Agenda

• Introduction
• Advanced Robotics Systems
• Robotics Development Workflow with MATLAB and Simulink
• Takeaways
Car as an Advanced Robotics System

Planning

Localization

Global Map

Motion control

Obstacle avoidance

SENSE

LIDAR
Camera
RADAR
GPS/IMU

PLAN

Motion Controllers

CONTROL

Actuator ECUs
Steering Actuator
GAS/Brake Actuator

PERCEIVE
Collaborative Robot as an Advanced Robotics System

Motion Planning → Optimization

Obstacle Avoidance → Object Detection

World

Torque, Camera, RGB-D, Temperature

SENSE

Motion Controllers

Actuator Control, Impedance Control, Safety Guards

PERCEIVE

PLAN

CONTROL

MATLAB EXPO 2016
Architecture - Advanced Robotics System

- Motion Planning
- Optimization
- Obstacle Avoidance
- Object Detection
- World

Middleware
- Data synchronization
- CPU/GPU Cluster, Server, Cloud

Real-time requirements
- Microcontroller, FPGA, Real-time PC
Technology Solutions for Autonomous Systems

- **Control System Tbx**
  - Computer Vision
  - Communications Tbx
- **Simscape Toolboxes**
  - HW Support Packages
  - Robotics System Tbx
  - WLAN System Toolbox
- **Simulink Real-Time**
  - Data Acquisition Tbx
  - Phased Array
  - Stateflow
- **Statistics & Machine Learning**
  - Robotics System Tbx
  - ROS

**MATLAB EXPO 2016**
Success Stories

MATLAB EXPO 2016
Technology Solutions for Autonomous Systems

Control System Tbx

Computer Vision

HW Support Packages

Simscape Toolboxes

Phased Array

Data Acquisition Tbx

Communications Tbx

Robotics System Tbx

WLAN System Toolbox

Stateflow

Simulink Real-Time

Statistics & Machine Learning

Robotics System Tbx

MATLAB EXPO 2016
Robotics System Toolbox

- Interface and Deployment to ROS
- Algorithms for Mobile Robotics
- Algorithms for Manipulators and Humanoids

Environment for prototyping, simulating, and deploying robotics applications
Robotics Applications with Robotics System Toolbox
Workflow Convergence is Needed

- Trace to requirements
- Design/debug algorithm
- System integration

- Testing on physical system
- Manual Coding
- Verification and Validation
Robotics Development Workflow with MATLAB and Simulink

RAPID ITERATIVE PROCESS

Idea

Requirements
- Built-in algorithms and apps
- System-Level Simulation MBD
- Co-simulation

Design and Simulation
- C/C++ automatic code generation
- Processor-in-the-loop (PIL)
- Debug C/C++ with MATLAB Engine

Prototype

Implementation
- Design independent of hardware implementation!
Let’s solve a real problem: Sign Detection System

Clearpath Husky Robot
- ROS Enabled
- Kinect (RGB and Point cloud)
- Hokuyo 2D Lidar

Simulation model in Gazebo

Track and Park
Reduce Speed
Collision Avoidance
System Level Design

ROS as Communication Framework

State Machine

Object Classifier

Obstacle Avoidance
Sign Recognition with Collision Avoidance
Robotics Development Workflow with MATLAB and ROS

RAPID ITERATIVE PROCESS

Idea → Requirements → Design and Simulation → Prototype → Implementation → Product

- Import logged data
- Train a classifier
- Test component
Importing Simulation and Experimental Data

Experimental data or simulation Data

Import ROS Data

Filter your logged field data by topic or time interval

%% Load the file
carData = rosbag('\car_field_test_042016.bag');

%% Select all messages on the scan topic
odomMsg = select(carData, 'Topic','/scan');

%% Get all RGB camera points
imagMsg = select(carData, 'Topic','/camera/rgb/image_raw');

Data ready to design algorithms

Robotics System Toolbox™
Visualize, Analyze, and Process Data: Classifier

Training data

Preprocessing

Feature Extraction

Training

Input

Output

MATLAB EXPO 2016
Visualize, Analyze, and Process Data: Classifier

Training data

Preprocessing

Feature Extraction

Training

Classifier

Input

Output

Computer Vision System Toolbox™

Statistics and Machine Learning Toolbox™

MATLAB EXPO 2016
Design and Test Algorithm

%% Create classifier object
sd = SignDetector();

%% Test algorithm with two stops signs
Iin = imread('gazeboTwoStopsTest.png');

%% Show algorithm result
[Iout,~,~,~,~] = sd.step(Iin);
imshow(Iout);
Design and Test Algorithm
Robotics Development Workflow with MATLAB and ROS

RAPID ITERATIVE PROCESS

- Test algorithm with an external simulator
- Tune your algorithm
- Integrate with other components
MATLAB and Simulink connect to the ROS network

- Multiple master support
- ROS publishers/subscribers
- ROS services
- ROS TF tree
- ROS Parameter server
Co-simulation with ROS

```matlab
%% Connect to ROS
rosinit '192.168.204.144';

%% Create subscribers
imSub = rossubscriber('/camera/rgb/image_raw');
scanSub = rossubscriber('/scan');

%% Create publisher
[velPub, velMsg] = rospublisher('/husky_velocity_controller/cmd_vel');
```
Co-simulation with ROS
Robotics Development Workflow with MATLAB and ROS

RAPID ITERATIVE PROCESS

Idea
Requirements
Design and Simulation
Prototype
Implementation
Product

System level design to target a different middleware or framework
Deployment

- Generate ROS Node with Simulink and Embedded Coder™
- Generate a shared library with MATLAB Coder™
- Create a Stand Alone Executable with MATLAB Compiler™

Determine deployment methods based on application
Implementation
Part1: Object Detection and Position Estimation
Part 2: Trajectory Planning

Optimization-based Trajectory Planning
Part 3: System Integration

System Integration and Control
Logic Design
Advanced Robotics Application Requires Multiple Technologies

MATLAB and Simulink: very powerful tools to design advanced robotics applications
Trajectory Planning with RGB-D Sensor
Robotics Development Workflow with MATLAB and Simulink

Deploy

Visualize

Experiment

Prototype

Simulate

ROS

Middleware
Questions

% Thank you