

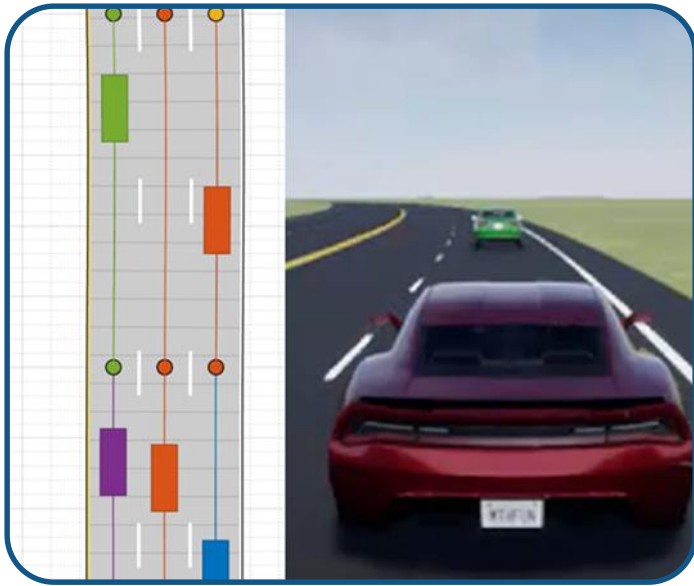
# MATLAB EXPO 2019

Diseño y Simulación de Sistemas de  
Conducción Autónoma con  
MATLAB y Simulink

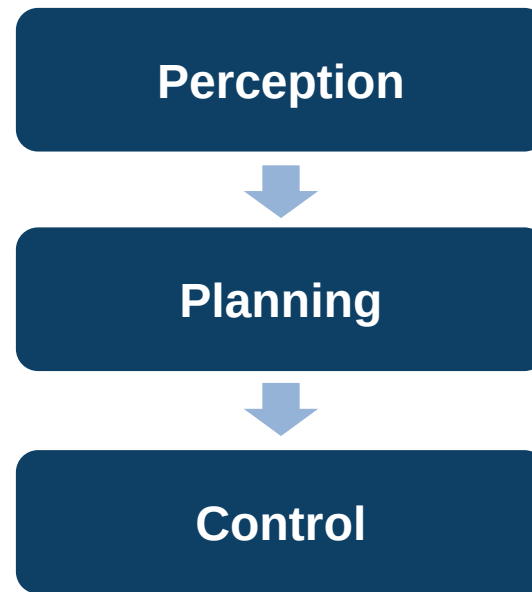
Lucas García



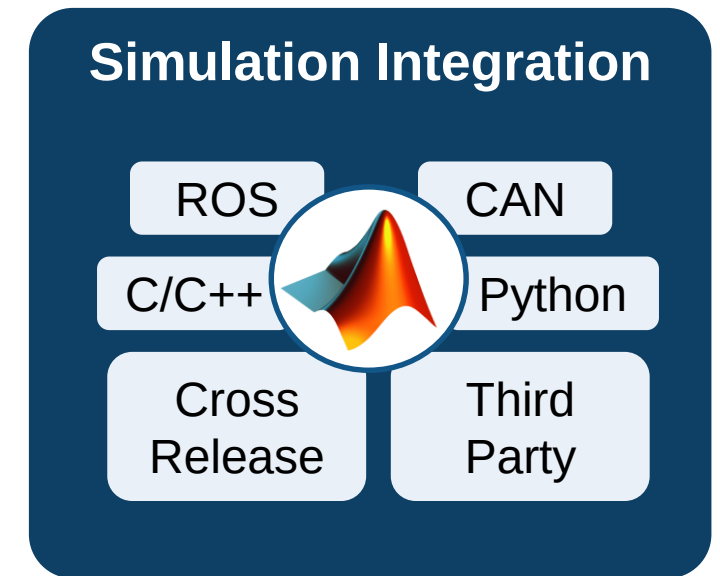
# Some common questions from automated driving engineers



How can I  
**synthesize scenarios**  
to test my designs?

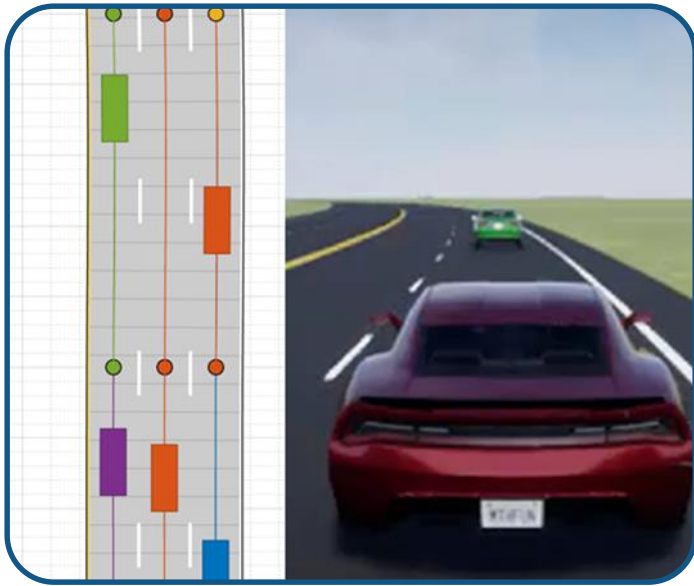


How can I  
**discover and design**  
in multiple domains?

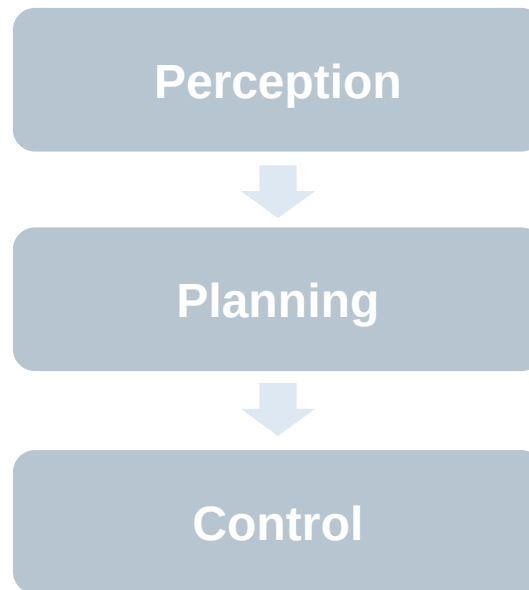


How can I  
**integrate**  
with other environments?

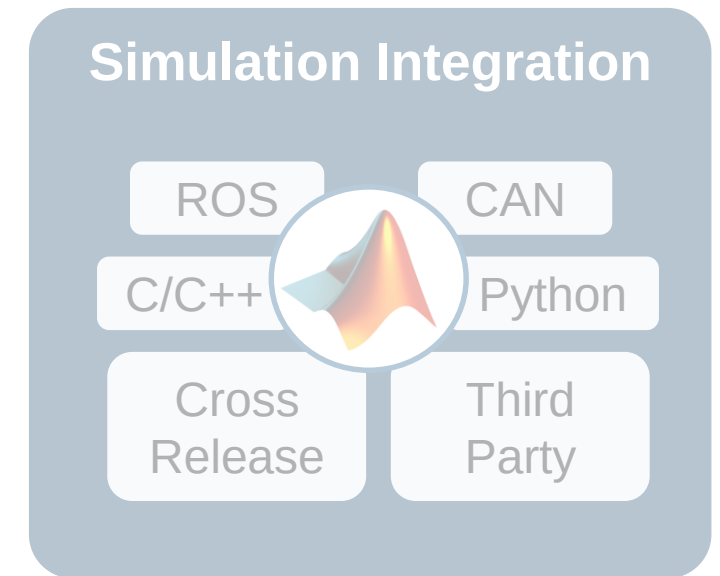
# Some common questions from automated driving engineers



How can I  
**synthesize scenarios**  
to test my designs?



How can I  
**discover and design**  
in multiple domains?



How can I  
**integrate**  
with other environments?

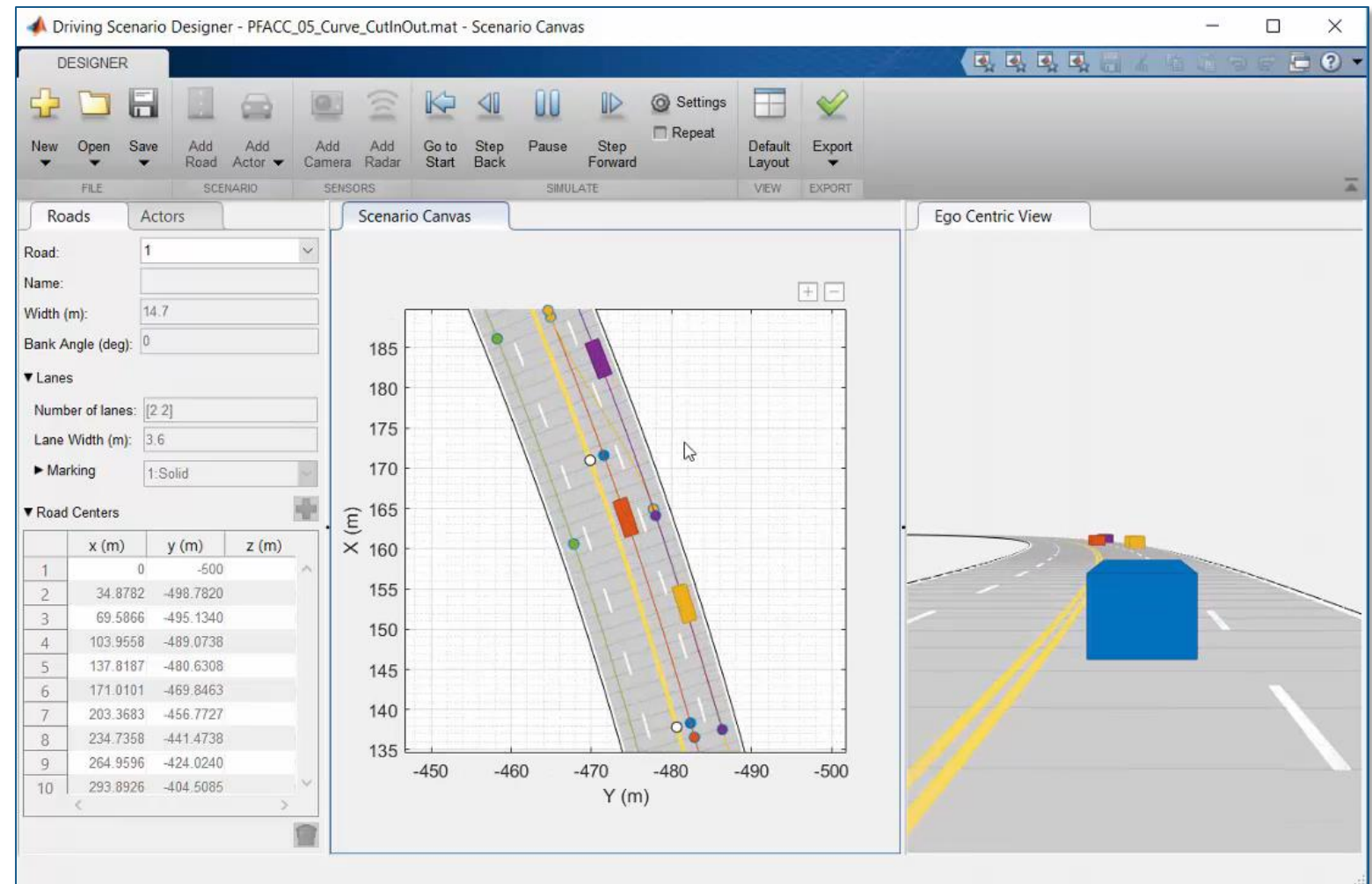
# Graphically author driving scenarios

## Driving Scenario Designer

- Create roads and lane markings
- Add actors and trajectories
- Specify actor size and radar cross-section (RCS)
- Explore pre-built scenarios
- Import OpenDRIVE roads

Automated Driving Toolbox™

R2018a

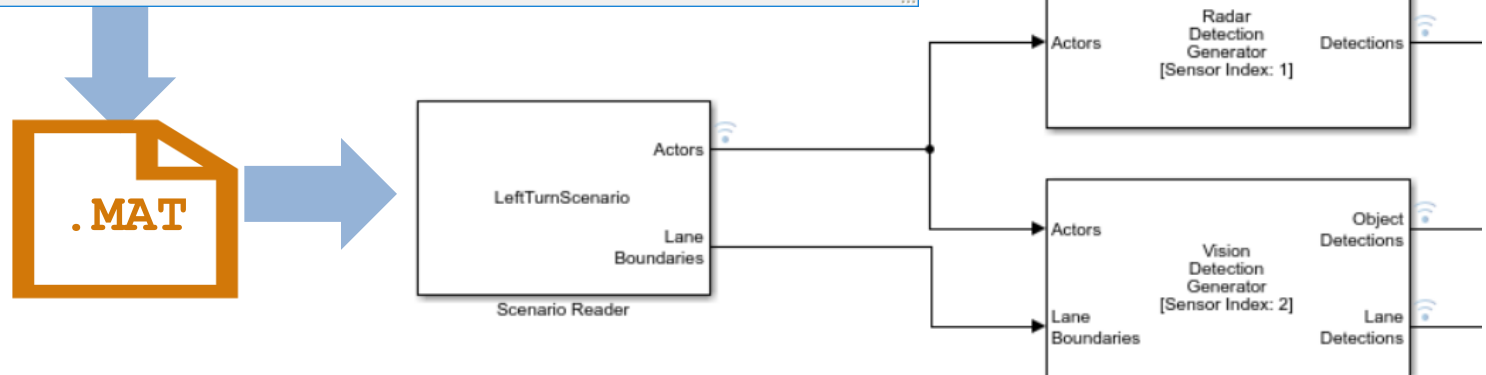
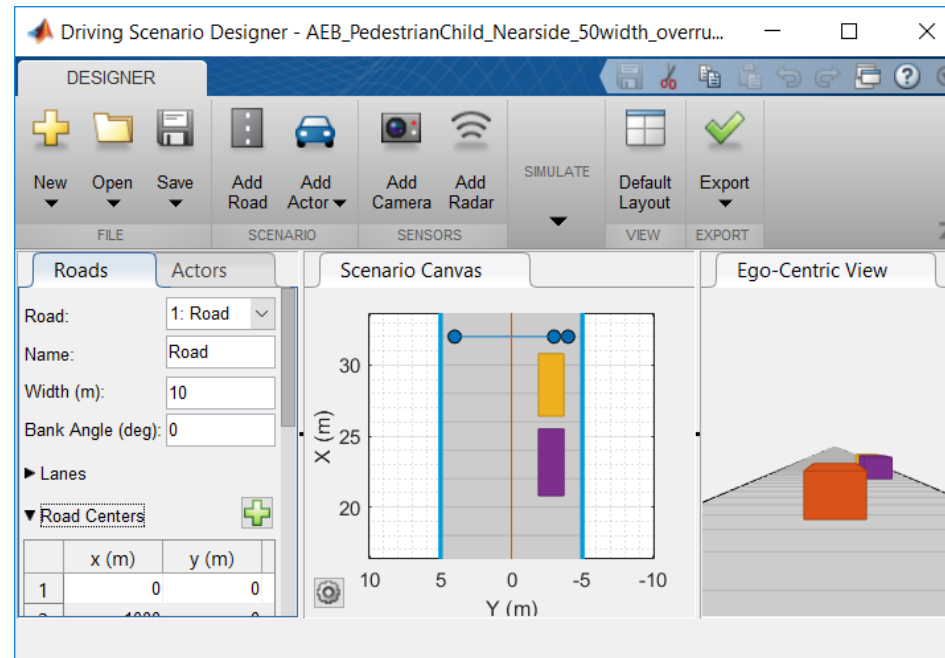


# Integrate driving scenarios into Simulink simulations

## Test Open-Loop ADAS Algorithm Using Driving Scenario

- Edit driving scenario
- Integrate into Simulink
- Add sensor models
- Visualize results
- Pace simulation

Automated Driving Toolbox™  
**R2019a**



- Specify driving scenario
- Design AEB logic
- Integrate sensor fusion
- Simulate system
- Generate C/C++ code
- Test with software in the loop (SIL) simulation

# R2018b



# Automate testing against driving scenarios

## Testing a Lane Following Controller with Simulink Test

- Specify driving scenario

*Simulink Test™*

*Automated Driving Toolbox™*

*Model Predictive Control Toolbox™*

**R2018b**

The screenshot shows the MATLAB Test Manager interface. The 'TESTS' tab is active, displaying a toolbar with icons for New, Open, Save, Cut, Copy, Paste, Delete, Run (highlighted with a red box), Stop, Debug, Parallel, Report, Visualize, Highlight in Model, Import, Export, Preferences, and Help. Below the toolbar, the 'Test Browser' pane shows a tree view of test scenarios under 'LaneFollowingTestScenarios'. The 'ACC\_ISO\_TargetDiscriminationTest' is selected and highlighted with a blue box, with a callout labeled 'Scenarios'. The 'Results and Artifacts' pane on the right shows the configuration for the selected test. It includes sections for DESCRIPTION\*, REQUIREMENTS\*, SYSTEM UNDER TEST\*, TEST HARNESS, SIMULATION SETTINGS OVERRIDES\*, PARAMETER OVERRIDES, CALLBACKS\*, and CLEANUP\*. Callouts point to specific elements: 'Requirements link' points to the 'REQUIREMENTS\*' section; 'Simulink Model' points to the 'Model' field in the 'SYSTEM UNDER TEST\*' section; 'Define scenario ID and data initialization' points to the 'PRE-LOAD' section; and 'Plot the results' points to the 'CLEANUP\*' section.

