

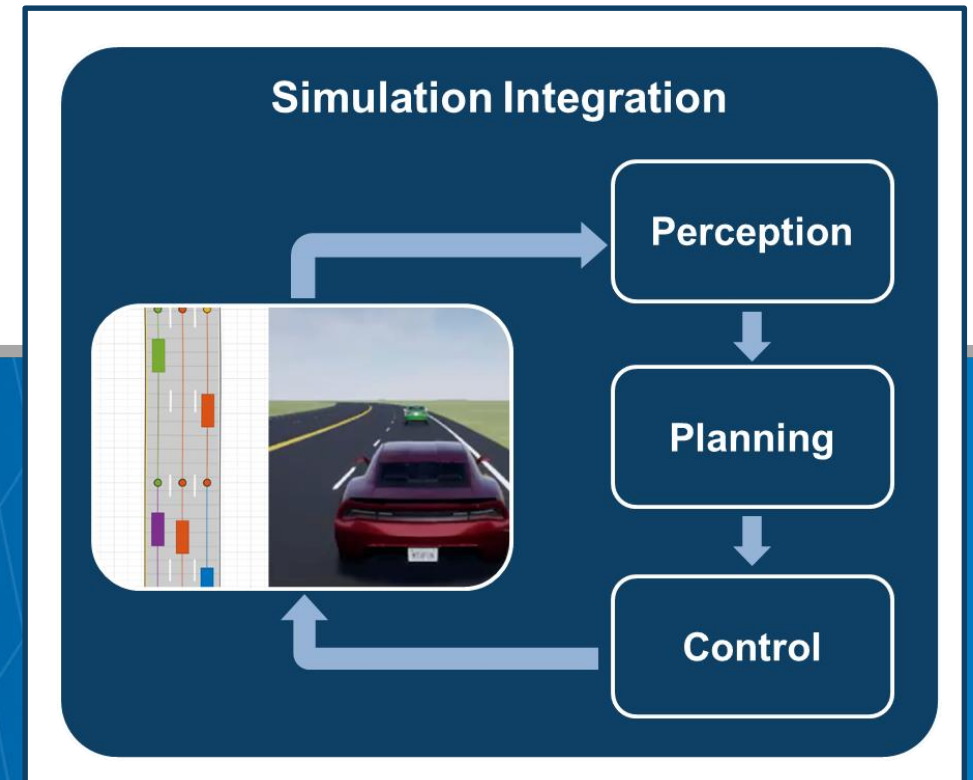
What's New in Automated Driving with MATLAB and Simulink

Mark Corless

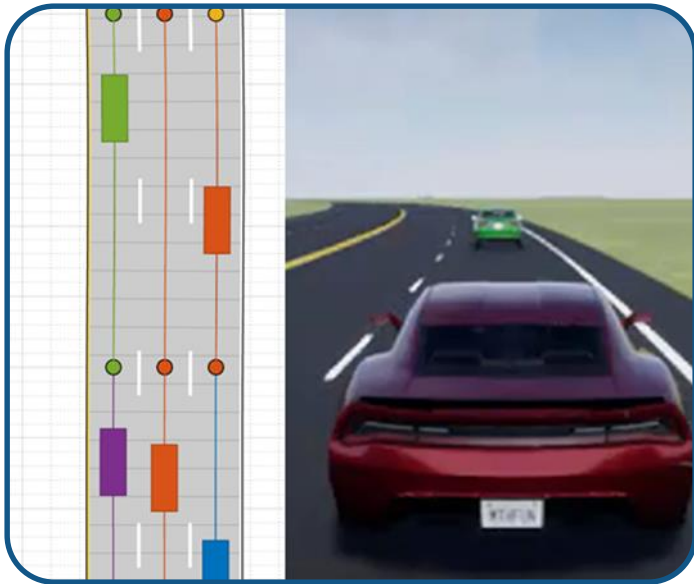
Industry Marketing
Automated Driving Segment Manager

