measures the relative state of motion between two frames of your choice. The block provides a selection of parameters not available in other blocks, including three rotation methods:

- Quaternion — Four-dimensional object used to describe the rotation of a frame. Quaternions are not vulnerable to the loss of a degree of freedom due to gimbal lock. The quaternion has the form

$$q = a + bi + cj + dk$$

a , b , c , and d are real numbers. i , j , k satisfy the relationships:

$$i^2 = j^2 = k^2 = -1$$

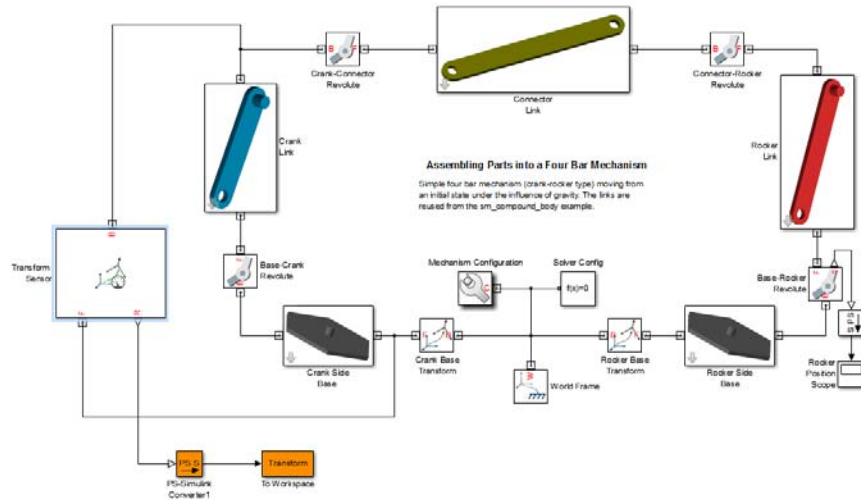
$$k = -1$$

During simulation, SimMechanics outputs the quaternion of a rotation as a four-element vector..

- Axis-Angle — Direction of rotation axis and angle of rotation about the axis. During simulation, SimMechanics outputs:
 - Rotation axis as a three-element vector.
 - Rotation angle as a scalar.
- Rotation Transform — Cosine matrix used to describe the rotation of a frame. During simulation, SimMechanics outputs the rotation transform as a 3-by-3 matrix.

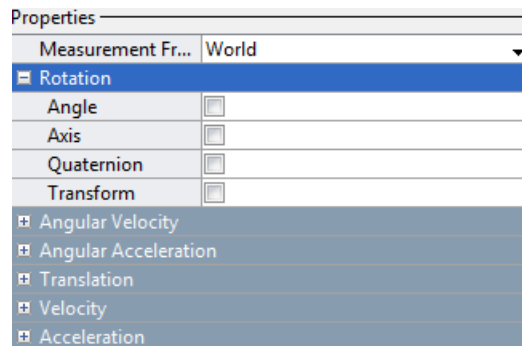
To sense motion with the Transform Sensor block:

- 1 Connect the base and follower frame ports of the Transform Sensor block to the frames you wish to sense.



2 Double-click the Transform Sensor block.

3 In the dialog box, expand the menu for the type of motion you wish to sense.



4 Click the box for each motion parameter you wish to sense and click **OK**.

5 To plot the time-dependence of a parameter, connect the physical signal output of the parameter to an appropriate Simulink Sinks block through a PS-Simulink Converter block.

Rigid Bodies

- “Properties of Rigid Bodies” on page 2-2
- “Solid Shapes” on page 2-10
- “Mass and Inertia” on page 2-14
- “About Extrusion and Revolution Geometries” on page 2-18
- “Extrusion and Revolution Cross-Section Rules” on page 2-23
- “Model a Simple Binary Link” on page 2-31
- “Model a Hollow Cone” on page 2-43
- “Model a Dome” on page 2-54
- “Model an I-Beam” on page 2-66
- “Model a Box Beam” on page 2-78
- “Model a Compound Binary Link” on page 2-90

Properties of Rigid Bodies

In this section...
“Flexible and Rigid Bodies” on page 2-2
“Geometry” on page 2-3
“Inertia” on page 2-7
“Graphic” on page 2-8

Bodies are the basic building blocks in any multibody system. You connect them with joints and other kinematic constraints to create an articulated system that performs a specialized function or moves along a specified path. The bodies that comprise these systems can be flexible or rigid. In SimMechanics, all bodies assume ideal rigid behavior.

Rigid bodies contain a set of parameters that determine their dynamic behavior and visual appearance. Three parameter sets of parameters characterize SimMechanics rigid bodies:

- Geometry
- Inertia
- Graphic

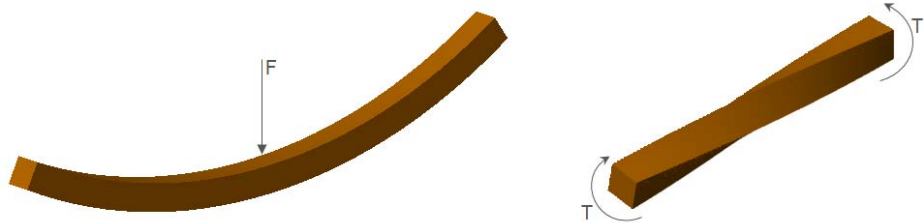
The three solid parameters are the input to the Solid block. In conjunction with frames, the three solid parameters completely define a SimMechanics rigid body.

Flexible and Rigid Bodies

Rigid bodies experience zero deformation regardless of applied force or torque. The stiffness of a rigid body is infinite, and its shape remains constant in all dynamic conditions. A rigid body cannot shrink, expand, twist, or bend.

Real bodies are not rigid. An applied force or torque always deforms a real body, but the deformation can be very small. Common structural materials have a large stiffness constant, and deformation remains negligible in

comparison with the scale of motion. On a macroscopic scale, these materials behave as perfectly rigid bodies. The rigid body approximation applies.



All SimMechanics bodies are rigid bodies. Forces and torques can change the state of motion of a rigid bodies, but not their shape or size. The rigid body approximation provides an accurate simplification for most mechanical parts under normal operating conditions.

Geometry

Geometry is one of three rigid body parameters you can specify in a SimMechanics Solid block. SimMechanics geometries are three-dimensional, and range from simple to complex. Simple geometries include predefined SimMechanics.

Complex geometries contain a greater level of detail. The most complex SimMechanics geometries are **General Extrusion** and **Revolution**. These geometries require a MATLAB® matrix specifying the set of [x y] coordinates that define a constant cross-sectional shape.

For a list of SimMechanics geometries, see “Solid Shapes” on page 2-10.

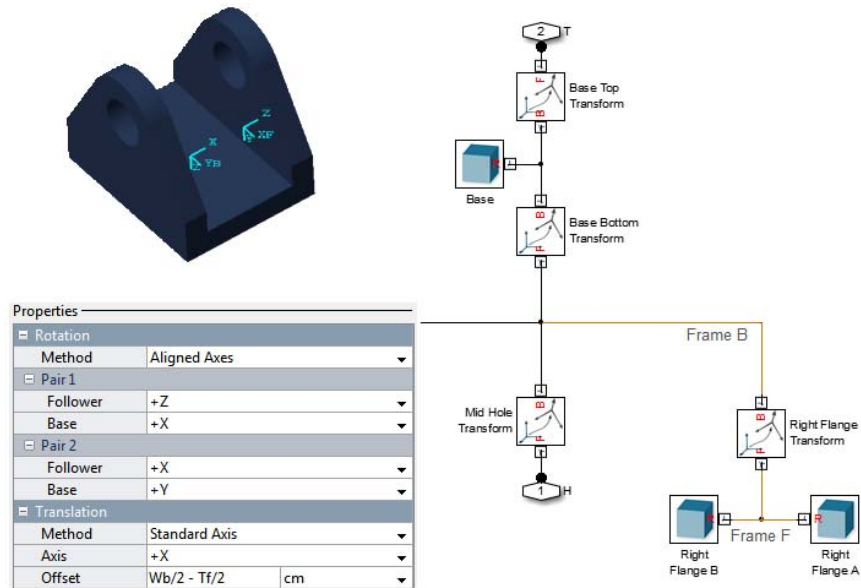
Compound Rigid Bodies

A rigid body can contain one or multiple Solid blocks. Rigid bodies that contain multiple Solid blocks are known as *compound* rigid bodies. Most SimMechanics rigid bodies are compound rigid bodies.

Each Solid block represents an elementary solid segment. Multiple solid segments connect through frames to form a cohesive whole that behaves as

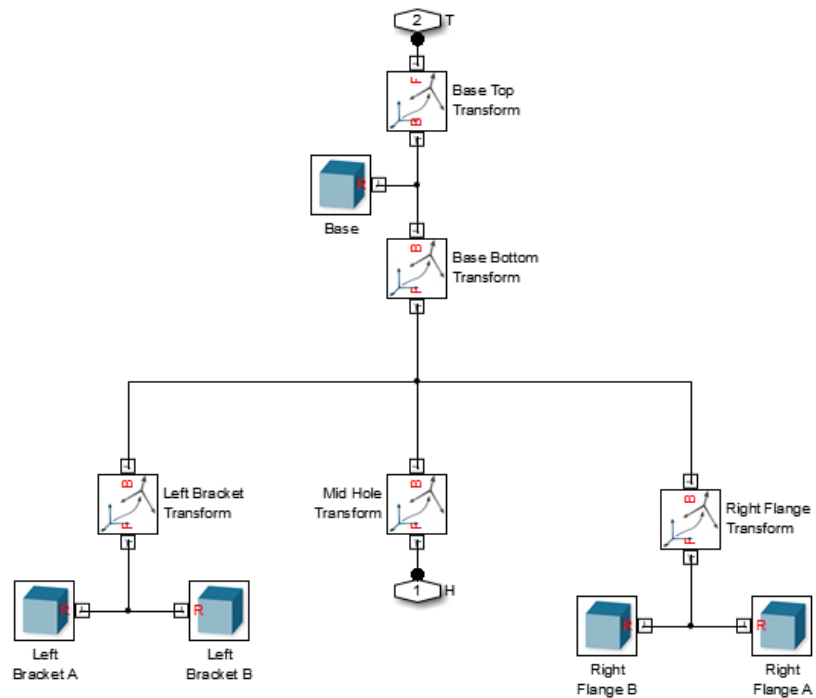
a single unit. In a compound rigid body, the Rigid Transform block holds connects solid elements into a single unit.

In the following figure, a Rigid Transform block connects solids Right Flange A and Right Flange B to solid Base.



In a compound rigid body, Rigid Transform blocks form a hierarchical network of frames. The frame hierarchy defines the configuration of solid segments within a rigid body. In simple cases, the frame hierarchy is flat, and all frames exist at a single hierarchical level. In more complex cases, the frame hierarchy can contain several hierarchical levels.

In the example model `sm_wing_landing_gear`, rigid body **Axle Bracket** contains a frame hierarchy with multiple hierarchical levels. In this block diagram, frames for the Left Bracket, Mid-Hole, and Right Flange are defined as *dependent* frames of the Base reference frame.



Parameterization of Geometry Dimensions

You can parameterize the shape of a Solid block. For example, you can specify the radius of a sphere in terms of parameter R . Parameterizing solid dimensions is useful in compound rigid bodies. You can define the shape and size of each solid element in terms of common parameters, and then specify the numerical value of the parameter in the model workspace or in a subsystem dialog box.

Solid parameterization plays an important role in geometries **General Extrusion** and **Revolution**. The $[x y]$ coordinates of a cross-sectional shape change each time you update solid dimensions. Manually editing these coordinates slows down the updating process. When you parameterize the $[x y]$ coordinates in terms of dimensional variables, you need only edit the dimensional variables themselves.

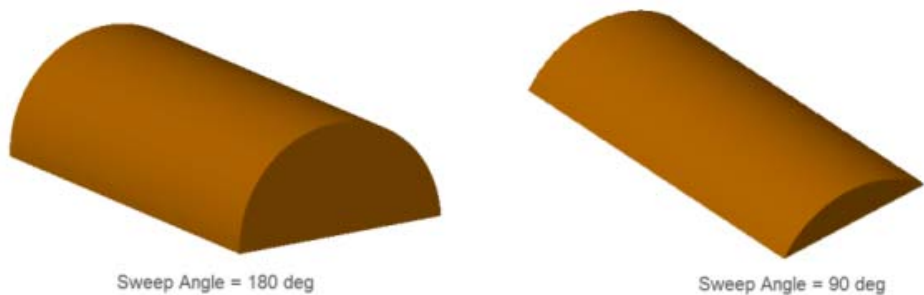
If you know the parametric equations for common 2-D shapes, specifying the [x y] coordinates of a cross-section is easier. For example, you can parameterize a circular cross-section in terms of the circle radius (R) and sweep angle (θ):

$$x=R\cos(\theta)$$

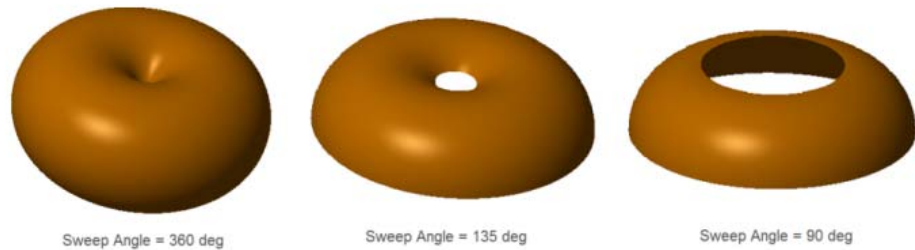
$$y=R\sin(\theta)$$

If the sweep angle is smaller than 360° , the parameterization represents a slice of a circle, rather than a whole circle. You can vary the radius of the circle and length of extrusion (or angle of revolution) to quickly change the geometry of a rigid body. You can parameterize other cross-sectional shapes in terms of meaningful dimensional parameters, and then edit the dimensional parameters to quickly update solid geometry.

The following figure illustrates two extrusions, both with circular cross-sections but different values for the sweep angle. Changing between the two geometries requires that you update only the numerical value of the sweep angle. You do not need to modify the cross-section definition.



The following figure illustrates three solids of revolution, each with the circular cross-section previously defined. Changing between the three geometries requires a simple change in sweep angle value at the mask level.



For more information on how to build parameterized models of custom shapes using SimMechanics geometries **General Extrusion** and **Revolution**, see:

- “Model a Dome” on page 2-54
- “Model a Hollow Cone” on page 2-43
- “Model a Dome” on page 2-54
- “Model a Box Beam” on page 2-78

Inertia

Inertia is the resistance of a rigid body to a change in linear or angular momentum. For a non-rotating rigid body, the important inertial parameter is the inertial mass. According to Newton’s law of acceleration, a rigid body’s inertial mass resists any change in motion that arises from a net force or torque:

$$\vec{F} = m\vec{a}$$

SimMechanics blocks **Solid** and **Inertia** accept a scalar input for the inertial mass parameter. You can input the inertial mass directly, or provide the mass density and let SimMechanics automatically compute the inertial mass based on rigid body geometry.

For rotating rigid bodies, the important inertial parameter is the inertia tensor. This tensor includes moments and products of inertia that characterize how resistant a rigid body is to any change in rotational velocity. Each time you use a **Solid** or **Inertia** block to represent a rigid body, you can select an automatic inertia calculation mode. Selecting this mode instructs the

SimMechanics computational engine to compute the inertia tensor based on mass properties and geometry specification.

Automatic inertia calculation imposes no geometry restrictions. You can use this mode with any valid geometry specification, standard or custom. You can also use this mode with any mass properties—including negative mass or density. Specifying rigid bodies with negative mass provides one way to create hollow rigid body sections.

The inertia tensor is a tensor of rank two. It is a matrix with nine elements, six of which are independent of each other:

$$\begin{matrix}
 I_{xx} & I_{xy} & I_{xz} \\
 I_{yx} & I_{yy} & I_{yz} \\
 I_{zx} & I_{zy} & I_{zz}
 \end{matrix}$$

The diagonal terms (I_{xx} , I_{yy} , I_{zz}) specify the moments of inertia. The three independent off-diagonal terms ($-I_{xy} = -I_{yx}$, $-I_{xz} = -I_{zx}$, $-I_{yz} = -I_{zy}$) specify the products of inertia.

The Solid and Inertia blocks accept a three-element row vector as input for **Moments of Inertia** and for **Products of Inertia**.

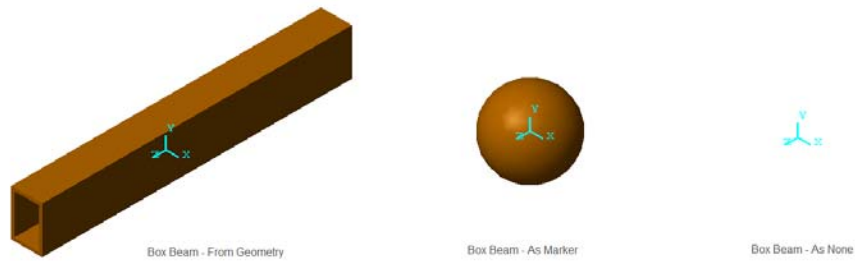
Inertia		
Type	Custom	
Mass	1	kg
Center of Mass	[0 0 0]	m
Moments of Ine...	[1 1 1]	kg*m^2
Products of Iner...	[0 0 0]	kg*m^2

Graphic

Graphic specifies how a rigid body appears in Mechanics Explorer. You have the choice to render the rigid body using its geometry, a simple marker, or nothing at all. Using solid geometry to render a rigid body provides a realistic display of that rigid body. Hiding a rigid body in Mechanics Explorer helps to clean up a cluttered screen during simulation, and it can help navigate

complex models. When the geometry of a rigid body is not important, but its position in space is, use a graphic marker.

The following figure illustrates a single rigid body, a box beam, displayed using **Graphic** settings From Geometry, Marker (using a spherical shape as a marker), and None. Each box beam instance contains a local reference frame that identifies its position and orientation in space. The local reference frame for each instance is highlighted in turquoise.



Graphic contains parameters for color specification of a rigid body. Color parameters include the defining RGB vector and opacity. Use color parameters to make rigid bodies easy to identify.

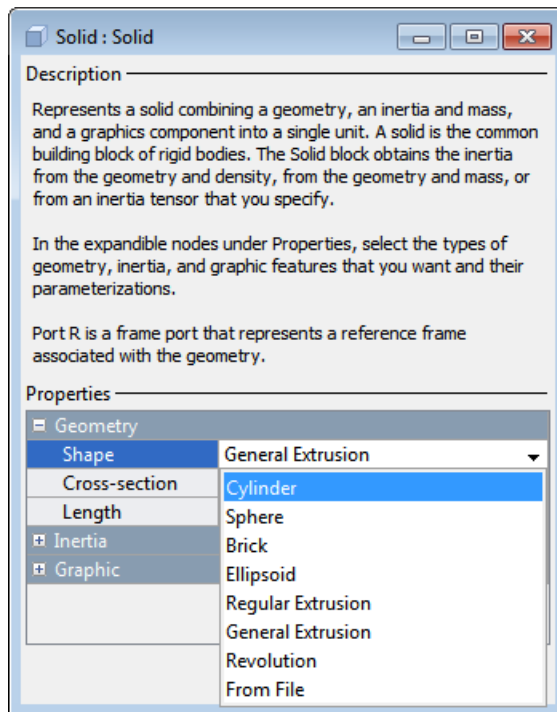
Solid Shapes

In this section...

“Simple Shapes” on page 2-10

“Advanced Shapes” on page 2-12

SimMechanics provides a set of shapes that you can use to represent rigid bodies. You can specify the shapes directly in the Solid block dialog box.

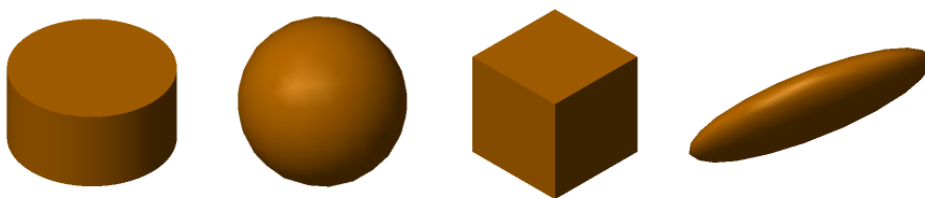


Simple Shapes

Shapes range from simple to advanced. Simple shapes require a small number of dimensional parameters. The following simple shapes are available.

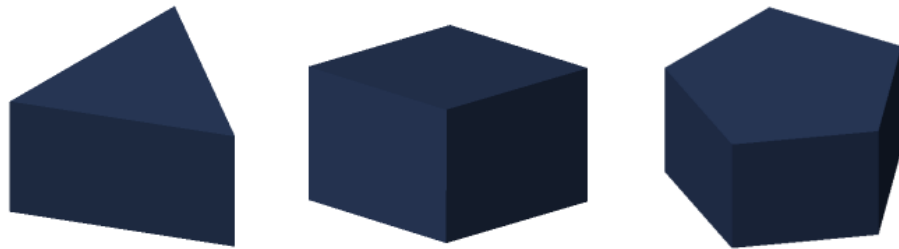
- **Cylinder** — Cylinder with custom dimensions, centroid at the solid reference frame origin, and symmetry axis along the solid reference frame z-axis.
- **Sphere** — Sphere with custom dimensions and center located at the solid reference frame origin.
- **Brick** — Brick with custom dimensions along the three Cartesian axes and centroid located at the block reference frame.
- **Ellipsoid** — Ellipsoid with custom dimensions with centroid located at the block reference frame.
- **Regular Extrusion** — Extruded solid with constant cross-section along the z-axis and centroid located at the block reference frame. The constant cross-section is a regular polygon with a custom number of sides.

Simple shapes are easier to use than advanced shapes. When modeling a rigid body, consider using a simple shape as a first approximation. After successful model assembly, you can add detail to the rigid body. The following figure shows the four simple shapes, ordered left to right: cylinder, Sphere, Brick, and Ellipsoid.



The **Regular Extrusion** shape is more versatile than other simple shapes. With this shape, you can model solids with constant cross sections. Cross-sections can have any number of sides, but all lengths and internal angles are equal.

The following figure shows a set of shapes you can model with the **Regular Extrusion** shape.



Advanced Shapes

Advanced shapes include:

- **General Extrusion** — Extruded solid with custom cross-section swept along the z-axis and centroid located at the block reference frame.
- **Revolution** — Solid of revolution with constant cross-section revolved about the z-axis and centroid located at the block reference frame.

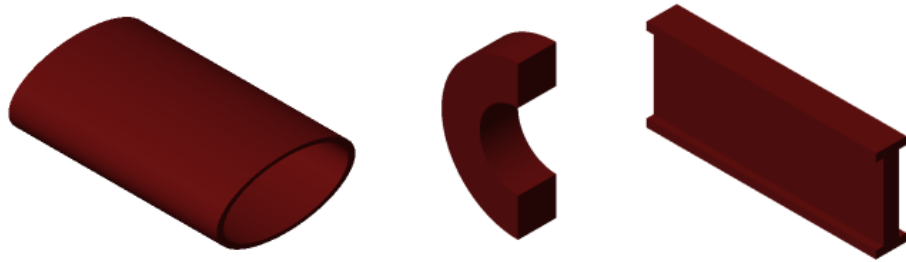
The shapes require a MATLAB cross-section matrix. To be valid, the matrix must observe a set of rules. See “Extrusion and Revolution Cross-Section Rules” on page 2-23.

General Extrusions

For extrusions with irregular cross-section, SimMechanics provides a **General Extrusion** geometry. This geometry is among the most versatile in SimMechanics. You can use it to model shapes with an increased level of detail.

This shape requires a MATLAB matrix that contains the cross-section coordinates. The matrix must follow a set of rules that are specific to the shape. See “Extrusion Rules” on page 2-26.

The following figure shows some shapes you can model with **General Extrusion**.



For General Extrusion examples, see:

- “Model an I-Beam” on page 2-66
- “Model a Box Beam” on page 2-78

Solids of Revolution

Solids that have a constant cross-section *about* an axis are solids of revolution. To model these solids, use the `Revolution` shape.

The `Revolution` shape requires a MATLAB matrix that contains the cross-section coordinates. The matrix must follow a set of rules specific to the `Revolution` geometry. See “`Revolution Rules`” on page 2-28.

The following figure shows some shapes you can model with `Revolution`.



Mass and Inertia

In this section...
“Relevant Blocks” on page 2-14
“Mass” on page 2-14
“Inertia” on page 2-15
“Import from CAD Assembly” on page 2-17

The inertial parameters of a rigid body influence its dynamic behavior in space. During simulation, SimMechanics uses the inertial parameters to solve the governing equations of motion for a multibody system. Without the parameters, simulation stops and SimMechanics issues a simulation error.

Two inertial parameters are relevant to multibody modeling:

- Mass — Inertial mass of a rigid body. The inertial mass expresses the resistance to linear acceleration of a mass element.
- Inertia tensor — Matrix that contains the moments and products of inertia of a rigid body. The inertia tensor expresses the resistance to angular acceleration of a mass element.

Relevant Blocks

Two blocks accept inertial parameters as input:

- Solid — Block that represents the geometry, inertia, and color of a solid element.
- Inertia — Block that represents *only* the inertial properties of a solid element

Use the Solid block to represent a solid element with shape and color. Use the Inertia block to represent a mass disturbance in a model.

Mass

The dialog box of the Solid block provides two methods that you can use to specify mass:

- Directly enter the total mass — Use this method if you know the numerical value of the solid mass.
- Enter the mass density — Use this method if you do not know the total mass. SimMechanics automatically determines the total mass from the mass density and geometry of a solid. To use this method, you must select **Calculate** from **Geometry** from the **Inertia Type** drop-down menu of the Solid dialog box.

The dialog box of the Inertia block requires that you directly enter the total mass.

For more information, see Solid and Inertia.

Inertia

SimMechanics provides two ways to specify inertia:

- Manual entry of inertia parameters
- Automatic calculation from geometry and either mass or density

Manual Entry of Inertia Parameters

You can use two methods to manually specify inertia. The two methods are common to both Solid and Inertia blocks.

- Point mass — Treat the mass element as a point mass. The inertia tensor is zero. A mass element with a zero inertia tensor does not spin about an internal axis.
- Custom — Manually enter inertial parameters of a mass element. You must compute, or have access to, the numeric values of the inertial parameters. The inertial parameters include the following.

Mass

The total inertial mass of the mass element.

Center of Mass

The [X Y Z] coordinates of the mass element center of mass. The coordinates are resolved in the reference port frame of the Solid or Inertia block.

Moments of Inertia

The *diagonal* elements of the inertia tensor.

$$\begin{matrix}
 I_{xx} & & \\
 & I_{yy} & \\
 & & I_{zz}
 \end{matrix}$$

Products of Inertia

The *off-diagonal* elements of the inertia tensor.

$$\begin{matrix}
 & I_{xy} & I_{zx} \\
 I_{xy} & & I_{yz} \\
 I_{zx} & I_{yz} &
 \end{matrix}$$

The inertia tensor is symmetric about the diagonal. The following equalities hold: $I_{xy}=I_{yx}$, $I_{yz}=I_{zy}$, and $I_{zx}=I_{xz}$.

Automatic Calculation of Inertia Parameters

The Solid block provides automatic calculation of inertia parameters. Use automatic calculation if you do not have access to the numerical values of the inertia parameters. SimMechanics automatically calculates the center-of-mass, moments of inertia, and products of inertia based on the solid geometry and either total mass or mass density.

Import from CAD Assembly

You can import a CAD assembly from a supported CAD platform. During import, SimMechanics interprets an XML file generated by the utility SimMechanics Link to automatically build a SimMechanics model from a CAD assembly. The XML file contains the mass and inertia parameters of each part present in the CAD assembly. Verify that the values of the mass and inertia are correct. You do not need to specify the values manually.

About Extrusion and Revolution Geometries

With **General Extrusion** and **Revolution** geometries, you can specify greater geometric detail than is possible with other geometries. Examples include the fin structure of a heat sink, hollow domes and cones, I-beams and box beams. See how to model a hollow cone and a circular dome:

- “Model a Hollow Cone” on page 2-43
- “Model a Dome” on page 2-54

See how to model an I-beam and a box beam:

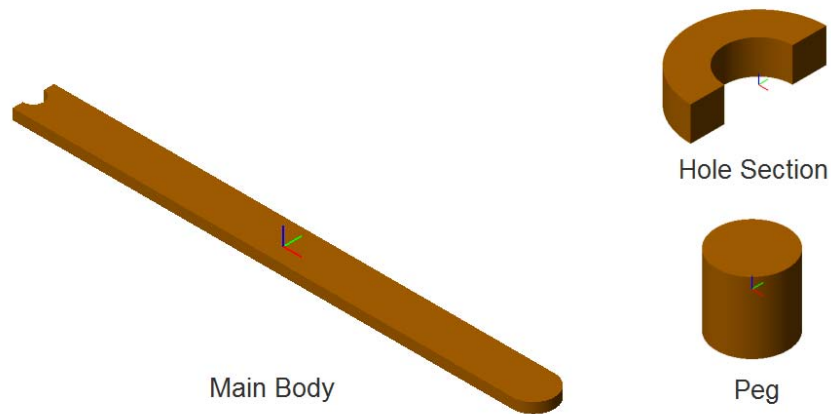
- “Model an I-Beam” on page 2-66
- “Model a Box Beam” on page 2-78

General Extrusion and **Revolution** geometries accept a **MATLAB** matrix as input. The **MATLAB** matrix defines the extrusion or revolution cross-section of the solid. In both cases, the cross-section must be *constant*, either along the extrusion axis or about the revolution axis. To learn more about cross-section specification, see “Extrusion and Revolution Cross-Section Rules” on page 2-23.

General Extrusion Examples

The key to identifying a general extrusion is to look for a constant cross-section. This cross-section can be simple or complex. The only requirement is that it have a constant cross-section along some direction.

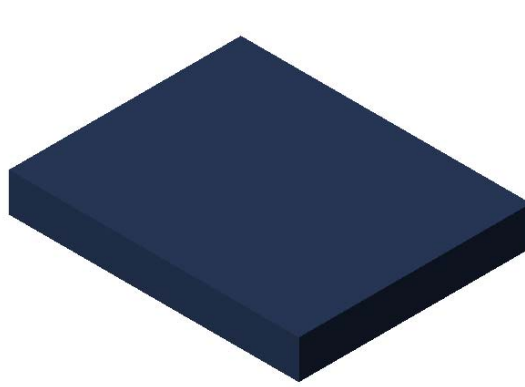
Binary link segments **Main Body** and **Bolt Hole Section** are examples of an extrusion. For these segments, standard shapes like **Brick** can provide only a rough approximation. To accurately model them, you must use **SimMechanics** shape **General Extrusion**.



In the binary link example, segments Main Body and Bolt Hole Section both have a constant cross-section, and you can model them with **General Extrusion**. The following figure illustrates the constant cross-sectional shapes of segments Main Body and Bolt Hole Section. Extruding these cross-sections along their normal axes generates two 3-D segments that you can connect to model part of the binary link.



A second example is a shock mount bracket. This rigid body contains five elementary segments: one base plate and four side plate halves. The base plate has a simple geometry, which you can model with shape **Brick**. However, the four side plates have a more complex geometry requiring shape **General Extrusion**.

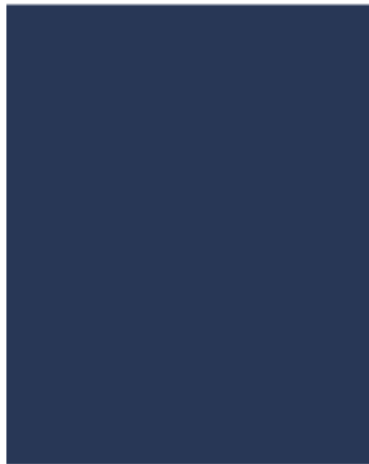


Base Plate



Side Plate Half

You can represent segment Side Plate Half with shape `General Extrusion` because it has a constant cross-section. `General Extrusion` works also with Base Plate, but because this is a simple shape, shape `Brick` is more practical. The following figure illustrates the cross-sectional shapes for the two segments, which you can extrude to model the corresponding 3-D segments.

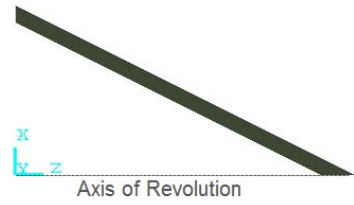
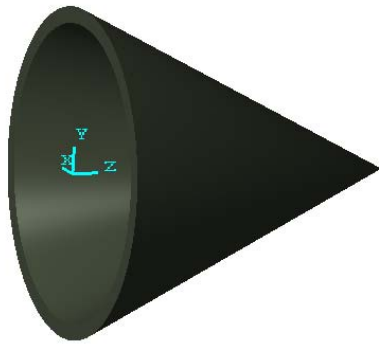


Solid of Revolution

You can model a solid of revolution if it contains a constant cross-section about the revolution axis. A cylindrical heat sink provides one example. A MATLAB matrix of $[x \ z]$ values defines the castellated cross-section, which SimMechanics then sweeps about the revolution axis to generate the 3-D cylindrical heat sink. The following figure shows the Mechanics Explorer display of a cylindrical heat sink modeled in SimMechanics.



Another example of a solid of revolution is the cone. This rigid body has a constant cross-section about its length axis, which you can revolve to generate a 3-D conical shape. The following figure displays the cone and its cross-section. As with the heat sink, you specify this cross-section as a matrix of $[x \ y]$ coordinates defining its vertices.



Extrusion and Revolution Cross-Section Rules

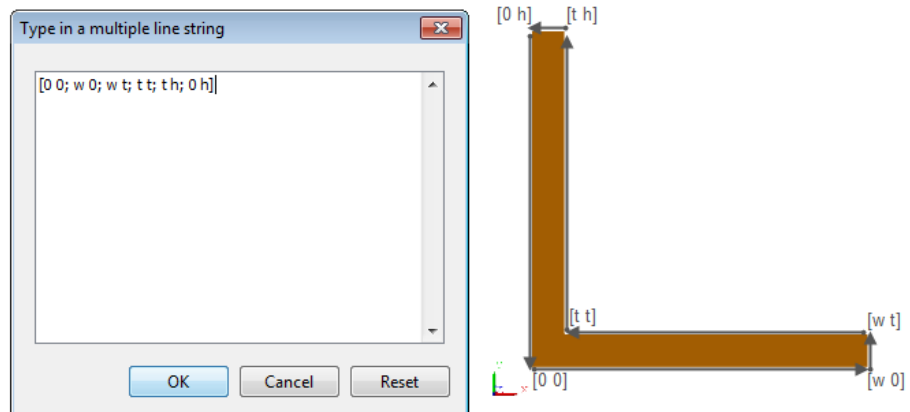
Shapes General Extrusion and Revolution require a MATLAB matrix that contains cross-section coordinates. To be valid, the matrix must satisfy a set of rules and guidelines. Some rules are specific to each shape:

- “Extrusion and Revolution Cross-Section Rules” on page 2-23
- “Revolution Rules” on page 2-28

Coordinates Define Closed Loop

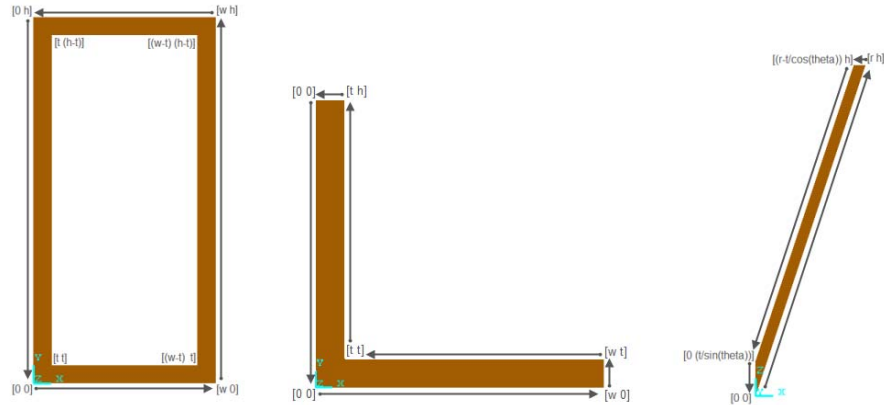
Each matrix row contains the coordinate pair of a single point in the cross-sectional plane. The set of all rows defines all points in the cross-sectional shape.

A straight line connects adjacent coordinate pairs. If the first and last coordinate pairs are not coincident, a straight line connects the two pairs. The set of all straight line segments forms a closed shape that defines the geometry cross-section.



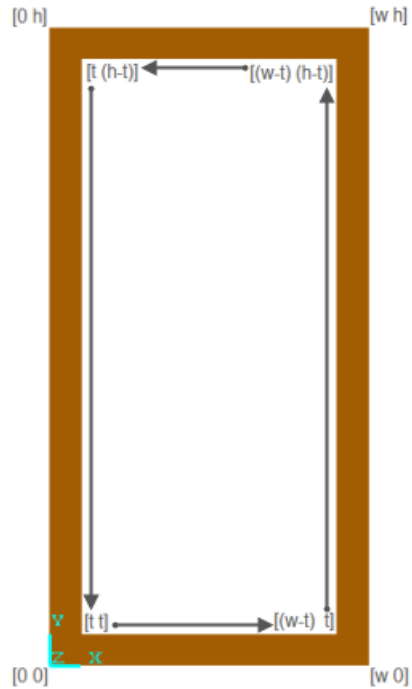
Counterclockwise Loop Encloses Solid Section

Coordinates for the outer profile of a hollow cross-section must follow a counterclockwise order. If a cross-section is not hollow, the coordinates for the single cross-sectional profile must follow a counterclockwise order. Failure to specify the outer cross-section as a counterclockwise matrix results in an error. The model does not simulate.



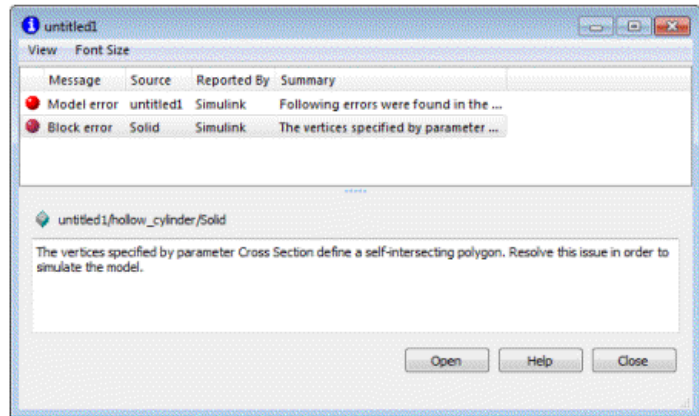
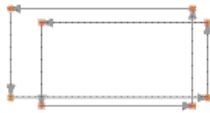
Clockwise Loop Encloses Hollow Section

Coordinates for the inner profile of a hollow cross-section must follow a clockwise order.



Coordinate Loops Must Not Self-Intersect

The MATLAB coordinate matrix must specify a closed loop that does not self-intersect. SimMechanics cannot represent self-intersecting cross-section profiles. If a MATLAB coordinate matrix represents a self-intersecting line, SimMechanics issues an error. The model does not simulate.

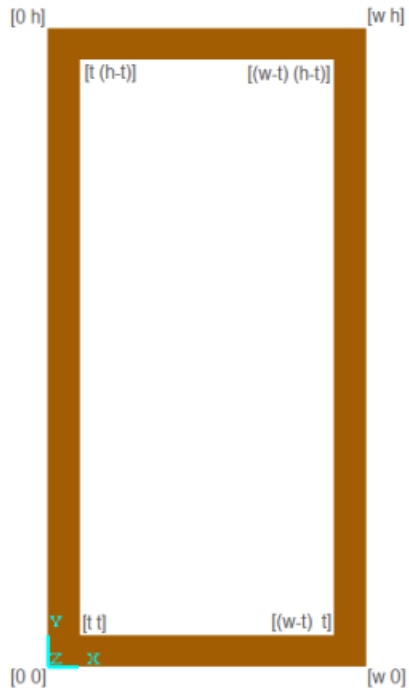


Extrusion Rules

Two guidelines apply to extrusion cross-sections.

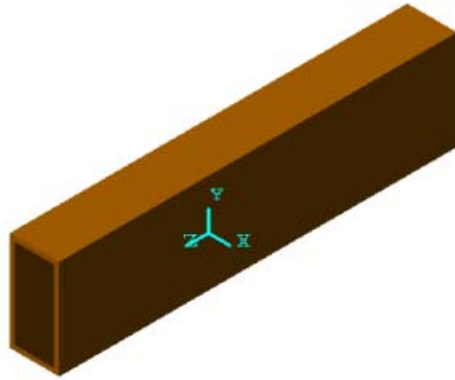
Extrusion Coordinates are [x y] Pairs

Solid geometry General Extrusion accepts an [x y] coordinate matrix as input. The complete set of coordinates denotes a closed 2-D path in the XY plane that represents the extrusion cross-section.



Extrusion Axis Aligns with Z-Axis

The extrusion axis aligns with the z-axis of the solid reference frame. The extrusion axis represents the direction along which SimMechanics extrudes the solid cross-section.



Revolution Rules

Three guidelines apply to revolution cross-sections.

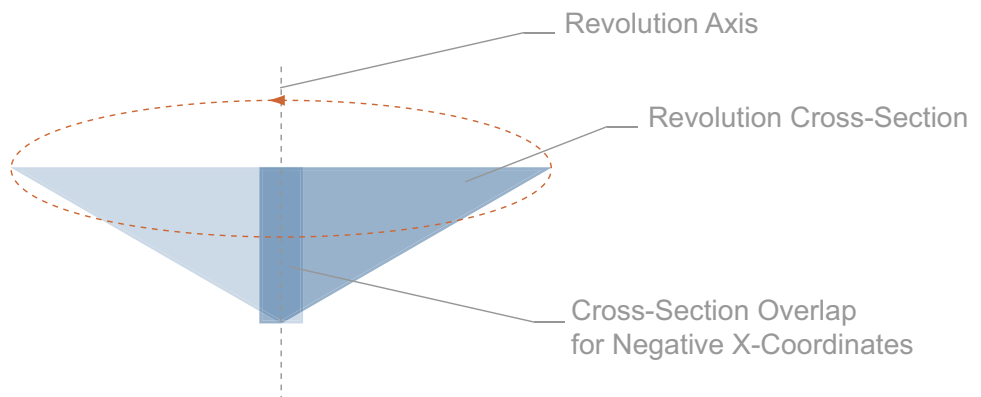
Revolution Coordinates are [x z] Pairs

Solid geometry Revolution accepts an [x z] coordinate matrix as input.



Revolution X-Coordinates Must Be Positive

The x-values of a revolution cross-section must be positive. This rule ensures that the cross-section is specified on a single side of the revolution axis, which prevents cross-section overlap due to revolution.



Revolution Axis Aligns with Z-Axis

The revolution axis aligns with the z-axis of the solid reference frame. The revolution axis represents the direction about which SimMechanics revolves the solid cross-section.



Model a Simple Binary Link

In this section...
“Build Model” on page 2-32
“Add Joint Frames” on page 2-33
“Add Geometry, Inertia, and Color” on page 2-33
“Generate and Mask Subsystem” on page 2-35
“Specify Subsystem Parameters” on page 2-38
“Visualize Model” on page 2-38
“Save Subsystem Block” on page 2-41

Rigid bodies are the basic building blocks of a multibody system. In this example, you model a simple binary link. The shape is cylindrical and the solid properties are parameterized in terms of mathematical variables. The link contains one frame at each end to which you can connect a joint.

