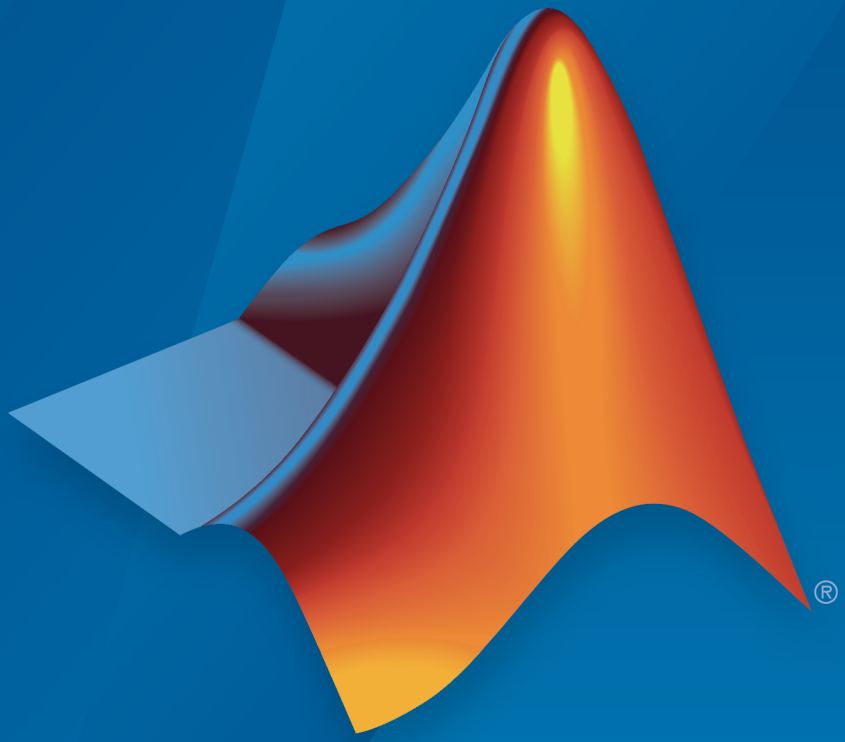
Robotics System Toolbox™

User's Guide



MATLAB® & SIMULINK®

R2016a

MathWorks®

How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

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

Phone: 508-647-7000

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

Robotics System Toolbox™ User Guide

Revision History

March 2015	Online only	New for Version 1.0 (R2015a)
September 2015	Online only	Revised for Version 1.1 (R2015b)
October 2015	Online only	Rereleased for Version 1.0.1 (Release 2015aSP1)
March 2016	Online only	Revised for Version 1.2 (R2016a)

Contents

1 Coordinate System Transformations

2 ROS Network Connection

3 ROS Publishers, Subscribers, and Services

4 ROS Log Files and Transformations

5 ROS Custom Message Support

6 Simulink ROS Concepts

7 Algorithm Design

8 Application Design

9 Code Generation

1

Coordinate System Transformations

Standard Units for Robotics System Toolbox

Robotics System Toolbox™ uses a fixed set of standards for units to ensure consistency across algorithms and applications. Unless specified otherwise, functions and classes in this toolbox represent all values in units based on the International System of Units (SI). The table below summarizes the relevant quantities and their SI derived units.

Quantity	Unit (abbrev.)
Length	meter (m)
Time	second (s)
Angle	radian (rad)
Velocity	meter/second (m/s)
Angular Velocity	radian/second (rad/s)
Acceleration	meter/second^2 (m/s^2)
Angular Acceleration	radian/second^2 (rad/s^2)
Mass	kilogram (kg)
Force	Newton (N)
Torque	Newton-meter (N-m)
Moment of Inertia	kilogram-meter^2 (kg-m^2)

More About

- “Coordinate Transformations in Robotics” on page 1-3

Coordinate Transformations in Robotics

In this section...

In robotics applications, many different coordinate systems can be used to define where robots, sensors, and other objects are located. In general, the location of an object in 3-D space is defined by its position and orientation. There are multiple possible representations for these quantities, some of which are specific to certain applications. Translation and rotation are alternative terms for position and orientation. Robotics System Toolbox supports representations that are commonly used in robotics and allows you to convert between them. You can transform between coordinate systems when you apply these representations to 3-D points. These supported representations are detailed below with brief explanations of their usage and numeric equivalent in MATLAB®. Each representation has an abbreviation for its name. This is used in the naming of arguments and conversion functions that are supported in this toolbox.

At the end of this section, you can find out about the conversion functions that we offer to convert between these representations.

Robotics System Toolbox assumes that positions and orientations are defined in a right-handed Cartesian coordinate system.

Axis-Angle

Abbreviation: axang

A rotation in 3-D space described by a scalar rotation around a fixed axis defined by a vector.

Numeric Representation: 1-by-3 unit vector and a scalar angle combined as a 1-by-4 vector

For example, a rotation of `pi/2` radians around the *y*-axis would be:

```
axang = [0 1 0 pi/2]
```

Euler Angles
Abbreviation: eul

Euler angles are three angles that describe the orientation of a rigid body. Each angle is a scalar rotation around a given coordinate frame axis. The Robotics System Toolbox supports two rotation orders. The `'ZYZ'` axis order is commonly used for robotics applications. We also support the `'ZYX'` axis order which is also denoted as "Roll Pitch Yaw (rpy)." Knowing which axis order you use is important for apply the rotation to points and in converting to other representations.

Numeric Representation: 1-by-3 vector of scalar angles

For example, a rotation around the *y* -axis of pi would be expressed as:

```
eul = [0 pi 0]
```

Note: The axis order is not stored in the transformation, so you must be aware of what rotation order is to be applied.

Homogeneous Transformation Matrix
Abbreviation: tform

A homogeneous transformation matrix combines a translation and rotation into one matrix.

Numeric Representation: 4-by-4 matrix

For example, a rotation of angle α around the *y* -axis and a translation of 4 units along the *y* -axis would be expressed as:

```
tform =
 cos α  0      sin α  0
 0      1      0      4
-sin α  0      cos α  0
 0      0      0      1
```

You should **pre-multiply** your transformation matrix with your homogeneous coordinates, which are represented as a matrix of row vectors (*n*-by-4 matrix of points). For example:

```
points = rand(100,4);
tformPoints = tform*points;
```

Quaternion

Abbreviation: quat

A quaternion is a four-element vector with a scalar rotation and 3-element vector. Quaternions are advantageous because they avoid singularity issues that are inherent in other representations. The first element, *w*, is a scalar to normalize the vector with the three other values, *[x y z]* defining the axis of rotation.

Numeric Representation: 1-by-4 vector

For example, a rotation of `pi/2` around the *y* -axis would be expressed as:

```
quat = [0.7071 0 0.7071 0]
```

Rotation Matrix

Abbreviation: rotm

A rotation matrix describes a rotation in 3-D space. It is a square, orthonormal matrix with a determinant of 1.

Numeric Representation: 3-by-3 matrix

For example, a rotation of **α** degrees around the *x*-axis would be:

```
rotm =

     1     0         0
     0     cos α     -sin α
     0     sin α     cos α
```

You should **pre-multiply** your rotation matrix with your coordinates, which are represented as a matrix of row vectors (*n*-by-3 matrix of points). For example:

```
points = rand(100,3);
rotPoints = rotm*points;
```

Translation Vector

Abbreviation: trvec

A translation vector is represented in 3-D Euclidean space as Cartesian coordinates. It only involves coordinate translation applied equally to all points. There is no rotation involved.

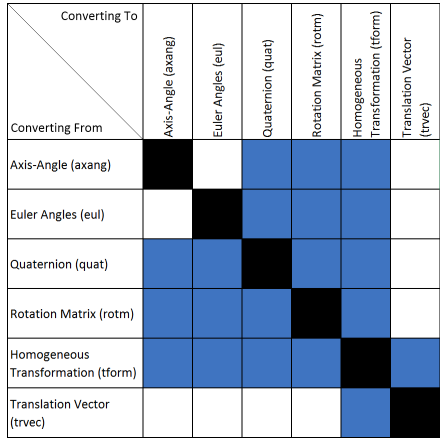
Numeric Representation: 1-by-3 vector

For example, a translation by 3 units along the x -axis and 2.5 units along the z -axis would be expressed as:

```
trvec = [3 0 2.5]
```

Conversion Functions and Transformations

Robotics System Toolbox provides conversion functions for the previously mentioned transformation representations. Not all conversions are supported by a dedicated function. Below is a table showing which conversions are supported (in blue). The abbreviations for the rotation and translation representations are shown as well.



The names of all the conversion functions follow a standard format. They follow the form `alpha2beta` where `alpha` is the abbreviation for what you are converting from and `beta` is what you are converting to as an abbreviation. For example, converting from Euler angles to quaternion would be `eul2quat`.

All the functions expect valid inputs. If you specify invalid inputs, the outputs will be undefined.

There are other conversion functions for converting between radians and degrees, Cartesian and homogeneous coordinates, and for calculating wrapped angle differences. For a full list of conversions, see “Coordinate System Transformations”.

More About

- “Standard Units for Robotics System Toolbox” on page 1-2

2

ROS Network Connection

ROS Network Setup

In this section...

Introduction

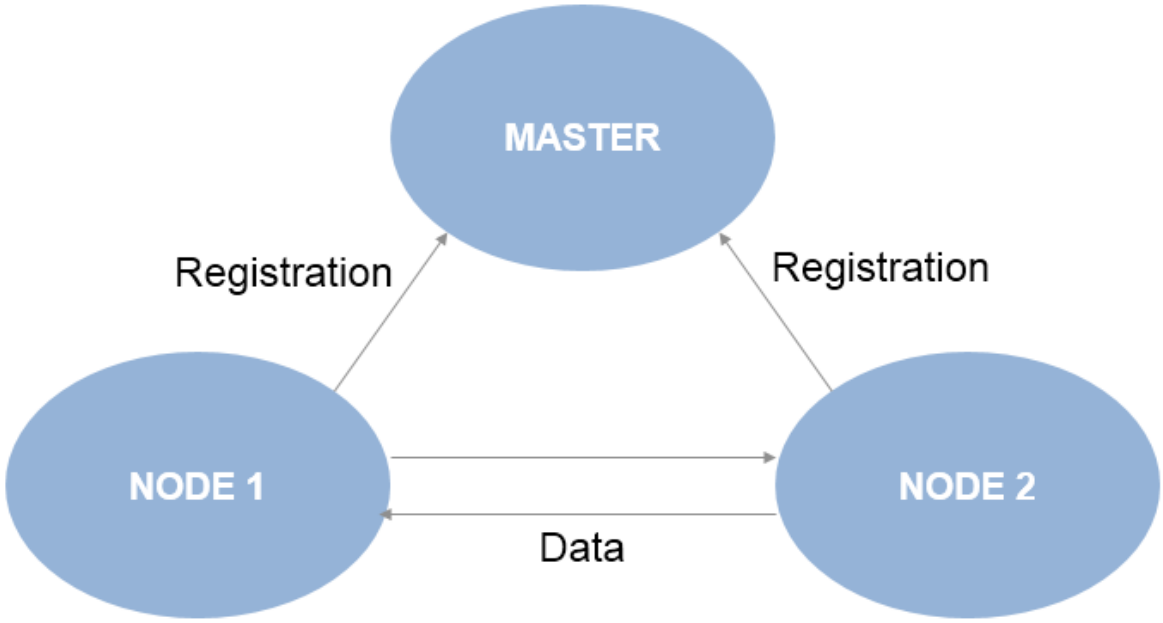
Setting up a ROS network allows for communication between different devices. Different participants or *nodes* all register with a ROS master to share information. The ROS master is unique and each ROS network only has one master. Each node is usually a separate device, although one device can have multiple nodes running. MATLAB acts as one of these nodes when using it to communicate with ROS.

All devices must be connected to the same actual or virtual network for ROS connections to work. You can create a new ROS master in MATLAB, or you can connect to an existing ROS master that is running on a different device. If you connect to an external master, you have to know the IP address or hostname of the device. The initial ROS master connection is done by calling `rosinit`. For more information on setting up and using the ROS network, see "Network Connection and Exploration".

Data communication is achieved by sending messages using entities called publishers, subscribers, and services. Publishers send data via topic names, which subscribers then receive over the network. Services use clients to request information from a server. For more information on sending messages, see "Publishers, Subscribers, and Services"

Network Connection Layout

The ROS network is a collection of nodes that are all connected to the ROS master. The number of nodes can be quite large depending on your application and devices. When nodes get registered with the master, communication with all other nodes becomes possible. Each node registers different publishers, subscribers, and services on the ROS master to send and receive information between nodes. Even though all nodes in the ROS network are registered with the master, data is exchanged directly between nodes. The following figure shows the layout of a ROS network with two ROS nodes. It is important that all nodes have bidirectional connectivity to share data across the network. Verifying these connections is important during setup.



Each node registers its own Node URI with the master. Other participants in the ROS network will use this URI to contact the node. Again, this URI must be reachable by every other node in the ROS network. To create a node in MATLAB, call `rosinit`. If a ROS master is already set up, MATLAB detects it and sets the Node URI appropriately. Otherwise, it creates both a ROS master and node that are connected.

By default, each MATLAB instance has a single "global" node. The node has a randomly-generated name assigned to it for uniqueness. All publishers, subscribers, service clients, and service servers will operate on this global node.

Examples

To better understand how the ROS network is set up in MATLAB, see the following examples:

- "Get Started with ROS"
- "Connect to a ROS Network"

See Also

`rosinit` | `rosnode` | `rostopic`

More About

- "Robot Operating System (ROS)"

3

ROS Publishers, Subscribers, and Services

Built-In Message Support

In this section...

MATLAB has support for a large library of ROS message types. How messages are structured, limitations for ROS messages, and supported ROS data types are described in order to understand how MATLAB works with ROS messages. Also, a full list of built-in message types are shown.

ROS Message Structure

In MATLAB, ROS messages are stored as handle objects. Therefore, all the rules of handle objects apply including copying, modifying and other performance considerations. For more information on handle objects, see “Using Handles”. Each handle points to the object for that specific message, which contains the information relevant to that message type. The message type has a built in structure for the data it contains.

ROS messages are similar to *structure arrays* with how they store the data relevant to that message type. Each message type has a specific set of properties with their corresponding values that are individually stored and accessed. You can specifically point to and modify each property on its own. All messages have the `MessageType` property to view the message type as a string. Also, you can use the `showdetails` function to view the contents of the message.

Here is a sample `'geometry_msgs/Point'` created in MATLAB using `rosmessage`. It contains 3 properties corresponding to a 3-D point in XYZ coordinates.

```
pointMsg = rosmessage('geometry_msgs/Point')
```

```
pointMsg =

  ROS Point message with properties:

    MessageType: 'geometry_msgs/Point'
              X: 0
              Y: 0
```

```
            Z: 0

  Use showdetails to show the contents of the message
```

You can access and modify each property by using the created `pointMsg` handle.

```
pointMsg.Y = 2

pointMsg =

  ROS Point message with properties:

    MessageType: 'geometry_msgs/Point'
              X: 0
              Y: 2
              Z: 0

  Use showdetails to show the contents of the message
```

To explore further the ROS message structure in MATLAB, see “Work with Basic ROS Messages”.

Limitations of ROS Messages in MATLAB

Because ROS messages use structure arrays, this creates a limitation on the validation of certain messages with multiple values. Because each value can be set separately, the message does not validate the properties as a whole entity. For example, a quaternion message contains w, x, y, and z properties, but the message does not enforce that the quaternion as a whole is valid. When modifying properties, you should ensure you are maintaining the rules required for that message.

Message properties can also have a variety of data types. MATLAB uses the rules set by ROS to determine what these data types are. However, if they are to be used in calculations, you might have to cast them to another value. The ROS data types do not convert directly to MATLAB data types, so the next section described the corresponding types.

ROS Data Type Conversions

ROS message types have predetermined properties and data types for the values of those properties. These data types must be mapped to MATLAB data types to be used in MATLAB. This table summarizes how ROS data types are converted to MATLAB data types.

ROS Data Type	Description	MATLAB
bool	Boolean / Unsigned 8-bit integer	`logical`
int8	Signed 8-bit integer	`int8`
uint8	Unsigned 8-bit integer	`uint8`
int16	Signed 16-bit integer	`int16`
uint16	Unsigned 16-bit integer	`uint16`
int32	Signed 32-bit integer	`int32`
uint32	Unsigned 32-bit integer	`uint32`
int64	Signed 64-bit integer	`int64`
uint64	Unsigned 64-bit integer	`uint64`
float32	32-bit IEEE floating point	`single`
float64	64-bit IEEE floating point	`double`
string	ASCII string (utf-8 only)	`char`
time	Seconds and nanoseconds as signed 32-bit integers	`Time` object
duration	Seconds and nanoseconds as signed 32-bit integers	`Duration` object