**Requirements link**

**Simulink Model**

**Scenarios**

**Define scenario ID and data initialization**

**Plot the results**

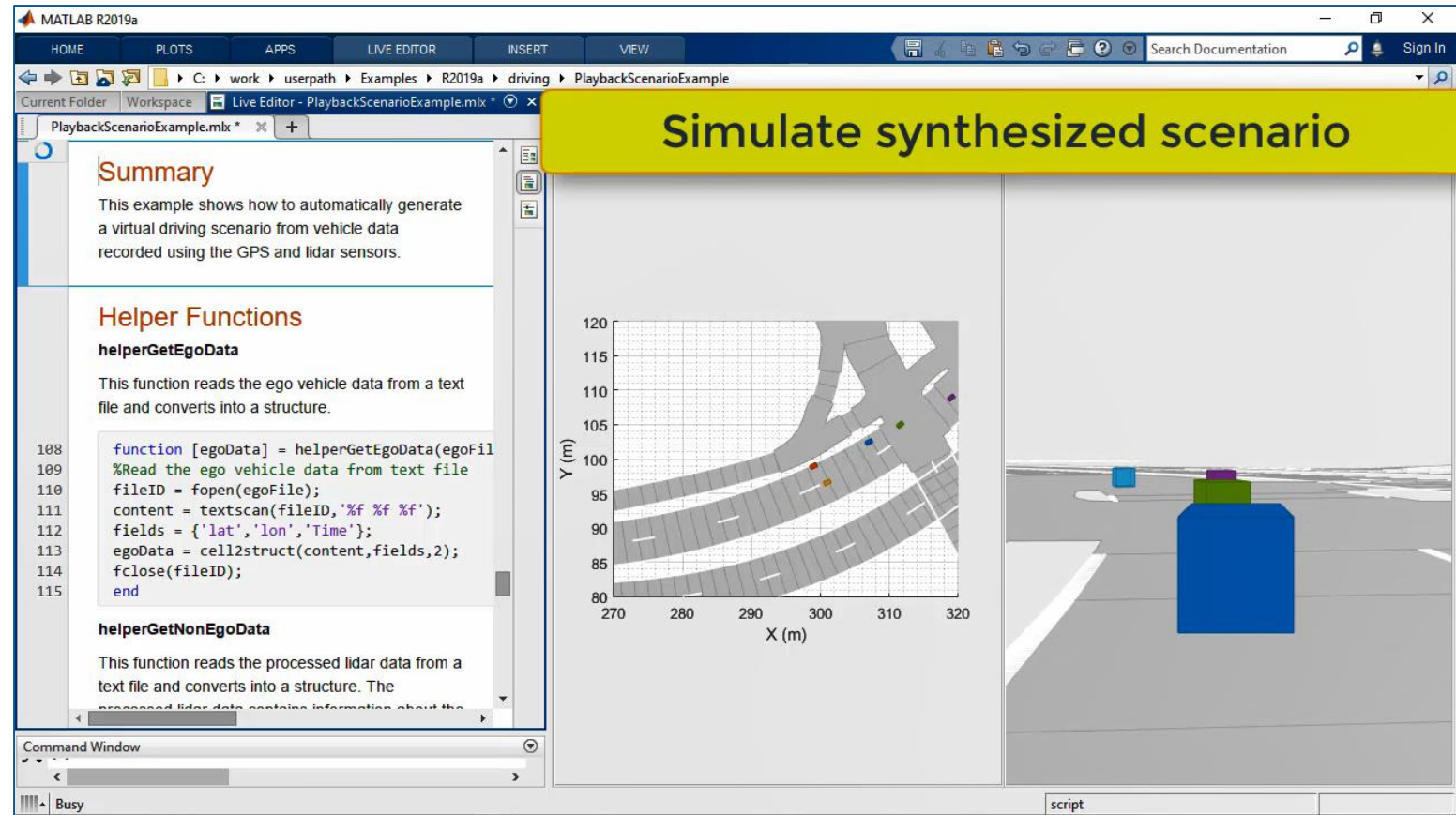
# Synthesize driving scenarios from recorded data

## Scenario Generation from Recorded Vehicle Data

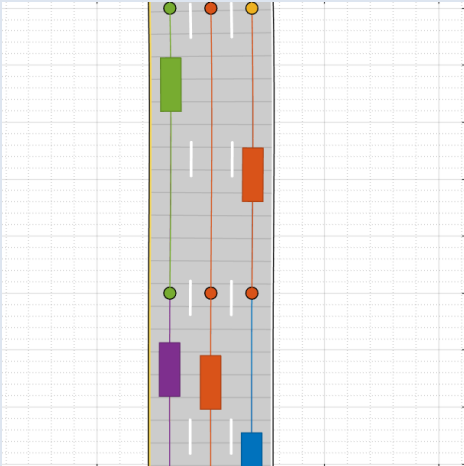
- Visualize video
- Import OpenDRIVE roads
- Import GPS
- Import object lists

*Automated Driving Toolbox™*

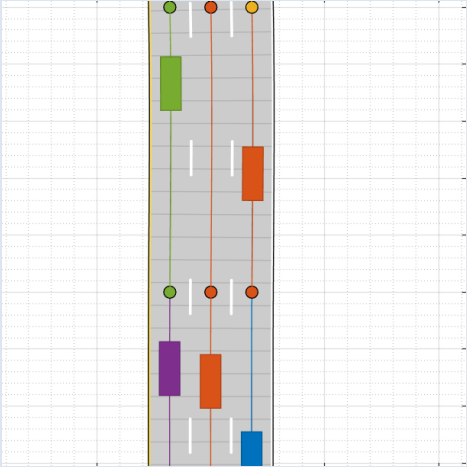
**R2019a**



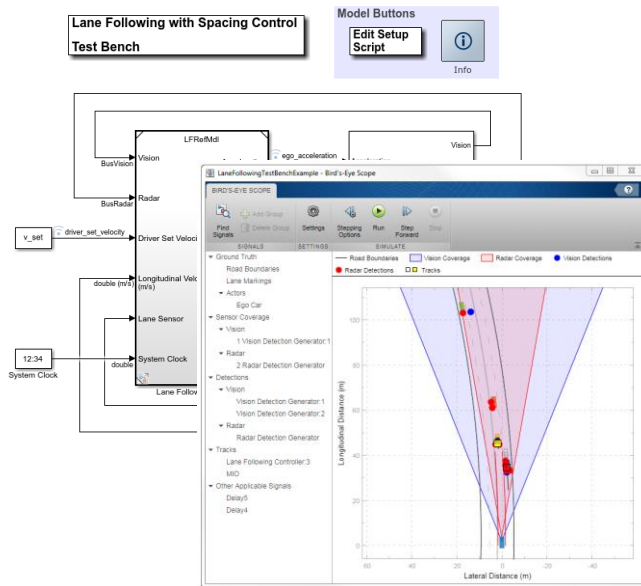
# How can I design with virtual scenarios?

|           |                                                                                                                      |
|-----------|----------------------------------------------------------------------------------------------------------------------|
| Scenes    | <b>Driving Scenarios (cuboid)</b>  |
| Testing   | Controls<br>Controls + sensor fusion                                                                                 |
| Authoring | Driving Scenario Designer App<br>drivingScenario programmatic API                                                    |
| Sensing   | Probabilistic radar detections<br>Probabilistic vision detections<br>Probabilistic lane detections                   |

# How can I design with virtual scenarios?

| Scenes    | <b>Driving Scenarios (cuboid)</b>  | <b>Unreal Engine</b>  |
|-----------|----------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| Testing   | Controls<br>Controls + sensor fusion                                                                                 | Controls<br>Controls + vision                                                                            |
| Authoring | Driving Scenario Designer App<br>drivingScenario programmatic API                                                    | Unreal Editor                                                                                            |
| Sensing   | Probabilistic radar detections<br>Probabilistic vision detections<br>Probabilistic lane detections                   | Ideal camera (viewer)                                                                                    |

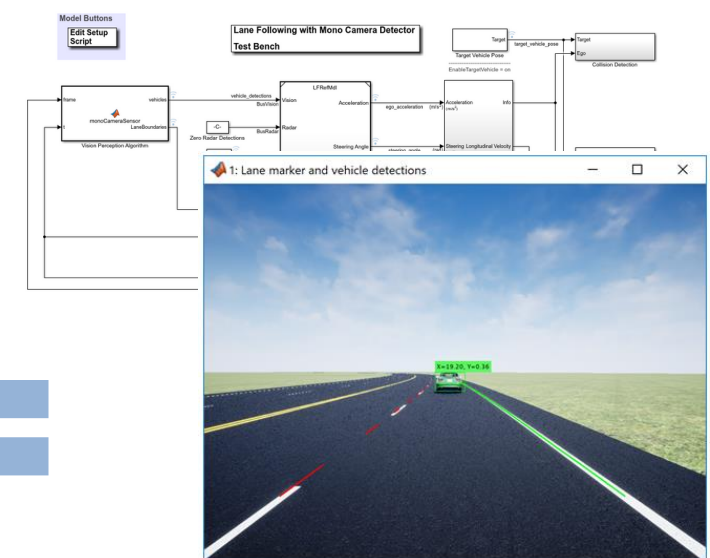
# Simulate controls and perception systems



```

40
41 classdef helperMonoSensor < handle
42
43 properties
44     % Sensitivity for the lane segmentation
45     LaneSegmentationSensitivity = 0.25;
46
47
48
49
50
51

```



## Lane Following Control with Sensor Fusion

Model Predictive Control Toolbox™  
 Automated Driving Toolbox™  
 Embedded Coder®

**R2018b**

MATLAB EXPO 2019

## Visual Perception Using Monocular Camera

Automated Driving Toolbox™

**R2017a**

## Lane-Following Control with Monocular Camera Perception

Model Predictive Control Toolbox™  
 Automated Driving Toolbox™  
 Vehicle Dynamics Blockset™

**R2018b**

# Simulate lane controls with vision based perception

## Lane-Following Control with Monocular Camera Perception

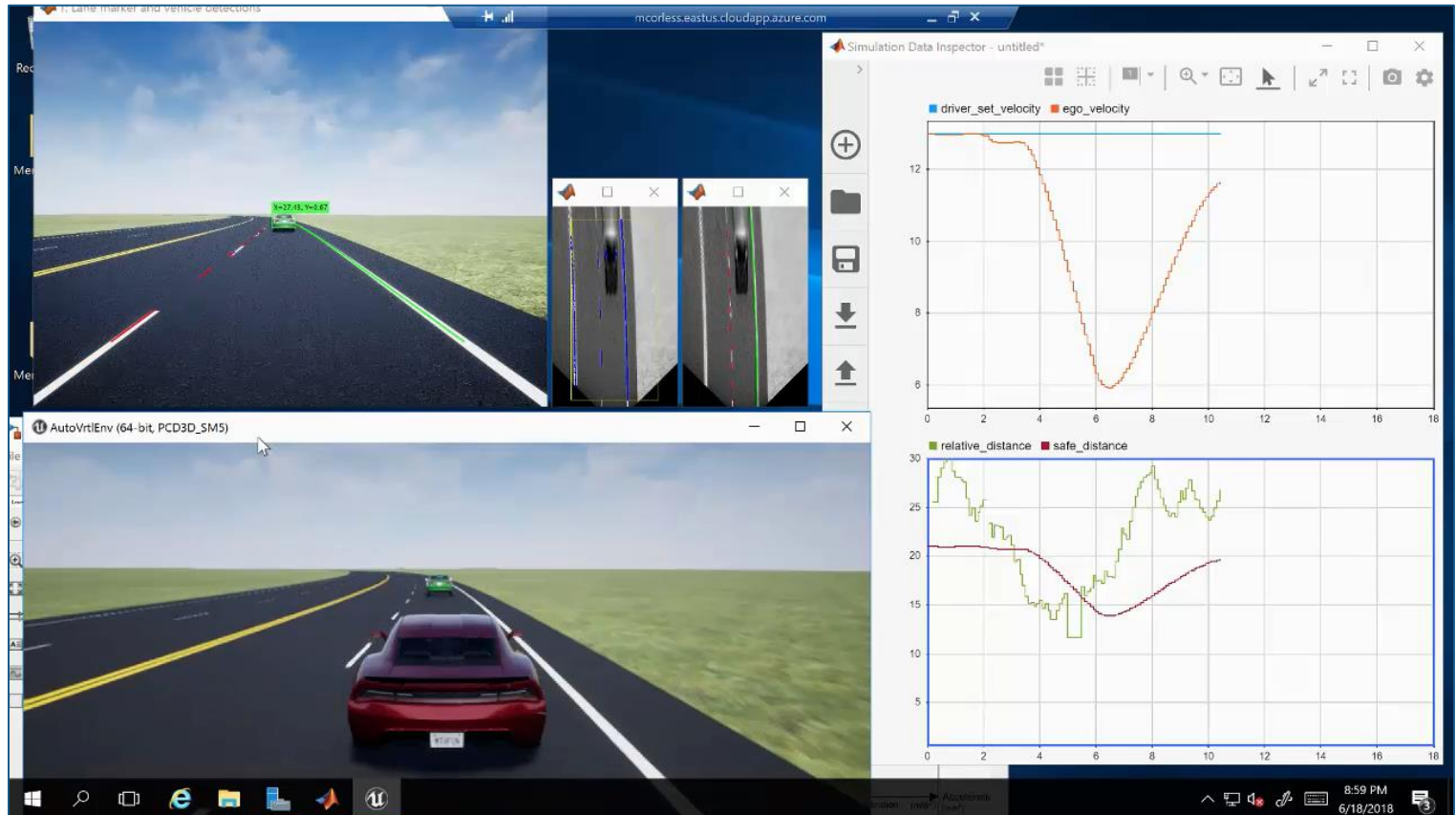
- Integrate Simulink controller
  - Lane follower
  - Spacing control
- Integrate MATLAB perception
  - Lane boundary detector
  - Vehicle detector
- Synthesize ideal camera image from Unreal Engine

*Model Predictive Control Toolbox™*

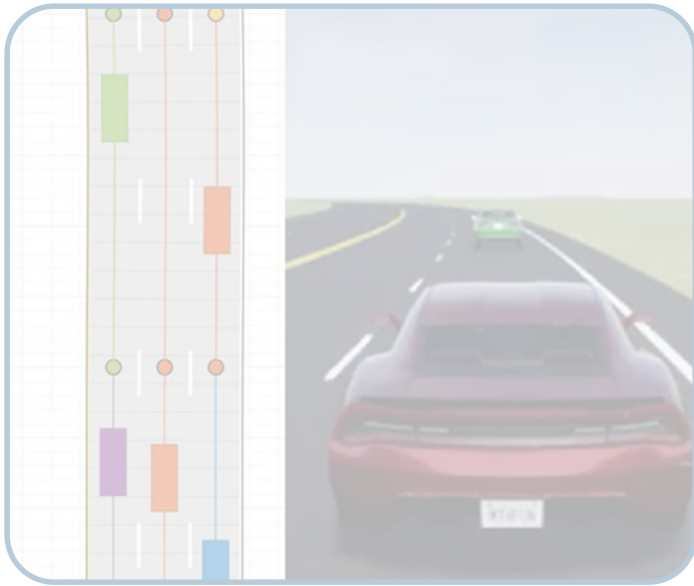
*Automated Driving Toolbox™*

*Vehicle Dynamics Blockset™*

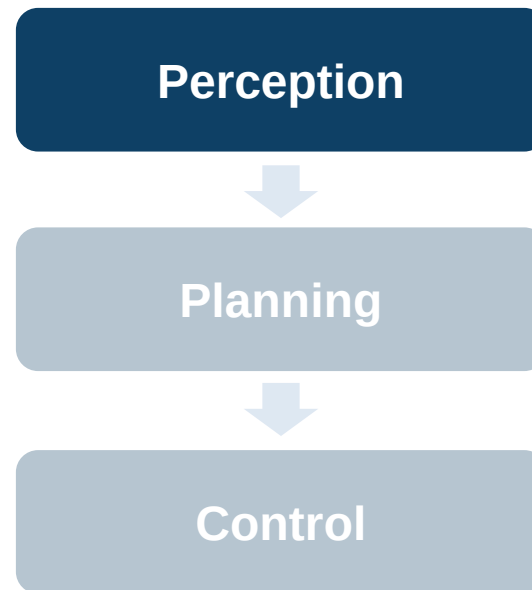
**R2018b**



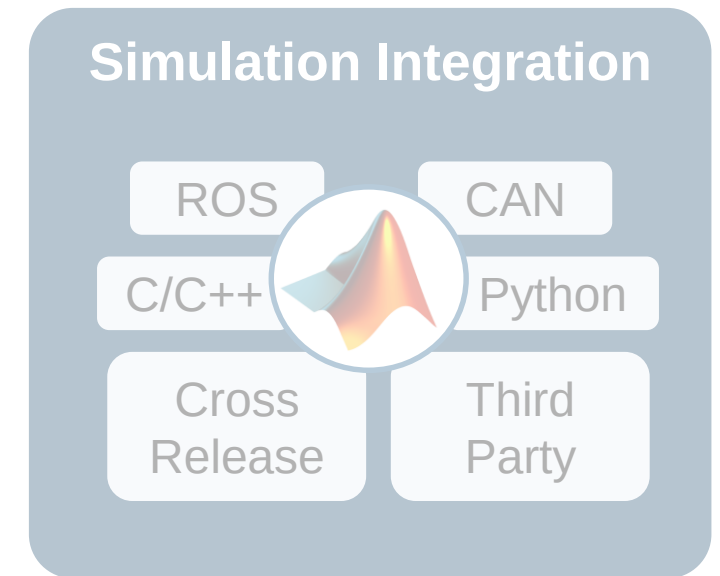
# Some common questions from automated driving engineers



How can I  
**synthesize scenarios**  
to test my designs?



