Interfacing MATLAB and ROS

By Yanliang Zhang (Ph.D.)
MathWorks Robotics Product and Marketing Manager
Yanliang.Zhang@mathworks.com

ICRA 2014, Hong Kong
6/5/2014

MATLAB/Simulink Robotics Portal: www.mathworks.com/robotics

MathWorks | Accelerating the pace of engineering and science

United States | Contact Us | How To Buy | Search MathWorks

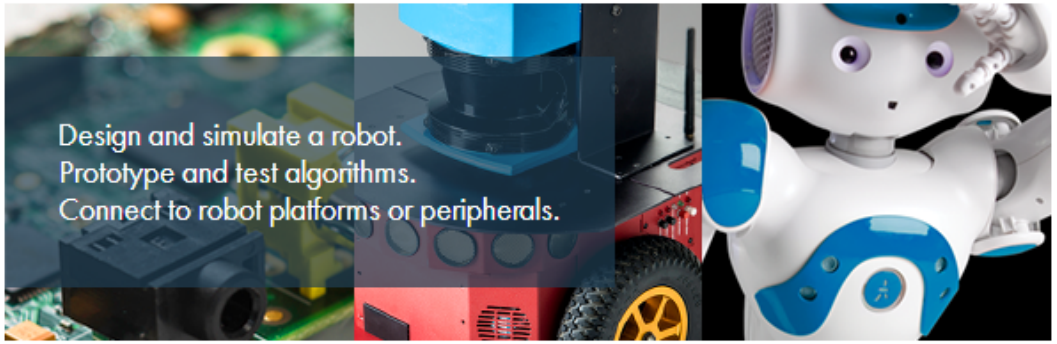
Create Account | Log In

Products & Services | Solutions | Academia | Support | User Community | Events | Company

Solutions > Robotics with MATLAB and Simulink

Robotics with MATLAB and Simulink

Contact sales | Trial software | Share



Three of the most critical questions that robotics engineers and scientists need to answer are:

- How do I design and simulate a robot?
- How do I prototype and test algorithms for my robot?
- How do I connect to my robot platforms and peripherals?

MATLAB® and Simulink® can help answer these questions, and accelerate and streamline the design, prototyping, and verification of robotics applications.

Ask the Expert

Yanliang Zhang
Robotics Industry Manager

» Email Yanliang

Meet MathWorks

May 31-June 7, 2014, Hong Kong, China

2014 IEEE International Conference on Robotics and Automation (ICRA 2014)

Try MATLAB and ROS I/O Now

Read data from ROS nodes into MATLAB for visualization and algorithms prototypting. Learn more and download the MATLAB ROS I/O package now.

Why?

MATLAB/Simulink Integration with Robots

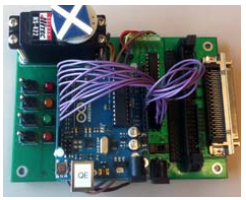
MATLAB/Simulink

MATLAB Non-target API

Simulink Non-target Lib

Simulink Target

ROS



Any Robot Running ROS

What Is ROS?

Robot Operating System (ROS) is a BSD-licensed, non real-time, software framework for robot development, providing operating system-like functionalities like hardware abstraction, device drivers, libraries, visualizers, message-passing, package management.

Existing Packages (Libraries):

There are many ready-to-use packages (using ROS integration and messaging conventions) which contain vetted implementation of common algorithms for each area of robotics:

- Stereo vision
- SLAM
- Control
- Navigation
- Motion Planning
- Manipulation
- Grasping
- Motion understanding
- Mobile robotics
- Perception
- Object Identification
- Segmentation
- Face recognition
- Natural Language
- Gesture recognition
- Motion tracking
- Structure from motion (SFM)
- Egomotion

The possibility of reusing software without having to rewrite it is one of the main drivers of ROS adoption within both universities and commercial companies.

ROS on Robots

Click Here


Fraunhofer IPA Care-O-bot


Videre Erratic


TurtleBot


Aldebaran Nao


Lego NXT


Shadow Hand


Willow Garage PR2


iRobot Roomba

Robotnik Guardian


Merlin miabotPro


AscTec Quadrotor


CoroWare Corobot


Clearpath Robotics Husky


Clearpath Robotics Kingfisher


Festo Didactic Robotino


Gostai Jazz


Neobotix mp-500


Neobotix mpo-500


Neobotix mpo-700


ROS-Industrial

Robotnik Modular Arm

ROS on Robots

Click Here

Robotnik Summit

Cyton-Gamma

Robonaut 2

Adept MobileRobots Pioneer family (P3DX, P3AT, PeopleBot, PowerBot, AmigoBot)

Adept MobileRobots Seekur family (Seekur, Seekur Jr.)