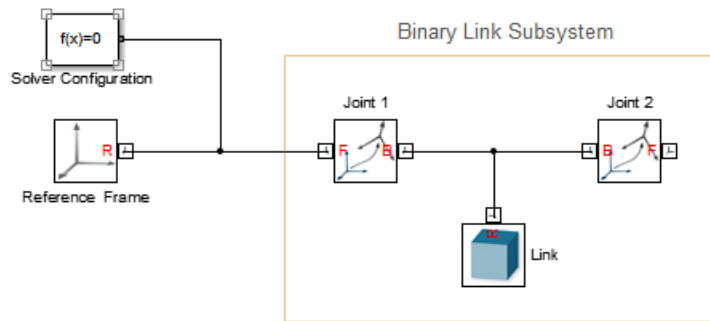
The binary link forms a separate subsystem that you can save and reuse during multibody assembly. Parameterizing solid properties in terms of mathematical variables allows you to change rigid body dimensions, inertia, and color from a single subsystem interface. You initialize the variables in the subsystem mask, and provide the numerical values in the subsystem dialog box. The following table identifies the variable associated with each solid property.

Solid Property	Variable
Cylinder radius	r
Cylinder length	l
Density	rho
Color	rgb

Once you have completed the model, you can add detail. See “Model a Compound Binary Link” on page 2-90

Build Model

Build the block diagram to represent the binary link. The following figure shows the complete block diagram for this example. The rectangular box encloses the blocks that form the binary link subsystem. The Solver Configuration block makes visualization possible during the modeling process.



To create the model:

- 1 Open a new Simulink Editor window.
- 2 Add the following blocks to the model.

Block	Quantity	Library
Solver Configuration	1	Simscape Utilities
Reference Frame	1	Frames & Transforms
Rigid Transform	2	
Solid	1	Body Elements

- 3 Connect and rename the blocks as shown in the previous figure.

Note Rotate each Rigid Transform block so its base (B) frame port faces the Solid block. Then connect the base frame port to the Solid reference (R) frame port.

Add Joint Frames

To connect joints at each end of the binary link, add two frames to the binary link model. Use the Rigid Transform block to add the frames. In the dialog box of the Rigid Transform block, specify the translation and rotation parameters required to give the new frame the desired position and orientation.

To add the frames:

- 1 Double-click the Joint 1 Rigid Transform block.

Note Before performing this procedure, make sure that you renamed the blocks in “Build Model” on page 2-32.

- 2 Expand **Translation**.

- 3 In the **Method** drop-down list, select **Standard Axis**.

- 4 In the **Offset** field, enter $-1/2$.

- 5 In the units drop-down list, enter or select **cm**.

- 6 Press **OK**.

- 7 Repeat these steps for the Joint 2 Rigid Transform block, but enter $+1/2$ in the **Offset** field.

Add Geometry, Inertia, and Color

In the Solid block dialog box, specify shape and enter the variable for each relevant solid parameter.

Add Geometry

Once you have created the block diagram, add shape and inertia to the rigid body:

- 1 Double-click the Solid block.
- 2 In the **Shape** drop-down menu of the dialog box, select **Cylinder**.
- 3 In the **Radius** field, enter r .
- 4 In the **Length** field, enter l .
- 5 In the units menus, enter **cm**.

Geometry		
Shape	Cylinder	▼
Radius	r	cm ▼
Length	l	cm ▼

Add Inertia

With the dialog box of the Solid block still open, specify solid inertia.

- 1 In the dialog box, expand the **Inertia** menu.
- 2 In the **Density** field, enter ρ .

Inertia		
Type	Calculate from Geometry ▼	
Based on	Density ▼	
Density	ρ	g/cm^3 ▼

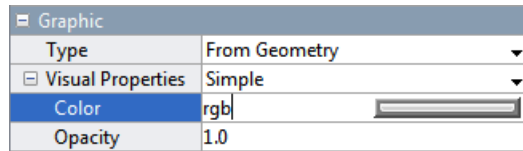
- 3 In the units field, enter g/cm^3 .

Note Field **Inertia Type** is set to **Calculate** from **Geometry** by default. The setting provides automatic calculation of the inertia tensor. In this setting, you need only provide mass or density for the solid material.

Add Color

With the dialog box of the Solid block still open, specify solid color.

- 1 In the dialog box, expand the **Graphic** menu.
- 2 Expand the **Visual Properties** sub-menu.
- 3 In the **Color** field, enter `rgb`.



- 4 In the Solid block dialog box, click **OK**.

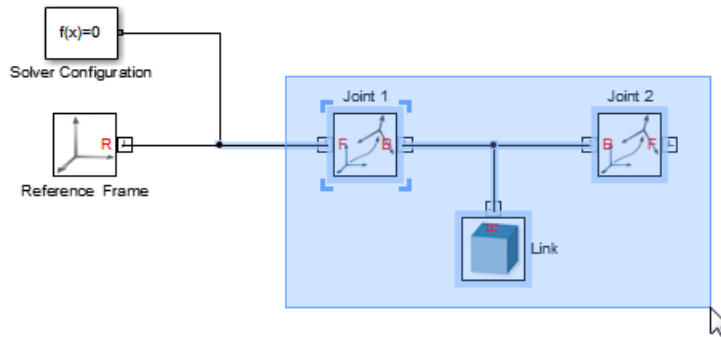
Generate and Mask Subsystem

Convert the binary link block diagram into a masked subsystem. The subsystem reduces clutter and simplifies models that contain multiple rigid bodies. The mask provides a single placeholder for rigid body parameters. You can initialize all relevant parameters in the subsystem mask, and provide their numerical values in the subsystem dialog box.

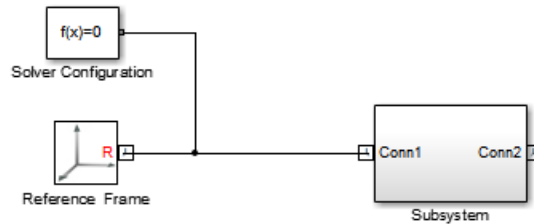
Generate Subsystem

To generate the subsystem:

- 1 Select the Solid and Rigid Transform blocks.

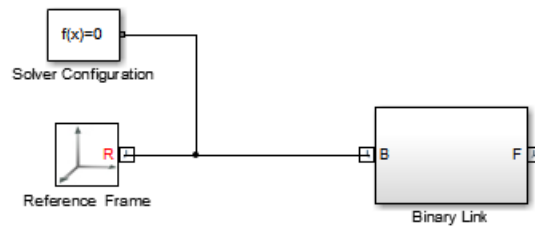


2 Press **Ctrl+G** to create a subsystem from the selection.



3 Rename the subsystem block *Binary Link*.

4 Double-click the subsystem block and rename Port blocks 1 and 2 *B* and *F*, respectively.



Mask Subsystem


Mask the subsystem that you created for the binary link rigid body.

- 1 In the model canvas, select the subsystem block.
- 2 Press **Ctrl+M** to create mask.

Note Leave the Mask Editor window open for the next procedure.

Specify Subsystem Parameters

In the subsystem mask, specify the subsystem parameters that you entered in the Solid and Rigid Transform blocks.

- 1 In the Mask Editor, click the **Parameters** tab.
- 2 Click the Add Parameter  button four times to add four parameter lines.
- 3 In the **Prompt**, **Variable**, and **Tab name** fields, enter the following information.

Prompt	Variable	Tab name
Radius (cm)	r	Dimensions
Length (cm)	l	Dimensions

Prompt	Variable	Tab name
Density (g/cm ³)	rho	Material Properties
Color [R G B]	rgb	Material Properties

4 Click **OK**.

Specify Subsystem Parameters

Specify the numerical value of each solid parameter in the subsystem dialog box.

- 1** Double-click the subsystem dialog box.
- 2** In the **Dimensions** tab, enter numerical values for **Radius** and **Length**.
- 3** In the **Material Properties** field, enter numerical values for **Density** and **Color**.
- 4** Click **OK**.

The following table provides the values for this example.

Parameter	Tab	Value
Radius	Dimensions	1
Length	Dimensions	10
Density	Material	2.70
Color	Material	[0 0.4 0.9]

Visualize Model


Update the model and examine the 3-D display of the rigid body. Confirm that the shape and color are correct.

Before completing this section, review the following sections:

- “Visualization with Mechanics Explorer” on page 6-11
- “Configure Mechanics Explorer Display” on page 6-2

- “Rotate, Pan, and Zoom View” on page 6-14

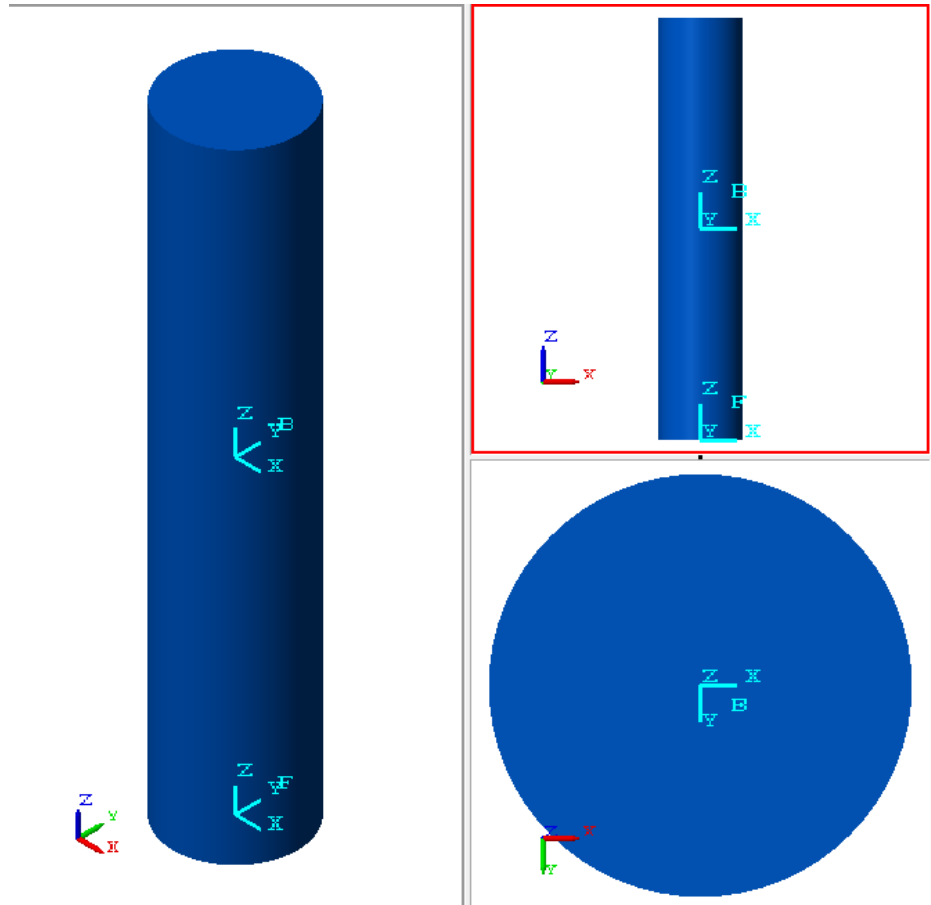
To verify the rigid body:

- 1** In the model window, select **Simulation > Update Diagram**, or press **Ctrl+D**.
- 2** In the Mechanics Explorer window that opens, click the isometric view icon .
- 3** Verify that the shape is correct.

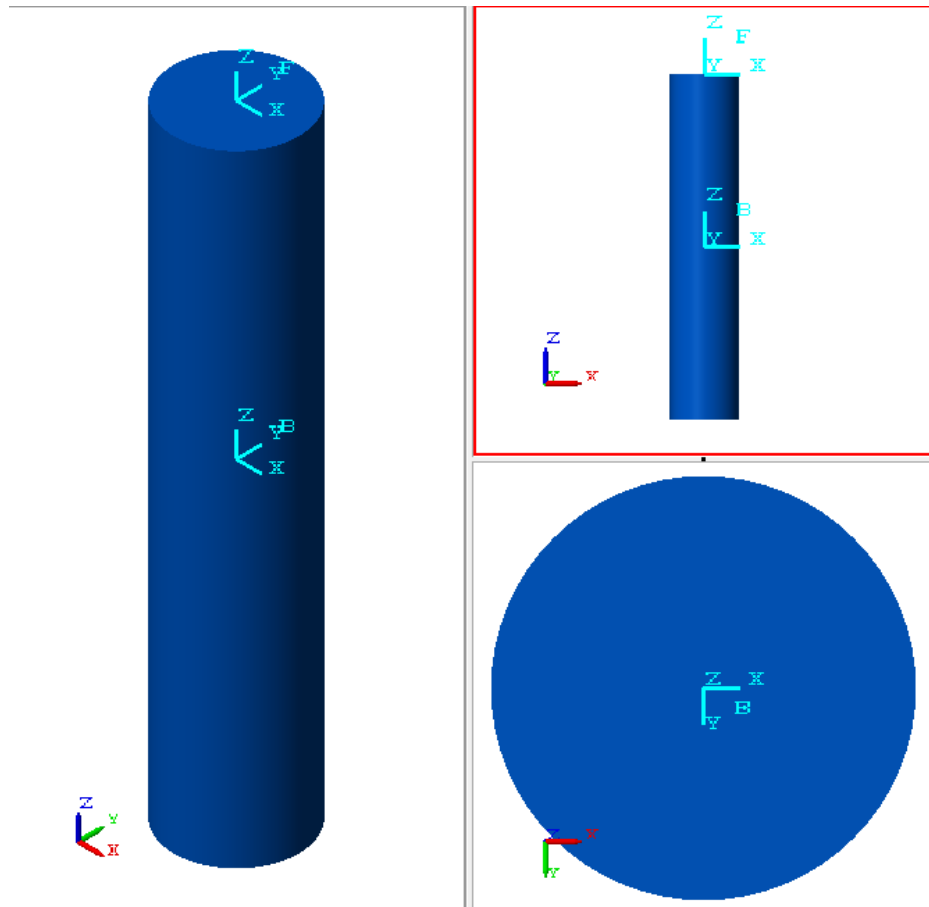


- 4** In the model navigation pane of Mechanics Explorer, select **Binary Link > Joint_1**.

- 5 In the visualization pane of Mechanics Explorer, verify that frame F of Joint_1 lies at the intersection of the cylinder axis and the *bottom* cylinder end plane.



- 6 In the model navigation pane of Mechanics Explorer, select **Binary Link > Joint_2**.
- 7 In the visualization pane of Mechanics Explorer verify that frame F of Joint_2 lies at the intersection of the cylinder axis and the *top* cylinder end plane.



Save Subsystem Block

Save the subsystem in a custom block library for reuse. During multibody assembly, you can drag and connect multiple instances of the subsystem block. You can change the dimensions, density, and color of each binary link individually.

To save the rigid body subsystem in a custom library:

- 1 On the Simulink Editor toolbar, select **File > New** and click **Library**.

2 In the library window, select **File > Save As** and enter the library name.

3 Drag the Binary Link subsystem block to the new library and save.

To reuse the binary link subsystem, open the library and drag the subsystem block into the model window.

Model a Hollow Cone

In this section...

“Parameterize Cross-Section Coordinates” on page 2-44

“Build Model” on page 2-46

“Add Geometry, Inertia, and Color” on page 2-47

“Generate and Mask Subsystem” on page 2-48

“Specify Subsystem Parameters” on page 2-51

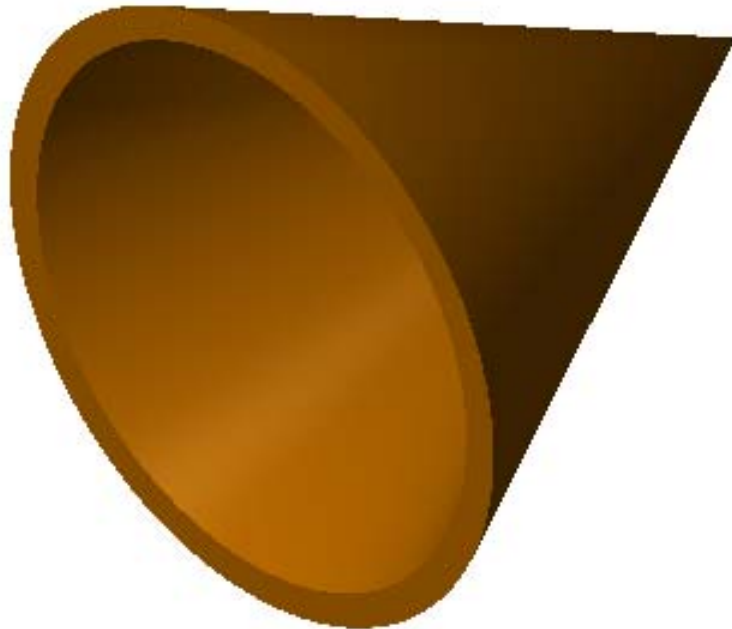
“Visualize Model” on page 2-52

“Modify Geometry” on page 2-53

Geometry specification is an important step in the accurate representation of a rigid body. For intricate shapes, SimMechanics provides two advanced shapes that you can use: **General Extrusion** and **Revolution**. You do not need to manually enter the inertial parameters of a complex shape — the **Solid** block provides an option to automatically calculate inertia from geometry.

This example shows a solid of revolution by modeling a cone-shaped, rigid body with a single **Solid** block. Emphasis is on the parameterization of the cone in terms of its dimensions. You can rigidly connect multiple **Solid** blocks with **Rigid Transform** blocks to build compound rigid bodies that contain a greater level of detail. For more information about the **Revolution** geometry, see “**Extrusion and Revolution Cross-Section Rules**” on page 2-23.

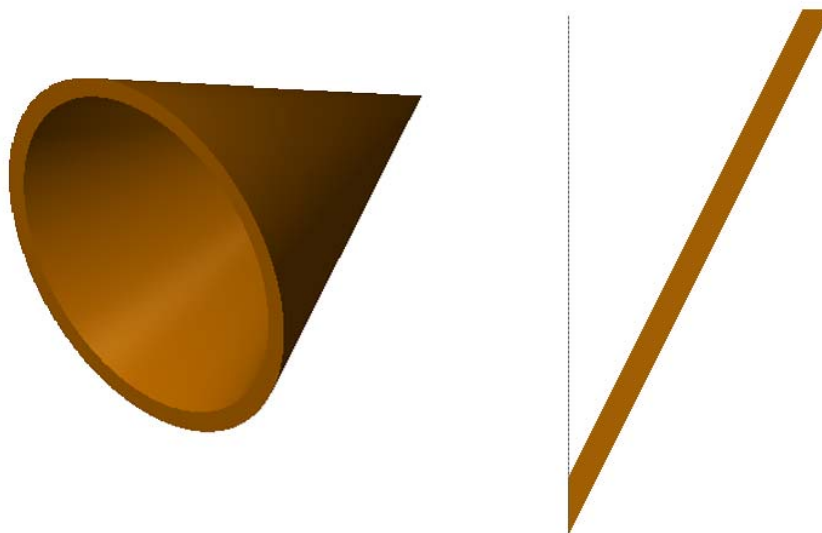
The following figure shows the **Mechanics Explorer** display of the cone that you model in this example.



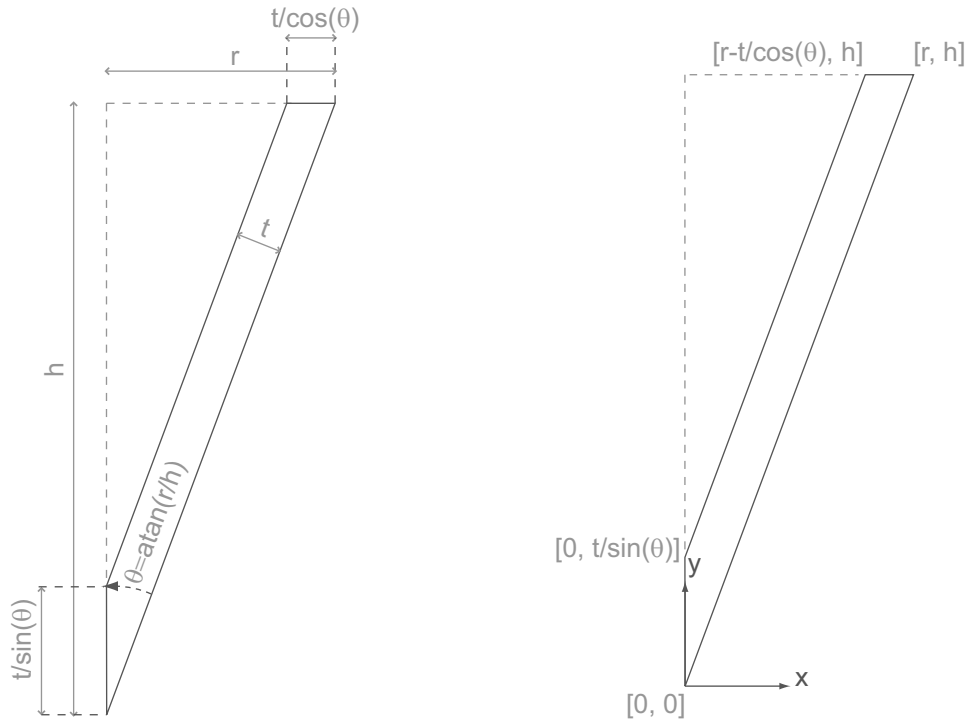
Parameterize Cross-Section Coordinates

Before you model the cone rigid body, identify the cone cross-section and its coordinates. You can then enter the cross-section coordinates as a MATLAB matrix in a cone subsystem mask.

The hollow cone has a trapezoidal cross-section.



Determine the vertex $[x \ y]$ coordinates of the cone cross-section, and then parameterize the coordinates in terms of relevant cone dimensions. Relevant parameters include the base radius (r), cone height (h), and wall thickness (t). The following figure shows the cross-section coordinates of the cone.

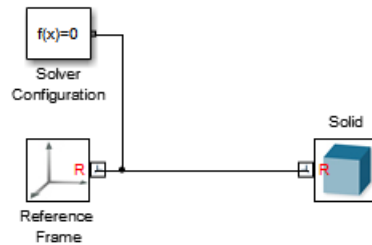


1 This table describes the relevant cone parameters.

Parameter	Description
r	Radius at cone base
h	Height of cone
t	Thickness of cone wall

Build Model

Create a model to represent the cone. The model contains a single frame at the tip of the cone. The following figure shows the complete block diagram of the cone.



To create the model:

- 1 Open a new Simulink model.
- 2 Add these blocks to the model.

Block	Quantity	Library
Solver Configuration	1	Simscape Utilities
Reference Frame	1	Frames & Transforms
Solid	1	Body Elements

- 3 Connect the blocks as shown in the previous figure.

Add Geometry, Inertia, and Color

Open the Solid block and specify relevant dimensional parameters in terms of variables.

- 1 Double-click the Solid block.
- 2 In the **Shape** drop-down list of the dialog box, select **Revolution**.
- 3 In **Cross-section**, enter `xz_coords`.
- 4 Expand **Inertia**.
- 5 In **Density**, enter `rho`.

6 Expand **Graphic > Visual Properties**.

7 In **Color**, enter rgb.

8 Click **OK**.

Generate and Mask Subsystem

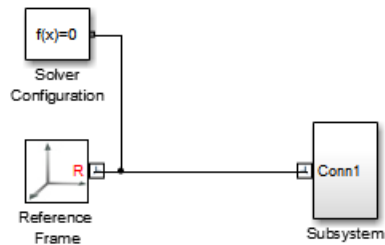
Generate a subsystem for the rigid body. Then, mask the subsystem. Later, you use the mask to parameterize geometry, inertia, and color.

Generate Subsystem

To generate the subsystem:

1 Right-click the Solid block.

2 In the context menu, click **Create Subsystem from Selection**.



Mask Subsystem

To mask the subsystem:

1 Right-click the Subsystem block.

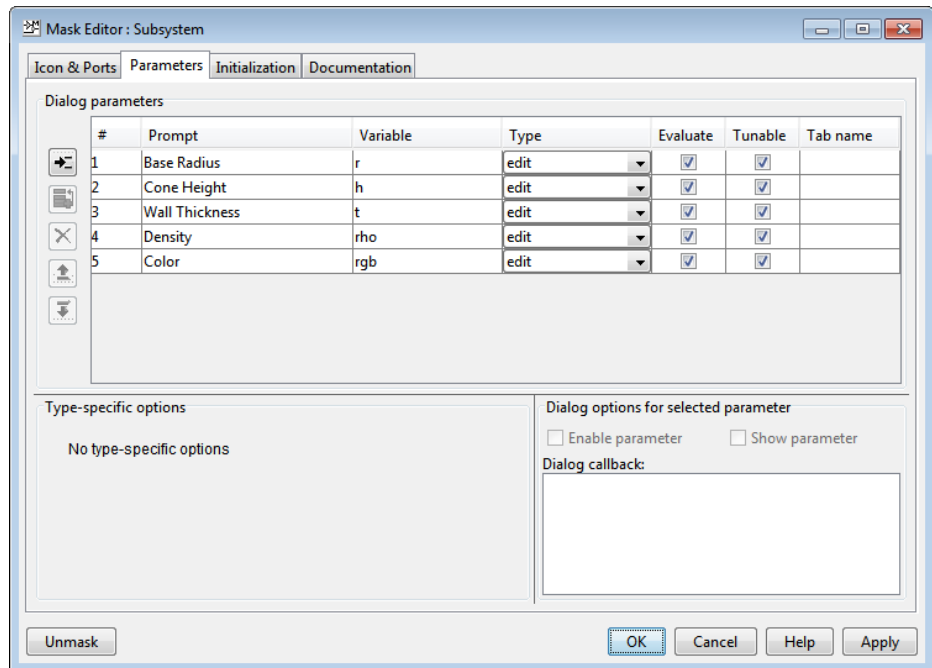
2 In the context menu, click **Mask > Create Mask**.

Parameterize Geometry, Inertia, and Color

In the subsystem mask, define geometry, inertia, and color parameters used in the Solid block.

- 1 In the model, right-click the Subsystem block.
- 2 In the context menu, click **Mask > Edit Mask**.
- 3 In the Mask Editor, click the **Parameters** tab.
- 4 In the **Dialog parameters** pane, click the **Add Parameter** to add five parameters.
- 5 In the **Prompt** and **Variable** fields, enter the following information.

Prompt	Variable
Base Radius	r
Cone Height	h
Wall Thickness	t
Density	rho
Color	rgb



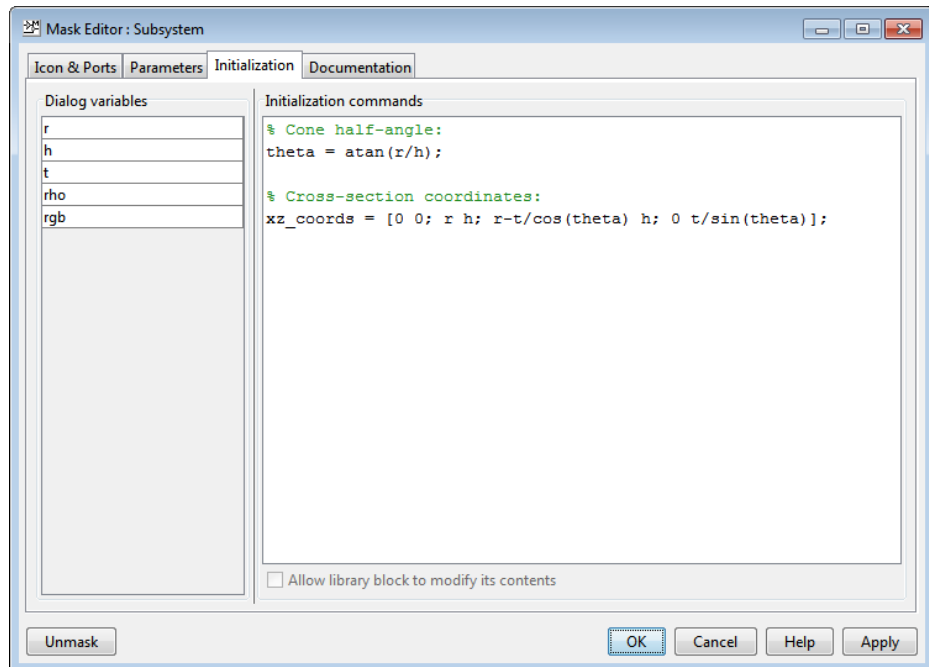
Initialize Cross-Section Coordinates

With the Mask Editor still open, use the **Initialization** tab to define the cross-section coordinate matrix.

- 1 In the Mask Editor, click the **Initialization** tab.
- 2 In the **Initialization Commands** pane, enter the following commands:

```
% Cone half-angle:
theta = atan(r/h);
```

```
% Cross-section coordinates:
xy_coords = [0 0; r h; r-t/cos(theta) h; 0 t/sin(theta)];
```

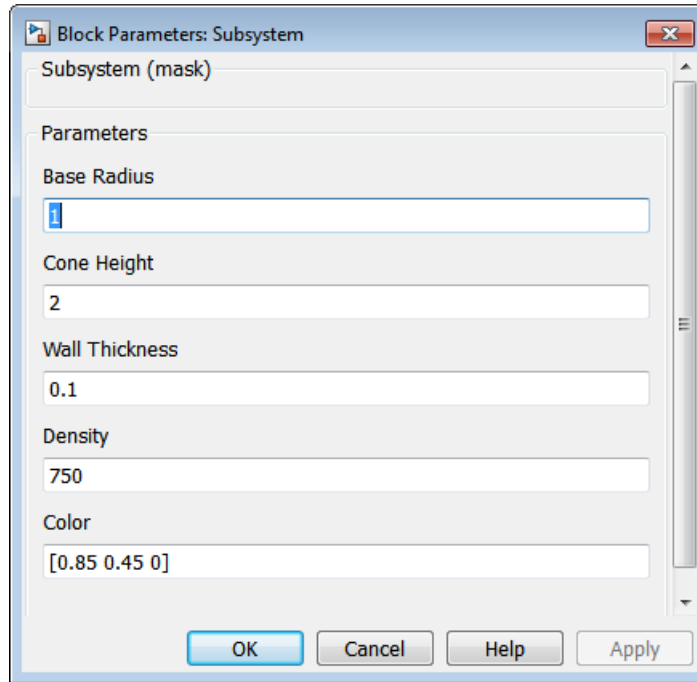


3 Click **OK**.

Specify Subsystem Parameters

Complete the rigid body model by specifying the dimensional parameter values.

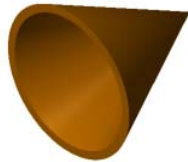
- 1** In the rigid body model, double-click the Subsystem block.
- 2** In the Block Parameters dialog box, enter cone dimensions, density, and color.



Visualize Model

You can now visualize the cone in the Mechanics Explorer window:

- 1 On the Simulink Editor menu bar, select **Simulation > Update Diagram**.



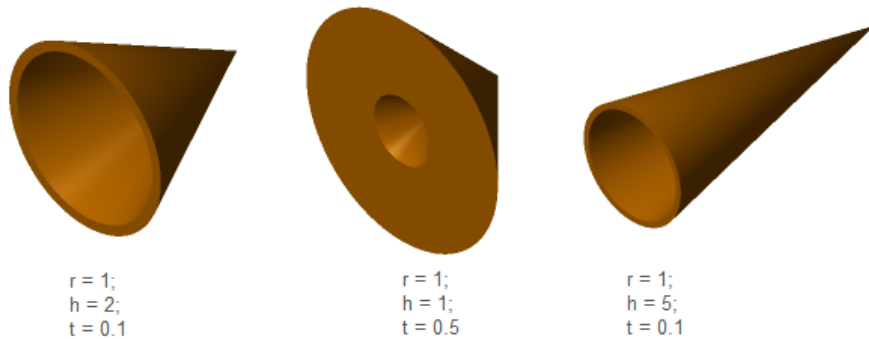
- 2 Pan, rotate and zoom the model to the view that you want. For more information, see “Rotate, Pan, and Zoom View” on page 6-14.

Modify Geometry

You can quickly modify cone dimensions, density, and color at the subsystem mask level.

- 1** In the model, double-click cone subsystem.
- 2** Enter new set of parameters.
- 3** On the Simulink Editor menu bar, select **Simulation > Update Diagram**.

The following figure displays the cone geometries you can generate using different parameters.

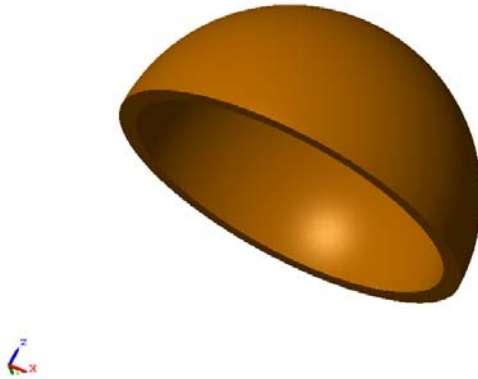


Model a Dome

In this section...
“Parameterize Cross-Section Coordinates” on page 2-55
“Build Model” on page 2-57
“Add Geometry” on page 2-58
“Generate and Mask Subsystem” on page 2-58
“Specify Subsystem Parameters” on page 2-62
“Visualize Model” on page 2-63
“Modify Geometry” on page 2-64

This example provides an example of a solid of revolution. The example models a dome-shaped rigid body with a single Solid block. Emphasis is on the parameterization of the dome in terms of its dimensions. You can rigidly connect multiple Solid blocks with Rigid Transform blocks to build compound rigid bodies that contain a greater level of detail. For more information about the Revolution geometry, see “Extrusion and Revolution Cross-Section Rules” on page 2-23.

The following figure shows the Mechanics Explorer display of the dome shape that you model in this example.

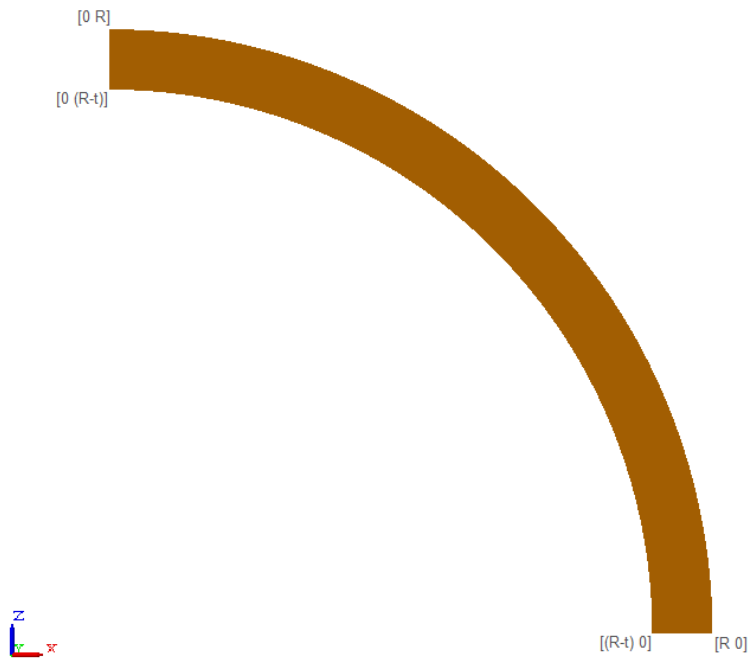


The circular dome is a hollow rigid body. You must specify *both* inner and outer cross-section profiles. SimMechanics implements two rules to distinguish between outer and inner coordinate matrices:

- The coordinates of the *outer* cross-section profile must follow a counterclockwise order.
- The coordinate matrix of the *inner* cross-section profile must follow a clockwise order.

Parameterize Cross-Section Coordinates

This example parameterizes the cross-section coordinates in terms of relevant cross-section dimensions. Relevant dimensions include dome radius (R) and wall thickness (t). Once the circular dome model is complete, you can change the dimension values to change the cross-section geometry. The following figure shows the cross-section of the circular dome in terms of the dimension parameters R and t .



Because the cross-section is circular, you must define an angle matrix that you can use to compute the cross-section coordinates. The coordinate matrix for the outer cross-section profile follows a counterclockwise order. Define an angle matrix that runs from 0° to 90° . Use the angle matrix to define the coordinate matrix:

```
phi = (0:1:90)'*pi/180;  
outer_coords = [R*cos(phi) R*sin(phi)];
```

The coordinate matrix for the inner cross-section profile follows a clockwise order. Define an angle matrix that runs inversely from 90° to 0° . Use the angle matrix to define the coordinate matrix:

```
phi = (90:-1:0)'*pi/180;  
outer_coords = [R*cos(phi) R*sin(phi)];
```

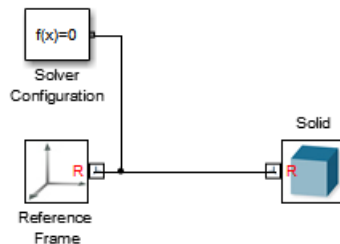
Build Model

The circular dome model requires a single Solid block. To model a circular dome rigid body:

- 1 Start a new Simulink model.
- 2 Add the following blocks to the model.

Block	Library	Quantity	Function
Reference Frame	Frames & Transforms	1	Provides ultimate reference frame to model. Frame is non-inertial.
Solid	Body Elements	1	Provides solid geometry, inertia, and graphic properties
Solver Configuration	Simscape Utilities	1	Enables model assembly, visualization, and simulation

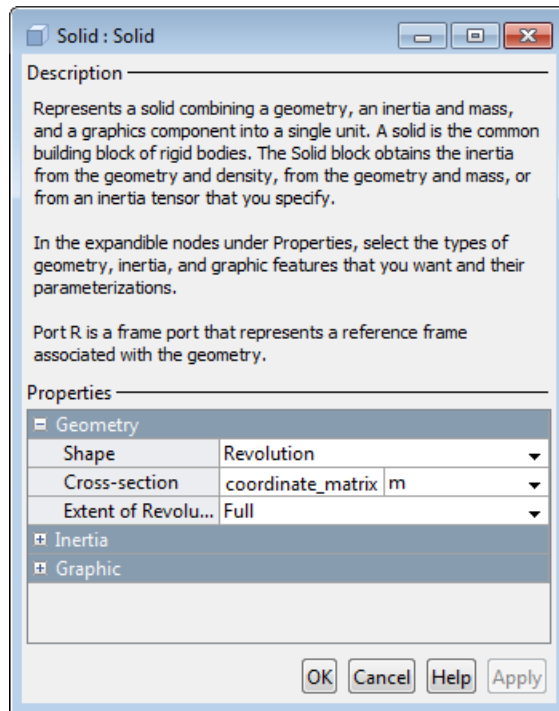
- 3 Connect and name the blocks according to the following figure.



Add Geometry

Specify the shape of the solid.

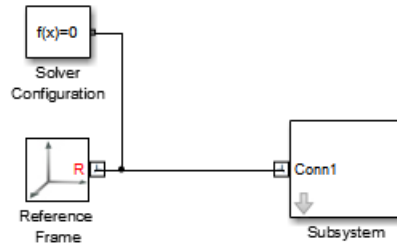
- 1 Double-click the Solid block.
- 2 With **Geometry** expanded, in the **Shape** drop-down list, select **Revolution**.
- 3 In **Cross-section**, enter `coordinate_matrix`.
- 4 In **Extent of Revolution**, select **Full**.



Generate and Mask Subsystem

Convert the rigid body into a separate subsystem. Then, mask the subsystem and specify solid parameters at the mask level.

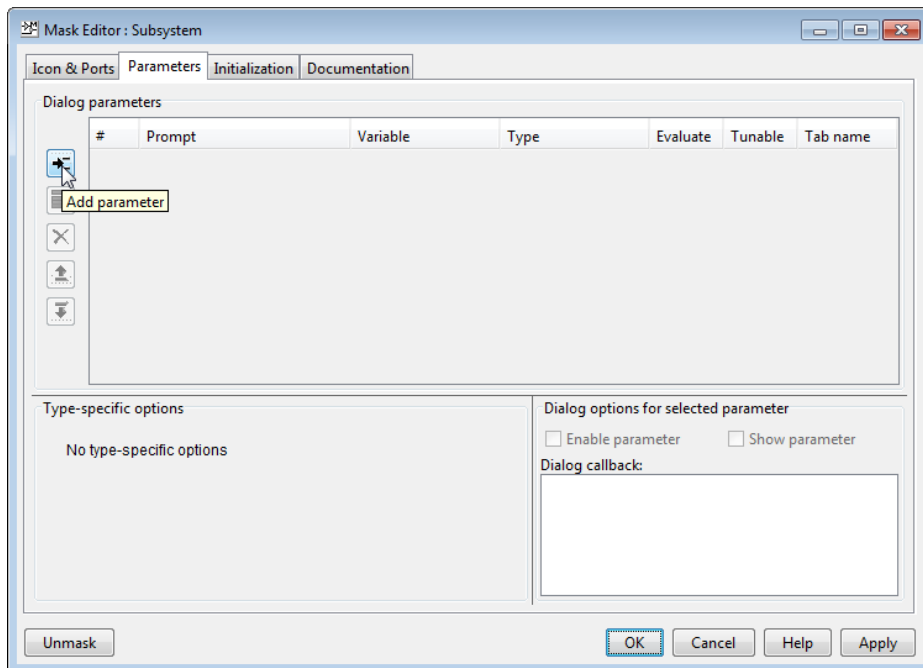
- 1 Click the Solid block that represents the circular dome.
- 2 Press **Ctrl+G** to generate a subsystem for the circular dome.



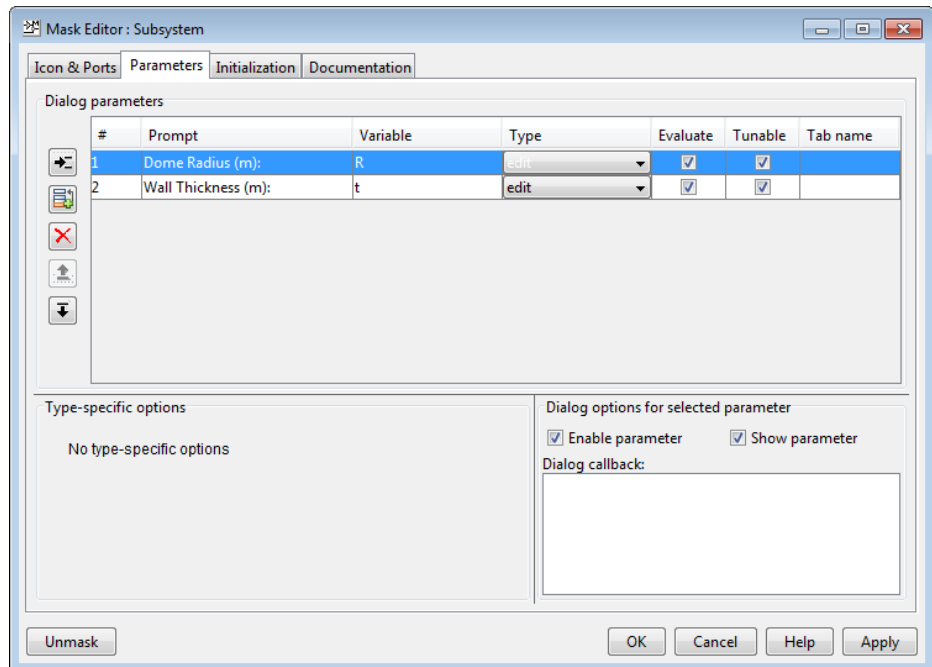
Mask Subsystem

Mask the subsystem so you can specify solid parameters and initialize cross-section coordinates from a single interface.

- 1 Press **Ctrl+M**.
- 2 In the Mask Editor, click **Parameters**.
- 3 On the left side of the **Dialog Parameters** pane, click the **Add parameter** button two times.



- 4** In each parameter line, enter **Prompt** and **Variable** for parameters R and t).