How can I  
**discover and design**  
in multiple domains?



How can I  
**integrate**  
with other environments?

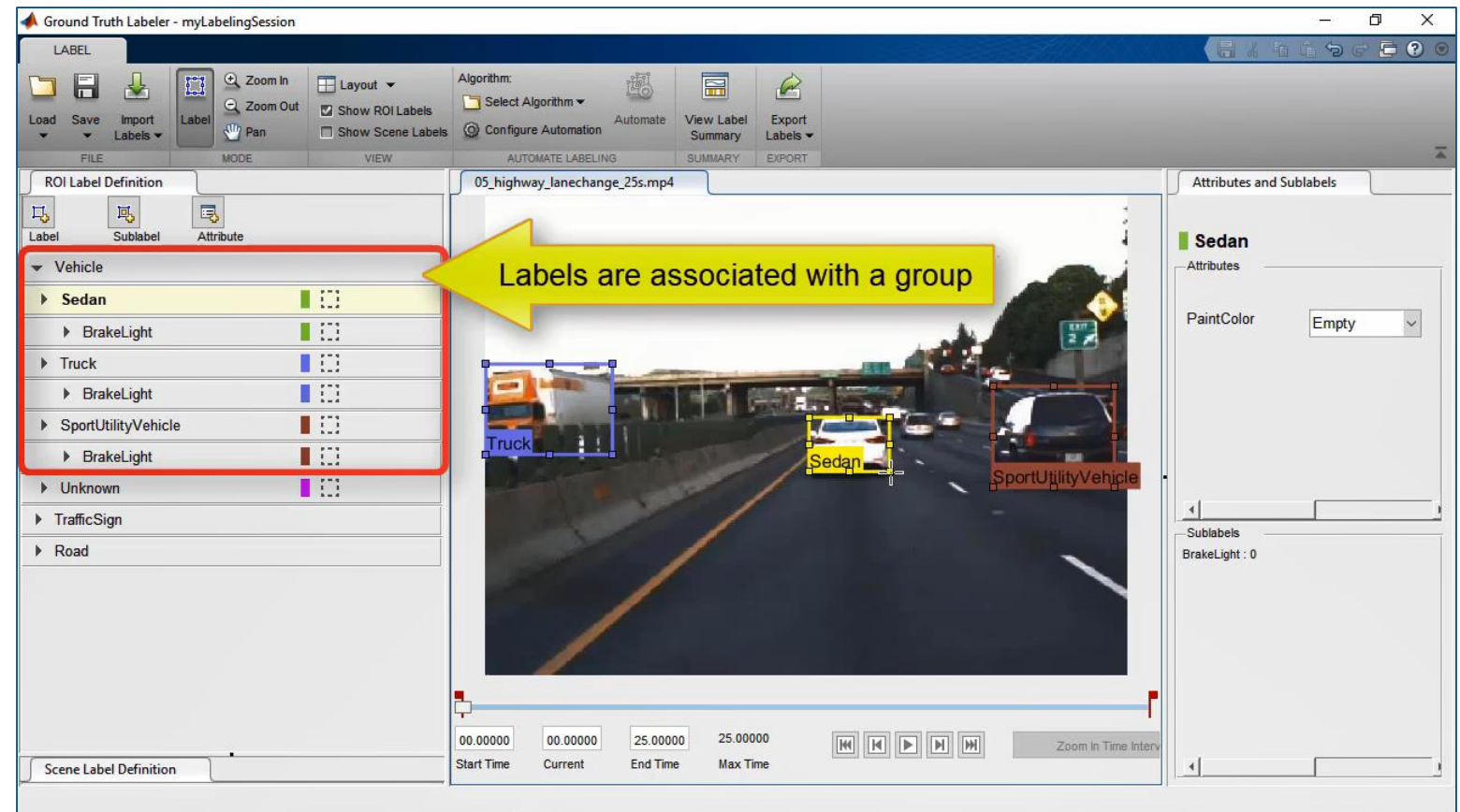
# Create region of interest labels and groups

## Get Started with the Ground Truth Labeler

- Label rectangles
- Label lane markings
- Label pixels
- Label scenes
- Create label groups
- Create sublabels
- Add label attributes

Automated Driving Toolbox™

Updated **R2019a**



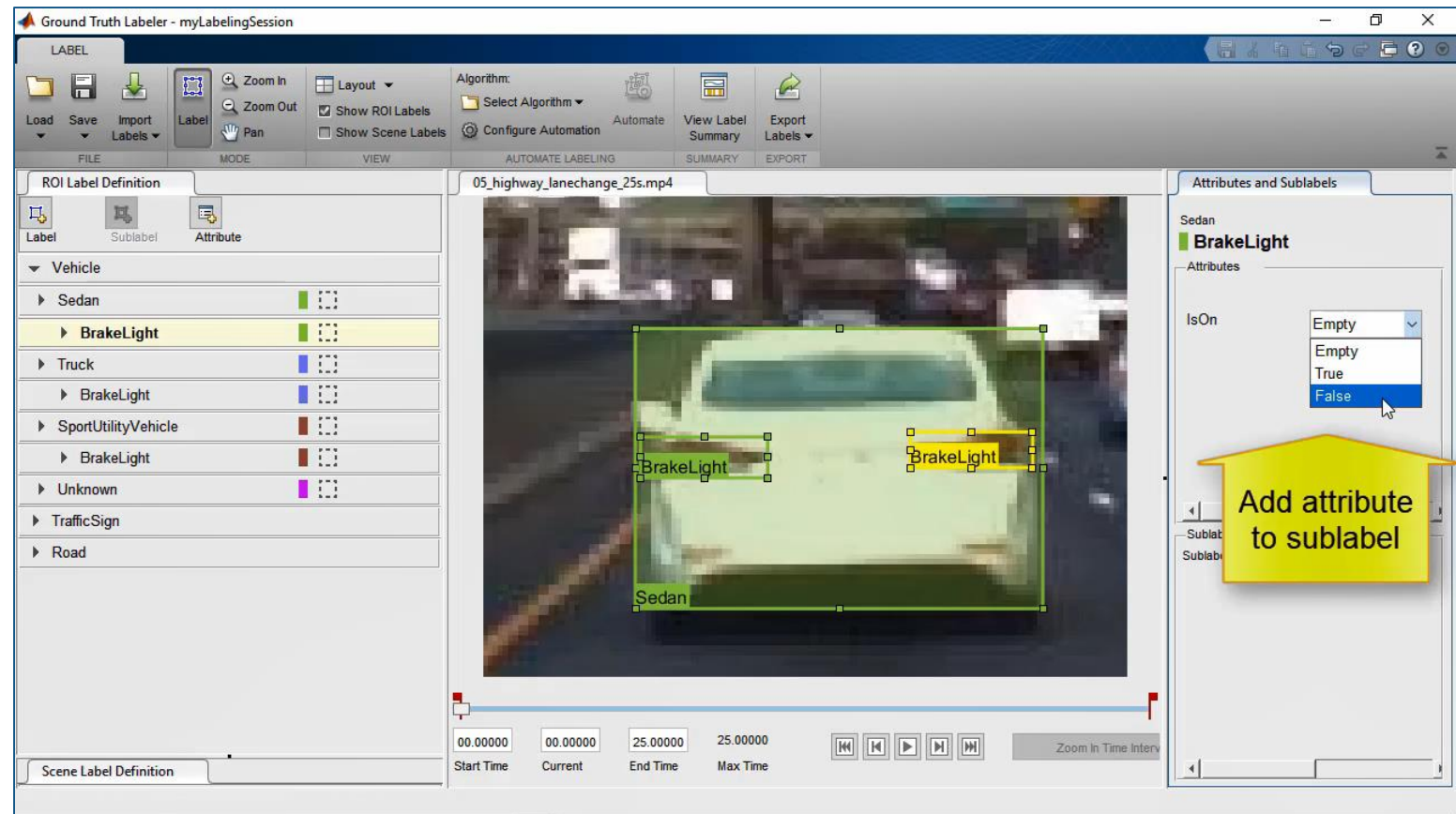
# Create sublabels and add attributes

## Get Started with the Ground Truth Labeler

- Label rectangles
- Label lane markings
- Label pixels
- Label scenes
- Create label groups
- Create sublabels
- Add label attributes

Automated Driving Toolbox™

Updated **R2019a**



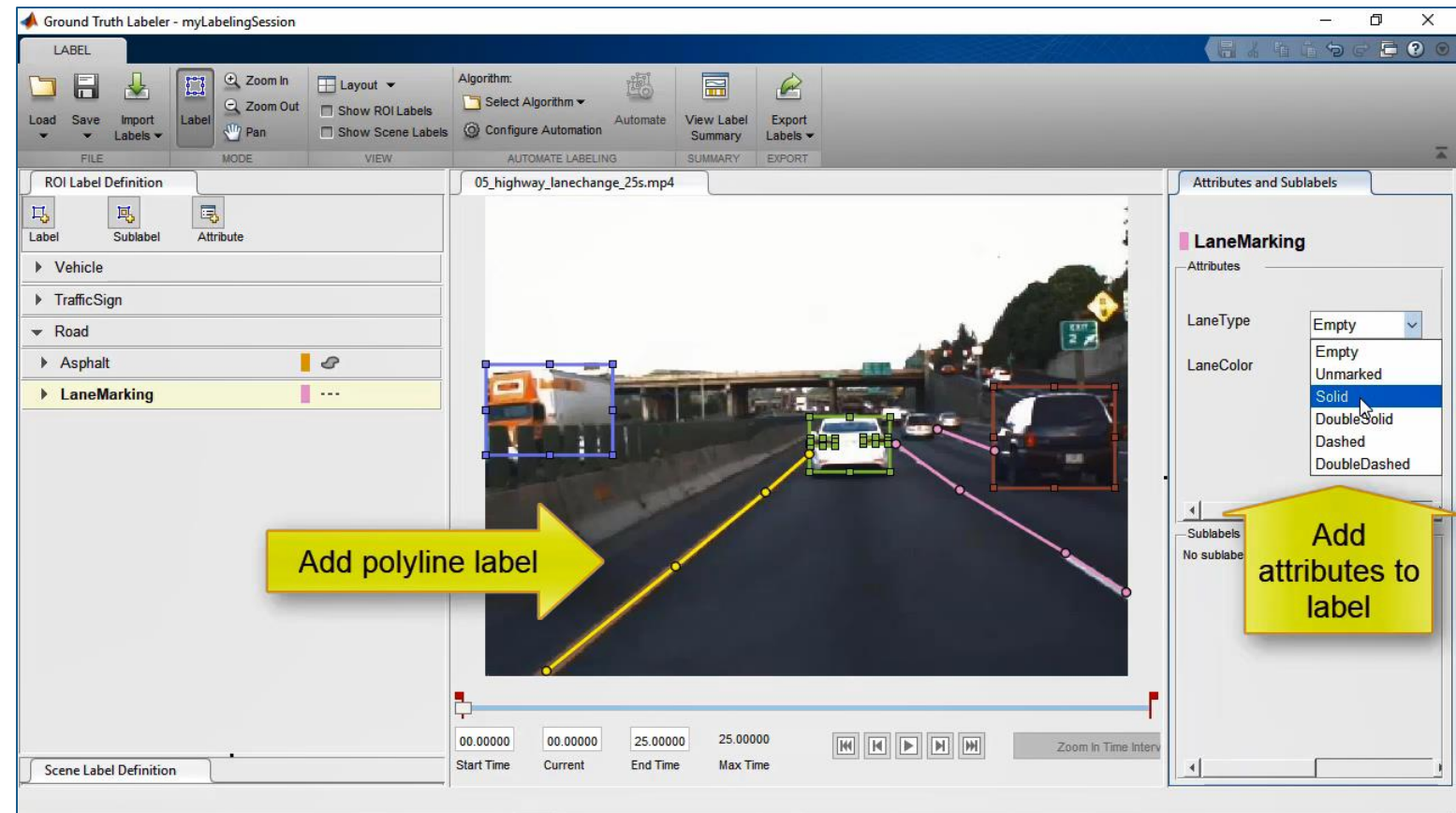
# Create polyline labels and add attributes

## Get Started with the Ground Truth Labeler

- Label rectangles
- Label lane markings
- Label pixels
- Label scenes
- Create label groups
- Create sublabels
- Add label attributes

Automated Driving Toolbox™

Updated **R2019a**



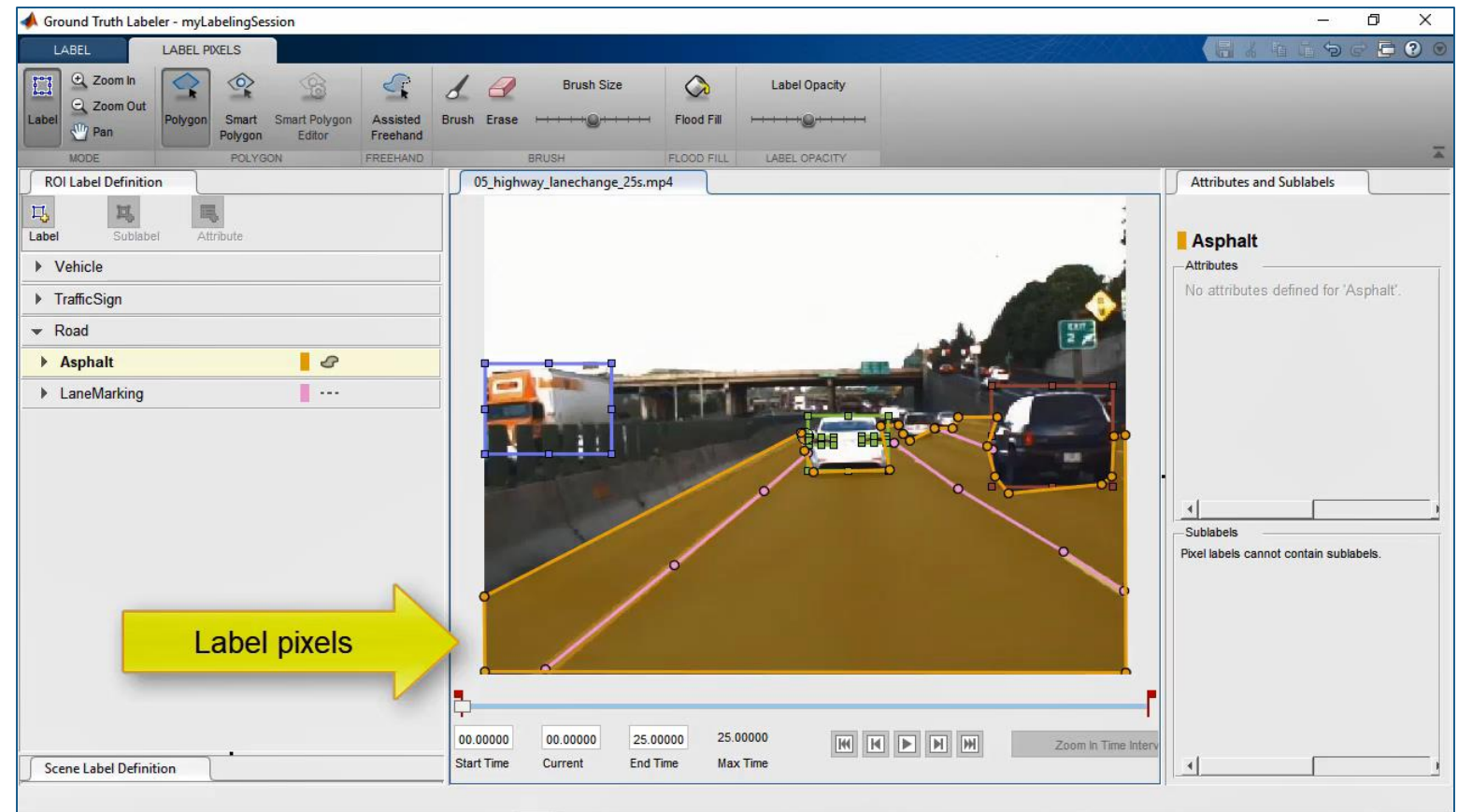
# Create pixel labels

## Get Started with the Ground Truth Labeler

- Label rectangles
- Label lane markings
- Label pixels
- Label scenes
- Create label groups
- Create sublabels
- Add label attributes