Marco Roggero

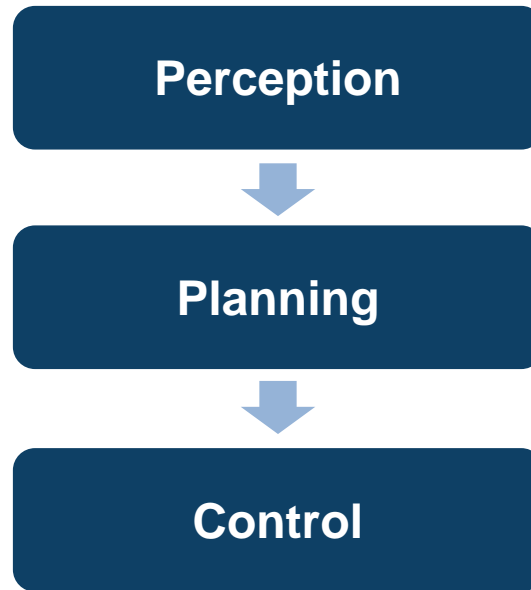
Application Engineering



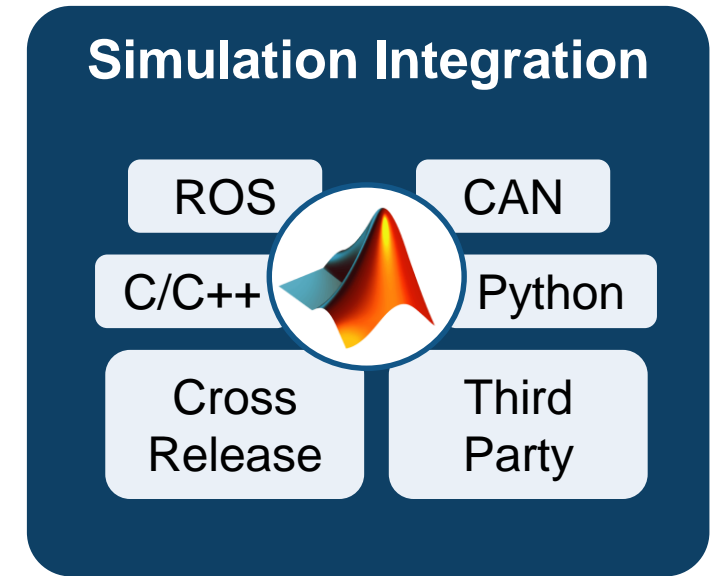
Some common questions from automated driving engineers



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

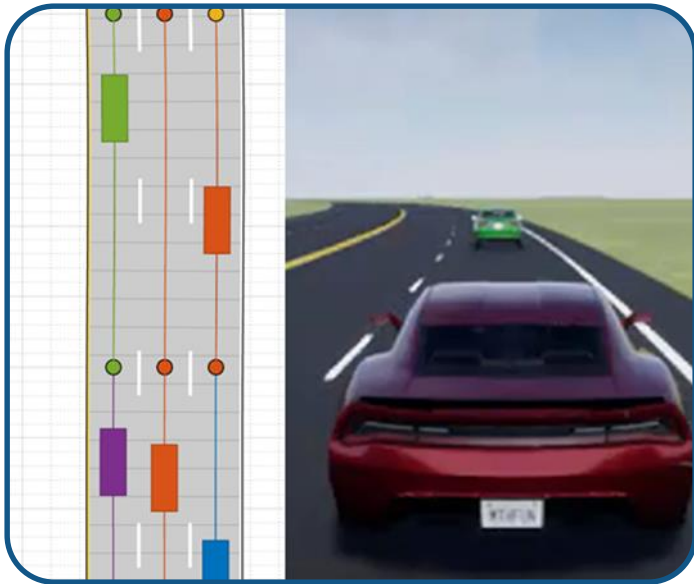


How can I **discover and design** in multiple domains?

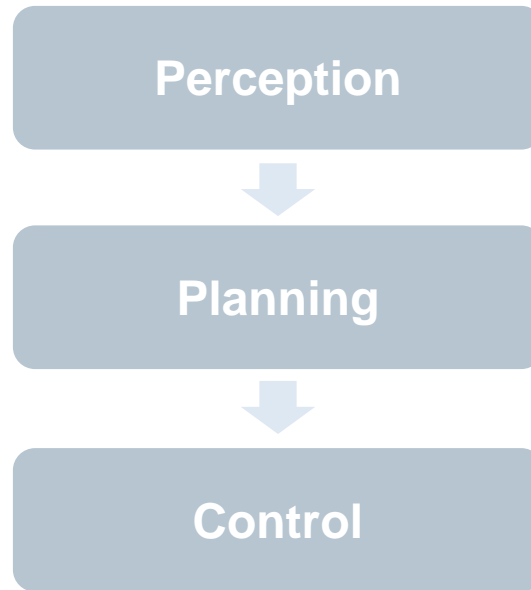


How can I **integrate** with other environments?

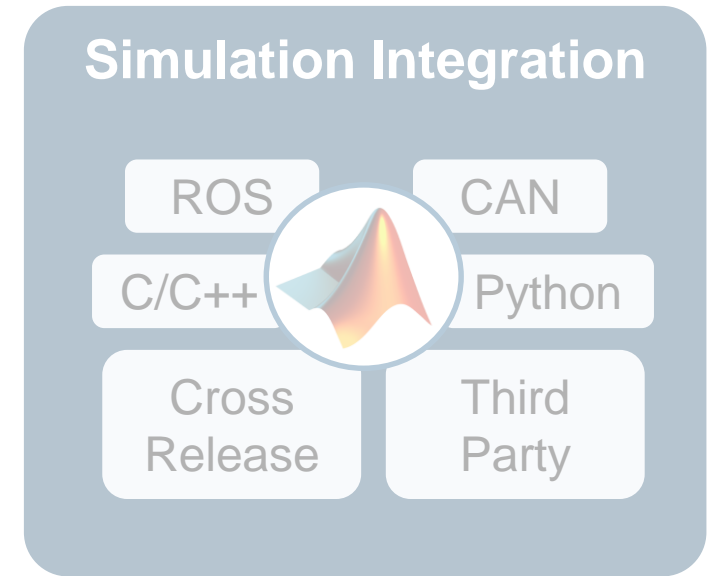
Some common questions from automated driving engineers



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?