Initialize Cross-Section Coordinates

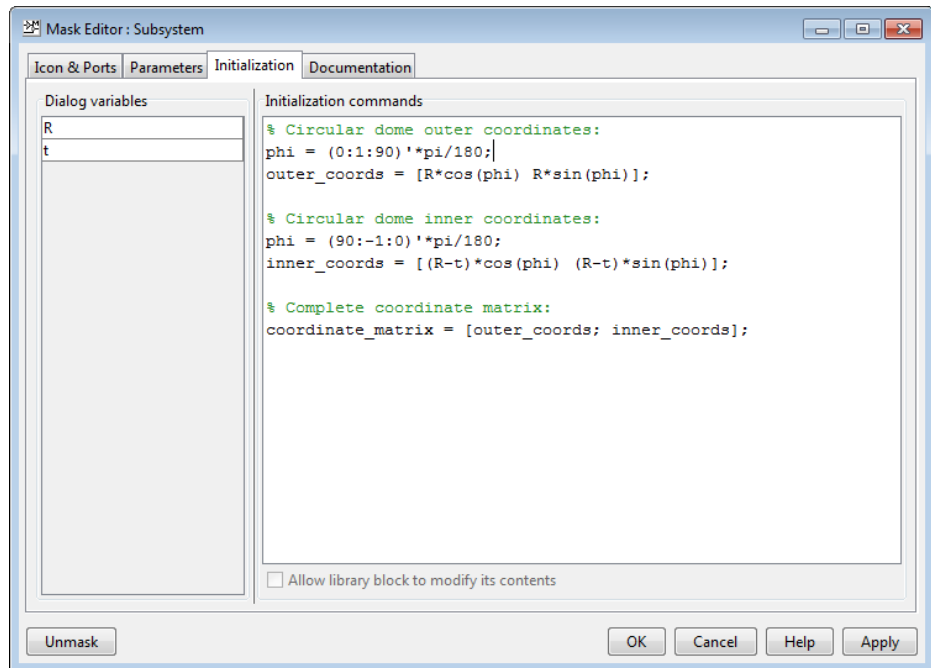
Define the cross-section coordinates of the solid.

- 1 In the Mask Editor, click **Initialization**.
- 2 In the **Initialization commands** pane, enter the parameterized cross-section coordinates:

```
% Circular dome outer coordinates:
phi = (0:1:90)'*pi/180;
outer_coords = [R*cos(phi) R*sin(phi)];

% Circular dome inner coordinates:
phi = (90:-1:0)'*pi/180;
inner_coords = [(R-t)*cos(phi) (R-t)*sin(phi)];

coordinate_matrix = [outer_coords; inner_coords];
```



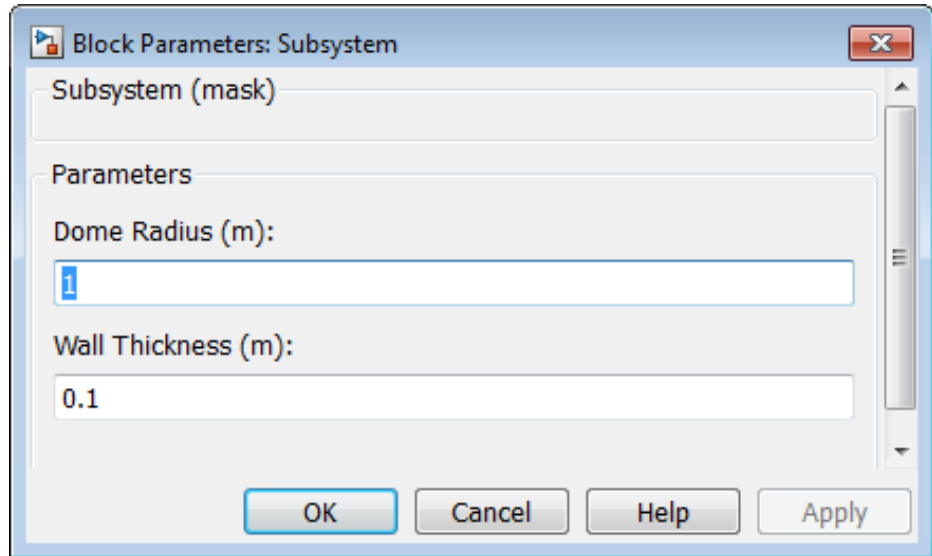
3 Click **OK**.

Specify Subsystem Parameters

Use the dialog box of the subsystem that you created to specify the rigid body parameters.

- 1 Double-click the circular dome subsystem block.
- 2 In the subsystem dialog box, enter the values of the two dimension parameters (R and t).

Parameter	Value
R	1
t	0.1

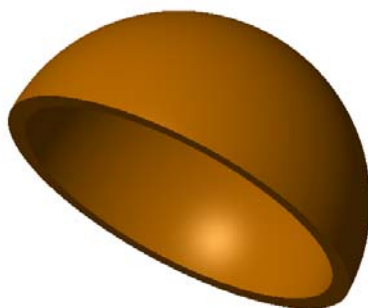


Click **OK**.

Visualize Model

- 1 Press **Ctrl+D**.
- 2 In Mechanics Explorer, verify model geometry.

Select different viewpoints to see the circular dome from different perspectives. The following image shows a custom view of the circular dome model with the dimensions outlined in the previous table.

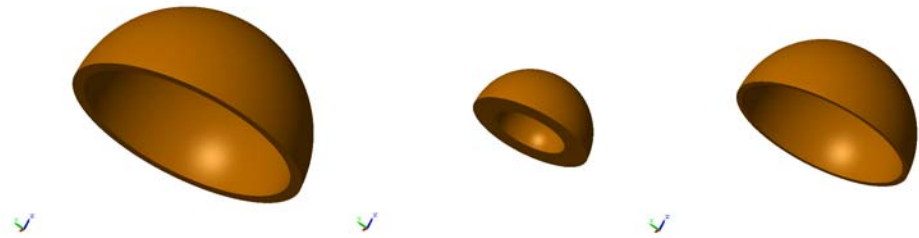


Modify Geometry

Edit the dimension parameters to change the circular dome geometry. The two dimension parameters completely define the circular dome cross-section, so you do not need to reenter a new coordinate matrix inside the Solid block. To change the circular dome geometry:

- 1** In your SimMechanics model, double-click the circular dome subsystem block.
- 2** In the dialog box, enter the new dimension parameters.
- 3** Click **OK**.
- 4** Press **Ctrl+D** to update the circular dome display in Mechanics Explorer.

The following figure shows three possible iterations of the circular dome geometry, with the dimension parameters listed in the table.



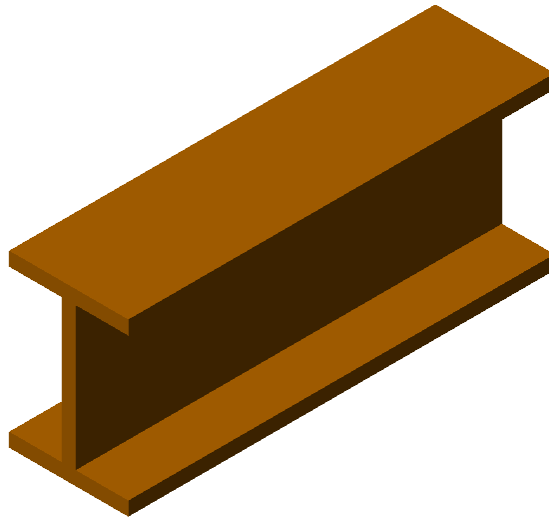
Iteration #	R	t
1	1	0.1
2	0.5	0.2
3	0.75	0.05

Model an I-Beam

In this section...
“Parameterize Cross-Section Coordinates” on page 2-67
“Build Model” on page 2-69
“Generate an I-Beam Subsystem” on page 2-71
“Visualize I-Beam in Mechanics Explorer” on page 2-74
“Change the I-Beam Geometry” on page 2-76

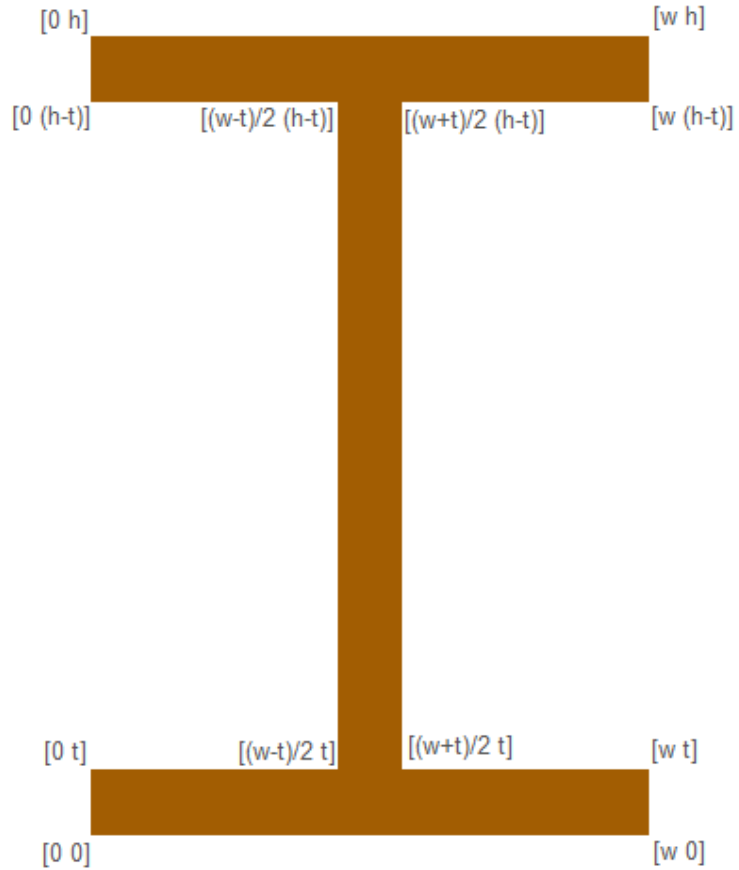
This example provides an example of a general extrusion. The example models an I-shaped beam with a single Solid block. Emphasis is on the parameterization of the beam in terms of its dimensions. You can rigidly connect multiple Solid blocks with Rigid Transform blocks to build compound rigid bodies that contain a greater level of detail. For more information about the General Extrusion geometry, see “Extrusion and Revolution Cross-Section Rules” on page 2-23.

The following figure shows the Mechanics Explorer display of the I-shaped beam that you model in this example.



Parameterize Cross-Section Coordinates

This example parameterizes the cross-section coordinates in terms of relevant cross-section dimensions. Relevant dimensions include beam height (h), flange width (w), and web thickness (t). Once the box beam model is complete, you can change the dimension values to change the cross-section geometry. The following figure shows the vertex coordinates of the I-beam cross-section in terms of h , w , and t .



A counterclockwise MATLAB matrix specifies the outer cross-section coordinates. For the I-beam example:

```
coordinate_matrix = [0 0; w 0; w t; (w+t)/2 t; (w+t)/2 (h-t);
```


$w (h-t); w h; 0 h; 0 (h-t); (w-t)/2 (h-t); (w-t)/2 t; 0 t];$

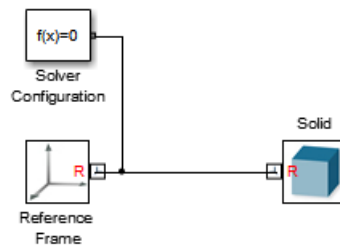
Build Model

The I-beam model requires a single Solid block. To model an I-beam rigid body:

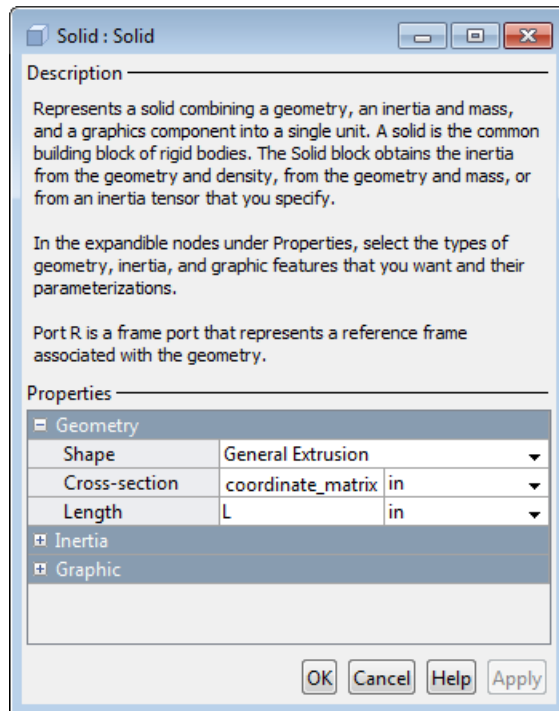
- 1 Start a new Simulink model.
- 2 Add the following blocks to the model.

Block	Library	Quantity	Function
Reference Frame	Frames & Transforms	1	Provides ultimate reference frame to model. Frame is non-inertial.
Solid	Body Elements	1	Provides solid geometry, inertia, and graphic properties.
Solver Configuration	Simscape Utilities	1	Enables model assembly, visualization, and simulation.

- 3 Connect and name the blocks according to the following figure.



- 4** Double-click the Solid block.
- 5** In the Geometry section of the Solid dialog box, select **Shape > General Extrusion**.
- 6** In **Cross-section**, enter `coordinate_matrix` and select units of in.
- 7** In **Length**, enter `L` and select units of in.

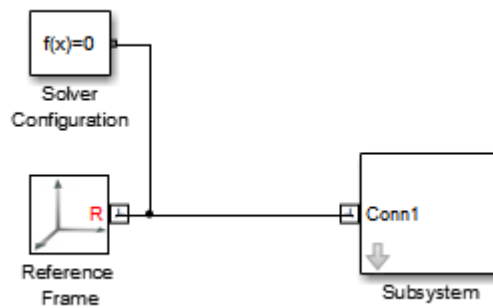


Note Parameters `coordinate_matrix` and `L` specify the cross-section and length of the I-beam. In this example, you provide the values of `coordinate_matrix` and `L` in a subsystem mask. By parameterizing the I-beam geometry in terms of `coordinate_matrix` and `L`, you can quickly change the I-beam geometry without having to reopen the Solid block dialog box.

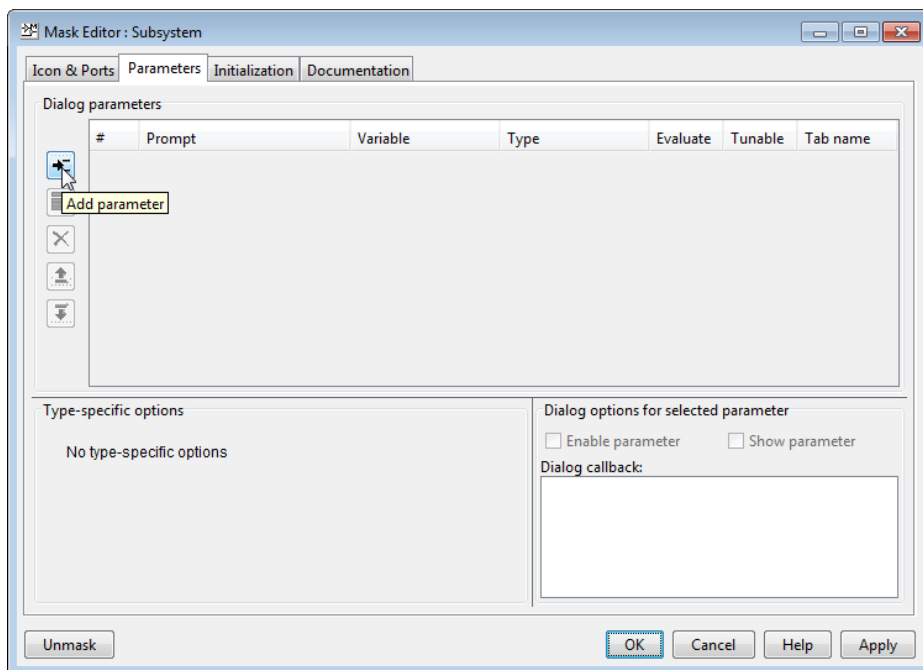
Generate an I-Beam Subsystem

The next step in modeling an I-beam is to generate and mask an I-beam subsystem.

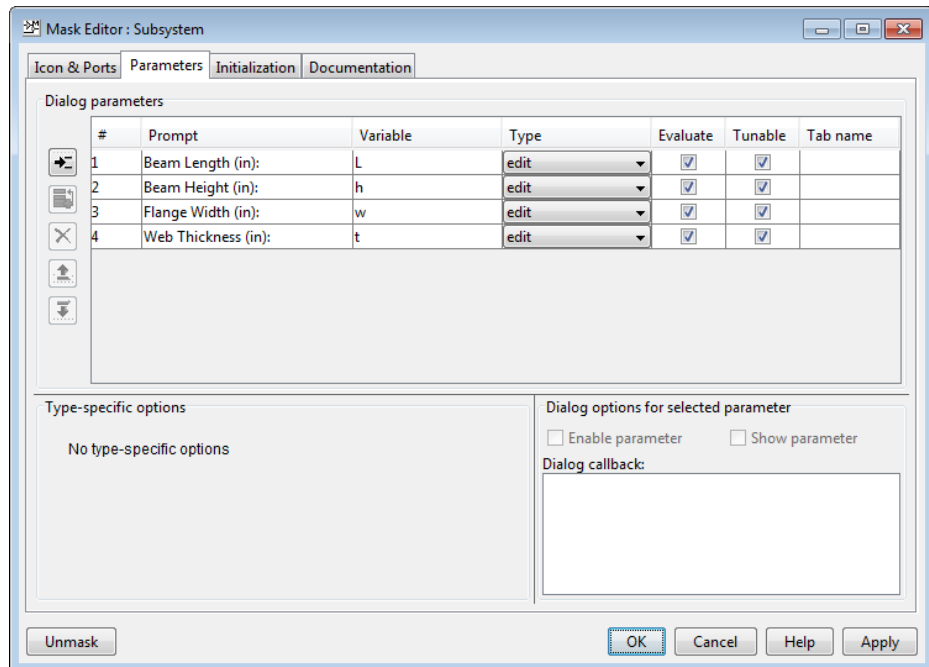
- 1 Click the Solid block that represents the I-beam.
- 2 Press **Ctrl+G** to generate a subsystem for the I-beam Solid block.



- 3 Press **Ctrl+M** to mask the box beam subsystem.
- 4 In the subsystem mask editor, click the **Parameters** tab.
- 5 On the left side of the **Dialog Parameters** pane, click the **Add parameter** button four times.



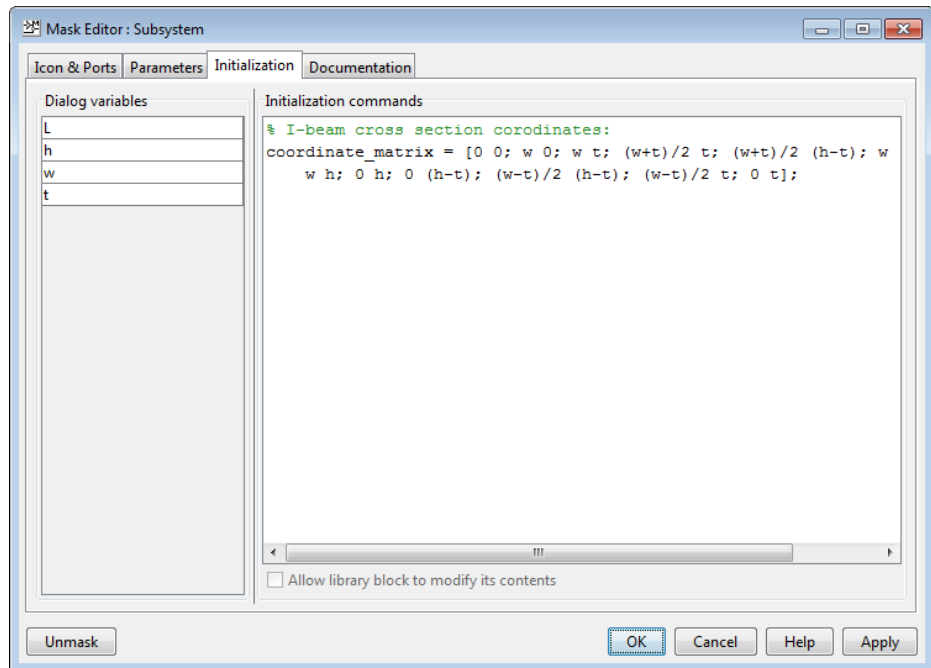
- 6 Enter **Prompt** and **Variable** for the four dimension parameters (L, h, w, and t).



7 Click the **Initialization** tab.

8 In the **Initialization commands** pane, enter the cross-section coordinate matrix in terms of the cross-section dimensions (h, w, t):

```
% I-beam cross-section coordinates:
coordinate_matrix = [0 0; w 0; w t; (w+t)/2 t; (w+t)/2 (h-t);
w (h-t); w h; 0 h; 0 (h-t); (w-t)/2 (h-t); (w-t)/2 t; 0 t];
```

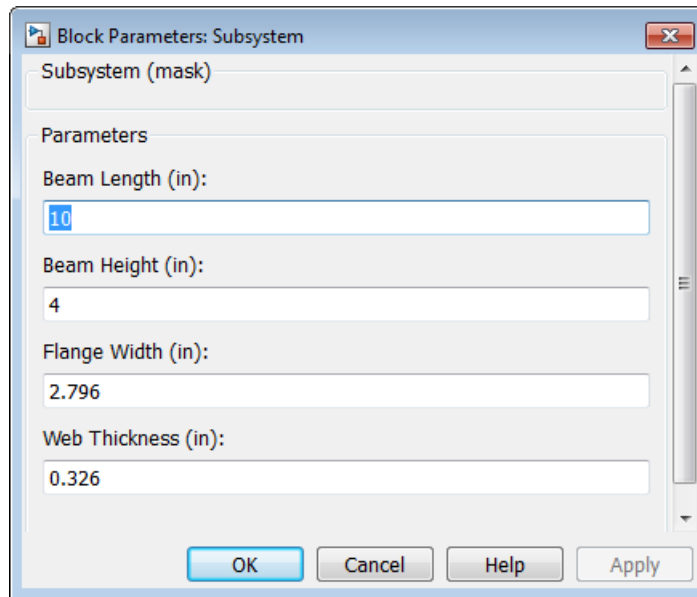


9 Click **OK**.

Visualize I-Beam in Mechanics Explorer

The I-beam model is now complete. Render the I-beam in Mechanics Explorer and verify that the geometry is correct.

- 1 Double-click the I-beam subsystem block.
- 2 In the subsystem dialog box, enter the values of the four dimension parameters (L, h, w, t).

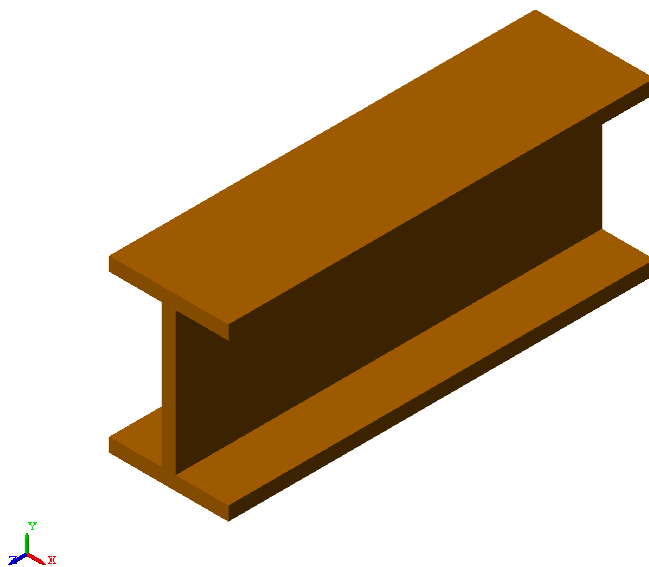


Click **OK**.

3 Press **Ctrl+D**.

A Mechanics Explorer window opens with a 3-D display of the I-beam. Select different viewpoints to see the I-beam from different perspectives. The following image shows an isometric view of the I-beam model with the dimensions outlined in the table. The beam has the ASTM designation S4×9.5

Parameter	Value
L	10
h	4
w	2.796
t	0.326

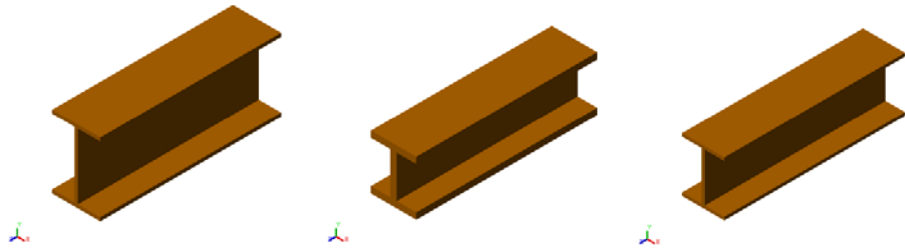


Change the I-Beam Geometry

Edit the dimension parameters to change the I-beam geometry. The four dimension parameters completely define the I-beam cross-section and length, so you do not need to reenter a new coordinate matrix or length inside the Solid block. To change the I-beam geometry:

- 1** In your SimMechanics model, double-click the I-beam subsystem block.
- 2** In the dialog box, enter the new dimension parameters.
- 3** Click **OK**.
- 4** Press **Ctrl+D** to update the I-beam display in Mechanics Explorer.

The following figure shows three possible iterations of the I-beam geometry, with the dimension parameters listed in the table.



ASTM Designation	L	h	w	t
S4×7.7	0.5	0.2	0.1	0.02
S3×7.5	0.5	0.1	0.2	0.005
S3×5.7	1	0.1	0.1	0.005

Model a Box Beam

In this section...

“Parameterize Cross-Section Coordinates” on page 2-79

“Build Model” on page 2-81

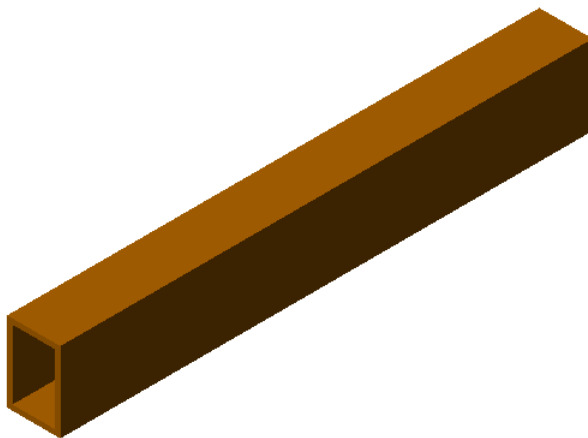
“Generate a Box Beam Subsystem” on page 2-83

“Visualize Box Beam in Mechanics Explorer” on page 2-86

“Change the Box Beam Geometry” on page 2-88

This example provides an example of a general extrusion. The example models a box-shaped beam with a single Solid block. Emphasis is on the parameterization of the beam in terms of its dimensions. You can rigidly connect multiple Solid blocks with Rigid Transform blocks to build compound rigid bodies that contain a greater level of detail. For more information about the General Extrusion geometry, see “Extrusion and Revolution Cross-Section Rules” on page 2-23.

The following figure shows the Mechanics Explorer display of the box beam that you model in this example.



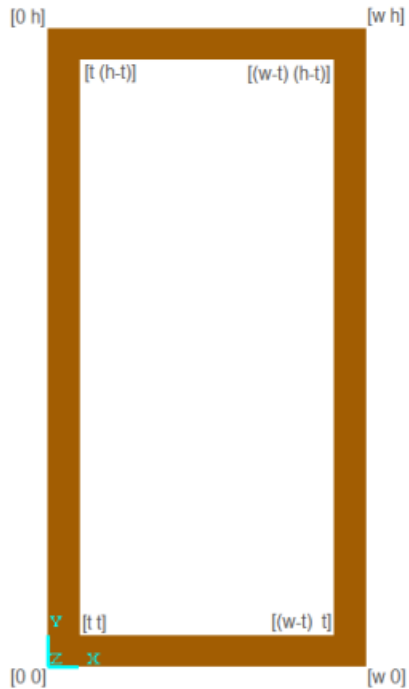
The box beam is a hollow rigid body. To completely specify the box beam cross-section, you must provide coordinates for *both* outer and inner cross-section profiles in a single MATLAB matrix. SimMechanics implements two rules that distinguish coordinates of outer and inner profiles:

- The coordinates of the *outer* cross-section profile must follow a counterclockwise order.
- The coordinate matrix of the *inner* cross-section profile must follow a clockwise order.

For more information, see “Extrusion and Revolution Cross-Section Rules” on page 2-23.

Parameterize Cross-Section Coordinates

This example parameterizes the cross-section coordinates in terms of relevant cross-section dimensions. Relevant dimensions include box beam height (h), width (w), and wall thickness (t). Once the box beam model is complete, you can change the dimension values to change the cross-section geometry. The following figure shows the vertex coordinates of the box beam cross-section in terms of h , w , and t .



A counterclockwise MATLAB matrix specifies the outer cross-section coordinates. For example:

```
outer_coordinates = [0 0; w 0; w h; 0 h; 0 0];
```

A *clockwise* MATLAB matrix specifies the inner cross-section coordinates. For example:

```
inner_coordinates = [t t; t (h-t); (w-t) (h-t); (w-t) t; t t];
```

The complete coordinate matrix is a combination of the outer and inner coordinate matrices:

```
coordinate_matrix = [outer_coordinates; inner_coordinates];
```

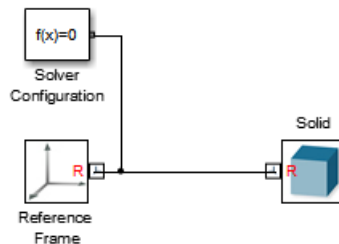
Build Model

The box beam model requires a single Solid block. To model a box beam rigid body:

- 1 Start a new Simulink model.
- 2 Add the following blocks to the model.

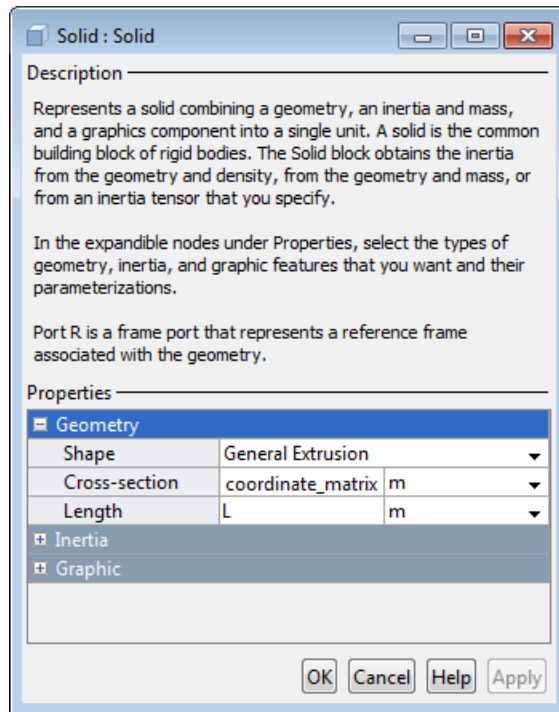
Block	Library	Quantity	Function
Reference Frame	Frames & Transforms	1	Provides ultimate reference frame to model. Frame is non-inertial.
Solid	Body Elements	1	Provides solid geometry, inertia, and graphic properties.
Solver Configuration	Simscape Utilities	1	Enables model assembly, visualization, and simulation.

- 3 Connect and name the blocks according to the following figure.



- 4 Double-click the Solid block.

- 5** In the Geometry section of the Solid dialog box, select **Shape > General Extrusion**.
- 6** In **Cross-section**, enter `coordinate_matrix`.
- 7** In **Length**, enter `L`.

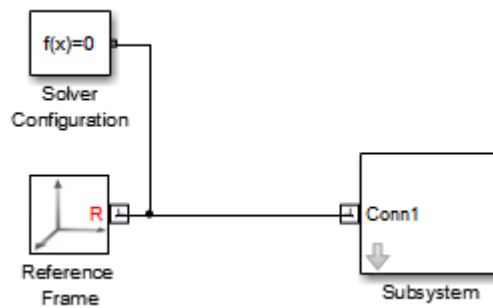


Note Parameters `coordinate_matrix` and `L` specify the cross-section and length of the box beam. In this example, you provide the values of `coordinate_matrix` and `L` in a subsystem mask. By parameterizing the box beam geometry in terms of `coordinate_matrix` and `L`, you can quickly change the box beam geometry without having to reopen the Solid block dialog box.

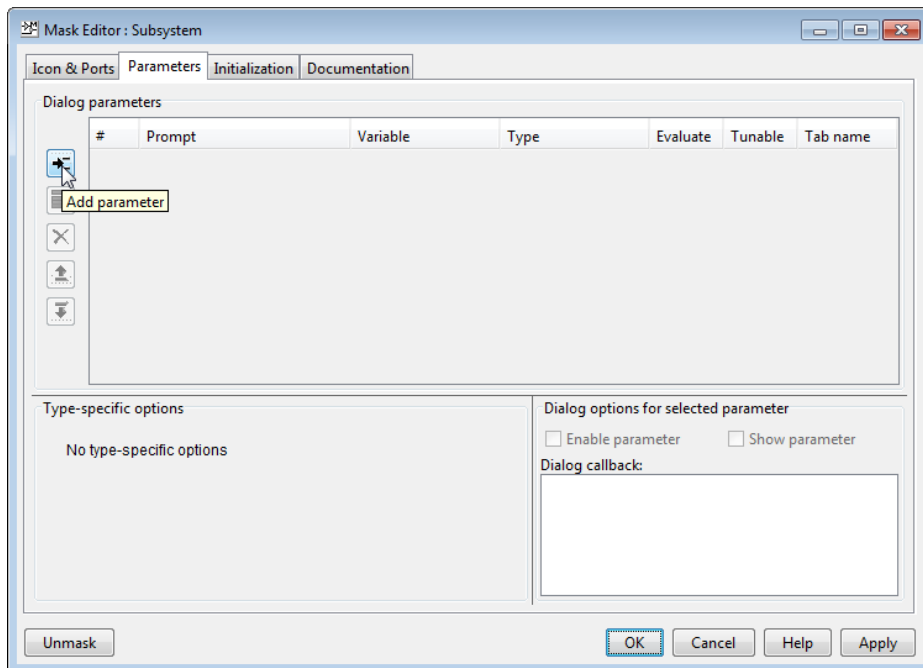
Generate a Box Beam Subsystem

The next step in modeling a box beam is to generate and mask a box beam subsystem.

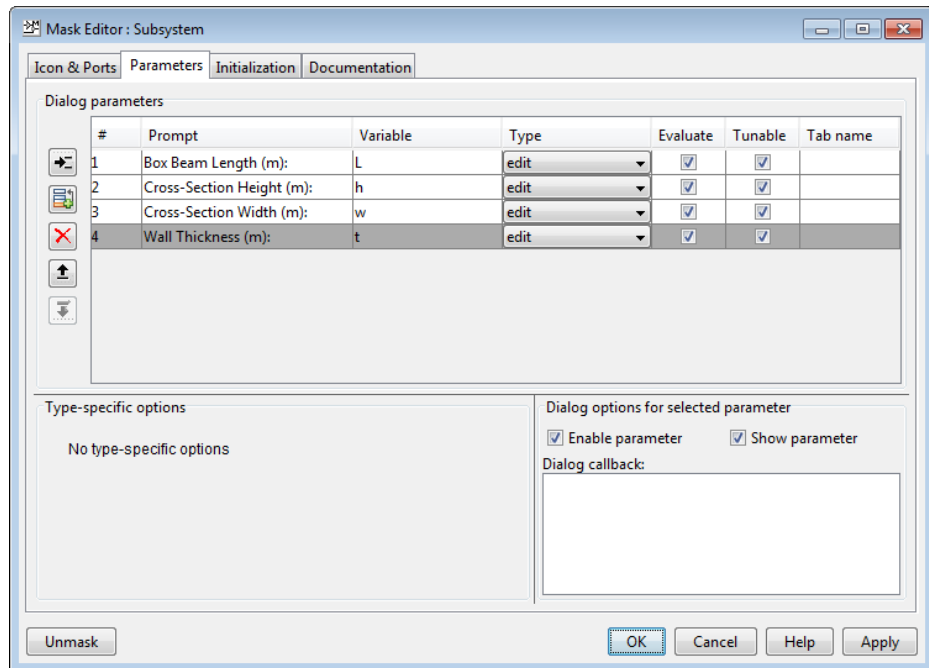
- 1 Click the box beam Solid block.
- 2 Press **Ctrl+G** to generate a subsystem for the box beam Solid block.



- 3 Press **Ctrl+M** to mask the box beam subsystem.
- 4 In the subsystem mask editor, click the **Parameters** tab.
- 5 On the left side of the **Dialog Parameters** pane, click the **Add parameter** button four times.



- 6 Enter **Prompt** and **Variable** for the four dimension parameters (L, h, w, and t).



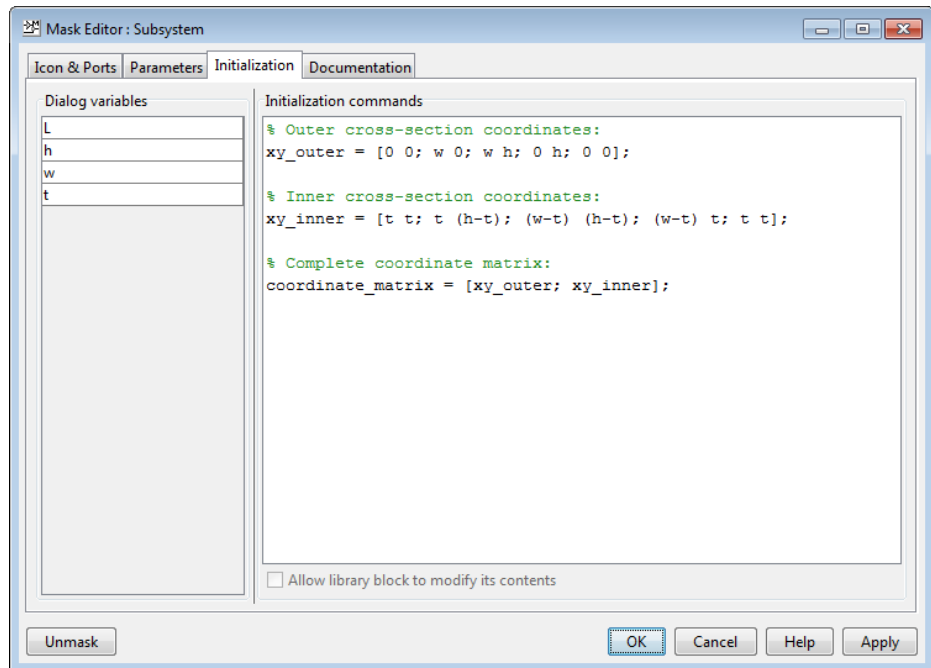
7 Click the **Initialization** tab.

8 In the **Initialization commands** pane, enter the cross-section coordinate matrix in terms of the cross-section dimensions (h, w, t):

```
% Outer cross-section coordinates:
xy_outer = [0 0; w 0; w h; 0 h; 0 0];
```

```
% Inner cross-section coordinates:
xy_inner = [t t; t (h-t); (w-t) (h-t); (w-t) t; t t];
```

```
% Complete coordinate matrix:
coordinate_matrix = [xy_outer; xy_inner];
```

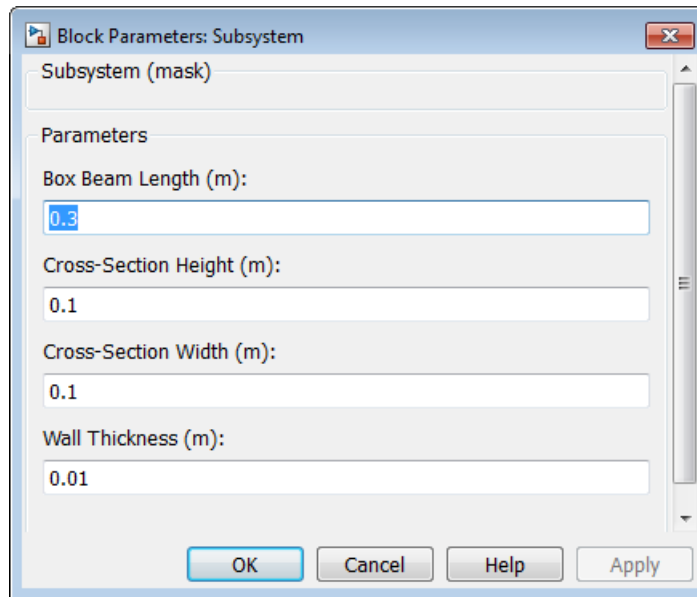


9 Click **OK**.

Visualize Box Beam in Mechanics Explorer

The box beam model is now complete. Render the box beam in Mechanics Explorer and verify that the geometry is correct.

- 1 Double-click the box beam subsystem block.
- 2 In the subsystem dialog box, enter the values of the four dimension parameters (L, h, w, t).

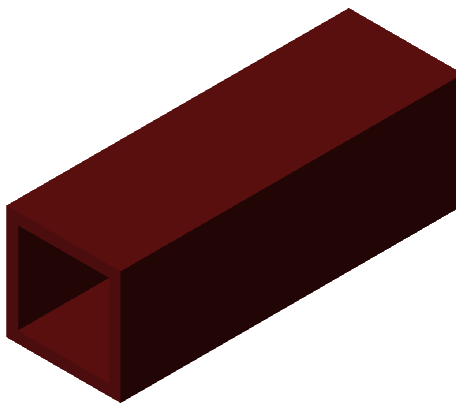


3 Click **OK**.

4 Press **Ctrl+D**.

A Mechanics Explorer window opens with a 3-D display of the box beam. Select different viewpoints to see the box beam from different perspectives. The following image shows an isometric view of the box beam model with the dimensions outlined in the table.

Parameter	Value
L	0.3
h	0.1
w	0.1
t	0.01

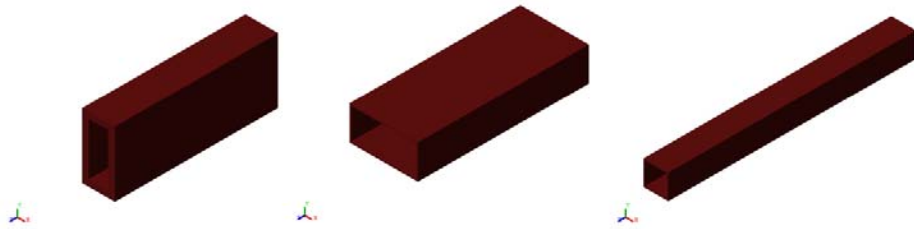


Change the Box Beam Geometry

Edit the dimension parameters to change the box beam geometry. The four dimension parameters completely define the box beam cross-section and length, so you do not need to reenter a new coordinate matrix or length inside the Solid block. To change the box beam geometry:

- 1** In your SimMechanics model, double-click the box beam subsystem block.
- 2** In the dialog box, enter the new dimension parameters.
- 3** Click **OK**.
- 4** Press **Ctrl+D** to update the box beam display in Mechanics Explorer.

The following figure shows three possible iterations of the box beam geometry, with the dimension parameters listed in the table.

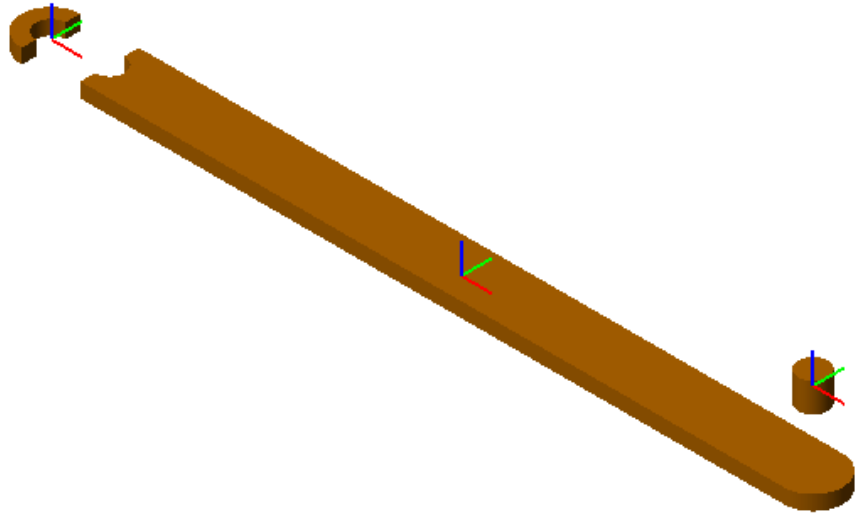


Iteration #	L	h	w	t
1	0.5	0.2	0.1	0.02
2	0.5	0.1	0.2	0.005
3	1	0.1	0.1	0.005

Model a Compound Binary Link

In this section...
“Build Model” on page 2-92
“Add Joint Frames” on page 2-93
“Add Geometry, Inertia, and Color” on page 2-94
“Generate and Mask Subsystem” on page 2-95
“Specify Subsystem Parameters” on page 2-99
“Visualize Model” on page 2-100
“Save Subsystem Block” on page 2-102

In this example, you model a binary link as a compound rigid body. The binary link contains one hole and one protruding peg. The hole and the peg each contain one frame to which you can connect a joint. You can use the binary link subsystem that you create in this example to assemble a complete four-bar linkage.