Automated Driving Toolbox™

Updated **R2019a**



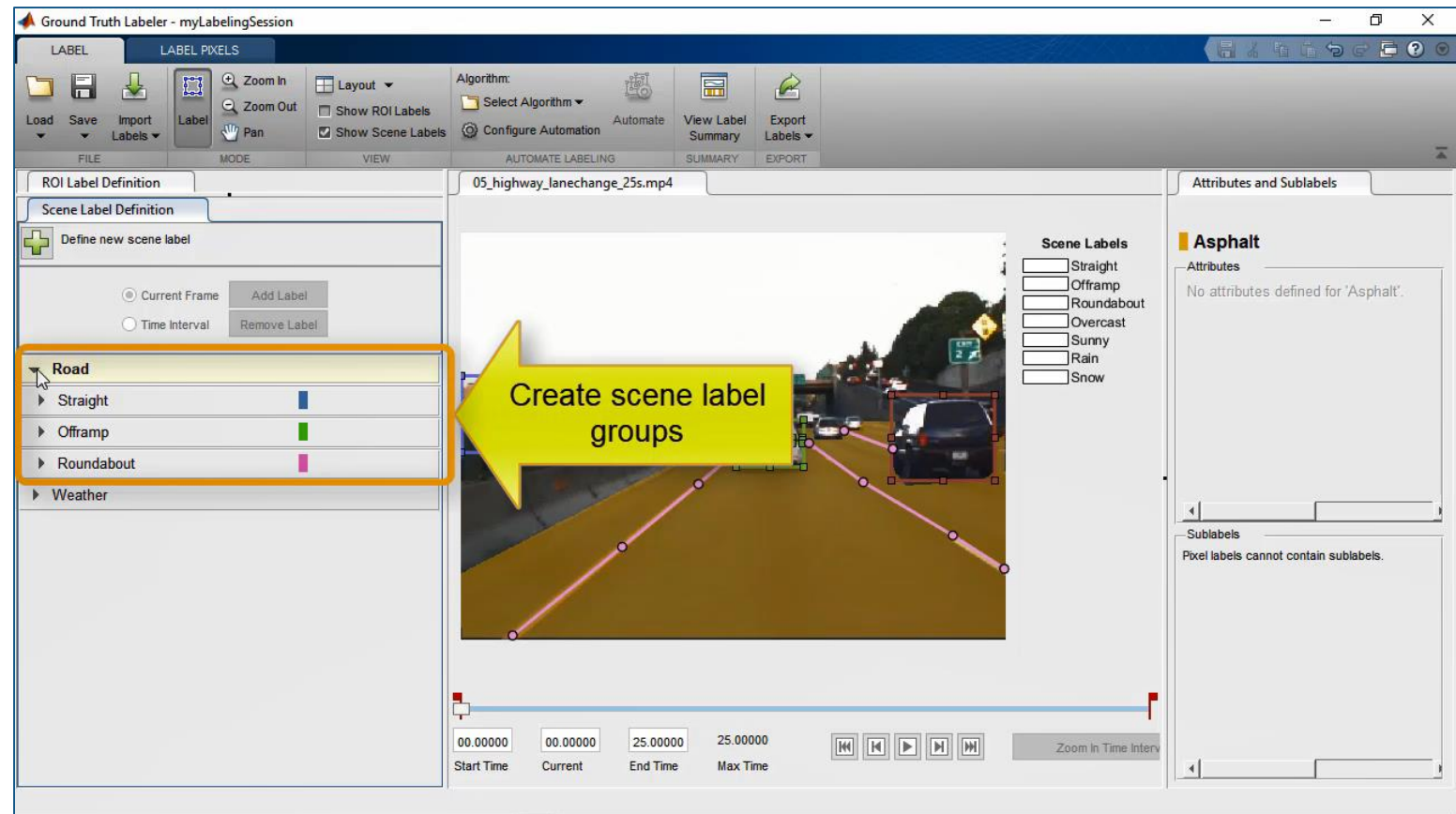
# Create scene labels and groups

## Get Started with the Ground Truth Labeler

- Label rectangles
- Label lane markings
- Label pixels
- Label scenes
- Create label groups
- Create sublabels
- Add label attributes

Automated Driving Toolbox™

Updated **R2019a**

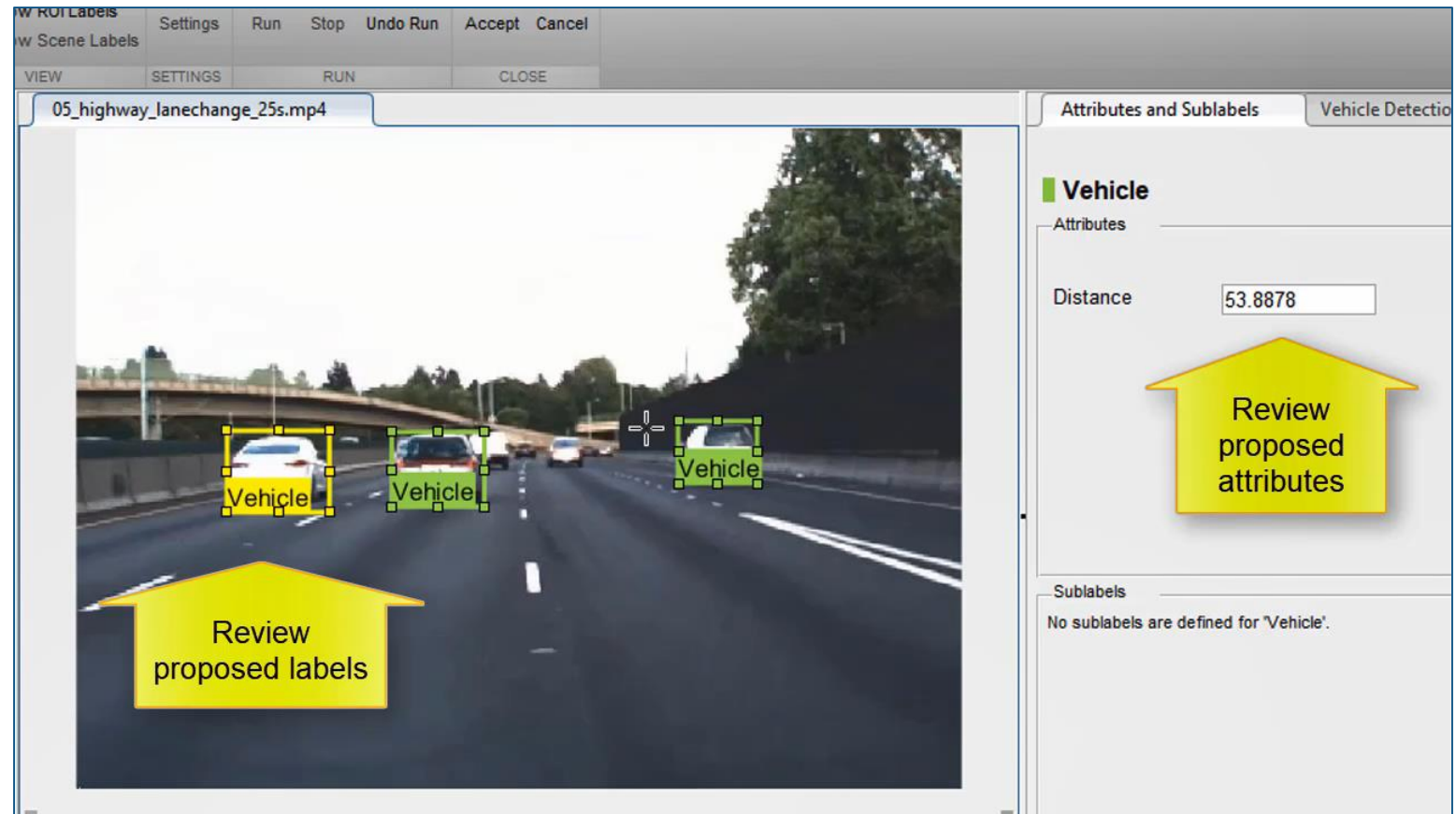


# Import custom automation algorithms

## Automate Attributes of Labeled Objects

- Import automation algorithm into Ground Truth Labeling app
- Detect vehicles from monocular camera
- Estimate distance to detected vehicles
- Run automation algorithm and interactively validate labels

*Automated Driving Toolbox™*  
**R2018b**

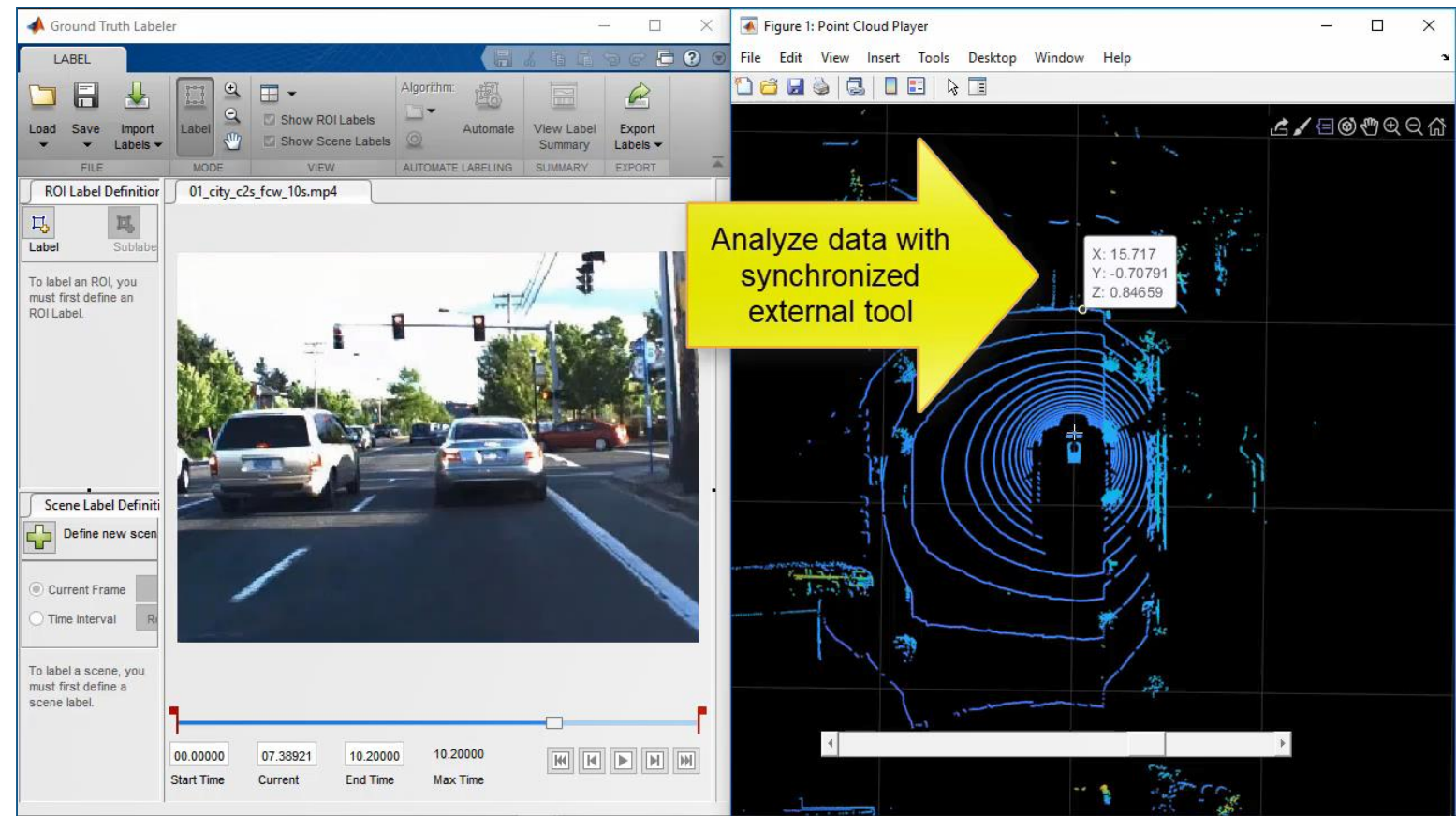


# Add custom visualizations for multi-sensor data

## Connect Lidar Display to Ground Truth Labeler

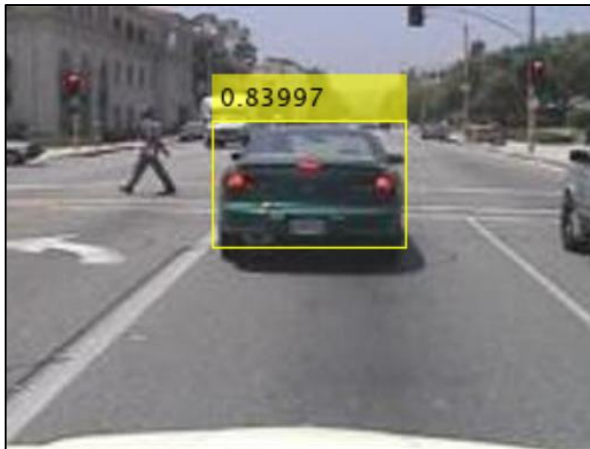
- Sync external tool to each frame change
- Control external tool through playback controls

Automated Driving Toolbox™  
**R2017a**



# Design camera, lidar, and radar perception algorithms

## Detect vehicle with camera

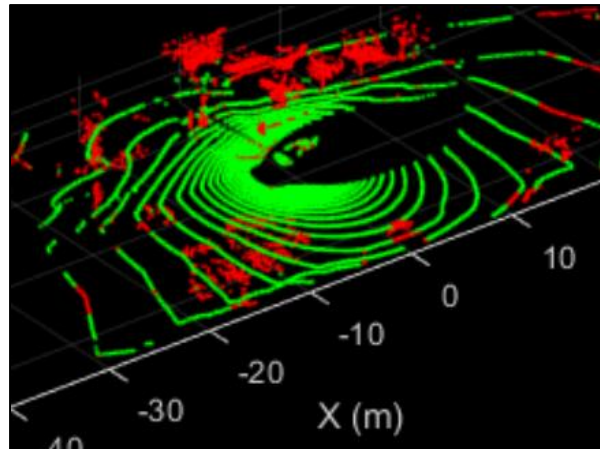


[Object Detection Using YOLO v2 Deep Learning](#)

*Computer Vision Toolbox™  
Deep Learning Toolbox™*

**R2019a**

## Detect ground with lidar

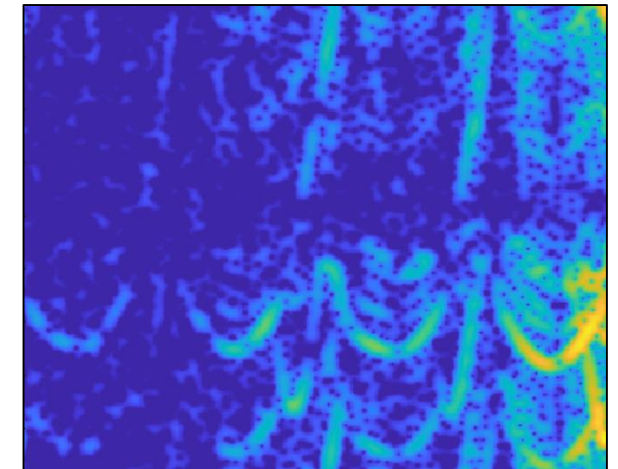


[Segment Ground Points from Organized Lidar Data](#)