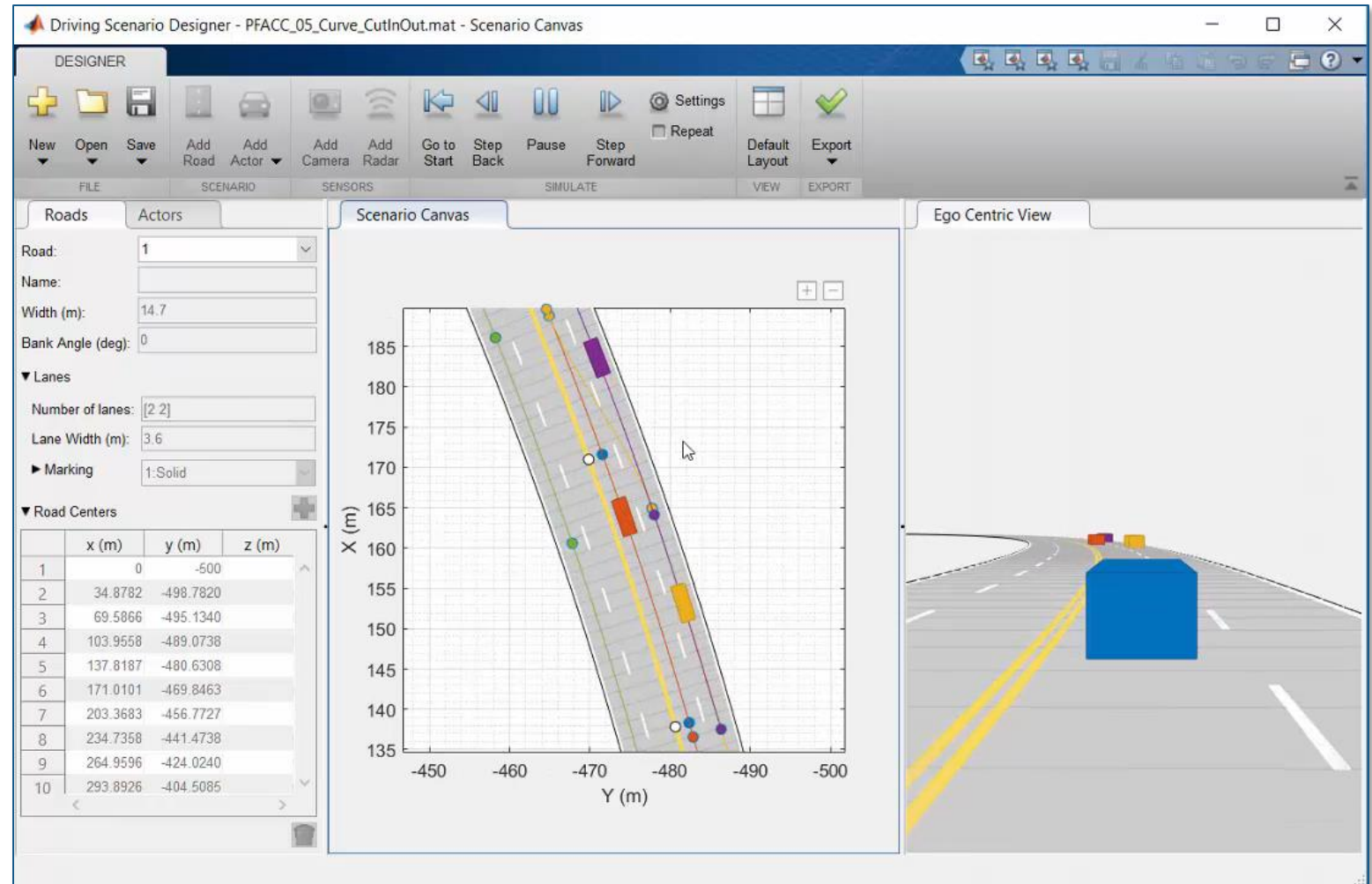
Graphically author driving scenarios

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™

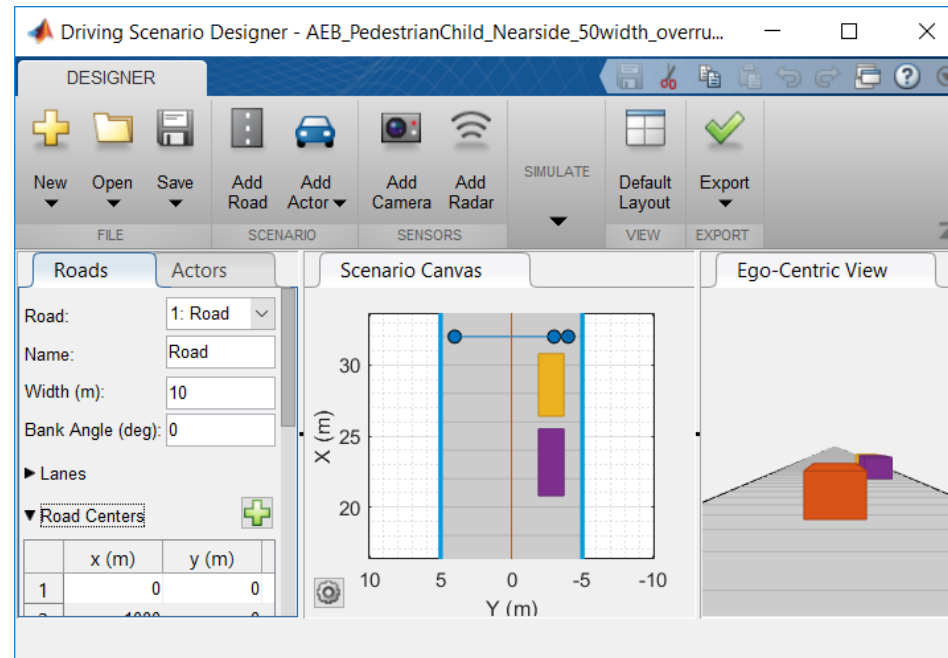
R2018a



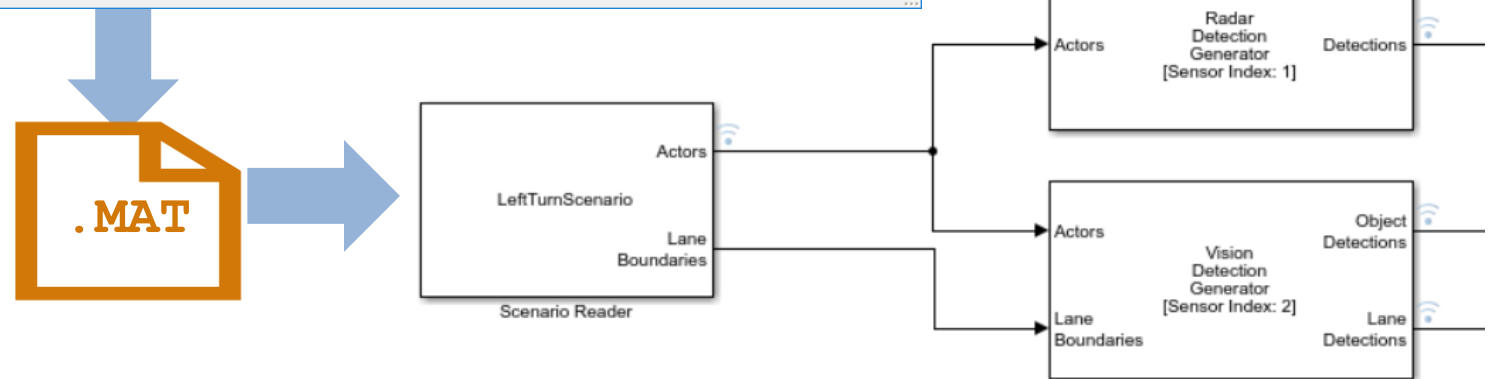
Integrate driving scenarios into Simulink simulations

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