To accurately model the binary link, divide it into three pieces:

- Main body
- Hole section
- Peg

Building a parameterized model involves three tasks:

- 1** Create the block diagram.
- 2** Generate the rigid body subsystem.
- 3** Specify the rigid body parameters.

Build Model

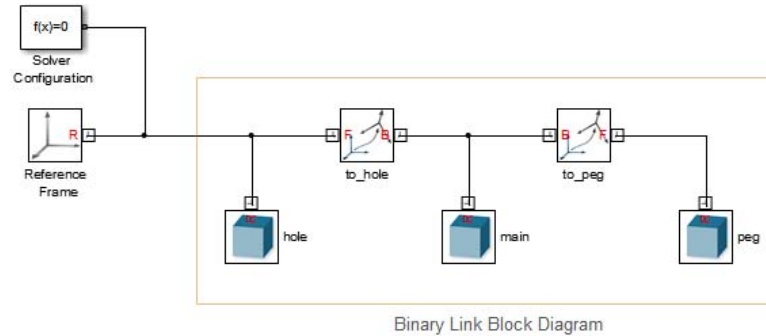
To start, create a block diagram with three rigidly connected Solid blocks. Each solid block represents one elementary piece of the binary link. Each Rigid Transform block provides a rigid connection between two spatially separated solid reference frames.

- 1 Start a new Simulink model.
- 2 Add the following blocks to the model canvas.

Block	Library	Quantity	Purpose
Solid	Body Elements	3	Specifies geometry, inertia, and color.
Rigid Transform	Frames and Transforms	2	Adds frames.
Reference Frame	Frames and Transforms	2	Defines local reference frame.
Solver Configuration	Simscape Utilities	1	Provided solver parameters that enable visualization during modeling.

- 3 Connect and name the blocks according to the following figure.

Note Rotate Rigid Transform block **to_hole** so frame port B faces Solid block **main**. This step ensures that all transformation parameters use frame port R of Solid block **main** as a reference.



In the dialog box, you can now specify the solid and frame parameters of each Solid and Rigid Transform block.

Add Joint Frames

The location of each end frame depends on the length of the binary link. By parameterizing the translation parameter in terms of the length, you can automatically adjust end frame position each time that you change the link length.

- 1 In the model, double-click each Rigid Transform block.
- 2 Expand **Translation**.
- 3 In **Method**, select Standard Axis.
- 4 In **Axis**, select +X.
- 5 In **Offset**, enter the appropriate value for the block.

Block	Offset
to_hole	$[-1/2 \ 0 \ 0]$
to_peg	$[1/2 \ 0 \ t]$

Note You must translate the peg along the +Z axis by a distance equal to the link thickness. If you do not, the peg appears inside the main link body.

6 In the units drop-down list, select cm.

7 Click OK.

Add Geometry, Inertia, and Color

Open the dialog box for each Solid block, and specify **Geometry**, **Inertia**, and **Graphic** parameters. The parameter names that you specify are common to all Solid blocks, except **Cross-section**. Later, when you generate a subsystem mask for the binary link model, you can enter the numeric values for the solid parameters.

To specify **Geometry**:

1 In the model canvas, double-click each Solid block.

2 In the dialog box, make sure that **Geometry** is expanded.

3 In the **Shape** drop-down list, select the appropriate shape for the block.

Block Name	Shape
hole	general Extrusion
main	General Extrusion
Peg	Cylinder

4 Enter the appropriate parameters for the shape that you specified.

Block Name	Shape	Parameter	Enter:
hole	General Extrusion	Cross-section	hole
		Length	t
main	General Extrusion	Cross-section	main
		Length	t

Block Name	Shape	Parameter	Enter:
peg	Cylinder	Radius	r
		Length	t

5 In the units drop-down lists, select cm.

To specify **Inertia**:

1 Expand **Inertia**.

2 In **Density**, enter rho.

3 In the units drop-down list, select $\text{g}/(\text{cm}^3)$.

To specify **Graphic**:

1 Expand **Graphic**.

2 Expand **Visual Properties**.

3 In **Color**, enter rgb.

4 Click **OK**.

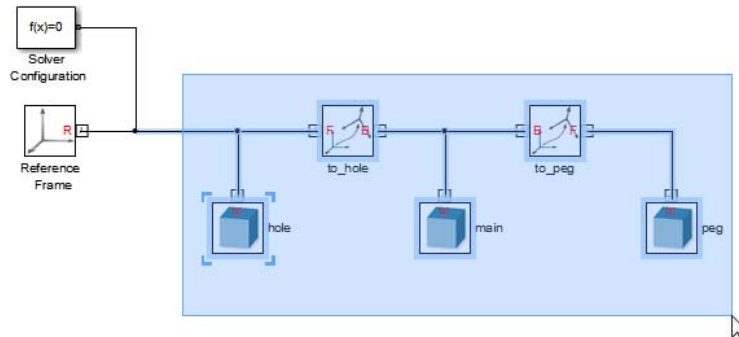
Generate and Mask Subsystem

Convert the binary link model into a subsystem. Later, you specify the numerical values for each subsystem parameter at the subsystem mask level. Working with subsystem blocks simplifies the modeling of multibody systems.

Generate Subsystem

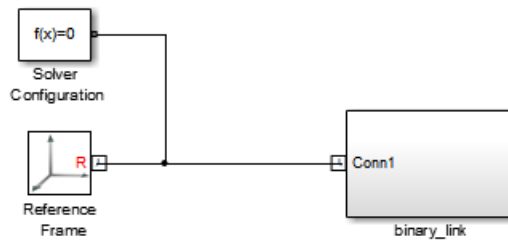
To generate a subsystem for the binary link model:

1 Select the blocks that make up the binary link model.



2 Right-click the highlighted section and select **Create Subsystem from Selection**.

3 Rename the subsystem `binary_link`.

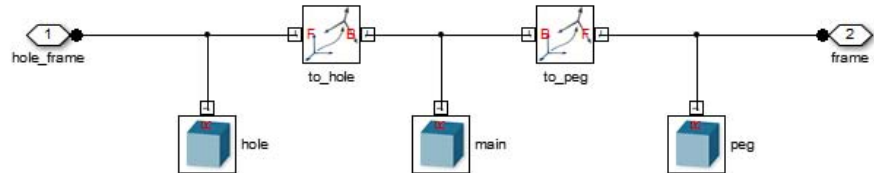


4 Double-click the subsystem block.

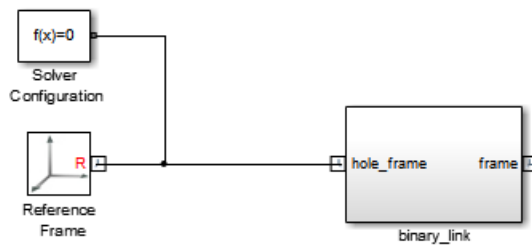
5 Copy and paste one instance of the PMC Port block



6 Connect and rename the PMC Port blocks according to the following figure.



Here is the resulting binary link subsystem. The subsystem contains two frame ports, one for each end of the binary link.



Mask Subsystem

A subsystem mask provides a convenient interface to define the cross-section coordinates for each General Extrusion shape. To mask the **binary_link** subsystem:

- 1 Right-click the subsystem block, and select **Mask > Create Mask**.
- 2 In the Mask Editor, click the **Parameters** tab.
- 3 Click the **Add Parameter** button to add six parameter lines.
- 4 In the **Prompt**, **Variable**, and **Tab Name** fields, enter the following strings.

Prompt	Variable	Tab Name
Length (cm)	l	Dimensions
Width (cm)	w	Dimensions
Thickness (cm)	t	Dimensions
Peg/Hole Radius (cm)	r	Dimensions
Mass Density (g/cm ³)	rho	Material
Color [R G B]:	rgb	Material

Dialog parameters

#	Prompt	Variable	Type	Evaluate	Tunable	Tab name
1	Length (cm)	l	edit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Dimensions
2	Width (cm)	w	edit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Dimensions
3	Thickness (cm)	t	edit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Dimensions
4	Peg/Hole Radius (cm)	r	edit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Dimensions
5	Mass Density (g/cm ³)	rho	edit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Material
6	Color [R G B]	rgb	edit	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Material

- 5 In the **Initialization** tab, define cross-section parameters for binary link segments **main** and **hole**:

Main Body Cross-Section

```
% Main body cross-section
top_xsec = [L/2 W/2; -L/2 W/2];
theta = (90:-1:-90)*pi/180;
semi_hole_xsec = [-L/2+r*cos(theta') r*sin(theta')];
bottom_xsec = [-L/2 -W/2; L/2 -W/2];
theta = (-90:1:90)*pi/180;
right_end_xsec = [L/2+(W/2)*cos(theta') (W/2)*sin(theta')];
main = [top_xsec;semi_hole_xsec;bottom_xsec;right_end_xsec];
```

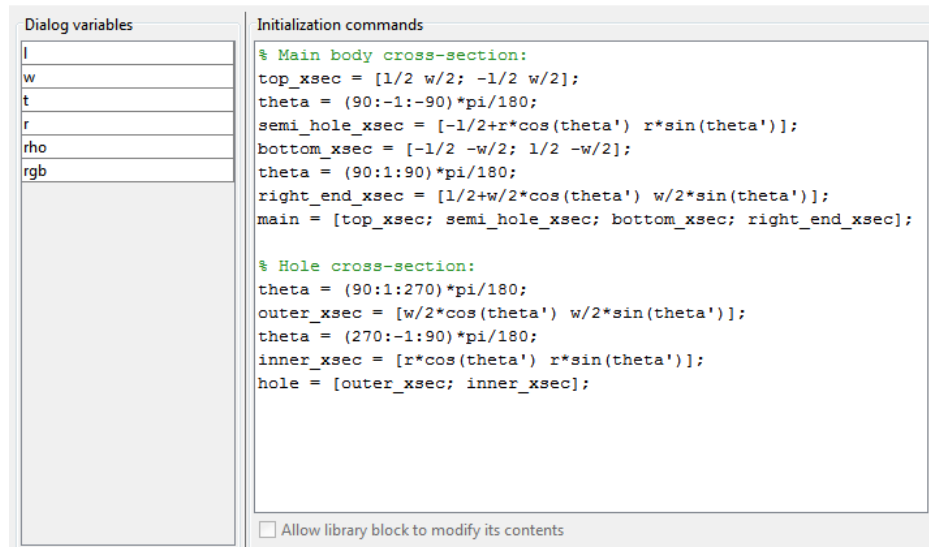
Hole Section Cross-Section

```
% Hole section cross-section
theta = (90:1:270)*pi/180;
```

```

outer_xsec = [(W/2)*cos(theta') (W/2)*sin(theta')];
theta = (270:-1:90)*pi/180;
inner_xsec = [r*cos(theta') r*sin(theta')];
hole = [outer_xsec; inner_xsec];

```



6 Click OK.

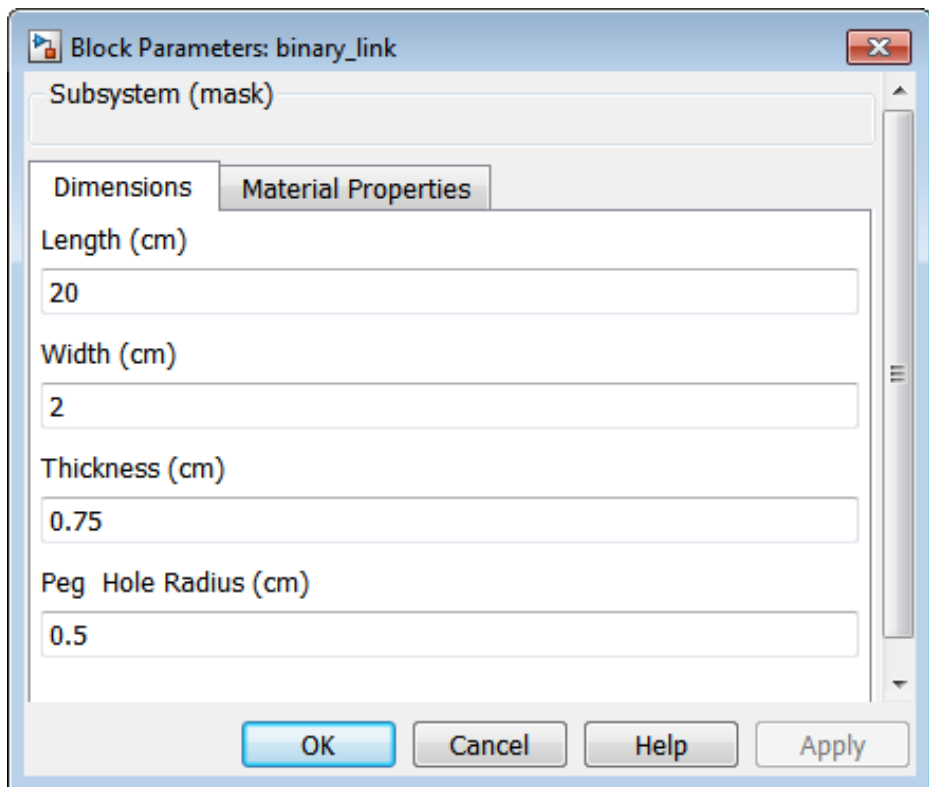
Specify Subsystem Parameters

The binary link model is nearly complete. The model contains two end frames to which you can connect joints, and an accurate shape. To complete the model, specify a numerical value for each parameter in the binary link subsystem. When you save the subsystem as a custom library block, the numerical values become the block's default values.

- 1 Double-click the binary_link subsystem block.
- 2 In the dialog box, specify a numerical value for each parameter.

The following table lists the numerical values used in this example.

Parameter	Tab	Numerical Value
Length (cm)	Dimensions	20
Width (cm)		2
Thickness (cm)		0.75
Peg Hole Radius (cm)	Material Properties	0.5
Color [R G B]		[0.8 0.45 0]



Visualize Model

Render the binary link in 3-D with the Mechanics Explorer utility. Verify that the shape and frames of the binary link are accurate.

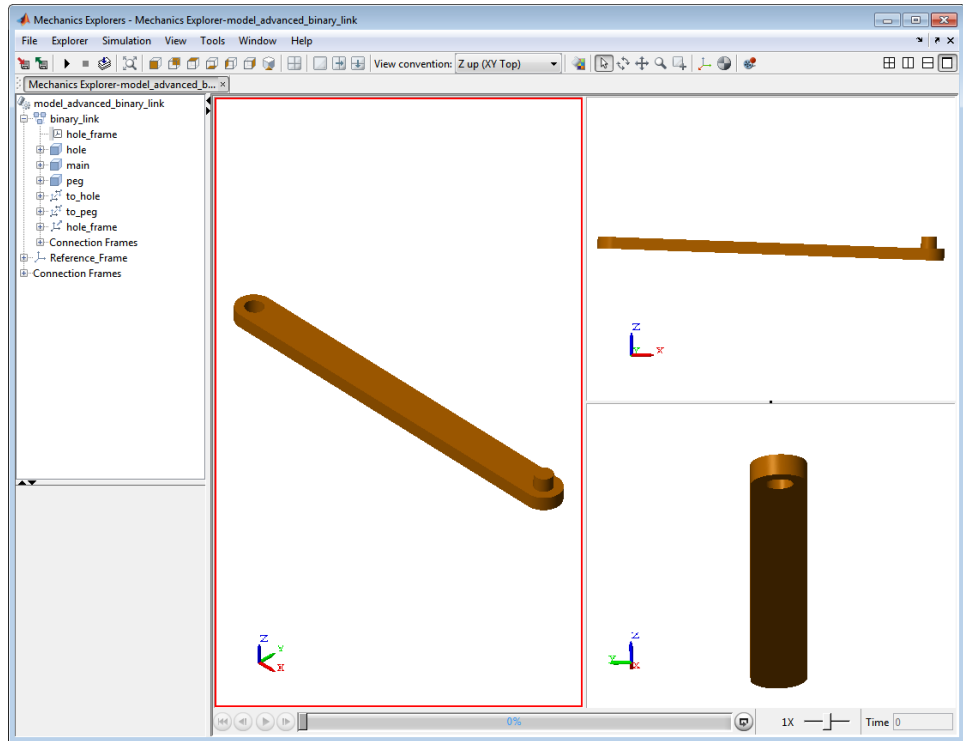
- 1 On the Simulink Editor menu bar, select **Simulation > Update Model**.
- 2 On the Mechanics Explorer window, pan, rotate, and zoom the display to examine the entire rigid body.

Shortcut Tip

To easily pan, rotate, and zoom, use the following shortcuts:

- Pan—Press **Shift** and hold the **Mouse Scroll** button. Move the mouse to pan the model window.
- Rotate—Click and hold the **Mouse Scroll** button. Move the mouse to rotate the window.
- Zoom—Press **Ctrl** and hold the **Mouse Scroll** button. Move the mouse up to zoom in, down to zoom out.

This figure shows the Mechanics Explorer display of the binary link model that you created. The figure shows a split visualization pane, each with a convenient viewpoint. For more information, see “Configure Mechanics Explorer Display” on page 6-2.



Save Subsystem Block

You can now use the binary link subsystem block in a multibody model. To reuse this subsystem, save the block in a custom library. For more information on how to create a custom library, see the Simulink documentation.

Mechanisms

- “Joints” on page 3-2
- “Joint Primitive Composition” on page 3-7
- “Multibody Assembly” on page 3-8
- “Model Report” on page 3-14
- “Assemble Double-Pendulum Model” on page 3-17
- “Refine Double-Pendulum Model” on page 3-28

Joints

In this section...
“Joint Frames” on page 3-2
“Joint Primitives” on page 3-3
“Joint Assembly” on page 3-4
“Joint Degrees of Freedom” on page 3-5

Joints constrain the mechanical degrees of freedom between two connecting rigid bodies. The primary purpose of joints is to limit motion of a mechanism or machine so an end effector can move along a specified path. Rigid bodies can contain the following degrees of freedom:

- Translational — linear displacement of one rigid body frame relative to another along a common axis.
- Rotational — angular displacement of one rigid body frame relative to another about a common axis

A free rigid body contains exactly six degrees of freedom: three rotational and three translational. The free rigid body can translate along any combination of three mutually orthogonal axes, and rotate about any combination of the same axes. When you connect two rigid bodies with a joint, you remove degrees of freedom between the two. Depending on the joint, you can remove anywhere from zero-six degrees of freedom. A joint that removes all six degrees of freedom is called **Weld** joint.

Note The Rigid Transform block is similar to the Weld Joint block. Both blocks remove all six mechanical degrees of freedom between the two connecting rigid bodies. However, the Rigid Transform block also allows you to maintain a specified distance and angle between the two rigid bodies.

Joint Frames

The joint block contains two frame ports, B and F. The ports identify the base and follower frames of a joint, respectively. You connect the base frame port

to one frame on one rigid body, and the follower frame port to another frame on a second rigid body. Switching the base and follower frames of a joint block has no effect on model assembly or simulation.

During simulation, joint blocks apply a time-varying transformation to the follower frame with respect to the base frame. The transformation depends on dynamic inputs (forces and torques) and the kinematic configuration of the model. Transformation components include rotation and translation about or along the joint primitive axes.

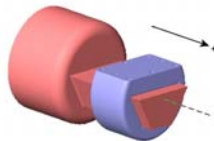
Joint Primitives

Each joint block contains a unique combination of *joint primitives* — elementary joint constructs that make up more advanced joints. The joint primitives represent the simplest joints you can find in SimMechanics. Three joint primitives exist: prismatic, revolute, and spherical. The following three sections briefly describe each primitive. The final section lists the primitives that make up each joint block.

Prismatic

Joint with one translational degree of freedom. The prismatic primitive allows the joint base and follower frames to translate relative to each other along a common axis. Joints with two prismatic primitives allow translation in a 2-D plane that contains the prismatic axes. Joints with three prismatic primitives allow translation in 3-D space.

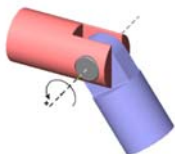
The following figure shows a schematic of the prismatic joint primitive.



Revolute

Joint with one rotational degree of freedom. The revolute primitive allows the joint base and follower frames to rotate relative to each other about a common axis. Joints with three revolute primitives allow rotation in 3-D space. The

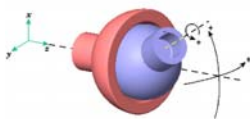
frames must each connect to a non-degenerate mass. The following figure shows a schematic of the revolute joint primitive.



Spherical

Joint with three rotational degrees of freedom. The spherical joint allows the joint base and follower frames to rotate about three mutually orthogonal axes.

The Spherical primitive is not a serial combination of revolute primitives. Such a combination is susceptible to Gimbal lock — an event in which two revolute axes align, resulting in the loss of one rotational degree of freedom. *The Spherical primitive is not susceptible to Gimbal lock at any time.* The following figure shows a schematic of the spherical joint primitive.



Joint Assembly

During assembly, each joint block fixes base and follower frames according to the following rules.

Joint primitive	Constraint
Prismatic	Align prismatic axes of base and follower frames. Origins need not be coincident.
Revolute	Align rotation axes of base and follower frames.
Spherical	No constraint. Axes can have any relative orientation. In compound joints, frame origins can maintain a non-zero distance between them.

Compound joints — joints that contain multiple joint primitives — impose

Joint Degrees of Freedom

The following table identifies the mechanical degrees of freedom of SimMechanics joints. Joints appear in alphabetical order. X, Y, and Z refer to the three axes of a Cartesian coordinate system.

Joint	Translational Degrees of Freedom			Rotational Degrees of Freedom		
	X	Y	Z	X	Y	Z
6-DOF*	✓	✓	✓	✓	✓	✓
Bearing	—	—	✓	✓	✓	✓
Bushing	✓	✓	✓	✓	✓	✓
Cartesian	✓	✓	✓	—	—	—
Cylindrical	—	—	✓	—	—	✓
Gimbal	—	—	—	✓	✓	✓
Planar	✓	✓	—	—	—	✓
Prismatic	—	—	✓	—	—	—
Rectangular	✓	✓	—	—	—	—

Joint	Translational Degrees of Freedom			Rotational Degrees of Freedom		
	X	Y	Z	X	Y	Z
Revolute	—	—	—	—	—	✓
Spherical*	—	—	—	✓	✓	✓
Telescoping*	—	—	✓	✓	✓	✓
Universal	—	—	—	✓	✓	—
Weld	—	—	—	—	—	—

Note Joints with asterisk * provide three rotational degrees of freedom through a *spherical* primitive. The spherical primitive is not susceptible to gimbal lock.

Joint Primitive Composition

The following table identifies the joint primitives that make up the different SimMechanics joints. For more information, see “Joints” on page 3-2.

		Prismatic Primitives			
		0	1	2	3
Revolute Primitives	0	Weld Joint	Prismatic Joint	Rectangular Joint	Cartesian Joint
	1	Revolute Joint	Cylindrical Joint	Planar Joint	—
	2	Universal Joint	—	—	—
	3	Gimbal Joint	Bearing Joint	—	Bushing Joint
Spherical Primitives	1	Spherical Joint	Telescoping Joint	—	6-DOF Joint

Multibody Assembly

In this section...
“Workflow” on page 3-8
“Identify Joint Requirements” on page 3-8
“Connect Rigid Bodies with Joints” on page 3-9
“Specify Joint State Targets” on page 3-10
“Check Assembly” on page 3-11

To model a mechanism or machine, you connect rigid bodies with joints that constrain their relative degrees of freedom. This process is known as **multibody assembly**.

Note Before you can assemble a multibody model, you must create the rigid body subsystems for that model.

Workflow

Assembling a multibody system involves the following steps:

- 1 Connect rigid bodies subsystems with joint blocks.
- 2 Specify joint state targets.
- 3 Check assembly.
- 4 Adjust frames and joint state targets if necessary.

Identify Joint Requirements

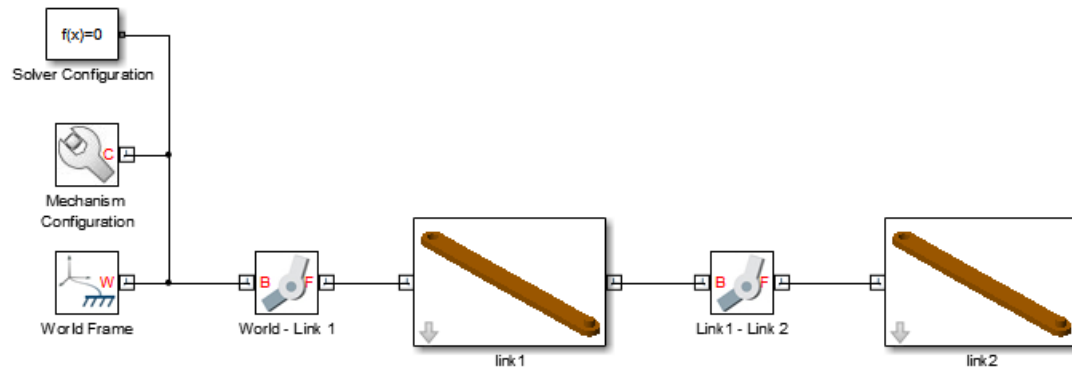
Each joint block connects precisely two rigid body frames. The type of joint block determines how the two rigid bodies can move with respect to each other. Two types of motion are possible:

- Rotation — provided by revolute and spherical joint primitives
- Translation — provided by prismatic joint primitives

To identify the correct joint block for your application, see “Joints” on page 3-2. The most commonly used joint blocks are Prismatic Joint, Revolute Joint, and Spherical Joint.

Connect Rigid Bodies with Joints

Drag the rigid body subsystem blocks onto the SimMechanics model. Then, select and drag the appropriate joint blocks from the Joints library. Connect the base and follower frames of each joint block to two frames on two distinct rigid body subsystems. The following figure shows the assembly of a double-pendulum model.



Caution Carefully check the position and orientation of the rigid body frames that you connect to joint blocks. Joint frames that possess either incorrect position or orientation can cause the model to assemble in an unexpected configuration.

In severe cases, joint frames cause kinematic conflicts that lead to assembly failure and simulation errors. Kinematic conflicts due to incorrect joint frames are more likely in closed kinematic loops such as the four-bar linkage.

Specify Joint State Targets

You can guide model assembly at time $t=0$. Each joint block provides a **State Targets** menu where you can specify the initial position and velocity of each joint primitive. You can select high or low priority for each joint state target.

During assembly, SimMechanics attempts to meet all specified state targets. The following rules apply to joint state targets:

- SimMechanics attempts to meet all state target values *precisely*.
- If two or more joint state targets are mutually incompatible, SimMechanics attempts to meet high priority targets *precisely*, and low priority targets *approximately*.
- If a state target is not met, the Model Report utility identifies the joint with the unmet state target.

Note Unmet state targets do not cause assembly failure or simulation errors, but can cause the model to assemble with an unexpected configuration.

The following figure shows the double-pendulum model with joint state targets of +15 deg for each revolute joint.



Check Assembly

Joints assemble base and follower frames according to a well-defined set of rules. For example, the Revolute Joint block makes the +Z axes of base and follower frames coincident in space. If the +Z axes of base and follower frames are not properly positioned, an unexpected joint configuration can result.

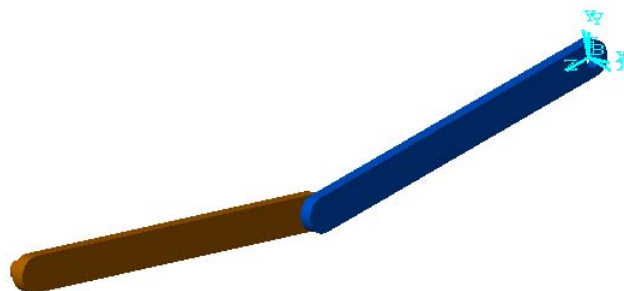
Use Mechanics Explorer to check the base and follower frames of each joint in a model.

- 1 In the tree-browser pane of Mechanics Explorer, click the name of each joint block.

Mechanics Explorer highlights the base and follower frames of the selected joint in the visualization pane.

- 2 Check the base and follower frames of the selected joint for an unexpected position or orientation.

The following figure shows an improperly configured double-pendulum model. The Z-axis of the top revolute joint, **World-Link1**, points along the length axis of the binary link. Since the +Z-axis specifies the rotational axis of the Revolute Joint block, **link1** rotates about its length axis, producing unexpected results during simulation.



Model Report

In this section...

“Open Model Report” on page 3-14

“Model Report Tabs” on page 3-14

“Status Icons” on page 3-16

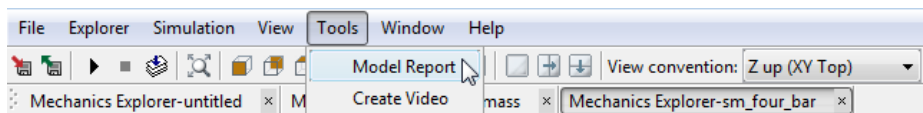
Model Report is a SimMechanics tool that provides model assembly status and parameters. Use Model Report to:

- Identify joints and constraints with assembly issues.
- Identify joints with unmet state targets.
- Compare specified and actual joint state targets.
- Obtain relevant statistics for a model

Open Model Report

Model Report is accessible from Mechanics Explorer. To open Model Report:

- 1 Update or simulate a SimMechanics model.
- 2 In the Mechanics Explorer menu bar, click **Tools > Model Report**.



- 3 Model Report opens with assembly status and parameters relevant to the current model.

Model Report Tabs

Model Report contains one header section and three tabs:

- **Header** — Provides model-wide assembly status.

- **Joints** — Provides assembly status and state target values for each joint block in a model. The following figure shows the **Joints** tab for example `sm_four_bar`.




Joint	Assembled	Primitive	Position					Velocity				
			Actual	Specified	Unit	Priority	Status	Actual	Specified	Units	Priority	Status
Base_Cran...	●	Rz	+150	+150	deg	High	●	-360	-360	deg/s	High	●
Base_Rock...	●	Rz	+173.824		deg			-179.769		deg/s		
Connecto...	●	Rz	+67.6893		deg			-249.628		deg/s		
Crank_Co...	●	Rz	-43.8653	-45	deg	Low	▲	+429.858		deg/s		

- **Constraints** — Provides assembly status for each constraint block in a model. The following figure shows the **Constraints** tab for example `sm_four_bar`.
- **Statistics** — Provides model-wide statistics. Parameters include number of joints, constraints, and kinematic degrees of freedom in a model. The following figure shows the **Statistics** tab for example `sm_four_bar`.

Type	Value
Number of rigidly connected components (excluding ground)	6
Number of joints (total)	6
Number of explicit tree joints	6
Number of implicit 6-DOF tree joints	0
Number of cut joints	0
Number of constraints	0
Number of tree degrees of freedom	6
Number of position constraint equations (total)	0
Number of position constraint equations (non-redundant)	0
Number of mechanism degrees of freedom (minimum)	6
State vector size	12
Average kinematic loop length	0

Status Icons

Model Report uses three icons to identify the assembly status of a model, joint block, or constraint block.

Status Icon	Description
	Assembled without issues. In joint blocks, the icon indicates state target was successfully met.
	Assembled with issues. In Joint blocks, the icon indicates the state target was approximately met.
	Not assembled. In Joint blocks, the icon indicates the state target was not met.

Assemble Double-Pendulum Model

In this section...

“Summary” on page 3-17
“Requirements” on page 3-17
“Build Multibody Model” on page 3-18
“Specify Binary Link Parameters” on page 3-19
“Inspect Frames” on page 3-19
“Change Frame Orientation” on page 3-22
“Change Gravity Vector” on page 3-24
“Specify Joint State Targets” on page 3-25
“Save Model” on page 3-27

In this example, you assemble a double-pendulum model. The model contains two binary link subsystems and a World frame that you interconnect with two revolute joints. The double-pendulum provides the simplest example of a multibody model. Once you have completed model assembly, you can proceed to actuate and sense pendulum motion.

Summary

This example shows you to:

- Connect rigid bodies with joints
- Align rigid body frames so joints assemble successfully
- Specify initial joint position and velocity

Requirements

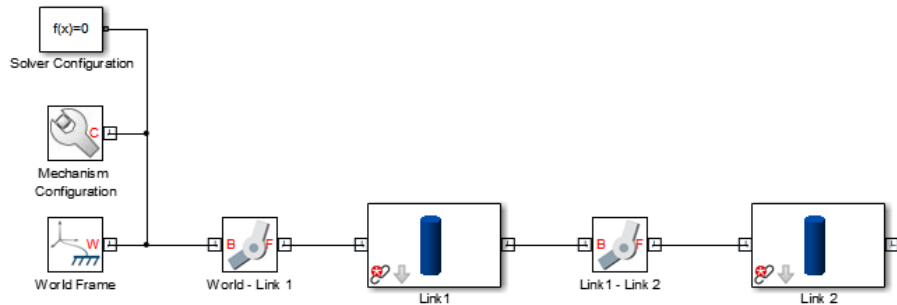
The example assumes the following:

- Completion of example “Model a Simple Binary Link” on page 2-31.
- Optional completion of example “Model a Compound Binary Link” on page 2-90.

- Mechanics Explorer set to open on model update (default setting).
- Block names matching example.

Build Multibody Model

Build the block diagram to represent the double-pendulum, shown in the following figure.



To build the block diagram follow these steps:

- 1 Drag the following blocks to a new Simulink Editor window.

Block	Library	Quantity	Notes
World Frame	Frames & Transforms	1	
Revolute Joint	Joints	2	
Binary Link	Custom Library	2	Custom block you create in example “Model a Simple Binary Link” on page 2-31
Solver Configuration	Simscape Utilities	1	

- 2 Connect and rename the blocks as shown in the previous figure.

Note The example assumes all blocks have the names indicated in the figure.

Specify Binary Link Parameters

Enter the numerical values for the solid parameters of each binary link subsystem.


- 1** Double-click the **Link 1** and **Link 2** subsystem blocks.
- 2** In the **Dimensions** tab of the subsystem dialog box, enter numerical values for **Radius** and **Length**.
- 3** In the **Material Properties** tab of the subsystem dialog box, enter numerical values for **Density** and **Color**.
- 4** Click **OK**.

The following table provides the numerical values used in this example. Except for color, the values apply to both links.


Parameter	Tab	Value
Radius	Dimensions	1
Length	Dimensions	20
Density	Material Properties	2.70
Color (Link 1)	Material Properties	[0 0.4 0.9]
Color (Link 2)	Material Properties	[0.8 0.45 0]

Inspect Frames

The end frames in each Binary Link subsystem contain a default orientation that is not well suited for double-pendulum assembly. To see this, update the model and inspect the frames in Mechanics Explorer.

- 1** In the Simulink Editor menu, select **Simulation > Update Diagram**.
- 2** In the Mechanics Explorer toolbar, click the  icon.

The visualization pane of Mechanics Explorer splits vertically into two panes of equal size.

- 3** Select the right visualization pane, and click the  icon.

The right visualization pane splits horizontally into two panes of equal size.

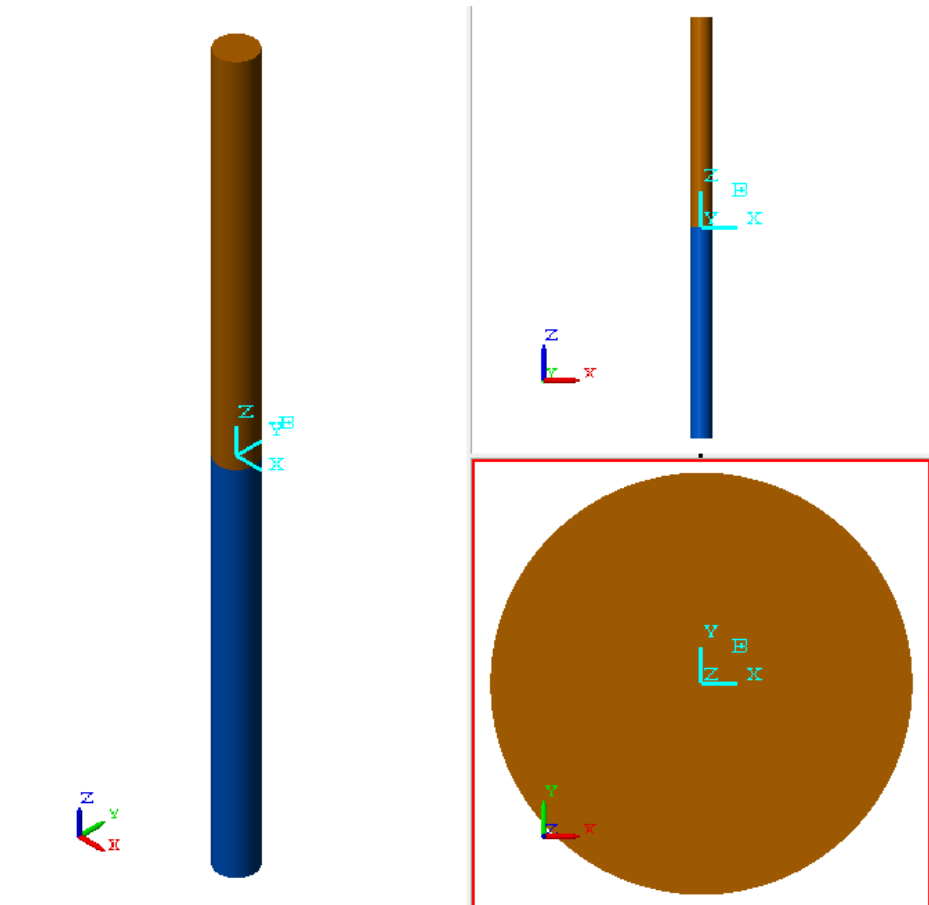
- 4** Select a different view for each visualization pane. The following table provides the views selected in this example.

Pane	View
Left	Isometric
Right Top	Front
Right Bottom	Top

Note Splitting the visualization pane into three different views allows you to accurately check the position and orientation of each frame.

- 5** In the model browser pane of Mechanics Explorer, select **Link1_Link2**.

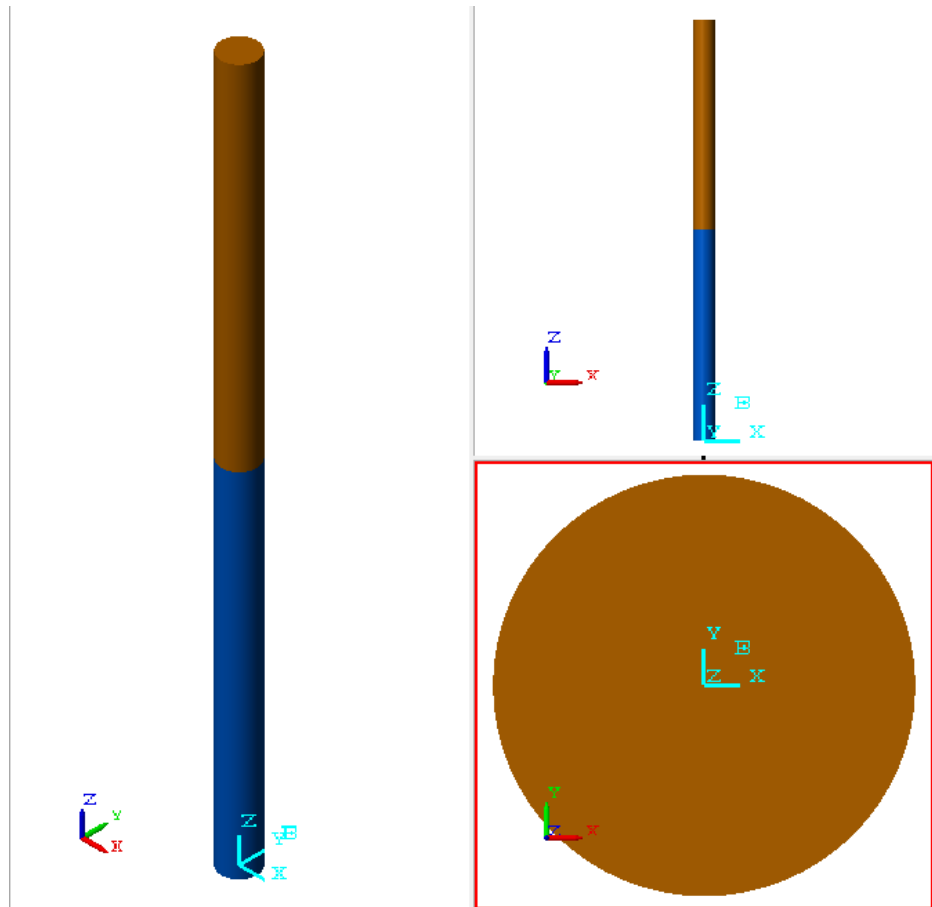
The block Link 1 — Link 2 represents the joint that connects the two binary links. Joint base and follower frames are coincident.



Note the Z-axes of the joint frames align with the cylinder axes of the two binary links. In SimMechanics, the Z-axis is the axis of revolution for all revolute joints. As a result, the binary links revolve about the wrong axes. To correct this issue, you must change the orientation of the Binary Link end frames.

- 6 In the model browser pane of Mechanics Explorer, select **World_Link_1**.

The block World — Link 1 represents a revolute joint that connects the World frame to Link 1. Joint base and follower frames are coincident.



Note the entire double pendulum assembly stands along the +Z axis of the World frame.

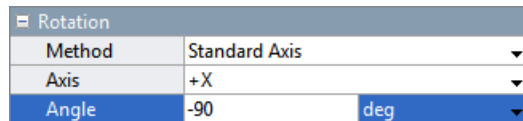
Change Frame Orientation

When you connect two rigid body frames with a revolute joint, you make their Z-axes coincident. The direction of the aligned Z-axes defines the rotation axis of the revolute joint. In the current model, the Z-axes of the end frames are parallel to the cylinder axes. This configuration allows rotation only about the cylinder axis.

To make the rotation axis or the revolute joints orthogonal to the cylinder axes, you must specify a rotation parameter in each Rigid Transform block. Each binary link subsystem contains two Rigid Transform blocks that require a rotation parameter.

A rotation parameter of -90° about the +X-axis aligns the Z-axes with the -Y axis. To specify the rotation parameter:

- 1 In each Binary Link subsystem block, click the **Look under mask** arrow.
- 2 In the subsystem block diagram, double-click each Rigid Transform block.
- 3 In the dialog box of each Rigid Transform block, expand the **Rotation** node.
- 4 In the **Method** drop-down menu, select **Standard Axis**.
- 5 In the **Axis** drop-down menu, select **+X**.
- 6 In the **Angle** field, enter **-90**.