Supported Message List

Here is an alphabetized list of messages types supported. You can get this list by calling `rosmsg list` in the MATLAB command prompt. When specifying message types, input strings must match the following strings exactly. To enable you to use custom message type that are not listed here, MATLAB also provides a custom message support package. For more information, see `roboticsAddons` to install the support package.

```
'ackermann_msgs/AckermannDrive'
'ackermann_msgs/AckermannDriveStamped'
'actionlib_msgs/GoalID'
'actionlib_msgs/GoalStatus'
'actionlib_msgs/GoalStatusArray'
'adhoc_communication/BroadcastCMgrRobotUpdate'
'adhoc_communication/BroadcastCMgrRobotUpdateRequest'
'adhoc_communication/BroadcastCMgrRobotUpdateResponse'
```

```
'adhoc_communication/BroadcastString'
'adhoc_communication/BroadcastStringRequest'
'adhoc_communication/BroadcastStringResponse'
'adhoc_communication/CMgrDimensions'
'adhoc_communication/CMgrRobotUpdate'
'adhoc_communication/ChangeMCMembership'
'adhoc_communication/ChangeMCMembershipRequest'
'adhoc_communication/ChangeMCMembershipResponse'
'adhoc_communication/ExpAuction'
'adhoc_communication/ExpCluster'
'adhoc_communication/ExpFrontier'
'adhoc_communication/ExpFrontierElement'
'adhoc_communication/GetGroupState'
'adhoc_communication/GetGroupStateRequest'
'adhoc_communication/GetGroupStateResponse'
'adhoc_communication/GetNeighbors'
'adhoc_communication/GetNeighborsRequest'
'adhoc_communication/GetNeighborsResponse'
'adhoc_communication/MmControl'
'adhoc_communication/MmListOfPoints'
'adhoc_communication/MmMapUpdate'
'adhoc_communication/MmPoint'
'adhoc_communication/MmRobotPosition'
'adhoc_communication/RecvString'
'adhoc_communication/SendCMgrRobotUpdate'
'adhoc_communication/SendCMgrRobotUpdateRequest'
'adhoc_communication/SendCMgrRobotUpdateResponse'
'adhoc_communication/SendExpAuction'
'adhoc_communication/SendExpAuctionRequest'
'adhoc_communication/SendExpAuctionResponse'
'adhoc_communication/SendExpCluster'
'adhoc_communication/SendExpClusterRequest'
'adhoc_communication/SendExpClusterResponse'
'adhoc_communication/SendExpFrontier'
'adhoc_communication/SendExpFrontierRequest'
'adhoc_communication/SendExpFrontierResponse'
'adhoc_communication/SendMmControl'
'adhoc_communication/SendMmControlRequest'
'adhoc_communication/SendMmControlResponse'
'adhoc_communication/SendMmMapUpdate'
'adhoc_communication/SendMmMapUpdateRequest'
'adhoc_communication/SendMmMapUpdateResponse'
'adhoc_communication/SendMmPoint'
'adhoc_communication/SendMmPointRequest'
```

```
'adhoc_communication/SendMmPointResponse'
'adhoc_communication/SendMmRobotPosition'
'adhoc_communication/SendMmRobotPositionRequest'
'adhoc_communication/SendMmRobotPositionResponse'
'adhoc_communication/SendOccupancyGrid'
'adhoc_communication/SendOccupancyGridRequest'
'adhoc_communication/SendOccupancyGridResponse'
'adhoc_communication/SendQuaternion'
'adhoc_communication/SendQuaternionRequest'
'adhoc_communication/SendQuaternionResponse'
'adhoc_communication/SendString'
'adhoc_communication/SendStringRequest'
'adhoc_communication/SendStringResponse'
'adhoc_communication/SendTwist'
'adhoc_communication/SendTwistRequest'
'adhoc_communication/SendTwistResponse'
'adhoc_communication/ShutDown'
'adhoc_communication/ShutDownRequest'
'adhoc_communication/ShutDownResponse'
'app_manager/App'
'app_manager/AppInstallationState'
'app_manager/AppList'
'app_manager/AppStatus'
'app_manager/ClientApp'
'app_manager/ExchangeApp'
'app_manager/GetAppDetails'
'app_manager/GetAppDetailsRequest'
'app_manager/GetAppDetailsResponse'
'app_manager/GetInstallationState'
'app_manager/GetInstallationStateRequest'
'app_manager/GetInstallationStateResponse'
'app_manager/Icon'
'app_manager/InstallApp'
'app_manager/InstallAppRequest'
'app_manager/InstallAppResponse'
'app_manager/KeyValue'
'app_manager/ListApps'
'app_manager/ListAppsRequest'
'app_manager/ListAppsResponse'
'app_manager/StartApp'
'app_manager/StartAppRequest'
'app_manager/StartAppResponse'
'app_manager/StatusCodes'
'app_manager/StopApp'
```

```
'app_manager/StopAppRequest'
'app_manager/StopAppResponse'
'app_manager/UninstallApp'
'app_manager/UninstallAppRequest'
'app_manager/UninstallAppResponse'
'applanix_msgs/Ack'
'applanix_msgs/AidingSensorIntegrationControl'
'applanix_msgs/AidingSensorParams'
'applanix_msgs/BaseGNSSModemStatus'
'applanix_msgs/BaseGNSSSetup'
'applanix_msgs/BinaryMessageSelect'
'applanix_msgs/COMPortMessages'
'applanix_msgs/COMPortParams'
'applanix_msgs/COMPortSetup'
'applanix_msgs/CalibratedInstallationParameters'
'applanix_msgs/CommonFooter'
'applanix_msgs/CommonHeader'
'applanix_msgs/DGPSSourceControl'
'applanix_msgs/DMIData'
'applanix_msgs/Event'
'applanix_msgs/EventSetup'
'applanix_msgs/GAMS'
'applanix_msgs/GAMSCalibrationControl'
'applanix_msgs/GAMSParams'
'applanix_msgs/GNSSAuxStatus'
'applanix_msgs/GNSSChannelStatus'
'applanix_msgs/GNSSControl'
'applanix_msgs/GNSSDGPSChannelStatus'
'applanix_msgs/GNSSDGPSStation'
'applanix_msgs/GNSSDGPSStationDatabase'
'applanix_msgs/GNSSDGPSStatus'
'applanix_msgs/GNSSSetup'
'applanix_msgs/GNSSStatus'
'applanix_msgs/GeneralParams'
'applanix_msgs/GeneralStatus'
'applanix_msgs/Groups'
'applanix_msgs/IINSolutionStatus'
'applanix_msgs/IMUData'
'applanix_msgs/IPAddress'
'applanix_msgs/InstallationCalibrationControl'
'applanix_msgs/IntegrationDiagnosticsControl'
'applanix_msgs/LoggingControl'
'applanix_msgs/LoggingStatus'
'applanix_msgs/NMEAMessageSelect'
```

```
'applanix_msgs/NavModeControl'
'applanix_msgs/NavigationPerformance'
'applanix_msgs/NavigationSolution'
'applanix_msgs/OutputGroup'
'applanix_msgs/PPSStatus'
'applanix_msgs/PortControl'
'applanix_msgs/PreciseGravitySpecs'
'applanix_msgs/ProgramControl'
'applanix_msgs/RawDMI'
'applanix_msgs/RawData'
'applanix_msgs/RawPPS'
'applanix_msgs/SaveRestoreControl'
'applanix_msgs/TimeDistance'
'applanix_msgs/TimeSyncControl'
'applanix_msgs/UserAccuracySpecs'
'applanix_msgs/UserTimeStatus'
'applanix_msgs/Version'
'ar_track_alvar/AlvarMarker'
'ar_track_alvar/AlvarMarkers'
'arbotix_msgs/Analog'
'arbotix_msgs/Digital'
'arbotix_msgs/Enable'
'arbotix_msgs/EnableRequest'
'arbotix_msgs/EnableResponse'
'arbotix_msgs/Relax'
'arbotix_msgs/RelaxRequest'
'arbotix_msgs/RelaxResponse'
'arbotix_msgs/SetSpeed'
'arbotix_msgs/SetSpeedRequest'
'arbotix_msgs/SetSpeedResponse'
'arbotix_msgs/SetupChannel'
'arbotix_msgs/SetupChannelRequest'
'arbotix_msgs/SetupChannelResponse'
'ardrone_autonomy/CamSelect'
'ardrone_autonomy/CamSelectRequest'
'ardrone_autonomy/CamSelectResponse'
'ardrone_autonomy/FlightAnim'
'ardrone_autonomy/FlightAnimRequest'
'ardrone_autonomy/FlightAnimResponse'
'ardrone_autonomy/LedAnim'
'ardrone_autonomy/LedAnimRequest'
'ardrone_autonomy/LedAnimResponse'
'ardrone_autonomy/Navdata'
'ardrone_autonomy/RecordEnable'
```

```
'ardrone_autonomy/RecordEnableRequest'
'ardrone_autonomy/RecordEnableResponse'
'ardrone_autonomy/matrix33'
'ardrone_autonomy/navdata_adc_data_frame'
'ardrone_autonomy/navdata_altitude'
'ardrone_autonomy/navdata_demo'
'ardrone_autonomy/navdata_euler_angles'
'ardrone_autonomy/navdata_games'
'ardrone_autonomy/navdata_gyros_offsets'
'ardrone_autonomy/navdata_hdvideo_stream'
'ardrone_autonomy/navdata_kalman_pressure'
'ardrone_autonomy/navdata_magneto'
'ardrone_autonomy/navdata_phys_measures'
'ardrone_autonomy/navdata_pressure_raw'
'ardrone_autonomy/navdata_pwm'
'ardrone_autonomy/navdata_raw_measures'
'ardrone_autonomy/navdata_rc_references'
'ardrone_autonomy/navdata_references'
'ardrone_autonomy/navdata_time'
'ardrone_autonomy/navdata_trackers_send'
'ardrone_autonomy/navdata_trims'
'ardrone_autonomy/navdata_video_stream'
'ardrone_autonomy/navdata_vision'
'ardrone_autonomy/navdata_vision_detect'
'ardrone_autonomy/navdata_vision_of'
'ardrone_autonomy/navdata_vision_perf'
'ardrone_autonomy/navdata_vision_raw'
'ardrone_autonomy/navdata_watchdog'
'ardrone_autonomy/navdata_wifi'
'ardrone_autonomy/navdata_wind_speed'
'ardrone_autonomy/navdata_zimmu_3000'
'ardrone_autonomy/vector21'
'ardrone_autonomy/vector31'
'audio_common_msgs/AudioData'
'axis_camera/Axis'
'base_local_planner/Position2DInt'
'baxter_core_msgs/AnalogIOState'
'baxter_core_msgs/AnalogIOStates'
'baxter_core_msgs/AnalogOutputCommand'
'baxter_core_msgs/AssemblyState'
'baxter_core_msgs/AssemblyStates'
'baxter_core_msgs/CameraControl'
'baxter_core_msgs/CameraSettings'
'baxter_core_msgs/CloseCamera'
```

```
'baxter_core_msgs/CloseCameraRequest'
'baxter_core_msgs/CloseCameraResponse'
'baxter_core_msgs/CollisionAvoidanceState'
'baxter_core_msgs/CollisionDetectionState'
'baxter_core_msgs/DigitalIOState'
'baxter_core_msgs/DigitalIOStates'
'baxter_core_msgs/DigitalOutputCommand'
'baxter_core_msgs/EndEffectorCommand'
'baxter_core_msgs/EndEffectorProperties'
'baxter_core_msgs/EndEffectorState'
'baxter_core_msgs/EndpointState'
'baxter_core_msgs/EndpointStates'
'baxter_core_msgs/HeadPanCommand'
'baxter_core_msgs/HeadState'
'baxter_core_msgs/ITBState'
'baxter_core_msgs/ITBStates'
'baxter_core_msgs/JointCommand'
'baxter_core_msgs/ListCameras'
'baxter_core_msgs/ListCamerasRequest'
'baxter_core_msgs/ListCamerasResponse'
'baxter_core_msgs/NavigatorState'
'baxter_core_msgs/NavigatorStates'
'baxter_core_msgs/OpenCamera'
'baxter_core_msgs/OpenCameraRequest'
'baxter_core_msgs/OpenCameraResponse'
'baxter_core_msgs/RobustControllerStatus'
'baxter_core_msgs/SEAJointState'
'baxter_core_msgs/SolvePositionIK'
'baxter_core_msgs/SolvePositionIKRequest'
'baxter_core_msgs/SolvePositionIKResponse'
'baxter_maintenance_msgs/CalibrateArmData'
'baxter_maintenance_msgs/CalibrateArmEnable'
'baxter_maintenance_msgs/TareData'
'baxter_maintenance_msgs/TareEnable'
'baxter_maintenance_msgs/UpdateSource'
'baxter_maintenance_msgs/UpdateSources'
'baxter_maintenance_msgs/UpdateStatus'
'bayesian_belief_networks/Observation'
'bayesian_belief_networks/Query'
'bayesian_belief_networks/QueryRequest'
'bayesian_belief_networks/QueryResponse'
'bayesian_belief_networks/Result'
'blob/Blob'
'bond/Constants'
```

```
'bond/Status'
'brics_actuator/CartesianPose'
'brics_actuator/CartesianTwist'
'brics_actuator/CartesianVector'
'brics_actuator/CartesianWrench'
'brics_actuator/JointAccelerations'
'brics_actuator/JointConstraint'
'brics_actuator/JointImpedances'
'brics_actuator/JointPositions'
'brics_actuator/JointTorques'
'brics_actuator/JointValue'
'brics_actuator/JointVelocities'
'brics_actuator/Poison'
'bride_tutorials/Dummy'
'bwi_planning/CostLearnerInterface'
'bwi_planning/CostLearnerInterfaceRequest'
'bwi_planning/CostLearnerInterfaceResponse'
'bwi_planning_common/PlannerAtom'
'bwi_planning_common/PlannerInterface'
'bwi_planning_common/PlannerInterfaceRequest'
'bwi_planning_common/PlannerInterfaceResponse'
'calibration_msgs/CalibrationPattern'
'calibration_msgs/CameraMeasurement'
'calibration_msgs/ChainMeasurement'
'calibration_msgs/DenseLaserObjectFeatures'
'calibration_msgs/DenseLaserPoint'
'calibration_msgs/DenseLaserSnapshot'
'calibration_msgs/Interval'
'calibration_msgs/IntervalStamped'
'calibration_msgs/IntervalStatus'
'calibration_msgs/JointStateCalibrationPattern'
'calibration_msgs/LaserMeasurement'
'calibration_msgs/RobotMeasurement'
'capabilities/Capability'
'capabilities/CapabilityEvent'
'capabilities/CapabilitySpec'
'capabilities/EstablishBond'
'capabilities/EstablishBondRequest'
'capabilities/EstablishBondResponse'
'capabilities/FreeCapability'
'capabilities/FreeCapabilityRequest'
'capabilities/FreeCapabilityResponse'
'capabilities/GetCapabilitySpec'
'capabilities/GetCapabilitySpecRequest'
```

```
'capabilities/GetCapabilitySpecResponse'
'capabilities/GetCapabilitySpecs'
'capabilities/GetCapabilitySpecsRequest'
'capabilities/GetCapabilitySpecsResponse'
'capabilities/GetInterfaces'
'capabilities/GetInterfacesRequest'
'capabilities/GetInterfacesResponse'
'capabilities/GetNodeletManagerName'
'capabilities/GetNodeletManagerNameRequest'
'capabilities/GetNodeletManagerNameResponse'
'capabilities/GetProviders'
'capabilities/GetProvidersRequest'
'capabilities/GetProvidersResponse'
'capabilities/GetRemappings'
'capabilities/GetRemappingsRequest'
'capabilities/GetRemappingsResponse'
'capabilities/GetRunningCapabilities'
'capabilities/GetRunningCapabilitiesRequest'
'capabilities/GetRunningCapabilitiesResponse'
'capabilities/GetSemanticInterfaces'
'capabilities/GetSemanticInterfacesRequest'
'capabilities/GetSemanticInterfacesResponse'
'capabilities/Remapping'
'capabilities/RunningCapability'
'capabilities/StartCapability'
'capabilities/StartCapabilityRequest'
'capabilities/StartCapabilityResponse'
'capabilities/StopCapability'
'capabilities/StopCapabilityRequest'
'capabilities/StopCapabilityResponse'
'capabilities/UseCapability'
'capabilities/UseCapabilityRequest'
'capabilities/UseCapabilityResponse'
'clearpath_base/AckermannSetpt'
'clearpath_base/ClearpathRobot'
'clearpath_base/DifferentialControl'
'clearpath_base/DifferentialOutput'
'clearpath_base/DifferentialSpeed'
'clearpath_base/Distance'
'clearpath_base/DistanceTiming'
'clearpath_base/Encoder'
'clearpath_base/Encoders'
'clearpath_base/FirmwareInfo'
'clearpath_base/PlatformInfo'
```

```
'clearpath_base/PlatformName'
'clearpath_base/PowerSource'
'clearpath_base/PowerStatus'
'clearpath_base/ProcessorStatus'
'clearpath_base/RawEncoders'
'clearpath_base/SafetyStatus'
'clearpath_base/SystemStatus'
'clearpath_base/TurnSetpt'
'clearpath_base/VelocitySetpt'
'cmvision/Blob'
'cmvision/Blobs'
'cob_base_drive_chain/ElmoRecorderConfig'
'cob_base_drive_chain/ElmoRecorderConfigRequest'
'cob_base_drive_chain/ElmoRecorderConfigResponse'
'cob_base_drive_chain/ElmoRecorderReadout'
'cob_base_drive_chain/ElmoRecorderReadoutRequest'
'cob_base_drive_chain/ElmoRecorderReadoutResponse'
'cob_camera_sensors/AcquireCalibrationImages'
'cob_camera_sensors/AcquireCalibrationImagesRequest'
'cob_camera_sensors/AcquireCalibrationImagesResponse'
'cob_camera_sensors/GetTOFImages'
'cob_camera_sensors/GetTOFImagesRequest'
'cob_camera_sensors/GetTOFImagesResponse'
'cob_footprint_observer/GetFootprint'
'cob_footprint_observer/GetFootprintRequest'
'cob_footprint_observer/GetFootprintResponse'
'cob_kinematics/GetPositionIKExtended'
'cob_kinematics/GetPositionIKExtendedRequest'
'cob_kinematics/GetPositionIKExtendedResponse'
'cob_light/LightMode'
'cob_light/SetLightMode'
'cob_light/SetLightModeRequest'
'cob_light/SetLightModeResponse'
'cob_object_detection_msgs/AcquireObjectImage'
'cob_object_detection_msgs/AcquireObjectImageRequest'
'cob_object_detection_msgs/AcquireObjectImageResponse'
'cob_object_detection_msgs/BaTestEnvironment'
'cob_object_detection_msgs/BaTestEnvironmentRequest'
'cob_object_detection_msgs/BaTestEnvironmentResponse'
'cob_object_detection_msgs/BagTrainObject'
'cob_object_detection_msgs/BagTrainObjectRequest'
'cob_object_detection_msgs/BagTrainObjectResponse'
'cob_object_detection_msgs/DetectObjects'
'cob_object_detection_msgs/DetectObjectsRequest'
```

```
'cob_object_detection_msgs/DetectObjectsResponse'
'cob_object_detection_msgs/Detection'
'cob_object_detection_msgs/DetectionArray'
'cob_object_detection_msgs/Mask'
'cob_object_detection_msgs/MaskArray'
'cob_object_detection_msgs/PoseRT'
'cob_object_detection_msgs/Rect'
'cob_object_detection_msgs/RectArray'
'cob_object_detection_msgs/SaveRecordedObject'
'cob_object_detection_msgs/SaveRecordedObjectRequest'
'cob_object_detection_msgs/SaveRecordedObjectResponse'
'cob_object_detection_msgs/StartObjectRecording'
'cob_object_detection_msgs/StartObjectRecordingRequest'
'cob_object_detection_msgs/StartObjectRecordingResponse'
'cob_object_detection_msgs/StopObjectRecording'
'cob_object_detection_msgs/StopObjectRecordingRequest'
'cob_object_detection_msgs/StopObjectRecordingResponse'
'cob_object_detection_msgs/TrainObject'
'cob_object_detection_msgs/TrainObjectRequest'
'cob_object_detection_msgs/TrainObjectResponse'
'cob_perception_msgs/PointCloud2Array'
'cob_phidgets/AnalogSensor'
'cob_phidgets/DigitalSensor'
'cob_phidgets/SetDataRate'
'cob_phidgets/SetDataRateRequest'
'cob_phidgets/SetDataRateResponse'
'cob_phidgets/SetDigitalSensor'
'cob_phidgets/SetDigitalSensorRequest'
'cob_phidgets/SetDigitalSensorResponse'
'cob_phidgets/SetTriggerValue'
'cob_phidgets/SetTriggerValueRequest'
'cob_phidgets/SetTriggerValueResponse'
'cob_relayboard/EmergencyStopState'
'cob_script_server/ScriptState'
'cob_sound/SayText'
'cob_sound/SayTextRequest'
'cob_sound/SayTextResponse'
'cob_srvs/GetPoseStampedTransformed'
'cob_srvs/GetPoseStampedTransformedRequest'
'cob_srvs/GetPoseStampedTransformedResponse'
'cob_srvs/SetDefaultVel'
'cob_srvs/SetDefaultVelRequest'
'cob_srvs/SetDefaultVelResponse'
'cob_srvs/SetFloat'
```

```
'cob_srvs/SetFloatRequest'
'cob_srvs/SetFloatResponse'
'cob_srvs/SetInt'
'cob_srvs/SetIntRequest'
'cob_srvs/SetIntResponse'
'cob_srvs/SetJointStiffness'
'cob_srvs/SetJointStiffnessRequest'
'cob_srvs/SetJointStiffnessResponse'
'cob_srvs/SetJointTrajectory'
'cob_srvs/SetJointTrajectoryRequest'
'cob_srvs/SetJointTrajectoryResponse'
'cob_srvs/SetMaxVel'
'cob_srvs/SetMaxVelRequest'
'cob_srvs/SetMaxVelResponse'
'cob_srvs/SetOperationMode'
'cob_srvs/SetOperationModeRequest'
'cob_srvs/SetOperationModeResponse'
'cob_srvs/SetString'
'cob_srvs/SetStringRequest'
'cob_srvs/SetStringResponse'
'cob_srvs/Trigger'
'cob_srvs/TriggerRequest'
'cob_srvs/TriggerResponse'
'cob_trajectory_controller/SetFloat'
'cob_trajectory_controller/SetFloatRequest'
'cob_trajectory_controller/SetFloatResponse'
'concert_msgs/ClientList'
'concert_msgs/ClientListRequest'
'concert_msgs/ClientListResponse'
'concert_msgs/ConcertClient'
'concert_msgs/ConcertClientConfiguration'
'concert_msgs/ConcertClients'
'concert_msgs/Constants'
'concert_msgs/Implementation'
'concert_msgs/Invite'
'concert_msgs/InviteRequest'
'concert_msgs/InviteResponse'
'concert_msgs/LinkConnection'
'concert_msgs/LinkEdge'
'concert_msgs/LinkGraph'
'concert_msgs/LinkNode'
'concert_msgs/StartSolution'
'concert_msgs/StartSolutionRequest'
'concert_msgs/StartSolutionResponse'
```

```
'concert_msgs/StopSolution'
'concert_msgs/StopSolutionRequest'
'concert_msgs/StopSolutionResponse'
'control_msgs/GripperCommand'
'control_msgs/JointControllerState'
'control_msgs/JointTolerance'
'control_msgs/JointTrajectoryControllerState'
'control_msgs/QueryCalibrationState'
'control_msgs/QueryCalibrationStateRequest'
'control_msgs/QueryCalibrationStateResponse'
'control_msgs/QueryTrajectoryState'
'control_msgs/QueryTrajectoryStateRequest'
'control_msgs/QueryTrajectoryStateResponse'
'control_toolbox/SetPidGains'
'control_toolbox/SetPidGainsRequest'
'control_toolbox/SetPidGainsResponse'
'controller_manager_msgs/ControllerState'
'controller_manager_msgs/ControllerStatistics'
'controller_manager_msgs/ControllersStatistics'
'controller_manager_msgs/ListControllerTypes'
'controller_manager_msgs/ListControllerTypesRequest'
'controller_manager_msgs/ListControllerTypesResponse'
'controller_manager_msgs/ListControllers'
'controller_manager_msgs/ListControllersRequest'
'controller_manager_msgs/ListControllersResponse'
'controller_manager_msgs/LoadController'
'controller_manager_msgs/LoadControllerRequest'
'controller_manager_msgs/LoadControllerResponse'
'controller_manager_msgs/ReloadControllerLibraries'
'controller_manager_msgs/ReloadControllerLibrariesRequest'
'controller_manager_msgs/ReloadControllerLibrariesResponse'
'controller_manager_msgs/SwitchController'
'controller_manager_msgs/SwitchControllerRequest'
'controller_manager_msgs/SwitchControllerResponse'
'controller_manager_msgs/UnloadController'
'controller_manager_msgs/UnloadControllerRequest'
'controller_manager_msgs/UnloadControllerResponse'
'costmap_2d/VoxelGrid'
'create_node/BatteryState'
'create_node/Drive'
'create_node/RawTurtlebotSensorState'
'create_node/RoombaSensorState'
'create_node/SetDigitalOutputs'
'create_node/SetDigitalOutputsRequest'
```

```
'create_node/SetDigitalOutputsResponse'
'create_node/SetTurtlebotMode'
'create_node/SetTurtlebotModeRequest'
'create_node/SetTurtlebotModeResponse'
'create_node/Turtle'
'create_node/TurtlebotSensorState'
'data_vis_msgs/DataVis'
'data_vis_msgs/ValueList'
'designator_integration_msgs/Designator'
'designator_integration_msgs/DesignatorCommunication'
'designator_integration_msgs/DesignatorCommunicationRequest'
'designator_integration_msgs/DesignatorCommunicationResponse'
'designator_integration_msgs/DesignatorRequest'
'designator_integration_msgs/DesignatorResponse'
'designator_integration_msgs/KeyValuePair'
'diagnostic_msgs/DiagnosticArray'
'diagnostic_msgs/DiagnosticStatus'
'diagnostic_msgs/KeyValue'
'diagnostic_msgs/SelfTest'
'diagnostic_msgs/SelfTestRequest'
'diagnostic_msgs/SelfTestResponse'
'dna_extraction_msgs/PerceiveChemLabTool'
'dna_extraction_msgs/PerceiveChemLabToolRequest'
'dna_extraction_msgs/PerceiveChemLabToolResponse'
'driver_base/ConfigString'
'driver_base/ConfigValue'
'driver_base/SensorLevels'
'dynamic_reconfigure/BoolParameter'
'dynamic_reconfigure/Config'
'dynamic_reconfigure/ConfigDescription'
'dynamic_reconfigure/DoubleParameter'
'dynamic_reconfigure/Group'
'dynamic_reconfigure/GroupState'
'dynamic_reconfigure/IntParameter'
'dynamic_reconfigure/ParamDescription'
'dynamic_reconfigure/Reconfigure'
'dynamic_reconfigure/ReconfigureRequest'
'dynamic_reconfigure/ReconfigureResponse'
'dynamic_reconfigure/SensorLevels'
'dynamic_reconfigure/StrParameter'
'dynamic_tf_publisher/AssocTF'
'dynamic_tf_publisher/AssocTFRequest'
'dynamic_tf_publisher/AssocTFResponse'
'dynamic_tf_publisher/DeleteTF'
```

```
'dynamic_tf_publisher/DeleteTFRequest'
'dynamic_tf_publisher/DeleteTFResponse'
'dynamic_tf_publisher/DissocTF'
'dynamic_tf_publisher/DissocTFRequest'
'dynamic_tf_publisher/DissocTFResponse'
'dynamic_tf_publisher/SetDynamicTF'
'dynamic_tf_publisher/SetDynamicTFRequest'
'dynamic_tf_publisher/SetDynamicTFResponse'
'dynamixel_controllers/RestartController'
'dynamixel_controllers/RestartControllerRequest'
'dynamixel_controllers/RestartControllerResponse'
'dynamixel_controllers/SetComplianceMargin'
'dynamixel_controllers/SetComplianceMarginRequest'
'dynamixel_controllers/SetComplianceMarginResponse'
'dynamixel_controllers/SetCompliancePunch'
'dynamixel_controllers/SetCompliancePunchRequest'
'dynamixel_controllers/SetCompliancePunchResponse'
'dynamixel_controllers/SetComplianceSlope'
'dynamixel_controllers/SetComplianceSlopeRequest'
'dynamixel_controllers/SetComplianceSlopeResponse'
'dynamixel_controllers/SetSpeed'
'dynamixel_controllers/SetSpeedRequest'
'dynamixel_controllers/SetSpeedResponse'
'dynamixel_controllers/SetTorqueLimit'
'dynamixel_controllers/SetTorqueLimitRequest'
'dynamixel_controllers/SetTorqueLimitResponse'
'dynamixel_controllers/StartController'
'dynamixel_controllers/StartControllerRequest'
'dynamixel_controllers/StartControllerResponse'
'dynamixel_controllers/StopController'
'dynamixel_controllers/StopControllerRequest'
'dynamixel_controllers/StopControllerResponse'
'dynamixel_controllers/TorqueEnable'
'dynamixel_controllers/TorqueEnableRequest'
'dynamixel_controllers/TorqueEnableResponse'
'dynamixel_msgs/JointState'
'dynamixel_msgs/MotorState'
'dynamixel_msgs/MotorStateList'
'epos_driver/EPOSState'
'epos_driver/MoveCycle'
'epos_driver/MoveCycleRequest'
'epos_driver/MoveCycleResponse'
'epos_driver/MoveTo'
'epos_driver/MoveToRequest'
```

```
'epos_driver/MoveToResponse'
'ethercat_hardware/ActuatorInfo'
'ethercat_hardware/BoardInfo'
'ethercat_hardware/MotorTemperature'
'ethercat_hardware/MotorTrace'
'ethercat_hardware/MotorTraceSample'
'ethercat_hardware/RawFTData'
'ethercat_hardware/RawFTDataSample'
'ethercat_hardware/SoftProcessorFirmwareRead'
'ethercat_hardware/SoftProcessorFirmwareReadRequest'
'ethercat_hardware/SoftProcessorFirmwareReadResponse'
'ethercat_hardware/SoftProcessorFirmwareWrite'
'ethercat_hardware/SoftProcessorFirmwareWriteRequest'
'ethercat_hardware/SoftProcessorFirmwareWriteResponse'
'ethercat_hardware/SoftProcessorReset'
'ethercat_hardware/SoftProcessorResetRequest'
'ethercat_hardware/SoftProcessorResetResponse'
'ethercat_trigger_controllers/MultiWaveform'
'ethercat_trigger_controllers/MultiWaveformTransition'
'ethercat_trigger_controllers/SetMultiWaveform'
'ethercat_trigger_controllers/SetMultiWaveformRequest'
'ethercat_trigger_controllers/SetMultiWaveformResponse'
'ethercat_trigger_controllers/SetWaveform'
'ethercat_trigger_controllers/SetWaveformRequest'
'ethercat_trigger_controllers/SetWaveformResponse'
'ethzasl_icp_mapper/CorrectPose'
'ethzasl_icp_mapper/CorrectPoseRequest'
'ethzasl_icp_mapper/CorrectPoseResponse'
'ethzasl_icp_mapper/GetBoundedMap'
'ethzasl_icp_mapper/GetBoundedMapRequest'
'ethzasl_icp_mapper/GetBoundedMapResponse'
'ethzasl_icp_mapper/GetMode'
'ethzasl_icp_mapper/GetModeRequest'
'ethzasl_icp_mapper/GetModeResponse'
'ethzasl_icp_mapper/LoadMap'
'ethzasl_icp_mapper/LoadMapRequest'
'ethzasl_icp_mapper/LoadMapResponse'
'ethzasl_icp_mapper/MatchClouds'
'ethzasl_icp_mapper/MatchCloudsRequest'
'ethzasl_icp_mapper/MatchCloudsResponse'
'ethzasl_icp_mapper/SetMode'
'ethzasl_icp_mapper/SetModeRequest'
'ethzasl_icp_mapper/SetModeResponse'
'explorer/Frontier'
```

```
'fingertip_pressure/PressureInfo'
'fingertip_pressure/PressureInfoElement'
'frontier_exploration/Frontier'
'frontier_exploration/GetNextFrontier'
'frontier_exploration/GetNextFrontierRequest'
'frontier_exploration/GetNextFrontierResponse'
'frontier_exploration/UpdateBoundaryPolygon'
'frontier_exploration/UpdateBoundaryPolygonRequest'
'frontier_exploration/UpdateBoundaryPolygonResponse'
'gateway_msgs/Advertise'
'gateway_msgs/AdvertiseAll'
'gateway_msgs/AdvertiseAllRequest'
'gateway_msgs/AdvertiseAllResponse'
'gateway_msgs/AdvertiseRequest'
'gateway_msgs/AdvertiseResponse'
'gateway_msgs/ConnectHub'
'gateway_msgs/ConnectHubRequest'
'gateway_msgs/ConnectHubResponse'
'gateway_msgs/ConnectionType'
'gateway_msgs/ErrorCodes'
'gateway_msgs/GatewayInfo'
'gateway_msgs/Remote'
'gateway_msgs/RemoteAll'
'gateway_msgs/RemoteAllRequest'
'gateway_msgs/RemoteAllResponse'
'gateway_msgs/RemoteGateway'
'gateway_msgs/RemoteGatewayInfo'
'gateway_msgs/RemoteGatewayInfoRequest'
'gateway_msgs/RemoteGatewayInfoResponse'
'gateway_msgs/RemoteRequest'
'gateway_msgs/RemoteResponse'
'gateway_msgs/RemoteRule'
'gateway_msgs/Rule'
'gateway_msgs/SetWatcherPeriod'
'gateway_msgs/SetWatcherPeriodRequest'
'gateway_msgs/SetWatcherPeriodResponse'
'gazebo_msgs/ApplyBodyWrench'
'gazebo_msgs/ApplyBodyWrenchRequest'
'gazebo_msgs/ApplyBodyWrenchResponse'
'gazebo_msgs/ApplyJointEffort'
'gazebo_msgs/ApplyJointEffortRequest'
'gazebo_msgs/ApplyJointEffortResponse'
'gazebo_msgs/BodyRequest'
'gazebo_msgs/BodyRequestRequest'
```

```
'gazebo_msgs/BodyRequestResponse'
'gazebo_msgs/ContactState'
'gazebo_msgs/ContactsState'
'gazebo_msgs/DeleteModel'
'gazebo_msgs/DeleteModelRequest'
'gazebo_msgs/DeleteModelResponse'
'gazebo_msgs/GetJointProperties'
'gazebo_msgs/GetJointPropertiesRequest'
'gazebo_msgs/GetJointPropertiesResponse'
'gazebo_msgs/GetLinkProperties'
'gazebo_msgs/GetLinkPropertiesRequest'
'gazebo_msgs/GetLinkPropertiesResponse'
'gazebo_msgs/GetLinkState'
'gazebo_msgs/GetLinkStateRequest'
'gazebo_msgs/GetLinkStateResponse'
'gazebo_msgs/GetModelProperties'
'gazebo_msgs/GetModelPropertiesRequest'
'gazebo_msgs/GetModelPropertiesResponse'
'gazebo_msgs/GetModelState'
'gazebo_msgs/GetModelStateRequest'
'gazebo_msgs/GetModelStateResponse'
'gazebo_msgs/GetPhysicsProperties'
'gazebo_msgs/GetPhysicsPropertiesRequest'
'gazebo_msgs/GetPhysicsPropertiesResponse'
'gazebo_msgs/GetWorldProperties'
'gazebo_msgs/GetWorldPropertiesRequest'
'gazebo_msgs/GetWorldPropertiesResponse'
'gazebo_msgs/JointRequest'
'gazebo_msgs/JointRequestRequest'
'gazebo_msgs/JointRequestResponse'
'gazebo_msgs/LinkState'
'gazebo_msgs/LinkStates'
'gazebo_msgs/ModelState'
'gazebo_msgs/ModelStates'
'gazebo_msgs/ODEJointProperties'
'gazebo_msgs/ODEPhysics'
'gazebo_msgs/SetJointProperties'
'gazebo_msgs/SetJointPropertiesRequest'
'gazebo_msgs/SetJointPropertiesResponse'
'gazebo_msgs/SetJointTrajectory'
'gazebo_msgs/SetJointTrajectoryRequest'
'gazebo_msgs/SetJointTrajectoryResponse'
'gazebo_msgs/SetLinkProperties'
'gazebo_msgs/SetLinkPropertiesRequest'
```

```
'gazebo_msgs/SetLinkPropertiesResponse'
'gazebo_msgs/SetLinkState'
'gazebo_msgs/SetLinkStateRequest'
'gazebo_msgs/SetLinkStateResponse'
'gazebo_msgs/SetModelConfiguration'
'gazebo_msgs/SetModelConfigurationRequest'
'gazebo_msgs/SetModelConfigurationResponse'
'gazebo_msgs/SetModelState'
'gazebo_msgs/SetModelStateRequest'
'gazebo_msgs/SetModelStateResponse'
'gazebo_msgs/SetPhysicsProperties'
'gazebo_msgs/SetPhysicsPropertiesRequest'
'gazebo_msgs/SetPhysicsPropertiesResponse'
'gazebo_msgs/SpawnModel'
'gazebo_msgs/SpawnModelRequest'
'gazebo_msgs/SpawnModelResponse'
'gazebo_msgs/WorldState'
'geographic_msgs/BoundingBox'
'geographic_msgs/GeoPoint'
'geographic_msgs/GeoPose'
'geographic_msgs/GeographicMap'
'geographic_msgs/GeographicMapChanges'
'geographic_msgs/GetGeographicMap'
'geographic_msgs/GetGeographicMapRequest'
'geographic_msgs/GetGeographicMapResponse'
'geographic_msgs/GetRoutePlan'
'geographic_msgs/GetRoutePlanRequest'
'geographic_msgs/GetRoutePlanResponse'
'geographic_msgs/KeyValue'
'geographic_msgs/MapFeature'
'geographic_msgs/RouteNetwork'
'geographic_msgs/RoutePath'
'geographic_msgs/RouteSegment'
'geographic_msgs/UpdateGeographicMap'
'geographic_msgs/UpdateGeographicMapRequest'
'geographic_msgs/UpdateGeographicMapResponse'
'geographic_msgs/WayPoint'
'geometry_msgs/Point'
'geometry_msgs/Point32'
'geometry_msgs/PointStamped'
'geometry_msgs/Polygon'
'geometry_msgs/PolygonStamped'
'geometry_msgs/Pose'
'geometry_msgs/Pose2D'
```

```
'geometry_msgs/PoseArray'
'geometry_msgs/PoseStamped'
'geometry_msgs/PoseWithCovariance'
'geometry_msgs/PoseWithCovarianceStamped'
'geometry_msgs/Quaternion'
'geometry_msgs/QuaternionStamped'
'geometry_msgs/Transform'
'geometry_msgs/TransformStamped'
'geometry_msgs/Twist'
'geometry_msgs/TwistStamped'
'geometry_msgs/TwistWithCovariance'
'geometry_msgs/TwistWithCovarianceStamped'
'geometry_msgs/Vector3'
'geometry_msgs/Vector3Stamped'
'geometry_msgs/Wrench'
'geometry_msgs/WrenchStamped'
'gps_common/GPSFix'
'gps_common/GPSStatus'
'graft/GraftControl'
'graft/GraftSensorResidual'
'graft/GraftState'
'graph_msgs/Edges'
'graph_msgs/GeometryGraph'
'grasp_stability_msgs/Control'
'grasp_stability_msgs/ControlRequest'
'grasp_stability_msgs/ControlResponse'
'grasp_stability_msgs/GraspStability'
'grasping_msgs/GraspableObject'
'grasping_msgs/Object'
'grasping_msgs/ObjectProperty'
'grizzly_msgs/Ambience'
'grizzly_msgs/Drive'
'grizzly_msgs/RawStatus'
'handle_detector/CylinderArrayMsg'
'handle_detector/CylinderMsg'
'handle_detector/HandleListMsg'
'hector_mapping/HectorDebugInfo'
'hector_mapping/HectorIterData'
'hector_nav_msgs/GetDistanceToObstacle'
'hector_nav_msgs/GetDistanceToObstacleRequest'
'hector_nav_msgs/GetDistanceToObstacleResponse'
'hector_nav_msgs/GetNormal'
'hector_nav_msgs/GetNormalRequest'
'hector_nav_msgs/GetNormalResponse'
```

```
'hector_nav_msgs/GetRecoveryInfo'
'hector_nav_msgs/GetRecoveryInfoRequest'
'hector_nav_msgs/GetRecoveryInfoResponse'
'hector_nav_msgs/GetRobotTrajectory'
'hector_nav_msgs/GetRobotTrajectoryRequest'
'hector_nav_msgs/GetRobotTrajectoryResponse'
'hector_nav_msgs/GetSearchPosition'
'hector_nav_msgs/GetSearchPositionRequest'
'hector_nav_msgs/GetSearchPositionResponse'
'hector_uav_msgs/Altimeter'
'hector_uav_msgs/AttitudeCommand'
'hector_uav_msgs/Compass'
'hector_uav_msgs/ControllerState'
'hector_uav_msgs/HeadingCommand'
'hector_uav_msgs/HeightCommand'
'hector_uav_msgs/MotorCommand'
'hector_uav_msgs/MotorPWM'
'hector_uav_msgs/MotorStatus'
'hector_uav_msgs/PositionXYCommand'
'hector_uav_msgs/RC'
'hector_uav_msgs/RawImu'
'hector_uav_msgs/RawMagnetic'
'hector_uav_msgs/RawRC'
'hector_uav_msgs/RuddersCommand'
'hector_uav_msgs/ServoCommand'
'hector_uav_msgs/Supply'
'hector_uav_msgs/ThrustCommand'
'hector_uav_msgs/VelocityXYCommand'
'hector_uav_msgs/VelocityZCommand'
'hector_uav_msgs/YawrateCommand'
'hector_worldmodel_msgs/AddObject'
'hector_worldmodel_msgs/AddObjectRequest'
'hector_worldmodel_msgs/AddObjectResponse'
'hector_worldmodel_msgs/GetObjectModel'
'hector_worldmodel_msgs/GetObjectModelRequest'
'hector_worldmodel_msgs/GetObjectModelResponse'
'hector_worldmodel_msgs/ImagePercept'
'hector_worldmodel_msgs/Object'
'hector_worldmodel_msgs/ObjectInfo'
'hector_worldmodel_msgs/ObjectModel'
'hector_worldmodel_msgs/ObjectState'
'hector_worldmodel_msgs/PerceptInfo'
'hector_worldmodel_msgs/PosePercept'
'hector_worldmodel_msgs/SetObjectName'
```

```
'hector_worldmodel_msgs/SetObjectNameRequest'
'hector_worldmodel_msgs/SetObjectNameResponse'
'hector_worldmodel_msgs/SetObjectState'
'hector_worldmodel_msgs/SetObjectStateRequest'
'hector_worldmodel_msgs/SetObjectStateResponse'
'hector_worldmodel_msgs/VerifyObject'
'hector_worldmodel_msgs/VerifyObjectRequest'
'hector_worldmodel_msgs/VerifyObjectResponse'
'hector_worldmodel_msgs/VerifyPercept'
'hector_worldmodel_msgs/VerifyPerceptRequest'
'hector_worldmodel_msgs/VerifyPerceptResponse'
'household_objects_database_msgs/DatabaseModelPose'
'household_objects_database_msgs/DatabaseModelPoseList'
'household_objects_database_msgs/DatabaseReturnCode'
'household_objects_database_msgs/DatabaseScan'
'household_objects_database_msgs/GetModelDescription'
'household_objects_database_msgs/GetModelDescriptionRequest'
'household_objects_database_msgs/GetModelDescriptionResponse'
'household_objects_database_msgs/GetModelList'
'household_objects_database_msgs/GetModelListRequest'
'household_objects_database_msgs/GetModelListResponse'
'household_objects_database_msgs/GetModelMesh'
'household_objects_database_msgs/GetModelMeshRequest'
'household_objects_database_msgs/GetModelMeshResponse'
'household_objects_database_msgs/GetModelScans'
'household_objects_database_msgs/GetModelScansRequest'
'household_objects_database_msgs/GetModelScansResponse'
'household_objects_database_msgs/SaveScan'
'household_objects_database_msgs/SaveScanRequest'
'household_objects_database_msgs/SaveScanResponse'
'household_objects_database_msgs/TranslateRecognitionId'
'household_objects_database_msgs/TranslateRecognitionIdRequest'
'household_objects_database_msgs/TranslateRecognitionIdResponse'
'hrpsys_gazebo_msgs/JointCommand'
'hrpsys_gazebo_msgs/NamedImu'
'hrpsys_gazebo_msgs/NamedWrench'
'hrpsys_gazebo_msgs/RobotState'
'hrpsys_gazebo_msgs/SyncCommand'
'hrpsys_gazebo_msgs/SyncCommandRequest'
'hrpsys_gazebo_msgs/SyncCommandResponse'
'humanoid_nav_msgs/ClipFootstep'
'humanoid_nav_msgs/ClipFootstepRequest'
'humanoid_nav_msgs/ClipFootstepResponse'
'humanoid_nav_msgs/PlanFootsteps'
```

```
'humanoid_nav_msgs/PlanFootstepsBetweenFeet'
'humanoid_nav_msgs/PlanFootstepsBetweenFeetRequest'
'humanoid_nav_msgs/PlanFootstepsBetweenFeetResponse'
'humanoid_nav_msgs/PlanFootstepsRequest'
'humanoid_nav_msgs/PlanFootstepsResponse'
'humanoid_nav_msgs/StepTarget'
'humanoid_nav_msgs/StepTargetService'
'humanoid_nav_msgs/StepTargetServiceRequest'
'humanoid_nav_msgs/StepTargetServiceResponse'
'iai_content_msgs/content_msg'
'iai_content_msgs/content_msgRequest'
'iai_content_msgs/content_msgResponse'
'iai_kinematics_msgs/ErrorCodes'
'iai_kinematics_msgs/GetKinematicSolverInfo'
'iai_kinematics_msgs/GetKinematicSolverInfoRequest'
'iai_kinematics_msgs/GetKinematicSolverInfoResponse'
'iai_kinematics_msgs/GetPositionFK'
'iai_kinematics_msgs/GetPositionFKRequest'
'iai_kinematics_msgs/GetPositionFKResponse'
'iai_kinematics_msgs/GetPositionIK'
'iai_kinematics_msgs/GetPositionIKRequest'
'iai_kinematics_msgs/GetPositionIKResponse'
'iai_kinematics_msgs/GetWeightedIK'
'iai_kinematics_msgs/GetWeightedIKRequest'
'iai_kinematics_msgs/GetWeightedIKResponse'
'iai_kinematics_msgs/JointLimits'
'iai_kinematics_msgs/KDLWeights'
'iai_kinematics_msgs/KinematicSolverInfo'
'iai_kinematics_msgs/MultiDOFJointState'
'iai_kinematics_msgs/PositionIKRequest'
'iai_kinematics_msgs/RobotState'
'image_cb_detector/ImagePoint'
'image_cb_detector/ObjectInImage'
'image_exposure_msgs/ExposureSequence'
'image_exposure_msgs/ImageExposureStatistics'
'image_exposure_msgs/SequenceExposureStatistics'
'image_view2/ImageMarker2'
'image_view2/PointArrayStamped'
'industrial_msgs/CmdJointTrajectory'
'industrial_msgs/CmdJointTrajectoryRequest'
'industrial_msgs/CmdJointTrajectoryResponse'
'industrial_msgs/DebugLevel'
'industrial_msgs/DeviceInfo'
'industrial_msgs/GetRobotInfo'
```

```
'industrial_msgs/GetRobotInfoRequest'
'industrial_msgs/GetRobotInfoResponse'
'industrial_msgs/RobotMode'
'industrial_msgs/RobotStatus'
'industrial_msgs/ServiceReturnCode'
'industrial_msgs/SetDrivePower'
'industrial_msgs/SetDrivePowerRequest'
'industrial_msgs/SetDrivePowerResponse'
'industrial_msgs/SetRemoteLoggerLevel'
'industrial_msgs/SetRemoteLoggerLevelRequest'
'industrial_msgs/SetRemoteLoggerLevelResponse'
'industrial_msgs/StartMotion'
'industrial_msgs/StartMotionRequest'
'industrial_msgs/StartMotionResponse'
'industrial_msgs/StopMotion'
'industrial_msgs/StopMotionRequest'
'industrial_msgs/StopMotionResponse'
'industrial_msgs/TriState'
'interaction_cursor_msgs/InteractionCursorFeedback'
'interaction_cursor_msgs/InteractionCursorUpdate'
'interactive_marker_proxy/GetInit'
'interactive_marker_proxy/GetInitRequest'
'interactive_marker_proxy/GetInitResponse'
'jaco_msgs/FingerPosition'
'jaco_msgs/HomeArm'
'jaco_msgs/HomeArmRequest'
'jaco_msgs/HomeArmResponse'
'jaco_msgs/JointAngles'
'jaco_msgs/JointVelocity'
'jaco_msgs/Start'
'jaco_msgs/StartRequest'
'jaco_msgs/StartResponse'
'jaco_msgs/Stop'
'jaco_msgs/StopRequest'
'jaco_msgs/StopResponse'
'jsk_footstep_controller/RequireLog'
'jsk_footstep_controller/RequireLogRequest'
'jsk_footstep_controller/RequireLogResponse'
'jsk_footstep_msgs/Footstep'
'jsk_footstep_msgs/FootstepArray'
'jsk_gui_msgs/Action'
'jsk_gui_msgs/AndroidSensor'
'jsk_gui_msgs/DeviceSensor'
'jsk_gui_msgs/DeviceSensorALL'
```

```
'jsk_gui_msgs/Gravity'
'jsk_gui_msgs/Imu'
'jsk_gui_msgs/MagneticField'
'jsk_gui_msgs/MultiTouch'
'jsk_gui_msgs/Query'
'jsk_gui_msgs/QueryRequest'
'jsk_gui_msgs/QueryResponse'
'jsk_gui_msgs/Tablet'
'jsk_gui_msgs/Touch'
'jsk_gui_msgs/TouchEvent'
'jsk_gui_msgs/VoiceMessage'
'jsk_hark_msgs/HarkPower'
'jsk_network_tools/Heartbeat'
'jsk_network_tools/HeartbeatResponse'
'jsk_pcl_ros/BoundingBox'
'jsk_pcl_ros/BoundingBoxArray'
'jsk_pcl_ros/BoundingBoxMovement'
'jsk_pcl_ros/CallPolygon'
'jsk_pcl_ros/CallPolygonRequest'
'jsk_pcl_ros/CallPolygonResponse'
'jsk_pcl_ros/CallSnapIt'
'jsk_pcl_ros/CallSnapItRequest'
'jsk_pcl_ros/CallSnapItResponse'
'jsk_pcl_ros/CheckCircle'
'jsk_pcl_ros/CheckCircleRequest'
'jsk_pcl_ros/CheckCircleResponse'
'jsk_pcl_ros/ClusterPointIndices'
'jsk_pcl_ros/ColorHistogram'
'jsk_pcl_ros/ColorHistogramArray'
'jsk_pcl_ros/DepthErrorResult'
'jsk_pcl_ros/EnvironmentLock'
'jsk_pcl_ros/EnvironmentLockRequest'
'jsk_pcl_ros/EnvironmentLockResponse'
'jsk_pcl_ros/EuclideanSegment'
'jsk_pcl_ros/EuclideanSegmentRequest'
'jsk_pcl_ros/EuclideanSegmentResponse'
'jsk_pcl_ros/ICPAlign'
'jsk_pcl_ros/ICPAlignRequest'
'jsk_pcl_ros/ICPAlignResponse'
'jsk_pcl_ros/Int32Stamped'
'jsk_pcl_ros/ModelCoefficientsArray'
'jsk_pcl_ros/ParallelEdge'
'jsk_pcl_ros/ParallelEdgeArray'
'jsk_pcl_ros/PointsArray'
```

```
'jsk_pcl_ros/PolygonArray'
'jsk_pcl_ros/PolygonOnEnvironment'
'jsk_pcl_ros/PolygonOnEnvironmentRequest'
'jsk_pcl_ros/PolygonOnEnvironmentResponse'
'jsk_pcl_ros/RobotPickupReleasePoint'
'jsk_pcl_ros/RobotPickupReleasePointRequest'
'jsk_pcl_ros/RobotPickupReleasePointResponse'
'jsk_pcl_ros/SetPointCloud2'
'jsk_pcl_ros/SetPointCloud2Request'
'jsk_pcl_ros/SetPointCloud2Response'
'jsk_pcl_ros/SlicedPointCloud'
'jsk_pcl_ros/SnapItRequest'
'jsk_pcl_ros/SparseOccupancyGrid'
'jsk_pcl_ros/SparseOccupancyGridArray'
'jsk_pcl_ros/SparseOccupancyGridCell'
'jsk_pcl_ros/SparseOccupancyGridColumn'
'jsk_pcl_ros/SwitchTopic'
'jsk_pcl_ros/SwitchTopicRequest'
'jsk_pcl_ros/SwitchTopicResponse'
'jsk_pcl_ros/TowerPickUp'
'jsk_pcl_ros/TowerPickUpRequest'
'jsk_pcl_ros/TowerPickUpResponse'
'jsk_pcl_ros/TowerRobotMoveCommand'
'jsk_pcl_ros/TowerRobotMoveCommandRequest'
'jsk_pcl_ros/TowerRobotMoveCommandResponse'
'jsk_pcl_ros/TransformScreenpoint'
'jsk_pcl_ros/TransformScreenpointRequest'
'jsk_pcl_ros/TransformScreenpointResponse'
'jsk_perception/Circle2D'
'jsk_perception/Circle2DArray'
'jsk_perception/EuclideanSegment'
'jsk_perception/EuclideanSegmentRequest'
'jsk_perception/EuclideanSegmentResponse'
'jsk_perception/Line'
'jsk_perception/LineArray'
'jsk_perception/PointsArray'
'jsk_perception/Rect'
'jsk_perception/RotatedRect'
'jsk_perception/RotatedRectStamped'
'jsk_perception/SetTemplate'
'jsk_perception/SetTemplateRequest'
'jsk_perception/SetTemplateResponse'
'jsk_perception/SparseImage'
'jsk_perception/WhiteBalance'
```

```
'jsk_perception/WhiteBalancePoints'
'jsk_perception/WhiteBalancePointsRequest'
'jsk_perception/WhiteBalancePointsResponse'
'jsk_perception/WhiteBalanceRequest'
'jsk_perception/WhiteBalanceResponse'
'jsk_rviz_plugins/OverlayMenu'
'jsk_rviz_plugins/OverlayText'
'jsk_topic_tools/List'
'jsk_topic_tools/ListRequest'
'jsk_topic_tools/ListResponse'
'jsk_topic_tools/TopicInfo'
'jsk_topic_tools/Update'
'jsk_topic_tools/UpdateRequest'
'jsk_topic_tools/UpdateResponse'
'keyboard/Key'
'kingfisher_msgs/Course'
'kingfisher_msgs/Drive'
'kingfisher_msgs/Helm'
'kingfisher_msgs/Sense'
'kobuki_msgs/BumperEvent'
'kobuki_msgs/ButtonEvent'
'kobuki_msgs/CliffEvent'
'kobuki_msgs/ControllerInfo'
'kobuki_msgs/DigitalInputEvent'
'kobuki_msgs/DigitalOutput'
'kobuki_msgs/DockInfraRed'
'kobuki_msgs/ExternalPower'
'kobuki_msgs/KeyboardInput'
'kobuki_msgs/Led'
'kobuki_msgs/MotorPower'
'kobuki_msgs/PowerSystemEvent'
'kobuki_msgs/RobotStateEvent'
'kobuki_msgs/SensorState'
'kobuki_msgs/Sound'
'kobuki_msgs/VersionInfo'
'kobuki_msgs/WheelDropEvent'
'kobuki_testsuite/ScanAngle'
'laser_assembler/AssembleScans'
'laser_assembler/AssembleScans2'
'laser_assembler/AssembleScans2Request'
'laser_assembler/AssembleScans2Response'
'laser_assembler/AssembleScansRequest'
'laser_assembler/AssembleScansResponse'
'leap_motion/leap'
```

```
'leap_motion/leapros'
'linux_hardware/LaptopChargeStatus'
'lizi/imu_calib'
'lizi/imu_calibRequest'
'lizi/imu_calibResponse'
'lizi/lizi_command'
'lizi/lizi_gps'
'lizi/lizi_pan_tilt'
'lizi/lizi_raw'
'lizi/lizi_status'
'lizi/set_odom'
'lizi/set_odomRequest'
'lizi/set_odomResponse'
'manipulation_msgs/CartesianGains'
'manipulation_msgs/ClusterBoundingBox'
'manipulation_msgs/Grasp'
'manipulation_msgs/GraspPlanning'
'manipulation_msgs/GraspPlanningErrorCode'
'manipulation_msgs/GraspPlanningRequest'
'manipulation_msgs/GraspPlanningResponse'
'manipulation_msgs/GraspResult'
'manipulation_msgs/GraspableObject'
'manipulation_msgs/GraspableObjectList'
'manipulation_msgs/GripperTranslation'
'manipulation_msgs/ManipulationPhase'
'manipulation_msgs/ManipulationResult'
'manipulation_msgs/PlaceLocation'
'manipulation_msgs/PlaceLocationResult'
'manipulation_msgs/SceneRegion'
'map_merger/LogMaps'
'map_merger/LogMapsRequest'
'map_merger/LogMapsResponse'
'map_merger/TransformPoint'
'map_merger/TransformPointRequest'
'map_merger/TransformPointResponse'
'map_msgs/GetMapROI'
'map_msgs/GetMapROIRequest'