Simulate driving scenarios into closed loop simulations

Automatic Emergency Braking (AEB) with Sensor Fusion

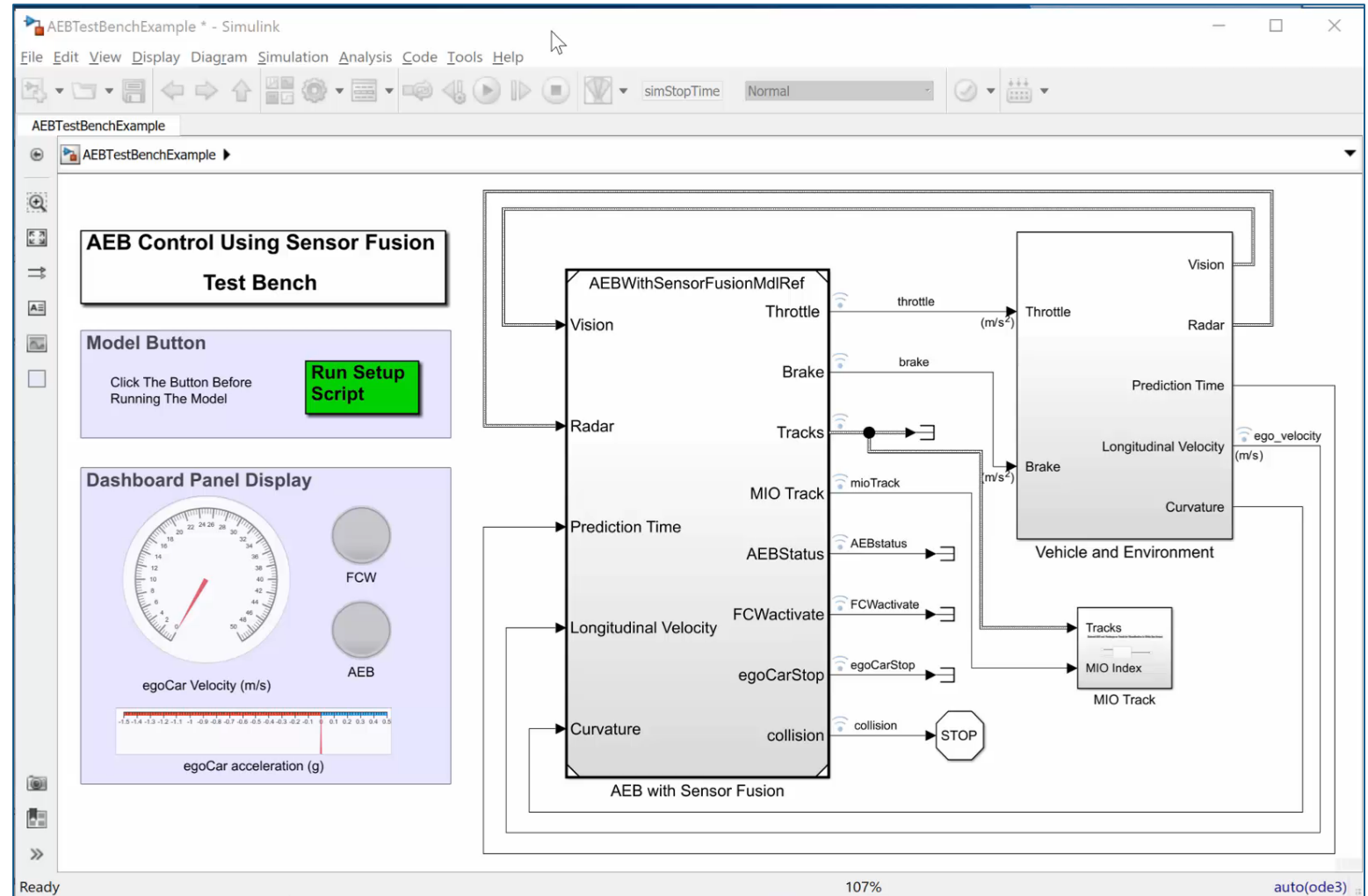
- Specify driving scenario
- Design AEB logic
- Integrate sensor fusion
- Simulate system
- Generate C/C++ code
- Test with software in the loop (SIL) simulation