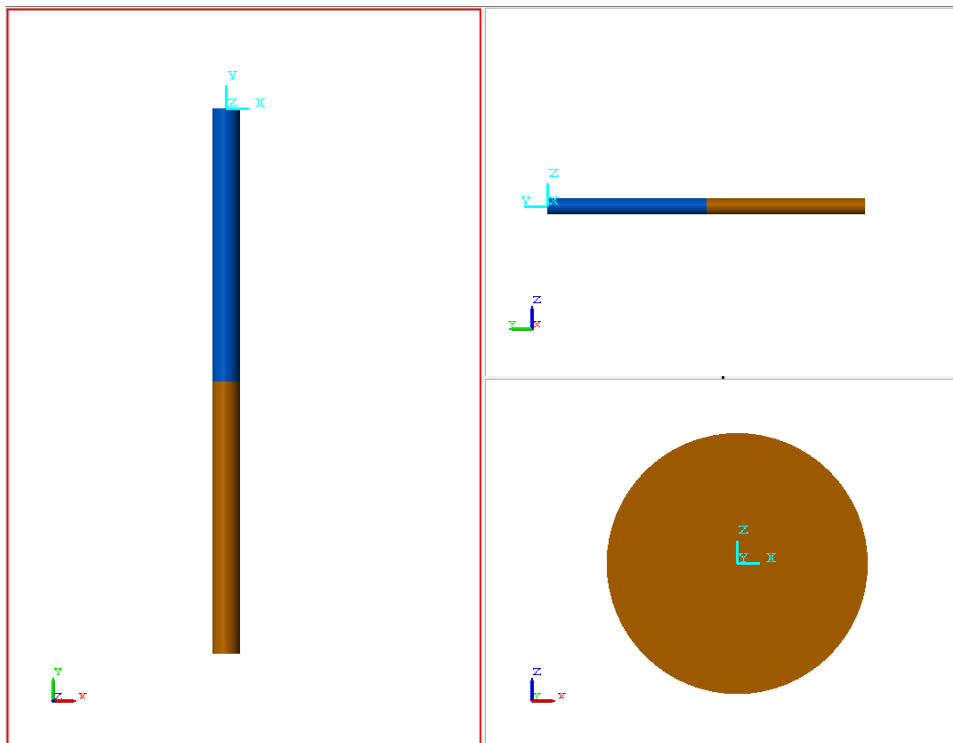


- 7 Click **OK**.
- 8 Press **Ctrl+D** to update the model.

Mechanics Explorer displays the updated model. Click **Link_Link_2** and **World_Link_1** to view the joint base and follower frames.

Note Specify the rotation parameter for the four Rigid Transform blocks (two per Binary Link subsystem) present in the model.

The following figure shows the new end frame orientation. Note the end frame Z-axes are now orthogonal to the cylinder axis.



Change Gravity Vector

The double-pendulum now aligns with the Y-axis. To simulate gravity along the Y-axis, you must change the **Gravity** parameter in the dialog box of the Mechanism Configuration block. To change the gravity vector, follow these steps:

- 1 In the model, double-click the Mechanism Configuration block.
- 2 In the **Gravity** field, enter $[0 \ -9.81 \ 0]$.
- 3 Click **OK**.

Specify Joint State Targets

You can change the default configuration of the double-pendulum at time $t=0$. The Revolute Joint block provides a **State Targets** menu with **Position** and **Velocity** parameters you specify. You can choose to specify joint position, velocity, or both.

The **Position** parameter specifies the initial angle of the joint follower frame with respect to the base frame about the common Z-axis. The parameter observes the right-hand rule.

The **velocity** parameter specifies the initial angular velocity of the follower frame with respect to the base frame Z-axis-axis. The parameter observes the right-hand rule.

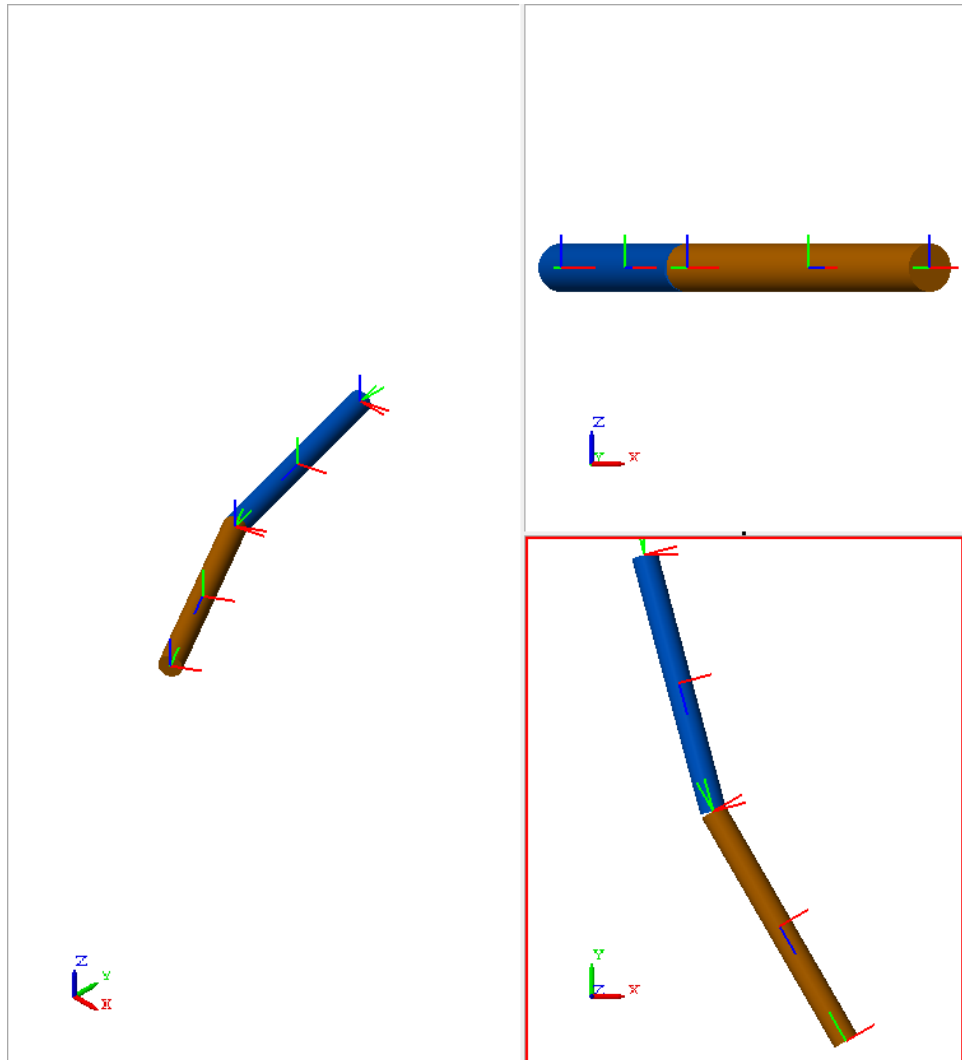
Specify the state targets for the two revolute joints in this example. The following table provides the values used in the example.

Joint	State Target	Numeric Value
Link 1 — Link 2	Position	15
World — Link 1		15

- 1** In the model, double-click each Revolute Joint block.
- 2** In the dialog box, expand the **State Targets** menu.
- 3** Click the **Position** check box.
- 4** In the **Value** field, enter 15.
- 5** Click **OK**.
- 6** Press **Ctrl+D** to update the model

Mechanics Explorer displays the updated model. Note the new configuration of the double-pendulum assembly. Each revolute joint now assembles with an initial 15 deg angle separating base and follower frames.

The following figure shows the new initial configuration of the model.



Assembly of the double-pendulum model is now complete. You can add internal forces to the revolute joints, sense motion between two frames, and simulate the model.

Save Model

Save the model in a convenient directory for future use. Name the model `double_pendulum`. Once you have completed this example, try adding detail to the model. See “Refine Double-Pendulum Model” on page 3-28.

Refine Double-Pendulum Model

In this section...

“Add Advanced Binary Link Subsystems” on page 3-28

“Update Model” on page 3-29

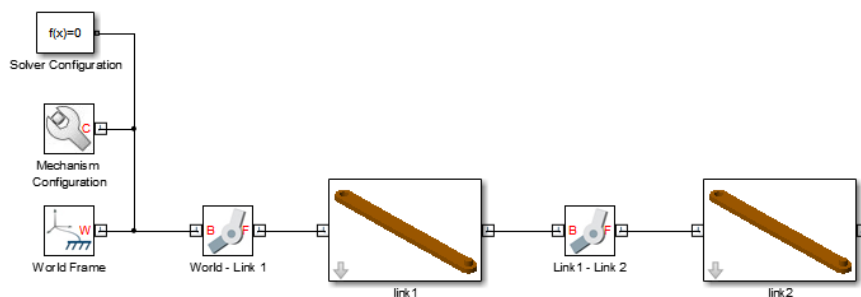
“Adjust Frames” on page 3-30

“Save Model” on page 3-32

Add Advanced Binary Link Subsystems

Now that the model is complete, you can add detail to the binary link subsystems, or replace them with more advanced versions. Example “Model a Compound Binary Link” on page 2-90 provides one such version. To replace the binary link subsystems, follow these steps:

- 1 Open the custom library that contains the advanced binary link block.
- 2 Delete the binary link subsystem blocks from the double-pendulum model.
- 3 Drag two instances of the advanced binary link model to the double-pendulum model canvas.
- 4 Connect and rename the binary link blocks according to the following figure.



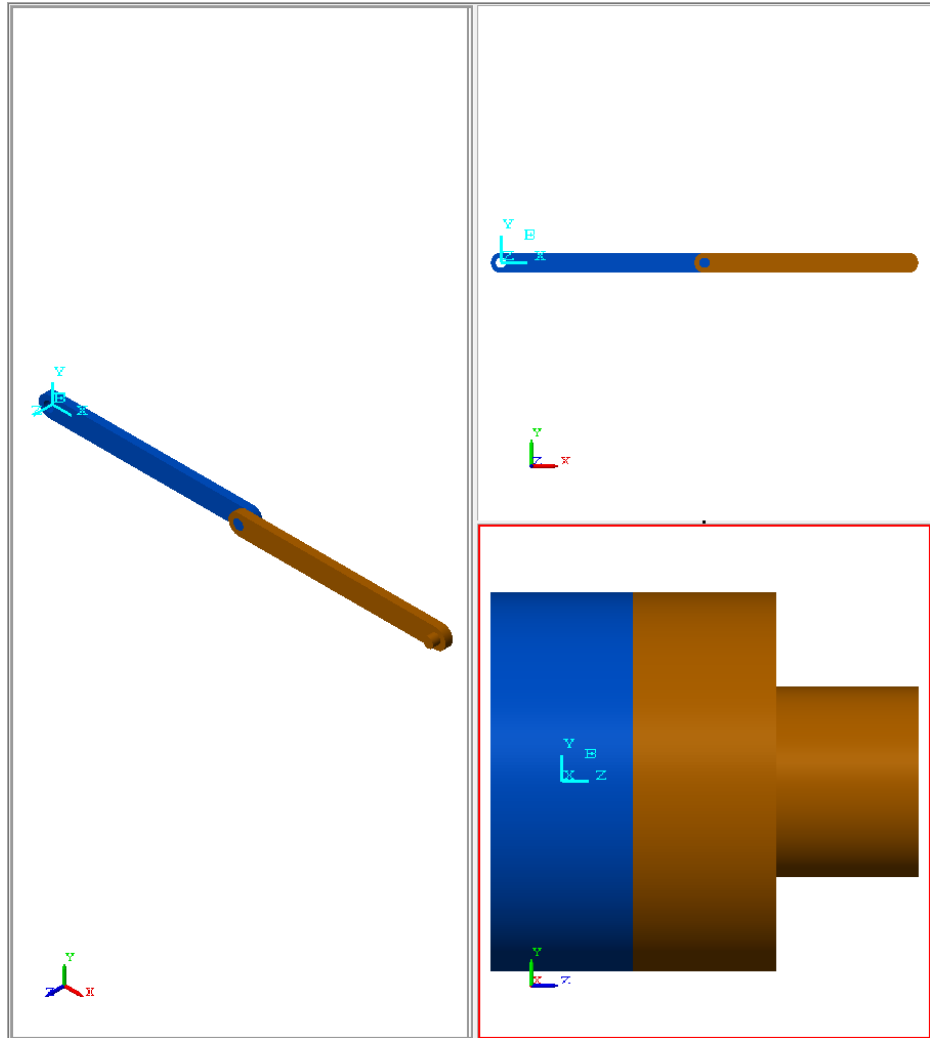
- 5 Double-click the **link1** subsystem block.

- 6** In the dialog box, click the **Material Properties** tab.
- 7** In **Color**, enter [0 0.4 0.9] and click **OK**.

Update Model

Before the model assembles properly, you must make minor adjustments to the joint frames. First identify which joint frames need adjustment. Clear any joint state targets, and examine the model in its initial configuration.

- 1** In the model, double-click **World-Link1** and **Link1-Link2** joint blocks.
- 2** Expand the **State Targets** menu.
- 3** Clear **Specify Position Target** and click **OK**.
- 4** In the Simulink Editor menu bar, select **Simulation > Update Diagram**.
- 5** In the tree browser of Mechanics Explorer, select the name of a joint block, and check that base and follower frame orientation is as expected.



Adjust Frames

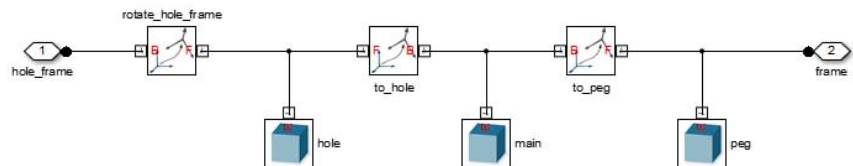
Note the joint World-Link1 requires adjustment to its frames. In its initial configuration, the joint aligns link1 with the X-axis. Since gravity acts along the -Y-axis, it is desirable to change the joint frames so the upper link aligns with the -Y-axis at time $t=0$.

One solution is to specify a joint position target of -90 deg. This solution causes the joint follower frame to assemble at a -90 deg angle about the $+Z$ axis with reference to the base frame. The rotation aligns the upper link with the Y -axis.

A second solution is to rotate the follower frame of the joint -90 deg about the $+Z$ -axis. Use the following steps:

- 1 In the double-pendulum model, click the **Look under mask** arrow of the link1 block.
- 2 Add one Rigid Transform block to the subsystem block diagram.
- 3 Connect and name the blocks according to the following figure.

Note Frame port B must face the hole_frame port block. If frame port F faces the hole_frame port, the model assembles in a different configuration.



- 4 Double-click the rotate_hole_frame block.
- 5 In the **Rotation** menu, select **Method** Standard Axis.
- 6 In the **Axis** drop-down menu, select $+Z$.
- 7 In **Angle**, enter -90 .
- 8 In the **Translation** menu, check that **Method** is set to None and click **OK**.
- 9 In the Simulink Editor window, click **Simulation > Update Diagram**.

The following figure shows the updated model. Notice the orientation of the double-pendulum. Both links now align with the $-Y$ axis.

Save Model

Save the model in a convenient directory for future use. Name the model `double_pendulum_detailed`. Once you have completed this example, try adding internal forces to the model. See “Add Internal Forces to Double-Pendulum Model” on page 4-12.

Actuation & Motion Sensing

- “Model Actuation Workflow” on page 4-2
- “Actuate Four-Bar Crank Joint” on page 4-3
- “Add Internal Forces to Double-Pendulum Model” on page 4-12
- “Actuation” on page 4-18
- “Force and Motion Sensing” on page 4-26

Model Actuation Workflow

You can specify an external force that acts on a rigid body frame, or an internal force that acts between two rigid body frames. To add a force or torque to a model, select and drag a block from the Forces and Torques library. Blocks in the Joints library contain actuation options that you can use to actuate a joint.

Before you can add a force or torque to a model, you must create a valid model that you can actuate. Then, use the following steps:

- 1 Determine if the application requires force, torque, or both force and torque as input.
- 2 Identify the position and orientation of the force and torque inputs.

Note SimMechanics forces and torques act on frames. You can use the Rigid Transform block to create a new frame with the correct actuation position and a convenient orientation.

- 3 Identify the types of forces and torques that are appropriate for the application. SimMechanics supports forces and torques with generic formulas, spring and damper forces, and forces with $1/r_2$ dependencies. See “Actuation” on page 4-18.
- 4 Select and drag the corresponding blocks from the Forces & Torques library onto the model window.
- 5 Connect the frame ports of the force blocks to the actuation frame ports or connection lines. See “Frame Connection Rules” on page 1-16.
- 6 Double-click each block and, depending on the block, enter the force parameters or select the force and torque input signals to specify.
- 7 Set up the model for dynamic simulation.
- 8 Press **Ctrl+T** to simulate the model.
- 9 Check for unexpected behavior that indicates force and torque parameters, input signals, or connection frames are incorrect.

Actuate Four-Bar Crank Joint

In this section...

“Open Model” on page 4-4

“Expose Actuation Input Port” on page 4-4

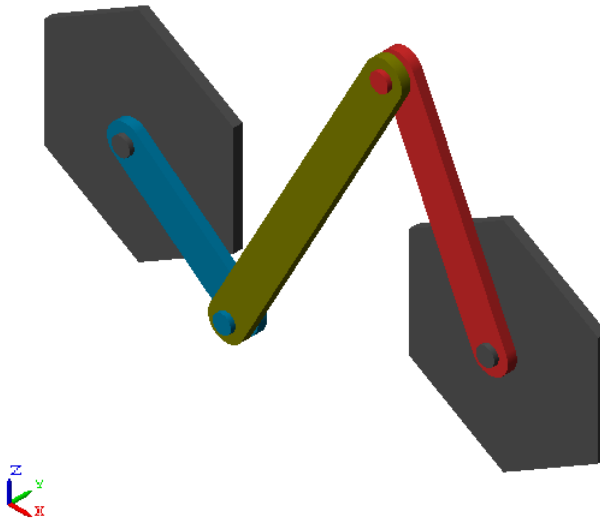
“Model Actuation Signal” on page 4-5

“Expose Motion Sensing Output Ports” on page 4-7

“Plot Motion Data” on page 4-8

“Simulate Model” on page 4-9

In this example, you add actuation input and sense motion outputs from model `sm_four_bar`. The following figure shows the four-bar model you actuate in this example.

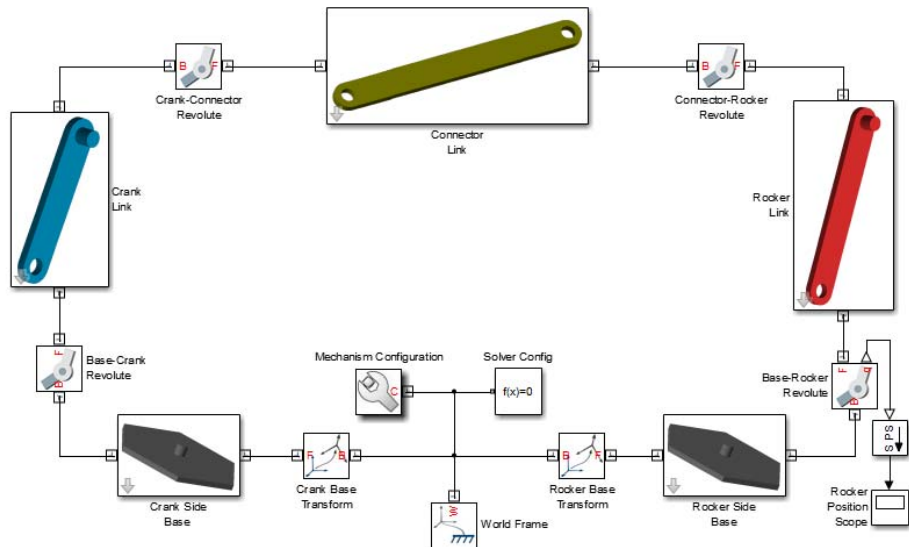


Open Model

Open the four-bar model. The model is present in your SimMechanics installation. To open the model:

- At the MATLAB command line, enter `sm_four_bar`.

The following figure shows the `sm_four_bar` model. In this example, you modify the model to add actuation input and sense motion outputs.



Expose Actuation Input Port

In the `sm_four_bar` model, locate joint block Base-Crank Revolute. Then, follow these steps to expose the actuation input port of the block:

- 1 Double-click the Base-Crank Revolute joint block.
- 2 In the dialog box, expand the **Actuation** menu.
- 3 In the **Mode** drop-down menu, select Torque.

4 Click **OK**.

Note The port associated with the actuation torque has label t .

Model Actuation Signal

The actuation input port of the joint block accepts a Simscape physical signal as input. In this example, you model the torque signal with a Simulink source block, and then use a Simulink-PS Converter block to convert the Simulink signal to a Simscape physical signal.

Connect Signal Blocks

Connect the blocks used to build the torque physical signal.

1 Add one Signal Builder block to the model.

Block	Library	Quantity	Description
Signal Builder	Simulink Sources	1	Build a piece-wise Simulink signal to model the time-dependence of input torque.

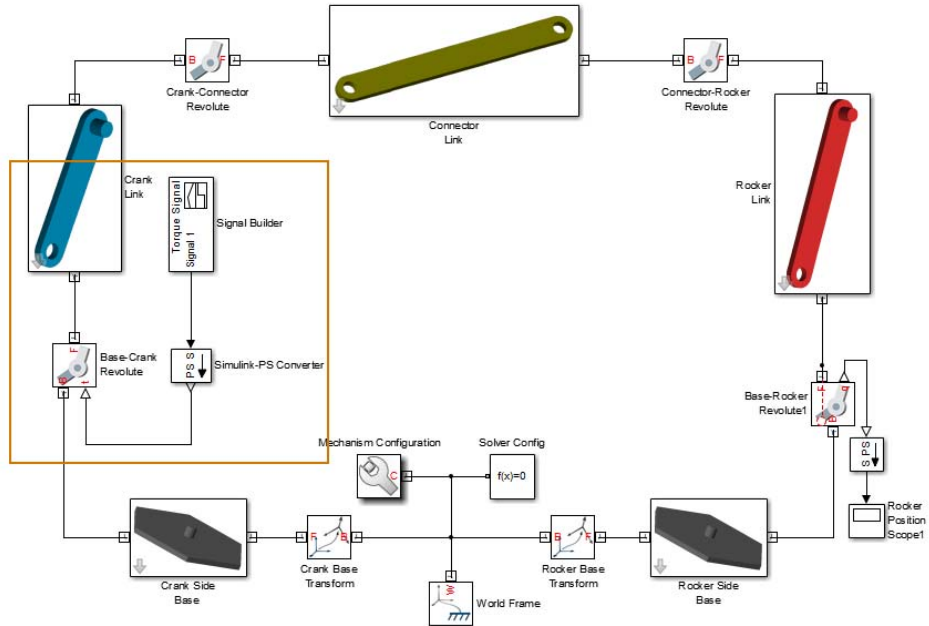
2 Add one Simulink-PS Converter block to the model.

Block	Library	Quantity	Description
Simulink-PS Converter	Simscape Utilities	1	Convert Simulink signal into Simscape physical signal

3 Connect the output port of the Signal Builder block to the input port of the Simulink-PS Converter block.

- 4 Connect the output port of the Simulink-PS Converter to the actuation input port of the Base-Crank Revolute joint block.

The modified `sm_four_bar` model should look similar to the following figure.

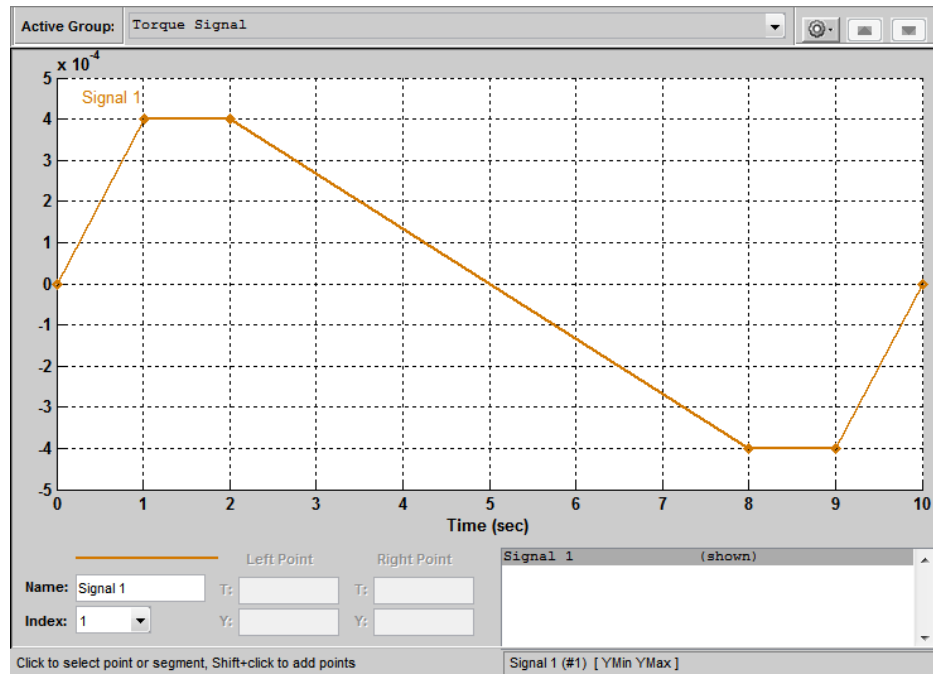


Build Torque Signal

Use the Signal Builder block to build the torque actuation signal:

- 1 Double-click the Signal Builder block.
- 2 In the Signal Builder window, build signal shown in the following figure.

Note See the block reference page for Signal Builder.



Specify Torque Physical Units

Use the Simulink-PS Converter block to specify the physical units of the torque signal.

- 1 Double-click the Simulink-PS Converter block.
- 2 In the **Input signal unit** field of the dialog box, enter $N*m$.
- 3 Click **OK**.

Expose Motion Sensing Output Ports

In the `sm_four_bar` model, locate joint block Base-Rocker Revolute. Then, follow these steps to expose the motion sensing output ports of the block:

- 1 Double-click the Base-Rocker Revolute joint block.
- 2 In the dialog box, expand the **Sensing** menu.

- 3** Check the boxes for **Position**, **Velocity**, and **Acceleration**.
- 4** Click **OK**.

Note The ports associated with the joint motion parameters have the following labels:

Motion Sensing Parameter	Port Label
Position	q
Velocity	w
Acceleration	b

Plot Motion Data

To plot the motion parameters of the Base-Rocker Revolute joint as a function of time, connect the motion sensing output ports to a Simulink Scope block.

The Scope block accepts only Simulink signals. You must convert the motion physical signals of the joint block to Simulink signals. To convert the signal, use the PS-Simulink Converter block.


- 1** Add one Scope block to the model.

Block	Library	Quantity	Description
Scope	Simulink Sinks	1	Plot dimensionless torque signal as a function of time.

- 2** Add three Simulink-PS Converter blocks to the model.

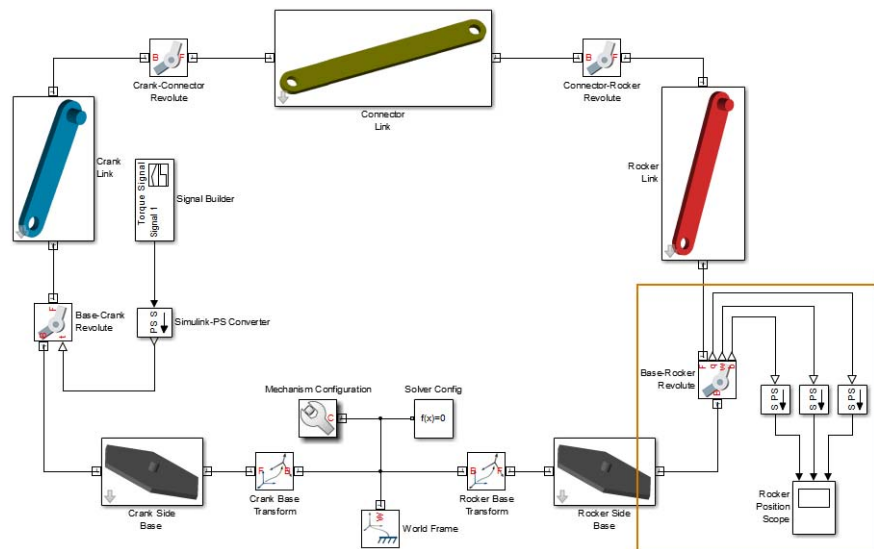
Block	Library	Quantity	Description
Simulink-PS Converter	Simscape Utilities	3	Convert Simscape physical signal to dimensionless Simulink signal

3 Double-click the Scope block.

4 In the dialog box, click the **Parameters** button .

5 In the **Number of Axes** field of the **General** tab, enter 3.

6 Connect the blocks as shown in the following figure.



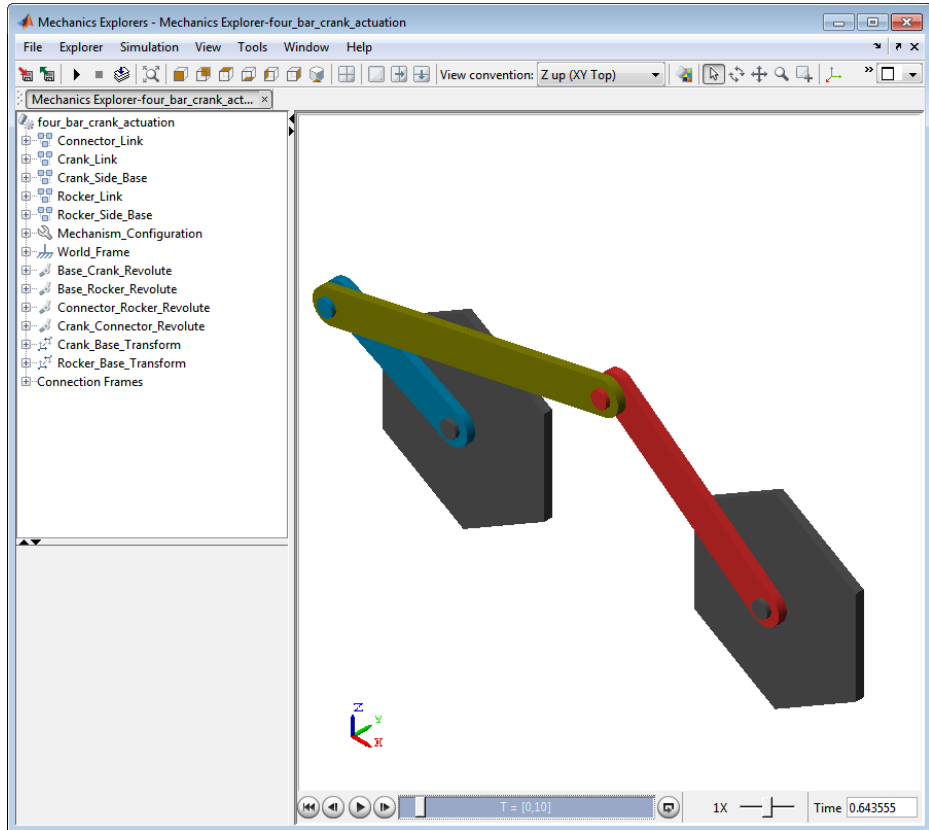
Simulate Model

The model is now ready for simulation.

1 In the model window, click the **Run** button

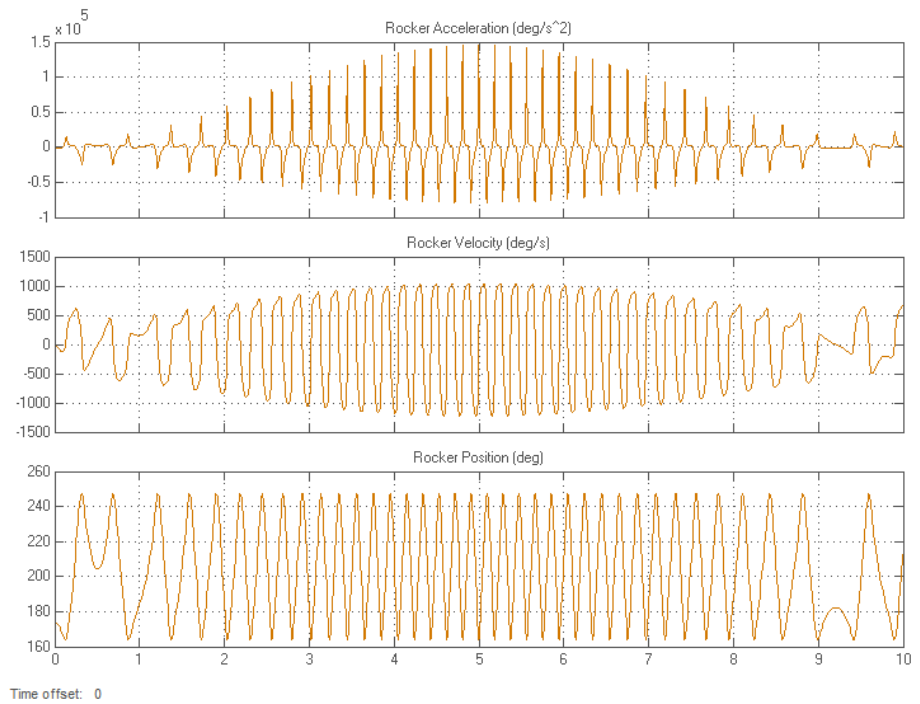


2 View simulation in Mechanics Explorer.



3 Double-click the Rocker Position Scope block.

4 Examine the Position, Velocity, and Acceleration plots of the Base-Rocker Revolute joint.



Add Internal Forces to Double-Pendulum Model

In this section...
“Summary” on page 4-12
“Requirements” on page 4-12
“Add Spring and Damper Forces” on page 4-12
“Add Motion Sensors” on page 4-13
“Simulate Damped Double-Pendulum Model” on page 4-15

In this example, you apply internal forces to the two joints of a double-pendulum model. Internal forces used in this example include both spring and damper forces. The spring force causes oscillation in the joint about an equilibrium position. The damper force causes a decay in the amplitude of joint oscillations. Before beginning, you must complete the example “Assemble Double-Pendulum Model” on page 3-17.

Summary

This example shows how to:

- Add spring and damper forces to joints
- Sense motion of joint frames

Requirements

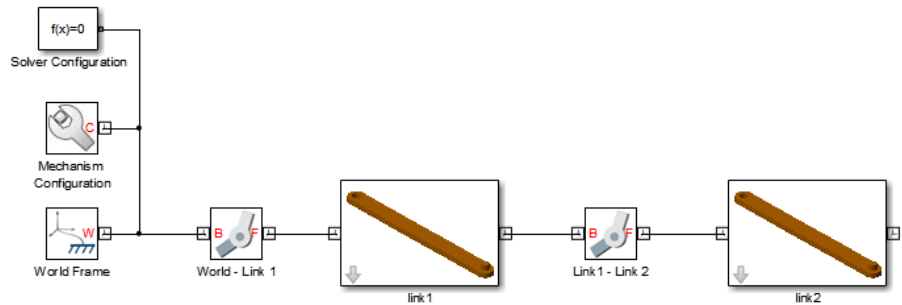
The example assumes the following:

- Completion of example “Model a Simple Binary Link” on page 2-31.
- “Assemble Double-Pendulum Model” on page 3-17
- Completion of example “Model a Compound Binary Link” on page 2-90.
- “Refine Double-Pendulum Model” on page 3-28

Add Spring and Damper Forces

Open the `double_pendulum_detailed` model you saved in example “Refine Double-Pendulum Model” on page 3-28. Then, add spring and damper forces

to each Revolute Joint block. The following figure shows the `double_pendulum` model used in this example.



- 1** In the model, double-click the two Revolute Joint blocks (**World-Link 1** and **Link 1 — Link 2**).
- 2** In the dialog box, expand **Internal Mechanics**.
- 3** In the **Spring Stiffness** and **Damping Coefficients** fields, enter the following values

Parameter	Numerical Value
Spring Stiffness	$5e-4 \text{ N*m/Deg}$
Damping Coefficient	$5e-5 \text{ N*m/ (Deg/s)}$

- 4** Leave the dialog boxes open for the next step.

Add Motion Sensors

Use motion sensors to analyze joint position and velocity as a function of time. To add motion sensors, use these steps:

- 1** In each Revolute Joint dialog box, expand the **Sensing** node.
- 2** Click the **Position** and **Velocity** check boxes.
- 3** Click **OK**.


The Revolute Joint blocks expose the following physical signal ports:

Port	Description
q	Outputs angular position of follower vs. base frame
w	Outputs angular velocity of follower vs. base frame

4 Add the following blocks to the model.

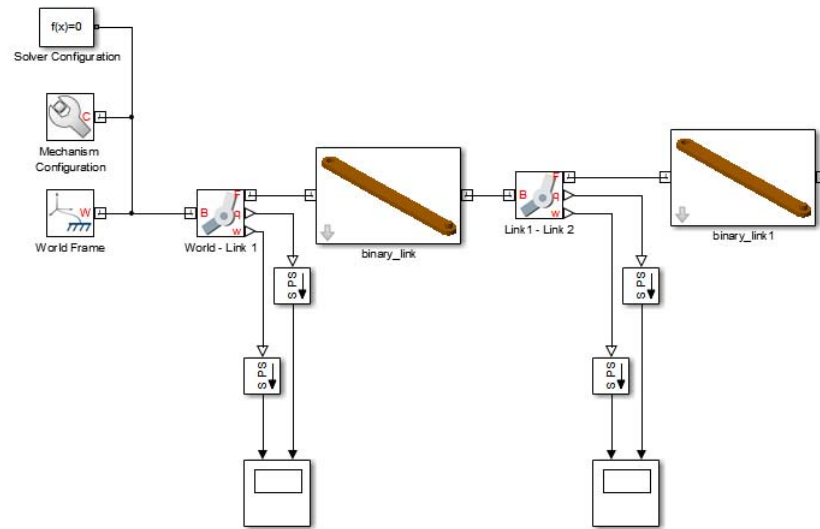
Block	Library	Quantity
PS-Simulink Converter	Simscape Utilities	4
Scope	Simulink Sinks	2

5 Double-click each Scope block.

6 In the dialog box, click the **Parameters** icon .

7 In **Number of Axes**, enter 2 and click **OK**.

8 Connect the blocks as shown in the following figure.

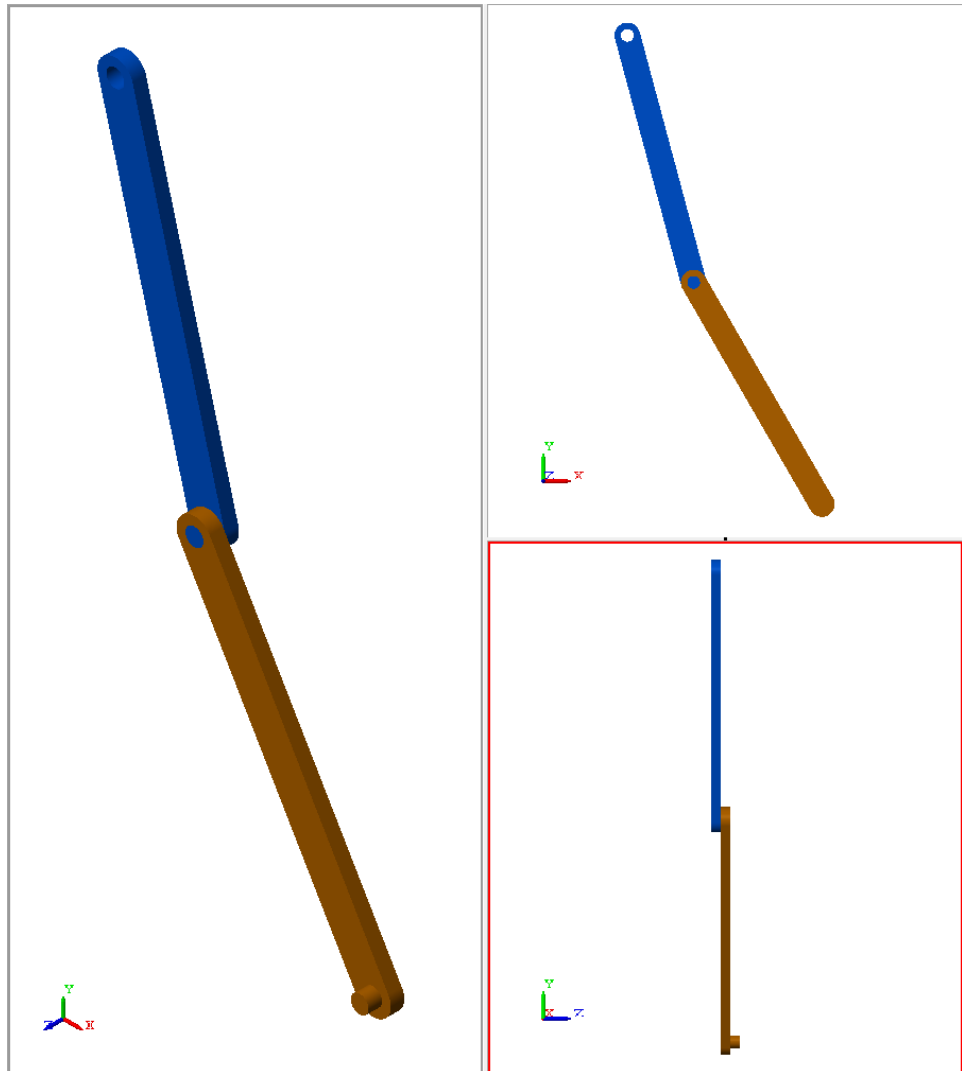


Simulate Damped Double-Pendulum Model

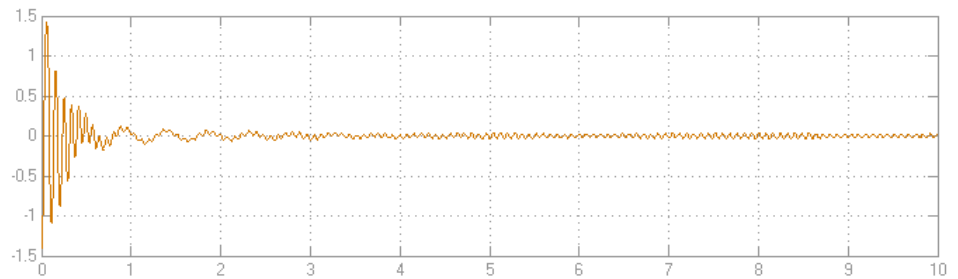
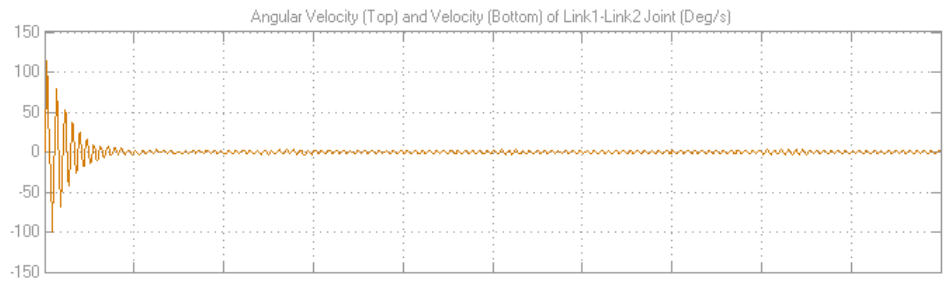
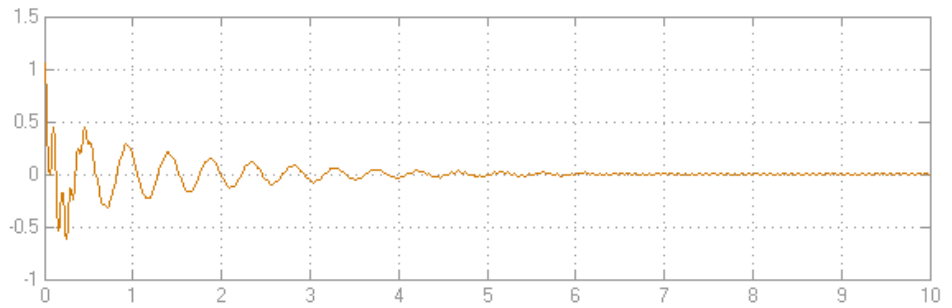
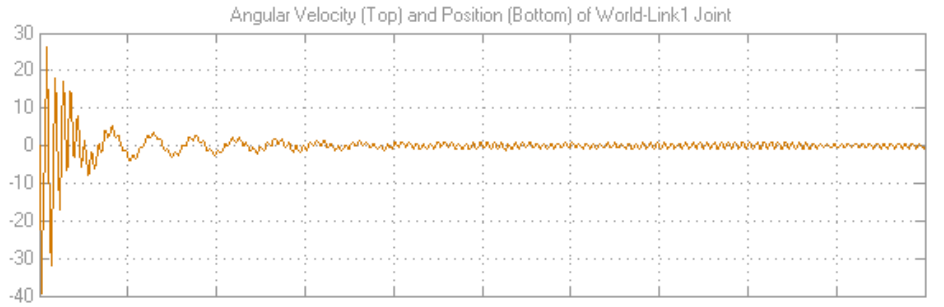
You can now simulate the damped double-pendulum model.

