Why MATLAB/Simulink?
Agenda

- MATLAB – ROS Workflow
- Simulink – ROS Workflow
- Deploying Standalone ROS Nodes from Simulink
MATLAB/Simulink – ROS Workflows

Desktop Prototyping

Deployment

Simulators
Hardware

ROS

Controls
Perception
Planning and decision making

Standalone ROS Nodes
Simple Loop Template

```matlab
rosinit('ipAddress');
mySub = rossubscriber('/sub_topic');
[myPub, pubMsg] = rospublisher('pub_topic');

recvMsg = mysub.LatestMessage;

ctrlOut = myAlgorithm(recvMsg);

pubMsg.FieldName = ctrlOut;
send(myPub, pubMsg);

currentTime = toc;
plot(currentTime, ctrlOut);
```
Simple Loop Template

```matlab
rosinit('ipAddress');
mySub = rossubscriber('/sub_topic');
[myPub, pubMsg] = rospublisher('pub_topic');
currentTime = 0;

tic
while(currentTime < 10)
    recvMsg = mySub.LatestMessage;

    ctrlOut = myAlgorithm(recvMsg);

    pubMsg.FieldName = ctrlOut;
    send(myPub, pubMsg);

    currentTime = toc;
    plot(currentTime, ctrlOut);
end
```
Simple Loop Template

SETUP

SENSE

PROCESS

CONTROL

VISUALIZE
Simple Loop Template

```plaintext
%CPU wall clock time
r = robotics.Rate(5);

%Global ROS node time
r = rosrate(5);

while(r.TotalElapsedTime < 10)
    % Algorithm code goes here
    waitfor(r)
end
```
Advance MATLAB Scheduling

- Asynchronous Execution
  - ROS subscriber callbacks

- Schedule one or more task in the background
  - MATLAB Timers

**NOTE:** Multiple callbacks and timers in the same MATLAB session share a single thread!
Agenda

• MATLAB – ROS Workflow
• Simulink – ROS Workflow
• Deploying Standalone ROS Nodes from Simulink
Simulink – ROS Workflows

- Simulators
- Hardware
- ROS
- Desktop Prototyping
- Controls
- Perception
- Planning and decision making
- Standalone ROS Nodes
- Deployment
Simulink - Templates

When model loads or runs

SETUP

SENSE

PROCESS

CONTROL

VISUALIZE
Simulink Template: Simulation

Go to File > Model Properties > Model Properties, select the Callbacks tab, and go to PreLoadFcn.

Here, you can set up MATLAB code to run when loading the model, such as initializing ROS and loading parameters.

Copyright 2017 The MathWorks, Inc.
Simulink - Templates

SETUP

SENSE

PROCESS

CONTROL

VISUALIZE

ROS industrial consortium asia pacific
Simulink Template: Multiple Inputs and Subscribe Rates

Go to File > Model Properties > Model Properties. Select the Callbacks tab, and go to PreLoadFcn.
Here, you can set up MATLAB code to run when loading the model, such as initializing ROS and loading parameters.

In this template, the two Subscribe blocks are linked to MATLAB variables representing different sample times. Sample time display (Display > Sample Time > All) is enabled to help you view the rates in the model.

Copyright 2017 The MathWorks, Inc.
Modeling Languages

• MATLAB code (MATLAB Function block)
  – Good for algorithms better expressed as code
  – Reuse existing MATLAB code

• Simulink blocks
  – Good for algorithms better expressed as block diagrams (control systems, signal flow, etc.)