Automated Driving Toolbox™

Stateflow®

Embedded Coder®

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

Test Manager

TESTS

FILE EDIT RUN RESULTS ENVIRONMENT RESOURCES

Test Browser Results and Artifacts

Filter tests by name or tags, e.g. tags: test

LaneFollowingTestScenarios

Scenarios

- ACC_ISO_TargetDiscriminationTest
- ACC_ISO_AutoRetargetTest
- ACC_ISO_CurveTest
- ACC_StopnGo
- LFACC_DoubleCurve_DecelTarget
- LFACC_DoubleCurve_AutoRetarget
- LFACC_DoubleCurve_StopnGo
- LFACC_Curve_CutInOut
- LFACC_Curve_CutInOut_TooClose

Scenarios

ACC_ISO_TargetDiscriminationT...

DESCRIPTION*

REQUIREMENTS*

scenarioId #1: ACC_ISO_TargetDiscriminationTest (LaneFollowingTestRequirements#1)

SYSTEM UNDER TEST*

Model: LaneFollowingTestBenchExample

TEST HARNESS

SIMULATION SETTINGS OVERRIDES*

PARAMETER OVERRIDES

CALLBACKS*

PRE-LOAD

POST-LOAD*

Runs after the model loads and the model PostLoadFcn callback

```
1 scenarioId = 1;
2 helperLFSetup;
```

CLEANUP*

Runs after simulations and all model callbacks

```
1 plotLFResults(sltest_simout.logout);
```

Requirements link

Simulink Model

Define scenario ID and data initialization

Plot the results

PROPERTY	VALUE
Name	ACC_ISO_TargetDiscri...
Type	Simulation Test
Model	LaneFollowingTestBenchEx...
Simulation Mode	Normal
Location	C:\02_ADST2018b\Demos\...
Enabled	<input checked="" type="checkbox"/>
Hierarchy	LaneFollowingTestScenario...
Tags	Type comma or space separa...

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

The image displays the MATLAB R2019a interface. The main window shows a script titled "PlaybackScenarioExample.mlx" with the following content:

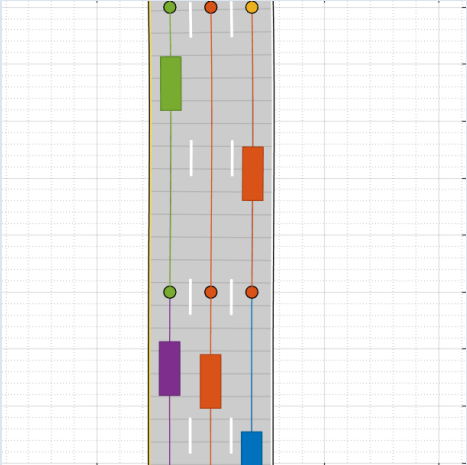
```
function [egoData] = helperGetEgoData(egoFile)
%Read the ego vehicle data from text file
fileID = fopen(egoFile);
content = textscan(fileID, '%f %f %f');
fields = {'lat', 'lon', 'Time'};
egoData = cell2struct(content, fields, 2);
fclose(fileID);
end
```

The script is being executed, and the Command Window shows the status "Busy".