*Computer Vision Toolbox™*

**R2018b**

## Detect pedestrian with radar

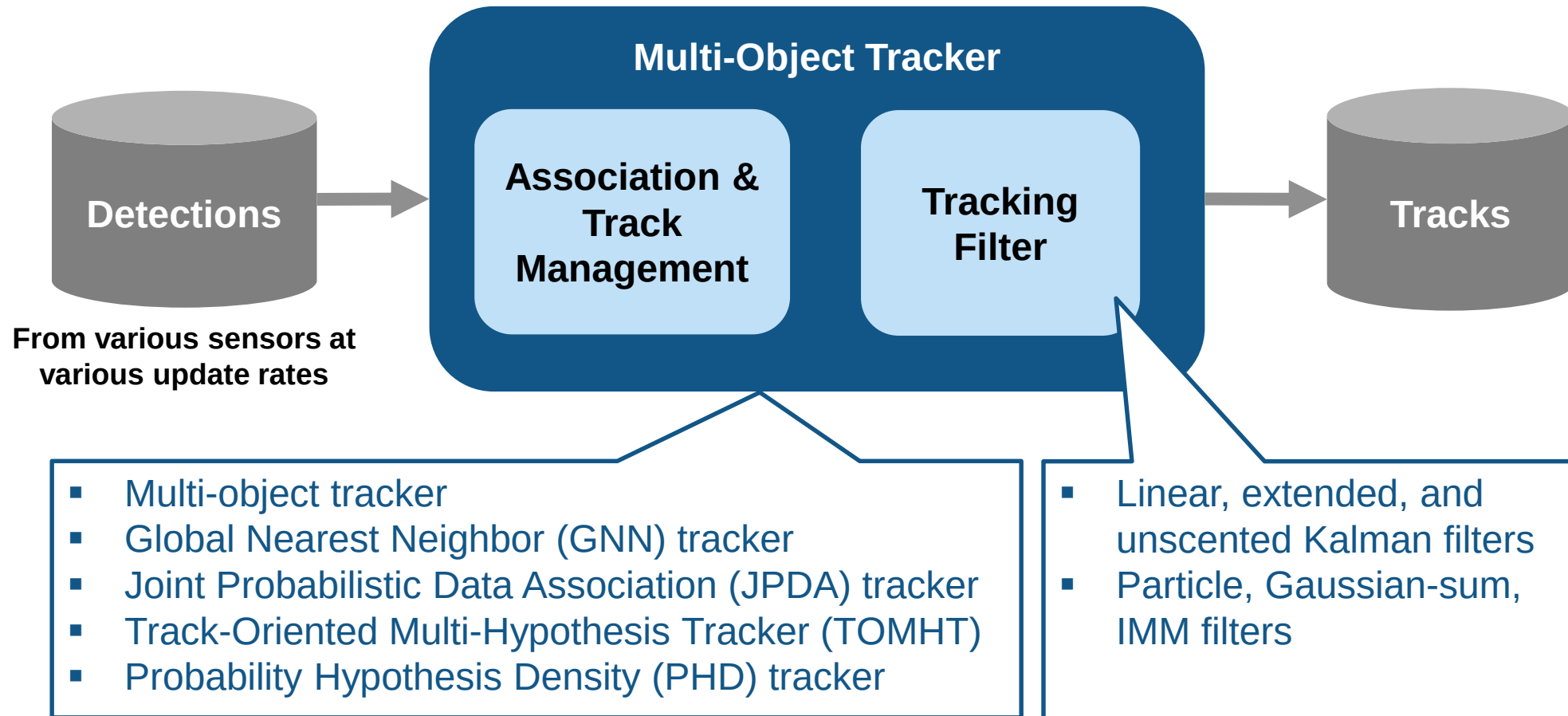


[Introduction to Micro-Doppler Effects](#)

*Phased Array System Toolbox™*

**R2019a**

# Design trackers



*Automated Driving Toolbox™*

*Sensor Fusion and Tracking Toolbox™*

Updated **R2019a**

# Design multi-object trackers

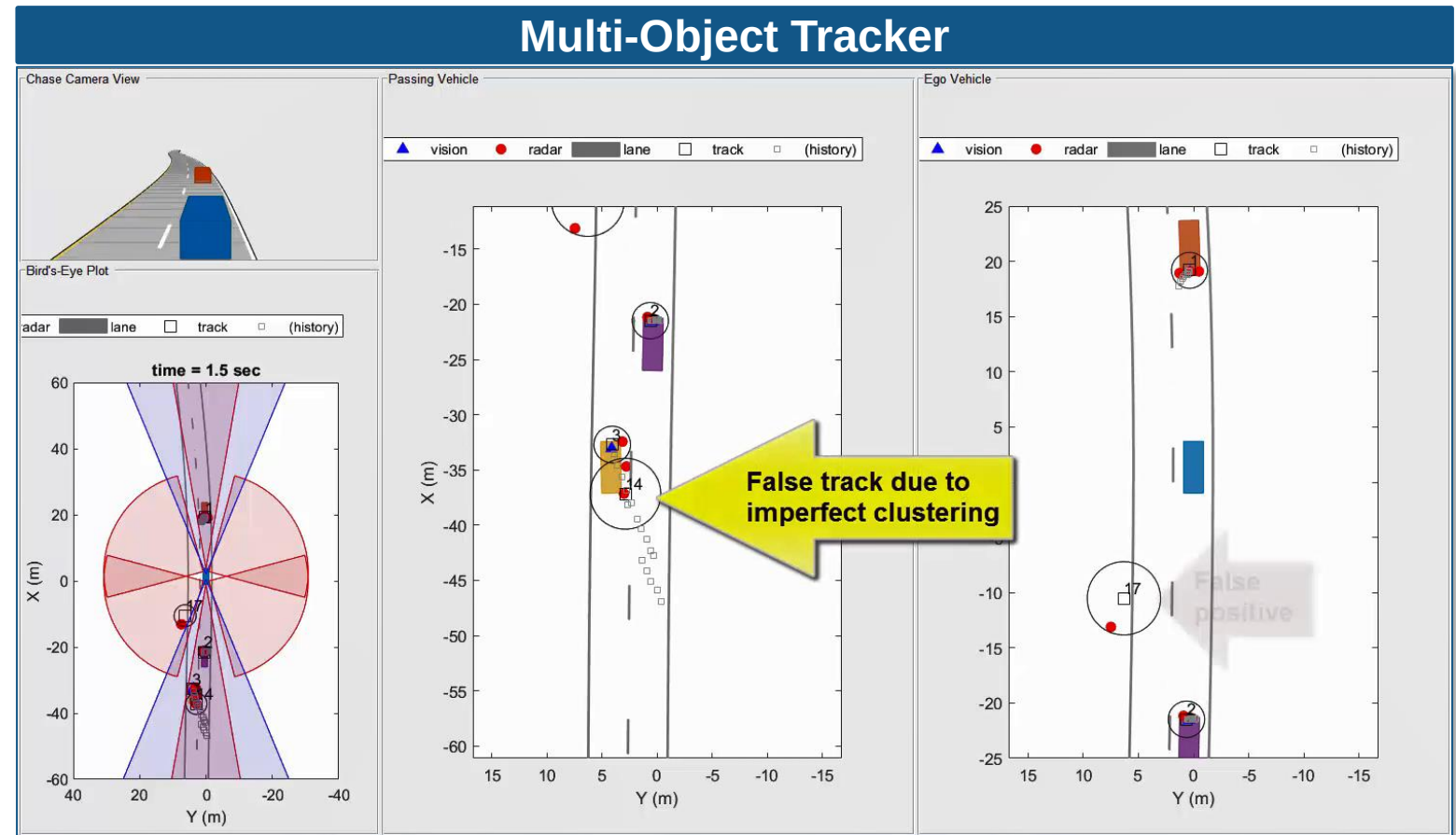
## Extended Object Tracking

- Design multi-object tracker
- Design extended object trackers
- Evaluate tracking metrics
- Evaluate error metrics
- Evaluate desktop execution time

*Sensor Fusion and  
Tracking Toolbox™*

*Automated Driving Toolbox™*

Updated **R2019a**



# Design extended object trackers

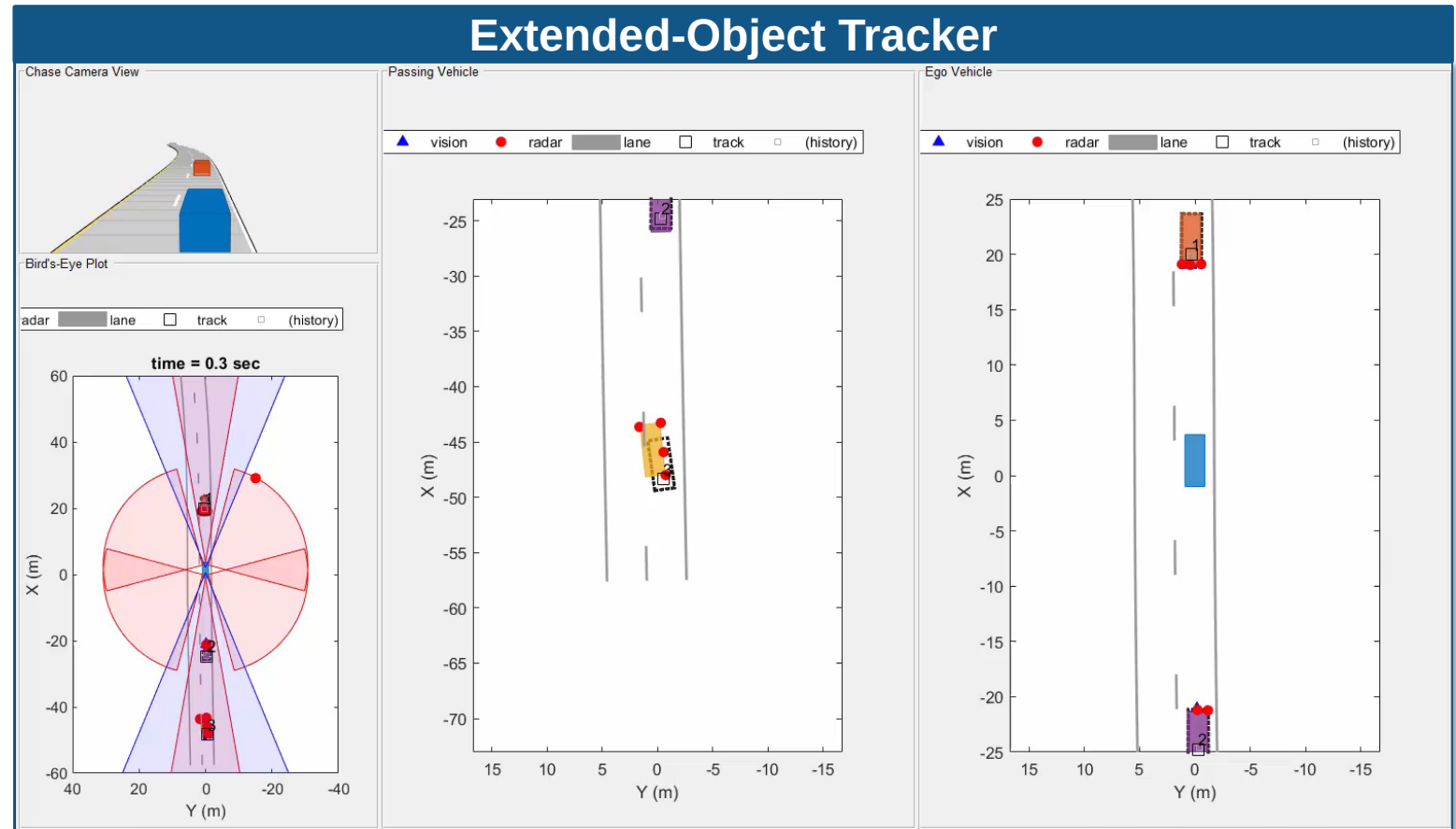
## Extended Object Tracking

- Design multi-object tracker
- Design extended object trackers
- Evaluate tracking metrics
- Evaluate error metrics
- Evaluate desktop execution time

*Sensor Fusion and  
Tracking Toolbox™*

*Automated Driving Toolbox™*

Updated **R2019a**



# Evaluate tracking performance

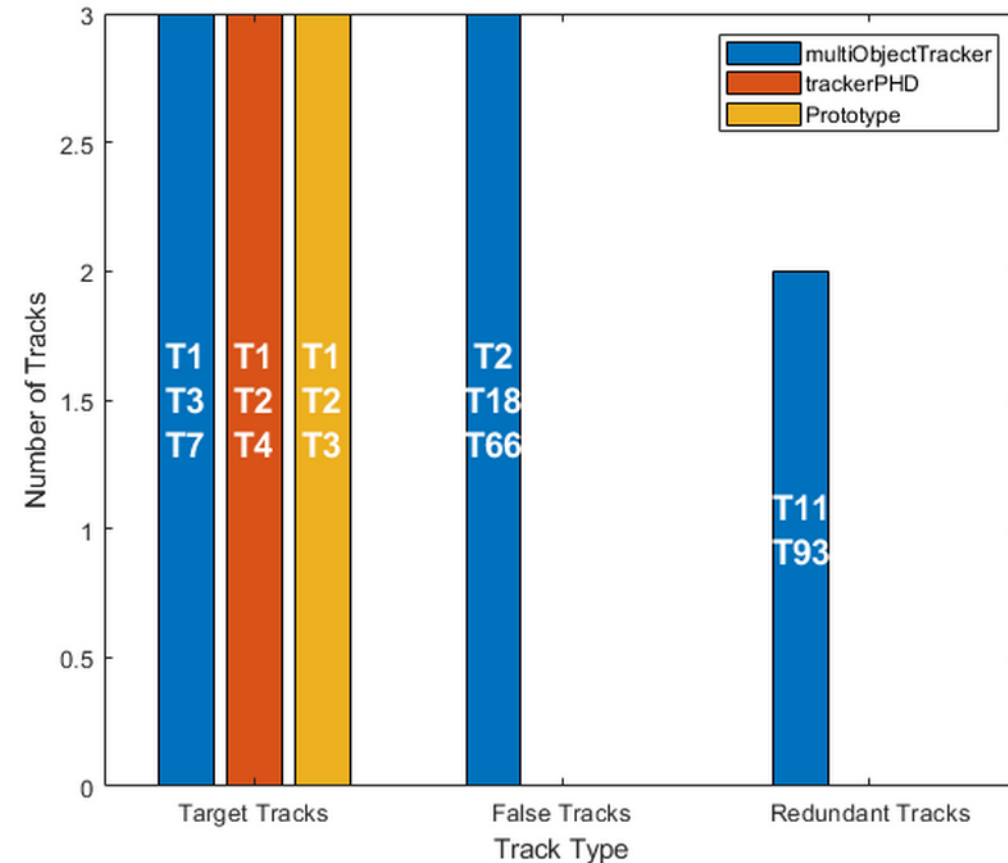
## Extended Object Tracking

- Design multi-object tracker
- Design extended object trackers
- Evaluate tracking metrics
- Evaluate error metrics
- Evaluate desktop execution time

*Sensor Fusion and  
Tracking Toolbox™*

*Automated Driving Toolbox™*

Updated **R2019a**



- Multi-object tracker
- Probability Hypothesis Density tracker
- Extended object (size and orientation) tracker

# Evaluate error metrics

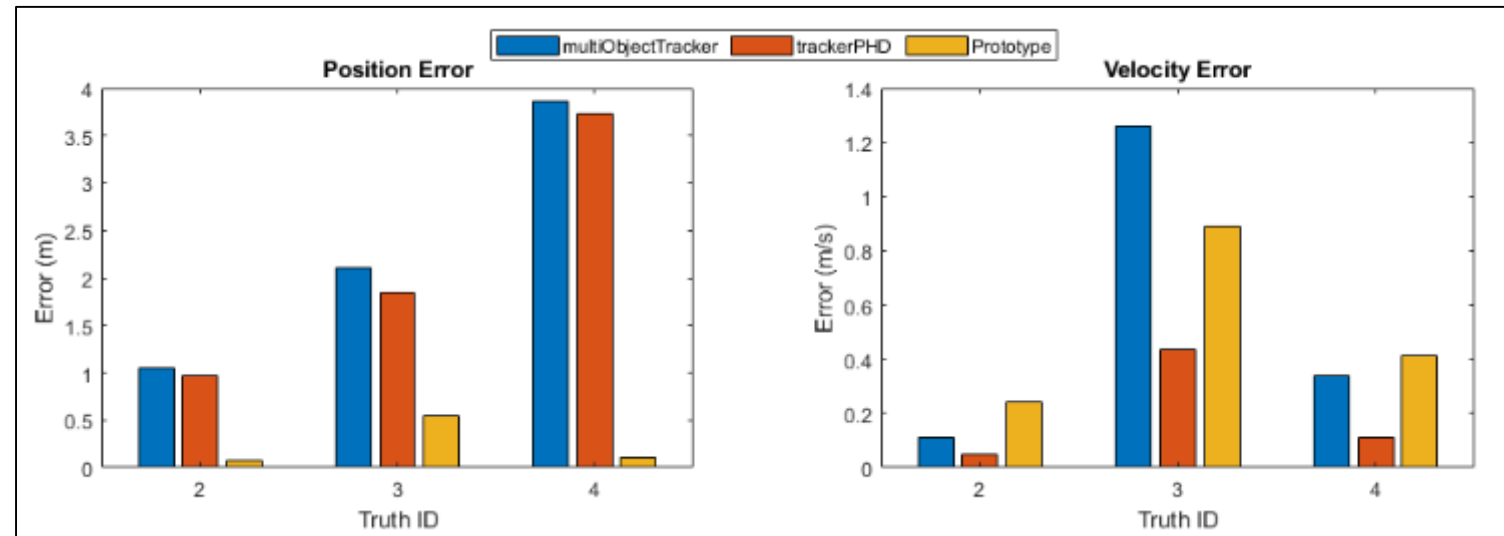
## Extended Object Tracking

- Design multi-object tracker
- Design extended object trackers
- Evaluate tracking metrics
- Evaluate error metrics
- Evaluate desktop execution time

*Sensor Fusion and  
Tracking Toolbox™*

*Automated Driving Toolbox™*

Updated **R2019a**



- Multi-object tracker
- Probability Hypothesis Density tracker
- Extended object (size and orientation) tracker