• Chart block (Stateflow)
  – Good for state machines, scheduling, decision flowcharts, etc.
Key Takeaways

• Simulink directly supports desktop prototyping and algorithm deployment
• Can create multirate algorithms
• Leverage MATLAB, Simulink and Stateflow modeling languages
• Split models into modular components
Clearpath Robotics Accelerates Algorithm Development for Industrial Robots

Challenge
Shorten development times for laser-based perception, computer vision, fleet management, and control algorithms used in industrial robots

Solution
Use MATLAB to analyze and visualize ROS data, prototype algorithms, and apply the latest advances in robotics research

Results
- Data analysis time cut by up to 50%
- Customer communication improved
- Cutting-edge SDV algorithms quickly incorporated

“ROS is good for robotics research and development, but not for data analysis. MATLAB, on the other hand, is not only a data analysis tool, it’s a data visualization and hardware interface tool as well, so it’s an excellent complement to ROS in many ways.”
- Ilia Baranov, Clearpath Robotics

Link to user story
Voyage develops longitudinal controls for self-driving taxis

Challenge
Develop a controller for a self-driving car to follow a target velocity and maintain a safe distance from obstacles

Solution
Use Simulink to design a longitudinal model predictive controller and tuned parameters based on experimental data imported into MATLAB using Robotics System Toolbox. Deploy the controller as a ROS node using Robotics System Toolbox. Generate source code using MATLAB Coder into a Docker Container.

Results
- Development speed tripled
- Easy integration with open-source software
- Simulink algorithms delivered as production software

“We were searching for a prototyping solution that was fast for development and robust for production. We decided to go with Simulink for controller development and code generation, while using MATLAB to automate development tasks.”
- Alan Mond, Voyage

Link to user story
Nara Institute of Science and Technology Researchers Develop Tactile Object Recognition Algorithms for a Robotic Hand

Challenge
Enable a robotic hand to identify objects based on tactile sensor input

Solution
Use MATLAB to develop machine learning and object recognition algorithms and use Robotics System Toolbox to establish connectivity between the algorithms and the ROS-enabled robot

Results
- Hundreds of manual steps eliminated
- Opportunities to try new algorithm ideas increased
- Expertise shared with other students and researchers

“With Robotics System Toolbox, we seamlessly connected to and controlled our robot directly from the algorithms we had developed in MATLAB, enabling us to minimize development time. We used the time we gained to further our research into new tactile object recognition algorithms.”

Takamitsu Matsubara
Nara Institute of Science and Technology

Link to user story
Agenda

• MATLAB – ROS Workflow
• Simulink – ROS Workflow
• Deploying Standalone ROS Nodes from Simulink
Deploying Algorithms to ROS

- Algorithm deployment workflows
- Generating and using standalone ROS nodes
- Object tracking example
- Key takeaways
Algorithm Deployment Workflows -> Coder Products

MATLAB Function

Code Generation

Standalone C/C++ code

Manual Scheduling

Standalone ROS nodes

Simulink model

*MATLAB Coder, Simulink Coder, Embedded Coder
Algorithm Deployment Workflows -> Coder Products + RST

MATLAB Function → Simulink model → Code Generation → Standalone ROS nodes
Automatically Generating ROS Nodes

Simulink model → Generate Code → Transfer to Target computer, Extract to Catkin Worksapce → Compile (catkin_make) → Executable ROS nodes

- Simulink model
- Generate Code
- Generated C++ code (archived)
- Transfer to Target computer, Extract to Catkin Worksapce
- Generated C/C++ code (extracted)
- Compile (catkin_make)
- Executable ROS nodes

Run Standalone ROS node
Automatically Generating ROS Nodes
Using the Generated ROS Node

- **ROS device object**
  - Start/stop existing nodes from MATLAB

- **External mode**
  - Run generated code while interacting with Simulink block diagram

- Use ROS to communicate with generated node at runtime
Key Takeaways

- Can automatically generate, transfer, build and run ROS nodes from Simulink
- Can generate standalone C/C++ code
- Once a ROS node is generated, you can:
  - Start and Stop node from MATLAB
  - Use External mode to access data and tune parameters from Simulink
  - Use ROS to communicate with node
What’s next?! 

- Run on range of systems: embedded to workstation
- For use in real-time systems
- For safety- and mission-critical applications and productions