'map_msgs/GetMapROIResponse'
'map_msgs/GetPointMap'
'map_msgs/GetPointMapROI'
'map_msgs/GetPointMapROIRequest'
'map_msgs/GetPointMapROIResponse'
'map_msgs/GetPointMapRequest'
'map_msgs/GetPointMapResponse'
```

```
'map_msgs/OccupancyGridUpdate'
'map_msgs/PointCloud2Update'
'map_msgs/ProjectedMap'
'map_msgs/ProjectedMapInfo'
'map_msgs/ProjectedMapsInfo'
'map_msgs/ProjectedMapsInfoRequest'
'map_msgs/ProjectedMapsInfoResponse'
'map_msgs/SaveMap'
'map_msgs/SaveMapRequest'
'map_msgs/SaveMapResponse'
'map_msgs/SetMapProjections'
'map_msgs/SetMapProjectionsRequest'
'map_msgs/SetMapProjectionsResponse'
'map_store/DeleteMap'
'map_store/DeleteMapRequest'
'map_store/DeleteMapResponse'
'map_store/ListMaps'
'map_store/ListMapsRequest'
'map_store/ListMapsResponse'
'map_store/MapListEntry'
'map_store/PublishMap'
'map_store/PublishMapRequest'
'map_store/PublishMapResponse'
'map_store/RenameMap'
'map_store/RenameMapRequest'
'map_store/RenameMapResponse'
'map_store/SaveMap'
'map_store/SaveMapRequest'
'map_store/SaveMapResponse'
'mavros/BatteryStatus'
'mavros/CommandBool'
'mavros/CommandBoolRequest'
'mavros/CommandBoolResponse'
'mavros/CommandHome'
'mavros/CommandHomeRequest'
'mavros/CommandHomeResponse'
'mavros/CommandInt'
'mavros/CommandIntRequest'
'mavros/CommandIntResponse'
'mavros/CommandLong'
'mavros/CommandLongRequest'
'mavros/CommandLongResponse'
'mavros/CommandTOL'
'mavros/CommandTOLRequest'
```

```
'mavros/CommandTOLResponse'
'mavros/FileChecksum'
'mavros/FileChecksumRequest'
'mavros/FileChecksumResponse'
'mavros/FileClose'
'mavros/FileCloseRequest'
'mavros/FileCloseResponse'
'mavros/FileEntry'
'mavros/FileList'
'mavros/FileListRequest'
'mavros/FileListResponse'
'mavros/FileMakeDir'
'mavros/FileMakeDirRequest'
'mavros/FileMakeDirResponse'
'mavros/FileOpen'
'mavros/FileOpenRequest'
'mavros/FileOpenResponse'
'mavros/FileRead'
'mavros/FileReadRequest'
'mavros/FileReadResponse'
'mavros/FileRemove'
'mavros/FileRemoveDir'
'mavros/FileRemoveDirRequest'
'mavros/FileRemoveDirResponse'
'mavros/FileRemoveRequest'
'mavros/FileRemoveResponse'
'mavros/FileRename'
'mavros/FileRenameRequest'
'mavros/FileRenameResponse'
'mavros/FileTruncate'
'mavros/FileTruncateRequest'
'mavros/FileTruncateResponse'
'mavros/FileWrite'
'mavros/FileWriteRequest'
'mavros/FileWriteResponse'
'mavros/Mavlink'
'mavros/OverrideRCIn'
'mavros/ParamGet'
'mavros/ParamGetRequest'
'mavros/ParamGetResponse'
'mavros/ParamPull'
'mavros/ParamPullRequest'
'mavros/ParamPullResponse'
'mavros/ParamPush'
```

```
'mavros/ParamPushRequest'
'mavros/ParamPushResponse'
'mavros/ParamSet'
'mavros/ParamSetRequest'
'mavros/ParamSetResponse'
'mavros/RCIn'
'mavros/RCOut'
'mavros/RadioStatus'
'mavros/SetMode'
'mavros/SetModeRequest'
'mavros/SetModeResponse'
'mavros/State'
'mavros/StreamRate'
'mavros/StreamRateRequest'
'mavros/StreamRateResponse'
'mavros/VFR_HUD'
'mavros/Waypoint'
'mavros/WaypointClear'
'mavros/WaypointClearRequest'
'mavros/WaypointClearResponse'
'mavros/WaypointGOTO'
'mavros/WaypointGOTORequest'
'mavros/WaypointGOTOResponse'
'mavros/WaypointList'
'mavros/WaypointPull'
'mavros/WaypointPullRequest'
'mavros/WaypointPullResponse'
'mavros/WaypointPush'
'mavros/WaypointPushRequest'
'mavros/WaypointPushResponse'
'mavros/WaypointSetCurrent'
'mavros/WaypointSetCurrentRequest'
'mavros/WaypointSetCurrentResponse'
'microstrain_3dmgx2_imu/AddOffset'
'microstrain_3dmgx2_imu/AddOffsetRequest'
'microstrain_3dmgx2_imu/AddOffsetResponse'
'ml_classifiers/AddClassData'
'ml_classifiers/AddClassDataRequest'
'ml_classifiers/AddClassDataResponse'
'ml_classifiers/ClassDataPoint'
'ml_classifiers/ClassifyData'
'ml_classifiers/ClassifyDataRequest'
'ml_classifiers/ClassifyDataResponse'
'ml_classifiers/ClearClassifier'
```

```
'ml_classifiers/ClearClassifierRequest'
'ml_classifiers/ClearClassifierResponse'
'ml_classifiers/CreateClassifier'
'ml_classifiers/CreateClassifierRequest'
'ml_classifiers/CreateClassifierResponse'
'ml_classifiers/LoadClassifier'
'ml_classifiers/LoadClassifierRequest'
'ml_classifiers/LoadClassifierResponse'
'ml_classifiers/SaveClassifier'
'ml_classifiers/SaveClassifierRequest'
'ml_classifiers/SaveClassifierResponse'
'ml_classifiers/TrainClassifier'
'ml_classifiers/TrainClassifierRequest'
'ml_classifiers/TrainClassifierResponse'
'mln_robosherlock_msgs/MLNAtoms'
'mln_robosherlock_msgs/MLNQuery'
'mln_robosherlock_msgs/MLNQueryRequest'
'mln_robosherlock_msgs/MLNQueryResponse'
'mongodb_store/GetParam'
'mongodb_store/GetParamRequest'
'mongodb_store/GetParamResponse'
'mongodb_store/MongoFind'
'mongodb_store/MongoFindRequest'
'mongodb_store/MongoFindResponse'
'mongodb_store/MongoInsert'
'mongodb_store/MongoInsertRequest'
'mongodb_store/MongoInsertResponse'
'mongodb_store/MongoUpdate'
'mongodb_store/MongoUpdateRequest'
'mongodb_store/MongoUpdateResponse'
'mongodb_store/SetParam'
'mongodb_store/SetParamRequest'
'mongodb_store/SetParamResponse'
'mongodb_store_msgs/MongoDeleteMsg'
'mongodb_store_msgs/MongoDeleteMsgRequest'
'mongodb_store_msgs/MongoDeleteMsgResponse'
'mongodb_store_msgs/MongoInsertMsg'
'mongodb_store_msgs/MongoInsertMsgRequest'
'mongodb_store_msgs/MongoInsertMsgResponse'
'mongodb_store_msgs/MongoQueryMsg'
'mongodb_store_msgs/MongoQueryMsgRequest'
'mongodb_store_msgs/MongoQueryMsgResponse'
'mongodb_store_msgs/MongoUpdateMsg'
'mongodb_store_msgs/MongoUpdateMsgRequest'
```

```
'mongodb_store_msgs/MongoUpdateMsgResponse'
'mongodb_store_msgs/SerialisedMessage'
'mongodb_store_msgs/StringList'
'mongodb_store_msgs/StringPair'
'mongodb_store_msgs/StringPairList'
'moveit_msgs/AllowedCollisionEntry'
'moveit_msgs/AllowedCollisionMatrix'
'moveit_msgs/AttachedCollisionObject'
'moveit_msgs/BoundingVolume'
'moveit_msgs/CollisionObject'
'moveit_msgs/ConstraintEvalResult'
'moveit_msgs/Constraints'
'moveit_msgs/ContactInformation'
'moveit_msgs/CostSource'
'moveit_msgs/DisplayRobotState'
'moveit_msgs/DisplayTrajectory'
'moveit_msgs/ExecuteKnownTrajectory'
'moveit_msgs/ExecuteKnownTrajectoryRequest'
'moveit_msgs/ExecuteKnownTrajectoryResponse'
'moveit_msgs/GetCartesianPath'
'moveit_msgs/GetCartesianPathRequest'
'moveit_msgs/GetCartesianPathResponse'
'moveit_msgs/GetConstraintAwarePositionIK'
'moveit_msgs/GetConstraintAwarePositionIKRequest'
'moveit_msgs/GetConstraintAwarePositionIKResponse'
'moveit_msgs/GetKinematicSolverInfo'
'moveit_msgs/GetKinematicSolverInfoRequest'
'moveit_msgs/GetKinematicSolverInfoResponse'
'moveit_msgs/GetMotionPlan'
'moveit_msgs/GetMotionPlanRequest'
'moveit_msgs/GetMotionPlanResponse'
'moveit_msgs/GetPlanningScene'
'moveit_msgs/GetPlanningSceneRequest'
'moveit_msgs/GetPlanningSceneResponse'
'moveit_msgs/GetPositionFK'
'moveit_msgs/GetPositionFKRequest'
'moveit_msgs/GetPositionFKResponse'
'moveit_msgs/GetPositionIK'
'moveit_msgs/GetPositionIKRequest'
'moveit_msgs/GetPositionIKResponse'
'moveit_msgs/GetStateValidity'
'moveit_msgs/GetStateValidityRequest'
'moveit_msgs/GetStateValidityResponse'
'moveit_msgs/Grasp'
```

```
'moveit_msgs/GripperTranslation'
'moveit_msgs/JointConstraint'
'moveit_msgs/JointLimits'
'moveit_msgs/KinematicSolverInfo'
'moveit_msgs/LinkPadding'
'moveit_msgs/LinkScale'
'moveit_msgs/LoadMap'
'moveit_msgs/LoadMapRequest'
'moveit_msgs/LoadMapResponse'
'moveit_msgs/MotionPlanDetailedResponse'
'moveit_msgs/MotionPlanRequest'
'moveit_msgs/MotionPlanResponse'
'moveit_msgs/MoveItErrorCodes'
'moveit_msgs/ObjectColor'
'moveit_msgs/OrientationConstraint'
'moveit_msgs/OrientedBoundingBox'
'moveit_msgs/PlaceLocation'
'moveit_msgs/PlannerInterfaceDescription'
'moveit_msgs/PlanningOptions'
'moveit_msgs/PlanningScene'
'moveit_msgs/PlanningSceneComponents'
'moveit_msgs/PlanningSceneWorld'
'moveit_msgs/PositionConstraint'
'moveit_msgs/PositionIKRequest'
'moveit_msgs/QueryPlannerInterfaces'
'moveit_msgs/QueryPlannerInterfacesRequest'
'moveit_msgs/QueryPlannerInterfacesResponse'
'moveit_msgs/RobotState'
'moveit_msgs/RobotTrajectory'
'moveit_msgs/SaveMap'
'moveit_msgs/SaveMapRequest'
'moveit_msgs/SaveMapResponse'
'moveit_msgs/TrajectoryConstraints'
'moveit_msgs/VisibilityConstraint'
'moveit_msgs/WorkspaceParameters'
'moveit_simple_grasps/GraspGeneratorOptions'
'multimaster_msgs_fkie/Capability'
'multimaster_msgs_fkie/DiscoverMasters'
'multimaster_msgs_fkie/DiscoverMastersRequest'
'multimaster_msgs_fkie/DiscoverMastersResponse'
'multimaster_msgs_fkie/GetSyncInfo'
'multimaster_msgs_fkie/GetSyncInfoRequest'
'multimaster_msgs_fkie/GetSyncInfoResponse'
'multimaster_msgs_fkie/LinkState'
```

```
'multimaster_msgs_fkie/LinkStatesStamped'
'multimaster_msgs_fkie/ListDescription'
'multimaster_msgs_fkie/ListDescriptionRequest'
'multimaster_msgs_fkie/ListDescriptionResponse'
'multimaster_msgs_fkie/ListNodes'
'multimaster_msgs_fkie/ListNodesRequest'
'multimaster_msgs_fkie/ListNodesResponse'
'multimaster_msgs_fkie/LoadLaunch'
'multimaster_msgs_fkie/LoadLaunchRequest'
'multimaster_msgs_fkie/LoadLaunchResponse'
'multimaster_msgs_fkie/MasterState'
'multimaster_msgs_fkie/ROSMaster'
'multimaster_msgs_fkie/SyncMasterInfo'
'multimaster_msgs_fkie/SyncServiceInfo'
'multimaster_msgs_fkie/SyncTopicInfo'
'multimaster_msgs_fkie/Task'
'multimaster_msgs_fkie/TaskRequest'
'multimaster_msgs_fkie/TaskResponse'
'multisense_ros/DeviceInfo'
'multisense_ros/Histogram'
'multisense_ros/RawCamCal'
'multisense_ros/RawCamConfig'
'multisense_ros/RawCamData'
'multisense_ros/RawImuData'
'multisense_ros/RawLidarCal'
'multisense_ros/RawLidarData'
'multisense_ros/StampedPps'
'nao_interaction_msgs/AudioMasterVolume'
'nao_interaction_msgs/AudioMasterVolumeRequest'
'nao_interaction_msgs/AudioMasterVolumeResponse'
'nao_interaction_msgs/AudioPlayback'
'nao_interaction_msgs/AudioPlaybackRequest'
'nao_interaction_msgs/AudioPlaybackResponse'
'nao_interaction_msgs/AudioRecorder'
'nao_interaction_msgs/AudioRecorderRequest'
'nao_interaction_msgs/AudioRecorderResponse'
'nao_interaction_msgs/AudioSourceLocalization'
'nao_interaction_msgs/FaceDetected'
'nao_interaction_msgs/LandmarkDetected'
'nao_interaction_msgs/MovementDetected'
'nao_interaction_msgs/VisionMotionSensitivity'
'nao_interaction_msgs/VisionMotionSensitivityRequest'
'nao_interaction_msgs/VisionMotionSensitivityResponse'
'nao_msgs/Bumper'
```

```
'nao_msgs/CmdPoseService'
'nao_msgs/CmdPoseServiceRequest'
'nao_msgs/CmdPoseServiceResponse'
'nao_msgs/CmdVelService'
'nao_msgs/CmdVelServiceRequest'
'nao_msgs/CmdVelServiceResponse'
'nao_msgs/FadeRGB'
'nao_msgs/GetInstalledBehaviors'
'nao_msgs/GetInstalledBehaviorsRequest'
'nao_msgs/GetInstalledBehaviorsResponse'
'nao_msgs/GetTruepose'
'nao_msgs/GetTrueposeRequest'
'nao_msgs/GetTrueposeResponse'
'nao_msgs/JointAngleTrajectory'
'nao_msgs/JointAnglesWithSpeed'
'nao_msgs/SetArmsEnabled'
'nao_msgs/SetArmsEnabledRequest'
'nao_msgs/SetArmsEnabledResponse'
'nao_msgs/SetTransform'
'nao_msgs/SetTransformRequest'
'nao_msgs/SetTransformResponse'
'nao_msgs/TactileTouch'
'nao_msgs/WordRecognized'
'nav2d_msgs/LocalizedScan'
'nav2d_msgs/RobotPose'
'nav2d_navigator/SendCommand'
'nav2d_navigator/SendCommandRequest'
'nav2d_navigator/SendCommandResponse'
'nav2d_operator/cmd'
'nav_msgs/GetMap'
'nav_msgs/GetMapRequest'
'nav_msgs/GetMapResponse'
'nav_msgs/GetPlan'
'nav_msgs/GetPlanRequest'
'nav_msgs/GetPlanResponse'
'nav_msgs/GridCells'
'nav_msgs/MapMetaData'
'nav_msgs/OccupancyGrid'
'nav_msgs/Odometry'
'nav_msgs/Path'
'network_monitor_udp/UdpMonitor'
'network_monitor_udp/UdpSink'
'nmea_msgs/Sentence'
'nodelet/NodeletList'
```

```
'nodelet/NodeletListRequest'
'nodelet/NodeletListResponse'
'nodelet/NodeletLoad'
'nodelet/NodeletLoadRequest'
'nodelet/NodeletLoadResponse'
'nodelet/NodeletUnload'
'nodelet/NodeletUnloadRequest'
'nodelet/NodeletUnloadResponse'
'object_recognition_msgs/GetObjectInformation'
'object_recognition_msgs/GetObjectInformationRequest'
'object_recognition_msgs/GetObjectInformationResponse'
'object_recognition_msgs/ObjectInformation'
'object_recognition_msgs/ObjectType'
'object_recognition_msgs/RecognizedObject'
'object_recognition_msgs/RecognizedObjectArray'
'object_recognition_msgs/Table'
'object_recognition_msgs/TableArray'
'octomap_msgs/BoundingBoxQuery'
'octomap_msgs/BoundingBoxQueryRequest'
'octomap_msgs/BoundingBoxQueryResponse'
'octomap_msgs/GetOctomap'
'octomap_msgs/GetOctomapRequest'
'octomap_msgs/GetOctomapResponse'
'octomap_msgs/Octomap'
'octomap_msgs/OctomapWithPose'
'p2os_driver/AIO'
'p2os_driver/BatteryState'
'p2os_driver/DIO'
'p2os_driver/GripState'
'p2os_driver/GripperState'
'p2os_driver/LiftState'
'p2os_driver/MotorState'
'p2os_driver/PTZState'
'p2os_driver/SonarArray'
'pano_ros/Pano'
'pcl_msgs/ModelCoefficients'
'pcl_msgs/PointIndices'
'pcl_msgs/PolygonMesh'
'pcl_msgs/Vertices'
'pddl_msgs/PDDLAction'
'pddl_msgs/PDDLActionArray'
'pddl_msgs/PDDLDomain'
'pddl_msgs/PDDLObject'
'pddl_msgs/PDDLProblem'
```

```
'pddl_msgs/PDDLStep'
'people_msgs/People'
'people_msgs/Person'
'people_msgs/PersonStamped'
'people_msgs/PositionMeasurement'
'people_msgs/PositionMeasurementArray'
'play_motion_msgs/IsAlreadyThere'
'play_motion_msgs/IsAlreadyThereRequest'
'play_motion_msgs/IsAlreadyThereResponse'
'play_motion_msgs/ListMotions'
'play_motion_msgs/ListMotionsRequest'
'play_motion_msgs/ListMotionsResponse'
'play_motion_msgs/MotionInfo'
'polled_camera/GetPolledImage'
'polled_camera/GetPolledImageRequest'
'polled_camera/GetPolledImageResponse'
'posedetection_msgs/Curve1D'
'posedetection_msgs/Detect'
'posedetection_msgs/DetectRequest'
'posedetection_msgs/DetectResponse'
'posedetection_msgs/FeatureOD'
'posedetection_msgs/FeatureODDetect'
'posedetection_msgs/FeatureODDetectRequest'
'posedetection_msgs/FeatureODDetectResponse'
'posedetection_msgs/Feature1D'
'posedetection_msgs/Feature1DDetect'
'posedetection_msgs/Feature1DDetectRequest'
'posedetection_msgs/Feature1DDetectResponse'
'posedetection_msgs/ImageFeatureOD'
'posedetection_msgs/ImageFeature1D'
'posedetection_msgs/Object6DPose'
'posedetection_msgs/ObjectDetection'
'pr2_calibration_launch/FkTest'
'pr2_calibration_launch/FkTestRequest'
'pr2_calibration_launch/FkTestResponse'
'pr2_controllers_msgs/JointControllerState'
'pr2_controllers_msgs/JointTrajectoryControllerState'
'pr2_controllers_msgs/Pr2GripperCommand'
'pr2_controllers_msgs/QueryCalibrationState'
'pr2_controllers_msgs/QueryCalibrationStateRequest'
'pr2_controllers_msgs/QueryCalibrationStateResponse'
'pr2_controllers_msgs/QueryTrajectoryState'
'pr2_controllers_msgs/QueryTrajectoryStateRequest'
'pr2_controllers_msgs/QueryTrajectoryStateResponse'
```

```
'pr2_gazebo_plugins/ModelJointsState'
'pr2_gazebo_plugins/PlugCommand'
'pr2_gazebo_plugins/SetModelsJointsStates'
'pr2_gazebo_plugins/SetModelsJointsStatesRequest'
'pr2_gazebo_plugins/SetModelsJointsStatesResponse'
'pr2_gripper_sensor_msgs/PR2GripperEventDetectorCommand'
'pr2_gripper_sensor_msgs/PR2GripperEventDetectorData'
'pr2_gripper_sensor_msgs/PR2GripperFindContactCommand'
'pr2_gripper_sensor_msgs/PR2GripperFindContactData'
'pr2_gripper_sensor_msgs/PR2GripperForceServoCommand'
'pr2_gripper_sensor_msgs/PR2GripperForceServoData'
'pr2_gripper_sensor_msgs/PR2GripperGrabCommand'
'pr2_gripper_sensor_msgs/PR2GripperGrabData'
'pr2_gripper_sensor_msgs/PR2GripperPressureData'
'pr2_gripper_sensor_msgs/PR2GripperReleaseCommand'
'pr2_gripper_sensor_msgs/PR2GripperReleaseData'
'pr2_gripper_sensor_msgs/PR2GripperSensorRTState'
'pr2_gripper_sensor_msgs/PR2GripperSensorRawData'
'pr2_gripper_sensor_msgs/PR2GripperSlipServoCommand'
'pr2_gripper_sensor_msgs/PR2GripperSlipServoData'
'pr2_mechanism_controllers/BaseControllerState'
'pr2_mechanism_controllers/BaseControllerState2'
'pr2_mechanism_controllers/BaseOdometryState'
'pr2_mechanism_controllers/DebugInfo'
'pr2_mechanism_controllers/Odometer'
'pr2_mechanism_controllers/OdometryMatrix'
'pr2_mechanism_controllers/SetProfile'
'pr2_mechanism_controllers/SetProfileRequest'
'pr2_mechanism_controllers/SetProfileResponse'
'pr2_mechanism_controllers/TrackLinkCmd'
'pr2_mechanism_msgs/ActuatorStatistics'
'pr2_mechanism_msgs/ControllerStatistics'
'pr2_mechanism_msgs/JointStatistics'
'pr2_mechanism_msgs/ListControllerTypes'
'pr2_mechanism_msgs/ListControllerTypesRequest'
'pr2_mechanism_msgs/ListControllerTypesResponse'
'pr2_mechanism_msgs/ListControllers'
'pr2_mechanism_msgs/ListControllersRequest'
'pr2_mechanism_msgs/ListControllersResponse'
'pr2_mechanism_msgs/LoadController'
'pr2_mechanism_msgs/LoadControllerRequest'
'pr2_mechanism_msgs/LoadControllerResponse'
'pr2_mechanism_msgs/MechanismStatistics'
'pr2_mechanism_msgs/ReloadControllerLibraries'
```

```
'pr2_mechanism_msgs/ReloadControllerLibrariesRequest'
'pr2_mechanism_msgs/ReloadControllerLibrariesResponse'
'pr2_mechanism_msgs/SwitchController'
'pr2_mechanism_msgs/SwitchControllerRequest'
'pr2_mechanism_msgs/SwitchControllerResponse'
'pr2_mechanism_msgs/UnloadController'
'pr2_mechanism_msgs/UnloadControllerRequest'
'pr2_mechanism_msgs/UnloadControllerResponse'
'pr2_msgs/AccelerometerState'
'pr2_msgs/AccessPoint'
'pr2_msgs/BatteryServer'
'pr2_msgs/BatteryServer2'
'pr2_msgs/BatteryState'
'pr2_msgs/BatteryState2'
'pr2_msgs/DashboardState'
'pr2_msgs/GPUStatus'
'pr2_msgs/LaserScannerSignal'
'pr2_msgs/LaserTrajCmd'
'pr2_msgs/PeriodicCmd'
'pr2_msgs/PowerBoardState'
'pr2_msgs/PowerState'
'pr2_msgs/PressureState'
'pr2_msgs/SetLaserTrajCmd'
'pr2_msgs/SetLaserTrajCmdRequest'
'pr2_msgs/SetLaserTrajCmdResponse'
'pr2_msgs/SetPeriodicCmd'
'pr2_msgs/SetPeriodicCmdRequest'
'pr2_msgs/SetPeriodicCmdResponse'
'pr2_power_board/PowerBoardCommand'
'pr2_power_board/PowerBoardCommand2'
'pr2_power_board/PowerBoardCommand2Request'
'pr2_power_board/PowerBoardCommand2Response'
'pr2_power_board/PowerBoardCommandRequest'
'pr2_power_board/PowerBoardCommandResponse'
'pr2_self_test_msgs/ConfirmConf'
'pr2_self_test_msgs/ConfirmConfRequest'
'pr2_self_test_msgs/ConfirmConfResponse'
'pr2_self_test_msgs/Plot'
'pr2_self_test_msgs/ScriptDone'
'pr2_self_test_msgs/ScriptDoneRequest'
'pr2_self_test_msgs/ScriptDoneResponse'
'pr2_self_test_msgs/TestInfo'
'pr2_self_test_msgs/TestInfoArray'
'pr2_self_test_msgs/TestParam'
```

```
'pr2_self_test_msgs/TestResult'
'pr2_self_test_msgs/TestResultRequest'
'pr2_self_test_msgs/TestResultResponse'
'pr2_self_test_msgs/TestStatus'
'pr2_self_test_msgs/TestValue'
'program_queue/CallProgram'
'program_queue/CallProgramRequest'
'program_queue/CallProgramResponse'
'program_queue/ClearQueue'
'program_queue/ClearQueueRequest'
'program_queue/ClearQueueResponse'
'program_queue/CreateProgram'
'program_queue/CreateProgramRequest'
'program_queue/CreateProgramResponse'
'program_queue/CreateUser'
'program_queue/CreateUserRequest'
'program_queue/CreateUserResponse'
'program_queue/DequeueProgram'
'program_queue/DequeueProgramRequest'
'program_queue/DequeueProgramResponse'
'program_queue/GetMyPrograms'
'program_queue/GetMyProgramsRequest'
'program_queue/GetMyProgramsResponse'
'program_queue/GetOutput'
'program_queue/GetOutputRequest'
'program_queue/GetOutputResponse'
'program_queue/GetProgram'
'program_queue/GetProgramRequest'
'program_queue/GetProgramResponse'
'program_queue/GetPrograms'
'program_queue/GetProgramsRequest'
'program_queue/GetProgramsResponse'
'program_queue/GetQueue'
'program_queue/GetQueueRequest'
'program_queue/GetQueueResponse'
'program_queue/Login'
'program_queue/LoginRequest'
'program_queue/LoginResponse'
'program_queue/Logout'
'program_queue/LogoutRequest'
'program_queue/LogoutResponse'
'program_queue/Output'
'program_queue/Program'
'program_queue/ProgramInfo'
```

```
'program_queue/QueueProgram'
'program_queue/QueueProgramRequest'
'program_queue/QueueProgramResponse'
'program_queue/RunProgram'
'program_queue/RunProgramRequest'
'program_queue/RunProgramResponse'
'program_queue/UpdateProgram'
'program_queue/UpdateProgramRequest'
'program_queue/UpdateProgramResponse'
'qt_tutorials/TwoInts'
'qt_tutorials/TwoIntsRequest'
'qt_tutorials/TwoIntsResponse'
'r2_msgs/Gains'
'r2_msgs/JointControl'
'r2_msgs/JointStatus'
'r2_msgs/JointStatusArray'
'r2_msgs/PDMCStatus'
'r2_msgs/ParseTableScene'
'r2_msgs/ParseTableSceneRequest'
'r2_msgs/ParseTableSceneResponse'
'r2_msgs/PoseCommand'
'r2_msgs/PoseCommandArray'
'r2_msgs/PoseCommandParams'
'r2_msgs/PoseCommandStatus'
'r2_msgs/PoseTwistStamped'
'r2_msgs/Power'
'r2_msgs/PowerRequest'
'r2_msgs/PowerResponse'
'r2_msgs/ResetTableScene'
'r2_msgs/ResetTableSceneRequest'
'r2_msgs/ResetTableSceneResponse'
'r2_msgs/Servo'
'r2_msgs/ServoRequest'
'r2_msgs/ServoResponse'
'r2_msgs/SetJointMode'
'r2_msgs/SetJointModeRequest'
'r2_msgs/SetJointModeResponse'
'r2_msgs/SetTipName'
'r2_msgs/SetTipNameRequest'
'r2_msgs/SetTipNameResponse'
'r2_msgs/TakeSnapshot'
'r2_msgs/TakeSnapshotRequest'
'r2_msgs/TakeSnapshotResponse'
'r2_msgs/TorsoStatus'
```

```
'razer_hydra/Hydra'
'razer_hydra/HydraPaddle'
'razer_hydra/HydraRaw'
'rmp_msgs/AudioCommand'
'rmp_msgs/Battery'
'rmp_msgs/BoolStamped'
'rmp_msgs/FaultStatus'
'rmp_msgs/MotorStatus'
'roboteq_msgs/Command'
'roboteq_msgs/Feedback'
'roboteq_msgs/Status'
'robotnik_msgs/AlarmSensor'
'robotnik_msgs/Alarms'
'robotnik_msgs/Axis'
'robotnik_msgs/Data'
'robotnik_msgs/Interfaces'
'robotnik_msgs/MotorStatus'
'robotnik_msgs/MotorsStatus'
'robotnik_msgs/axis_record'
'robotnik_msgs/axis_recordRequest'
'robotnik_msgs/axis_recordResponse'
'robotnik_msgs/enable_disable'
'robotnik_msgs/enable_disableRequest'
'robotnik_msgs/enable_disableResponse'
'robotnik_msgs/encoders'
'robotnik_msgs/get_digital_input'
'robotnik_msgs/get_digital_inputRequest'
'robotnik_msgs/get_digital_inputResponse'
'robotnik_msgs/get_mode'
'robotnik_msgs/get_modeRequest'
'robotnik_msgs/get_modeResponse'
'robotnik_msgs/home'
'robotnik_msgs/homeRequest'
'robotnik_msgs/homeResponse'
'robotnik_msgs/inputs_outputs'
'robotnik_msgs/ptz'
'robotnik_msgs/set_analog_output'
'robotnik_msgs/set_analog_outputRequest'
'robotnik_msgs/set_analog_outputResponse'
'robotnik_msgs/set_digital_output'
'robotnik_msgs/set_digital_outputRequest'
'robotnik_msgs/set_digital_outputResponse'
'robotnik_msgs/set_float_value'
'robotnik_msgs/set_float_valueRequest'
```

```
'robotnik_msgs/set_float_valueResponse'
'robotnik_msgs/set_height'
'robotnik_msgs/set_heightRequest'
'robotnik_msgs/set_heightResponse'
'robotnik_msgs/set_mode'
'robotnik_msgs/set_modeRequest'
'robotnik_msgs/set_modeResponse'
'robotnik_msgs/set_odometry'
'robotnik_msgs/set_odometryRequest'
'robotnik_msgs/set_odometryResponse'
'robotnik_msgs/set_ptz'
'robotnik_msgs/set_ptzRequest'
'robotnik_msgs/set_ptzResponse'
'rocon_app_manager_msgs/App'
'rocon_app_manager_msgs/AppList'
'rocon_app_manager_msgs/Constants'
'rocon_app_manager_msgs/ErrorCodes'
'rocon_app_manager_msgs/GetAppList'
'rocon_app_manager_msgs/GetAppListRequest'
'rocon_app_manager_msgs/GetAppListResponse'
'rocon_app_manager_msgs/GetPlatformInfo'
'rocon_app_manager_msgs/GetPlatformInfoRequest'
'rocon_app_manager_msgs/GetPlatformInfoResponse'
'rocon_app_manager_msgs/Icon'
'rocon_app_manager_msgs/Init'
'rocon_app_manager_msgs/InitRequest'
'rocon_app_manager_msgs/InitResponse'
'rocon_app_manager_msgs/Invite'
'rocon_app_manager_msgs/InviteRequest'
'rocon_app_manager_msgs/InviteResponse'
'rocon_app_manager_msgs/KeyValue'
'rocon_app_manager_msgs/PairingClient'
'rocon_app_manager_msgs/PlatformInfo'
'rocon_app_manager_msgs/Remapping'
'rocon_app_manager_msgs/SimpleInvite'
'rocon_app_manager_msgs/SimpleInviteRequest'
'rocon_app_manager_msgs/SimpleInviteResponse'
'rocon_app_manager_msgs/StartApp'
'rocon_app_manager_msgs/StartAppRequest'
'rocon_app_manager_msgs/StartAppResponse'
'rocon_app_manager_msgs/Status'
'rocon_app_manager_msgs/StatusRequest'
'rocon_app_manager_msgs/StatusResponse'
'rocon_app_manager_msgs/StopApp'
```

```
'rocon_app_manager_msgs/StopAppRequest'
'rocon_app_manager_msgs/StopAppResponse'
'rocon_interaction_msgs/ErrorCodes'
'rocon_interaction_msgs/GetInteraction'
'rocon_interaction_msgs/GetInteractionRequest'
'rocon_interaction_msgs/GetInteractionResponse'
'rocon_interaction_msgs/GetInteractions'
'rocon_interaction_msgs/GetInteractionsRequest'
'rocon_interaction_msgs/GetInteractionsResponse'
'rocon_interaction_msgs/Interaction'
'rocon_interaction_msgs/InteractiveClient'
'rocon_interaction_msgs/InteractiveClients'
'rocon_interaction_msgs/RemoconStatus'
'rocon_interaction_msgs/RequestInteraction'
'rocon_interaction_msgs/RequestInteractionRequest'
'rocon_interaction_msgs/RequestInteractionResponse'
'rocon_interaction_msgs/Roles'
'rocon_interaction_msgs/SetInteractions'
'rocon_interaction_msgs/SetInteractionsRequest'
'rocon_interaction_msgs/SetInteractionsResponse'
'rocon_interaction_msgs/Strings'
'rocon_service_pair_msgs/TestiesPair'
'rocon_service_pair_msgs/TestiesPairRequest'
'rocon_service_pair_msgs/TestiesPairResponse'
'rocon_service_pair_msgs/TestiesRequest'
'rocon_service_pair_msgs/TestiesResponse'
'rocon_std_msgs/GetPlatformInfo'
'rocon_std_msgs/GetPlatformInfoRequest'
'rocon_std_msgs/GetPlatformInfoResponse'
'rocon_std_msgs/Icon'
'rocon_std_msgs/KeyValue'
'rocon_std_msgs/MasterInfo'
'rocon_std_msgs/PlatformInfo'
'rocon_std_msgs/Remapping'
'rocon_std_msgs/StringArray'
'rocon_std_msgs/Strings'
'rosapi/DeleteParam'
'rosapi/DeleteParamRequest'
'rosapi/DeleteParamResponse'
'rosapi/GetParam'
'rosapi/GetParamNames'
'rosapi/GetParamNamesRequest'
'rosapi/GetParamNamesResponse'
'rosapi/GetParamRequest'
```

```
'rosapi/GetParamResponse'
'rosapi/GetTime'
'rosapi/GetTimeRequest'
'rosapi/GetTimeResponse'
'rosapi/HasParam'
'rosapi/HasParamRequest'
'rosapi/HasParamResponse'
'rosapi/MessageDetails'
'rosapi/MessageDetailsRequest'
'rosapi/MessageDetailsResponse'
'rosapi/Nodes'
'rosapi/NodesRequest'
'rosapi/NodesResponse'
'rosapi/Publishers'
'rosapi/PublishersRequest'
'rosapi/PublishersResponse'
'rosapi/SearchParam'
'rosapi/SearchParamRequest'
'rosapi/SearchParamResponse'
'rosapi/ServiceHost'
'rosapi/ServiceHostRequest'
'rosapi/ServiceHostResponse'
'rosapi/ServiceNode'
'rosapi/ServiceNodeRequest'
'rosapi/ServiceNodeResponse'
'rosapi/ServiceProviders'
'rosapi/ServiceProvidersRequest'
'rosapi/ServiceProvidersResponse'
'rosapi/ServiceRequestDetails'
'rosapi/ServiceRequestDetailsRequest'
'rosapi/ServiceRequestDetailsResponse'
'rosapi/ServiceResponseDetails'
'rosapi/ServiceResponseDetailsRequest'
'rosapi/ServiceResponseDetailsResponse'
'rosapi/ServiceType'
'rosapi/ServiceTypeRequest'
'rosapi/ServiceTypeResponse'
'rosapi/Services'
'rosapi/ServicesRequest'
'rosapi/ServicesResponse'
'rosapi/SetParam'
'rosapi/SetParamRequest'
'rosapi/SetParamResponse'
'rosapi/Subscribers'
```

```
'rosapi/SubscribersRequest'
'rosapi/SubscribersResponse'
'rosapi/TopicType'
'rosapi/TopicTypeRequest'
'rosapi/TopicTypeResponse'
'rosapi/Topics'
'rosapi/TopicsForType'
'rosapi/TopicsForTypeRequest'
'rosapi/TopicsForTypeResponse'
'rosapi/TopicsRequest'
'rosapi/TopicsResponse'
'rosapi/TypeDef'
'rosauth/Authentication'
'rosauth/AuthenticationRequest'
'rosauth/AuthenticationResponse'
'rosbridge_library/AddTwoInts'
'rosbridge_library/AddTwoIntsRequest'
'rosbridge_library/AddTwoIntsResponse'
'rosbridge_library/Num'
'rosbridge_library/SendBytes'
'rosbridge_library/SendBytesRequest'
'rosbridge_library/SendBytesResponse'
'rosbridge_library/TestArrayRequest'
'rosbridge_library/TestArrayRequestRequest'
'rosbridge_library/TestArrayRequestResponse'
'rosbridge_library/TestChar'
'rosbridge_library/TestDurationArray'
'rosbridge_library/TestEmpty'
'rosbridge_library/TestEmptyRequest'
'rosbridge_library/TestEmptyResponse'
'rosbridge_library/TestHeader'
'rosbridge_library/TestHeaderArray'
'rosbridge_library/TestHeaderTwo'
'rosbridge_library/TestMultipleRequestFields'
'rosbridge_library/TestMultipleRequestFieldsRequest'
'rosbridge_library/TestMultipleRequestFieldsResponse'
'rosbridge_library/TestMultipleResponseFields'
'rosbridge_library/TestMultipleResponseFieldsRequest'
'rosbridge_library/TestMultipleResponseFieldsResponse'
'rosbridge_library/TestNestedService'
'rosbridge_library/TestNestedServiceRequest'
'rosbridge_library/TestNestedServiceResponse'
'rosbridge_library/TestRequestAndResponse'
'rosbridge_library/TestRequestAndResponseRequest'
```

```
'rosbridge_library/TestRequestAndResponseResponse'
'rosbridge_library/TestRequestOnly'
'rosbridge_library/TestRequestOnlyRequest'
'rosbridge_library/TestRequestOnlyResponse'
'rosbridge_library/TestResponseOnly'
'rosbridge_library/TestResponseOnlyRequest'
'rosbridge_library/TestResponseOnlyResponse'
'rosbridge_library/TestTimeArray'
'rosbridge_library/TestUInt8'
'rosbridge_library/TestUInt8FixedSizeArray16'
'roscpp/Empty'
'roscpp/EmptyRequest'
'roscpp/EmptyResponse'
'roscpp/GetLoggers'
'roscpp/GetLoggersRequest'
'roscpp/GetLoggersResponse'
'roscpp/Logger'
'roscpp/SetLoggerLevel'
'roscpp/SetLoggerLevelRequest'
'roscpp/SetLoggerLevelResponse'
'roscpp_tutorials/TwoInts'
'roscpp_tutorials/TwoIntsRequest'
'roscpp_tutorials/TwoIntsResponse'
'roseus/AddTwoInts'
'roseus/AddTwoIntsRequest'
'roseus/AddTwoIntsResponse'
'roseus/String'
'roseus/StringStamped'
'roseus/StringString'
'roseus/StringStringRequest'
'roseus/StringStringResponse'
'rosgraph_msgs/Clock'
'rosgraph_msgs/Log'
'rospy_message_converter/TestArray'
'rospy_tutorials/AddTwoInts'
'rospy_tutorials/AddTwoIntsRequest'
'rospy_tutorials/AddTwoIntsResponse'
'rospy_tutorials/BadTwoInts'
'rospy_tutorials/BadTwoIntsRequest'
'rospy_tutorials/BadTwoIntsResponse'
'rospy_tutorials/Floats'
'rospy_tutorials/HeaderString'
'rosruby_tutorials/TwoInts'
'rosruby_tutorials/TwoIntsRequest'
```

```
'rosruby_tutorials/TwoIntsResponse'
'rosserial_arduino/Adc'
'rosserial_arduino/Test'
'rosserial_arduino/TestRequest'
'rosserial_arduino/TestResponse'
'rosserial_msgs/Log'
'rosserial_msgs/RequestMessageInfo'
'rosserial_msgs/RequestMessageInfoRequest'
'rosserial_msgs/RequestMessageInfoResponse'
'rosserial_msgs/RequestParam'
'rosserial_msgs/RequestParamRequest'
'rosserial_msgs/RequestParamResponse'
'rosserial_msgs/TopicInfo'
'rovio_shared/head_ctrl'
'rovio_shared/head_ctrlRequest'
'rovio_shared/head_ctrlResponse'
'rovio_shared/man_drv'
'rovio_shared/wav_play'
'rovio_shared/wav_playRequest'
'rovio_shared/wav_playResponse'
'rtt_ros_msgs/GetPeerList'
'rtt_ros_msgs/GetPeerListRequest'
'rtt_ros_msgs/GetPeerListResponse'
'rtt_ros_msgs/RunScript'
'rtt_ros_msgs/RunScriptRequest'
'rtt_ros_msgs/RunScriptResponse'
's3000_laser/enable_disable'
's3000_laser/enable_disableRequest'
's3000_laser/enable_disableResponse'
'saphari_msgs/BodyPart'
'saphari_msgs/Equipment'
'saphari_msgs/Human'
'saphari_msgs/PerceiveEquipment'
'saphari_msgs/PerceiveEquipmentRequest'
'saphari_msgs/PerceiveEquipmentResponse'
'scheduler_msgs/CurrentStatus'
'scheduler_msgs/KnownResources'
'scheduler_msgs/Request'
'scheduler_msgs/Resource'
'scheduler_msgs/SchedulerRequests'
'schunk_sdh/TactileMatrix'
'schunk_sdh/TactileSensor'
'segbot_gui/QuestionDialog'
'segbot_gui/QuestionDialogRequest'
```

```
'segbot_gui/QuestionDialogResponse'
'segbot_sensors/RangeArray'
'segbot_simulation_apps/DoorHandlerInterface'
'segbot_simulation_apps/DoorHandlerInterfaceRequest'
'segbot_simulation_apps/DoorHandlerInterfaceResponse'
'segway_rmp/SegwayStatus'
'segway_rmp/SegwayStatusStamped'
'sensor_msgs/CameraInfo'
'sensor_msgs/ChannelFloat32'
'sensor_msgs/CompressedImage'
'sensor_msgs/FluidPressure'
'sensor_msgs/Illuminance'
'sensor_msgs/Image'
'sensor_msgs/Imu'
'sensor_msgs/JointState'
'sensor_msgs/Joy'
'sensor_msgs/JoyFeedback'
'sensor_msgs/JoyFeedbackArray'
'sensor_msgs/LaserEcho'
'sensor_msgs/LaserScan'
'sensor_msgs/MagneticField'
'sensor_msgs/MultiDOFJointState'
'sensor_msgs/MultiEchoLaserScan'
'sensor_msgs/NavSatFix'
'sensor_msgs/NavSatStatus'
'sensor_msgs/PointCloud'
'sensor_msgs/PointCloud2'
'sensor_msgs/PointField'
'sensor_msgs/Range'
'sensor_msgs/RegionOfInterest'
'sensor_msgs/RelativeHumidity'
'sensor_msgs/SetCameraInfo'
'sensor_msgs/SetCameraInfoRequest'
'sensor_msgs/SetCameraInfoResponse'
'sensor_msgs/Temperature'
'sensor_msgs/TimeReference'
'shape_msgs/Mesh'
'shape_msgs/MeshTriangle'
'shape_msgs/Plane'
'shape_msgs/SolidPrimitive'
'shared_serial/Close'
'shared_serial/Connect'
'shared_serial/ConnectRequest'
'shared_serial/ConnectResponse'
```

```
'shared_serial/Flush'
'shared_serial/Recv'
'shared_serial/RecvRequest'
'shared_serial/RecvResponse'
'shared_serial/Send'
'shared_serial/SendRecv'
'shared_serial/SendRecvRequest'
'shared_serial/SendRecvResponse'
'shared_serial/SendTo'
'shared_serial/SendToRequest'
'shared_serial/SendToResponse'
'sherlock_sim_msgs/TestService'
'sherlock_sim_msgs/TestServiceRequest'
'sherlock_sim_msgs/TestServiceResponse'
'simple_robot_control/ReturnJointStates'
'simple_robot_control/ReturnJointStatesRequest'
'simple_robot_control/ReturnJointStatesResponse'
'smach_msgs/SmachContainerInitialStatusCmd'
'smach_msgs/SmachContainerStatus'
'smach_msgs/SmachContainerStructure'
'sound_play/SoundRequest'
'speech_recognition_msgs/SpeechRecognitionCandidates'
'sr_edc_ethercat_drivers/ActuatorInfo'
'sr_edc_ethercat_drivers/MotorTrace'
'sr_edc_ethercat_drivers/MotorTraceSample'
'sr_robot_msgs/AuxSpiData'
'sr_robot_msgs/Biotac'
'sr_robot_msgs/BiotacAll'
'sr_robot_msgs/ChangeControlType'
'sr_robot_msgs/ChangeControlTypeRequest'
'sr_robot_msgs/ChangeControlTypeResponse'
'sr_robot_msgs/ChangeMotorSystemControls'
'sr_robot_msgs/ChangeMotorSystemControlsRequest'
'sr_robot_msgs/ChangeMotorSystemControlsResponse'
'sr_robot_msgs/ControlType'
'sr_robot_msgs/EthercatDebug'
'sr_robot_msgs/ForceController'
'sr_robot_msgs/ForceControllerRequest'
'sr_robot_msgs/ForceControllerResponse'
'sr_robot_msgs/FromMotorDataType'
'sr_robot_msgs/GetSegmentedLine'
'sr_robot_msgs/GetSegmentedLineRequest'
'sr_robot_msgs/GetSegmentedLineResponse'
'sr_robot_msgs/JointControllerState'
```

```
'sr_robot_msgs/JointMusclePositionControllerState'
'sr_robot_msgs/JointMuscleValveControllerCommand'
'sr_robot_msgs/JointMuscleValveControllerState'
'sr_robot_msgs/ManualSelfTest'
'sr_robot_msgs/ManualSelfTestRequest'
'sr_robot_msgs/ManualSelfTestResponse'
'sr_robot_msgs/MidProxData'
'sr_robot_msgs/MidProxDataAll'
'sr_robot_msgs/MotorSystemControls'
'sr_robot_msgs/NullifyDemand'
'sr_robot_msgs/NullifyDemandRequest'
'sr_robot_msgs/NullifyDemandResponse'
'sr_robot_msgs/SetDebugData'
'sr_robot_msgs/SetDebugDataRequest'
'sr_robot_msgs/SetDebugDataResponse'
'sr_robot_msgs/SetEffortControllerGains'
'sr_robot_msgs/SetEffortControllerGainsRequest'
'sr_robot_msgs/SetEffortControllerGainsResponse'
'sr_robot_msgs/SetMixedPositionVelocityPidGains'
'sr_robot_msgs/SetMixedPositionVelocityPidGainsRequest'
'sr_robot_msgs/SetMixedPositionVelocityPidGainsResponse'
'sr_robot_msgs/SetPidGains'
'sr_robot_msgs/SetPidGainsRequest'
'sr_robot_msgs/SetPidGainsResponse'
'sr_robot_msgs/ShadowPST'
'sr_robot_msgs/SimpleMotorFlasher'
'sr_robot_msgs/SimpleMotorFlasherRequest'
'sr_robot_msgs/SimpleMotorFlasherResponse'
'sr_robot_msgs/Tactile'
'sr_robot_msgs/TactileArray'
'sr_robot_msgs/UBIO'
'sr_robot_msgs/UBIOAll'
'sr_robot_msgs/cartesian_data'
'sr_robot_msgs/cartesian_position'
'sr_robot_msgs/command'
'sr_robot_msgs/config'
'sr_robot_msgs/contrlr'
'sr_robot_msgs/is_hand_occupied'
'sr_robot_msgs/is_hand_occupiedRequest'
'sr_robot_msgs/is_hand_occupiedResponse'
'sr_robot_msgs/joint'
'sr_robot_msgs/joints_data'
'sr_robot_msgs/reverseKinematics'
'sr_robot_msgs/sendupdate'
```

```
'sr_robot_msgs/which_fingers_are_touching'
'sr_robot_msgs/which_fingers_are_touchingRequest'
'sr_robot_msgs/which_fingers_are_touchingResponse'
'sr_ronex_msgs/BoolArray'
'sr_ronex_msgs/GeneralIOState'
'sr_ronex_msgs/ImpulseSample'
'sr_ronex_msgs/PWM'
'sr_ronex_msgs/ReceiverData'
'sr_ronex_msgs/SPI'
'sr_ronex_msgs/SPIPacketIn'
'sr_ronex_msgs/SPIRequest'
'sr_ronex_msgs/SPIResponse'
'sr_ronex_msgs/SPIState'
'sr_ronex_msgs/TCATState'
'statistics_msgs/Stats1D'
'std_msgs/Bool'
'std_msgs/Byte'
'std_msgs/ByteMultiArray'
'std_msgs/Char'
'std_msgs/ColorRGBA'
'std_msgs/Duration'
'std_msgs/Empty'
'std_msgs/Float32'
'std_msgs/Float32MultiArray'
'std_msgs/Float64'
'std_msgs/Float64MultiArray'
'std_msgs/Header'
'std_msgs/Int16'
'std_msgs/Int16MultiArray'
'std_msgs/Int32'
'std_msgs/Int32MultiArray'
'std_msgs/Int64'
'std_msgs/Int64MultiArray'
'std_msgs/Int8'
'std_msgs/Int8MultiArray'
'std_msgs/MultiArrayDimension'
'std_msgs/MultiArrayLayout'
'std_msgs/String'
'std_msgs/Time'
'std_msgs/UInt16'
'std_msgs/UInt16MultiArray'
'std_msgs/UInt32'
'std_msgs/UInt32MultiArray'
'std_msgs/UInt64'
```

```
'std_msgs/UInt64MultiArray'
'std_msgs/UInt8'
'std_msgs/UInt8MultiArray'
'std_srvs/Empty'
'std_srvs/EmptyRequest'
'std_srvs/EmptyResponse'
'stdr_msgs/AddCO2Source'
'stdr_msgs/AddCO2SourceRequest'
'stdr_msgs/AddCO2SourceResponse'
'stdr_msgs/AddRfidTag'
'stdr_msgs/AddRfidTagRequest'
'stdr_msgs/AddRfidTagResponse'
'stdr_msgs/AddSoundSource'
'stdr_msgs/AddSoundSourceRequest'
'stdr_msgs/AddSoundSourceResponse'
'stdr_msgs/AddThermalSource'
'stdr_msgs/AddThermalSourceRequest'
'stdr_msgs/AddThermalSourceResponse'
'stdr_msgs/CO2SensorMeasurementMsg'
'stdr_msgs/CO2SensorMsg'
'stdr_msgs/CO2Source'
'stdr_msgs/CO2SourceVector'
'stdr_msgs/DeleteCO2Source'
'stdr_msgs/DeleteCO2SourceRequest'
'stdr_msgs/DeleteCO2SourceResponse'
'stdr_msgs/DeleteRfidTag'
'stdr_msgs/DeleteRfidTagRequest'
'stdr_msgs/DeleteRfidTagResponse'
'stdr_msgs/DeleteSoundSource'
'stdr_msgs/DeleteSoundSourceRequest'
'stdr_msgs/DeleteSoundSourceResponse'
'stdr_msgs/DeleteThermalSource'
'stdr_msgs/DeleteThermalSourceRequest'
'stdr_msgs/DeleteThermalSourceResponse'
'stdr_msgs/FootprintMsg'
'stdr_msgs/KinematicMsg'
'stdr_msgs/LaserSensorMsg'
'stdr_msgs/LoadExternalMap'
'stdr_msgs/LoadExternalMapRequest'
'stdr_msgs/LoadExternalMapResponse'
'stdr_msgs/LoadMap'
'stdr_msgs/LoadMapRequest'
'stdr_msgs/LoadMapResponse'
'stdr_msgs/MoveRobot'
```

```
'stdr_msgs/MoveRobotRequest'
'stdr_msgs/MoveRobotResponse'
'stdr_msgs/Noise'
'stdr_msgs/RegisterGui'
'stdr_msgs/RegisterGuiRequest'
'stdr_msgs/RegisterGuiResponse'
'stdr_msgs/RfidSensorMeasurementMsg'
'stdr_msgs/RfidSensorMsg'
'stdr_msgs/RfidTag'
'stdr_msgs/RfidTagVector'
'stdr_msgs/RobotIndexedMsg'
'stdr_msgs/RobotIndexedVectorMsg'
'stdr_msgs/RobotMsg'
'stdr_msgs/SonarSensorMsg'
'stdr_msgs/SoundSensorMeasurementMsg'
'stdr_msgs/SoundSensorMsg'
'stdr_msgs/SoundSource'
'stdr_msgs/SoundSourceVector'
'stdr_msgs/ThermalSensorMeasurementMsg'
'stdr_msgs/ThermalSensorMsg'
'stdr_msgs/ThermalSource'
'stdr_msgs/ThermalSourceVector'
'stereo_msgs/DisparityImage'
'stereo_wall_detection/DetectWall'
'stereo_wall_detection/DetectWallRequest'
'stereo_wall_detection/DetectWallResponse'
'tf/FrameGraph'
'tf/FrameGraphRequest'
'tf/FrameGraphResponse'
'tf/tfMessage'
'tf2_msgs/FrameGraph'
'tf2_msgs/FrameGraphRequest'
'tf2_msgs/FrameGraphResponse'
'tf2_msgs/TF2Error'
'tf2_msgs/TFMessage'
'theora_image_transport/Packet'
'topic_proxy/AddPublisher'
'topic_proxy/AddPublisherRequest'
'topic_proxy/AddPublisherResponse'
'topic_proxy/GetMessage'
'topic_proxy/GetMessageRequest'
'topic_proxy/GetMessageResponse'
'topic_proxy/MessageInstance'
'topic_proxy/PublishMessage'
```

```
'topic_proxy/PublishMessageRequest'
'topic_proxy/PublishMessageResponse'
'topic_proxy/RequestMessage'
'topic_proxy/RequestMessageRequest'
'topic_proxy/RequestMessageResponse'
'topic_tools/MuxAdd'
'topic_tools/MuxAddRequest'
'topic_tools/MuxAddResponse'
'topic_tools/MuxDelete'
'topic_tools/MuxDeleteRequest'
'topic_tools/MuxDeleteResponse'
'topic_tools/MuxList'
'topic_tools/MuxListRequest'
'topic_tools/MuxListResponse'
'topic_tools/MuxSelect'
'topic_tools/MuxSelectRequest'
'topic_tools/MuxSelectResponse'
'trajectory_msgs/JointTrajectory'
'trajectory_msgs/JointTrajectoryPoint'
'trajectory_msgs/MultiDOFJointTrajectory'
'trajectory_msgs/MultiDOFJointTrajectoryPoint'
'turtle_actionlib/Velocity'
'turtlebot_calibration/ScanAngle'
'turtlebot_msgs/PanoramaImg'
'turtlebot_msgs/SetFollowState'
'turtlebot_msgs/SetFollowStateRequest'
'turtlebot_msgs/SetFollowStateResponse'
'turtlebot_msgs/TakePanorama'
'turtlebot_msgs/TakePanoramaRequest'
'turtlebot_msgs/TakePanoramaResponse'
'turtlesim/Color'
'turtlesim/Kill'
'turtlesim/KillRequest'
'turtlesim/KillResponse'
'turtlesim/Pose'
'turtlesim/SetPen'
'turtlesim/SetPenRequest'
'turtlesim/SetPenResponse'
'turtlesim/Spawn'
'turtlesim/SpawnRequest'
'turtlesim/SpawnResponse'
'turtlesim/TeleportAbsolute'
'turtlesim/TeleportAbsoluteRequest'
'turtlesim/TeleportAbsoluteResponse'
```

```
'turtlesim/TeleportRelative'
'turtlesim/TeleportRelativeRequest'
'turtlesim/TeleportRelativeResponse'
'um6/Reset'
'um6/ResetRequest'
'um6/ResetResponse'
'underwater_sensor_msgs/DVL'
'underwater_sensor_msgs/Pressure'
'universal_teleop/Control'
'universal_teleop/Event'
'uuid_msgs/UniqueID'
'velodyne_msgs/VelodynePacket'
'velodyne_msgs/VelodyneScan'
'view_controller_msgs/CameraPlacement'
'visp_camera_calibration/CalibPoint'
'visp_camera_calibration/CalibPointArray'
'visp_camera_calibration/ImageAndPoints'
'visp_camera_calibration/ImagePoint'
'visp_camera_calibration/calibrate'
'visp_camera_calibration/calibrateRequest'
'visp_camera_calibration/calibrateResponse'
'visp_hand2eye_calibration/TransformArray'
'visp_hand2eye_calibration/compute_effector_camera'
'visp_hand2eye_calibration/compute_effector_cameraRequest'
'visp_hand2eye_calibration/compute_effector_cameraResponse'
'visp_hand2eye_calibration/compute_effector_camera_quick'
'visp_hand2eye_calibration/compute_effector_camera_quickRequest'
'visp_hand2eye_calibration/compute_effector_camera_quickResponse'
'visp_hand2eye_calibration/reset'
'visp_hand2eye_calibration/resetRequest'
'visp_hand2eye_calibration/resetResponse'
'visp_tracker/Init'
'visp_tracker/InitRequest'
'visp_tracker/InitResponse'
'visp_tracker/KltPoint'
'visp_tracker/KltPoints'
'visp_tracker/KltSettings'
'visp_tracker/MovingEdgeSettings'
'visp_tracker/MovingEdgeSite'
'visp_tracker/MovingEdgeSites'
'visualization_msgs/ImageMarker'
'visualization_msgs/InteractiveMarker'
'visualization_msgs/InteractiveMarkerControl'
'visualization_msgs/InteractiveMarkerFeedback'
```

```
'visualization_msgs/InteractiveMarkerInit'
'visualization_msgs/InteractiveMarkerPose'
'visualization_msgs/InteractiveMarkerUpdate'
'visualization_msgs/Marker'
'visualization_msgs/MarkerArray'
'visualization_msgs/MenuEntry'
'wfov_camera_msgs/WFOVCompressedImage'
'wfov_camera_msgs/WFOVImage'
'wfov_camera_msgs/WFOVTrigger'
'wge100_camera/BoardConfig'
'wge100_camera/BoardConfigRequest'
'wge100_camera/BoardConfigResponse'
'wifi_ddwrt/Network'
'wifi_ddwrt/SiteSurvey'
'wireless_msgs/Connection'
'wireless_msgs/Network'
'wireless_msgs/Quality'
'wireless_msgs/Scan'
'yocs_msgs/Column'
'yocs_msgs/ColumnList'
'yocs_msgs/Wall'
'yocs_msgs/WallList'
'zeroconf_msgs/AddListener'
'zeroconf_msgs/AddListenerRequest'
'zeroconf_msgs/AddListenerResponse'
'zeroconf_msgs/AddService'
'zeroconf_msgs/AddServiceRequest'
'zeroconf_msgs/AddServiceResponse'
'zeroconf_msgs/DiscoveredService'
'zeroconf_msgs/ListDiscoveredServices'
'zeroconf_msgs/ListDiscoveredServicesRequest'
'zeroconf_msgs/ListDiscoveredServicesResponse'
'zeroconf_msgs/ListPublishedServices'
'zeroconf_msgs/ListPublishedServicesRequest'
'zeroconf_msgs/ListPublishedServicesResponse'
'zeroconf_msgs/Protocols'
'zeroconf_msgs/PublishedService'
'zeroconf_msgs/RemoveListener'
'zeroconf_msgs/RemoveListenerRequest'
'zeroconf_msgs/RemoveListenerResponse'
'zeroconf_msgs/RemoveService'
'zeroconf_msgs/RemoveServiceRequest'
'zeroconf_msgs/RemoveServiceResponse'
```