# Compare relative execution times of object trackers

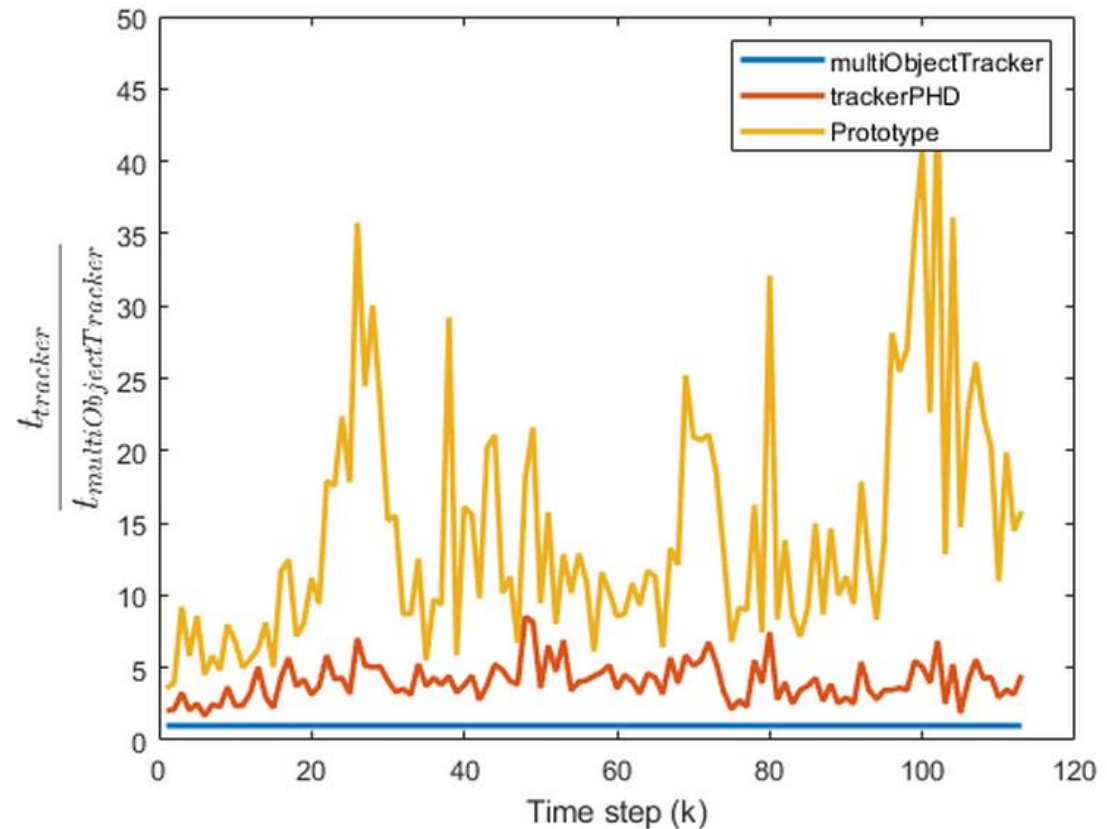
## Extended Object Tracking

- Design multi-object tracker
- Design extended object trackers
- Evaluate tracking performance
- Evaluate error metrics
- Evaluate desktop execution time

*Sensor Fusion and  
Tracking Toolbox™*

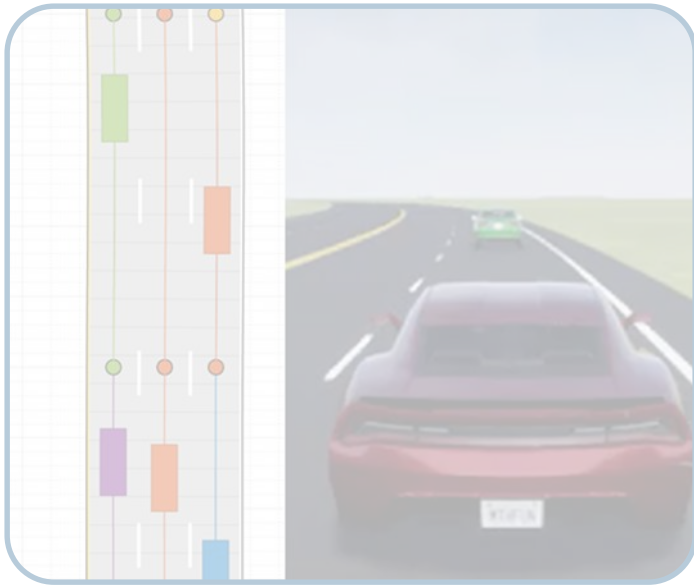
*Automated Driving Toolbox™*

Updated **R2019a**

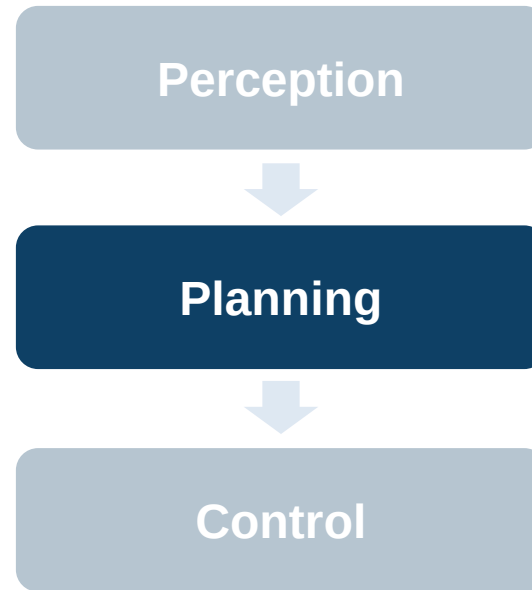


- Multi-object tracker
- Probability Hypothesis Density tracker
- Extended object (size and orientation) tracker

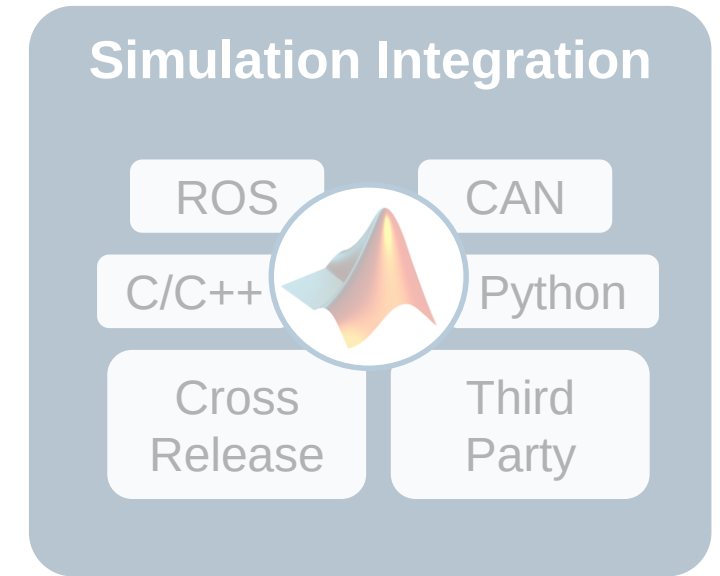
# Some common questions from automated driving engineers



How can I  
**synthesize scenarios**  
to test my designs?



How can I  
**discover and design**  
in multiple domains?



How can I  
**integrate**  
with other environments?

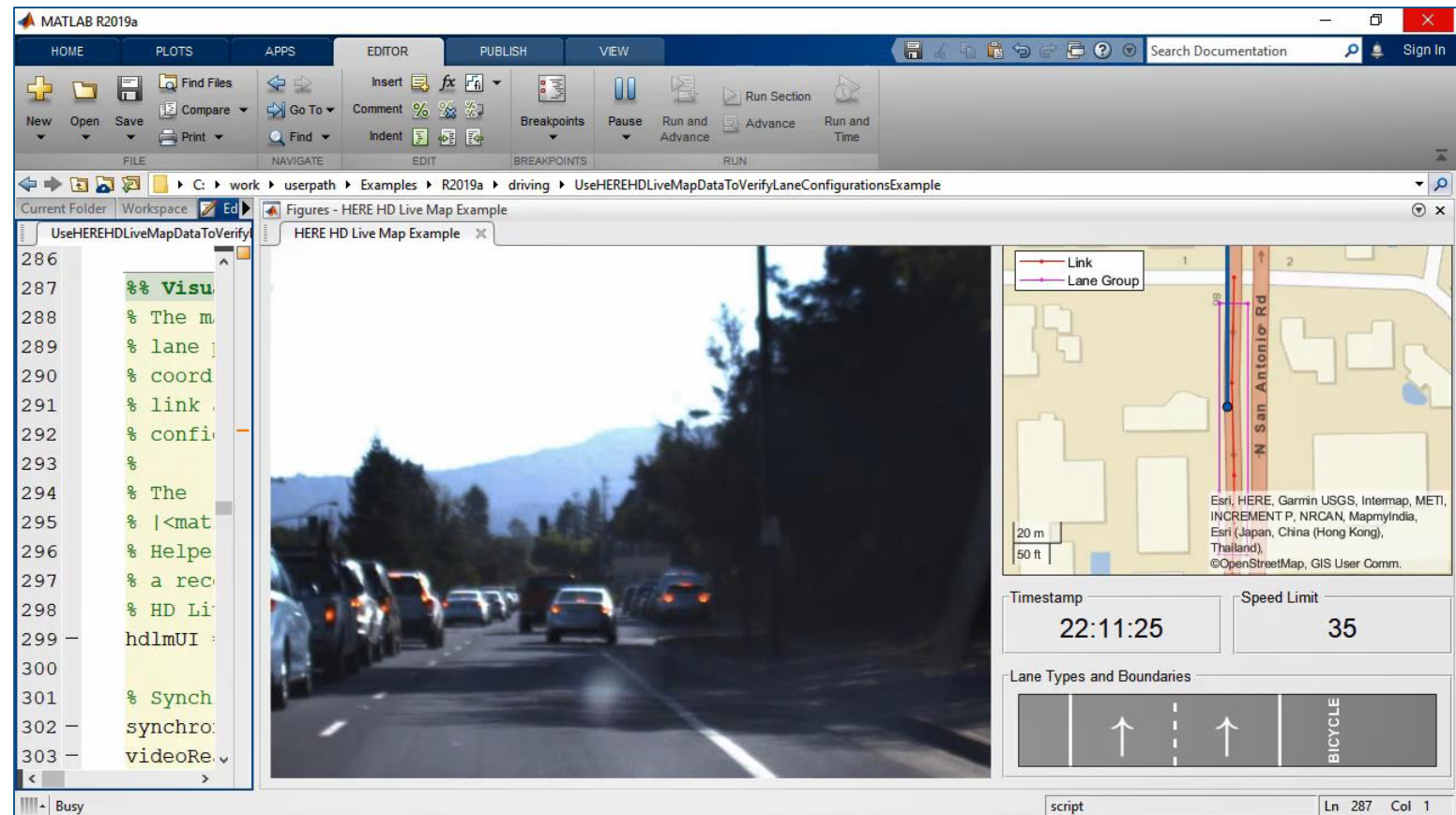
# Visualize HERE HD Live Map recorded data

## Use HERE HD Live Map Data to Verify Lane Configurations

- Load camera and GPS data
- Retrieve speed limit
- Retrieve lane configurations
- Visualize composite data

Automated Driving Toolbox™

R2019a



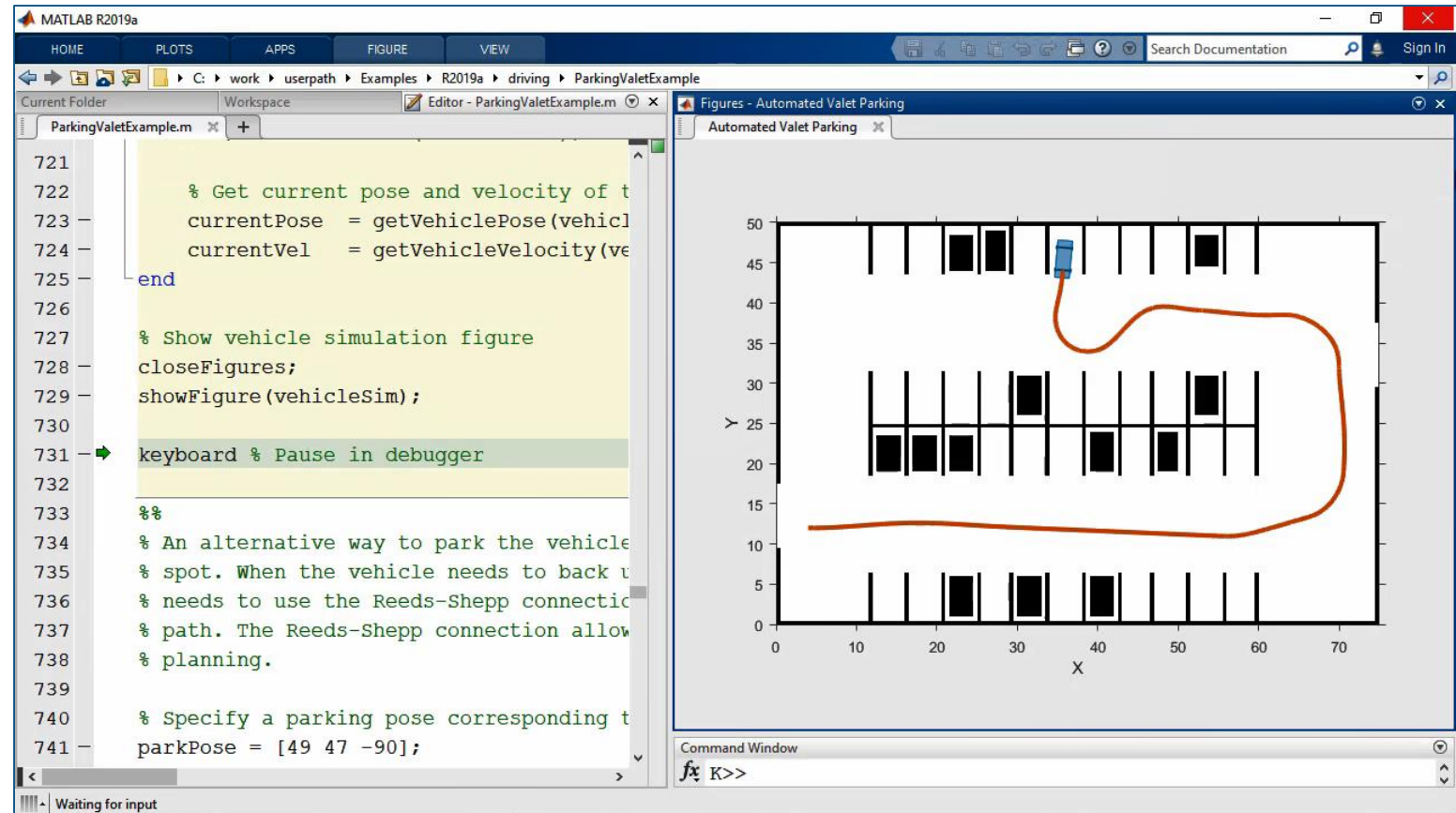
# Design path planner

## Automated Parking Valet

- Create cost map of environment
- Inflate cost map for collision checking
- Specify goal poses
- Plan path using rapidly exploring random tree (RRT\*)

Automated Driving Toolbox™

**R2018a**



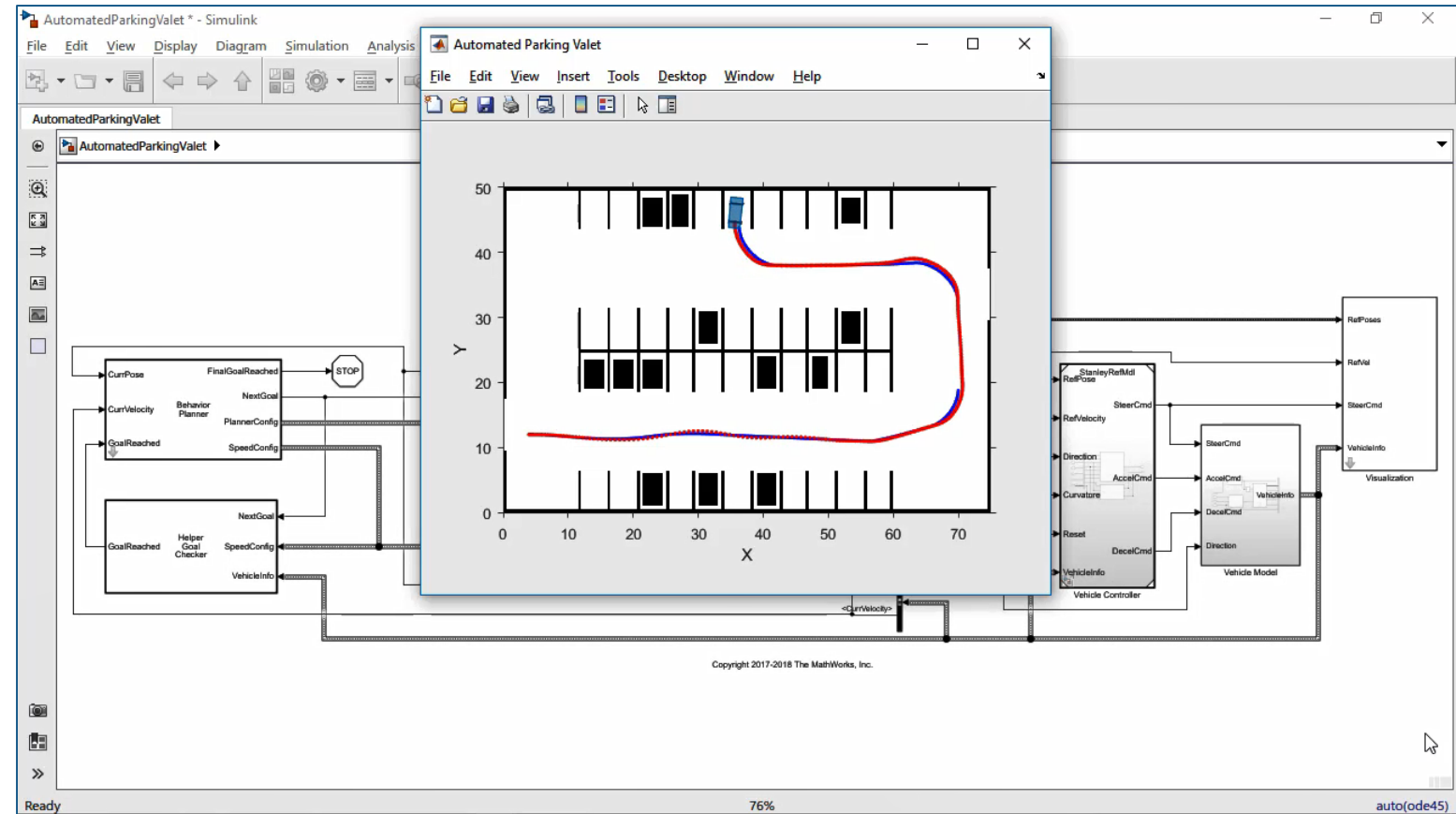
# Design path planner and controller

## Automated Parking Valet with Simulink

- Integrate path planner
- Design lateral controller (based on vehicle kinematics)
- Design longitudinal controller (PID)
- Simulate closed loop with vehicle dynamics