- 1 In the model, double-click each Scope block.
- 2 In the Simulink Editor menu bar, select **Simulation > Run**.
- 3 In **View Convention** of the Mechanics Explorer toolbar, select Y up (XY Front).

Mechanics Explorer displays the double-pendulum with the Y axis directed upwards.



The scope blocks provide position and velocity charts for each joint. The following two figures show the angular position and velocity for joints **World-Link1** and **Link1-Link2**, respectively.



Actuation

In this section...
“Forces & Torques Blocks” on page 4-18
“Joints Blocks” on page 4-24

You can actuate a SimMechanics model with forces and torques. Two types of blocks provide actuation capability:

- Forces & Torques — Apply force or torque to a rigid body frame.
- Joints — Apply force or torque between base and follower joint frames.

Note Motion actuation is not available in SimMechanics models. You add a force or torque to a frame, but you cannot specify the motion state of the frame.

Forces & Torques Blocks

The Forces & Torques library contains blocks to represent different types of forces. Use the blocks to actuate a rigid body or to apply an internal force between two rigid bodies in a multibody system.

The following Forces & Torques blocks are available.

Block	Description	Specification Mode
External Force and Torque	Generic force that originates outside the system	Connect Physical Signal that specifies the magnitude of the force
Inverse Square Law Force	Force proportional to the reciprocal of the distance between two frames	Enter force constants in the block dialog box

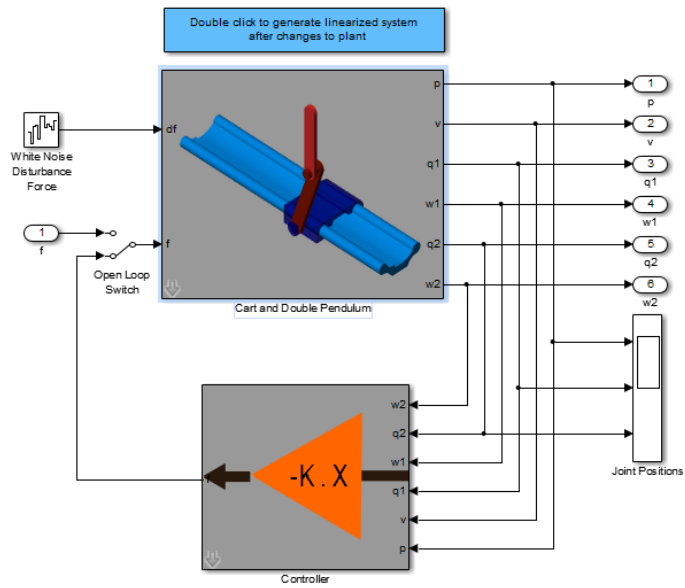
Block	Description	Specification Mode
Spring and Damper Force	Force proportional to the distance and relative velocity of two frames	

External Force and Torque

The block accepts a physical signal as input. You can model a generic force equation with Simulink blocks, convert the resultant Simulink signal to a Simscape physical signal with the Simulink-PS Converter block, and connect the physical signal to the External Force and Torque block.

Model `sm_cart_double_pendulum` provides an example of the External Force and Torque block. In the model:

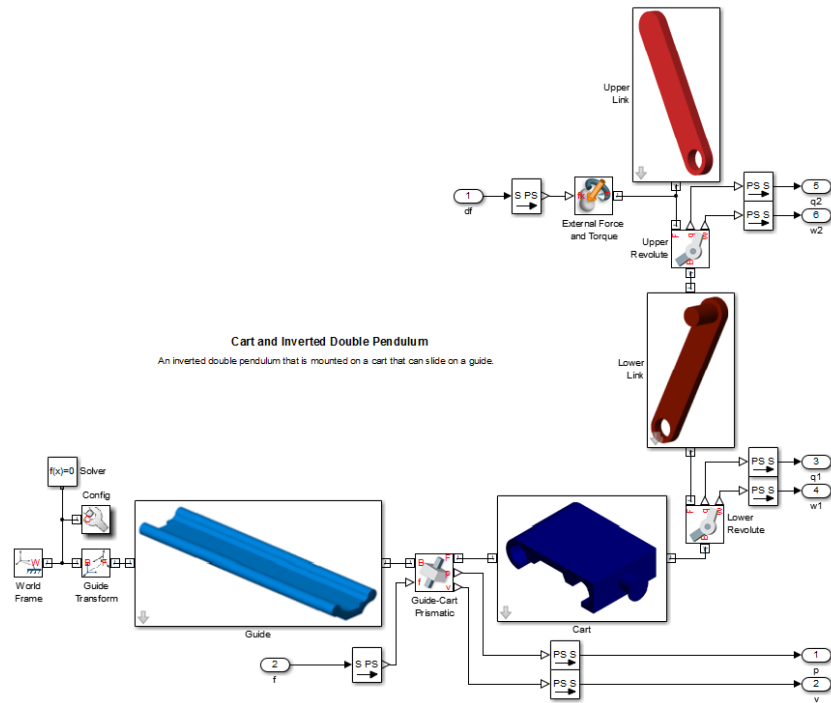
- Simulink block Band-Limited White Noise generates a random signal.



Inverted Double Pendulum on a Sliding Cart

An inverted double pendulum that is mounted on a cart that can slide on a guide. The inverted pendulum is controlled by a controller. Make any changes to the system and click on the blue box above to generate linearized model for the system before running the simulation. The control gains are computed using the linearized model. The pole placement technique is used to compute the control gains from the linearized model. The controller keeps the double pendulum vertical in the presence of a random disturbance force.

- Simscape block Simulink-PS Converter transforms the Simulink signal into a Simscape physical signal.



- A physical signal line routes the random physical signal to the input port of the External Force and Torque block. The force block generates a random disturbance force at the revolute joint between the two pendulum links.

Inverse Square Law Force

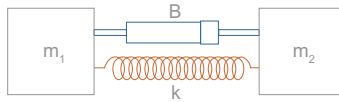
The block represents a force that decays with the square of the distance between two frames. The force acts along the axis that connects the two frame origins. The magnitude of the force satisfies the algebraic expression $F=C/r^2$. Parameter C is a force constant which you enter in the dialog box of the block. Parameter r is the distance between the two rigid body frames.

Examples of the Inverse Square Law Force include the gravitational and Coulomb forces, listed in the following table.

Force	Equation	Constant	Parameters
Gravity	$F = G \frac{m_1 m_2}{r^2}$	$G m_1 m_2$	<ul style="list-style-type: none"> • G — Universal gravitational constant • m_1 — Mass of rigid body 1 • m_2 — Mass of rigid body 2
Coulomb	$F = k_e \frac{q_1 q_2}{r^2}$	$k_e q_1 q_2$	<ul style="list-style-type: none"> • k_e — Coulomb constant • q_1 — Electric charge of rigid body 1 • q_2 — Electric charge of rigid body 2

Spring and Damper Force

The block models spring and damper forces that act reciprocally between two rigid body frames in a parallel circuit. The following figure shows a schematic of the force model. A spring with stiffness constant k and a damper with damping coefficient B represent the spring and damper forces between masses m_1 and m_2 .



The spring and damper force is a linear force that satisfies the mathematical expression:

$$F = -k (r - r_0) - Bv$$

In the expression, the term $-k(r - r_0)$ represents the spring force between the base and follower frames of the block. Parameters k and r_0 are the spring stiffness constant and the equilibrium distance between the base and follower frames, respectively, which you specify in the dialog box of the block. Parameter r is the actual distance between the base and follower frames.

The term $-Bv$ represents the damper force between the two rigid body frames. Parameter B is the damping coefficient, which you provide in the dialog box of the block. Parameter v is the relative velocity between the base and follower frames of the block.

When a disturbance displaces one frame relative to another from an equilibrium position, the spring force acts to return the frame to the equilibrium position. The magnitude of the spring force is directly proportional to the displacement of the two frames from the equilibrium position.

The damping force resists the relative motion between two frames as they attempt to return to the equilibrium position. The motion of the two frames depends on the damping ratio $\zeta = B/B_c$, where B_c is the critical damping coefficient. The damping ratio is a dimensionless parameters that determines the frequency response of a linear harmonic oscillator, where:

$$B_c = 2\sqrt{km}$$

The following table summarizes the damping states that are possible in a harmonic oscillator:

Damping Condition		Motion Behavior
Undamped	$\zeta=0$	Frames oscillate indefinitely about equilibrium position at natural resonant frequency ω_0
Underdamped	$0<\zeta<1$	Frames oscillate for a finite period of time about the equilibrium position

Damping Condition		Motion Behavior
		as the oscillation amplitude decays to zero
Critically Damped	$\zeta=1$	Frames return to equilibrium position in minimal time interval
Overdamped	$\zeta>1$	Frames return to equilibrium position in time intervals that increase in direct proportion to the value of the damping ratio ζ

Joins Blocks

Blocks in the Joints library contain actuation options. In the block dialog box, you can select the actuation mode for each joint primitive. Actuation modes include Force for prismatic primitives, and Torque for revolute and spherical primitives.

Joint actuation forces and torques act between the base and follower frames of a joint block. To apply a force or torque between two arbitrary frames, use a Forces & Torques block.

You specify the magnitude of the actuation force or torque with physical signals. Selecting Force or Torque for a joint primitive exposes a physical signal port for that primitive. Connect a physical signal to the port to specify the actuation signal.

Force signals act to generate rectilinear motion between base and follower frames. The physical signal for the force specifies the force magnitude. The joint primitive specifies the force direction. For example, force actuation of the Z Prismatic Primitive acts along the aligned Z axes of the base frame and follower frames. The port associated with the force signal accepts a single scalar number.

Torque signals act to generate rotational motion between base and follower frames. The physical signal for the torque specifies the torque magnitude. The joint primitive specifies the torque direction. For example, torque actuation of the X Revolute Primitive acts about the aligned Z axes of the base and follower frames. The port associated with the torque signal accepts a single scalar number for revolute primitives, with the option of a 3-vector containing all torque components for spherical primitives.

The following table summarizes the actuation modes available in Joint blocks.

Actuation Mode	Description	Specification Mode
Force	Actuate linear motion along aligned primitive axes of base and follower frames	Physical signal port Physical signal port
Torque	Actuate rotational motion about aligned primitive axes of base and follower frames	

Force and Motion Sensing

In this section...
“Motion Sensing” on page 4-26
“Force Sensing” on page 4-27

SimMechanics provides blocks you can use to sense motion and actuation parameters. Motion sensing enables the use of control system algorithms in a model. You can acquire motion data from one or more frames, feed the motion data through a control algorithm, and generate an actuation signal to control the motion of the frames. Use the force sensing capability to analyze the magnitude of a force applied between two frames.

Motion Sensing

Blocks with motion sensing capability measure and record the relative position, velocity, and acceleration between two frames. Motion parameters can be rotational or translational. Rotational motion parameters include the angular position, velocity and acceleration about one or more axes. Translational motion parameters include the linear position, velocity and acceleration along one or more axes.

Blocks in the Joints library contain motion sensing capability between the base and follower frames of the joint. To measure motion parameters between two arbitrary frames, use the Transform Sensor block in the Frames & Transforms library.

Transform Sensor Block

The Transform Sensor block is the only dedicated motion sensor block in the SimMechanics libraries. The block provides the option to measure multiple motion parameters between any two frames. The following list outlines the rotational and translational parameters you can measure with the Transform Sensor block:

Rotational Parameters

- Angle (**P**osition only)

- Axis (**Position** only)
- X, Y, Z Cartesian components of rotational motion (**Velocity** and **Acceleration** only)
- Quaternion
- Transform

Translational Parameters

- X, Y, Z Cartesian components of translational motion
- Radius
- Azimuth
- Distance
- Inclination

Joint Blocks

Joint blocks offer motion sensing between base and follower frames. In the dialog box of a joint block, you can select to measure position, velocity, and acceleration parameters. Each joint primitive provides the option to measure the three parameters about the joint primitive axis. With the spherical joint primitive, you can select to resolve the three parameters about the three Cartesian axes (X,Y,Z) in either the base or follower frame.

Force Sensing

The following blocks provide force-sensing capability:

- Inverse Square Law Force
- Spring and Damper Force

If you select to sense force, the block displays a physical signal port that outputs the force data. You can display the force measurements graphically with Simulink scope blocks.

Simulation and Analysis

Simulation

- “Configure Model for Simulation” on page 5-2
- “Update and Simulate Model in Mechanics Explorer” on page 5-4
- “Simulation Limitations” on page 5-7
- “Simulation Errors” on page 5-8

Configure Model for Simulation

During simulation, SimMechanics employs a Simulink global solver to determine the configuration of a model as a function of time. You can select the best solver for your application from a list of solvers that Simulink provides. Simulation parameters include the numerical step used to progress through the simulation and the solver tolerance values. Adjust the parameters to optimize speed and accuracy of the simulation.

For solver selection and parameter specification, see:

- “Choose a Solver” in the Simulink documentation.
- “Setting Up Solvers for Physical Models” in the Simscape documentation.

Specify Solver Settings

To select a global solver for your model:

- 1** On the Simulink menu bar, click **Simulation > Model Configuration Parameters**.
- 2** On the Tree View pane, select **Solver**.
- 3** In **Solver Options**, click **Type** and select **Variable-step** or **Fixed-step**.

Note For best performance, select **Variable-step**. For model deployment, select **Fixed-step**.

- 4** Click **Solver** and select the appropriate solver for your application. The default solver is ODE45 (Dormand-Prince).

To modify the global solver parameters for your model:

- 1** In the **Solver options** pane of the **Model Configuration Parameters** window, enter the desired values for step size and tolerance parameters.

Reducing the values of the step size and tolerance parameters enhances simulation accuracy, but decreases simulation speed. Adjust the parameters to obtain an optimal trade-off between simulation speed and accuracy.

Update and Simulate Model in Mechanics Explorer

In this section...
“Update Model” on page 5-4
“Simulate Model” on page 5-5

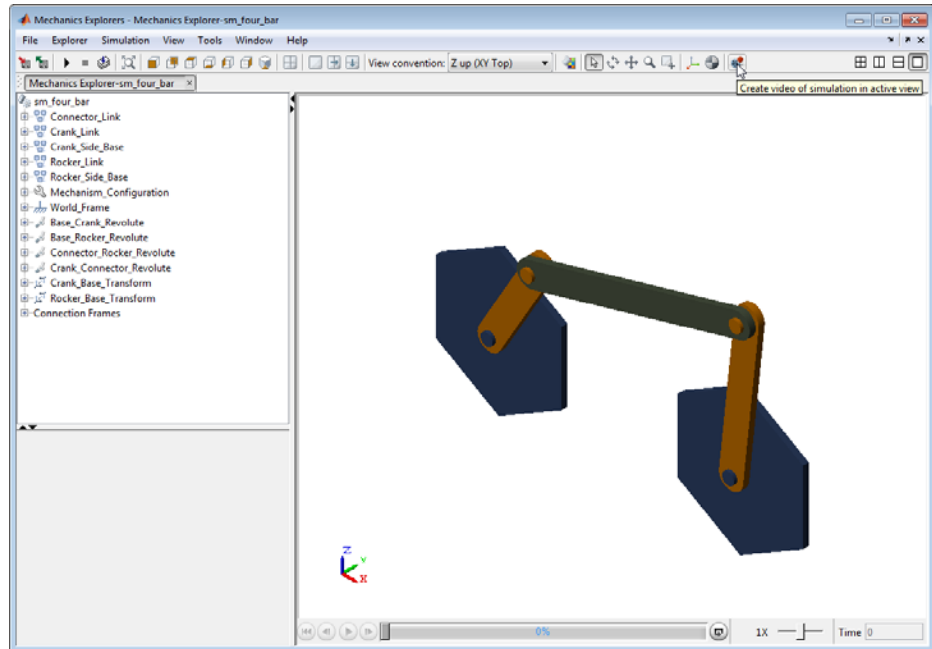
Each time you update a model in the Simulink Editor window, Mechanics Explorer updates the model display. Simulating a model causes Mechanics Explorer to display a dynamic animation of the model. You can record and save simulation videos to share or for future reference.

Update Model

The procedure to update a SimMechanics model in Mechanics Explorer is the same as Simulink models:

- 1** In the Simulink Editor window that contains your model, select **Simulation > Update Diagram**.
- 2** Mechanics Explorer opens with a 3-D display of the updated model.

Note By default, Mechanics Explorer opens when you first update a model. Subsequent updates to a model change the display of the open Mechanics Explorer window.



After you update a model, you can change view point, toggle the visibility of frames and centers of mass of compound rigid bodies, and rotate, pan, or zoom the display in Mechanics Explorer. See “Rotate, Pan, and Zoom View” on page 6-14.

You can change the default display settings. See “Configure Mechanics Explorer Display” on page 6-2.

Simulate Model

The procedure to simulate a SimMechanics model is the same as Simulink models.

- In the Simulink Editor window for your model, select **Simulation > Run**.

Note You can use the shortcut **Ctrl+T** to simulate a model.

When you simulate a model, Mechanics Explorer displays a physics-based animation of the model in 3-D. Record the simulation to share or for future reference. See “Record and Play Simulation Video” on page 6-18.

Simulation Limitations

In this section...
“Limited Visualization with For Each Subsystems” on page 5-7
“No Visualization with Referenced Models” on page 5-7
“No Support for Simscape Local Solvers” on page 5-7

Limited Visualization with For Each Subsystems

Models with one or more For Each Subsystem blocks simulate with limited visualization. The Mechanics Explorer visualization utility displays the model in only one of the instances which the For Each Subsystem block provides. The visualization limitation does not affect model simulation — SimMechanics simulates the model for all instances of the block.

No Visualization with Referenced Models

Models with Model blocks (known as referenced models) simulate with no visualization. During model simulation, SimMechanics issues a warning at the MATLAB command line. The Mechanics Explorer visualization utility does not open.

No Support for Simscape Local Solvers

SimMechanics software does not support Simscape local solvers. If you select a local solver in the Simscape Solver Configuration block, the solver does not apply to the SimMechanics portion of a model. SimMechanics blocks continue to use the Simulink global solver that you select in **Model Configuration Parameters** for your model.

Note SimMechanics requires the Simulink global solver to be *continuous*. If the global solver is discrete, SimMechanics issues an error and the model does not simulate. This requirement applies to both fixed- and variable-step solvers.

Simulation Errors


In this section...
“Frame Position Violation” on page 5-8
“Data Validation Error” on page 5-9
“Degenerate Mass Distribution” on page 5-10
“Gimbal Lock” on page 5-11
“No Solver Block” on page 5-11

SimMechanics models must satisfy certain simulation requirements. If requirements are not met, you may be able to update, but not simulate, the model. The following issues highlight the source of common simulation errors in SimMechanics.

Frame Position Violation

During model assembly and simulation, SimMechanics uses the kinematic constraints of the model to determine the correct position and orientation of each frame. Model elements that impose kinematic constraints include joints, constraints, and rigid connections between frames. Under certain conditions, the combination of constraints in a model create linear and angular position conflicts for one or more frames that make assembly impossible. In the event of a frame position conflict, SimMechanics issues an error message that identifies the type of error. The model does not simulate.

To correct position violations:

- 1 Update the model. SimMechanics issues a position violation error.
- 2 In the Mechanics Explorer menu bar, select **Tools > Model Report**.
- 3 In the Model Report window, identify the name of each unassembled joint. Joints marked with the  icon are unassembled.
- 4 Determine if either base or follower frames of the unassembled joint require adjustment.

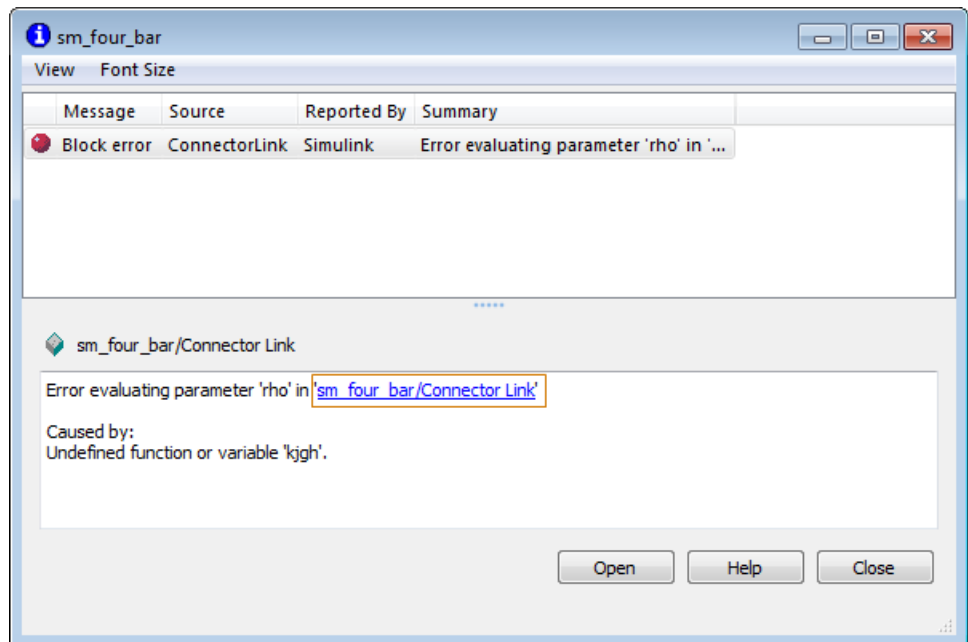
- 5 Once you have identified which frame requires adjustment, adjust the frame with a Rigid Transform block. Modify the parameters of an existing Rigid Transform block, or add a new block to specify the correct parameters.

Data Validation Error

During model simulation, SimMechanics relies on the block parameters to determine the correct state of each model frame as a function of time. Parameters that accept numerical input require a valid numerical expression or variable name. In the event that SimMechanics encounters an invalid block parameter, a dialog box opens identifying the block and the incorrect parameter.

To remove the simulation error:

- 1 In the dialog box that contains the error message, click the name of the block provided.



- 2** At the block level, verify that the incorrect block parameter contains either a valid numerical expression or a MATLAB variable that has been defined in either:
 - Subsystem mask, if one exists
 - Model workspace
 - MATLAB workspace

Degenerate Mass Distribution

Blocks that provide force and torque input require that each port frame connect directly to a non-degenerate mass distribution. A mass distribution is degenerate if it contains:

- Zero mass
- Zero inertia tensor (e.g. point mass)

Affected blocks include **Joint** and **Forces and Torques**. Different joint primitives impose different requirements on the mass distribution:

- Prismatic primitive — Requires non-zero mass.
- Revolute primitive — Requires non-zero moment of inertia about a single axis of revolution.
- Spherical primitive — Requires non-zero moments of inertia about all axes of revolution.

To correct a degenerate mass issue:

- 1** Run the simulation.
- 2** In the **Simulation Diagnostics** window, click the name of the block with the degenerate mass error.
- 3** In the model, verify that each port frame connects to a block or subsystem that contains a non-degenerate mass distribution.

The following figure shows a simple model with a degenerate mass error. The follower frame port of block **Revolute Joint** connects directly to the

base frame port of block **Revolute Joint1**, which contains zero mass and inertia tensor.

Gimbal Lock

Some SimMechanics joints are vulnerable to *Gimbal Lock*. Gimbal lock is a mechanical phenomenon where two revolute axes become aligned, so that one rotational degree of freedom is lost. The occurrence of gimbal lock can cause numerical singularities that force the model simulation to stop. In the event that a simulation error occurs due to gimbal lock, a dialog box opens identifying the type and source of the error.

Joints vulnerable to gimbal lock include:

- Gimbal Joint
- Bearing Joint
- Bushing Joint

To avoid gimbal lock, SimMechanics provides equivalent blocks that contain one Spherical primitive instead of three Revolute primitives. The Spherical joint primitive is not vulnerable to gimbal lock at any time. The following table lists block replacements for Gimbal, Telescoping, and Bushing joint blocks.

Vulnerable to Gimbal Lock	Not vulnerable to Gimbal Lock
Gimbal joint	Spherical Joint
Bearing Joint	Telescoping joint
Bushing Joint	6-DOF Joint

No Solver Block

Each frame network requires a Solver Configuration block. The block provides solver parameters required for the simulation of SimMechanics models. If a model contains a frame network without a Solver Configuration block, SimMechanics issues an error and the model does not simulate.

To correct the error, connect exactly one Solver Configuration block to each distinct frame network.

Visualization and Animation

- “Configure Mechanics Explorer Display” on page 6-2
- “Visualization with Mechanics Explorer” on page 6-11
- “Rotate, Pan, and Zoom View” on page 6-14
- “Record and Play Simulation Video” on page 6-18
- “Visualization Troubleshooting” on page 6-24

Configure Mechanics Explorer Display

In this section...
“Change Background Color” on page 6-2
“Change View Point” on page 6-5
“Change View Convention” on page 6-7
“Display Multiple Screens” on page 6-8
“Toggle Visibility of Frames and Mass Centers” on page 6-10

You can customize the display of Mechanics Explorer. Settings you can change include:

- Background color
- View point
- View convention
- Number of display windows for a model
- Visibility of frames and centers of mass

Change Background Color

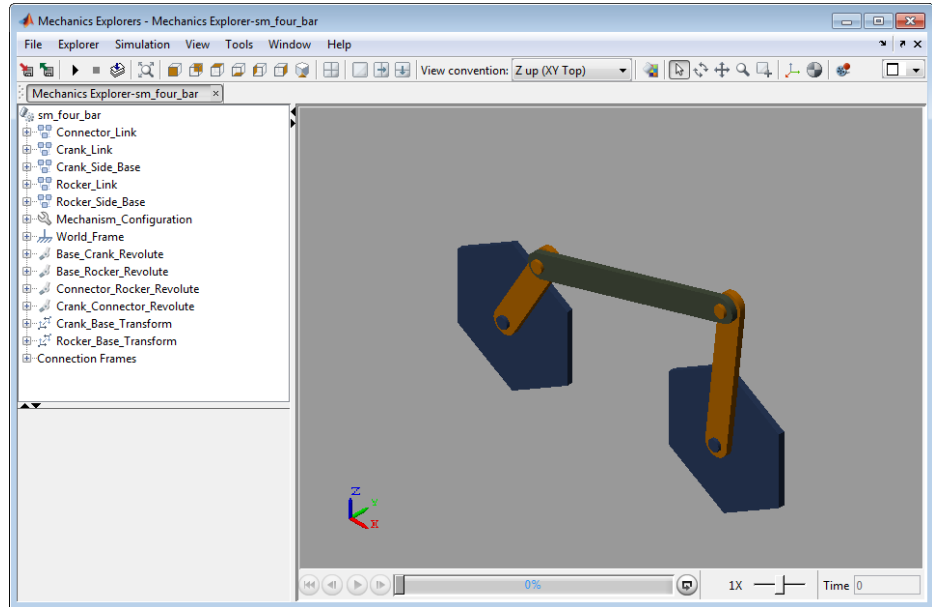
To change the background color, use the following procedure. The procedure uses the `sm_four_bar` as an example.


- 1 At the MATLAB command line, enter `sm_four_bar`.

Note Alternatively, open a SimMechanics model of your choice.

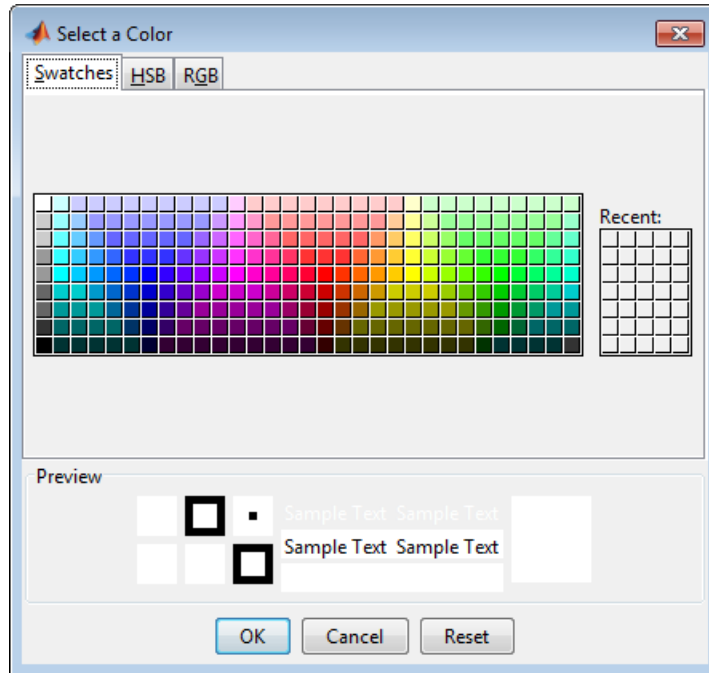
- 2 In the Simulink Editor window for the model, select **Simulation > Update Diagram**.

Note Mechanics Explorer opens with a display of your model against the default grey background.




3 In the Mechanics Explorer toolbar, click the  icon.

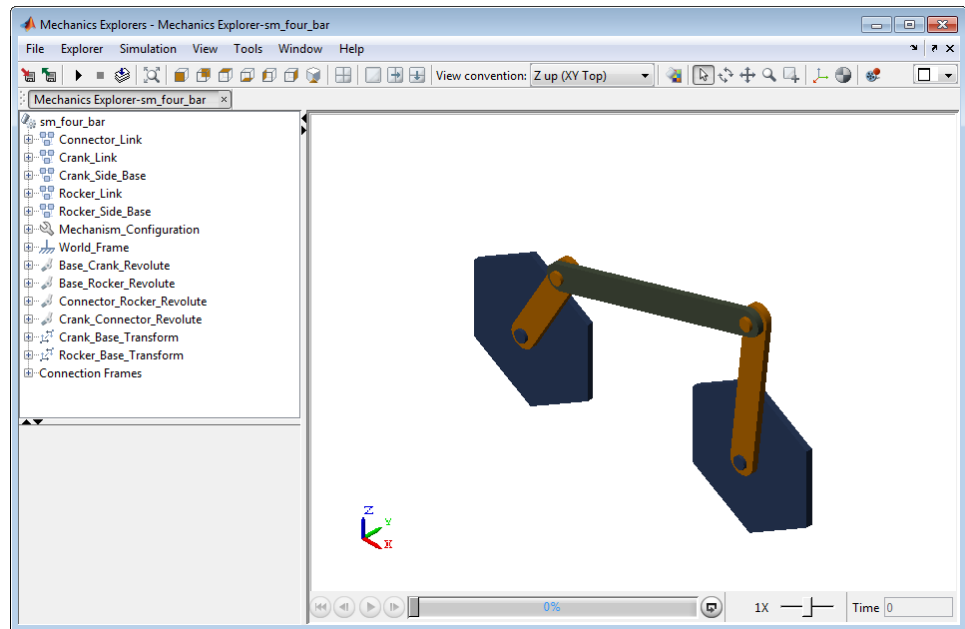
4 In the **Select a Color** dialog box, click a color or select the **HSB** or **RGB** tabs to specify color HSB or RGB values, respectively.




5 Click **OK**.

6 In the Mechanics Explorer toolbar, click the  icon.



Clicking the  icon saves the current Mechanics Explorer configuration to the SimMechanics model. If you close the Mechanics Explorer window and update the model, Mechanics Explorer opens with the new configuration.








Change View Point

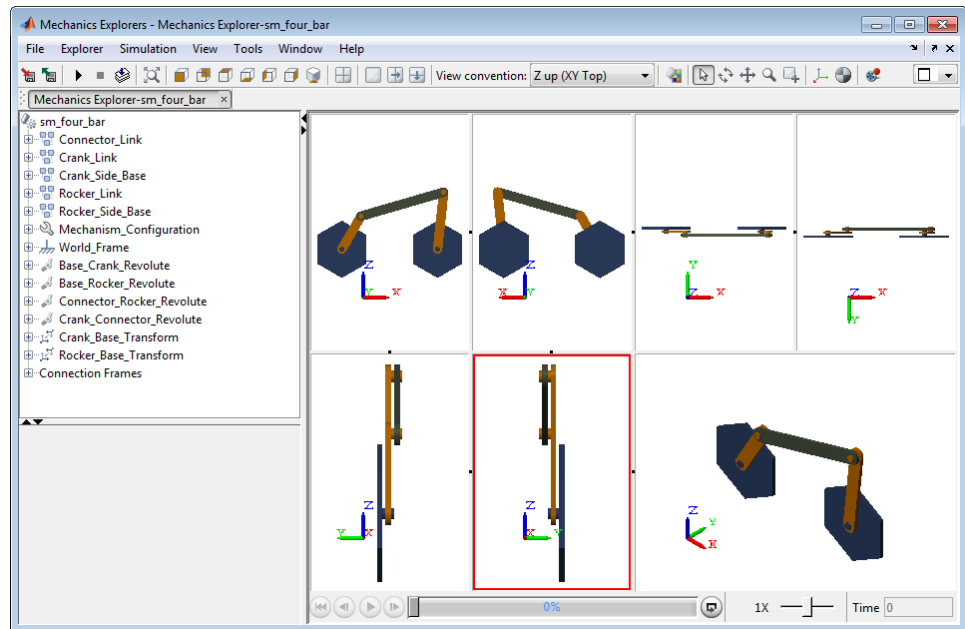
Mechanics Explorer provides seven view presets that you can use to change the perspective of a model. Each preset has an icon in the Mechanics Explorer toolbar .

Click an icon to select the corresponding view preset. The following table describes the seven presets in Mechanics Explorer.

View Icon	View Name	View Description
	Front view	Display model ZX plane with Y axis pointing into screen
	Back view	Display model ZX plane with Y axis pointing out of screen

View Icon	View Name	View Description
	Top view	Display model XY plane with Z axis pointing out of screen.
	Bottom view	Display model XY plane with Z axis pointing into screen
	Left view	Display model YZ plane with X axis pointing into screen
	Right view	Display model YZ plane with X axis pointing out of screen
	Isometric view	Display model in 3-D with axes X, Y, and Z at 120° to each other.

The following figure shows the seven view presets in Mechanics Explorer. The top row shows the following four presets ordered left to right: front, bottom, top, bottom. The bottom row shows the following three presets ordered left to right: left, right, isometric.

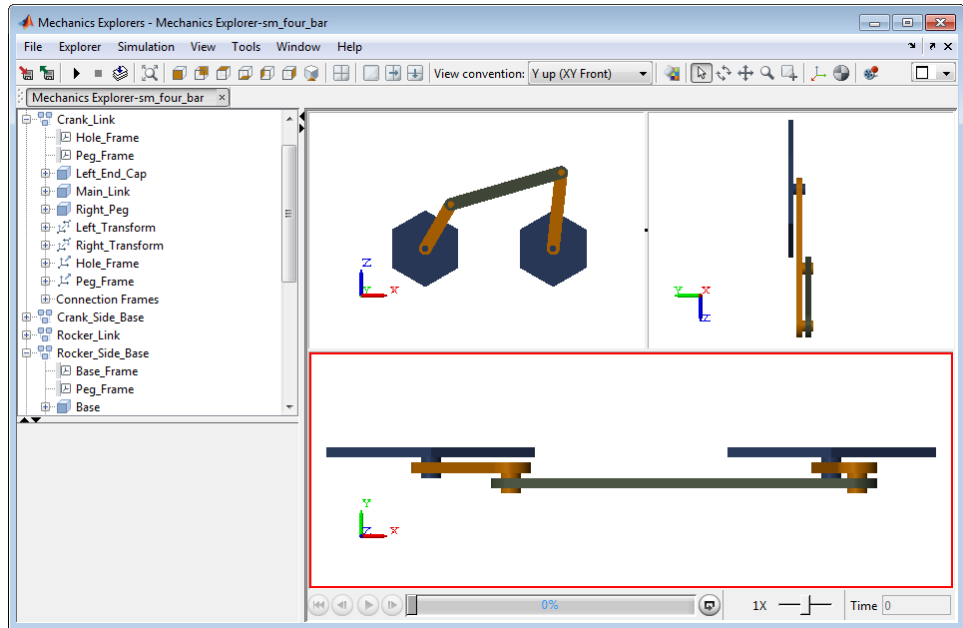


Change View Convention


You can choose from three view conventions:

- Z axis up—displays the model ZX plane in front view
- Z axis down—displays the model YZ plane in front view
- Y axis up—displays the model XY plane in front view



To select a view convention, click the **View convention** drop-down menu, and select one of the three view conventions. The following figure shows a four-bar model in front view using the three view conventions. The top row shows view conventions Z up and Z down ordered left to right. The bottom row shows view convention Y up.





Display Multiple Screens

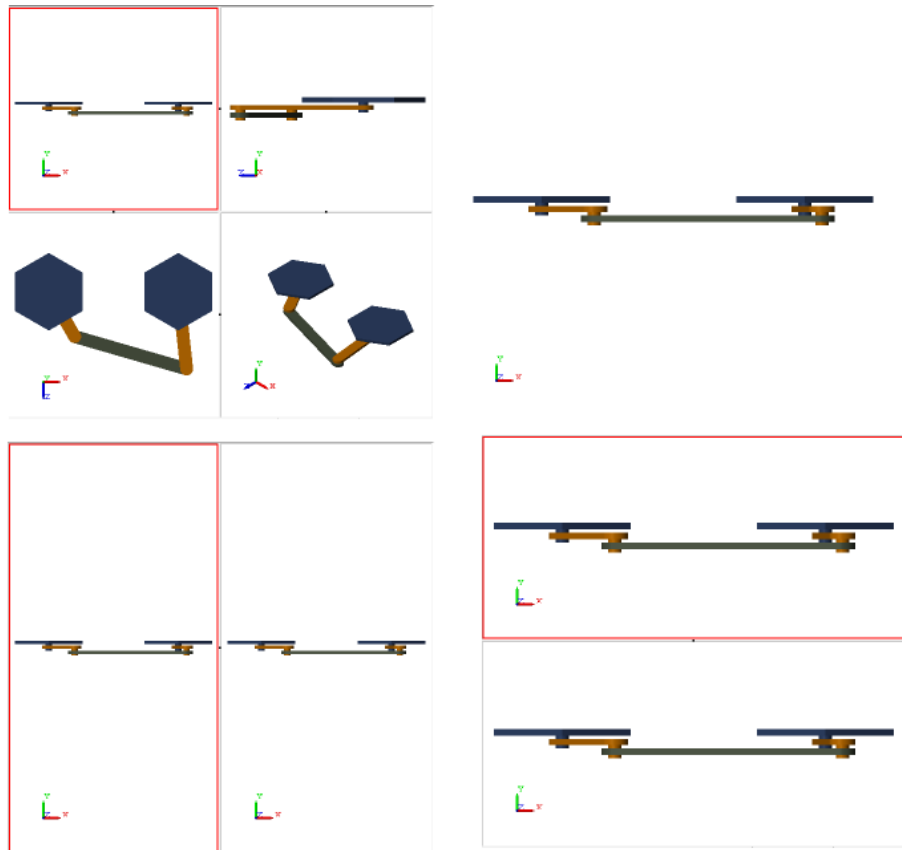
You can divide the Mechanics Explorer screen into multiple screens, each with an independent view of a model. The Mechanics Explorer toolbar provides icons  to split the active window into two windows vertically or horizontally.

Each time you split the active window, you generate two smaller, equally sized windows. You can split the active window an arbitrary number of times to generate as many view screens as you need. The following table describes the screen split icons.



Icon	Icon Description
	Split the active screen into four standard views
	Display a single screen


Icon	Icon Description
	Split the active screen vertically into two equally sized screens
	Split the active screen horizontally into two equally sized screens


The following image shows Mechanics Explorer with four standard views, in single screen mode, with two vertically split screens, and with two horizontally split screens.



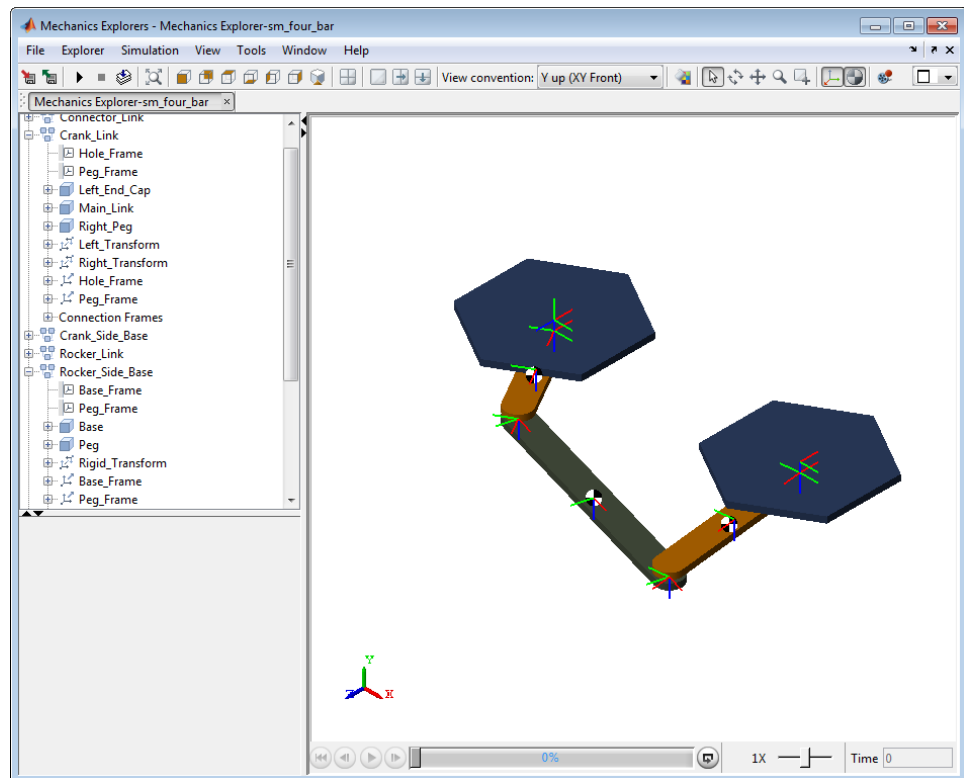
Toggle Visibility of Frames and Mass Centers

The Mechanics Explorer provides icons   so you can display and hide frames and center-of-mass markers.

To toggle frame visibility, click the  icon.

To toggle the visibility of center-of-mass markers, click the  icon.

The following figure shows a four-bar model that displays frames and center-of-mass markers.



Visualization with Mechanics Explorer

In this section...
“Mechanics Explorer Window” on page 6-11
“Model Report” on page 6-13
“Simulation Video” on page 6-13

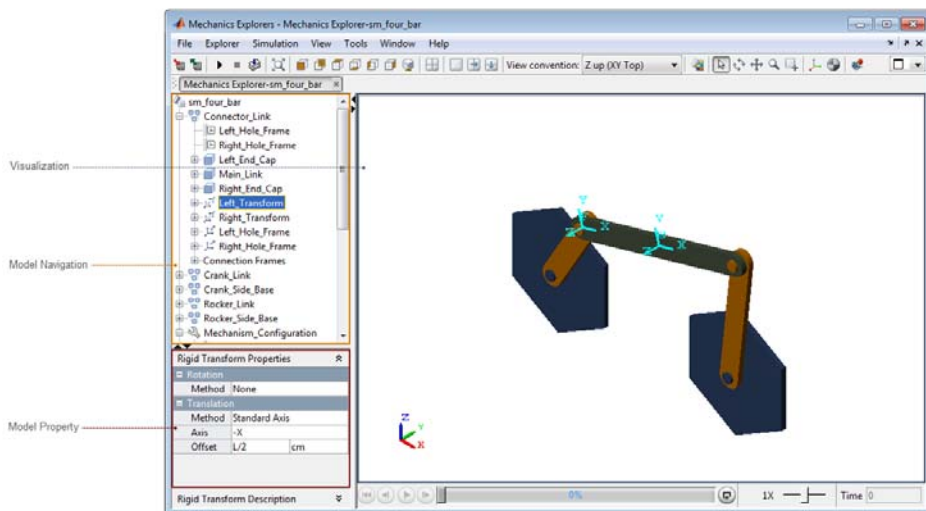
Mechanics Explorer is a utility that provides 3-D visualization, model navigation, and troubleshooting tools for SimMechanics models. By default, each time you update or simulate a SimMechanics model, Mechanics Explorer displays the updated model. Use Mechanics Explorer frequently throughout the modeling process to uncover errors in rigid body geometry and frames or in multibody assembly.