See Also

`rosmessage` | `rosmsg` | `showdetails`

Related Examples

- "Work with Basic ROS Messages"
- "Exchange Data with ROS Publishers and Subscribers"

4

ROS Log Files and Transformations

ROS Log Files (rosbags)

In this section...

Introduction

A rosbag or bag is a file format in ROS for storing ROS message data. These bags are often created by subscribing to one or more ROS topics, and storing the received message data in an efficient file structure. MATLAB® can read these rosbag files and help with filtering and extracting message data. The following sections detail the structure of rosbags in MATLAB and the workflow for extracting data from them.

MATLAB rosbag Structure

When accessing rosbag log files, call `rosbag` and specify the file path to the object. MATLAB then creates a `BagSelection` object that contains an index of all the messages from the rosbag.

The `BagSelection` object has the following properties related to the rosbag:

- `FilePath`: a string of the absolute path to the rosbag file.
- `StartTime`: a scalar indicating the time the first message was recorded
- `EndTime`: a scalar indicating the time the last message was recorded
- `NumMessages`: a scalar indicating how many messages are contained in the file
- `AvailableTopics`: a list of what topic and message types were recorded in the bag. This is stored as table data that lists the number of messages, message type, and message definition for each topic. For more information on table data types, see "Access Data in a Table". Here is an example output of this table:

  ```
  ans =
  ```

```
                        NumMessages          MessageType              MessageDefinitio
                        ___________    ____________________________    ________________

    /clock                  12001          rosgraph_msgs/Clock         [1x185  char]
    /gazebo/link_states     11999          gazebo_msgs/LinkStates      [1x1247 char]
    /odom                   11998          nav_msgs/Odometry           [1x2918 char]
    /scan                     965          sensor_msgs/LaserScan       [1x2123 char]
```

- `MessageList`: a list of every message in the bag with rows sorted by time stamp of when the message was recorded. This list can be indexed and you can select a portion of the list this way. Calling `select` allows you to select subsets based on time stamp, topic or message type.

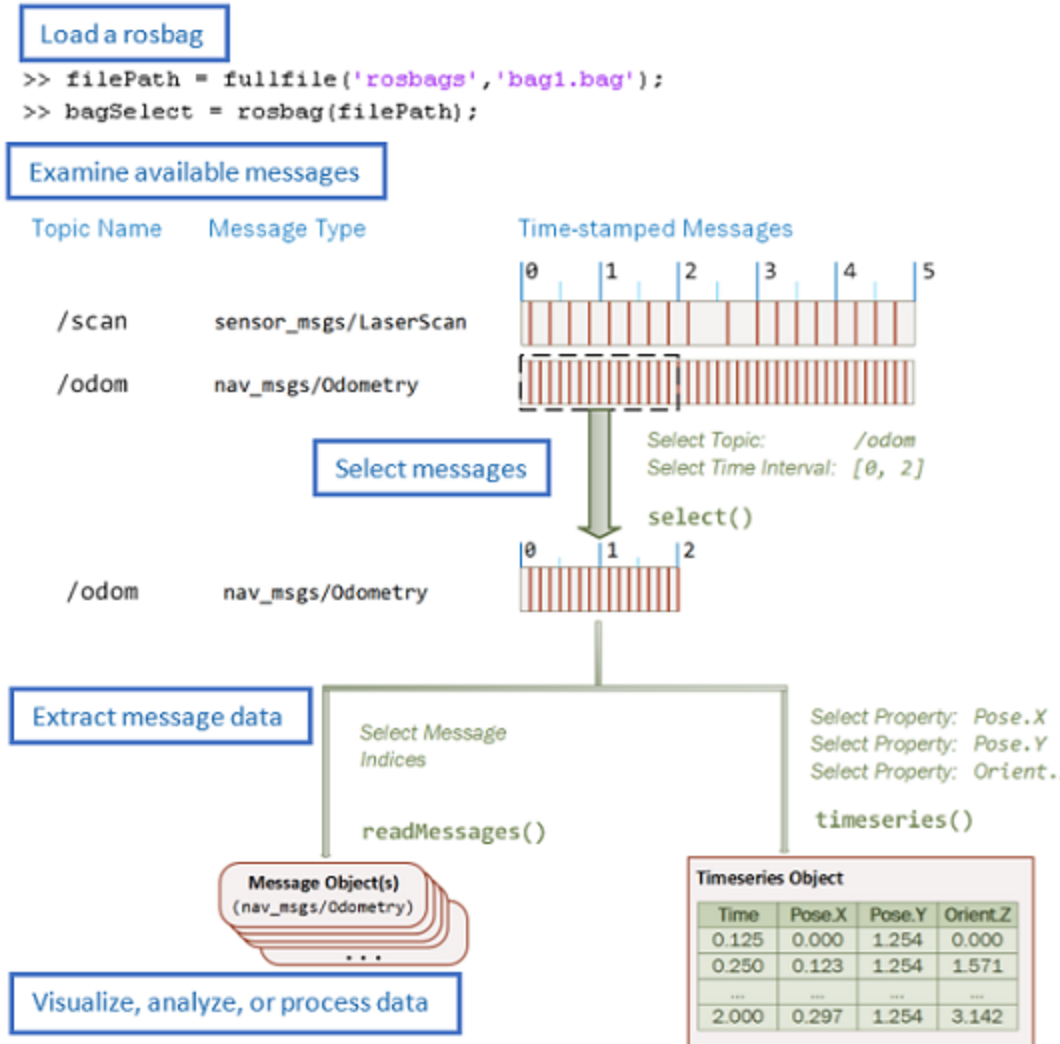
Also, note that the `BagSelection` object contains an index for all the messages. However, you must still use functions to extract the data. For extracting this information, see `readMessages` for getting messages based on indices as a cell array or see `timeseries` for reading the data of specified properties as a time series.

Workflow for rosbag Selection

When working with rosbags, there is a general procedure of how you should extract data.

- **Load a rosbag**: Call `rosbag` and the file path to load file and create `BagSelection`.
- **Examine available messages**: Examine `BagSelection` properties (`AvailableTopics`, `NumMessages`, `StartTime`, `EndTime`, and `MessageList`) to determine how to select a subset of messages for analysis.
- **Select messages**: Call `select` to create a selection of messages based on your desired properties.
- **Extract message data**: Call `readMessages` or `timeseries` to get message data as either a cell array or time series data structure.
- **Visualize, analyze or process data**: Use the extracted data for your specific application. You can plot data or develop algorithms to process data.

The following figure also shows the workflow.


Load a rosbag
>> filePath = fullfile('rosbags','bag1.bag');
>> bagSelect = rosbag(filePath);
Examine available messages
Topic Name
Message Type
Time-stamped Messages
/scan
sensor_msgs/LaserScan
/odom
nav_msgs/Odometry
0
1
2
3
4
5
Select messages
Select Topic: /odom
Select Time Interval: [0, 2]
select()
/odom
nav_msgs/Odometry
Extract message data
Select Message Indices
readMessages()
Select Property: Pose.X
Select Property: Pose.Y
Select Property: Orient.Z
timeseries()
Message Object(s)
(nav_msgs/Odometry)
Timeseries Object
Time
Pose.X
Pose.Y
Orient.Z
0.125
0.000
1.254
0.000
0.250
0.123
1.254
1.571
2.000
0.297
1.254
3.142
Visualize, analyze, or process data
Time Series Plot:/odom Properties
/odom Properties

Limitations

There are a few limitations in the rosbag support within MATLAB:

- MATLAB can only parse uncompressed rosbags. See the ROS Wiki for a tool to decompress a compressed rosbag.
- Only rosbags in the v2.0 format are supported. See the ROS Wiki for more information on different bag formats
- The file path to the rosbag must always be accessible. Because the message selection process does not retrieve any data, the file needs to be available for reading when the message data is accessed.

See Also

Using BagSelection Objects | `readMessages` | `rosbag`

Related Examples

- “Work with rosbag Logfiles”

5

ROS Custom Message Support

Create Custom Messages from ROS Package

In this example, you will go through the procedure for creating ROS custom messages in MATLAB. It assumes you have already gone through the installation process shown in "Install Robotics System Toolbox Add-ons" on page 5-13. Also, you must have a ROS package that contains the required `msg`, `srv`, and `package.xml` files. The correct file contents and folder structure are described in "Custom Message Contents" on page 5-8. This structure follows the standard ROS package conventions. Therefore, if you have any existing packages, they should match this structure.

It is recommended you start this procedure after opening a new MATLAB session to ensure that there are no lingering changes to MATLAB preferences from previous work. After ensuring that your custom message package is correct, note the folder path location. Then, call `rosgenmsg` with the specified path and follow the steps output in the command window. The following example has three messages, `A`, `B`, and `C`, that have dependencies on each other. It illustrates that you can use a folder containing multiple messages and generate them all at the same time.

Here are the first steps to setting up custom messages in MATLAB:

- Open MATLAB in a new session
- Place your custom message folder in a location and note the folder path. In this example, we provide a location of example packages and copy them to `userFolder`, which is then used for the custom message generation. Make sure that the `userFolder` directory exists prior to running this code.

  ```
  examplePackages = fullfile(fileparts(which('rosgenmsg')), 'examples', 'packages');
  userFolder = 'c:\MATLAB\custom_msg_test';
  copyfile(examplePackages, userFolder)
  ```

- Specify the folder path of the custom messages.

  ```
  folderpath = userFolder;
  ```

  (Optional) If you have an existing catkin workspace (`catkin_ws`), you can specify the path to its `src` folder instead. However, this workspace might contain a large number of packages and message generation will be run for all of them.

  ```
  folderpath = fullfile('catkin_ws','src');
  ```

- Specify the folder path for custom message files and call `rosgenmsg` to create custom messages for MATLAB.

```
rosgenmsg(folderpath)

Checking subfolder "A" for custom messages.

Checking subfolder "B" for custom messages.

Checking subfolder "C" for custom messages.

Building custom message files for the following packages:
   A
   B
   C


Generating MATLAB classes for message packages in
C:\MATLAB\custom_msgs\matlab_gen\jar

Loading file A-1.0.jar.
Generating MATLAB code for A/DependsOnB message type.
Generating MATLAB code for B/Standalone message type.

Loading file B-1.0.jar.

Loading file C-1.0.jar.
Generating MATLAB code for C/DependsOnB message type.

To use the custom messages, follow these steps:

1. Edit javaclasspath.txt, add the following file locations as new lines, and
save the file:

C:\MATLAB\custom_msgs\matlab_gen\jar\A-1.0.jar
C:\MATLAB\custom_msgs\matlab_gen\jar\B-1.0.jar
C:\MATLAB\custom_msgs\matlab_gen\jar\C-1.0.jar

2. Add the custom message folder to the MATLAB path by executing:

addpath('C:\MATLAB\custom_msgs\matlab_gen\msggen')
savepath

3. Restart MATLAB and verify that you can use the custom messages.
   Type "rosmsg list" and ensure that the output contains the generated
   custom message types.
```

Tip If you see the following warning

```
Objects of *** class exist - not clearing java
```

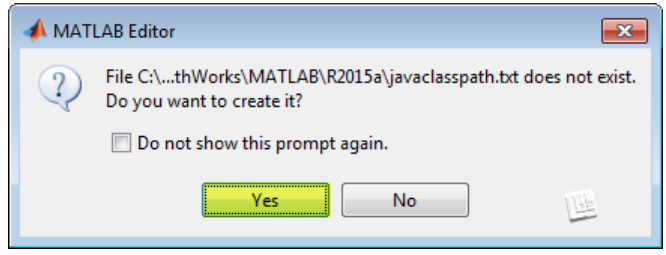
Try either calling `rosgenmsg` at the beginning of your MATLAB session or make sure that no Java objects are created with any startup functions called.

- You will then follow steps 1–3 from the output of `rosgenmsg`.

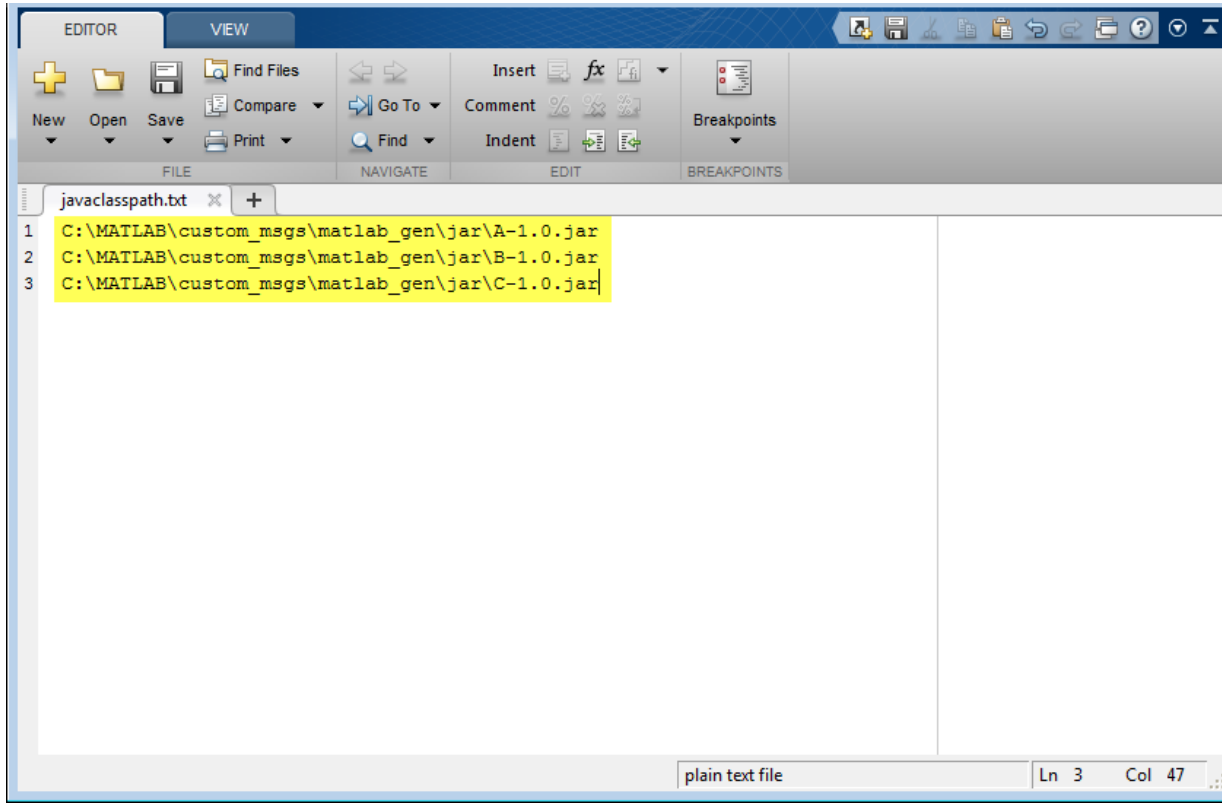
1 Edit javaclasspath.txt, add the following file locations as new lines, and save the file:

```
C:\MATLAB\custom_msgs\matlab_gen\jar\A-1.0.jar
C:\MATLAB\custom_msgs\matlab_gen\jar\B-1.0.jar
C:\MATLAB\custom_msgs\matlab_gen\jar\C-1.0.jar
```

Click the `javaclasspath.txt` link and it will open the file in the Editor. Copy and paste the different jar file locations as new lines in the file. If this file does not exist, you will be prompted to create it. Click **Yes** and then copy and paste the file locations into the file.



Here is what the `javaclasspath.txt` should look like after adding lines. Other paths may also already exist in this file:



2 Add the custom message folder to the MATLAB path by executing:

```
addpath('C:\MATLAB\custom_msgs\matlab_gen\msggen')
savepath
```

Add the given files to the MATLAB path by running `addpath` and `savepath` in the command window. You can either highlight the commands shown and press **F9** or copy and paste it into the MATLAB Command Window.

```
addpath('C:\MATLAB\custom_msgs\matlab_gen\msggen')
savepath
```

3 Restart MATLAB and verify that you can use the custom messages. Type "rosmsg list" and ensure that the output contains the generated custom message types.

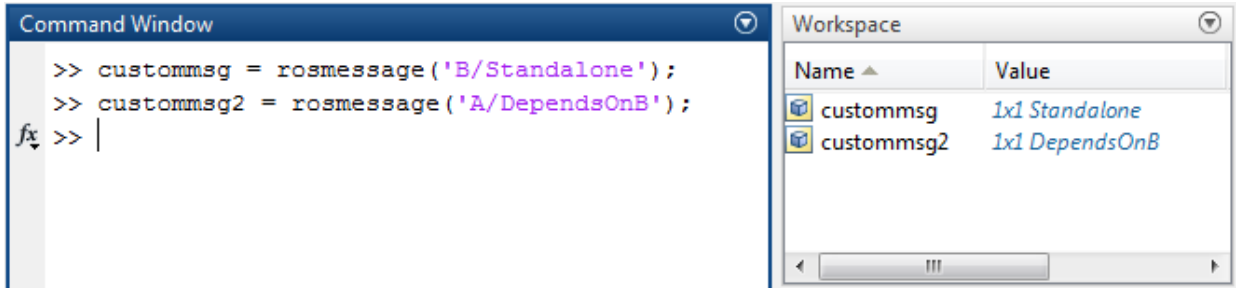
Restart MATLAB for the path changes to be applied. You can then use the custom messages like any other ROS messages supported in Robotics System Toolbox. Verify these changes by either calling `rosmsg list` and search for your message types, or use `rosmessage` to create a new message.

```
custommsg = rosmessage('B/Standalone')
```

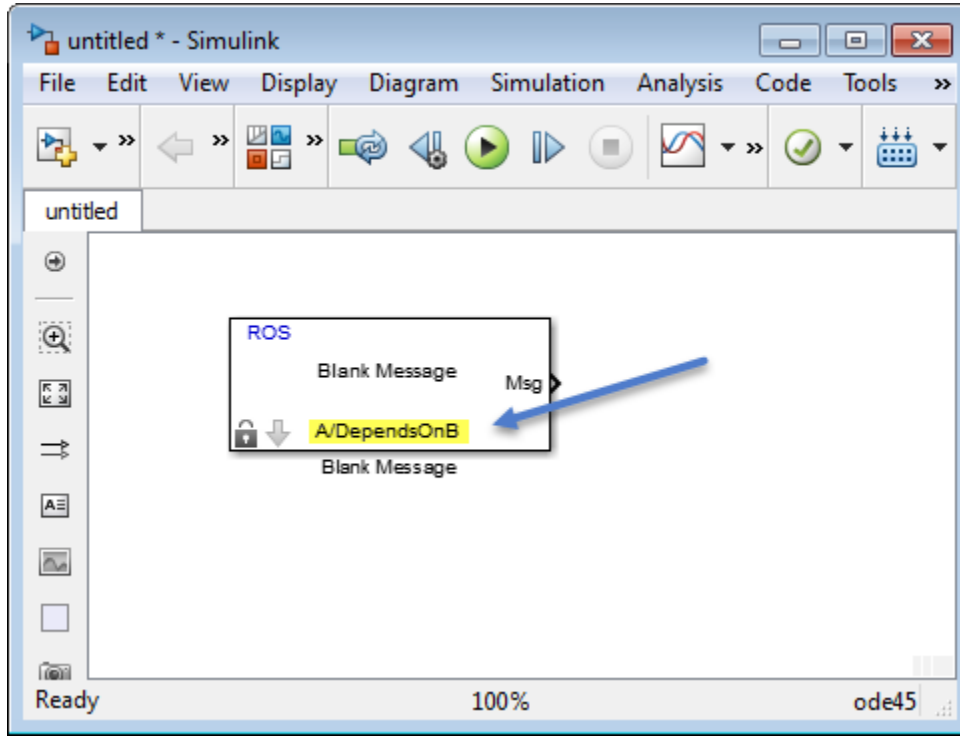
```
custommsg =

 ROS Standalone message with properties:

     MessageType: 'B/Standalone'
     IntProperty: 0
   StringPropert: ''

 Use showdetails to show the contents of the message
```

This final verification shows that you have performed the custom message generation process correctly. You can now send and receive these messages over a ROS network using MATLAB and Simulink®. The new custom messages can be used like normal message types. You should see them create objects specific to their message type and be displayed in your workspace.

```
custommsg = rosmessage('B/Standalone');
custommsg2 = rosmessage('A/DependsOnB');
```



Custom messages can also be used with the ROS Simulink blocks.



See Also

`roboticsAddons` | `rosgenmsg`

Related Examples

- "Install Robotics System Toolbox Add-ons" on page 5-13

ROS Custom Message Support

In this section...

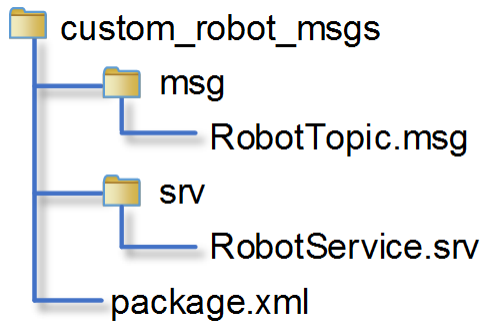
Custom Message Overview

Custom messages are user-defined messages that you can use to extend the set of message types currently supported in Robotics System Toolbox. If you are sending and receiving supported message types, you do not need to use custom messages. To see a list of supported message types, call `rosmsg list` in the MATLAB Command Window.

To install custom message support, call `roboticsAddons` and follow the instructions for installation. Custom message creation requires ROS packages, which are detailed in the ROS Wiki at Packages. After ensuring that you have valid ROS packages for custom messages, call `rosgenmsg` to generate the necessary MATLAB code to use custom messages. For an example on how to generate a ROS custom message in MATLAB, see“Create Custom Messages from ROS Package” on page 5-2.

Custom Message Contents

ROS custom messages are specified in ROS package folders that contain a `package.xml` file and optional `msg` and `srv` directories. The `msg` folder contains all your custom message type definitions. You should also add all custom service type definitions to the `srv` folder. For example, the package `custom_robot_msgs` has this folder and file structure.



