Robotics System Toolbox™ Release Notes

MATLAB® & SIMULINK®

MathWorks®

How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

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

Phone: 508-647-7000

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

Robotics System Toolbox™ Release Notes

Contents

R2016a

Monte Carlo Localization Algorithm: Estimate robot location in a known map 1-2

Particle Filter Algorithm: Estimate state for nonlinear systems 1-2

Fixed-Rate Execution: Run MATLAB code at a constant rate 1-2

Robotics System Toolbox Support Package for TurtleBot based Robots: Connect to TurtleBot hardware 1-2

String support for ROS parameters in Simulink 1-3

String array support for ROS messages in Simulink 1-3

Code generation from Simulink using Simulink Coder 1-3

`roboticsSupportPackages` function replaced with `roboticsAddons` 1-3

R2015aSP1

Bug Fixes

R2015b

Vector Field Histogram Plus (VFH+) obstacle avoidance algorithm 3-2

Access to ROS parameters from Simulink 3-2

Code generation for coordinate transforms and select robotics algorithms 3-2

R2015a

Path planning, path following, and map representation algorithms 4-2

Functions for converting between different rotation and translation representations 4-2

Bidirectional communication with live ROS-enabled robots 4-2

Interface to Gazebo and other ROS-enabled simulators 4-2

Data import from rosbag log files 4-2

ROS node generation from Simulink models (with Embedded Coder) 4-2

R2016a

Version: 1.2

New Features

Compatibility Considerations

Monte Carlo Localization Algorithm: Estimate robot location in a known map

Monte Carlo Localization utilizes a particle filter to localize a robot in a known environment. You can supply a `BinaryOccupancyGrid` object of your map and range sensor data from the robot to the robotics.MonteCarloLocalization object to estimate the pose (location and orientation) of the robot. You have the option of using global localization or specifying an initial pose to improve performance. As sensor data is supplied to the algorithm, particles converge on the best estimate of the robot location.

Particle Filter Algorithm: Estimate state for nonlinear systems

The robotics.ParticleFilter class enables you to create a particle filter for state estimation. The algorithm uses particles and sensor data to try to match the posterior distribution of the current state. It first predicts the current state based on a given system model and then corrects the estimate based on sensor data inputs. You can specify a fixed number of particles to use, number of state variables to estimate, and your method for final estimation based on the particle weights. You can customize your particle filter by giving a state transition function and measurement likelihood model to match your system.

Fixed-Rate Execution: Run MATLAB code at a constant rate

Execute loops at a constant rate based off either your system time or ROS time. By creating a robotics.Rate object, you can call `waitfor` to pause a loop until the next time step. This feature ensures that loops are run at a fixed rate when accurate timing of commands is required.

You can also use `rosrate` to base timing off the current time published in a ROS network. Therefore, messages and control commands can be published at a fixed rate to a ROS-enabled system.

Robotics System Toolbox Support Package for TurtleBot based Robots: Connect to TurtleBot hardware

Robotics System Toolbox™ Support Package for TurtleBot® based Robots allows robotics researchers to acquire sensor data from TurtleBot-based robots (either simulated or physical robots). You can use the data for visualization and analysis, and send commands to control the robots.

String support for ROS parameters in Simulink

Support for using strings as ROS parameters is now available in Simulink®. When using strings, they must be cast as a `uint8` array of ASCII values. See “ROS String Parameters” for more information.

String array support for ROS messages in Simulink

You can now use an array of strings when using the `Publish`, `Subscribe`, and `Blank Message` blocks to create, send, and receive messages using a ROS network in Simulink.

Code generation from Simulink using Simulink Coder

You can now generate standalone ROS nodes from Simulink models with just Simulink Coder™. If you have Embedded Coder®, you can customize the generated code with additional optimization, readability, and code configuration options.

`roboticsSupportPackages` function replaced with `roboticsAddons`

The `roboticsSupportPackages` function is no longer available. Instead, use `roboticsAddOns` to access Add-ons for Robotics System Toolbox.

R2015aSP1

Version: 1.0.1

Bug Fixes

R2015b

Version: 1.1

New Features

Vector Field Histogram Plus (VFH+) obstacle avoidance algorithm

The VFH+ obstacle avoidance algorithm is a reactive algorithm that calculates obstacle-free robot movements using range sensor information. You can use this algorithm to have your robot avoid unknown obstacles while driving through dynamic or partially known environments. See robotics.VectorFieldHistogram for more information.

Access to ROS parameters from Simulink

Simulink workflows now support ROS parameters. You can get and set parameter values using the new Get Parameter and Set Parameter blocks.

Code generation for coordinate transforms and select robotics algorithms

For select Robotics System Toolbox algorithms, you can now generate C/C++ code using MATLAB® Coder. You can create MEX-files and shared libraries from your MATLAB application. These code generation workflows are supported for the coordinate transformation functions (Coordinate System Transformations), the VFH+ obstacle avoidance algorithm, and the Pure Pursuit controller algorithm (robotics.PurePursuit). See Code Generation for more information.

R2015a

Version: 1.0

New Features

Path planning, path following, and map representation algorithms

The Robotics System Toolbox provides algorithms for path planning, path following, and map representations. The support in this release includes classes for Binary Occupancy Grids, Probabilistic Roadmaps (PRM), and a Pure Pursuit controller.

Functions for converting between different rotation and translation representations

Coordinate system transformations are provided as functions for converting between many different representations including quaternions, rotation matrices, homogeneous transformation matrices, and Euler angles. Other functions are available for converting between radians and degrees and for angle calculations. For more information, see Coordinate System Transformations.

Bidirectional communication with live ROS-enabled robots

Communication with ROS using publishers and subscribers is available in MATLAB and Simulink. Many message types are readily supported. Robotics System Toolbox can also access ROS services, the parameter server, and the tf transformation tree in MATLAB.

Interface to Gazebo and other ROS-enabled simulators

ROS-enabled simulators allow prototyping of algorithms and testing systems developed in MATLAB. Connection to a Gazebo simulator is supported with an example interacting with the simulator shown here: Reading Model and Simulation Properties from Gazebo.

Data import from rosbag log files

This release of the Robotics System Toolbox includes the ability to access rosbags, which are logfiles from ROS. You can access whole log files or portions and manipulate the data as desired (see Working with rosbag Logfiles).

ROS node generation from Simulink models (with Embedded Coder)

This release includes ROS node generation using Simulink. You can use Simulink to create models that exchange messages with a ROS network. Using Embedded Coder, you can generate C++ code for standalone ROS nodes from these models.