Adept MobileRobots PatrolBot family (PatrolBot, GuiaBot)

Robotnik SummitXL

AMIGO

TUlip

Eddiebot

Allegro Hand SimLab

REEM

ROS Used in Teaching

Pretty much *any* university robotics lab these days is using ROS, and many are also contributing code back to it, including top institutions. Some of the top contributors:

- Stanford
- MIT
- CMU
- Brown University
- CCNY
- UT Austin/ART
- SIUE
- Rice University
- Penn
- TUM
- Uni Freiburg
- WPI
- Cornell
- Georgia Tech
- USC
- Columbia University
- Imperial College (London)
- UT Austin
- Washington University St Louis
- University of Maryland
- University of Colorado at Boulder

How?

Download and Install MATLAB ROS I/O:
www.mathworks.com/ROS

A Simple Example (All Codes inside MATLAB)

- Create (Connect to) a ROS Master

  ```
  >> roscore = rosmatlab.roscore(11311);
  ```

- Create a ROS node

  ```
  >> node = rosmatlab.node('NODE', roscore.RosMasterUri);
  ```

- Create a Publisher

  ```
  >> publisher = rosmatlab.publisher('TOPIC', 'std_msgs/String', node);
  ```

- Add a Subscriber

  ```
  >> subscriber = rosmatlab.subscriber('TOPIC', 'std_msgs/String', 1,
  node);
  ```

- Bind Functions to Subscriber

  ```
  >> subscriber.setOnNewMessageListeners({@function1, @function2});
  ```

A Simple Example (All Codes inside MATLAB)

Now you can use different MATLAB toolboxes

```
function function1(message)
    disp(char(message.getData()));
end

function function2(message)
    disp(sprintf('Message received: %s',datestr(now)));
end
```

A Simple Example (All Codes inside MATLAB)

- Create a new message for use by the publisher

  ```
  >> msg = rosmatlab.message('std_msgs/String', node);
  ```

- Set the data field of the std_msgs/String message

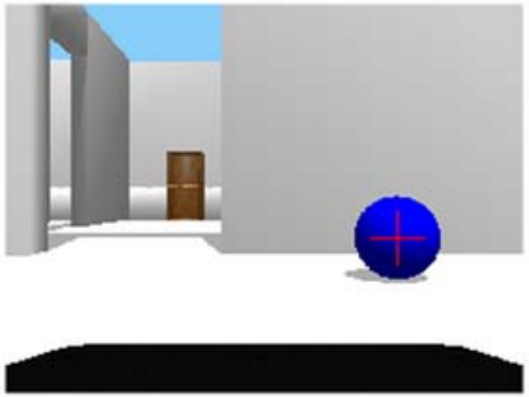
  ```
  >> msg.setData(sprintf('Message created: %s', datestr(now)));
  ```

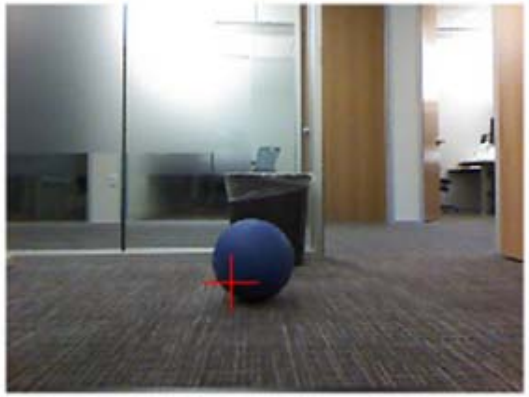
- Publish the Message

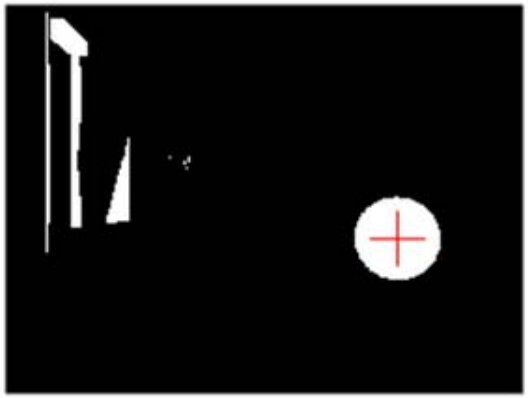
  ```
  >> publisher.publish(msg);
  ```

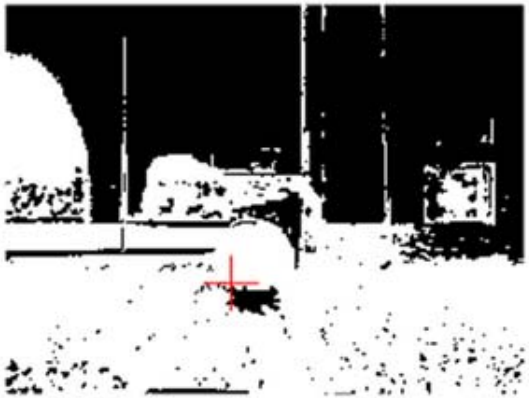
```
Message created: dd-mmm-yyyy HH:MM:SS

Message received: dd-mmm-yyyy HH:MM:SS
```