The package contains one custom message type in `RobotTopic.msg` and one custom service type in `RobotService.srv`. MATLAB uses these files to generate the necessary files for using the custom messages contained in the package. For more information on creating `msg` and `srv` files, see Creating a ROS msg and srv and Defining Custom Messages on the ROS Wiki. The syntax of these files is described on the pages specific to msg and srv.

In all packages, you must define a `package.xml` file, which has the following contents:

- Name — `custom_robot_msgs`
- Version — `1.1.01`
- Dependency — `message_generation`
- Other dependencies on message packages (optional) — `geometry_msgs`, `std_msgs`

Here is a sample `package.xml` file with the previously shown contents.

```
<package>
 <name>custom_robot_msgs</name>
 <version>1.1.01</version>

 <build_depend>message_generation</build_depend>
 <build_depend>geometry_msgs</build_depend>
 <build_depend>std_msgs</build_depend>
</package>
```

Note:

- You must have write access to the custom messages folder.

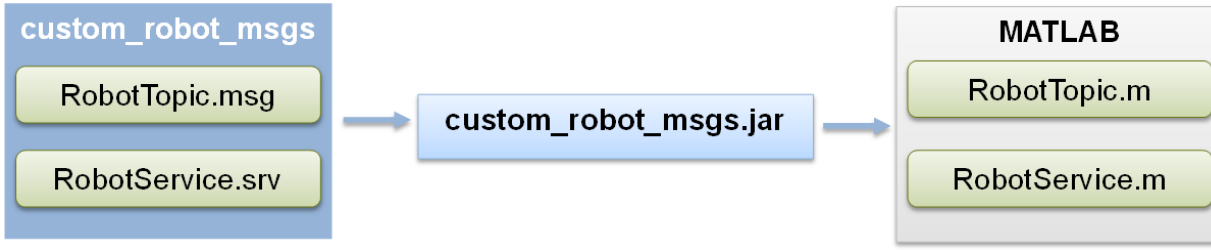
- At any time, there should only be one custom messages folder on the MATLAB path. This folder can contain multiple packages, but it is recommended that you keep them all in one unique folder.

- ROS actions are not currently supported and will be ignored during the custom message generation.
- ROS packages will not be processed if both of these conditions are met:
  - A package with the same name already exists
  - The version number of that existing package is the same

Custom Message Creation Workflow

Once you have your custom message structure set up as described in the previous section, you can create the code needed to use these custom messages. First, you call `rosgenmsg` with your known path to the custom message files to create MATLAB code.

Then, the two main creation steps that are handled by the `rosgenmsg` function. This function takes your custom message files (`.msg`, `.srv` and `package.xml`) and converts each message type to working MATLAB code. The `rosgenmsg` function will look for `.msg` files in the `msg` folder and for `.srv` files in the `srv` folder. This code is a group of classes that define the message properties when you create new custom messages. The basic procedure takes the custom message files and converts them to `.jar` files and then creates MATLAB program for each topic and service. Do not modify the `.jar` files because MATLAB uses them internally.



After the `rosgenmsg` function creates these files, you must add the files to the Java class path and the MATLAB path before you can use the custom messages. These steps are given as prompts in the MATLAB Command Window:

1. **Add location of files to `javaclasspath.txt`**: Add the specified paths as new lines of text in the `javaclasspath.txt` file. If this file does not exist, a message in the command window prompts you to create it. This text file defines the static class path for Java classes. For more information on the Java class path, see "Java Class Path".
2. **Add location of class files to MATLAB path**: Use `addpath` to add new locations of files with the .m extension to the MATLAB path and use `savepath` to save these changes.
3. **Restart MATLAB and verify messages are available**: After restarting MATLAB, call `rosmsg list` or `rosmessage` to check that you can use the messages as expected.

For an example of this procedure, see "Create Custom Messages from ROS Package" on page 5-2. This example uses sample custom message files to create custom messages in MATLAB.

You need to complete this procedure only once for a specific set of custom messages. After that, you can use the new custom messages like any other ROS message in MATLAB and take advantage of the full ROS functionality that Robotics System Toolbox provides. Repeat this generation procedure when you would like to update or create new message types.

You must maintain the Java class path and MATLAB path that contain the files directories. Make sure that the MATLAB path has only one folder at a time that contains custom message artifacts. Also, ensure you add the correct paths to the `javaclasspath.txt`, as the prompt directs. Do not modify the path. This file is used to load Java files at the start of each MATLAB session.

Sharing Custom Messages

After creating your custom message files, you can share them with other users. Other people do not need to call `rosgenmsg` to access your messages. Instead, to share your messages, access the `_matlab_gen` folder and follow the same three steps for specifying paths as described previously. If you have access to these files, either over a network or shared drive, add the `matlab_gen/jar` folder path to the `javaclasspath.txt` file and the `matlab_gen/msggen` path to the MATLAB path. After restarting MATLAB, other users can use the custom messages like any other ROS message.

Code Generation with Custom messages

Custom message and service types can be used with ROS Simulink blocks for generating C++ code for a standalone ROS node. The generated code (`.tgz` archive) will include Simulink definitions for the custom messages, but it will not include the ROS custom message packages. When the generated code is built in the destination Linux System, it expects the custom message packages to be available in the catkin workspace or on the `ROS_PACKAGE_PATH`. Please ensure that you either install or copy the custom message package to your Linux system before building the generated code.

See Also

`roboticsAddons` | `rosgenmsg`

Related Examples

- “Install Robotics System Toolbox Add-ons” on page 5-13
- “Create Custom Messages from ROS Package” on page 5-2

Install Robotics System Toolbox Add-ons

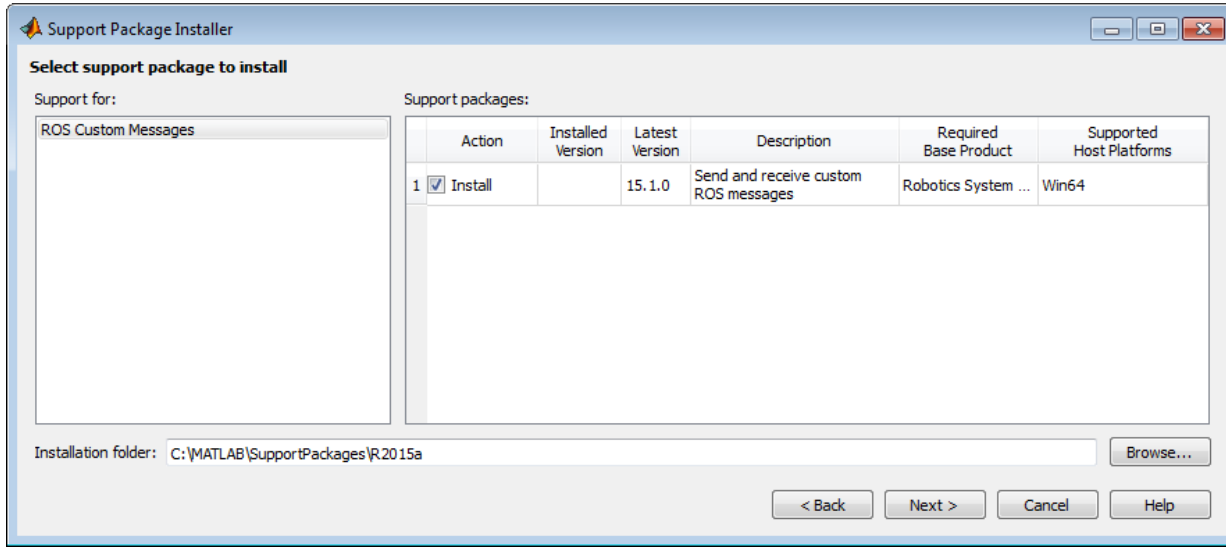
Robotics System Toolbox Add-ons enable you to generate ROS custom messages to send and receive in MATLAB.

1. In a MATLAB Command Window, type:

   ```
   roboticsAddons
   ```

2. Follow the instructions for installation. For more information about the options on a particular screen, click **Help**.
3. At **Select support package to install**, select the component you want to add. For example:

   - `ROS Custom Messages`



4. Accept or change the installation folder and click **Next**.

   Note: You must have write privileges for the installation folder.

5. Follow the remaining prompts to download and install the add-on.

When a new version of MATLAB software is released, repeat this process to check for updates. You can also check for updates between releases.

Related Examples

- “Create Custom Messages from ROS Package” on page 5-2

More About

- “ROS Custom Message Support” on page 5-8

6

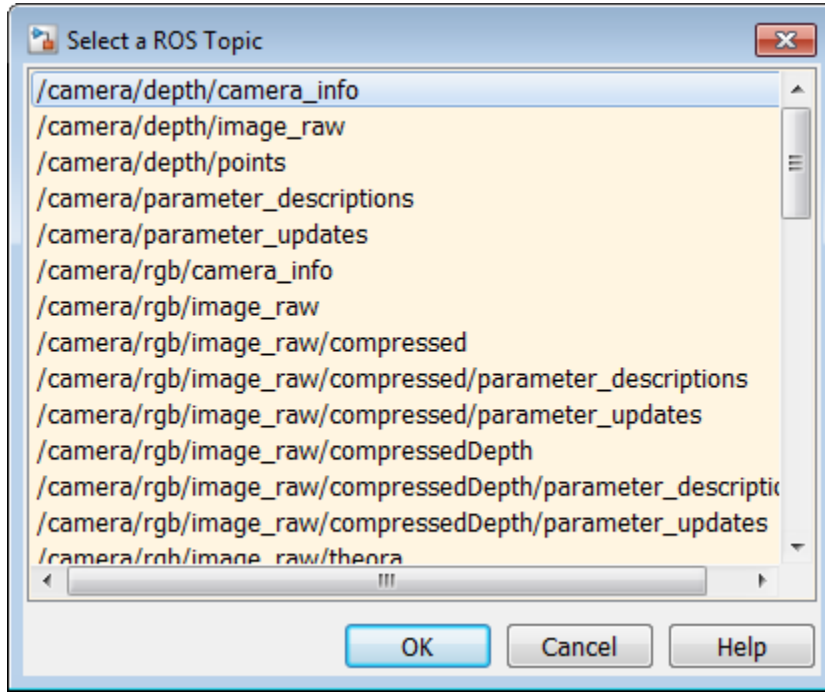
Simulink ROS Concepts

Selecting ROS Topics, Messages, and Parameters

In this section...

Selecting ROS Topics

When using Simulink with ROS, you can publish or subscribe to topics on the ROS network. In the dialog boxes for the `Publish` and `Subscribe` blocks, you can select from a list of topics on the ROS network. You must be currently connected to a ROS network to get a list of topics. You can select a topic using the following:



This dialog shows the list of topics available on the ROS master. Selecting a topic from the list automatically populates the **Topic** and **Message type** parameters for the

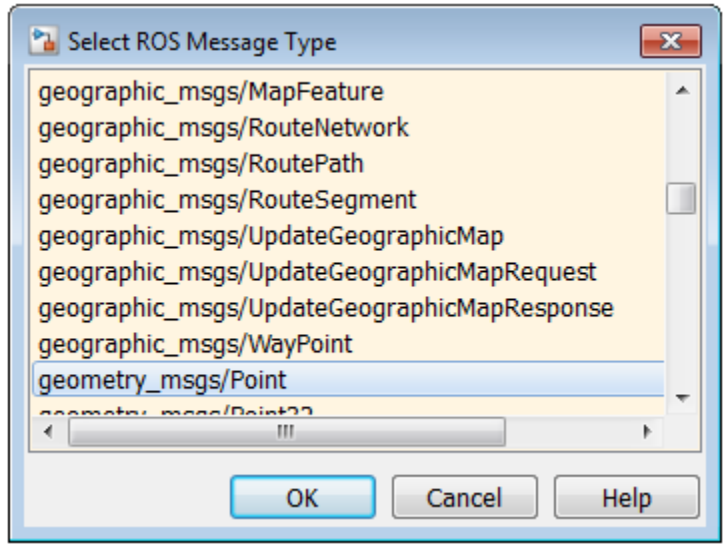
corresponding block mask dialog. If the message type is not supported in MATLAB ROS, Simulink will throw an error. Once the topic is selected, it is saved with the block. Even if the topic is not longer available on the network, the block will still use that topic name.

To refresh the list, close and open the dialog again.

To use a topic not currently posted on the ROS network or if you are not currently connected, use the "Specify your own" option under the **Topic Source** parameter in your block mask dialog.

Selecting ROS Message Types

Simulink ROS allows you to select from a list of message types currently supported by MATLAB ROS when setting the **Message type** for `Publish`, `Subscribe`, or `BlankMessage` blocks.



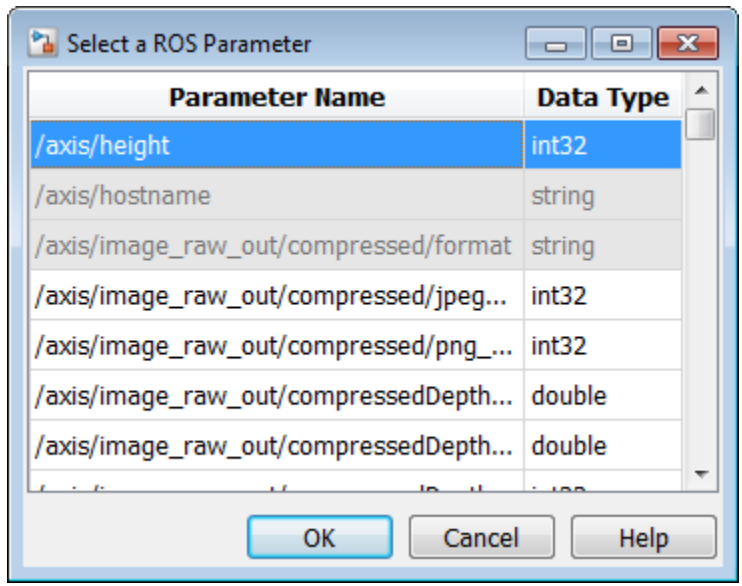
This is the list of all message types supported in MATLAB ROS including any custom message types. You can begin typing in the name of your desired message type or manually search through the list.

The selected message type is stored with the block and saved with the model.

Note: When using code generation, message type information is not included. You must ensure that your Linux ROS environment has the ROS packages installed that contain the necessary message type definitions.

Selecting ROS Parameter Names

When using the `Get Parameter` and `Set Parameter` blocks, you have the option of "Select from ROS Network" in the block parameters, which gets a list of parameters currently on the server. When clicking **Select**, you should see this dialog box.



This is the list of parameters you can select from the ROS parameter server. The parameters that are grayed out have unsupported data types. Select a parameter name that is not grayed out and click **OK**. This should auto-fill the **Name** and **Data type** into the block parameters.

See Also

`Blank Message` | `Get Parameter` | `Publish` | `Set Parameter` | `Subscribe`

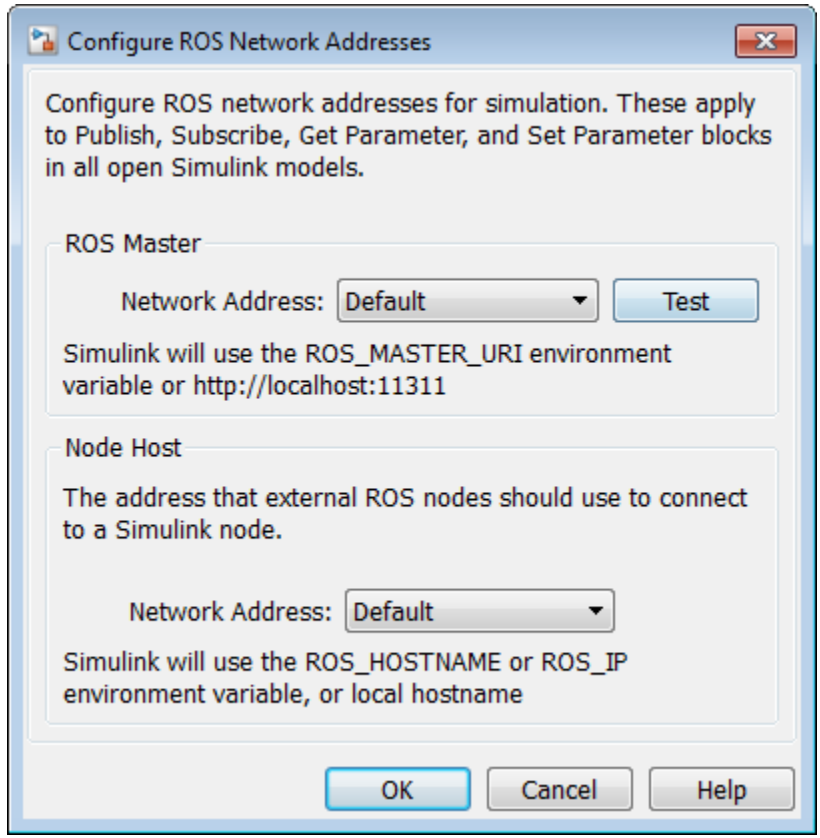
Related Examples

- “ROS Parameters in Simulink” on page 6-14

- “Managing Array Sizes in Simulink ROS” on page 6-10

Configure ROS Network Addresses

During model initialization, Simulink connects to a ROS master and also creates a node associated with the model. The ROS master URI and Node Host are specified in the "Configure ROS Network Addresses" dialog. You can access this in the menu under *Tools>>Robot Operating System (ROS)* by selecting "Configure ROS Network Addresses".



The **Network Address** parameter can be set to "Default" or "Custom".

For the ROS master URI, if **Network Address** is set to "Default", Simulink uses the following rules to set the ROS Master URI:

- Use `ROS_MASTER_URI` environment variable if it is set.

- If a MATLAB global ROS node exists, use the Master URI associated with the global node. The global node is created automatically when `rosinit` is called.
- Use address `http://localhost:11311` if other two rules do not apply.

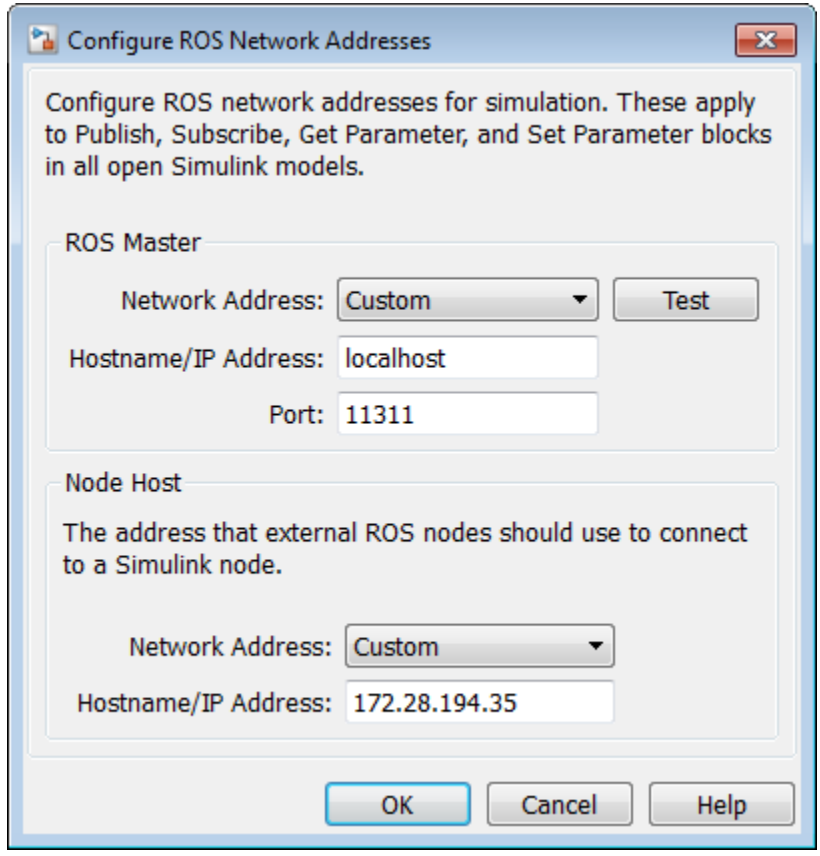
For the Node Host, if **Network Address** is set to "Default", Simulink uses the following rules to set the ROS Node Host:

- Use `ROS_HOSTNAME` environment variable if it is set.
- Use `ROS_IP` environment variable if it is set.
- Use hostname or IP address of the first network interface on the system if available.
- Use address `http://localhost:11311` if other rules do not apply.

For both, these are the same rules that MATLAB uses to resolve its ROS network addresses.

Otherwise, if you chose "Custom", you can set all the variables as shown below. This overrides the environment variables.

Note: These addresses are saved in MATLAB preferences, not the model. Therefore, this information is shared across all Simulink models and multiple MATLAB installs of the same release.



You can also use the "Test" button to ensure you can connect to the ROS master. If you get an error, call `rosinit` to setup a local ROS network, or if you specified a remote ROS master, check your settings are correct.

The custom ROS master or node host settings are not used in generated code when deploying a standalone node.

See Also

`rosinit`

Related Examples

- "Get Started with ROS in Simulink®"

- “Connect to a ROS-enabled Robot from Simulink®”

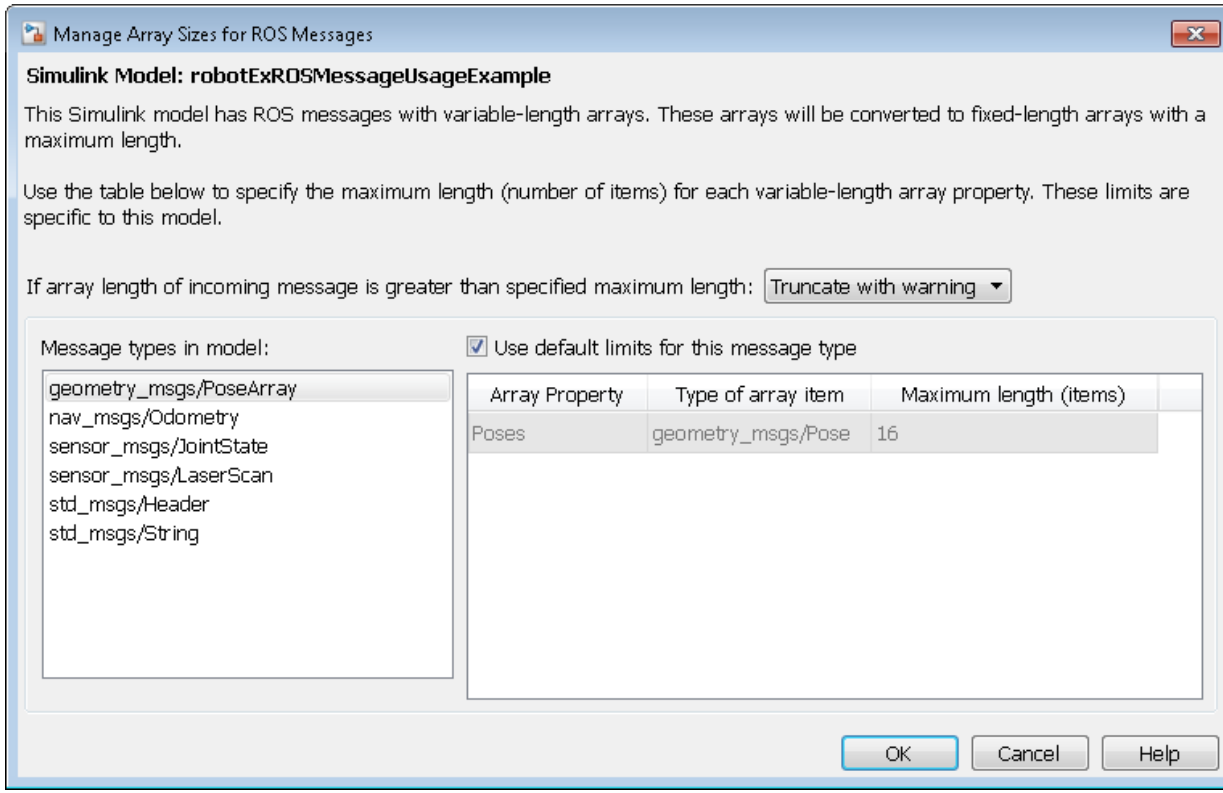
More About

- “Simulink and ROS Interaction” on page 6-12
- “Selecting ROS Topics, Messages, and Parameters” on page 6-2
- “Simulink Support and Limitations” on page 6-19

Managing Array Sizes in Simulink ROS

A ROS message is represented as a bus signal. For more information on bus signals, see "Buses".

If you are working with variable-length signals in Simulink, the non-virtual bus used for messages cannot contain variable-length arrays as properties. All variable-length arrays are converted to fixed-length arrays for non-virtual buses. Therefore, you must manage the maximum size for these fixed-size arrays. To manage array sizes, select *Tools>>Robot Operating System* in the menu and select "Manage Array Sizes". If your model uses ROS messages with variable-length arrays, the following dialog box opens. Otherwise, Simulink displays a message.



Because the message properties have a variable length, it is possible that they can be truncated if they exceed the maximum size set for that array. You have the option of

`Truncate with warning` or `Truncate silently`. Either way, the simulation will run, but `Truncate with warning` displays a warning in the Diagnostic Viewer that the message property has been truncated. When using generated code, the warning will be emitted using Log Statements in ROS. The warning will be a `ROS_WARN_NAMED` log statement and the *name* is the model name.

The **Message types in model** section shows all the ROS message types that are currently used by Publish, Subscribe and Blank Message blocks in your Simulink model. You have the option to use the default limits for this message type by clicking the check box. Otherwise, select each message type individually to set the **Maximum length (items)** of each **Array Property** as desired. This maximum length is applied to all instances of that message type for that model. The maximum length is also stored with the model. Therefore, it is possible to have two models accessing the same message type with different maximum length limits.

Managing the size of your variable-length arrays can help improve performance. If you limit the size of the array to only include relevant data, you can process data more effectively. However, when running these models, consider possible issues associated with truncation and what could happen to your system if some data is ignored.

Note: If you would like to know the appropriate maximum lengths for different message types. You can simulate the model and observe the sizes output in the warning. To see an example of using ROS messages and working with variable-length arrays, see “Work with ROS messages in Simulink®”.

See Also

`Publish` | `Subscribe`

More About

- “Simulink Support and Limitations” on page 6-19

Simulink and ROS Interaction

When using Simulink to communicate with a ROS network or work with ROS functionality, there are several points to note regarding its interaction with MATLAB and the ROS network.

In this section...

MATLAB ROS Information

Simulink uses the functionality built into MATLAB to communicate with the ROS network during simulation. When trying to debug issues in Simulink, you can use MATLAB to view topics or messages available on the ROS master. For more information on ROS topics and messages, see `rosnode`, `rostopic`, or `rosmsg`.

By default, Simulink uses MATLAB ROS capabilities to resolve network information such as the address of the ROS master. This network information can also be specified in Simulink using the "Configure ROS Network Addresses" on page 6-6 dialog.

Simulink ROS Node

Each model is associated with a unique ROS node. At the start of each simulation, Simulink creates the node and deletes it when the simulation is terminated. If multiple models are open and being simulated, each model will get its own dedicated node, but all the nodes will connect to the same ROS master. This is because all the models use the same ROS network address settings.

In simulation, the Simulink ROS node name is `<modelName>_<random#>`. This takes the model name and adds a random number to the end to avoid node name conflicts.

In generated code, the node name is `<modelName>` (casing preserved). The model name is also used in the archive used for generated code. Do **not** rename the tgz file from code generation (e.g. ModelName.tgz). The file name is used to get the ROS package name and initiate the build.

Differences Between Simulation and Generated Code

In simulation, the model execution does not match real elapsed time. The blocks in the model are evaluated in a loop that only simulates the progression of time, and whose speed depends on complexity of the model and computer speed. It is not intended to track actual clock time.

\In generated code, the model execution attempts to match actual elapsed time (the Fixed-step size defines the actual time step, in seconds, that is used for the model update loop). However, this does not guarantee real-time performance, as it is dependent on other processes running on the Linux system and the complexity of the model. For more information, see the *Tasking Mode* section in the "Generate a standalone ROS node from Simulink®" example.

Publishers and Subscribers in Simulink

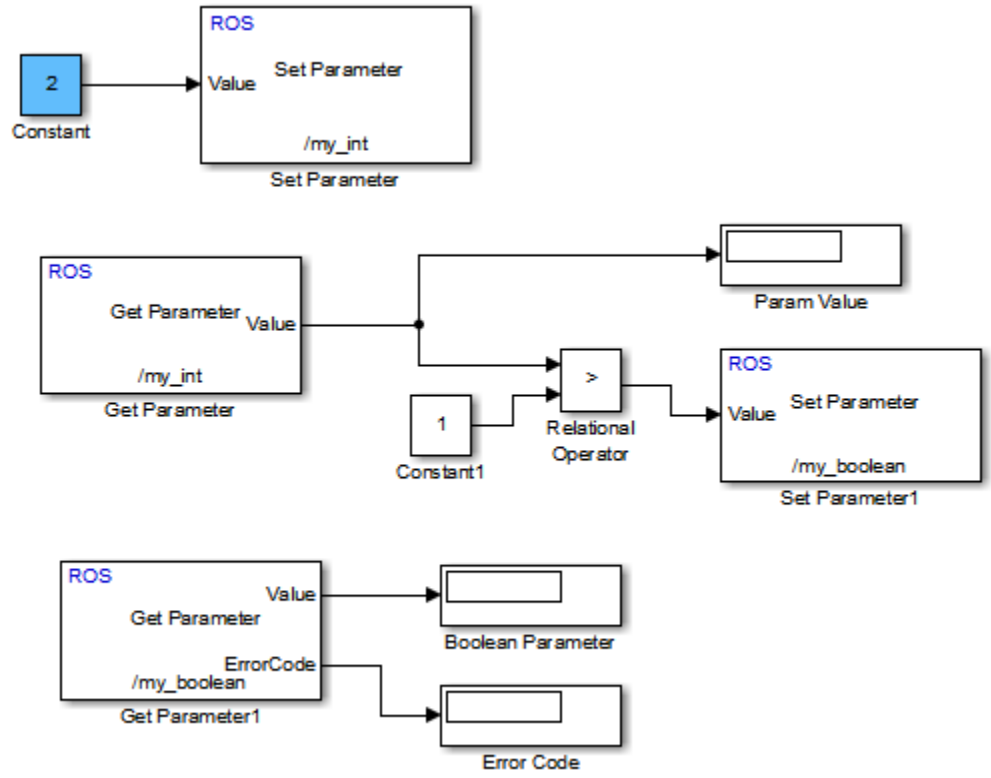
All publishers and subscribers created using `Publish` and `Subscribe` blocks will connect with the ROS node for that model. They are created during the model initialization and topic names are resolved at the same time. The publishers and subscribers are deleted when the simulation is terminated.

NOTE: If a custom topic name is specified for a `Subscribe` block, the topic is not required to exist when the model is initialized. The `Subscribe` block will output blank messages until it receives a message on the topic name you specify. This allows you to setup and test models before the rest of the network has been setup.

ROS Parameters in Simulink

Get and Set ROS Parameters

This model gets and sets ROS parameters using Simulink®. This example illustrates how to use ROS parameters in Simulink and to share data over the ROS network. An integer value is set as a parameter on the ROS network. This integer is retrieved from the parameter server and compared to a constant. The output Boolean from the comparison is also set on the network. Change the constant block in the top left (blue) when you run the model to set network parameters based on user input conditions.



See Also

`Get Parameter` | `Set Parameter`

More About

- "ROS String Parameters" on page 6-16

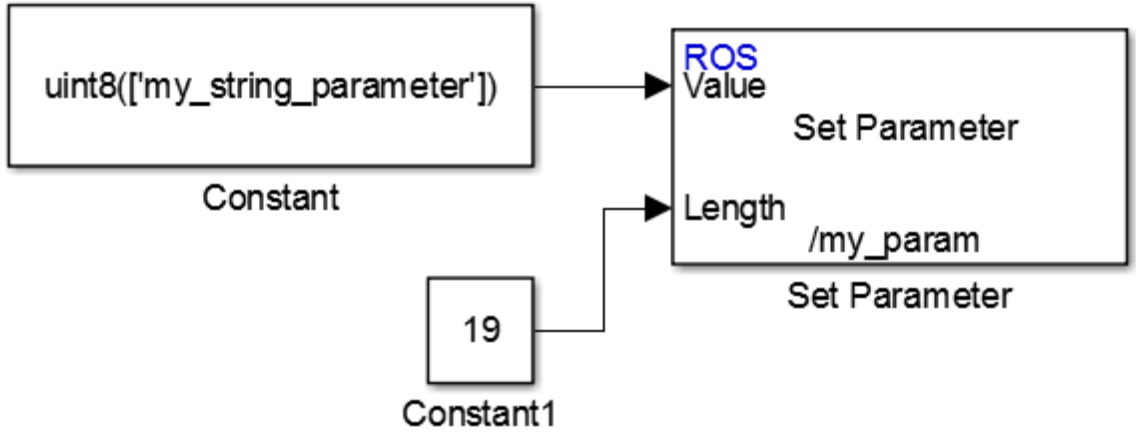
ROS String Parameters

In this section...

To use ROS string parameters in Simulink, cast them to `uint8` arrays. These examples show how to get, set, compare, and manipulate strings for ROS parameters. To run these examples, you must first set up a ROS network. Use `rosinit`.

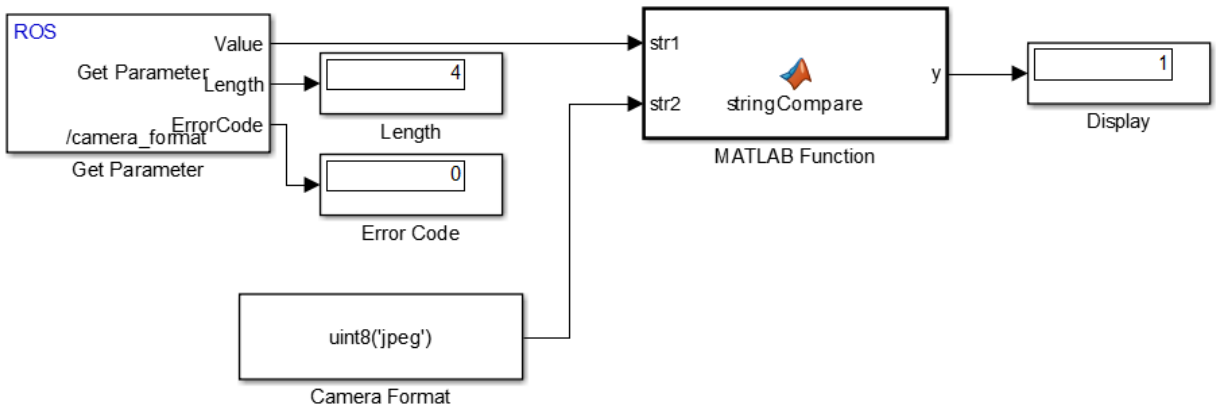
Set String Parameter on ROS Network

To create your string parameter, use a `Constant` block and cast it to `uint8` by specifying `uint8(['my_string_parameter'])` in **Constant Value** of the block mask. The string is passed into the `Set Parameter` block along with the extra input, Length, specified with a second `Constant` block. The Length refers to the maximum expected string length and is required for all string parameters. For more information, see the `Set Parameter` block.



Get ROS String Parameter and Compare to Specified String

You can compare string parameters to specified strings to validate settings or trigger subsystems. To get the parameter off the server, use the `Get Parameter` block. Then, use the MATLAB Function block to compare the parameter to a `uint8` string from a `Constant` block. This model checks to see if a previously set camera format parameter is named `'jpeg'`.

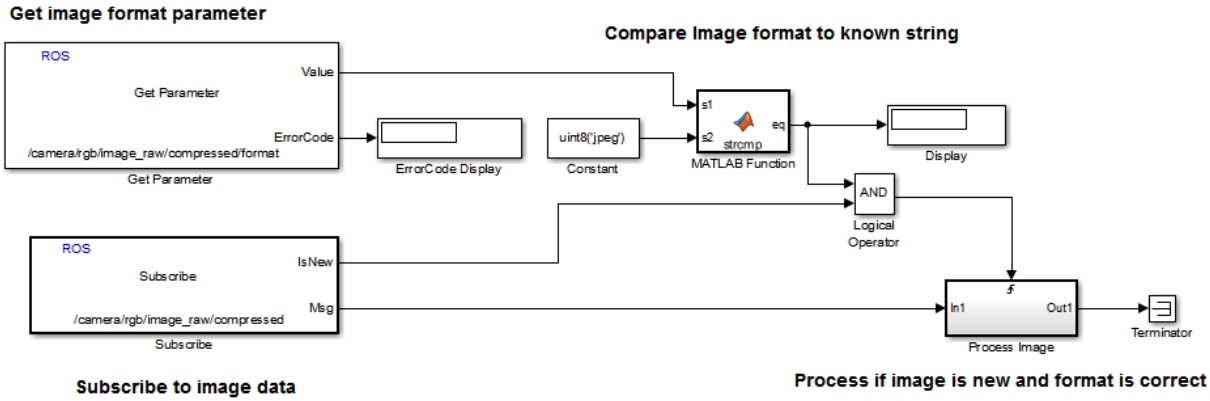


The following code is used inside the `stringCompare MATLAB Function` block. The function compares each character of its two input strings to see if they match. The output is a single Boolean indicating whether the strings match.

```
function y = stringCompare(str1,str2)
%#codegen
minLength = min(length(str1),length(str2));
st1 = str1(1:minLength);
st2 = str2(1:minLength);
y = all(st1(:)==st2(:));
```

ROS String Parameters in Simulink

This model shows how to access string parameters and use them to trigger subsystem operations. It gets an image format off the set up ROS parameter server. It is retrieved as a `uint8` array that is compared using the `strcmp` MATLAB function block. When a new image is received from the Subscribe block and the format is `unint8('jpeg')`, it triggers the "Process Image" block to perform a task on the image data.



See Also

Blocks

`Get Parameter` | `Set Parameter`

More About

- “ROS Parameters in Simulink” on page 6-14

Simulink Support and Limitations

Robotics System Toolbox does not support the following ROS features in Simulink:

- ROS Services
- ROS Parameter Server
- Transformation trees

If your application requires these features, consider using MATLAB ROS functionality. You can write a ROS node using MATLAB that can publish services, parameters, and transformation trees to a topic as ROS messages. Simulink can then subscribe to that topic to work with those messages. The following functions are used in MATLAB to work with these features:

- ROS Services: `rosservice`, `rossvcserver`, `rossvcclient`, `call`
- ROS Parameter Servers: `rosparam`
- Transformation trees: `rostf`, `transform`, `getTransform`

Related Examples

- “ROS String Parameters” on page 6-16

More About

- “Simulink and ROS Interaction” on page 6-12
- “Managing Array Sizes in Simulink ROS” on page 6-10

Algorithm Design

Occupancy Grids

In this section...