Automated Driving Toolbox™

**R2018b**



# Generate C/C++ code for path planner and controller

## Code Generation for Path Planning and Vehicle Control

- Simulate system
- Configure for code generation
- Generate C/C++ code
- Test using Software-In-the-Loop
- Measure execution time of generated code

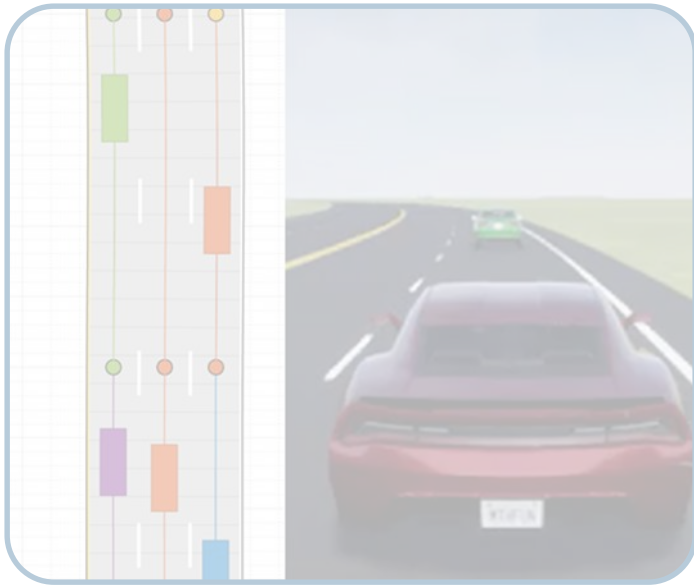
Automated Driving Toolbox™

Embedded Coder

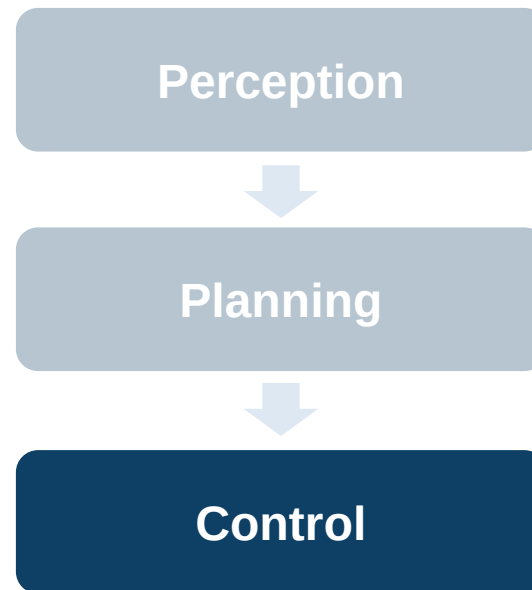
**R2019a**

```
186
187 // model step function
188 void step0();
189
190 // model step function
191 void step1();
192
193 // model terminate function
194 void terminate();
195
196 // Constructor
197 AutomatedParkingValetModelClass();
198
199 // Destructor
200 ~AutomatedParkingValetModelClass();
201
202 // Root input: '<Root>/Costmap' set method
203 void setCostmap(costmapBus localArgInput);
204
205 // Root input: '<Root>/GoalPose' set method
206 void setGoalPose(real_T localArgInput[3]);
207
```

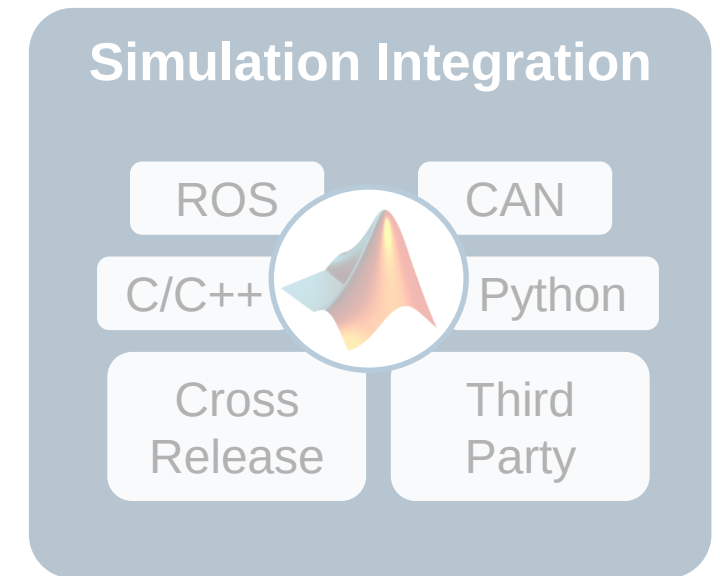
# Some common questions from automated driving engineers



How can I  
**synthesize scenarios**  
to test my designs?



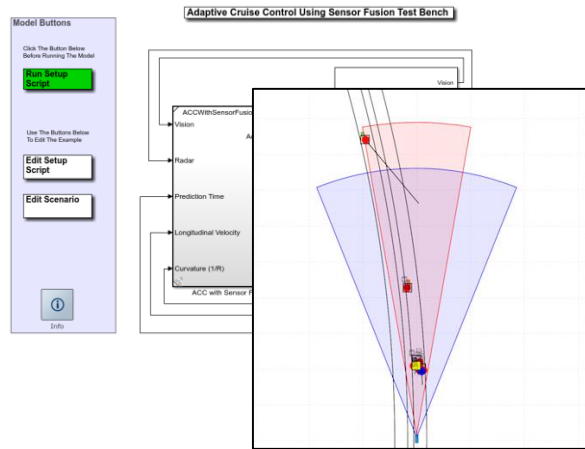
How can I  
**discover and design**  
in multiple domains?



How can I  
**integrate**  
with other environments?

# Design lateral and longitudinal Model Predictive Controllers

## Longitudinal Control

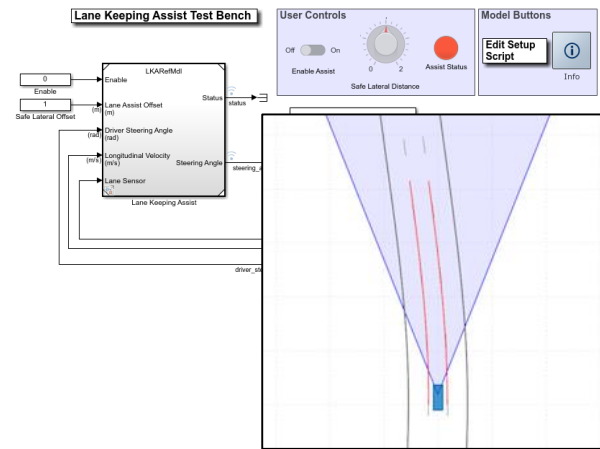


### Adaptive Cruise Control with Sensor Fusion

Automated Driving Toolbox™  
Model Predictive Control  
Toolbox™  
Embedded Coder®

**R2017b**

## Lateral Control

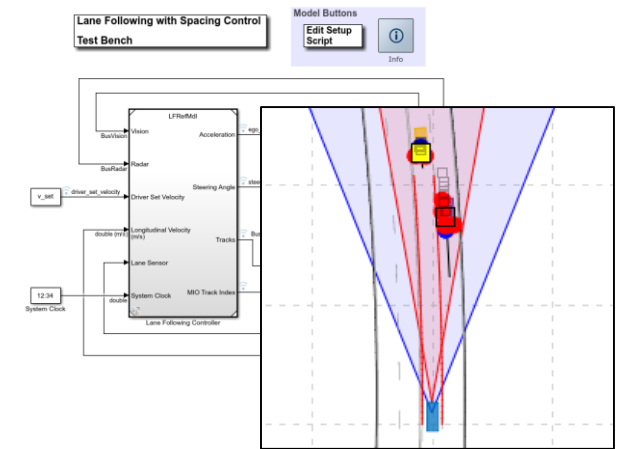


### Lane Keeping Assist with Lane Detection

Automated Driving Toolbox™  
Model Predictive Control  
Toolbox™  
Embedded Coder®

**R2018a**

## Longitudinal + Lateral



### Lane Following Control with Sensor Fusion and Lane Detection

Automated Driving Toolbox™  
Model Predictive Control Toolbox™  
Embedded Coder®

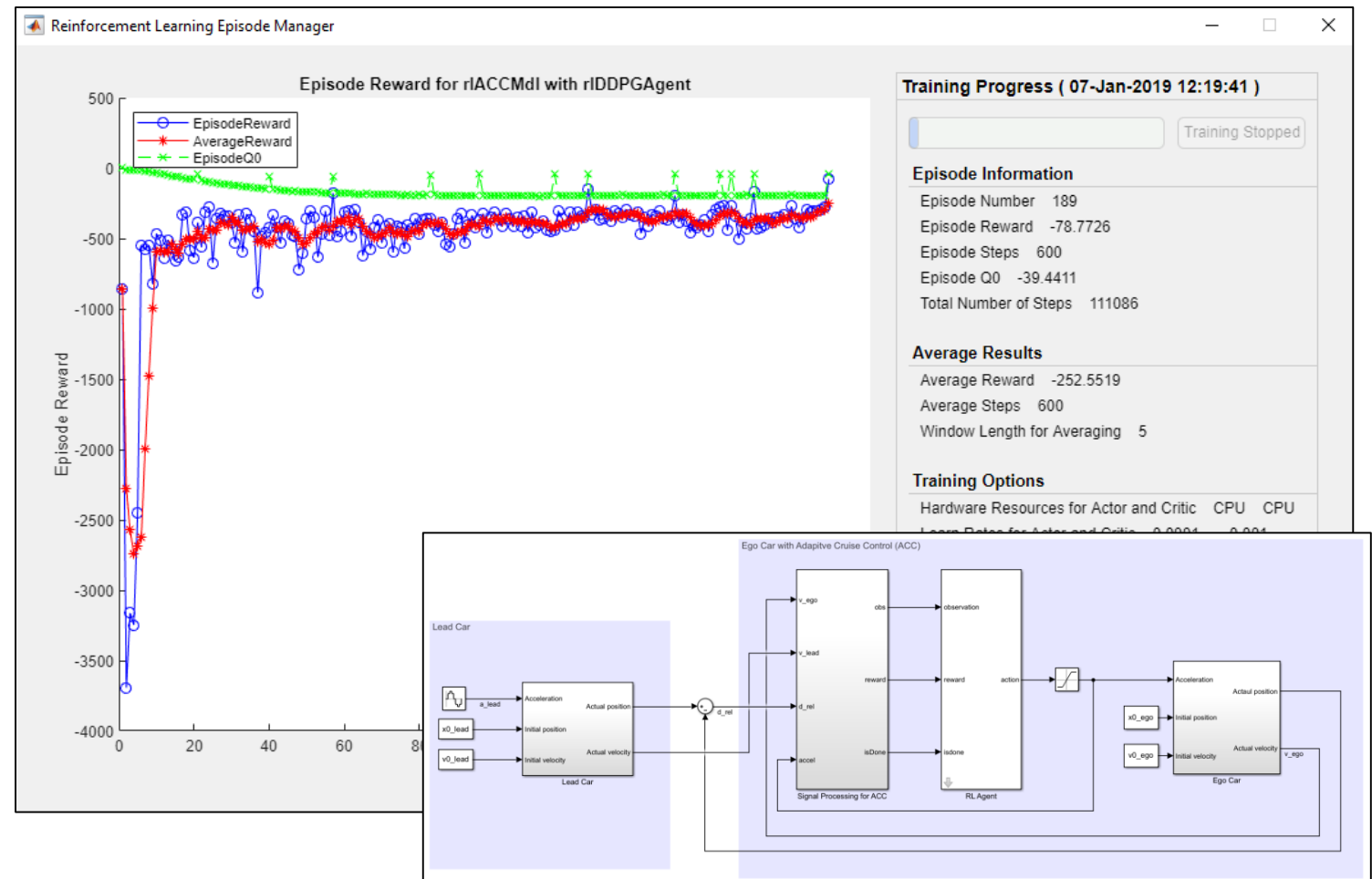
**R2018b**

# Train reinforcement learning networks for ADAS controllers

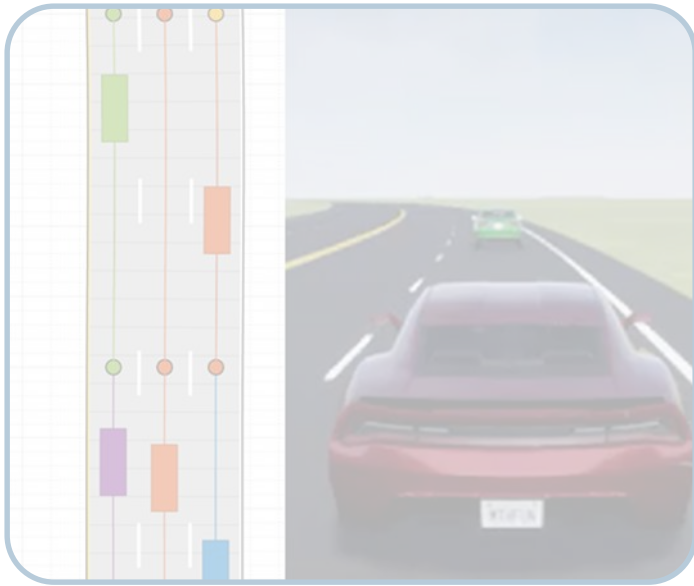
## Train Deep Deterministic Policy Gradient (DDPG) Agent for Adaptive Cruise Control

- Create environment interface
- Create agent
- Train agent
- Simulate trained agent

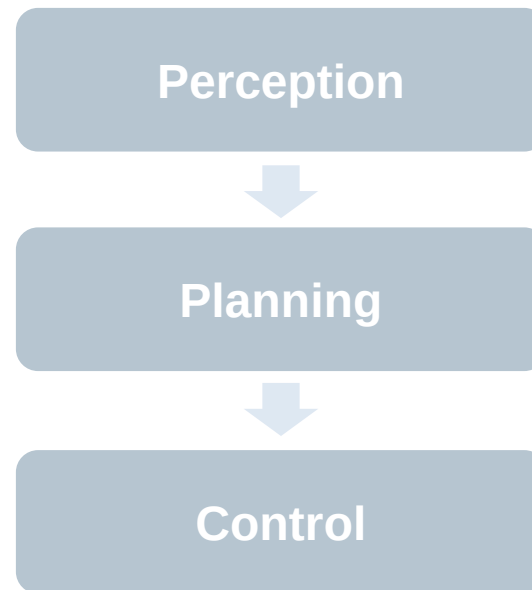
Reinforcement Learning Toolbox™  
**R2019a**



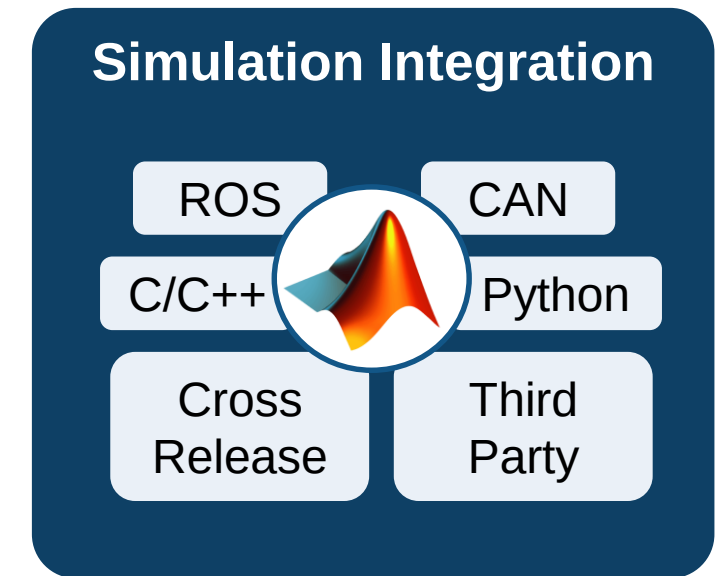
# Some common questions from automated driving engineers



How can I  
**synthesize scenarios**  
to test my designs?