On the right side of the interface, there is a yellow banner that reads "Simulate synthesized scenario". Below this banner, there are two visualizations:

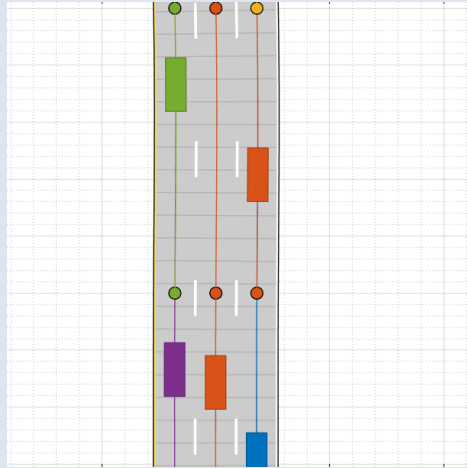
- A 2D plot showing a road network with a grid. The X-axis is labeled "X (m)" and ranges from 270 to 320. The Y-axis is labeled "Y (m)" and ranges from 80 to 120. Several colored dots (red, orange, green, blue, purple) are scattered across the road network, representing vehicle positions.
- A 3D visualization of a road scene. A large blue rectangular object is in the foreground, representing a vehicle. Other smaller colored rectangular objects are visible in the background, representing other vehicles or objects in the scene.

How can I design with virtual scenarios?

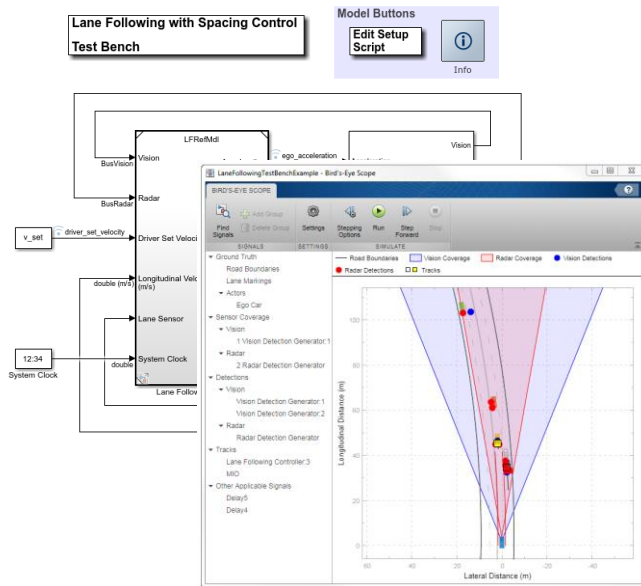
Scenes	Driving Scenarios (cuboid) 
Testing	Controls Controls + sensor fusion
Authoring	Driving Scenario Designer App drivingScenario programmatic API
Sensing	Probabilistic radar detections Probabilistic vision detections Probabilistic lane detections

How can I design with virtual scenarios?