Binary Occupancy Grid

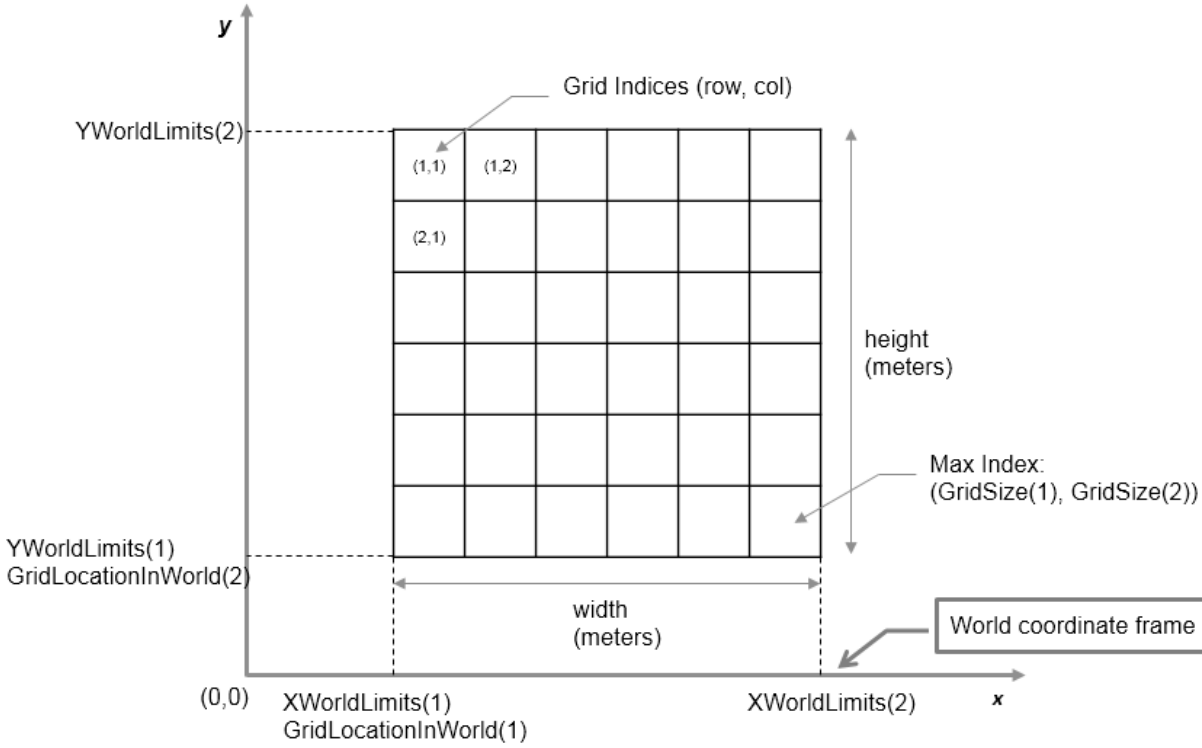
Occupancy grids are used to represent a robot’s workspace as a discrete grid. Information about the environment can be collected from sensors in real time or be loaded from prior knowledge. For robotics applications, laser range finders, bump sensors, cameras, and depth sensors are commonly used to find obstacles. A binary occupancy grid uses simple `true` and `false` values to represent the occupied workspace (obstacles) and free workspace respectively. This map shows where obstacles are and whether a robot can move through that space.

World and Grid Coordinates

When working with occupancy grids, you have the option to use either world or grid coordinates. This section describes the difference between these coordinate systems and how to work with them in MATLAB.

The absolute reference frame in which the robot operates is referred to as the "world" frame in the occupancy grid. Most operations using Robotics System Toolbox are performed in the world frame, and this is the default selection when using MATLAB functions in this toolbox. World coordinates are used as an absolute coordinate frame with a fixed origin and have unlimited resolution when specifying locations. However, all locations are converted to grid locations because of data storage and resolution limits.

Grid coordinates define the actual resolution of the occupancy grid and the finite locations of obstacles. The origin of grid coordinates is in the top-left of the grid with the first location having a location of `(1,1)`. However, the location of the grid in world coordinates is defined by the property, `GridLocationInWorld` and specifies the bottom-left location of the grid. When creating a robotics.BinaryOccupancyGrid object, other properties such as `XWorldLimits` and `YWorldLimits` are defined by the input `width`, `height`, and `resolution`. The figure below shows a visual representation of these properties and the relation between world and grid coordinates.



Grid Coordinates from World Coordinate Inputs

When setting occupancy locations, you can input the locations in either grid or world coordinates. However, based on the limits of the grid, the locations are set to the closest grid locations. Edges of the grid belong to the lower left grid location.

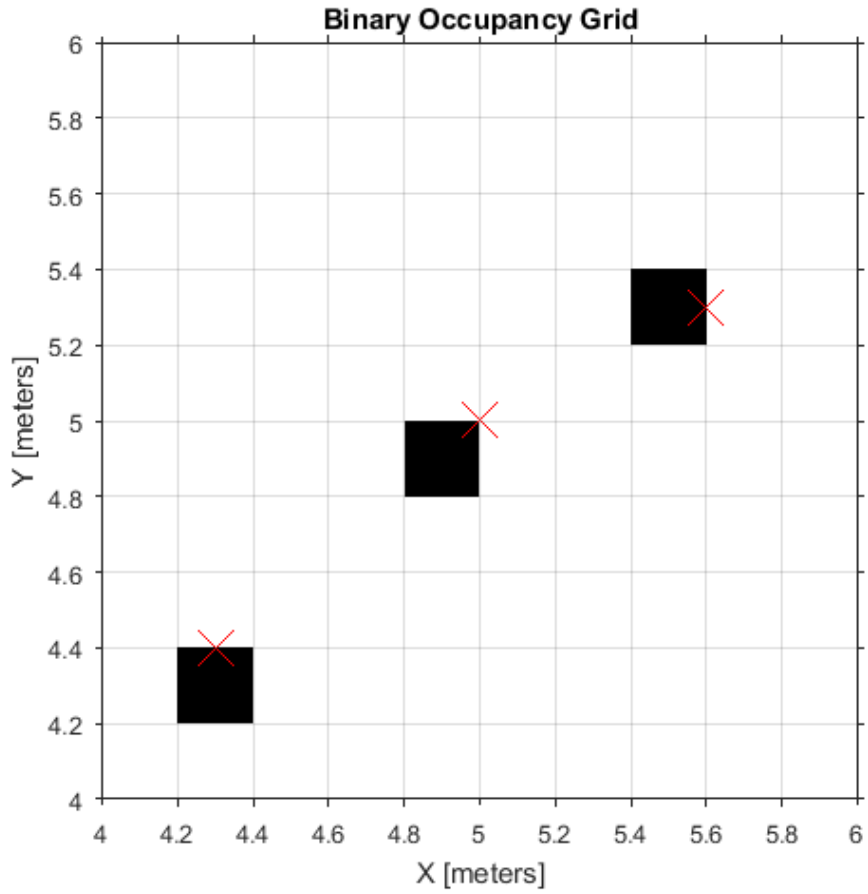
This example shows how the locations are interpreted on the grid. An occupancy grid is created and three points are set as occupied by obstacles. The original input points are then plotted over the map to show how they are interpreted. Also note that the entire grid cell is set as occupied if any point within the grid cell is set as occupied.

Create occupancy grid and set obstacle locations.

```
map = robotics.BinaryOccupancyGrid(10,10,5);
xy = [5 5; 4.3 4.4; 5.5 5.4];
setOccupancy(map,xy, 1);
```

Display map, original points and set axes.

```
show(map);
hold on
plot(xy(:,1),xy(:,2),'xr','MarkerSize', 20)
grid on
set(gca,'XTick', 0:0.2:10,'YTick',0:0.2:10)
xlim([4 6])
ylim([4 6])
```

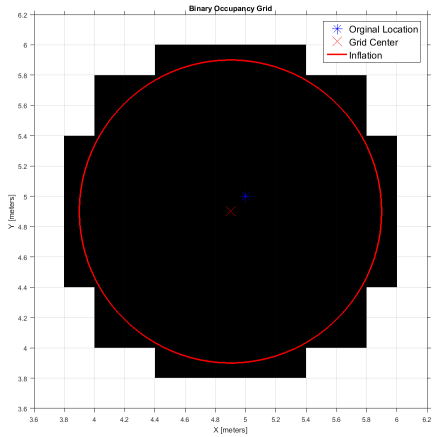


Inflation of Coordinates

The `inflate` method (robotics.BinaryOccupancyGrid.inflate) is used to account for the robot's dimensions. By inflating the obstacle locations, you can create buffer zones for the robot to have space to navigate without hitting obstacles. When inflating obstacles, the specified `radius` is converted to the number of cells rounded up from the value `resolution*radius`. Each occupied cell is then inflated by this cell value. This inflation can have some important effects if the accuracy of the map at individual grid locations is extremely important. A basic inflation example is shown to illustrate how the radius value is used. The result of `inflate` by *1m* on an obstacle at world location`[5 5]` is shown. The input `radius` is plotted as a circle from the obstacle grid location. ***Note:*** *The center location of the obstacle is converted to a grid location (from `[5 5]` in blue to `[4.9 4.9]` in red), which accounts for a small shift in the circle center as well.*

This example shows how to create the map, set the obstacle locations and inflate it by a radius of 1m. Extra plots on the figure help illustrate the inflation and shifting due to conversion to grid locations.

```
map = robotics.BinaryOccupancyGrid(10,10,5);
setOccupancy(map,[5 5], 1);
inflate(map,1);
show(map)
hold on
theta = linspace(0,2*pi);
x = 4.9+cos(theta);
y = 4.9+sin(theta);
plot(5,5,'*b','MarkerSize',10)
plot(4.9,4.9,'xr','MarkerSize',10)
plot(x,y,'-r','LineWidth',2);
xlim([3.6 6.2])
ylim([3.6 6.2])
grid on
legend('Original Location','Grid Center','Inflation')
```



As you can see from the above figure, even cells that barely overlap with the inflation radius are labeled as occupied.

See Also

robotics.BinaryOccupancyGrid | `readBinaryOccupancyGrid` | `writeBinaryOccupancyGrid`

Related Examples

- “Update an Occupancy Grid From Range Sensor Data”

Particle Filter Parameters

In this section...

A *particle filter* is a recursive, Bayesian state estimator that uses discrete particles to approximate the posterior distribution of the estimated state.

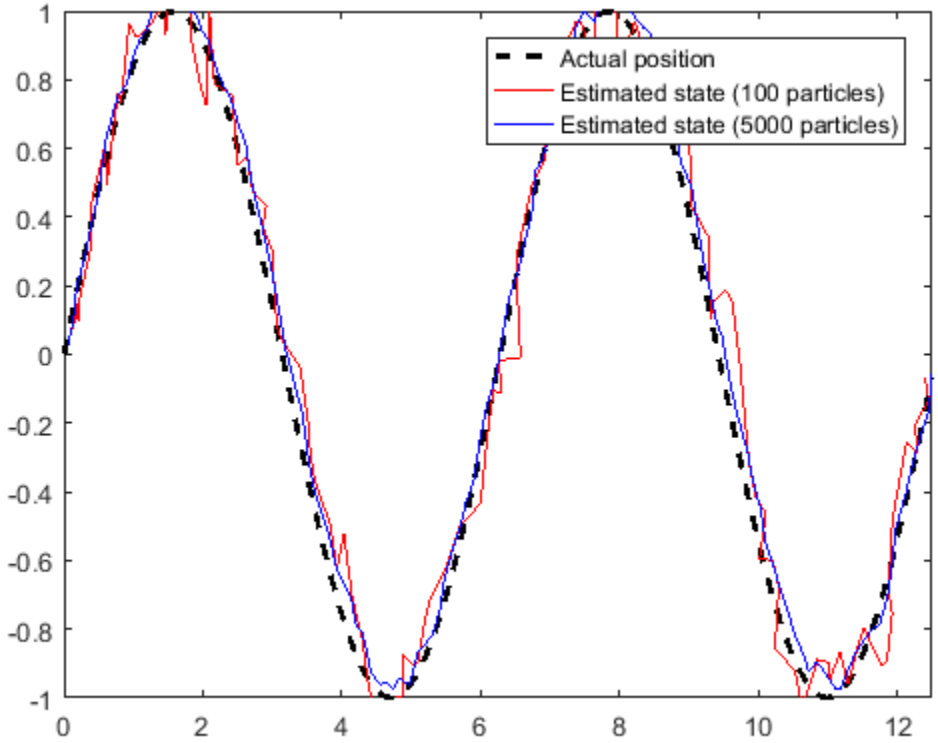
To use the particle filter properly, you must specify parameters such as the number of particles, the initial particle location, and the state estimation method. Also, if you have a specific motion and sensor model, you specify these parameters in the state transition function and measurement likelihood function, respectively. The details of these parameters are detailed on this page. For more information on the particle filter workflow, see "Particle Filter Workflow" on page 7-14.

Number of Particles

To specify the number of particles, use the `initialize` method. Each particle is a hypothesis of the current state. The particles are distributed across your state space based on either a specified mean and covariance, or on the specified state bounds. Depending on the `StateEstimationMethod` property, either the particle with the highest weight or the mean of all particles is taken to determine the best state estimate.

The default number of particles is 1000. Unless performance is an issue, do not use fewer than 1000 particles. A higher number of particles can improve the estimate but sacrifices performance speed, because the algorithm has to process more particles. Tuning the number of particles is the best way to improve the tracking of your particle filter.

These results, which are based on"Estimate Robot Position in a Loop Using Particle Filter" on page 7-19 example, show the difference in tracking accuracy when using 100 particles and 5000 particles.



Initial Particle Location

When you initialize your particle filter, you can specify the initial location of the particles using:

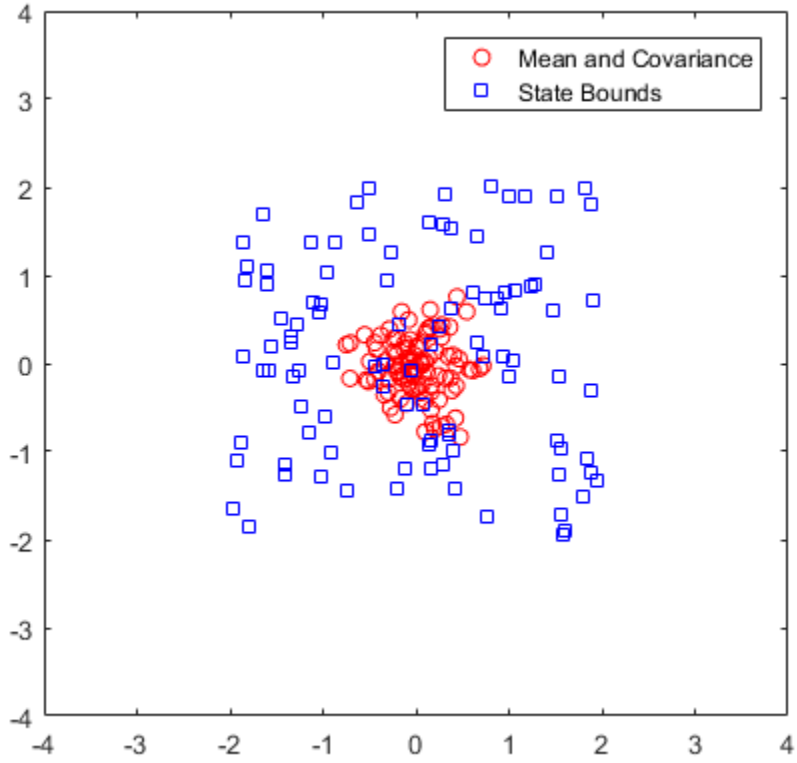
- Mean and covariance
- State bounds

If you have an initial estimated state, specify it as your initial mean with a covariance relative to your system. This covariance correlates to the uncertainty of your system. The `ParticleFilter` object distributes particles based on your covariance around the given mean. The algorithm uses this distribution of particles to get the best estimation of state,

so an accurate initialization of particles helps to converge to the best state estimation quickly.

If an initial state is unknown, you can evenly distribute your particles across a given state bounds. The state bounds are the limits of your state. For example, when estimating the position of a robot, the state bounds are limited to the environment that the robot can actually inhabit. In general, an even distribution of particles is a less efficient way to initialize particles to improve the speed of convergence.

The plot shows how the mean and covariance specification can cluster particles much more effectively in a space rather than specifying the full state bounds.



State Transition Function

The state transition function, `StateTransitionFcn`, of a particle filter helps to evolve the particles to the next state. It is used during the prediction step of the "Particle Filter Workflow" on page 7-14. In the `ParticleFilter` object, it is specified as a callback function that takes the previous particles, and any other necessary parameters, and outputs the predicted location. The function header syntax is:

```
function predictParticles = stateTransitionFcn(pf,prevParticles,varargin)
```

By default, the state transition function assumes a Gaussian motion model with constant velocities. The function uses a Gaussian distribution to determine the position of the particles in the next time step.

For your application, it is important to have a state transition function that accurately describes how you expect the system to behave. To accurately evolve all the particles, you must develop and implement a motion model for your system. If particles are not distributed around the next state, the `ParticleFilter` object does not find an accurate estimate. Therefore, it is important to understand how your system can behave so that you can track it accurately.

You also must specify system noise in `StateTransitionFcn`. Without random noise applied to the predicted system, the particle filter does not function as intended.

Although you can predict many systems based on their previous state, sometimes the system can include extra information. The use of `varargin` in the function enables you to input any extra parameters that are relevant for predicting the next state. When you call `predict`, you can include these parameters using:

```
predict(pf,param1,param2)
```

Because these parameters match the state transition function you defined, calling `predict` essentially calls the function as:

```
predictParticles = stateTransitionFcn(pf,prevParticles,param1,param2)
```

The output particles, `predictParticles`, are then either used by the "Measurement Likelihood Function" on page 7-11 to correct the particles, or used in the next prediction step if correction is not required.

Measurement Likelihood Function

After predicting the next state, you can use measurements from sensors to correct your predicted state. By specifying a `MeasurementLikelihoodFcn` in the `ParticleFilter` object, you can correct your predicted particles using the `correct` function. This measurement likelihood function, by definition, gives a weight for the state hypotheses (your particles) based on a given measurement. Essentially, it gives you the likelihood that the observed measurement actually matches what each particle observes. This likelihood is used as a weight on the predicted particles to help with correcting them and getting the best estimation. Although the prediction step can prove accurate for a small number of intermediate steps, to get accurate tracking, use sensor observations to correct the particles frequently.

The specification of the `MeasurementLikelihoodFcn` is similar to the `StateTransitionFcn`. It is specified as a function handle in the properties of the `ParticleFilter` object. The function header syntax is:

```
function likelihood = measurementLikelihoodFcn(pf,predictParticles,measurement,varargi
```

The output is the likelihood of each predicted particle based on the measurement given. However, you can also specify more parameters in `varargin`. The use of `varargin` in the function enables you to input any extra parameters that are relevant for correcting the predicted state. When you call `correct`, you can include these parameters using:

```
correct(pf,measurement,param1,param2)
```

These parameters match the measurement likelihood function you defined:

```
likelihood = measurementLikelihoodFcn(pf,predictParticles,measurement,param1,param2)
```

The `correct` function uses the `likelihood` output for particle resampling and giving the final state estimate.

Resampling Policy

The resampling of particles is a vital step for continuous tracking of objects. It enables you to select particles based on the current state, instead of using the particle distribution given at initialization. By continuously resampling the particles around the current estimate, you can get more accurate tracking and improve long-term performance.

When you call `correct`, the particles used for state estimation can be resampled depending on the `ResamplingPolicy` property specified in the `ParticleFilter` object. This property is specified as a robotics.ResamplingPolicy object. The `TriggerMethod` property on that object tells the particle filter which method to use for resampling.

You can trigger resampling at either a fixed interval or when a minimum effective particle ratio is reached. The fixed interval method resamples at a set number of iterations, which is specified in the `SamplingInterval` property. The minimum effective particle ratio is a measure of how well the current set of particles approximates the posterior distribution. The number of effective particles is calculated by:

$$N_{eff} = \frac{1}{\sum_{i=1}^{N}\left(w^i\right)^2}$$

In this equation, N is the number of particles, and w is the normalized weight of each particle. The effective particle ratio is then N_{eff} / `NumParticles`. Therefore, the effective particle ratio is a function of the weights of all the particles. After the weights of the particles reach a low enough value, they are not contributing to the state estimation. This low value triggers resampling, so the particles are closer to the current state estimation and have higher weights.

State Estimation Method

The final step of the particle filter workflow is the selection of a single state estimate. The particles and their weights sampled across the distribution are used to give the best estimation of the actual state. However, you can use the particles information to get a single state estimate in multiple ways. With the `ParticleFilter` object, you can either choose the best estimate based on the particle with the highest weight or take a mean of all the particles. Specify the estimtation method in the `StateEstimationMethod` property as either `'mean'`(default) or `'maxweight'`.

Because you can estimate the state from all of the particles in many ways, you can also extract each particle and its weight from the robotics.ParticleFilter using the `Particles` property.

See Also

robotics.ParticleFilter | robotics.ParticleFilter.initialize | robotics.ParticleFilter.predict | robotics.ParticleFilter.correct

Related Examples

- "Track a Car-Like Robot using Particle Filter"
- "Estimate Robot Position in a Loop Using Particle Filter"

More About

- "Particle Filter Workflow" on page 7-14

Particle Filter Workflow

In this section...

A *particle filter* is a recursive, Bayesian state estimator that uses discrete particles to approximate the posterior distribution of the estimated state.

The particle filter algorithm computes the state estimate recursively and involves two steps:

- Prediction – The algorithm uses the previous state to predict the current state based on a given system model.
- Correction – The algorithm uses the current sensor measurement to correct the state estimate.

The algorithm also periodically redistributes, or resamples, the particles in the state space to match the posterior distribution of the estimated state.

The estimated state consists of all the state variables. Each particle represents a discrete state hypothesis. The set of all particles is used to help determine the final state estimate.

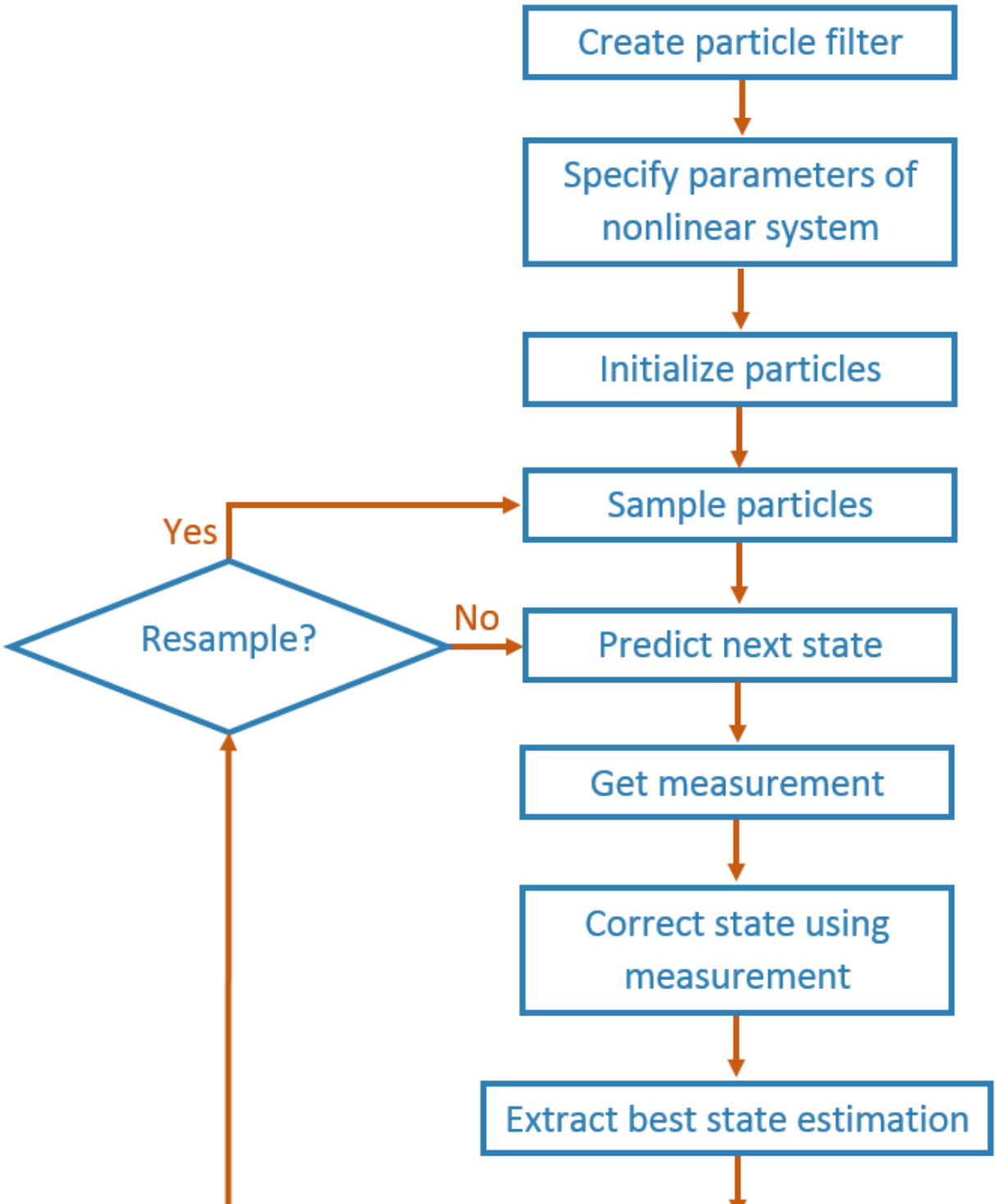
You can apply the particle filter to arbitrary nonlinear system models. Process and measurement noise can follow arbitrary non-Gaussian distributions.

To use the particle filter properly, you must specify parameters such as the number of particles, the initial particle location, and the state estimation method. Also, if you have a specific motion and sensor model, you specify these parameters in the state transition function and measurement likelihood function, respectively. For more information, see “Particle Filter Parameters” on page 7-7.

Follow this basic workflow to create and use a particle filter. This page details the estimation workflow and shows an example of how to run a particle filter in a loop to continuously estimate state.

Estimation Workflow

When using a particle filter, there is a required set of steps to create the particle filter and estimate state. The prediction and correction steps are the main iteration steps for continuously estimating state.


Create particle filter
Specify parameters of nonlinear system
Initialize particles
Sample particles
Resample?
Yes
No
Predict next state
Get measurement
Correct state using measurement
Extract best state estimation

Create Particle Filter

Create a `ParticleFilter` object by calling robotics.ParticleFilter.

Set Parameters of Nonlinear System

Modify these `ParticleFilter` parameters to fit for your specific system or application:

- `StateTransitionFcn`
- `MeasurementLikelihoodFcn`
- `ResamplingPolicy`
- `ResamplingMethod`
- `StateEstimationMethod`

Default values for these parameters are given for basic operation.

The `StateTransitionFcn` and `MeasurementLikelihoodFcn` functions define the system behavior and measurement integration. They are vital for the particle filter to track accurately. For more information, see "Particle Filter Parameters" on page 7-7.

Initialize Particles

Use the `initialize` method to set the number of particles and the initial state. See robotics.ParticleFilter.initialize.

Sample Particles from a Distribution

You can sample the initial particle locations in two ways:

- Initial pose and covariance — If you have an idea of your initial state, it is recommended you specify the initial pose and covariance. This specification helps to cluster particles closer to your estimate so tracking is more effective from the start.
- State bounds — If you do not know your initial state, you can specify the possible limits of each state variable. Particles are uniformly distributed across the state bounds for each variable. Widely distributed particles are not as effective at tracking, because fewer particles are near the actual state. Using state bounds usually requires more particles, computation time, and iterations to converge to the actual state estimate.

Predict

Based on a specified state transition function, particles evolve to estimate the next state. Use `predict` to execute the state transition function specified in the `StateTransitionFcn` property. See robotics.ParticleFilter.predict.

Get Measurement

The measurements collected from sensors are used in the next step to correct the current predicted state.

Correct

Measurements are then used to adjust the predicted state and correct the estimate. Specify your measurements using the `correct` function. robotics.ParticleFilter.correct uses the `MeasurementLikelihoodFcn` to calculate the likelihood of sensor measurements for each particle. Resampling of particles is required to update your estimation as the state changes in subsequent iterations. This step triggers resampling based on the `ResamplingMethod` and `ResamplingPolicy` properties.

Extract Best State Estimation

After calling `correct`, the best state estimate is automatically extracted based on the `Weights` of each particle and the `StateEstimationMethod` property specified in robotics.ParticleFilter. The best estimated state and covariance is output by the `correct` function.

Resample Particles

This step is not separately called, but is executed when you call `correct`. Once your state has changed enough, resample your particles based on the newest estimate. The `correct` method checks the `ResamplingPolicy` for the triggering of particle resampling according to the current distribution of particles and their weights. If resampling is not triggered, the same particles are used for the next estimation. If your state does not vary by much or if your time step is low, you can call the predict and correct methods without resampling.

Continuously Predict and Correct

Repeat the previous prediction and correction steps as needed for estimating state. The correction step determines if resampling of the particles is required. Multiple calls for `predict` or `correct` might be required when:

- No measurement is available but control inputs and time updates are occur at a high frequency. Use the `predict` method to evolve the particles to get the updated predicted state more often.
- Multiple measurement reading are available. Use `correct` to integrate multiple readings from the same or multiple sensors. The function corrects the state based on each set of information collected.

Estimate Robot Position in a Loop Using Particle Filter

Use the `ParticleFilter` object to track a robot as it moves in a 2-D space. The measured position has random noise added. Using `predict` and `correct`, track the robot based on the measurement and on an assumed motion model.

Initialize the particle filter and specify the default state transition function, the measurement likelihood function, and the resampling policy.

```
pf = robotics.ParticleFilter;
pf.StateEstimationMethod = 'mean';
pf.ResamplingMethod = 'systematic';
```

Sample 1000 particles with an initial position of [0 0] and unit covariance.

```
initialize(pf,1000,[0 0],eye(2));
```

Prior to esimation, define a sine wave path for the dot to follow. Create an array to store the predicted and estimated position. Define the amplitude of noise.

```
t = 0:0.1:4*pi;
dot = [t; sin(t)]';
robotPred = zeros(length(t),2);
robotCorrected = zeros(length(t),2);
noise = 0.1;
```

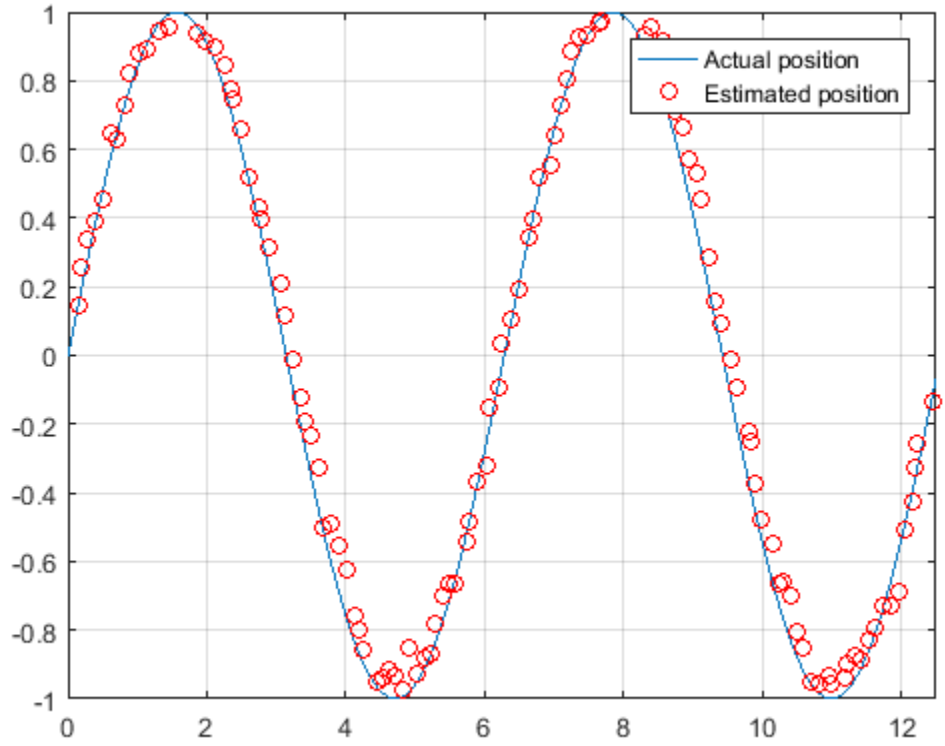
Begin the loop for predicting and correcting the estimated position based on measurements. The resampling of particles occurs based on the `ResamplingPolicy` property. The robot moves based on a sine wave function with random noise added to the measurement.

```
for i = 1:length(t)
    % Predict next position. Resample particles if necessary.
    [robotPred(i,:),robotCov] = predict(pf);
    % Generate dot measurement with random noise. This is
    % equivalent to the observation step.
```

```
    measurement(i,:) = dot(i,:) + noise*(rand([1 2])-noise/2);
    % Correct position based on the given measurement to get best estimation.
    % Actual dot position is not used. Store corrected position in data array.
    [robotCorrected(i,:),robotCov] = correct(pf,measurement(i,:));
end
```

Plot the actual path versus the estimated position. Actual results may vary due to the randomness of particle distributions.

```
plot(dot(:,1),dot(:,2),robotCorrected(:,1),robotCorrected(:,2),'or')
xlim([0 t(end)])
ylim([-1 1])
legend('Actual position','Estimated position')
grid on
```



The figure shows how close the estimate state matches the actual position of the robot. Try tuning the number of particles or specifying a different initial position and covariance to see how it affects tracking over time.

See Also

robotics.ParticleFilter | robotics.ParticleFilter.initialize | robotics.ParticleFilter.predict | robotics.ParticleFilter.correct

Related Examples

- "Track a Car-Like Robot using Particle Filter"
- "Estimate Robot Position in a Loop Using Particle Filter"

More About

- "Particle Filter Parameters" on page 7-7

Probabilistic Roadmaps (PRM)

In this section...

A probabilistic roadmap (PRM) is a network graph of possible paths in a given map based on free and occupied spaces. The robotics.PRM class randomly generates nodes and creates connections between these nodes based on the PRM algorithm parameters. Nodes are connected based on the obstacle locations specified in `Map`, and on the specified `ConnectionDistance`. You can customize the number of nodes, `NumNodes`, to fit the complexity of the map and the desire to find the most efficient path. The PRM algorithm uses the network of connected nodes to find an obstacle-free path from a start to an end location. To plan a path through an environment effectively, tune the `NumNodes` and `ConnectionDistance` properties.

When creating or updating the robotics.PRM class, the node locations are randomly generated, which can affect your final path between multiple iterations. This selection of nodes occurs when you specify `Map` initially, change the parameters, or `update` is called. To get consistent results with the same node placement, use `rng` to save the state of the random number generation. See "Tune the Connection Distance" on page 7-26 for an example using `rng`.

Tune the Number of Nodes

This example shows how to use the `NumNodes` property to tune the PRM algorithm. `NumNodes` specifies the number of points, or nodes, placed on the map, which the algorithm uses to generate a roadmap. Using the `ConnectionDistance` property as a threshold for distance, the algorithm connects all points that do not have obstacles blocking the direct path between them.

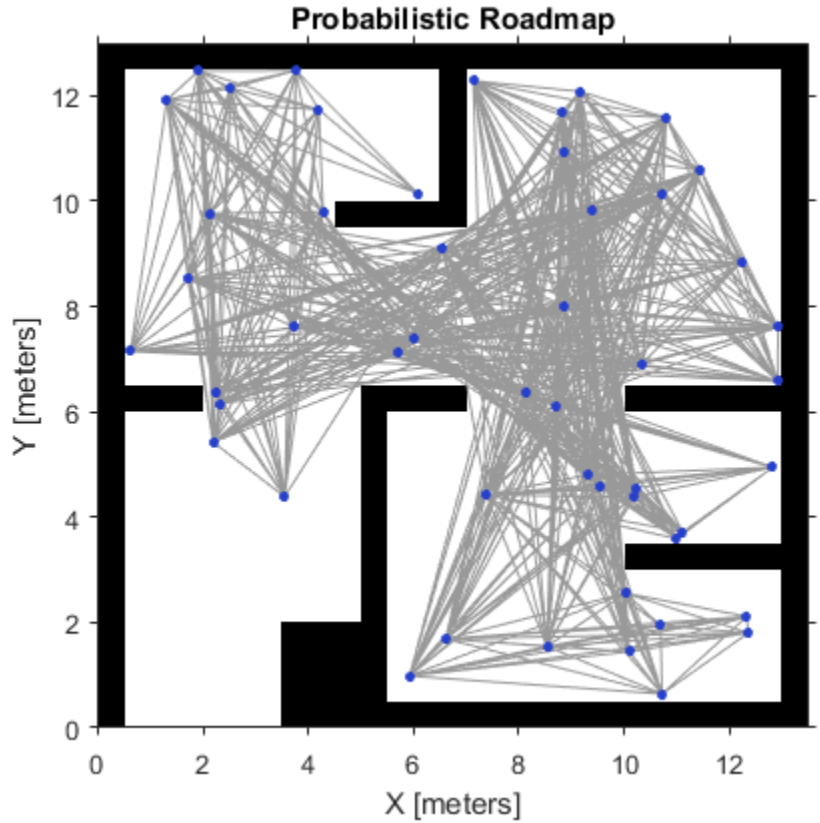
Increasing the number of nodes can increase the efficiency of the path by giving more feasible paths. However, the increased complexity increases computation time. To get good coverage of the map, you might need a large number of nodes. Due to the random placement of nodes, some areas of the map may not have enough nodes to connect to the rest of the map. In this example, you create a large and small number of nodes in a roadmap.

Load a map file and create an occupancy grid.

```
filePath = fullfile(fileparts(which('PathPlanningExample')),'data','exampleMaps.mat');
load(filePath)
map = robotics.BinaryOccupancyGrid(simpleMap,2);
```

Create a simple roadmap with 50 nodes.