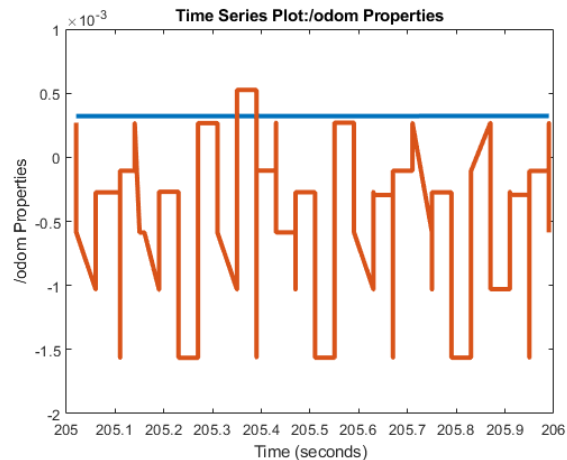
How can I  
**discover and design**  
in new domains?



How can I  
**integrate**  
with other environments?

# Integrate with ROS

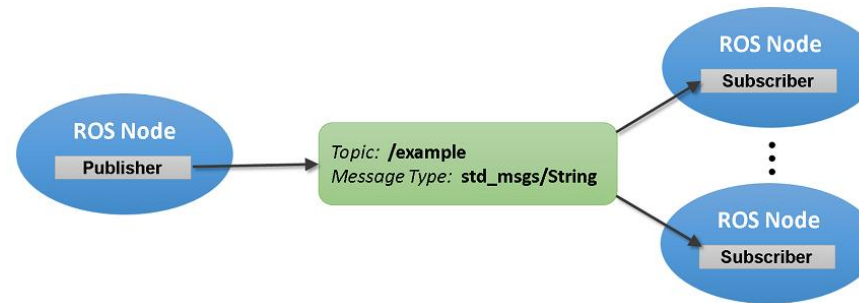
## Replay logged ROS data



[Work with rosbag Logfiles](#)

*Robotic System Toolbox™*

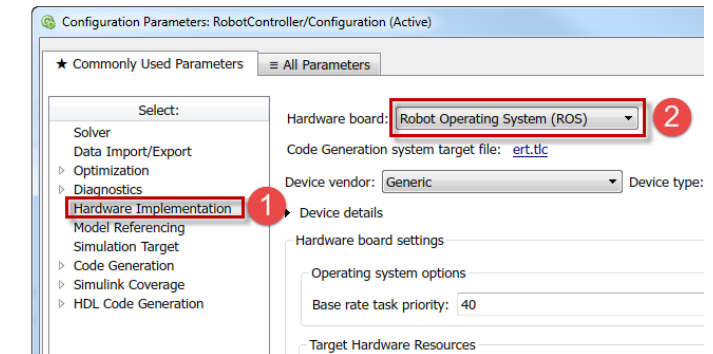
## Connect to live ROS data



[Exchange Data with ROS Publishers and Subscribers](#)

*Robotic System Toolbox™*

## Generate standalone ROS node

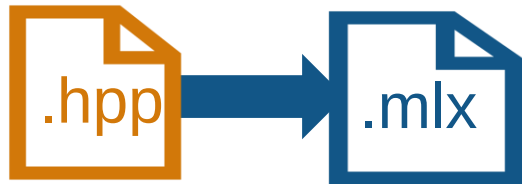


[Generate a Standalone ROS Node from Simulink](#)

*Robotic System Toolbox™  
Simulink Coder™*

# Call C++, Python, and OpenCV from MATLAB

## Call C++



[Import C++ Library  
Functionality into MATLAB](#)

MATLAB®

**R2019a**

## Call Python

```
tw = ...  
py.textwrap.TextWrapper(...  
    pyargs(...  
        'initial_indent', '% ',...  
        'subsequent_indent', '% ',...  
        'width', int32(30)))
```

[Call Python from MATLAB](#)

MATLAB®

**R2014a**

## Call OpenCV & OpenCV GPU

```
cv::Rect  
cv::KeyPoint  
cv::Size  
cv::Mat  
cv::Ptr  
...
```



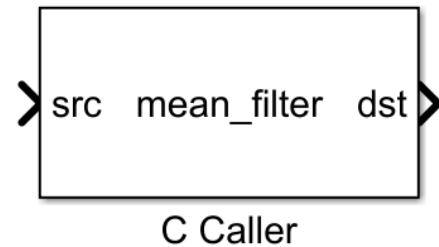
[Install and Use Computer Vision  
Toolbox OpenCV Interface](#)

Computer Vision System Toolbox™  
OpenCV Interface Support Package

Updated **R2018b**

# Call C code from Simulink

## Call C code



[Bring Custom Image Filter Algorithms as Reusable Blocks in Simulink](#)

Simulink®

**R2017b**

## Create buses from C structs

```
typedef struct {
    double coeff;
    double init;
    fault_T fault;
} params_T;
```

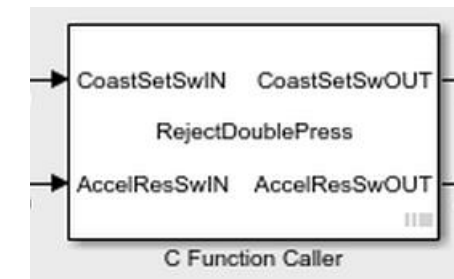
|                | Name  | DataType      |
|----------------|-------|---------------|
| Base Workspace |       |               |
| params_T       |       |               |
| coeff          | coeff | double        |
| init           | init  | double        |
| fault          | fault | Enum: fault_T |

[Import Structure and Enumerated Types](#)

Simulink®

**R2017a**

## Test and verify C code



| AGGREGATED COVERAGE RESULTS |          |           |      |
|-----------------------------|----------|-----------|------|
| ANALYZED MODEL              | DECISION | CONDITION | MCDC |
| RejectDoublePress.c         | 100%     | 100%      | 100% |

[Custom C Code Verification with Simulink Test](#)

Simulink Test™

Simulink Coverage™

**R2019a**

# Cross-release simulation through code generation

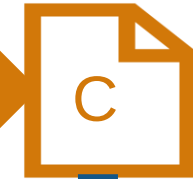
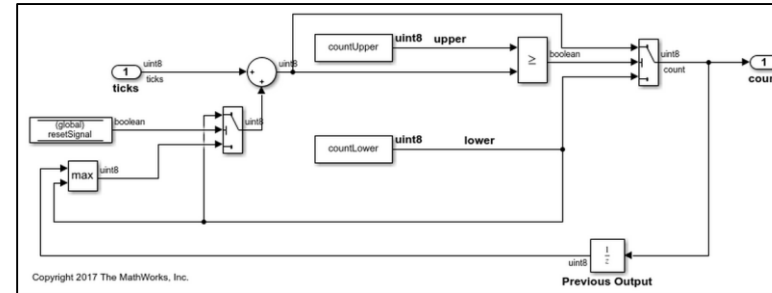
## Integrate Generated Code by Using Cross-Release Workflow

- Generate code from previous release (R2010a or later)
- Import generated code as a block in current release
- Tune parameters
- Access internal signals

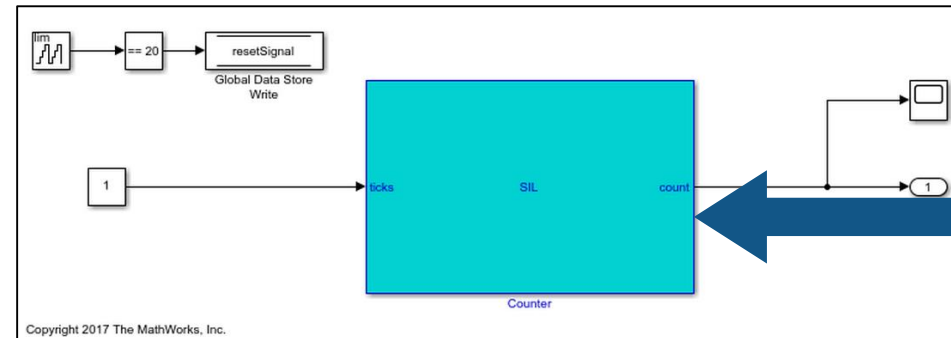
Embedded Coder

**R2016a**

## Previous Release

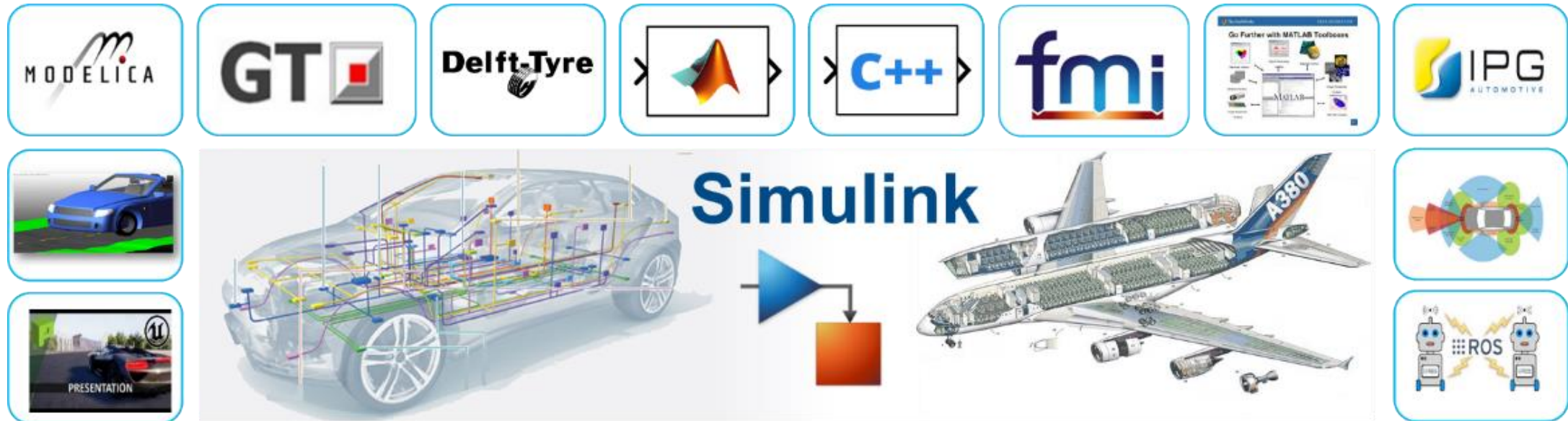


## Current Release



**crossReleaseImport**

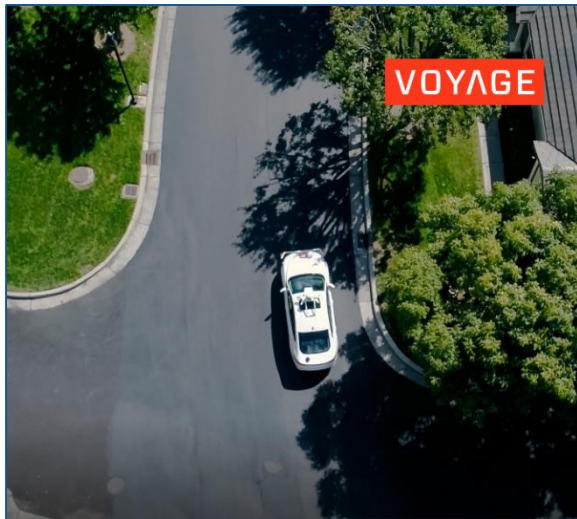
# Connect to third party tools



152 Interfaces to 3<sup>rd</sup> Party  
Modeling and Simulation Tools  
(as of March 2019)



# MathWorks can help you customize MATLAB and Simulink for your automated driving application



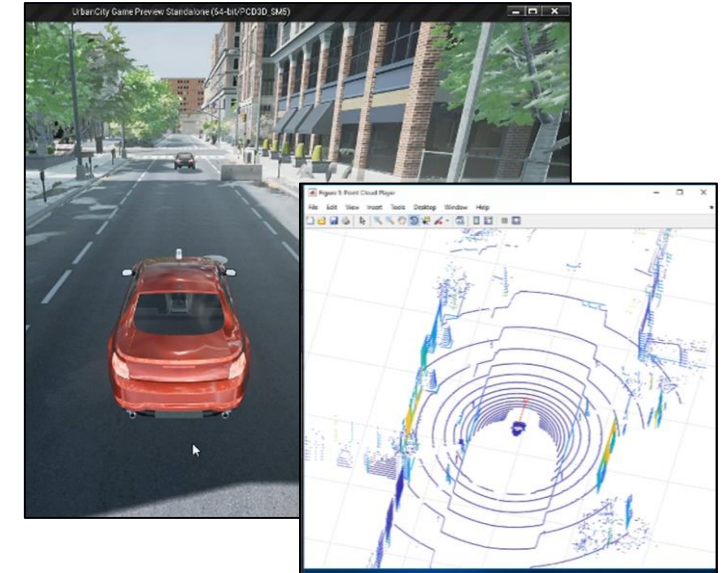
## Voyage develops MPC controller and integrates with ROS

- 2018 MathWorks Automotive Conference



## Autoliv labels ground truth lidar data

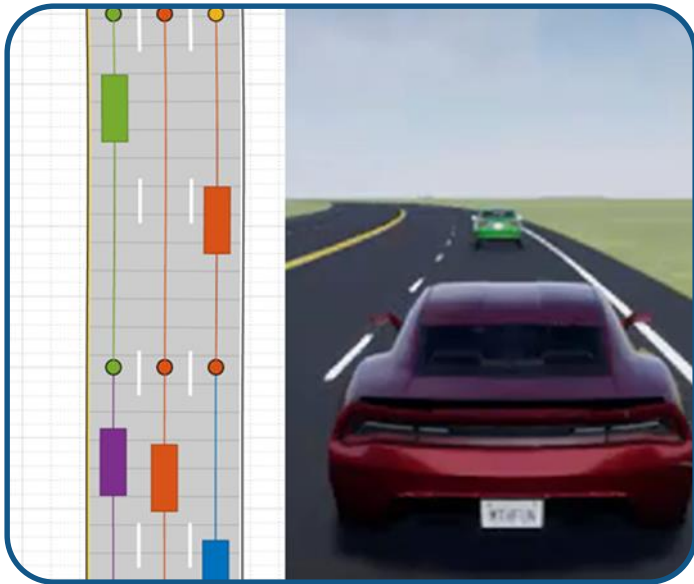
- Joint presentation with Autoliv
- SAE Paper 2018-01-0043
- 2018 MathWorks Automotive Conference



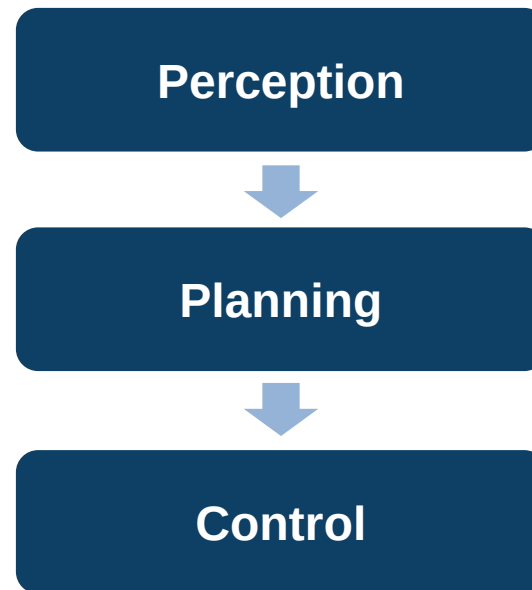
## Ford tests algorithms with synthetic Lidar data from Unreal Engine

- Joint paper with Ford
- SAE Paper 2017-01-0107

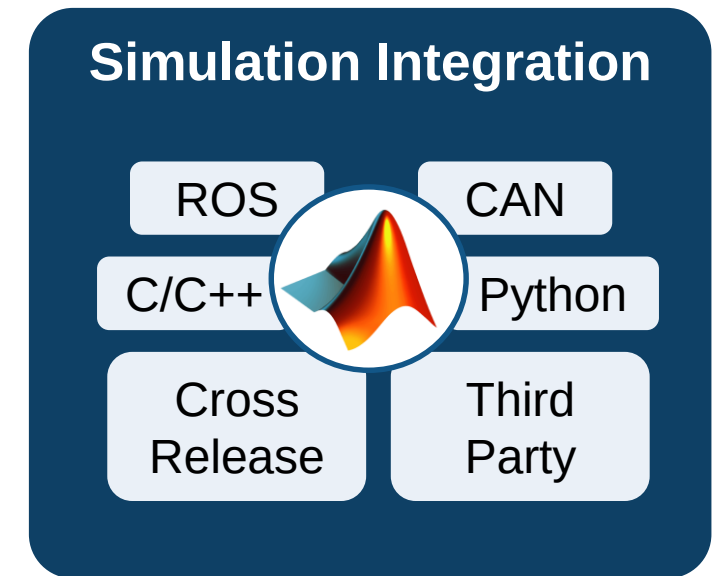
# Some common questions from automated driving engineers



**Synthesize scenarios**  
to test my designs



**Discover and design**  
in multiple domains



**Integrate**  
with other environments