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

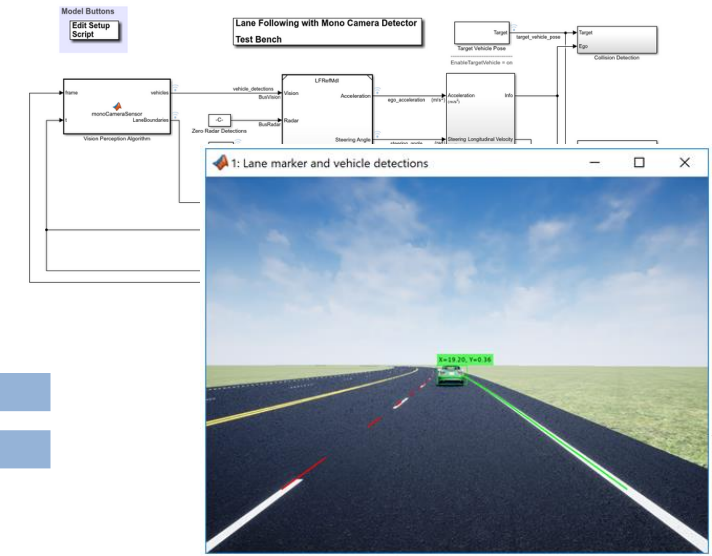


Simulate controls and perception systems



```

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

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

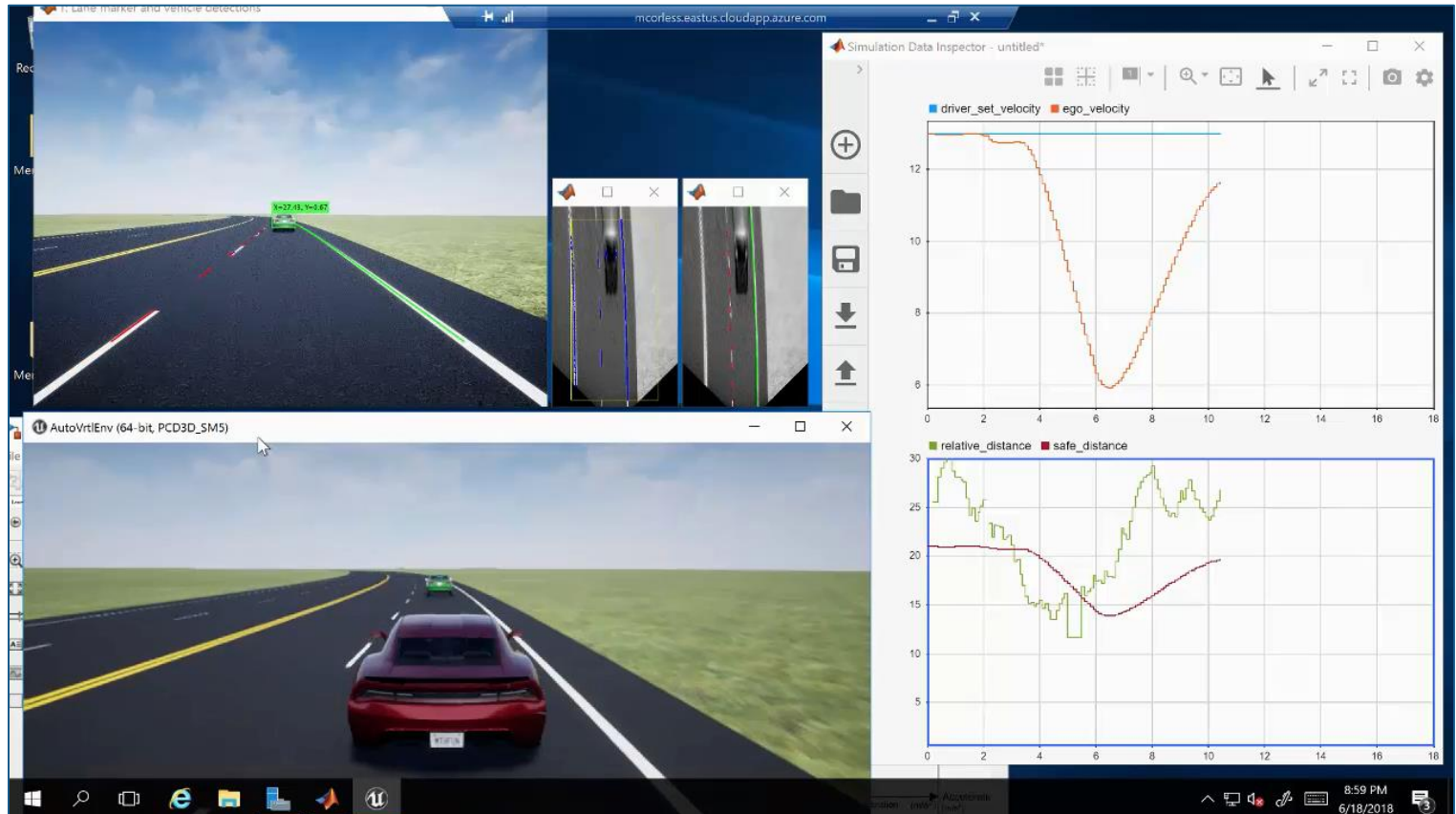
Lane-Following Control with Monocular Camera Perception

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

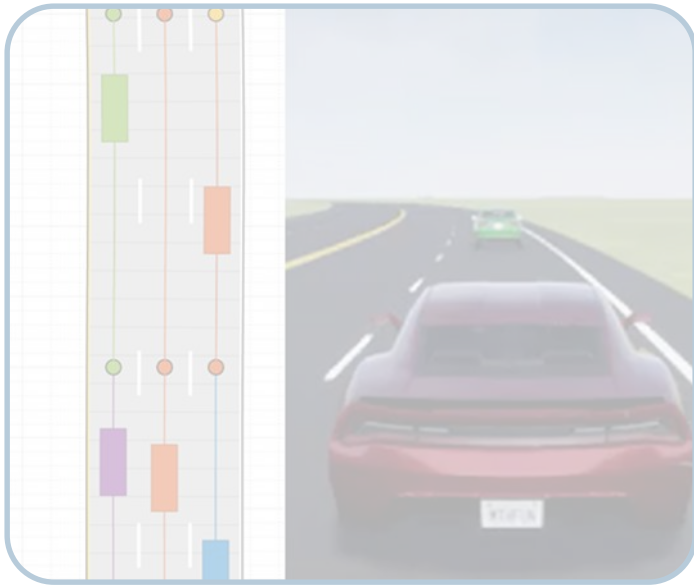