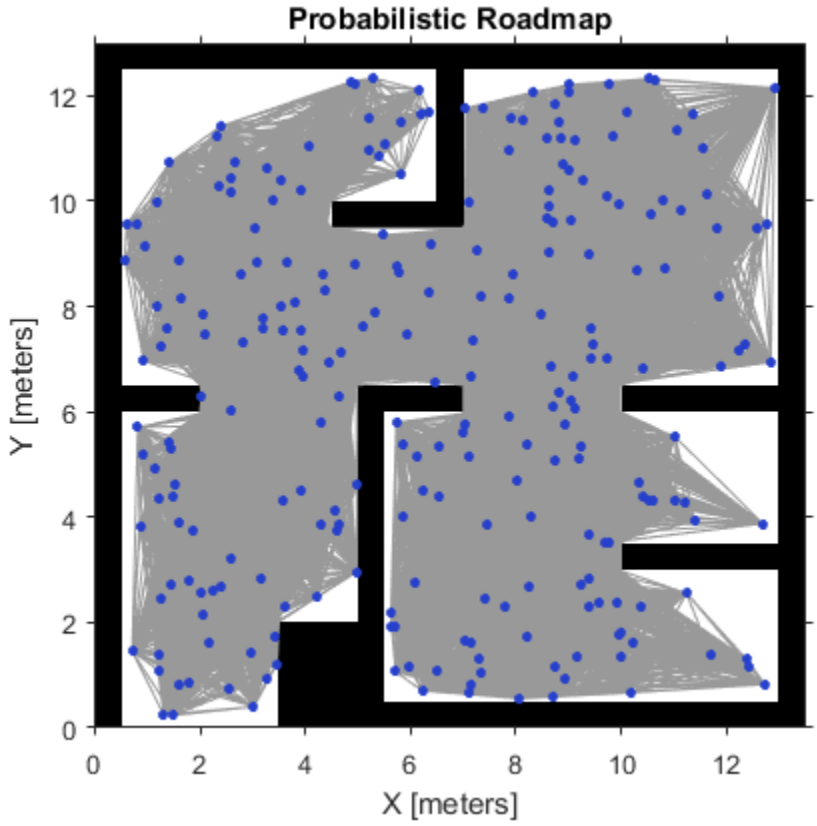
```
prmSimple = robotics.PRM(map,50);
show(prmSimple)
```



Create a dense roadmap with 250 nodes.

```
prmComplex = robotics.PRM(map,250);
```

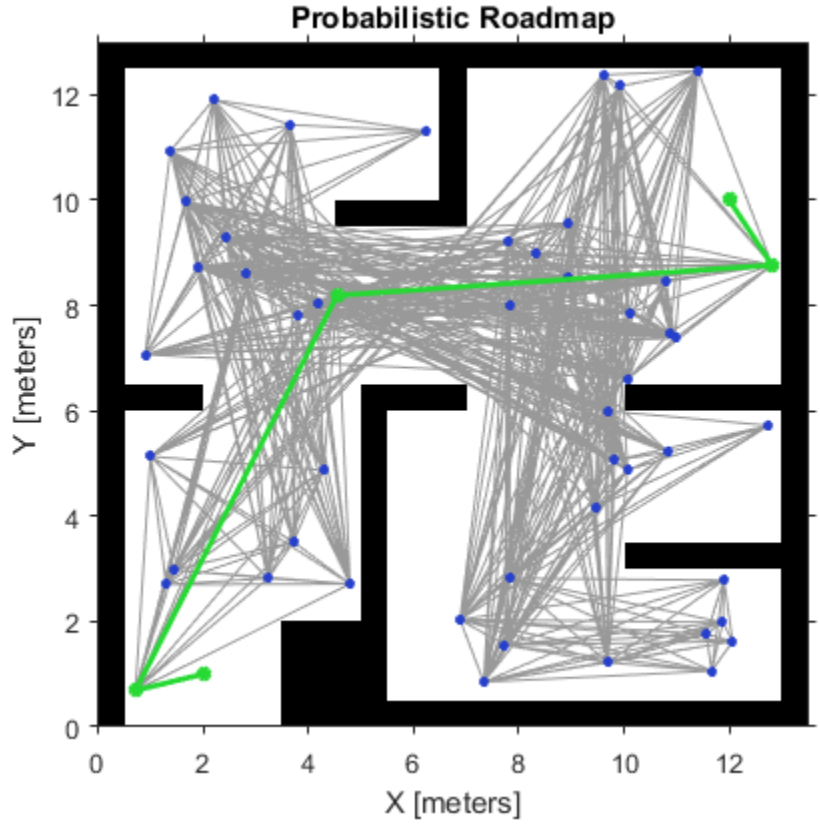
```
show(prmComplex)
```



The additional nodes increase the complexity but yield more options to improve the path. Given these two maps, you can calculate a path using the PRM algorithm and see the effects.

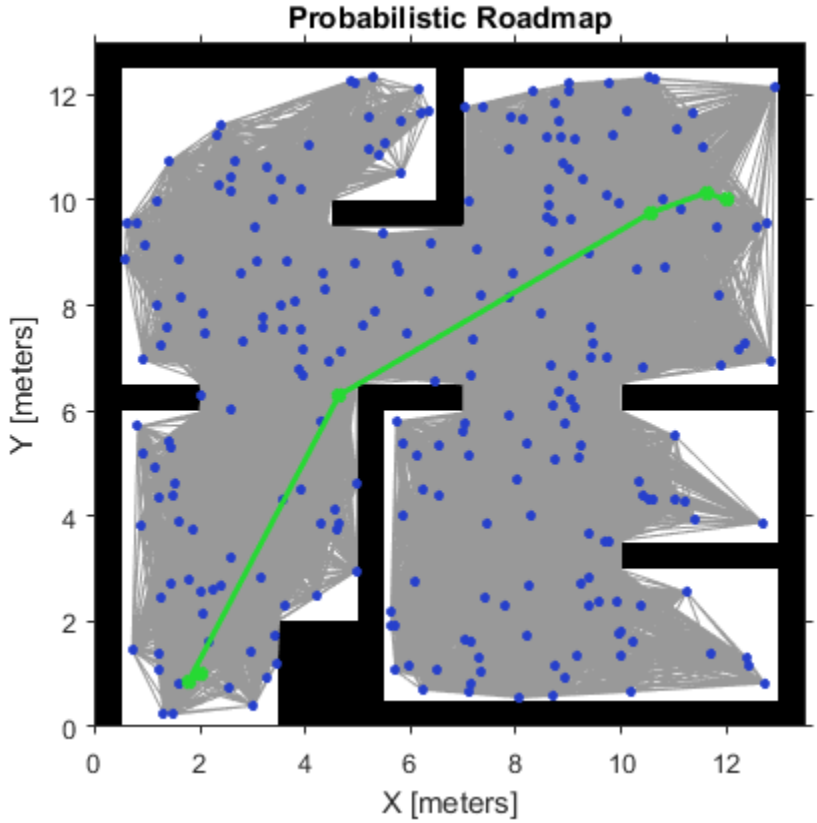
Calculate a simple path.

```
startLocation = [2 1];
endLocation = [12 10];
path = findpath(prmSimple,startLocation,endLocation);
show(prmSimple)
```



Calculate a complex path.

```
path = findpath(prmComplex, startLocation, endLocation);
show(prmComplex)
```



Increasing the nodes allows for a more direct path, but adds more computation time to finding a feasible path. Because of the random placement of points, the path is not always more direct or efficient. Using a small number of nodes can make paths worse than depicted and even restrict the ability to find a complete path.

Tune the Connection Distance

This example shows how to use the `ConnectionDistance` property to tune the PRM algorithm. `ConnectionDistance` is an upper threshold for points that are connected in the roadmap. Each node is connected to all nodes within this connection distance that do not have obstacles between them. By lowering the connection distance, you can limit the number of connections to reduce the computation time and simplify the map. However,

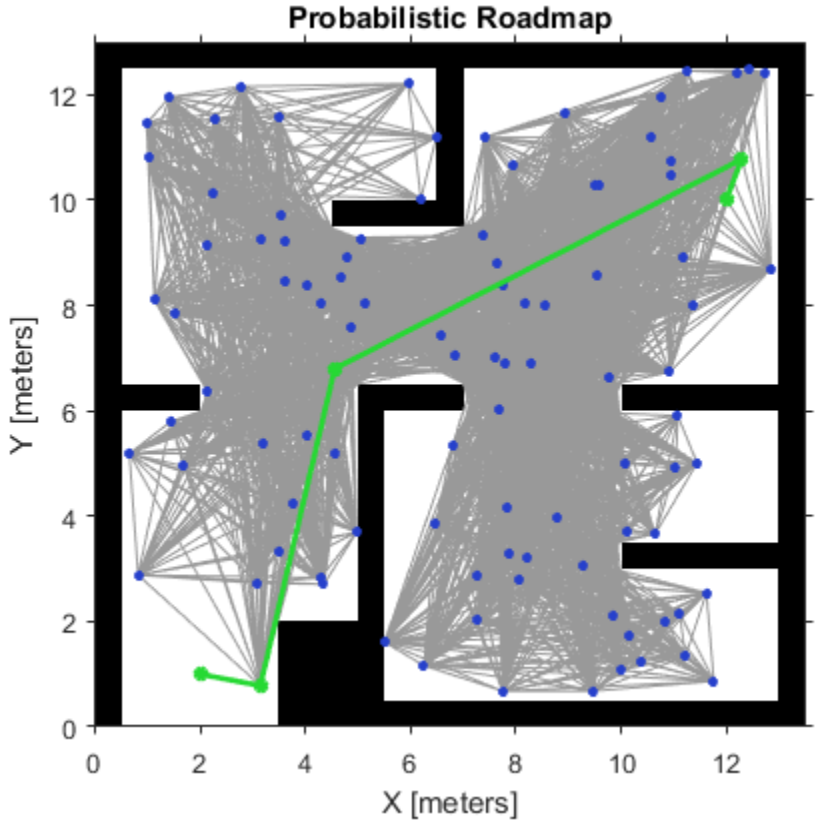
a lowered distance limits the number of available paths from which to find a complete obstacle-free path. When working with simple maps, you can use a higher connection distance with a small number of nodes to increase efficiency. For complex maps with lots of obstacles, a higher number of nodes with a lowered connection distance increases the chance of finding a solution.

Load a map file and create an occupancy grid.

```
filePath = fullfile(fileparts(which('PathPlanningExample')),'data','exampleMaps.mat');
load(filePath)
map = robotics.BinaryOccupancyGrid(simpleMap,2);
```

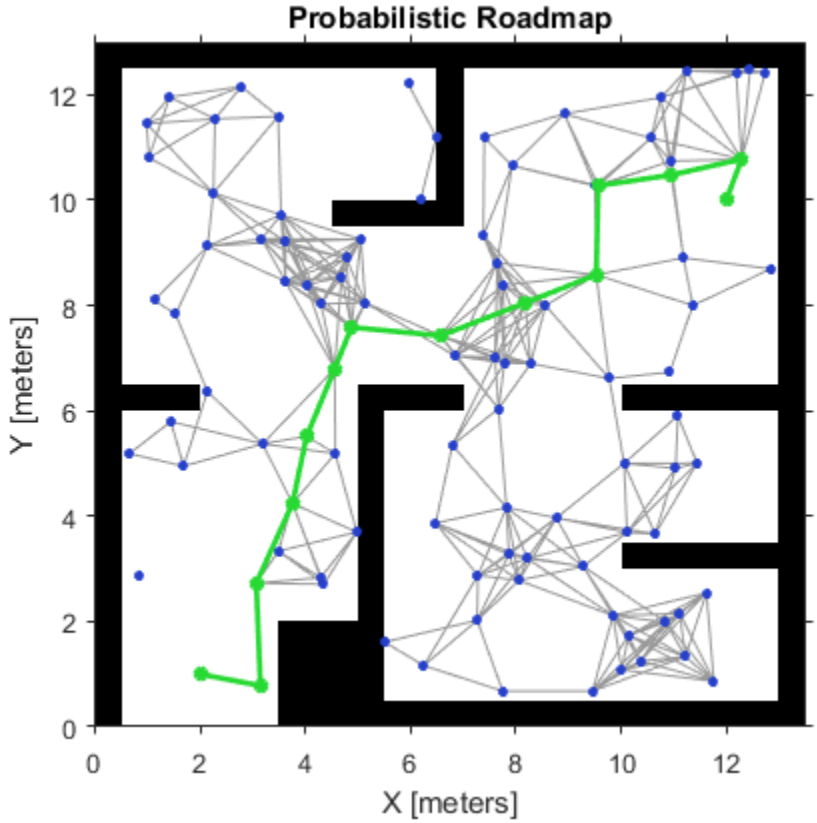
Create a roadmap with 100 nodes and calculate the path. The default `ConnectionDistance` is set to `inf`. Save the random number generation settings using the `rng` function. The saved settings enable you to reproduce the same points and see the effect of changing `ConnectionDistance`.

```
rngState = rng;
prm = robotics.PRM(map,100);
startLocation = [2 1];
endLocation = [12 10];
path = findpath(prm,startLocation,endLocation);
show(prm)
```



Reload the random number generation settings to have PRM use the same nodes. Lower `ConnectionDistance` to 2 m. Show the calculated path.

```
rng(rngState);
prm.ConnectionDistance = 2;
path = findpath(prm,startLocation,endLocation);
show(prm)
```



Create or Update PRM

This example shows how to create or update your roadmap. To create the roadmap, call `prm = robotics.PRM(map,__)` or specify the `prm.Map`property. Then, call the `update`, `findpath`, or `show` method. At this point, the nodes are randomly generated and the connections are made.

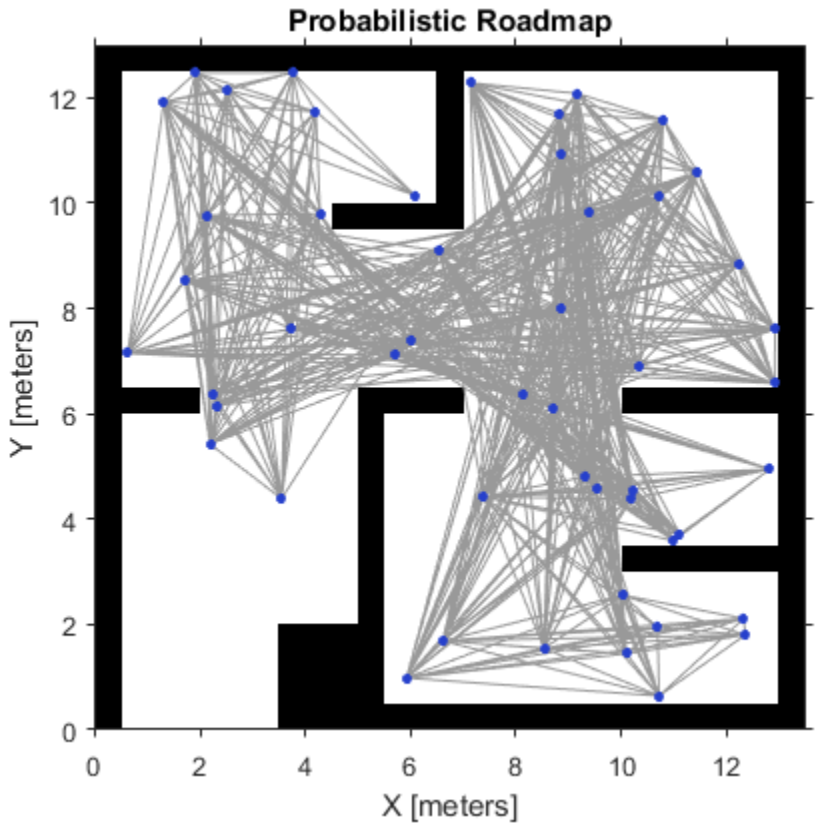
This roadmap changes only if you call `update` or change the properties in the `robotics.PRM` object. When properties change, any method (`update`, `findpath`, or `show`) called on the object triggers the roadmap points and connections to be recalculated. Because recalculating the map can be computationally intensive, you can reuse the same roadmap by calling `findpath` with different starting and ending locations.

Load a map file and create an occupancy grid.

```
filePath = fullfile(fileparts(which('PathPlanningExample')),'data','exampleMaps.mat');
load(filePath)
map = robotics.BinaryOccupancyGrid(simpleMap,2);
```

Create a roadmap. Your nodes and connections might look different due to the random placement of nodes.

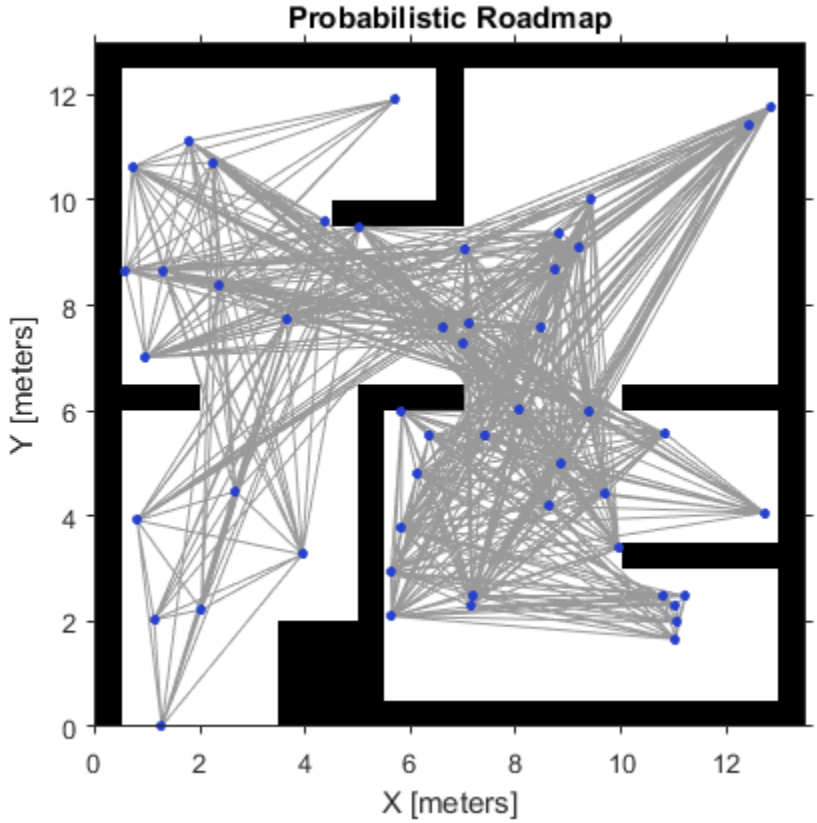
```
prm = robotics.PRM(map,100);
show(prm)
```



Call `update` or change a parameter to update the PRM nodes and connections.

```
update(prm)
```

```
show(prm)
```



The PRM algorithm recalculates the node placement and generates a new network of nodes.

References

[1] Kavraki, L.E., P. Svestka, J.-C. Latombe, and M.H. Overmars. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*. Vol. 12, No. 4, Aug 1996 pp. 566—580.

See Also

robotics.PRM | robotics.PRM.findpath | robotics.PRM.show | robotics.PRM.update

Pure Pursuit Controller

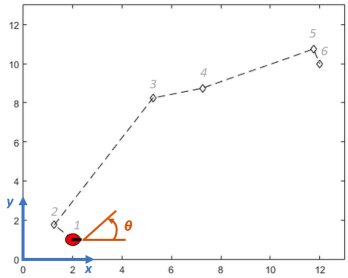
In this section...

PurePursuit is a path tracking algorithm. It computes the angular velocity command that moves the robot from its current position to reach some look-ahead point in front of the robot. The linear velocity is assumed constant, hence you can change the linear velocity of the robot at any point. The algorithm then moves the look-ahead point on the path based on the current position of the robot until the last point of the path. You can think of this as the robot constantly chasing a point in front of it. The property LookAheadDistance decides how far the look-ahead point is placed.

The `PurePursuit` class (robotics.PurePursuit) is not a traditional controller, but acts as a tracking algorithm for path following purposes. In the Robotics System Toolbox, you create a `PurePursuit` controller and specify a list of waypoints. The desired linear and maximum angular velocities can be specified. These properties are determined based on the robot's specifications. Given the pose (position and orientation) of the robot as an input, the `step` function can be used to calculate the linear and angular velocities commands for the robot. How the robot uses these commands is dependent on the system you are using, so consider how robots can execute a motion given these commands. The final important property is the `LookAheadDistance`, which tells the robot how far along on the path to track towards. This property is explained in more detail in a section below.

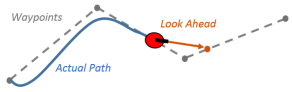
Reference Coordinate System

It is important to understand the reference coordinate frame used by the `PurePursuit` algorithm for its inputs and outputs. The figure below shows the reference coordinate system. The input waypoints are `[x y]` coordinates, which are used to compute the robot velocity commands. The robot's pose is input as a pose and orientation (theta) list of points as `[x y theta]`. The positive *x* and *y* directions are in the right and up directions respectively (blue in figure). The *theta* value is the angular orientation of the robot measured counterclockwise in radians from the *x*-axis (robot currently at `0` radians).

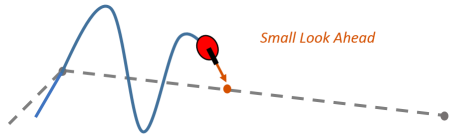


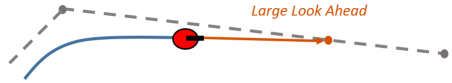
Look Ahead Distance

The `LookAheadDistance` property is the main tuning property for the `PurePursuit` controller. The look ahead distance is how far along the path the robot should look from the current location to compute the angular velocity commands. The figure below shows the robot and the look-ahead point. As displayed in this image, note that the actual path does not match the direct line between waypoints.



The effect of changing this parameter can change how your robot tracks the path and there are two major goals: regaining the path and maintaining the path. In order to quickly regain the path between waypoints, a small `LookAheadDistance` will cause your robot to move quickly towards the path. However, as can be seen in the figure below, the robot overshoots the path and oscillates along the desired path. In order to reduce the oscillations along the path, a larger look ahead distance can be chosen, however, it might result in larger curvatures near the corners.





The `LookAheadDistance` property should be tuned for your application and robot system. Different linear and angular velocities will affect this response as well and should be considered for the path following controller.

Limitations

There are a few limitations to note about this `PurePursuit` algorithm:

- As shown above, the controller cannot exactly follow direct paths between waypoints. Parameters must be tuned to optimize the performance and to converge to the path over time.
- This `PurePursuit` algorithm does not stabilize the robot at a point. In your application, a distance threshold for a goal location should be applied to stop the robot near the desired goal. This can be seen in the path following example: "Path Following for a Differential Drive Robot".

References

[1] Coulter, R. *Implementation of the Pure Pursuit Path Tracking Algorithm*. Carnegie Mellon University, Pittsburgh, Pennsylvania, Jan 1990.

See Also

robotics.PurePursuit | robotics.VectorFieldHistogram | robotics.PRM

Related Examples

- "Path Following for a Differential Drive Robot"

Vector Field Histogram

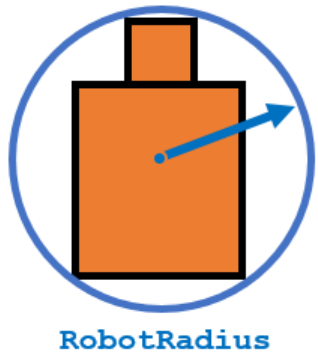
In this section...

The vector field histogram (VFH) algorithm computes obstacle-free steering directions for a robot based on range sensor readings. Range sensor readings are used to compute polar density histograms to identify obstacle location and proximity. Based on the specified parameters and thresholds, these histograms are converted to binary histograms to indicate valid steering directions for the robot. The VFH algorithm factors in robot size and turning radius to output a steering direction for the robot to avoid obstacles and follow a target direction.

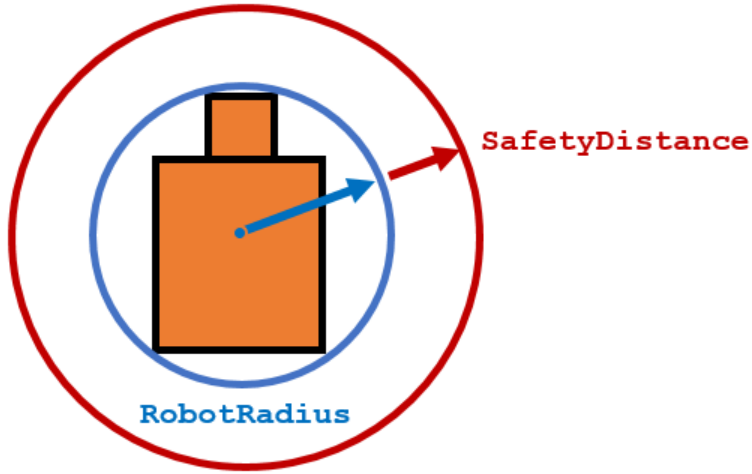
Robot Dimensions

To calculate steering directions, you must specify information about the robot size and its driving capabilities. The VFH algorithm requires only four input parameters for the robot. These parameters are properties of the robotics.VectorFieldHistogram class: `RobotRadius`, `SafetyDistance`, `MinTurningRadius`, and `DistanceLimits`.

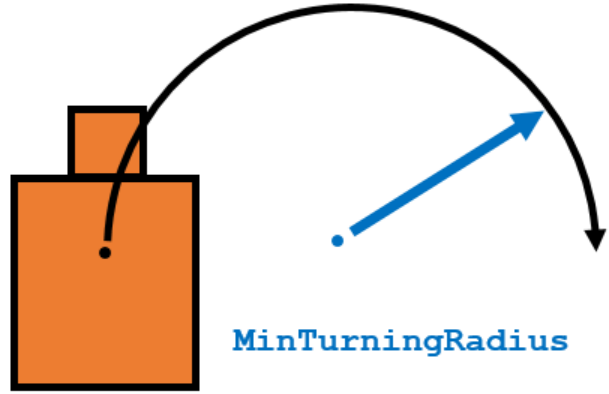
- `RobotRadius` specifies the radius of the smallest circle that can encircle all parts of the robot. This radius ensures that the robot avoids obstacles based on its size.



- `SafetyDistance` optionally specifies an added distance on top of the `RobotRadius`. You can use this property to add a factor of safety when navigating an environment.

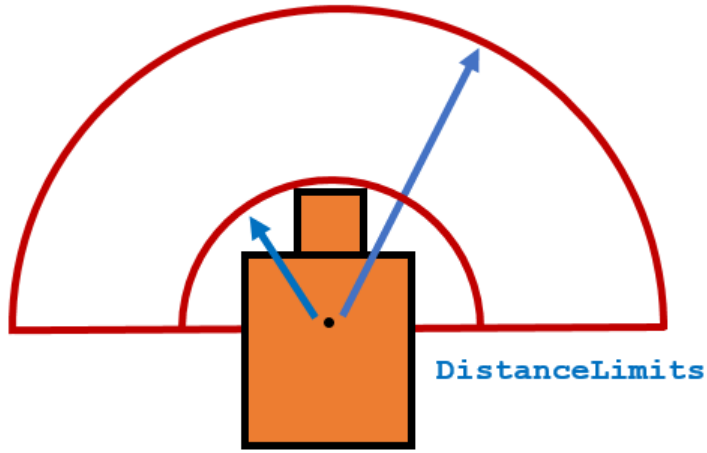


- `MinTurningRadius` specifies the minimum turning radius for the robot traveling at the desired velocity. The robot may not be able to make sharp turns at high velocities. This property factors in navigating around obstacles and gives it enough space to maneuver.



- `DistanceLimits` specifies the distance range that you want to consider for obstacle avoidance. You specify the limits in a two-element vector, `[lower upper]`. The

`lower` limit is used to ignore sensor readings that intersect with parts on the robot, sensor inaccuracies at short distances, or sensor noise. The `upper` limit is the effective range of the sensor or is based on your application. You might not want to consider all obstacles in the full sensor range.



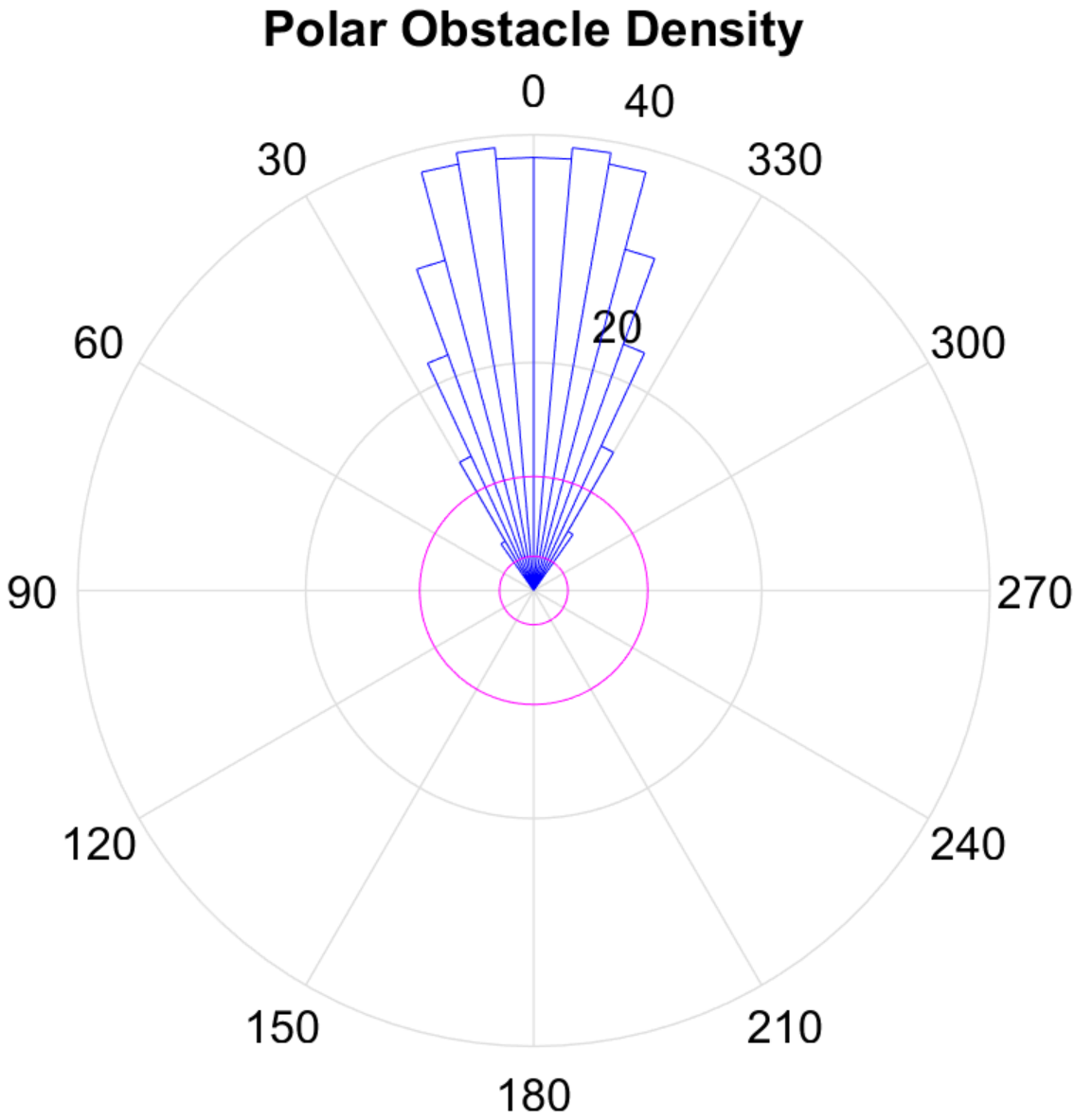
Note: All information about the range sensor readings assumes that your range finder is mounted in the center of your robot. If the range sensor is mounted elsewhere, transform your range sensor readings from the laser coordinate frame to the robot base frame. See "Transform Laser Scan Data" on page 8-2 for an example.

Cost Function Weights

Cost function weights are used to calculate the final steering directions. The VFH algorithm considers multiple steering directions based on your current, previous, and target directions. By setting the `CurrentDirectionWeight`, `PreviousDirectionWeight`, and `TargetDirectionWeight` properties, you can modify the steering behavior of your robot. Changing these weights affects the responsiveness of the robot and how it reacts to obstacles. To make the robot head towards its goal location, set `TargetDirectionWeight` higher than the sum of the other weights. This high `TargetDirectionWeight` value helps to ensure the computed steering direction is close to the target direction. Depending on your application, you might need to tune these weights.

Histogram Properties

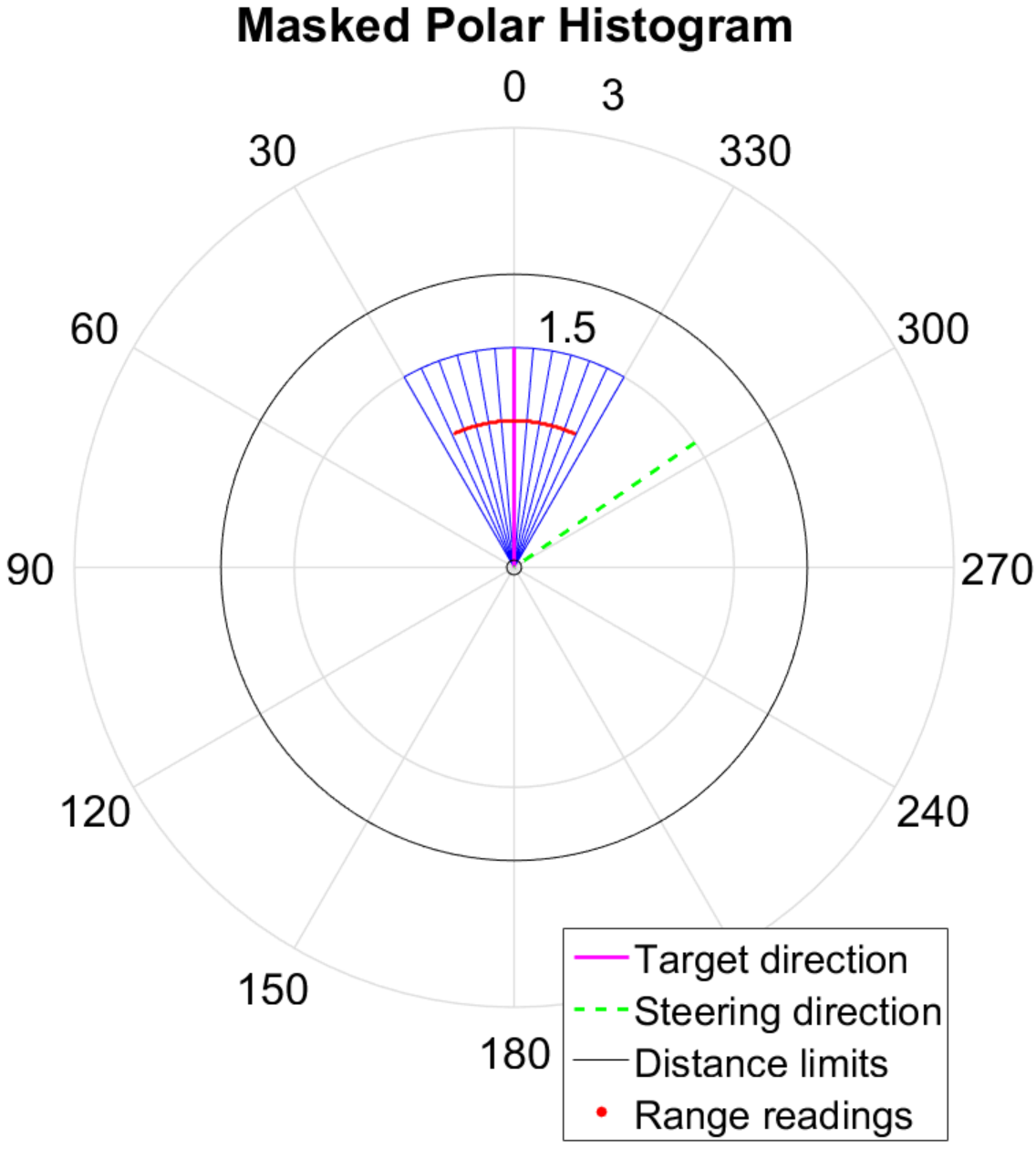
The VFH algorithm calculates a histogram based on the given range sensor data. It takes all directions around the robot and converts them to angular sectors that are specified by the `NumAngularSectors` property. This property is non-tunable and remains fixed once the step method is called on the robotics.VectorFieldHistogram object. The range sensor data is used to calculate a polar density histogram over these angular sectors.


Polar Obstacle Density
0
40
30
330
60
20
300
90
270
120
240
150
210
180

Note: Using a small `NumAngularSectors` value can cause the VFH algorithm to miss smaller obstacles. Missed obstacles do not appear on the histogram.

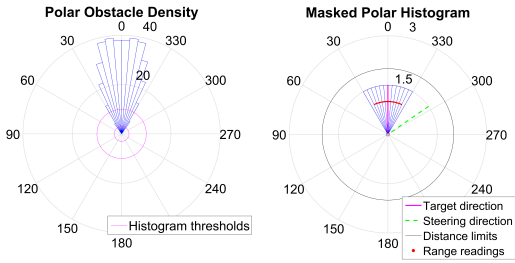
This histogram displays the angular sectors in blue and the histogram thresholds in pink. The `HistogramThresholds` property is a two-element vector that determines the values of the masked histogram, specified as `[lower upper]`. Polar obstacle density values higher than the upper threshold are represented as occupied space (1) in the masked histogram. Values smaller than the lower threshold are represented as free space (0). Values that fall between the limits are set to the values in the previous binary histogram, with the default being free space (0). The masked histogram also factors in the `MinTurningRadius`, `RobotSize`, and `SafetyDistance`.

The polar density plot has the following corresponding masked histogram plot. This plot shows the target and steering directions, range readings, and distance limits.


Masked Polar Histogram
0
3
30
330
60
1.5
300
90
270
120
240
150
180
Target direction
Steering direction
Distance limits
Range readings

Tune Parameters Using show

When working with a robotics.VectorFieldHistogram object, you can visualize the properties and parameters of the algorithm using the robotics.VectorFieldHistogram.show method. This method displays the polar density plot and masked binary histogram. It also displays the algorithm parameters and the output steering direction for the VFH.



You can then tune parameters to help you prototype your obstacle avoidance application. For example, if you see that certain obstacles do not appear in the **Masked Polar Histogram** plot (right), then in the **Polar Obstacle Density** plot, consider adjusting the histogram thresholds to appropriate values. After you make the adjustments in the **Masked Polar Histogram** plot, the range sensor readings, shown in red, should match up with locations in the masked histogram (blue). Also, you can see the target and steering directions. You specify the target direction. The steering direction is the main output from the VFH algorithm. Adjusting the “Cost Function Weights” on page 7-37 can help you tune the output of the final steering direction.

Although you can use the `show` method in a loop, it slows computation speed due to the graphical plotting. If you are running this algorithm for real-time applications, get and display the VFH data in separate operations.

See Also

robotics.VectorFieldHistogram | robotics.VectorFieldHistogram.show

Monte Carlo Localization Algorithm

In this section...

Overview

The Monte Carlo Localization (MCL) algorithm is used to estimate the position and orientation of a robot in its environment using a known map of the environment, range sensor data, and odometry sensor data. See the robotics.MonteCarloLocalization class page to see how to construct a `MonteCarloLocalization` object and use this algorithm.

To localize the robot, the MCL algorithm uses a particle filter to estimate its position. The particles represent the distribution of the likely states for the robot. Each particle represents a possible robot state. The particles converge around a single location as the robot moves in the environment and senses different parts of the environment using a range sensor. The robot motion is sensed using an odometry sensor.

The particles are updated in this process:

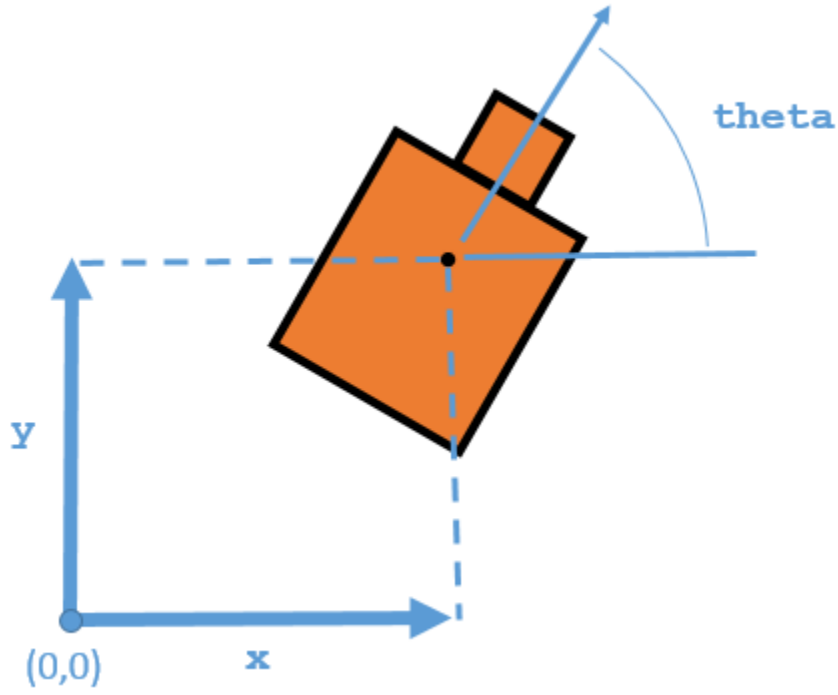
1. Particles are propagated based on the change in the pose and the specified motion model, `MotionModel`.
2. The particles are assigned weights based on the likelihood of receiving the range sensor reading for each particle. This reading is based on the sensor model you specify in `SensorModel`.
3. Based on these weights, a robot state estimate is extracted based on the particle weights. The group of particles with the highest weight are used to estimate the position of the robot.
4. Finally, the particles are resampled based on the specified `ResamplingInterval`. Resampling adjusts particle positions and improves performance by adjusting the number of particles used. It is a key feature for adjusting to changes and keeping particles relevant for estimating the robot state.