Demonstration:
Ball Tracking on Gazebo and Real TurtleBot









Husky + ROS I/O Code Example

- Step by Step instructions can be found on Clearpath's *BLOG*: http://www.clearpathrobotics.com/blog/ros-for-matlab/
- Or Google: ***HUSKY MATLAB***

Editor - C:\Users\ibaranov\Documents\MATLAB\Husky_CMD_VEL.m

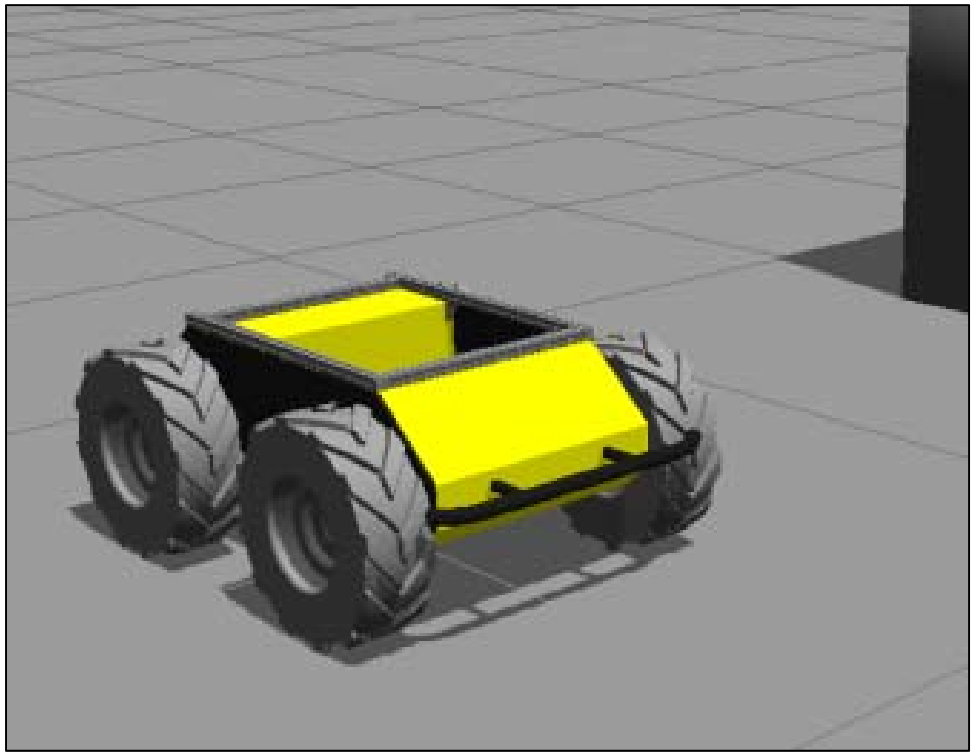
EDITOR | PUBLISH | VIEW

New | Open | Save | Find Files | Compare | Print | Insert | Comment | Indent | Go To | Find | Breakpoints | Run

FILE | EDIT | NAVIGATE | BREAKPOINTS

Husky_CMD_VEL.m

```
1    % Ilia Baranov for Clearpath Robotics
2
3    function Husky_CMD_VEL
4        % Connect to the IP address of the Husky
5        node = rosmatlab.node('NODE', '10.25.0.228', 11311);
6
7        % Create Subscriber to ROS TWIST Message, called husky/cmd_vel
8        % buffer size = 1
9        subscriber = rosmatlab.subscriber('husky/cmd_vel', 'geometry_m
10
11       % When message is received, call function to show value
12       subscriber.setOnNewMessageListeners({@display_vel});
13
14       % Create a publisher for the ROS TWIST message, called husky/c
15       publisher = rosmatlab.publisher('husky/cmd_vel', 'geometry_msg
16
17       % Set Linear velocity components of command
18       msgLin = rosmatlab.message('geometry_msgs/Vector3', node);
19       msgLin.setX(0);
20       msgLin.setY(0);
21       msgLin.setZ(0);
```



Join us at IROS 2014, Chicago, Sept. 14–18, 2014

WORKSHOPS & TUTORIALS

Morning sessions are 8:30 AM until 12:00 PM, with a coffee break at 10:00 AM. Afternoon sessions are 1:30 PM until 5:00 PM, with a coffee break at 3:00 PM.

Sunday, September 14, Half-day Workshops and Tutorials

Title	Organizers
An Open-source Recipe for Teaching (and Learning) Robotics with a Simulator: Setup a Laptop in 5 Minutes, Write a Control, Navigation, Vision or Manipulation Program in 100 Lines of Code *(Morning Session)*	Renaud Detry*, Peter Corke, Marc Andreas Freese
Taxonomies of Interconnected Systems: Topology in Distributed Robotics *(Morning Session)*	Ryan Williams*, Andrea Gasparri, Gaurav Sukhatme
How to Use MATLAB-ROS Interface to Prototype Robotics Algorithms for ROS-powered Robots *(Afternoon Session)*	Yanliang Zhang