Mechanics Explorer Window

The Mechanics Explorer window contains three primary panes:

- Visualization—Display a 3-D graphic representation of a multibody model.
- Model Navigation—Navigate the model by subsystem, block or port.
- Model Property—Inspect block properties and port connections in a model.

The following figure shows the three panes of Mechanics Explorer.



Visualization

The visualization pane of Mechanics Explorer displays a 3-D view of a SimMechanics model. The 3-D view is static when you update a model (**Ctrl+D**), or dynamic when you simulate a model (**Ctrl+T**). You can choose from seven preset views: front, back, top, back, left, right, and isometric. You can rotate, pan, and zoom a model. See “Visualization with Mechanics Explorer” on page 6-11.

Model Navigation

Identify a subsystem, block, or port with the model navigation pane. When you click the name of a subsystem, block, or port in the model navigation pane, the visualization pane highlights the corresponding entity with a light blue color. Use the model navigation pane to highlight multibody subsystems, rigid body subsystems, and frames in the visualization pane.

Model Property

Each time you click the name of an entity in the model navigation pane, the model property pane displays the parameters and frames associated with the

selected entity. Use the model property pane to review the parameters and frames that belong to a subsystem, block, or port.

Model Report


Mechanics Explorer provides the Model Report tool to uncover model assembly problems. Model Report identifies the status of each joint and constraint in a model, and flags assembly errors. For joints with state targets, Model Report includes the actual and specified state targets. The report flags joints that have unmet state targets. The following image shows the Model Report window for model `sm_four_bar`.

Joint	Assembled	Primitive	Position				Velocity					
			Actual	Specified	Unit	Priority	Status	Actual	Specified	Units	Priority	Status
Base_Cran...	●	Rz	+150	+150	deg	High	●	-360	-360	deg/s	High	●
Base_Rock...	●	Rz	+173.824		deg			-179.769		deg/s		
Connecto...	●	Rz	+67.6893		deg			-249.628		deg/s		
Crank_Co...	●	Rz	-43.8653	-45	deg	Low	▲	+429.858		deg/s		

For more information, see

Simulation Video

You can record the video of a simulation for future reference or to share.

Mechanics Explorer provides a **Record** button  so that you can create the simulation video. All videos have a quality setting of 30 frames per second (fps) and AVI format. You can open a simulation video externally with any video player that supports AVI files. See “Record and Play Simulation Video” on page 6-18.

Rotate, Pan, and Zoom View


In this section...

“Rotate, Pan, and Zoom Shortcuts” on page 6-14

“Rotate View” on page 6-14

“Pan View” on page 6-15

“Zoom View” on page 6-16

The Mechanics Explorer toolbar contains three buttons to rotate, pan, and zoom the model view. Click the desired icon to enable the corresponding functionality in the visualization pane. The rotate, pan, and zoom buttons are .


Rotate, Pan, and Zoom Shortcuts

Mechanics Explorer provides mouse shortcuts that you can use to quickly pan, zoom in and out, and rotate a model. All mouse shortcuts use the mouse scroll wheel commonly available with the mouse tool. Mouse shortcuts provide a quicker and more intuitive way of interacting with a model in mechanics Explorer.

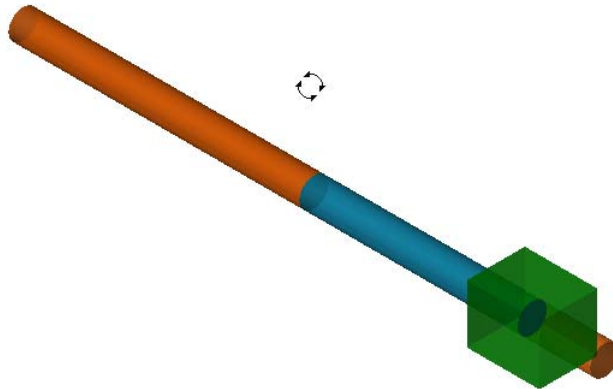
The following table lists all mouse shortcuts available in Mechanics Explorer.

Function	Mouse Shortcut
Rotate	Press Scroll Wheel + Move Mouse
Pan	Press Scroll Wheel + Shift + Move Mouse
Zoom	Press Scroll Wheel + Ctrl + Move Mouse

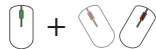
Rotate View

Mechanics Explorer includes a toolbar with a **Rotate view** button . Press the button to select the **Rotate view mode**. When you move the mouse to the visualization pane of Mechanics Explorer, the mouse arrow turns into a


rotation icon. Click the mouse anywhere in the model to set the rotation pivot point, then move the mouse to rotate about the pivot point.

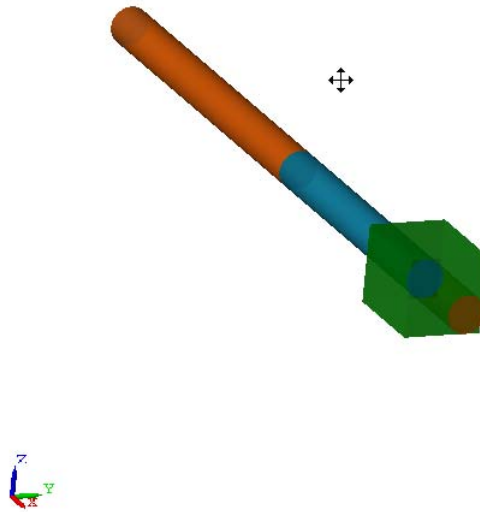


Alternatively, use the mouse shortcut to rotate the model:



Pan View


In the Mechanics Explorer toolbar, press the **Pan view** button . When you move the mouse to the visualization pane of Mechanics Explorer, the mouse arrow turns into a pan icon. Click the mouse anywhere in the model and move the mouse to pan the view.

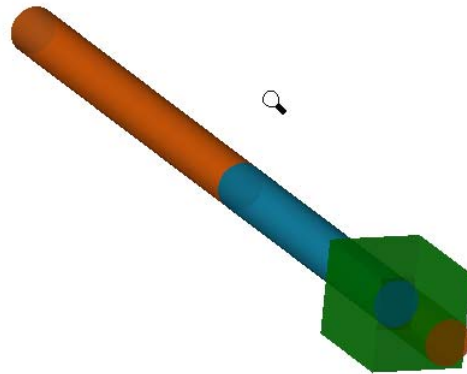


Alternatively, use the mouse shortcut to pan the model:

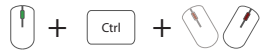


Zoom View

In the Mechanics Explorer toolbar, press the **Zoom in/out** button . When you move the mouse to the visualization pane of Mechanics Explorer, the mouse arrow turns into a zoom icon. Click the mouse anywhere in the model and move the mouse up to zoom in and down to zoom out.



Alternatively, use the mouse shortcut to pan the model:




Record and Play Simulation Video

In this section...
“Record and Save Simulation Video” on page 6-18
“Change Video Playback Speed” on page 6-20
“Play Simulation Video” on page 6-22

To share a SimMechanics simulation, record and save a video of the simulation for future playback. You can record videos with a quality setting of 30 fps (frames per second) and save them in AVI format.

The following example illustrates the animation capability of Mechanics Explorer. The example uses the four-bar model that accompanies your SimMechanics installation.

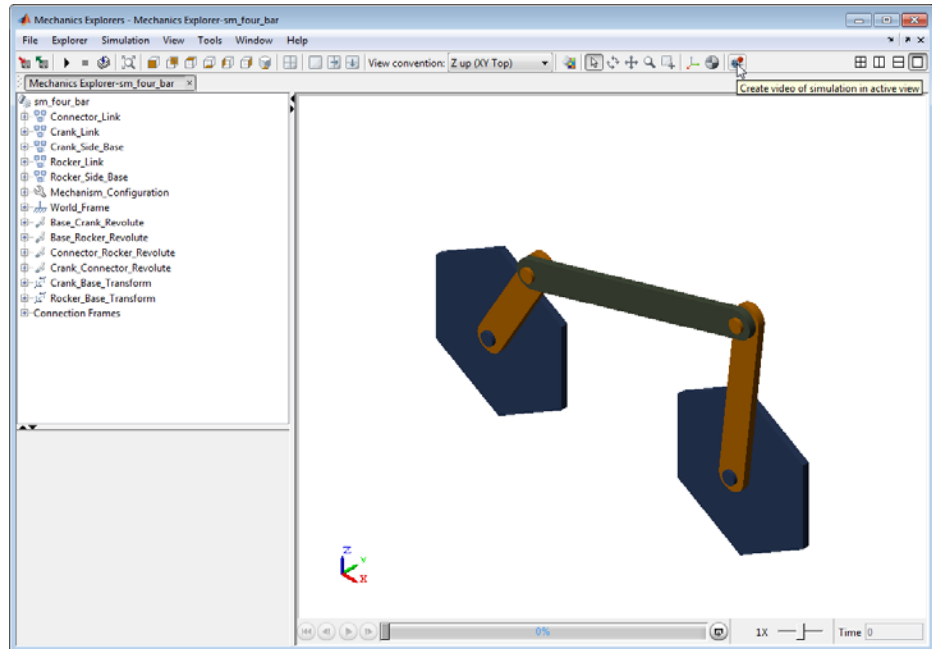
Record and Save Simulation Video

The Mechanics Explorer toolbar contains an icon to record and save simulation videos . Use the icon to record a simulation video of a four-bar model.

- 1 At the MATLAB command line, enter `sm_four_bar`.
- 2 In the Simulink Editor window, select **Simulation > Run**.

Note You must start the simulation before you can record the simulation video.

- 3 In the Mechanics Explorer window, press the **Record** button.

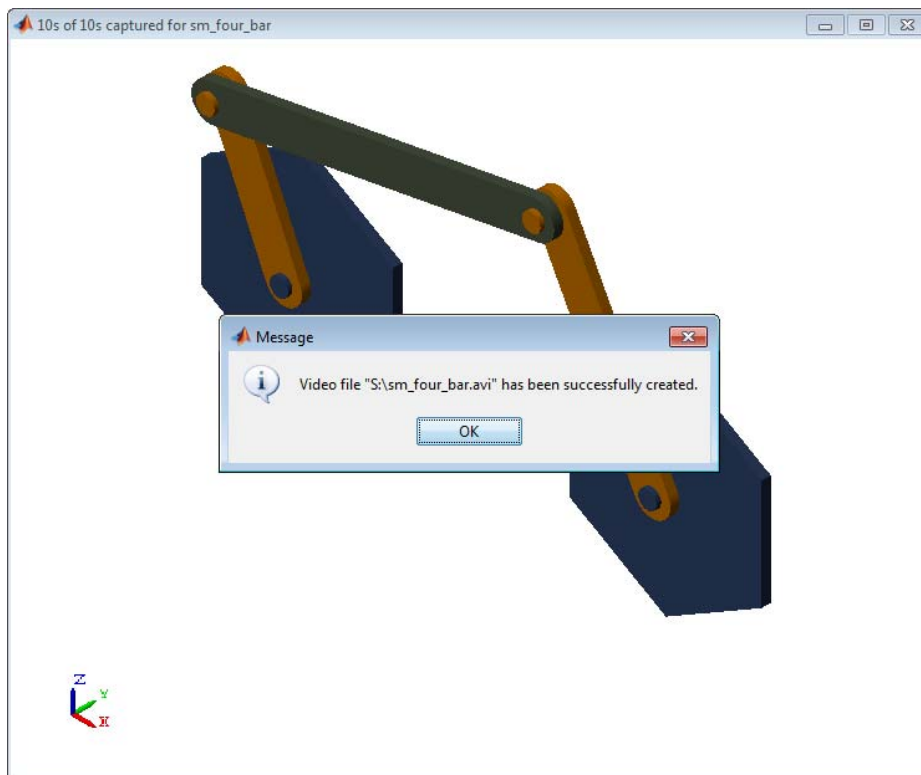


- 4 In the **Select video file** window, enter the name of the file.
- 5 Select the directory in which to save the file.
- 6 Press **Save**.

Note When you press **Save**, a new animation window opens. The window displays the animation in slow motion. The title bar of the window provides the progress status of the video recording. At the end of the recording, a new window opens informing you the recording has finished.

- 7 When the recording ends, click **OK** to close the window that contains the message Video file "S:\sm_four_bar.avi" has been successfully created.

Note If you renamed the video file name, the message replaces `sm_four_bar.avi` with the new file name.



Change Video Playback Speed

Simulation videos always play at a speed of 30 frames per second. Each video frame corresponds to a simulation time step, which you can specify in the **Model Configuration Parameters**. By default, when the simulation time step differs from the 1/30s video frame length, the simulation and video run at different speeds. This section shows how you can change the default settings to make the simulation and video run speeds equal.

The simulation step size depends on the solver you select. The simulation step size is constant for fixed-step solvers, and variable for variable-step solvers. You can set the step size for fixed-step solvers, or the minimum and maximum step sizes for variable-step solvers.

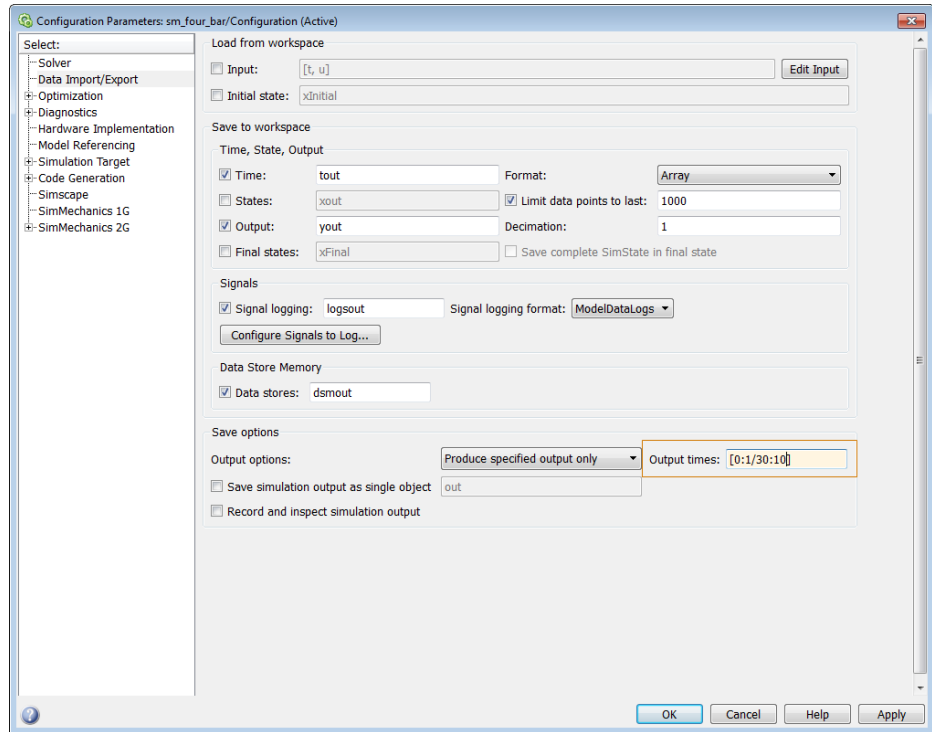
If you select a variable-step solver, the video plays *all* simulation time steps at the same rate of 1/30s, causing video time distortion. Selecting a fixed-step solver eliminates video time distortion, but the resulting video plays at a different speed unless you specify a simulation time step of 1/30s.

To record a video that plays at simulation speed, Use the following procedure. This procedure applies to both fixed- and variable-step solvers.

Note Always pick the solver that works best for your model. You can change the data output options to make simulation and video run times equal.

- 1** In the Simulink Editor window for your model, select **Simulation > Model Configuration Parameters**.
- 2** Select **Data Import/Export**.
- 3** In **Output Options**, select **Produce specified output only**.
- 4** In **Output times**, enter the array that defines the simulation output times, using a 1/30 second time interval:

[0:1/30:10]



The array that defines the simulation output times has the form [start time: sample time: end time]. All times use the second as a unit.

You can specify a sample time other than 1/30 to control the speed of the output video. A sample time of 2/30 causes the output video to play back at twice the simulation speed. A sample time of 1/60 causes the output video to play back at half the simulation speed.

Start time must be greater than or equal to the simulation start time. End time must be smaller than or equal to the simulation end time. Adjust start and end times to control the length of the video.

Play Simulation Video

Once you have recorded and saved a simulation video, you can play it in any media player that supports AVI files. To play the recorded video, navigate to

the directory where you saved the video. Then, select the name of the video file and open to play the video.

Visualization Troubleshooting

In this section...
“Mechanics Explorer Fails to Open” on page 6-24
“Mechanics Explorer Does not Display Imported Model” on page 6-24

The following issues can occur with Mechanics Explorer:

Mechanics Explorer Fails to Open

By default, Mechanics Explorer is set to open the first time you update a model. If a Mechanics Explorer window is already open for your model, the open window updates the model display.

If Mechanics Explorer fails to open during model update, check the model configuration parameters:

- 1** In the Simulink Editor menu bar, select **Simulation > Model Configuration Parameters**.
- 2** Expand the **SimMechanics 2G** node.
- 3** Click **Explorer**.
- 4** Check that **Open Mechanics Explorer on model update or simulation** is checked.

Mechanics Explorer Does not Display Imported Model

Imported models use a set of STL files that specify rigid body geometry. If you rename or remove the files, Mechanics Explorer does not display the corresponding rigid bodies.

Simulation still runs in the absence of the STL files, but without visualization. In the event this issue occurs, open the directory of the XML file you are importing, and check STL files exist. Open the XML file and check that the STL file names are correct.

CAD Import

About CAD Import

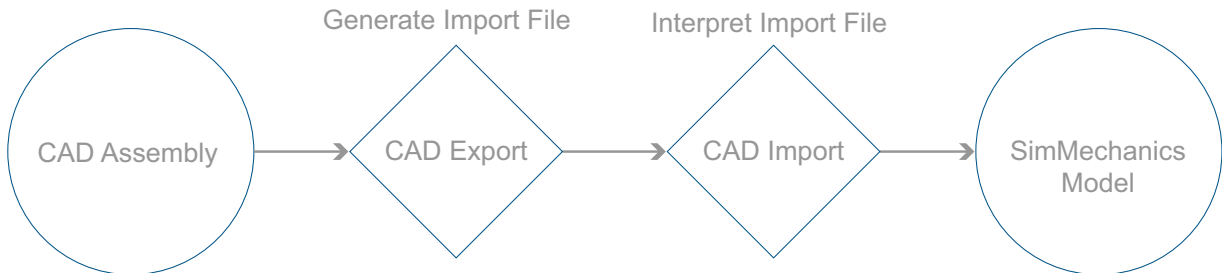
- “CAD Translation” on page 7-2
- “Install and Register SimMechanics Link Software” on page 7-8
- “About the SimMechanics Import XML File” on page 7-14
- “Import Robot Assembly” on page 7-24
- “Import Stewart Platform Model” on page 7-30
- “CAD Import Issues” on page 7-33

CAD Translation

In this section...
“Software Requirements” on page 7-2
“CAD Export” on page 7-3
“CAD Import” on page 7-4

CAD translation is a modeling process that converts a CAD assembly into a SimMechanics model. The complete process contains two sequential steps:

- 1 CAD Export** — The SimMechanics Link utility generates an import file. The file reflects assembly structure and contains part parameters.
- 2 CAD Import** — SimMechanics interprets the import file and generates a new model. The model structure and part parameters mirror the original CAD assembly.



Software Requirements

The complete CAD translation process requires the following software.

Software	Notes	CAD Export	CAD Import
CAD Platform		✓	
MATLAB	Registration as computing server required	✓	

Software	Notes	CAD Export	CAD Import
SimMechanics			✓
SimMechanics Link		✓	

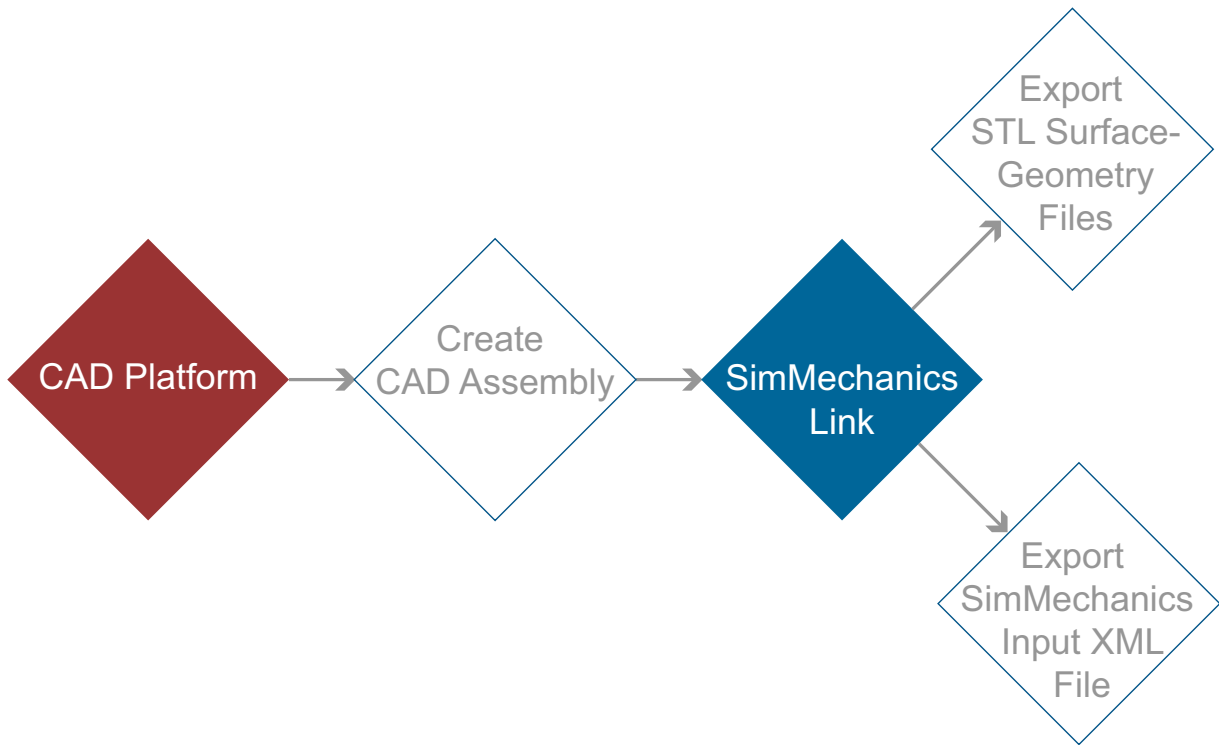
CAD Export

CAD export relies on the free utility SimMechanics Link. You can download the utility directly from the Mathworks website. See “Install and Register SimMechanics Link Software”. Following download and installation, registration of SimMechanics Link with a supported CAD platform adds the utility as an Add-In tool. You can now export a CAD assembly.

During CAD export, the SimMechanics Link utility generates one XML file and a set of STL files. The following table describes each file type.

File Type	Quantity	Purpose	Required for Model Generation	Required for Model Visualization
XML	1 total	Provide structure and parameters of CAD assembly in SimMechanics format	✓	
STL	1 per distinct CAD part	Provide 3-D surface geometry of CAD parts		✓

The files contain the assembly structure and part parameters required to generate an equivalent SimMechanics model. Assembly structure includes assembly-subassembly dependencies, which translate into SimMechanics system-subsystem dependencies. Part parameters include reference frames, mass and inertia, color, and location of part STL files.

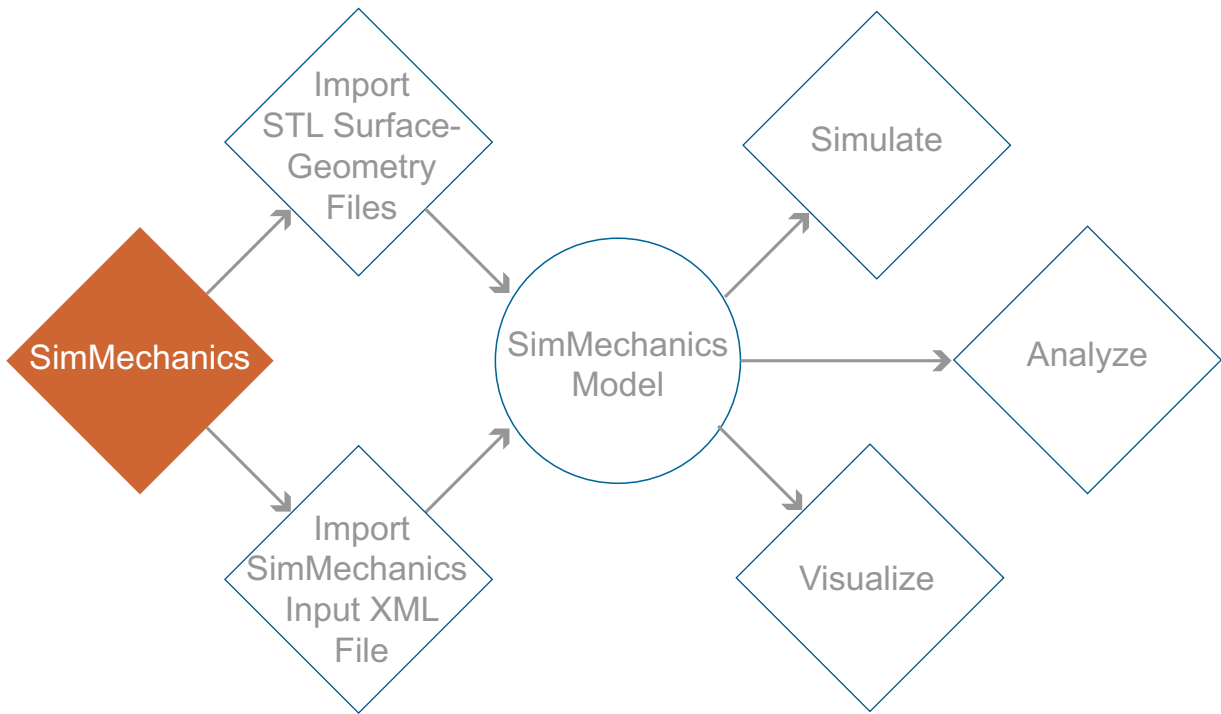


CAD Export Process

CAD Import

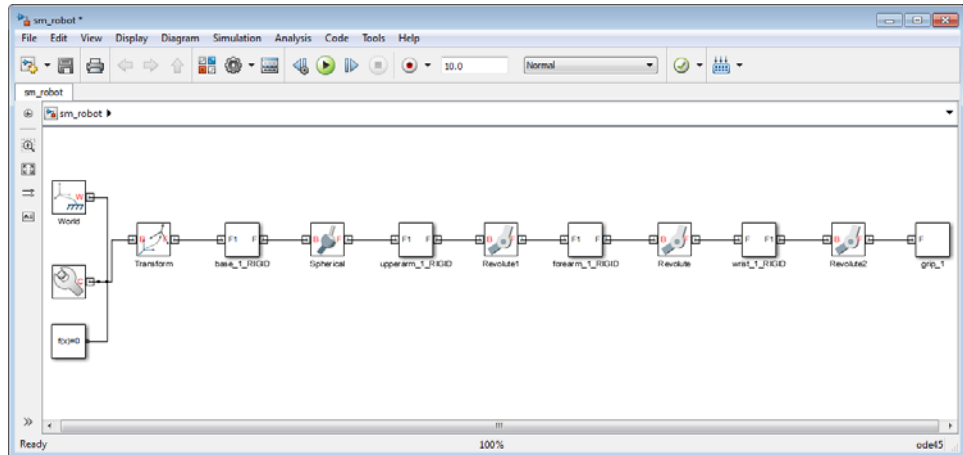
CAD import is the second and final step of CAD translation. During CAD import, SimMechanics interprets the XML import file. Then, based on the structure and parameters provided in the XML file, SimMechanics automatically generates a new model.

CAD Import does not require access to the original CAD assembly or associated CAD platform. Access to the surface-geometry STL files is not required for simulation, but it is required for visualization. You can simulate an imported model that contains no STL files, but the Mechanics Explorer visualization utility cannot display a representation of the model.



CAD Import Process

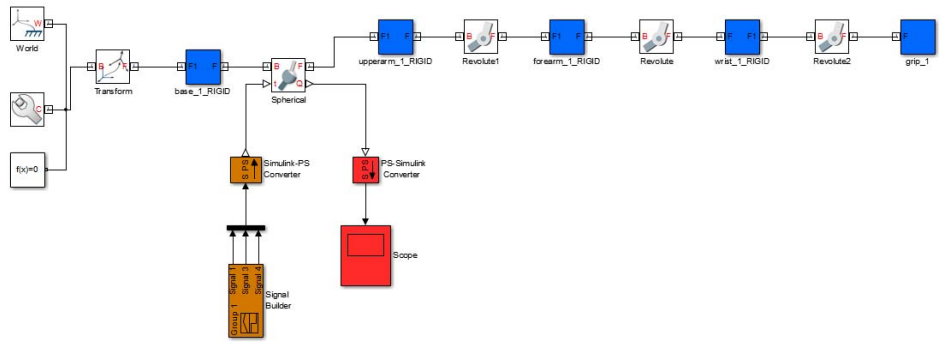
In the model, each CAD part maps into a rigid body subsystem. Each CAD constraint, or set of CAD constraints, map into a joint. Block names for SimMechanics subsystems are based on the original CAD parts and subassemblies which the subsystems represent. SimMechanics appends the suffix `RIGID` to the stem of a rigid body name. For example, CAD part `base` translates into rigid body subsystem `base_RIGID`. The following figure shows the imported SimMechanics model of a CAD robot assembly.



Modify the imported model to study the model behavior under different dynamic conditions. You can:

- Add actuation inputs to joints (e.g. **Spherical** in **sm_robot**), rigid body subsystems (e.g. **base_1_RIGID**), or multibody subsystems (e.g. **grip_1**).
- Measure the motion state parameters of a frame with respect to another frame.

The following figure shows a modified imported robot model. A Signal Builder block provides an actuation input to the spherical joint. A scope displays the resulting motion of the joint.



Install and Register SimMechanics Link Software

In this section...
“Installation Requirements” on page 7-8
“Download SimMechanics Link Software” on page 7-8
“Install SimMechanics Link Software” on page 7-9
“Register SimMechanics Link Utility with CAD Platform” on page 7-10
“Link External Application to SimMechanics Link Software” on page 7-10
“Register MATLAB as Automation Server” on page 7-11
“Unregister SimMechanics Link Software” on page 7-13

Installation Requirements

SimMechanics Link requires an active MATLAB installation on your machine. The following installation requirements apply:

Requirement	Example
MATLAB and SimMechanics Link must share the same release.	If the MATLAB release is R2012b, the SimMechanics Link release must also be R2012b
A supported CAD platform exists in your machine	SolidWorks®, Autodesk Inventor®, or PTC® Creo™ (Pro/ENGINEER®)
MATLAB, SimMechanics Link, and CAD platforms architecture must match	If MATLAB and CAD platform architectures are 64-bit, the SimMechanics Link architecture must also be 64-bit

Download SimMechanics Link Software

The SimMechanics Link utility is available as a free download from the MathWorks website. Follow these steps to download and install the utility:

- 1 Visit the SimMechanics Link download website at http://www.mathworks.com/products/simmechanics/download_smlink.html.

- 2 Select the SimMechanics Link version that corresponds to the MATLAB version on your machine.
- 3 Enter your contact information and click **Submit**.
- 4 In the new web page, open the zip file appropriate for your operating system (e.g. *smlink.r2012b.win64.zip*).

Caution Do not rename the zip file.

- 5 Save the file `install_addon.m` appropriate for your operating system in the zip file directory.

Install SimMechanics Link Software

Use the MATLAB command line to install SimMechanics Link software.

- 1 Start a new MATLAB session.
- 2 At the MATLAB command line, enter the command:

```
path(path, 'download_file_directory')
```

String `download_file_directory` identifies the directory that contains the download files. The command adds the directory to the MATLAB search path.

Note Skip this step if the download file directory is already on the MATLAB search path.

- 3 At the MATLAB command line, enter the command:

```
install_addon('zip_file_name')
```

String `'zip_file_name'` is the name of the download zip file (e.g. *smlink.r2012b.win64*). The command extracts the zip archive files to the MATLAB root directory.

Caution You may need Administrator privileges to extract the files to the MATLAB root directory. If the MATLAB command window issues “Permission denied to create file” messages, try restarting MATLAB as an Administrator.

- 4** Verify the MATLAB command window issues the message Installation of smlink complete.

Register SimMechanics Link Utility with CAD Platform

Complete the installation by registering your the SimMechanics Link utility with your CAD platform. The registration procedure makes SimMechanics Link available in your CAD platform as an Add-In tool. Once you have completed the linking procedure, you can use the Add-In tool to export a CAD assembly directly from your CAD platform.

The registration procedure is different for each supported CAD platform. The following table provides platform-specific registration information. Click the link that matches your CAD platform, and complete the registration procedure.

To register with CAD platform...	...click here
Autodesk Inventor	“Register SimMechanics Link with Inventor®”
PTC Creo (Pro/ENGINEER)	“Register SimMechanics Link with Creo”
SolidWorks	“Register SimMechanics Link with SolidWorks”

Link External Application to SimMechanics Link Software

You can link an unsupported CAD platform or other external application to SimMechanics software. For this task, SimMechanics Link provides an application programming interface (API) with a set of functions that you

can use to create a C/C++ custom export module. For an overview of custom export using the API, see “Custom Export with SimMechanics Link API”.

Register MATLAB as Automation Server

Each time you use the SimMechanics Link utility with a CAD platform or other external application, the utility attempts to connect to MATLAB.

Registration Requirements

Successful connection requires the following to be true:

- Matching MATLAB and SimMechanics Link release numbers (e.g. both release numbers R2012b)
- MATLAB registration as automation server.

Enable Automation Server Mode

You can register MATLAB as an automation server in two ways:

Condition	Registration Procedure
MATLAB session open in desktop mode	<p>At the MATLAB command line, enter <code>regmatlabserver</code>.</p> <p>The command registers the current MATLAB session as an automation server.</p>
	<p>At the MATLAB command line, enter <code>enableservice('AutomationServer',true)</code>.</p> <p>The command enables the current MATLAB session as an automation server.</p>

Condition	Registration Procedure
MATLAB session not open	<p>At the operating system command prompt, enter</p> <pre>matlab -automation -desktop</pre> <p>The prompt starts a new MATLAB session in automation server mode.</p>
	<p>At the operating system command prompt, enter command <code>matlab -regserver</code>.</p> <p>The command opens a new MATLAB session in automation server mode. You can close the MATLAB session.</p>

A single MATLAB automation server registration can be active at a time. If multiple MATLAB sessions are open in your system, you must *first* disable the active registration and *then* register the desired MATLAB session as an automation server using the `regmatlabserver` command.

Caution If your system does not have an active MATLAB automation server registration, SimMechanics Link issues a error when it attempts to connect. In the event of a connection error, check that a MATLAB automation server is active in your system. If necessary, register MATLAB as an automation server.

Connection from External Application to MATLAB Automation Server

Invoking the SimMechanics Link utility from an external application produces one of the following results:

Condition	Required Action	Result
No MATLAB session open	None	<ul style="list-style-type: none"> • New MATLAB session opens in automation server mode • SimMechanics Link connects to MATLAB automation server
MATLAB server open in automation server mode	None	<ul style="list-style-type: none"> • SimMechanics Link connects to MATLAB automation server
MATLAB session open in desktop mode	Register MATLAB session as automation server.	<ul style="list-style-type: none"> • SimMechanics Link connects to MATLAB automation server

Unregister SimMechanics Link Software

SimMechanics Link contains no uninstaller. If you no longer wish to use the SimMechanics Link utility in your CAD platform, you can unregister the utility. The following table provides information on the unlinking procedure for each CAD platform. Click the link that matches your CAD platform.

To link CAD platform...	...click here
Autodesk Inventor	"Register SimMechanics Link with Inventor"
PTC Creo (Pro/ENGINEER)	"Register SimMechanics Link with Creo"
SolidWorks	"Register SimMechanics Link with SolidWorks"

To register a different version of SimMechanics Link with your CAD platform, first unregister any currently registered version you may have. Then, register the desired version. To register and unregister the utility, follow the links provided in the previous table.

About the SimMechanics Import XML File

In this section...
“Organization of SimMechanics XML Import File” on page 7-14
“Root Assembly” on page 7-15
“Organization of Assemblies” on page 7-19
“Organization of Parts” on page 7-20

The SimMechanics XML import file contains the hierarchical structure of a CAD assembly and the physical parameters that describe each CAD part. SimMechanics imports the file to automatically generate an equivalent SimMechanics model with little additional work on your part. You can modify the automatically generated model as any other SimMechanics model.

Each block in an imported model receives a unique identifier name and a complete set of parameters based on the XML file data. You do not need to manually specify imported parameters.

The following sections describe the information present in the SimMechanics XML import file.

Organization of SimMechanics XML Import File

CAD assemblies are hierarchical systems: a CAD root assembly contains other CAD subassemblies, each made of CAD parts. The SimMechanics XML import file mirrors the hierarchical structure of a CAD assembly. The file organizes CAD assembly information in the order Root Assembly→Assemblies→Parts.

The following figure shows the SimMechanics XML import file for a CAD assembly with name robot. Content in sections RootAssembly, Assemblies, and Parts is removed for clarity.

```

<SimMechanicsImportXML version="1.0"
  <Created by="" on="04/12/12|12:00:36" using="SimMechanics Link Version 4.0" from="SolidWorks 18.0.0"/>
  <ModelUnits mass="kilogram" length="centimeter"/>
  <DataUnits mass="kilogram" length="meter"/>

  <RootAssembly name="robot" uid="robot" version="291">
    ...
  </RootAssembly>
<Assemblies>
  ...
</Assemblies>
<Parts>
  ...
</Parts>
</SimMechanicsImportXML>

```

Root Assembly

The section `RootAssembly` of the SimMechanics XML import file organizes information into two separate subsections:

- InstanceTree
- Constraints

```

<RootAssembly name="robot" uid="robot" version="291">
  <AssemblyFile name="robot.SLDASM" type="SolidWorks Assembly"/>
  <InstanceTree>
    ...
  </InstanceTree>
  <Constraints>
    ...
  </Constraints>
</RootAssembly>

```

InstanceTree

Each part contains one body-fixed reference frame that represents a unique set of position and orientation coordinates. `InstanceTree` defines a reference frame for each assembly found in the root assembly. One frame provides an ultimate reference frame with origin coordinates (0,0,0). Rigid transformations translate and rotate the previous frame in `InstanceTree` to obtain the reference frame for another CAD assembly.

Instance sections contain the rigid transformation that defines the reference frame for a CAD part. The following figure shows an instance section in the SimMechanics XML import file for a root assembly with name robot.

```
<RootAssembly name="robot" uid="robot" version="291">
  <AssemblyFile name="robot.SLDASM" type="SolidWorks Assembly"/>
  <InstanceTree>
    <Instance name="base-1" uid="base-1" grounded="true" entityUid="base:*Default">
      <Transform>
        <Rotation>1 0 0 0 1 0 0 0 1</Rotation>
        <Translation>0 0 0</Translation>
      </Transform>
    </Instance>
  </InstanceTree>
</RootAssembly>
```

The InstanceTree section defines the hierarchical organization of the CAD assembly. The section organizes CAD assemblies and parts according to their place in the root assembly hierarchy. The following figure displays a SimMechanics XML import file for a CAD root assembly with name Robot. The root assembly contains five assemblies:

- base-1
- upperarm-1
- forearm-1
- wrist-1
- grip-1

All assemblies contain a single part, except assembly Grip. The assembly Grip is a multibody system that connects multiple parts with joints. Grip contains seven distinct parts:

- metacarples-1
- firstfingerlink-1
- firstfingerlinkL-1
- secondfingerlink-1
- secondfingerlink-2
- fingertips-1
- fingertips-2

Instance content is removed for clarity.

```

<RootAssembly name="robot" uid="robot" version="291">
  <AssemblyFile name="robot.SLDASM" type="SolidWorks Assembly"/>
  <InstanceFree>
    <Instance name="base-1" uid="base-1" grounded="true" entityUid="base:*Default">
      ...
    </Instance>
    <Instance name="upperarm-1" uid="upperarm-1" entityUid="upperarm:*Default">
      ...
    </Instance>
    <Instance name="forearm-1" uid="forearm-1" entityUid="forearm:*Default">
      ...
    </Instance>
    <Instance name="wrist-1" uid="wrist-1" entityUid="wrist:*Default">
      ...
    </Instance>
    <Instance name="grip-1" uid="grip-1" entityUid="grip">
      ...
      <Instance name="metacarples-1" uid="metacarples-1" grounded="true" entityUid="metacarples:*Default">
        ...
      </Instance>
      <Instance name="firstfingerlink-1" uid="firstfingerlink-1" entityUid="firstfingerlink:*Default">
        ...
      </Instance>
      <Instance name="firstfingerlinkL-1" uid="firstfingerlinkL-1" entityUid="firstfingerlinkL:*Default">
        ...
      </Instance>
      <Instance name="secondfingerlink-1" uid="secondfingerlink-1" entityUid="secondfingerlink:*Default">
        ...
      </Instance>
      <Instance name="secondfingerlink-2" uid="secondfingerlink-2" entityUid="secondfingerlink:*Default">
        ...
      </Instance>
      <Instance name="fingertips-1" uid="fingertips-1" entityUid="fingertips:*Default">
        ...
      </Instance>
      <Instance name="fingertips-2" uid="fingertips-2" entityUid="fingertips:*Default">
        ...
      </Instance>
    </Instance>
  </InstanceFree>
</RootAssembly>

```

Constraints

CAD constraints define how two CAD parts can move relative to each other. One CAD constraint connects two CAD parts. Each CAD constraint specifies the mechanical degrees of freedom present between two CAD parts. Two CAD parts can translate along, and rotate about, up to three mutually orthogonal axes.

During CAD import, SimMechanics translates the CAD constraints into SimMechanics joints. Most CAD constraints have a SimMechanics equivalent, but the equivalence may not be a one-to-one-correspondence. A single SimMechanics joint may require a combination of multiple CAD constraints providing the same degrees of freedom.

Note Not all CAD constraints have a SimMechanics equivalent. CAD gear constraints are one example. You cannot translate a CAD gear constraint into SimMechanics Second Generation models.

The **Constraints** section specifies the position, orientation, and type of joint that connects each pair of CAD assemblies. Two constraints specify one joint. The following figure shows the constraints section of the SimMechanics XML import file for a joint between two CAD parts with names `upperarm-1` and `forearm-1`. In the figure, two constraints define a revolute joint that connects the two CAD parts: **Concentric** and **Coincident**. Each constraint specifies the position and orientation of the revolute joint relative to the reference frame for each CAD part.


```

<Constraints>
  <Concentric name="Concentric2">
    <ConstraintGeometry geomType="cylinder">
      <InstancePath>
        <Uid>upperarm-1</Uid>
      </InstancePath>
      <Position>0.10033 -0.0449642 0</Position>
      <Axis>0 1 1.66911e-016</Axis>
    </ConstraintGeometry>
    <ConstraintGeometry geomType="cylinder">
      <InstancePath>
        <Uid>forearm-1</Uid>
      </InstancePath>
      <Position>-0.01651 -0.01651 0</Position>
      <Axis>0 1 0</Axis>
    </ConstraintGeometry>
  </Concentric>
  <Coincident name="Coincident3">
    <ConstraintGeometry geomType="plane">
      <InstancePath>
        <Uid>upperarm-1</Uid>
      </InstancePath>
      <Position>0 -0.001016 0</Position>
      <Axis>0 -1 0</Axis>
    </ConstraintGeometry>
    <ConstraintGeometry geomType="plane">
      <InstancePath>
        <Uid>forearm-1</Uid>
      </InstancePath>
      <Position>0 0 0</Position>
      <Axis>0 1 0</Axis>
    </ConstraintGeometry>
  </Coincident>

```

Organization of Assemblies

The Assemblies section provides the same information present in RootAssembly, but with a local non-inertial reference frame acting as the ultimate reference frame. An InstanceTree section assigns a local reference frame to each part in an assembly. Each local reference frame appears

in a separate Instance subsection. In the Instance subsection, a rigid transformation rotates and translates a parent frame to obtain the new local reference frame.