The algorithm outputs the estimated pose and covariance. These estimates are the mean and covariance of the highest weighted cluster of particles. For continuous tracking, you can repeat these steps in a loop to continuously propagate particles, evaluate their likelihood, and get the best state estimate.

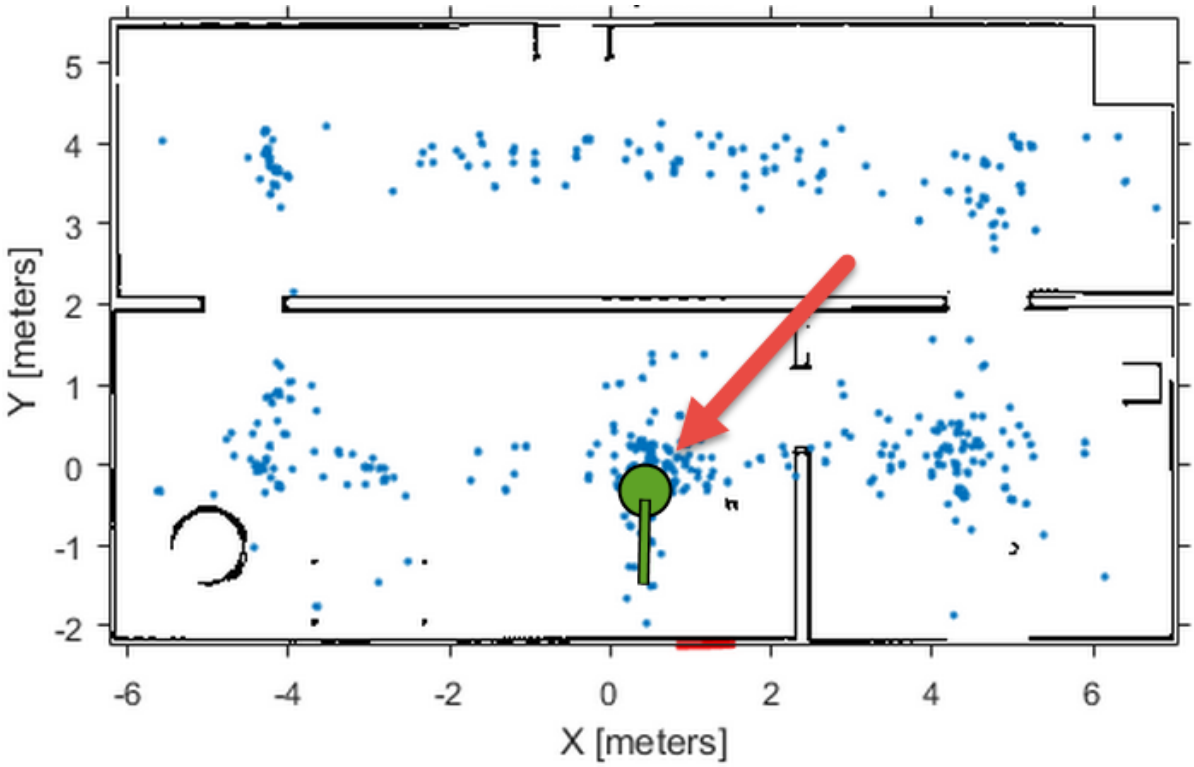
For more information on particle filters as a general application, see "Particle Filter Workflow" on page 7-14.

State Representation

When working with a localization algorithm, the goal is to estimate the state of your system. For robotics applications, this estimated state is usually a robot pose. For the `MonteCarloLocalization` object, you specify this pose as a three-element vector. The pose corresponds to an *x*-*y* position, `[x y]`, and an angular orientation, `theta` .



The MCL algorithm estimates these three values based on sensor inputs of the environment and a given motion model of your system. The output of the `step` method of the `MonteCarloLocalization` object includes the `pose`, which is the best estimated state of the `[x y theta]` values. Particles are distributed around an initial pose, `InitialPose`, or sampled uniformly using global localization. The pose is computed as the mean of the highest weighted cluster of particles once these particles have been corrected based on measurements.



This plot shows the highest weighted cluster and the final robot pose displayed over the samples particles in green. With more iterations of the MCL algorithm and measurement corrections, the particles should converge to the true location of the robot. However, it is possible that particle clusters can have high weights for false estimates and converge on the wrong location. If the wrong convergence occurs, resample the particles by resetting the MCL algorithm with an updated `InitialPose`.

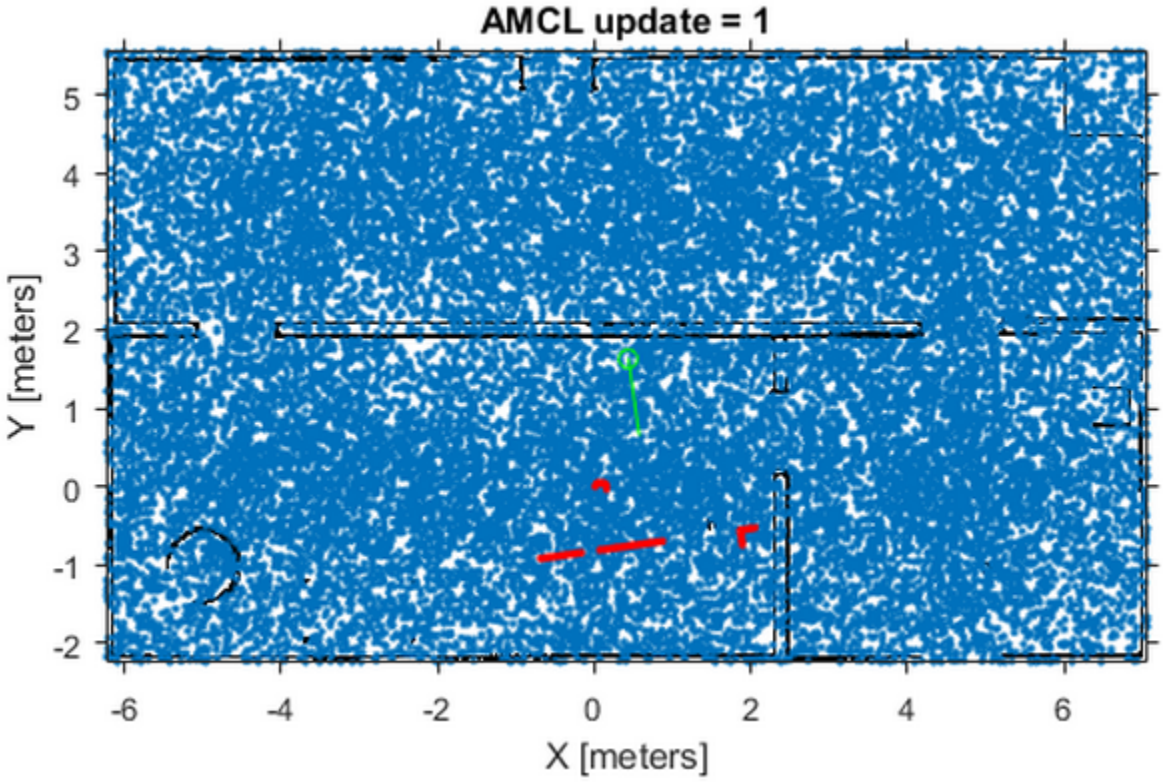
Initialization of Particles

When you first create the `MonteCarloLocalization` algorithm, specify the minimum and maximum particle limits by using the `ParticleLimits` property. A higher number of particles increases the likelihood that the particles converge on the actual location. However, a lower particle number is faster. The number of particles adjusts dynamically within the limits based on the weights of particle clusters. This adjustment helps to reduce the number of particles over time so localization can run more efficiently.

Particle Distribution

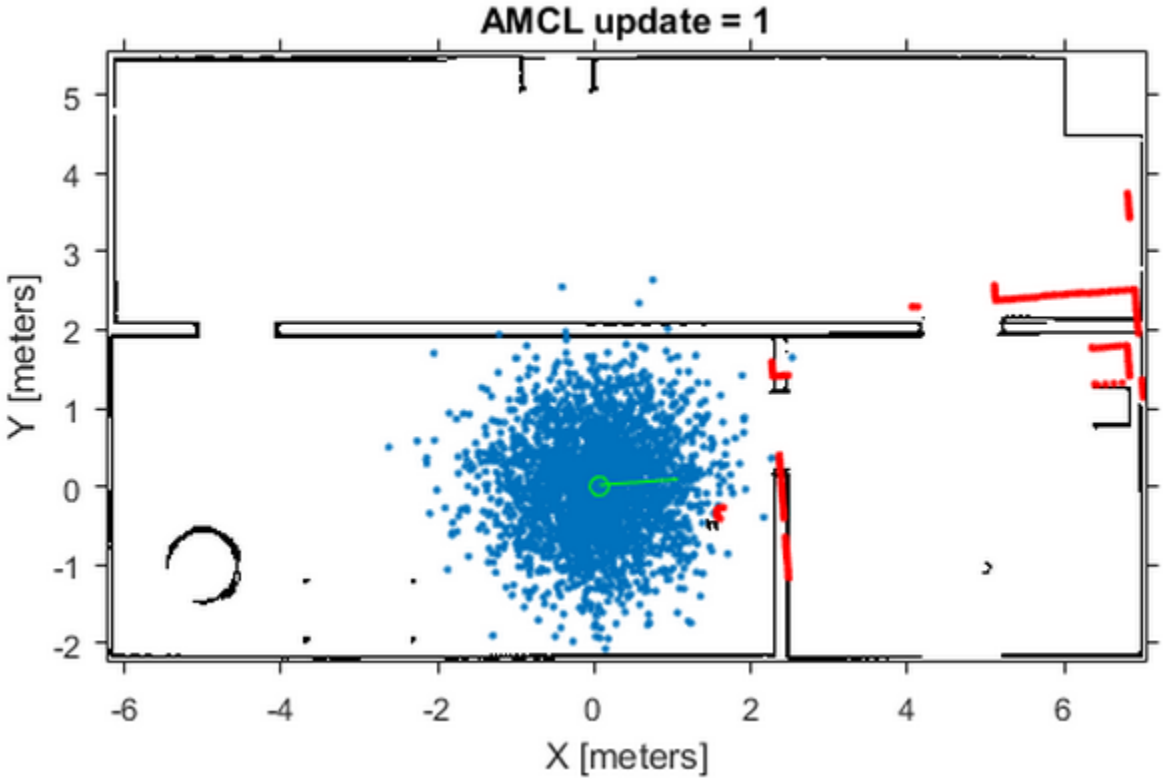
Particles must be sampled across a specified distribution. To initialize particles in the state space, you can use either an initial pose or global localization. With global localization, you can uniformly distribute particles across your expected state space (pulled from the `Map` property of your`SensorModel` object). In the default MCL object, set the `GlobalLocalization` property to `true`.

```
mcl = robotics.MonteCarloLocalization;
mcl.GlobalLocalization = true;
```



However, global localization requires a larger number of particles to effectively sample particles across the state space and increase the likelihood of successful convergence on the actual state. This large distribution greatly reduces initial performance until particles begin to converge and particle number can be reduced.

By default, global localization is set to `false`. Without global localization, you must specify the `InitialPose` and `InitialCovariance` properties, which helps to localize the particles. Using this initial pose, particles are more closely grouped around an estimated state. A close grouping of particles enables you to use fewer of them, and increases the speed and accuracy of tracking during the first iterations.



These images were taken from the "Localize TurtleBot using Monte Carlo Localization" example, which shows how to use the MCL algorithm with the TurtleBot® in a known environment.

Resampling Particles and Updating Pose

To continuously localize your robot, you must resample the particles and update the algorithm. Use the `UpdateThreshold` and `ResamplingInterval` properties to control when resampling and updates to the estimated state occur.

The `UpdateThreshold` is a three-element vector that defines the minimum change in the robot pose, `[x y theta]`, to trigger an update. Changing a variable by more than this minimum triggers an update, causing the `step` method to return a new state estimate. This change in robot pose is based on the odometry, which is specified in the

`step` method. Tune these thresholds based on your sensor properties and the motion of your robot. Random noise or minor variations greater than your threshold can trigger an unnecessary update and affect your performance. The `ResamplingInterval` property defines the number of updates needed to trigger particle resampling. For example, a resampling interval of 2 resamples at every other update.

The benefit of resampling particles is that you update the possible locations that contribute to the final estimate. Resampling redistributes the particles based on their weights and evolves particles based on the "Motion Model" on page 7-51. In this process, the particles with lower weight are eliminated, helping the particles converge to the true state of the robot. The number of particles dynamically changes to improve speed or tracking.

The performance of the algorithm depends on proper resampling. If particles are widely dispersed and the initial pose of the robot is not known, the algorithm maintains a high particle count. As the algorithm converges on the true location, it reduces the number of particles and increases the speed of performance. You can tune your `ParticleLimits` property to limit the minimum and maximum particles used to help with the performance.

Motion and Sensor Model

The motion and sensor models for the MCL algorithm are similar to the `StateTransitionFcn` and `MeasurementLikelihoodFcn` functions for the robotics.ParticleFilter object, which are described in "Particle Filter Parameters" on page 7-7. For the MCL algorithm, these models are more specific to robot localization. After calling `step`, to make changes to the `MotionModel` or `SensorModel` properties, you must first call `release` on your object.

Sensor Model

By default, the `MonteCarloLocalization` uses a robotics.LikelihoodFieldSensorModel object as the sensor model. This sensor model contains parameters specific to the range sensor used, 2-D map information for the robot's environment, and measurement noise characteristics. The sensor model uses the parameters with range measurements to compute the likelihood of the measurements given the current position of the robot. Without factoring in these parameters, some measurement errors can skew the state estimate or increase weight on irrelevant particles.

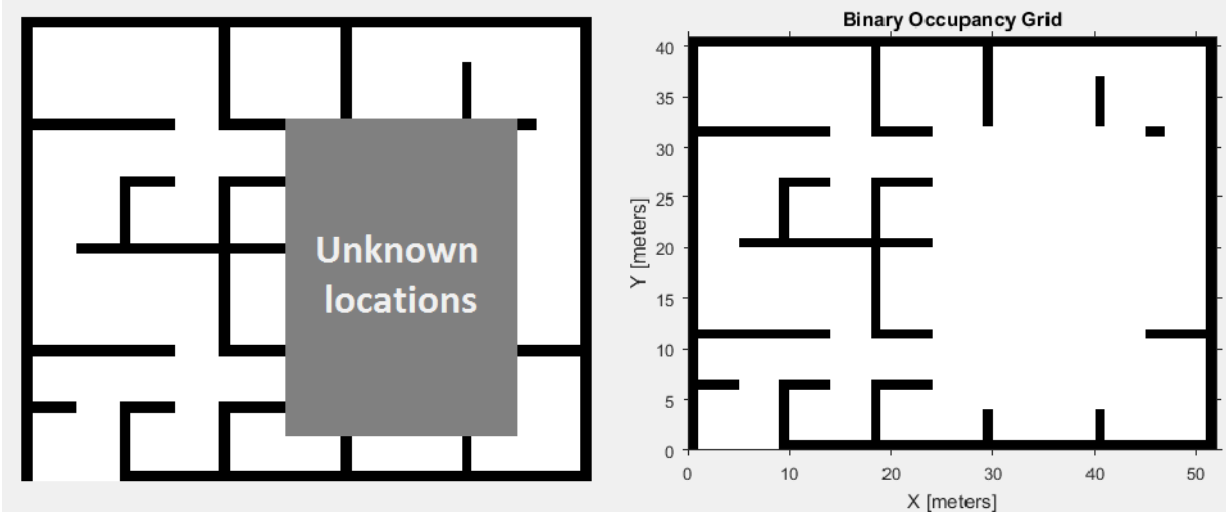
The range sensor properties are:

- `SensorPose` – The pose of the range sensor relative to the robot location. This pose is used to transform the range readings into the robot's coordinate frame.
- `SensorLimits` – The minimum and maximum range limits. Measurement outside of these ranges are not factored into the likelihood calculation.
- `NumBeams` – Number of beams used to calculate likelihood. You can improve performance speed by reducing the number of beams used.

Range measurements are also known to give false readings due to system noise or other environmental interference. To account for the sensor error, specify these parameters:

- `MeasurementNoise` – Standard deviation for measurement noise. This deviation applies to the range reading and accounts for any interference with the sensor. Set this value based on information from your range sensor.
- `RandomMeasurementWeight` — Weight for probability of random measurement. Set a low probability for random measurements. The default is 0.05.
- `ExpectedMeasurementWeight` — Weight for probability of expected measurement. Set a high probability for expected measurements. The default is 0.95.

The sensor model also stores a map of the robot's environment as an occupancy grid. Use robotics.BinaryOccupancyGrid to specify your map with occupied and free spaces. Set any unknown spaces in the map as free locations. Setting them to free locations prevents the algorithm from matching detected objects to these areas of the map.



Also, you can specify `MaximumLikelihoodDistance`, which limits the area for searching for obstacles. The value of `MaximumLikelihoodDistance` is the maximum distance to the nearest obstacle that is used for likelihood computation.

Motion Model

The motion model for robot localization helps to predict how particles should evolve throughout time when resampling. It is a representation of robot kinematics. The motion model included by default with the MCL algorithm is an odometery-based differential drive motion model (robotics.OdometryMotionModel). Without a motion model, predicting the next step is more difficult. It is important to know the capabilities of your system so that the localization algorithm can plan particle distributions to get better state estimates. Be sure to consider errors from the wheel encoders or other sensors used to measure the odometry. The errors in the system define the spread of the particle distribution.

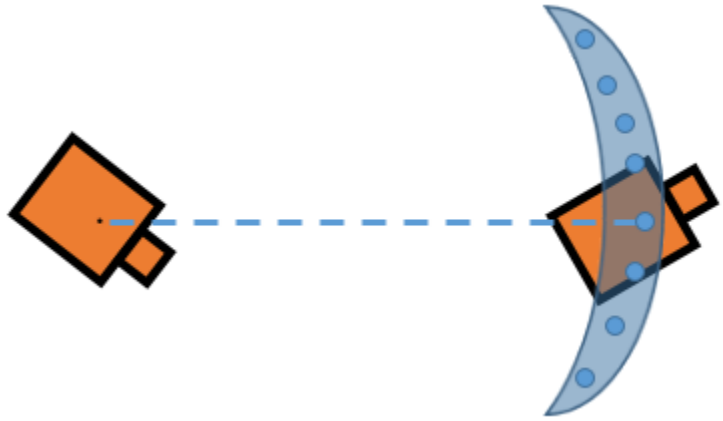
You can specify the error expected based on the motion of your robot as a four-element vector, `Noise`. These four elements are specified as weights on the standard deviations for [1]:

- Rotational error due to rotational motion
- Rotational error due to translational motion
- Translational error due to translational motion
- Translational error due to rotational motion

For differential drive robots, when a robot moves from a starting pose to a final pose, the change in pose can be treated as:

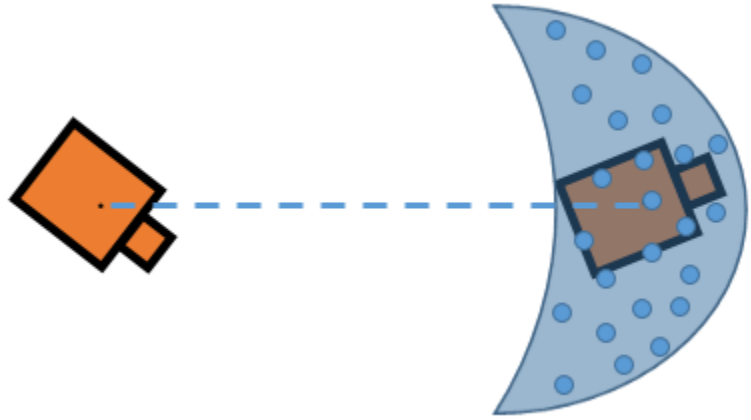
1. Rotation to the final position
2. Translation in a direct line to the final position
3. Rotation to the goal orientation

Assuming these steps, you can visualize the effect of errors in rotation and translation. Errors in the initial rotation result in your possible positions being spread out in a C-shape around the final position.



Large translational errors result in your possible positions being spread out around the direct line to the final position.

Large errors in both translation and rotation can result in wider-spread positions.



Also, rotational errors affect the orientation of the final pose. Understanding these effects helps you to define the Gaussian noise in the `Noise` property of the `MotionModel` object for your specific application. As the images show, each parameter does not directly control the dispersion and can vary with your robot configuration and geometry. Also, multiple pose changes as the robot navigates through your environment can increase the affects of these errors over many different steps. By accurately defining these parameters, particles are distributed appropriately to give the MCL algorithm enough hypotheses to find the best estimate for the robot's location.

References

[1] Thrun, Sebastian, and Dieter Fox. *Probabilistic Robotics.* 3rd ed. Cambridge, Mass: MIT Press, 2006. p.136.

See Also

robotics.MonteCarloLocalization | robotics.MonteCarloLocalization.step | robotics.LikelihoodFieldSensorModel | robotics.OdometryMotionModel

Related Examples

- "Localize TurtleBot using Monte Carlo Localization"

8

Application Design

Transform Laser Scan Data

This example shows how to transform laser scan data using a ROS transformation tree. When working with laser scan data, your sensor might not be mounted in the center of the robot. Many algorithms make this assumption, so that you might need to transform your data so it is relative to the robot center. This example uses a ROS transformation tree to receive the transformations between different coordinate frames. To transform the sensor data, you must be connected to a ROS network and have transformations available.

Create the ROS transformation tree from the ROS network using `rostf`. Get the transform between `'/laser'` and `'/base_link'` coordinate frames. Change the coordinate frame names based on your robot configuration.

```
tftree = rostf;
tf = getTransform(tftree,'/laser','/base_link');
```

Get the rotation and translation matrix from the transform.

```
quat = [tf.Transform.Rotation.W tf.Transform.Rotation.X ...
    tf.Transform.Rotation.Y tf.Transform.Rotation.Z];
rotm = quat2rotm(quat);
trvec = [tf.Transform.Translation.X tf.Transform.Translation.Y ...
    tf.Transform.Translation.Z];
```

Create a homogenous transform from the translation and rotation matrices.

```
tform = trvec2tform(trvec);
tform(1:3,1:3) = rotm(1:3,1:3);
```

Set up a subscriber to get laser scan data. Get the laser scan data as Cartesian points. Pad the points with zeros and convert them to homogeneous coordinates.

```
scansub = rossubscriber('/scan');
scan = receive(scansub)
cartScanData = scan.readCartesian;
cartScanData(:,3) = 0;
homScanData = cart2hom(cartScanData);
```

Apply the homogeneous transform and convert scan data back to Cartesian points.

```
trPts = tform*homScanData';
cartScanDataTransformed = hom2cart(trPts');
```

Get the polar angles and ranges from the Cartesian points.

```
[angles,ranges] = cart2pol(cartScanDataTransformed(:,1), cartScanDataTransformed(:,2));
```

See Also

robotics.VectorFieldHistogram | `apply` | `getTransform` | `rostf`

Obstacle Avoidance with Turtlebot and VFH

This example shows how to use a TurtleBot with Vector Field Histograms (VFH) in order to perform obstacle avoidance when driving a robot in an environment. The robot wanders by driving forward until obstacles get in the way. The `robotics.ros.VectorFieldHistogram` class computes steering directions to avoid objects while trying to drive forwards.

Optional: If you do not already have a TurtleBot (simulated or real) set up, install a virtual machine with the Gazebo simulator and TurtleBot package. See "Get Started with Gazebo and a Simulated TurtleBot" to install and set up a TurtleBot in Gazebo.

Connect to the TurtleBot using the IP address obtained from setup.

```
rosinit('192.168.1.1')
```

Create a publisher and subscriber to share information with the VFH class. The subscriber receives the laser scan data from the robot. The publisher sends velocity commands to the robot.

The topics used are for the simulated TurtleBot. Adjust the topic names for your specific robot.

```
laserSub = rossubscriber('/scan');
[velPub, velMsg] = rospublisher('/mobile_base/commands/velocity');
```

Set up VFH object for obstacle avoidance. Specify algorithm properties for robot specifications. Set target direction to `0` in order to drive straight.

```
vfh = robotics.VectorFieldHistogram;
vfh.DistanceLimits = [0.05 1];
vfh.RobotRadius = 0.1;
vfh.MinTurningRadius = 0.2;
vfh.SafetyDistance = 0.1;

targetDir = 0;
```

Create a loop that collects data, calculates steering direction, and drives the robot. Use the ROS subscriber to collect laser scan data. Calculate the steering direction with the VFH step method based on the input laser scan data. Convert the steering direction to a desired linear and an angular velocity. If a steering direction is not found, the robot stops and searches by rotating in place. Drive the robot by sending a message containing the angular velocity and the desired linear velocity using the ROS publisher.

```
while true
 % Get laser scan data
 laserScan = receive(laserSub);
 ranges = double(laserScan.Ranges);
 angles = double(laserScan.readScanAngles);

 % Call VFH step method to computer steering direction
 steerDir = vfh.step(ranges, angles, targetDir);

 % Calculate velocities
 if ~isnan(steerDir) % If steering direction is valid
  desiredV = 0.2;
  w = exampleHelperComputeAngularVelocity(steerDir, 1);
 else % Stop and search for valid direction
  desiredV = 0.0;
  w = 0.5;
 end

 % Assign and send velocity commands
 velMsg.Linear.X = desiredV;
 velMsg.Angular.Z = w;
 velPub.send(velMsg);
end
```

This code shows how you can use the Robotics System Toolbox algorithms to control robots and react to dynamic changes in their environment. The loop should operate continuously until you stop it, but other conditions can be set to exit the loop (i.e. robot position or time elapsed) based on information on the ROS network.

See Also

robotics.VectorFieldHistogram | `rospublisher` | `rossubscriber`

Related Examples

- “Get Started with Gazebo and a Simulated TurtleBot”
- “Communicate with the TurtleBot”

Execute Code at a Fixed-Rate

In this section...

Introduction

Using the robotics.Rate object or the `rosrate` function allows you to time iterations of your robotics applications. By executing code at constant intervals, you can accurately time and schedule tasks. These examples show different applications for the `Rate` object including its uses with ROS and sending commands for robot control.

Depending on your application, the `rosrate` function and robotics.Rate object offer different options. If you would like to execute code based on the system time of your computer, create an object using robotics.Rate. However, if you are connected to a ROS network and want to base code execution on the ROS time, you can use the `rosrate` function.

Send Fixed-rate Control Commands To A Robot

This example shows to send regular commands to a robot at a fixed rate. It uses the `Rate` object to exectue a loop that publishes `std_msgs/Twist` messages to the network at a regular interval.

Setup ROS network. Specify the IP address if your ROS network already exists.

```
rosinit
```

```
Initializing ROS master on http://AH-SRADFORD:11311/.
Initializing global node /matlab_global_node_73378 with NodeURI http://AH-SRADFORD:640
```

Setup publisher and message for sending `Twist` commands.

```
[pub,msg] = rospublisher('/cmd_vel',rostype.geometry_msgs_Twist);
```

```
msg.Linear.X = 0.5;
msg.Angular.Z = -0.5;
```

Create `Rate` object with specified loop parameters.

```
desiredRate = 10;
rate = robotics.Rate(desiredRate)
rate.OverrunAction = 'drop';
```

```
rate =

  Rate with properties:

         DesiredRate: 10
      DesiredPeriod: 0.1000
      OverrunAction: 'slip'
    TotalElapsedTime: 4.2067e-04
          LastPeriod: NaN
```

Run loop and hold each iteration using `waitfor(rate)`. Send the `Twist` message inside the loop. Reset the `Rate` object before the loop to reset timing.

```
reset(rate)

while rate.TotalElapsedTime < 10
   send(pub,msg)
   waitfor(rate);
end
```

View statistics of fixed-rate execution. Look at `AveragePeriod` to verify the desired rate was maintained.

```
statistics(rate)
```

```
ans =

              Periods: [1x100 double]
           NumPeriods: 100
        AveragePeriod: 0.1000
    StandardDeviation: 2.3231e-04
          NumOverruns: 0
```

Shut down ROS network

```
rosshutdown
```

```
Shutting down global node /matlab_global_node_73378 with NodeURI http://AH-SRADFORD:640
Shutting down ROS master on http://AH-SRADFORD:11311/.
```

Fixed-rate Publishing of ROS Image Data

This example shows how to regularly publish and receive image messages using ROS and the `rosrate` function. The `rosrate` function creates a `Rate` object to regularly access the `/camera/rgb/image_raw` topic on the ROS network using a subscriber. The rgb image is converted to a grayscale using `rgb2gray` and republished at a regular interval. Parameters such as the IP address and topic names vary with your robot and setup.

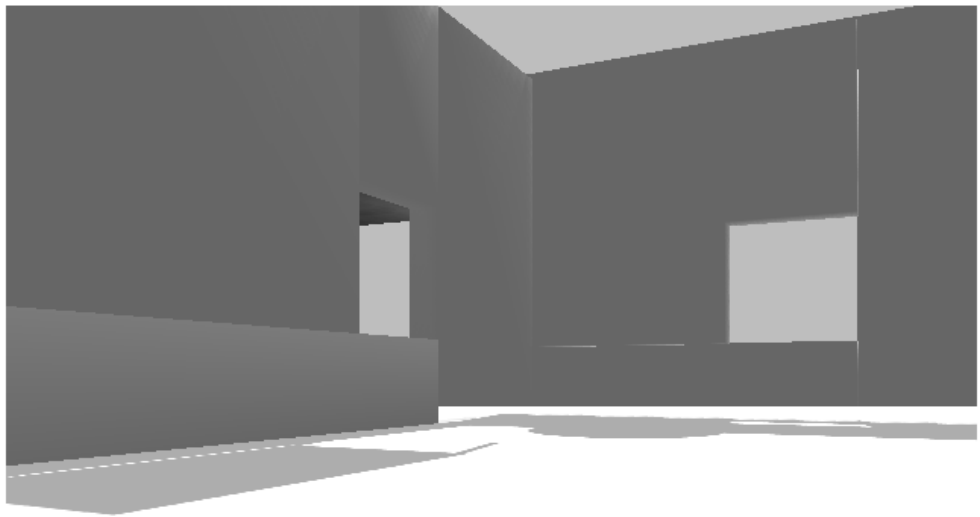
Connect to ROS network. Setup subscriber, publisher, and data message.

```
ipaddress = '172.28.194.188'; % Replace with your network address
rosinit(ipaddress)
sub = rossubscriber('/camera/rgb/image_raw');
pub = rospublisher('/camera/gray/image_gray','sensor_msgs/Image');
msgGray = rosmessage('sensor_msgs/Image');
msgGray.Encoding = 'mono8';
```

```
Initializing global node /matlab_global_node_04172 with NodeURI http://172.28.194.235:6
```

Receive the first image message. Extract image and convert to a grayscale image. Display grayscale image and publish the message.

```
msgImg = receive(sub);

img = readImage(msgImg);
grayImg = rgb2gray(img);
imshow(grayImg)

writeImage(msgGray,grayImg)
send(pub,msgGray)
```



Create ROS `Rate` object to execute at 10 Hz. Set a loop time and the `OverrunAction` for handling

```
desiredRate = 10;
loopTime = 5;
overrunAction = 'slip';
rate = rosrate(desiredRate);
r.OverrunAction = overrunAction;
```

Begin loop to receive, process and send messages every 0.1 seconds (10 Hz). Reset the `Rate` object before beginning.

```
reset(rate)
```

```
for i = 1:desiredRate*loopTime

    msgImg = receive(sub);

    img = readImage(msgImg);
    grayImg = rgb2gray(img);
    writeImage(msgGray,grayImg)

    send(pub,msgGray)

    waitfor(rate);
end
```

View the statistics for the code execution. `AveragePeriod` and `StandardDeviation` show how well the code maintained the `desiredRate`. `OverRuns` occur when data processing takes longer than the desired period length.

```
statistics(rate)
```

```
ans =

              Periods: [1x50 double]
           NumPeriods: 50
        AveragePeriod: 0.1024
    StandardDeviation: 0.0193
          NumOverruns: 1
```

Shut down ROS node

```
rosshutdown
```

```
Shutting down global node /matlab_global_node_04172 with NodeURI http://172.28.194.235
```

Overrun Actions for Fixed Rate Execution

The `Rate` object uses the `OverrunAction` property to decide how to handle code that takes longer than the desired period to operate. The options are `'slip'` (default) or `'drop'`. This example shows how the `OverrunAction` affects code execution.

Setup desired rate and loop time. `slowFrames` is an array of times when the loop should be stalled longer than the desired rate.

```
desiredRate = 1;
loopTime = 20;
slowFrames = [3 7 12 18];
```

Create the `Rate` object and specify the `OverrunAction` property. `'slip'` indicates that the `waitfor` function will return immediately if the time for `LastPeriod` is greater than the `DesiredRate` property.

```
rate = robotics.Rate(desiredRate);
rate.OverrunAction = 'slip';
```

Reset `Rate` object and begin loop. This loop will execute at the desired rate until the loop time is reached. When the `TotalElapsedTime` reaches a slow frame time, it will stall for longer than the desired period.

```
reset(rate);

while rate.TotalElapsedTime < loopTime
    if ~isempty(find(slowFrames == floor(rate.TotalElapsedTime)))
        pause(desiredRate + 0.1)
    end
    waitfor(rate);
end
```

View statistics on the `Rate` object. Notice the number of periods.

```
stats = statistics(rate)
```

```
stats =

             Periods: [1x20 double]
          NumPeriods: 20
       AveragePeriod: 1.0205
   StandardDeviation: 0.0432
         NumOverruns: 4
```

Change the `OverrunAction` to `'drop'`. `'drop'` indicates that the `waitfor` function will return at the next time step, even if the `LastPeriod` is greater than the `DesiredRate` property. This effectively drops the iteration that was missed by the slower code execution.

```
rate.OverrunAction = 'drop';
```

Reset `Rate` object and begin loop.

```
reset(rate);

while rate.TotalElapsedTime < loopTime
    if ~isempty(find(slowFrames == floor(rate.TotalElapsedTime)))
        pause(1.1)
    end
    waitfor(rate);
end
stats2 = statistics(rate)


stats2 =

             Periods: [1x16 double]
          NumPeriods: 16
       AveragePeriod: 1.2500
   StandardDeviation: 0.4475
         NumOverruns: 4
```

Using the `'drop'` over run action resulted in 16 periods when the `'slip'` resulted in 20 periods. This difference is because the `'slip'` did not wait until the next interval based on the desired rate. Essentially, using `'slip'` tries to keep the `AveragePeriod` property as close to the desired rate. Using `'drop'` ensures the code will execute at an even interval relative to `DesiredRate` with some iterations being skipped.

See Also

robotics.Rate | robotics.Rate.waitfor | Using ROS Rate Objects | `rosrate`

9

Code Generation

Code Generation from MATLAB Code

Several Robotics System Toolbox functions are enabled to generate C/C++ code. Code generation from MATLAB code requires the MATLAB Coder™ product. To generate code from robotics functions, follow these steps:

- Write your function or application that uses Robotics System Toolbox functions that are enabled for code generation. For code generation, some of these functions have requirements that you must follow. See “Code Generation Support, Usage Notes and Limitations” on page 9-4.
- Add the `%#codegen` directive to your MATLAB code.
- Follow the workflow for code generation from MATLAB code using either the MATLAB Coder app or the command-line interface.

Using the app, the basic workflow is:

1. Set up a project. Specify your top-level functions and define input types.

   The app screens your code for code generation readiness. It reports issues such as a function that is not supported for code generation.
2. Check for run-time issues.

   The app generates and runs a MEX version of your function. This step detects issues that can be hard to detect in the generated C/C++ code.
3. Configure the code generation settings for your application.
4. Generate C/C++ code.
5. Verify the generated C/C++ code. If you have an Embedded Coder® license, you can use software-in-the-loop execution (SIL) or processor-in-the-loop (PIL) execution.

For a tutorial, see “C Code Generation Using the MATLAB Coder App”.

Using the command-line interface, the basic workflow is:

- To detect issues and verify the behavior of the generated code, generate a MEX version of your function.
- Use `coder.config` to create a code configuration object for a library or executable.
- Modify the code configuration object properties as required for your application.
- Generate code using the `codegen` command.

- Verify the generated code. If you have an Embedded Coder license, you can use software-in-the-loop execution (SIL) or processor-in-the-loop (PIL) execution.

For a tutorial, see “C Code Generation at the Command Line”.

More About

- “Code Generation Support, Usage Notes and Limitations” on page 9-4

Code Generation Support, Usage Notes and Limitations

To generate code from MATLAB code that contains Robotics System Toolbox functions, classes, or System objects, you must have the MATLAB Coder software.

The following functions support code generation using MATLAB Coder, but have the following limitations.

Name	Remarks and Limitations
Algorithm Design	
robotics.PurePursuit	Supports MATLAB Function block: No
robotics.VectorFieldHistogram	Supports MATLAB Function block: No
robotics.ParticleFilter	Supports MATLAB Function block: No
Coordinate System Transformations	
`angdiff`	Supports MATLAB Function block: Yes
`axang2quat`	Supports MATLAB Function block: Yes
`axang2rotm`	Supports MATLAB Function block: Yes
`axang2tform`	Supports MATLAB Function block: Yes
`cart2hom`	Supports MATLAB Function block: Yes
`eul2quat`	Supports MATLAB Function block: Yes
`eul2rotm`	Supports MATLAB Function block: Yes
`eul2tform`	Supports MATLAB Function block: Yes
`hom2cart`	Supports MATLAB Function block: Yes
`quat2axang`	Supports MATLAB Function block: Yes
`quat2eul`	Supports MATLAB Function block: Yes
`quat2rotm`	Supports MATLAB Function block: Yes
`quat2tform`	Supports MATLAB Function block: Yes
`rotm2axang`	Supports MATLAB Function block: Yes
`rotm2eul`	Supports MATLAB Function block: Yes
`rotm2quat`	Supports MATLAB Function block: Yes
`rotm2tform`	Supports MATLAB Function block: Yes

Name	Remarks and Limitations
`tform2axang`	Supports MATLAB Function block: Yes
`tform2eul`	Supports MATLAB Function block: Yes
`tform2quat`	Supports MATLAB Function block: Yes
`tform2rotm`	Supports MATLAB Function block: Yes
`tform2trvec`	Supports MATLAB Function block: Yes
`trvec2tform`	Supports MATLAB Function block: Yes

More About

- “Code Generation from MATLAB Code” on page 9-2