A Constraints section specifies the kinematic constraints between two parts. The set of constraints between two parts define the kinematic degrees of freedom between them, and are equivalent to SimMechanics joints. During CAD import, SimMechanics interprets each set of CAD constraints, and replaces them with the appropriate set of joints.

Organization of Parts

Part Names

Each part receives a unique name. By default, part names originate from the part file names. You can change a part name in SimMechanics, after CAD import, or in the SimMechanics XML import file, before CAD import. The following figure displays the part name section of the SimMechanics XML import file. Colored Boxes highlight part and source file identification information.

```
<Part name="wrist" uid="wrist:*Default" version="323">
  <ModelUnits mass="kilogram" length="centimeter"/>
  <PartFile name="wrist.SLDPRT" type="SolidWorks Part"/>
  <MassProperties>
    <Mass>0.151682</Mass>
    <CenterOfMass>-0.00457306 3.6667e-009 2.08473e-009</CenterOfMass>
    <Inertia>2.71068e-005 4.63034e-005 3.87938e-005 1.54966e-011 -5.65388e-012 -4.38201e-012</Inertia>
  </MassProperties>
  <GeometryFile name="wrist_Default_sldprt.STL" type="STL"/>
  <VisualProperties>
    <Ambient r="1" g="0.788235" b="0.576471" a="1"/>
    <Diffuse r="1" g="0.788235" b="0.576471" a="1"/>
    <Specular r="1" g="0.788235" b="0.576471" a="1"/>
    <Emissive r="0" g="0" b="0" a="1"/>
    <Shininess>0.3125</Shininess>
  </VisualProperties>
</Part>
```

Note SimMechanics represents a CAD part as a single rigid body subsystem. A rigid body subsystem inherits its name from the corresponding CAD part.

Physical Units

The SimMechanics XML input file defines the physical units used to resolve the values of inertial parameters. Units originate from the exported CAD assembly file. You can update the physical units in SimMechanics, after CAD import, or in the SimMechanics XML import file, before CAD import. The following figure highlights the physical units used to resolve the inertial properties of CAD part with name Wrist.

```
<Part name="wrist" uid="wrist*:Default" version="323">
  <ModelUnits mass="kilogram" length="centimeter"/>
  <PartFile name="wrist.SLDPRT" type="SolidWorks Part"/>
  <MassProperties>
    <Mass>0.151682</Mass>
    <CenterOfMass>-0.00457306 3.6667e-009 2.08473e-009</CenterOfMass>
    <Inertia>2.71068e-005 4.63034e-005 3.87938e-005 1.54966e-011 -5.65388e-012 -4.38201e-012</Inertia>
  </MassProperties>
  <GeometryFile name="wrist_Default_sldprt.STL" type="STL"/>
  <VisualProperties>
    <Ambient r="1" g="0.788235" b="0.576471" a="1"/>
    <Diffuse r="1" g="0.788235" b="0.576471" a="1"/>
    <Specular r="1" g="0.788235" b="0.576471" a="1"/>
    <Emissive r="0" g="0" b="0" a="1"/>
    <Shininess>0.3125</Shininess>
  </VisualProperties>
</Part>
```

Solid Parameters

Solid parameters include inertia and graphic properties. Inertia governs the dynamic response of the solid to an applied force or torque. The SimMechanics XML import file specifies the following inertial parameters:

- Mass
- Center of mass
- Moments and products of inertia

The following figure displays the solid parameters section of the SimMechanics XML import file for a CAD part with name Wrist. A box encloses the inertial properties of the CAD part.

```

<Part name="wrist" uid="wrist*:*Default" version="323">
  <ModelUnits mass="kilogram" length="centimeter"/>
  <PartFile name="wrist.SLDPRT" type="SolidWorks Part"/>
  <MassProperties>
    <Mass>0.151682</Mass>
    <CenterOfMass>-0.00457306 3.6667e-009 2.08473e-009</CenterOfMass>
    <Inertia>2.71068e-005 4.63034e-005 3.87938e-005 1.54966e-011 -5.65388e-012 -4.38201e-012</Inertia>
  </MassProperties>
  <GeometryFile name="wrist_Default_sldprt.STL" type="STL"/>
  <VisualProperties>
    <Ambient x="1" g="0.788235" b="0.576471" a="1"/>
    <Diffuse x="1" g="0.788235" b="0.576471" a="1"/>
    <Specular x="1" g="0.788235" b="0.576471" a="1"/>
    <Emissive x="0" g="0" b="0" a="1"/>
    <Shininess>0.3125</Shininess>
  </VisualProperties>
</Part>

```

Graphic properties govern the visual representation of the solid in Mechanics Explorer. Properties include color and shininess. The following table describes the graphic properties present in the SimMechanics XML import file.

Graphic Property	Type	Description
Ambient Color	RGBA vector	Color of light that hits the solid surface
Diffuse Color	RGBA vector	Color of the solid surface in pure white light
Specular Color	RGBA vector	Color of specular reflection from the solid surface
Emissive Color	RGBA vector	Color of solid self-illumination
Shininess	Scalar	Intensity of specular highlights from the solid surface

The following figure highlights the graphic properties section of the SimMechanics XML import file.

```

<Part name="wrist" uid="wrist*:*Default" version="323">
  <ModelUnits mass="kilogram" length="centimeter"/>
  <PartFile name="wrist.SLDPRT" type="SolidWorks Part"/>
  <MassProperties>
    <Mass>0.151682</Mass>
    <CenterOfMass>-0.00457306 3.6667e-009 2.08473e-009</CenterOfMass>
    <Inertia>2.71068e-005 4.63034e-005 3.87938e-005 1.54966e-011 -5.65388e-012 -4.38201e-012</Inertia>
  </MassProperties>
  <GeometryFile name="wrist_Default_sldprt.STL" type="STL"/>
  <VisualProperties>
    <Ambient r="1" g="0.788235" b="0.576471" a="1"/>
    <Diffuse r="1" g="0.788235" b="0.576471" a="1"/>
    <Specular r="1" g="0.788235" b="0.576471" a="1"/>
    <Emissive r="0" g="0" b="0" a="1"/>
    <Shininess>0.3125</Shininess>
  </VisualProperties>
</Part>

```

Geometry File References

A set of STL files specifies the 3-D geometry of the solid surface for each CAD part. STL files specify only geometry, without reference to other graphic properties, like color. The SimMechanics XML import file specifies a single STL geometry file for each part in the CAD assembly. The following figure highlights the geometry file reference in the SimMechanics XML input file for a part with name Wrist.

```

<Part name="wrist" uid="wrist*:*Default" version="323">
  <ModelUnits mass="kilogram" length="centimeter"/>
  <PartFile name="wrist.SLDPRT" type="SolidWorks Part"/>
  <MassProperties>
    <Mass>0.151682</Mass>
    <CenterOfMass>-0.00457306 3.6667e-009 2.08473e-009</CenterOfMass>
    <Inertia>2.71068e-005 4.63034e-005 3.87938e-005 1.54966e-011 -5.65388e-012 -4.38201e-012</Inertia>
  </MassProperties>
  <GeometryFile name="wrist_Default_sldprt.STL" type="STL"/>
  <VisualProperties>
    <Ambient r="1" g="0.788235" b="0.576471" a="1"/>
    <Diffuse r="1" g="0.788235" b="0.576471" a="1"/>
    <Specular r="1" g="0.788235" b="0.576471" a="1"/>
    <Emissive r="0" g="0" b="0" a="1"/>
    <Shininess>0.3125</Shininess>
  </VisualProperties>
</Part>

```

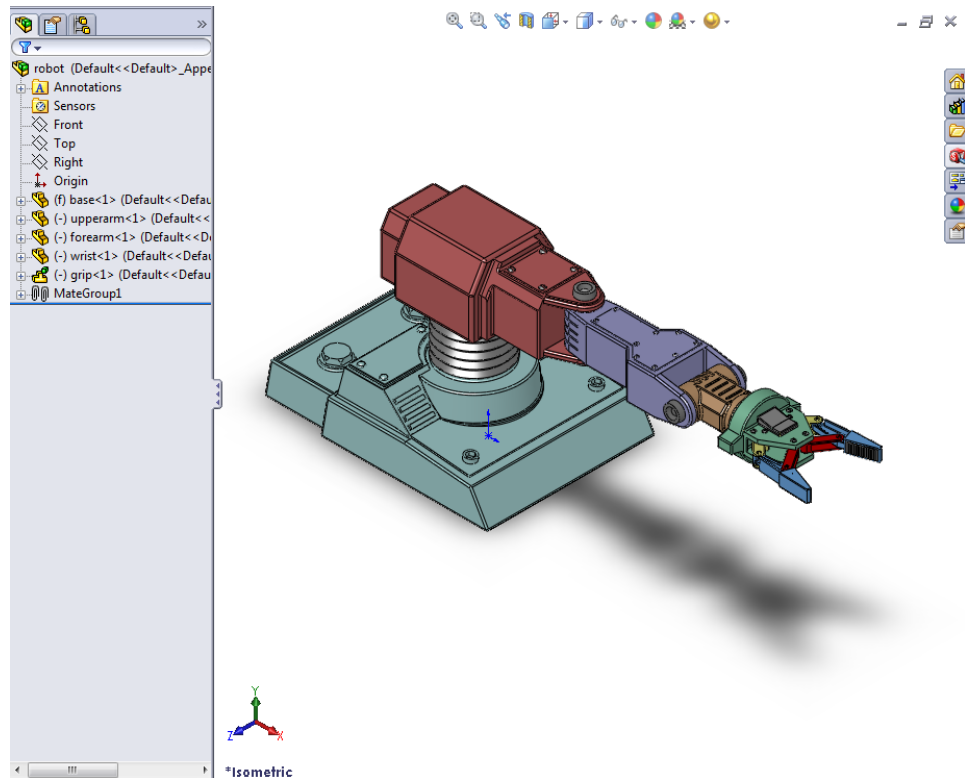
Import Robot Assembly

In this section...
“Verify Import Files” on page 7-25
“Import Robot Assembly” on page 7-27
“Visualize and Simulate Robot Assembly” on page 7-27

In this example, you import a CAD assembly with name robot into SimMechanics. SimMechanics provides the `smimport` command so that you can import a CAD assembly. The command is the only SimMechanics tool you need to import a CAD assembly. The CAD import procedure is the same for all CAD platforms.

Note This example uses an XML file and a set of STL files that are present in your SimMechanics installation. You can export the XML and STL files directly from a supported CAD platform, but the names of the files may differ from the example.

The following figure shows the original CAD assembly inside the SolidWorks CAD platform.



Verify Import Files

Before you import the `sm_robot` CAD assembly, verify the existence of all required files. Files include one SimMechanics Import XML file and a set of STL files that specify the geometry of all CAD parts.

- 1 At the MATLAB command line, enter the following command to change the current working directory to the subdirectory that contains the robot example files:

```
cd(fullfile(matlabroot, 'toolbox', 'physmod', 'sm', 'smdemos',
'import', 'robot'))
```

- 2 At the MATLAB command line, enter `ls` or `dir` to list all files in the `\robot` directory.

3 Verify that the following files exist:

SimMechanics Import XML File

Name	Type	Description
sm_robot.xml	SimMechanics Import XML File	The XML file defines the structure of the CAD assembly. The file also contains the parameters required to model each part in the CAD assembly.

STL Geometry Files

Name	Type	Description
robot_base_Default_sldprt.STL	STL Geometry files	The STL files define the surface geometry of each robot CAD part
robot_fingertips_Default_sldprt.STL		
robot_firstfingerlink_Default_sldprt.STL		
robot_firstfingerlinkL_Default_sldprt.STL		
robot_forearm_Default_sldprt.STL		
robot_metacarples_Default_sldprt.STL		
robot_secondfingerlink_Default_sldprt.STL		
robot_upperarm_Default_sldprt.STL		
robot_wrist_Default_sldprt.STL		

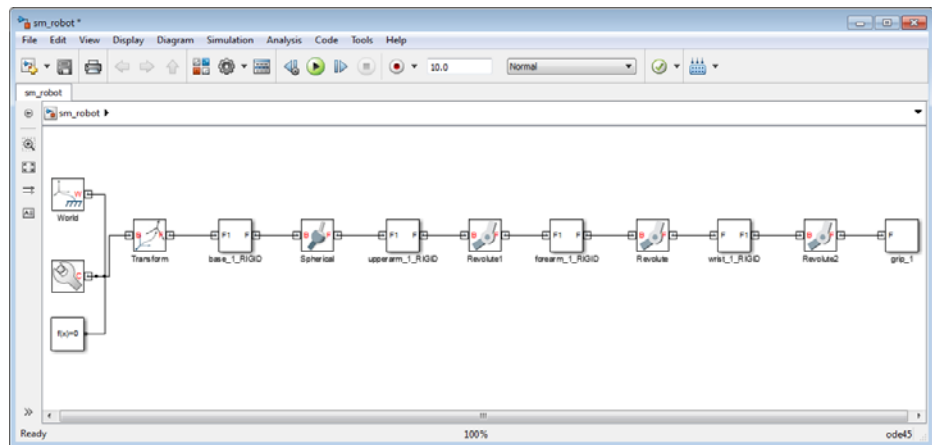
The utility SimMechanics Link automatically generates all files from any of the three CAD platforms that SimMechanics supports. If the files do not exist in your \robot directory, you can export them from SolidWorks with the SimMechanics Link utility. The SimMechanics Link installation contains a set of CAD files for the robot assembly that you can export. If you export the files for this example, the names of the files may be different.

To export the files, see “Export Robot CAD Assembly”.

Import Robot Assembly

Once you have verified that all required files exist, proceed to import the assembly.

- 1 At the MATLAB command line, enter `smimport('sm_robot.xml')`.
- 2 Confirm that SimMechanics opens a new model with name `sm_robot`.



Note SimMechanics automatically generates the new model without extra input on your part. Review the model and check for errors and inconsistencies in the block diagram.

- 3 In the Simulink Editor window that contains the model, select **File > Save As**.
- 4 In the **Save As** dialog box, enter the desired file name and select a convenient directory in which store the model file.

Visualize and Simulate Robot Assembly

- 1 In the Simulink Editor window that contains the robot model, select **Simulation > Update Diagram** or press **Ctrl+D**.

Note When you update the diagram, SimMechanics automatically updates the model display in Mechanics Explorer. SimMechanics relies on the set of STL files to represent the 3-D geometry of each CAD part. If the files are not available, SimMechanics still generates the model, but Mechanics Explorer cannot display the assembly.

- 2** In the Mechanics Explorer toolbar, set **View Convention** to Y up (XY Front).

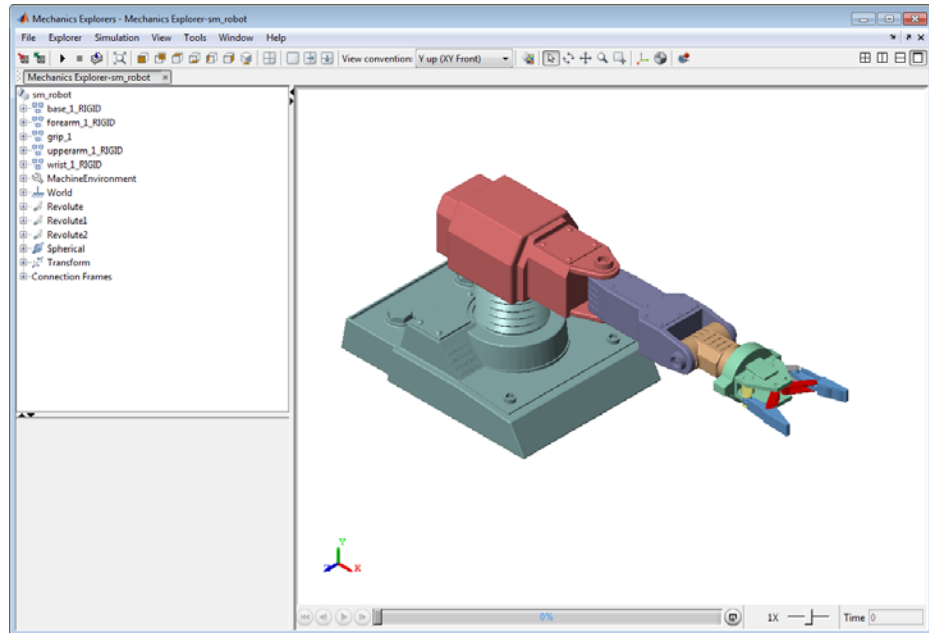
Note Most CAD systems use a Y up default view convention. The convention differs from the Mechanics Explorer default setting, Z up. Selecting the Y up view convention causes Mechanics Explorer to display the assembly with the same orientation used in the CAD platforms.

- 3** In the toolbar, click the icon for the desired viewpoint.

Note Selecting the Y up view convention does not affect the Mechanics Explorer display until you click a view point. You have the choice between seven standard viewpoints: front, back, top, down, left, right, and isometric. Once you select a view point, you can rotate, pan, and zoom to adjust the display of your model. For more information, see:

- “Configure Mechanics Explorer Display” on page 6-2
 - “Rotate, Pan, and Zoom View” on page 6-14
-

- 4** Confirm that a Mechanics Explorer window opens with a static display of the robot assembly.



5 In the Simulink Editor window for the model, select **Simulation > Run** or press **Ctrl+T** to simulate the model.

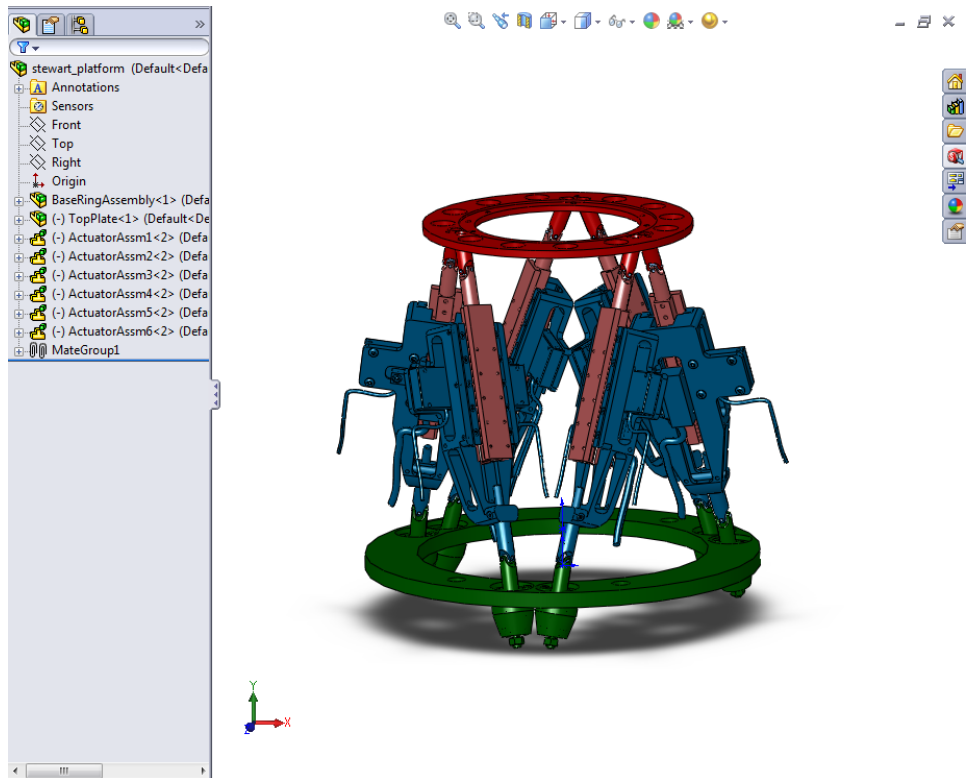
Tip The model lacks actuation inputs. When you simulate the model, the robot arm moves strictly due to gravity effects. You can change the gravity specification in the Mechanism Configuration block.

You can add actuation inputs to the model. Add a block from the Forces & Torques library to actuate a rigid body. Select an actuation mode in the model joint blocks to actuate a joint. For more information, see “Model Actuation Workflow” on page 4-2.

Import Stewart Platform Model

In this example, you import a CAD assembly with name `stewart_platform` into SimMechanics. SimMechanics provides the `smimport` command so that you can import a CAD assembly. The command is the only SimMechanics tool you need to import a CAD assembly.

Note This example requires export of the Stewart platform according to SimMechanics Link example “Export Robot Assembly from SolidWorks Software”.



1 Navigate to the directory

```
<matlabroot>/toolbox/physmod/sm/smdemos/...  
...import/stewart_platform
```

2 Check that the following files exist:

- **XML file** — Provides the structure of the CAD assembly and the parameters of the CAD parts.
- **Set of STL files** — Provide the surface geometry of the 26 unique CAD parts present in the CAD assembly.

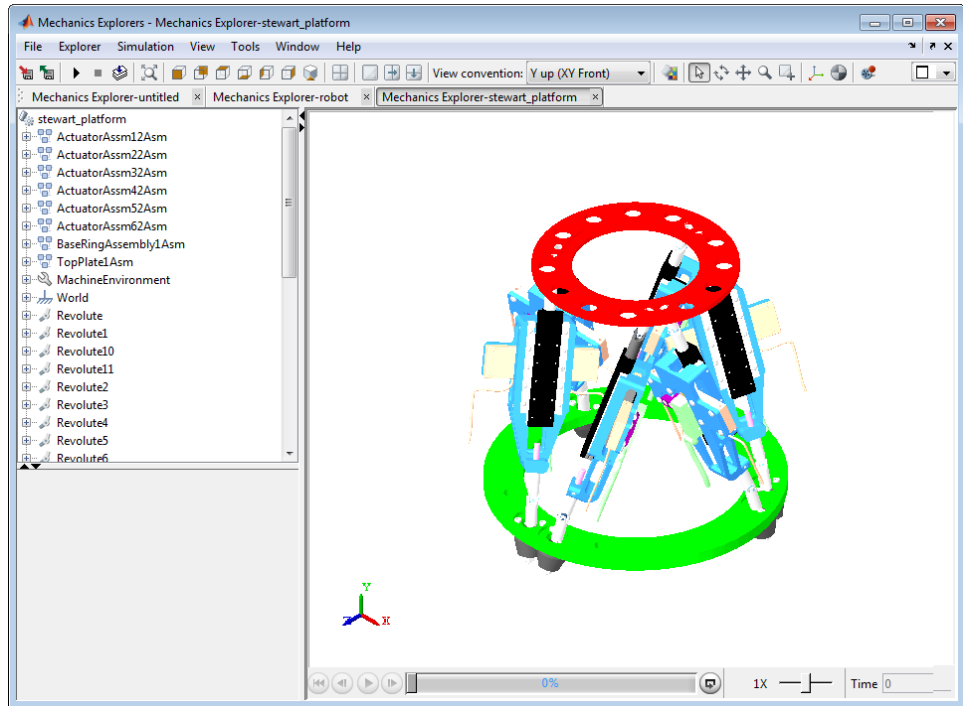
The utility SimMechanics Link automatically generates all files from any of the three CAD platforms that SimMechanics supports. If the files do not exist, you must export them from your CAD platform with SimMechanics Link. For more information, see “CAD Translation” on page 7-2.

3 At the MATLAB command line, enter
`smimport('stewart_platform.xml')`.

SimMechanics imports the files and automatically generates the corresponding model.

4 On the Simulink menu bar, click **Simulation > Update Diagram**.

Mechanics Explorer opens with a 3-D visual representation of the SimMechanics model.



Tip Actuate the `stewart_platform` model with blocks from the **Forces and Torques** library. Then, simulate the model and analyze its dynamic behavior in Mechanics Explorer.

CAD Import Issues

In this section...

“SimMechanics Replaces Unsupported Constraint with Rigid Connection” on page 7-33

“Default View Convention is Z-Axis Up” on page 7-34

“Invalid STL Files Cause Import Issues” on page 7-35

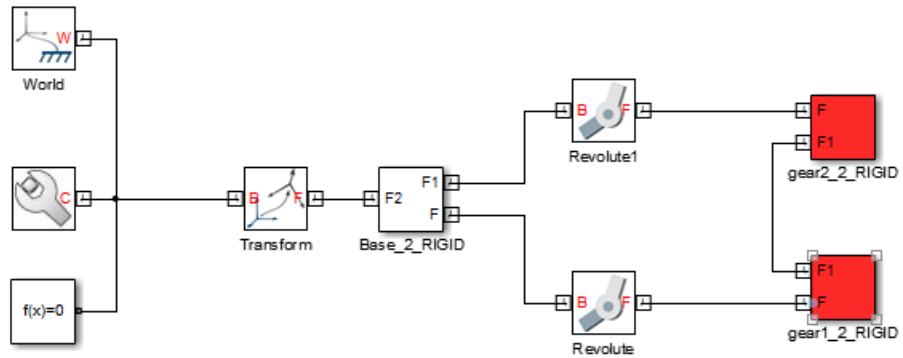
SimMechanics Replaces Unsupported Constraint with Rigid Connection

If a CAD assembly contains one or more unsupported CAD constraints, SimMechanics:

- Issues a warning message at the MATLAB command line. The following figure shows the warning message for a model that contains an unsupported gear constraint.

```
>> smimport('GearAssembly')  
Warning: An unknown constraint exists between gear1_2_RIGID and  
gear2_2_RIGID. A rigid connection has been added between port F1 o  
gear1 2 RIGID and port F1 of gear2 2 RIGID for this constraint.
```

- Replaces the unsupported constraint with a rigid connection. The following figure shows the imported model for an assembly that contains the unsupported gear constraint.



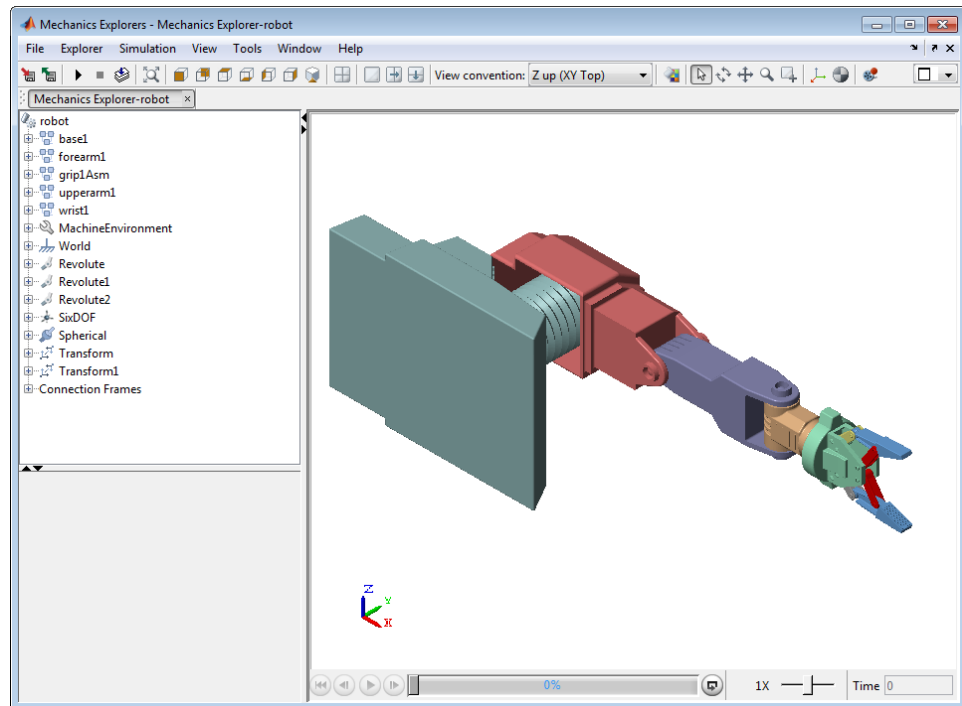
The warning message identifies:

- Name of each block originally connected to an unsupported constraint.
- Label of each frame port used to create the rigid connection that replaces the unsupported constraint.

Use the information provided by the warning message to locate the replacement rigid connection. Then, determine if a SimMechanics joint block can adequately represent the desired degrees of freedom between the two rigid body subsystems. If the answer is yes, add the block between the two rigid body subsystems.

Default View Convention is Z-Axis Up

The Mechanics Explorer visualization utility uses a **Z-axis up** default view convention. The view convention differs from common CAD platforms, which use a **Y-axis up** convention. The difference in view convention can cause imported models to appear sideways.



To restore the orientation of an imported model, change the Mechanics Explorer view convention:

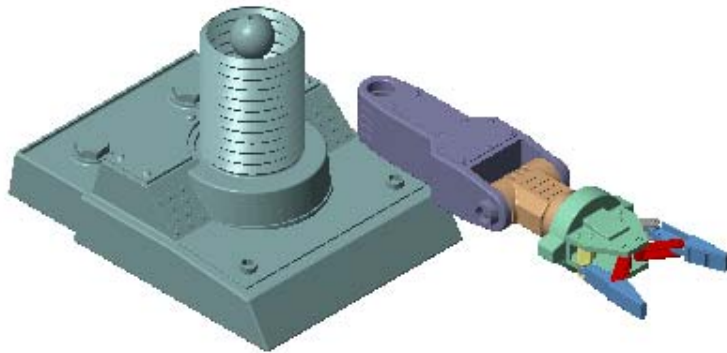
- 1** In the Mechanics Explorer toolbar, click the **View Convention** drop-down menu.
- 2** Select **Y up (ZX Top)**.
- 3** In the toolbar, click the icon for the desired view point. Perform this step even if you wish to retain the current view point.
- 4** Mechanics Explorer displays the model using the new view convention.

Invalid STL Files Cause Import Issues

SimMechanics relies on stereolithographic (STL) files to generate the 3-D surface geometry of a CAD part. In some cases, the STL file of a part may fail to load. Failure to load a file affects visualization of the part, but not model

generation or simulation. During model simulation, Mechanics Explorer displays a 3-D representation of the model with the exception of parts without associated STL files.

The following figure shows an imported robot model that contains an invalid STL file for the upper arm. The model simulates without issue, but without a graphic for the upper arm system.



About STL Files

STL files represent the 3-D surface geometry of a CAD part as a set of 2-D *triangle* areas. To completely describe the triangles, the STL files contain:

- xyz coordinates of the triangle vertices.
- xyz components of the triangle normal vectors.

In the event that an STL file contains a normal vector with zero length, SimMechanics issues a warning message and the STL file does not load.

Deployment

Code Generation

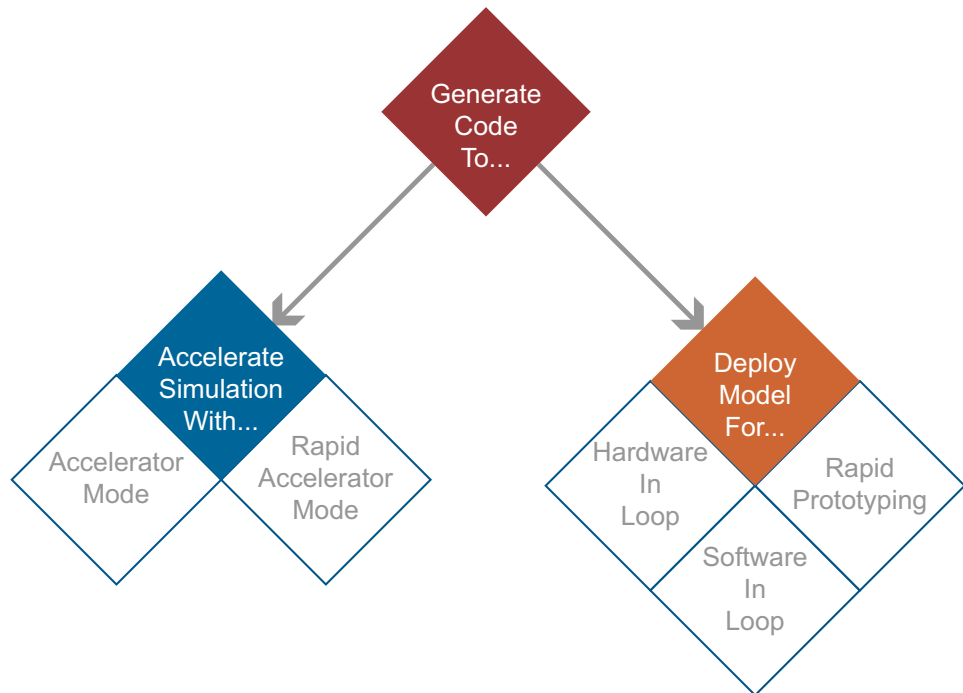
About Code Generation

In this section...

“Simulation Accelerator Modes” on page 8-2

“Model Deployment” on page 8-3

SimMechanics supports code generation with Simulink Coder™. You can generate C/C++ code from a SimMechanics model to accelerate simulation or to deploy a model.



Simulation Accelerator Modes

Simulink can generate C/C++ executable code to shorten simulation time. Two simulation modes generate code:

- Accelerator
- Rapid Accelerator

SimMechanics supports the two accelerator modes. You can access the simulation accelerator modes in the Simulink Editor window for your model. Click **Simulation > Mode**, and select Accelerator or Rapid Accelerator. Accelerator modes do not require additional Simulink code generation products.

Note Simulation accelerator modes do not support model visualization. When you simulate a SimMechanics model in Accelerator or Rapid Accelerator modes, Mechanics Explorer does not open with a 3-D display of your model.

Model Deployment

With Simulink Coder, you can generate standalone C/C++ code for deployment outside the Simulink environment. The code replicates the source SimMechanics model. You can use the stand-alone code for applications that include:

- Hardware-In-Loop (HIL) testing
- Software-In-Loop (SIL) testing
- Rapid prototyping

Note SimMechanics supports, but does not perform, code generation for model deployment. Code generation for model deployment requires the Simulink Coder product.

Code Generation Limitations

Code Generation imposes special considerations on SimMechanics models. This section highlights some special considerations for SimMechanics models. Refer to this section if you run into code generation issues.

Note Code generation for deployment requires Simulink Coder.

Variable-Step Simulink Solver Requires `rsim.tlc` target

Code generation is compatible with fixed- and variable- step solvers. If you select a variable-step solver, you must specify system target file `rsim.tlc`. To specify the `rsim.tlc` system target file, follow these steps:

- 1** In the Simulink Editor window for your model, select **Simulation > Model Configuration Parameters**.
- 2** In the left pane of the **Model Configuration Parameters** dialog box, select **Code Generation**.
- 3** In **System target file**, enter `rsim.tlc`.
- 4** Click **Apply**.
- 5** Click **Build** to generate code for the active model.

Simulink Solver Must Be Continuous

Both fixed- and variable-step solvers can be continuous or discrete. Generating code from a SimMechanics model requires a continuous solver. SimMechanics blocks use continuous time samples, and are incompatible with discrete solvers. If you attempt to generate code with a discrete solver, Simulink Coder issues an error.

If you receive an error stating that SimMechanics does not support a discrete solver, select a continuous Simulink solver. To change the Simulink solver, follow these steps:

- 1** In the Simulink Editor window for your model, select **Simulation > Model Configuration Parameters**.
- 2** In **Solver**, under **Solver options**, click **Solver**.
- 3** In the drop-down menu, select any solver with the exception of discrete (no continuous states).

Accelerator Simulation Modes Do Not Support Visualization

SimMechanics supports Accelerator and Rapid Accelerator simulation modes. Selecting an accelerator mode generates executable code that shortens the time required to run a simulation. However, the simulation produces no visualization output. Mechanics Explorer does not open, and you cannot visualize the model simulation. To restore visualization, select the Normal simulation mode.

If you simulate a model in Accelerator or Rapid Accelerator mode, SimMechanics issues a warning indicating that accelerator modes do not support visualization. To remove the warning, disable visualization with Mechanics Explorer:

- 1** In the Simulink Editor window for your model, select **Simulation > Model Configuration Parameters**.
- 2** In the **Model Configuration Parameters** window, expand the **SimMechanics 2G** node.
- 3** Select **Explorer**.
- 4** Clear the **Open Mechanics Explorer on model update or simulation** check box.

Note Clearing the **Open Mechanics Explorer on model update or simulation** check box disables Mechanics Explorer. When you return to Normal simulation mode, check the box to restore visualization with Mechanics Explorer.

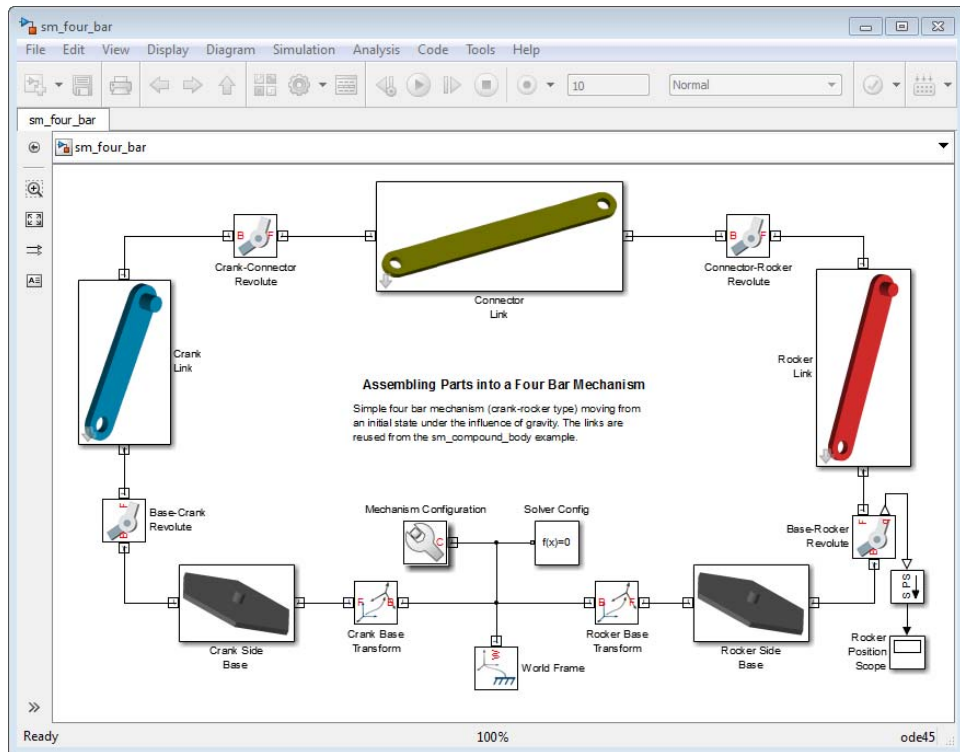
SimMechanics Does Not Support Run-Time Parameters

Model parameters are fixed during code generation. To change model parameters, edit the parameters in SimMechanics and regenerate code for the model. You can only change model parameters in SimMechanics itself.

Configure Four-Bar Model for Code Generation

You can generate code from a SimMechanics model for deployment outside the MATLAB environment. This example shows how to configure a four-bar model for code generation using a variable-step solver with the objective of execution efficiency. The example uses the default Simulink solver ode45 (Dormand-Prince).

The four-bar model is present in your SimMechanics installation. To open the model, at the MATLAB command line type `sm_four_bar`. A new Simulink Editor window opens with the block diagram of the four-bar model.



To configure the model for code generation, follow these steps:

- 1** In the Simulink Editor window for your model, select **Simulation > Model Configuration Parameters**.
- 2** In the **Model Configuration Parameters** dialog box, select **Code Generation**.
- 3** In **Target Selection**, enter `rsim.tlc`.

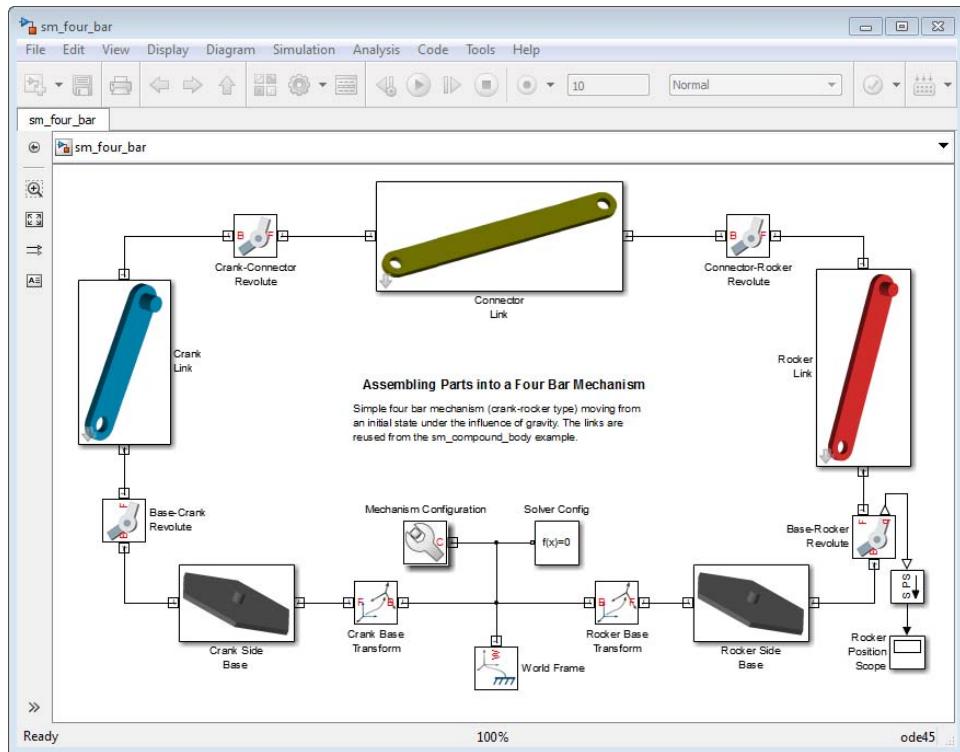
Note You must use the `rsim.tlc` target each time you use a variable-step solver. You can change the solver type in the **Solver** section of the **Model Configuration Parameters** window.

- 4** In **Code Generation Advisor**, select **Execution Efficiency**.
- 5** Click **Apply**.
- 6** To generate C code for your model, click **Build**.

Simulate Four-Bar Model in Rapid Accelerator Mode

You can run a SimMechanics model in Accelerator and Rapid Accelerator modes. When you select an accelerator mode, SimMechanics generates executable code that accelerates the model simulation. This example shows how to configure a four-bar model for Rapid Accelerator simulation mode. The simulation uses the default Simulink solver ode45 (Dormand-Prince).

The four-bar model is present in your SimMechanics installation. To open the model, at the MATLAB command line type `sm_four_bar`. A new Simulink Editor window opens with the block diagram of the four-bar model.



To configure the model for Rapid Acceleration simulation mode, follow these steps:

- 1 In the Simulink Editor window for your model, select **Simulation**.
- 2 In the drop-down menu, select **Mode > Rapid Accelerator**.
- 3 Select **Simulation > Model Configuration Parameters**.
- 4 In **Code Generation**, under **System target file**, enter `rsim.tlc`.

Note You must use the `rsim.tlc` target each time you generate code with a variable-step solver. Both Accelerator and Rapid Accelerator modes generate executable code that requires the `rsim.tlc` target to be used with variable-step solvers.

- 5 Expand the **SimMechanics 2G** node.
- 6 Select **Explorer**.
- 7 Clear the **Open Mechanics Explorer on model update or simulation** check box.

Note Clearing the **Open Mechanics Explorer on model update or simulation** check box disables visualization with Mechanics Explorer. Disabling visualization prevents SimMechanics from issuing a warning message when you simulate a model in Accelerator or Rapid Accelerator mode.

- 8 Press **Ctrl+T** to simulate the model.

Note The Rapid Accelerator mode incurs an initial time cost to generate the executable code. Once the code is generated, the simulation proceeds more rapidly. Rapid Accelerator mode is suggested for large or complex SimMechanics models with long simulation times.

The Rapid Accelerator mode does not support visualization. Mechanics Explorer does not open, and you can not view a dynamic simulation of the

model. All other simulation capabilities remain functional, including graphics and scopes.