Model Predictive Control Toolbox™

Automated Driving Toolbox™

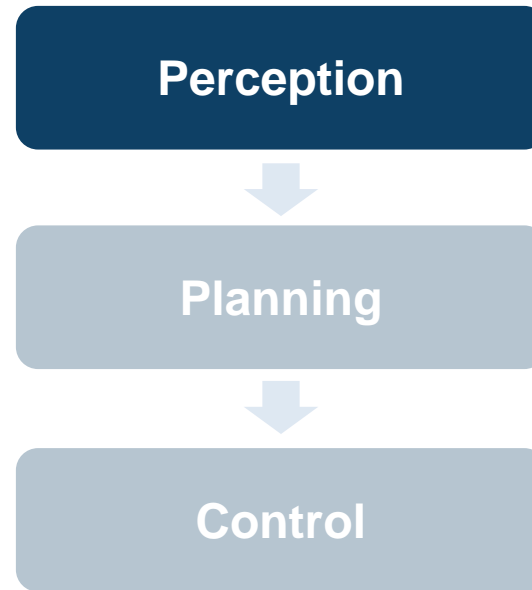
Vehicle Dynamics Blockset™



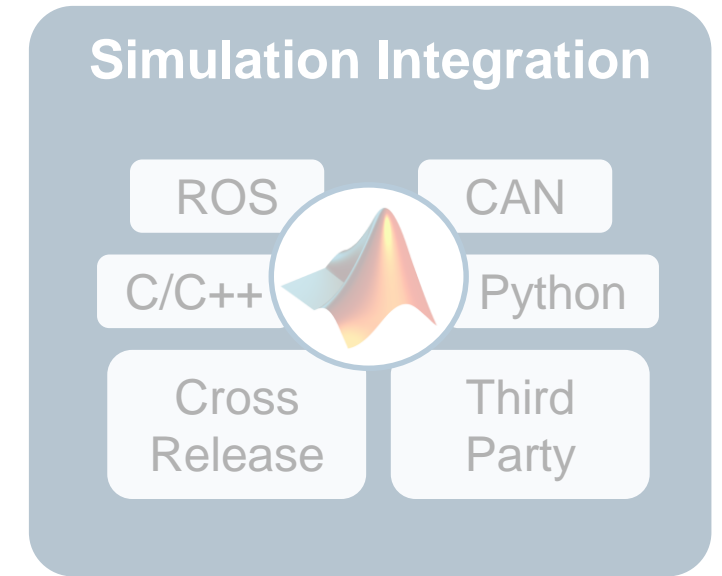
Some common questions from automated driving engineers



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 multi-object trackers

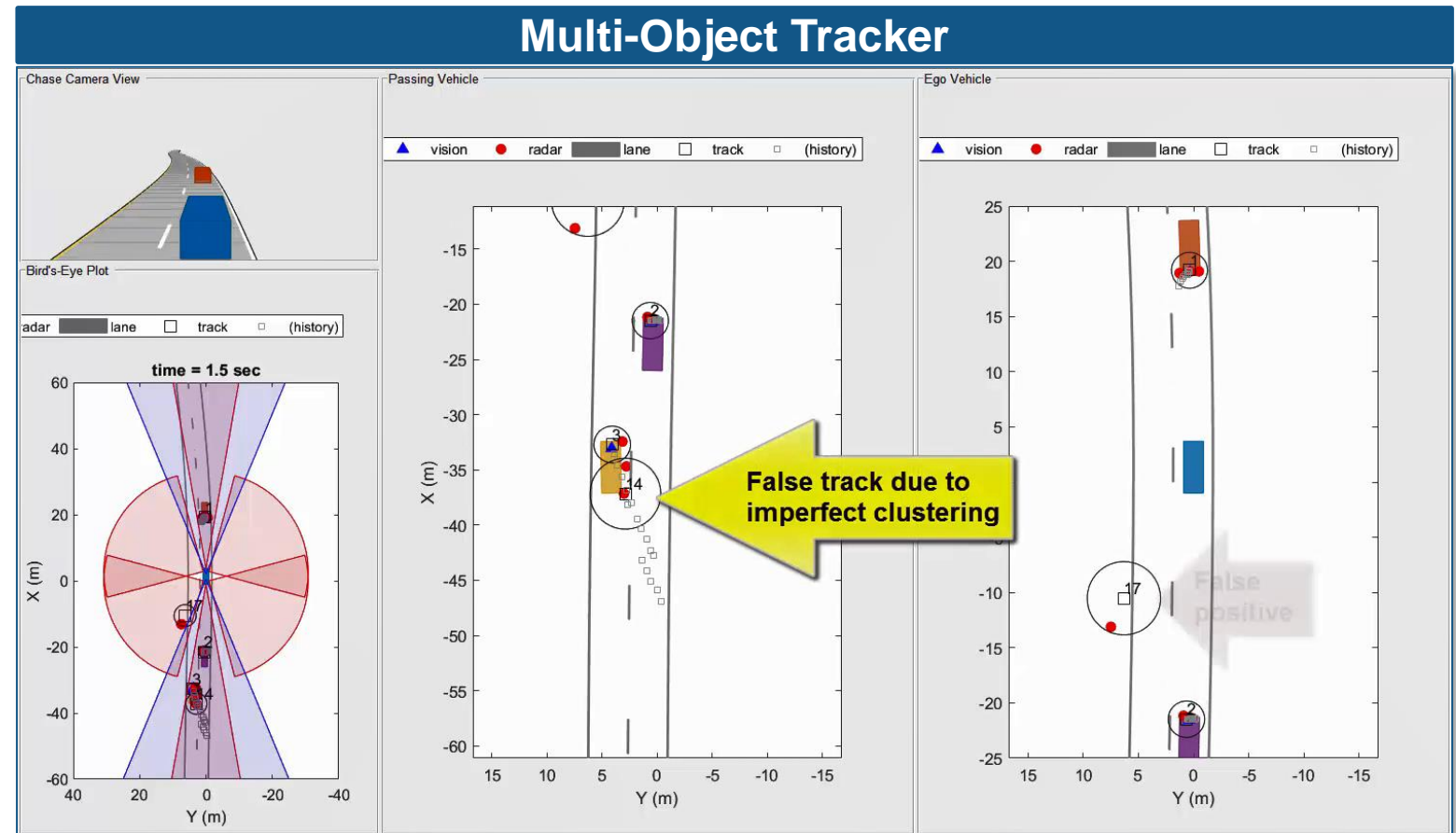
Extended Object Tracking

- 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

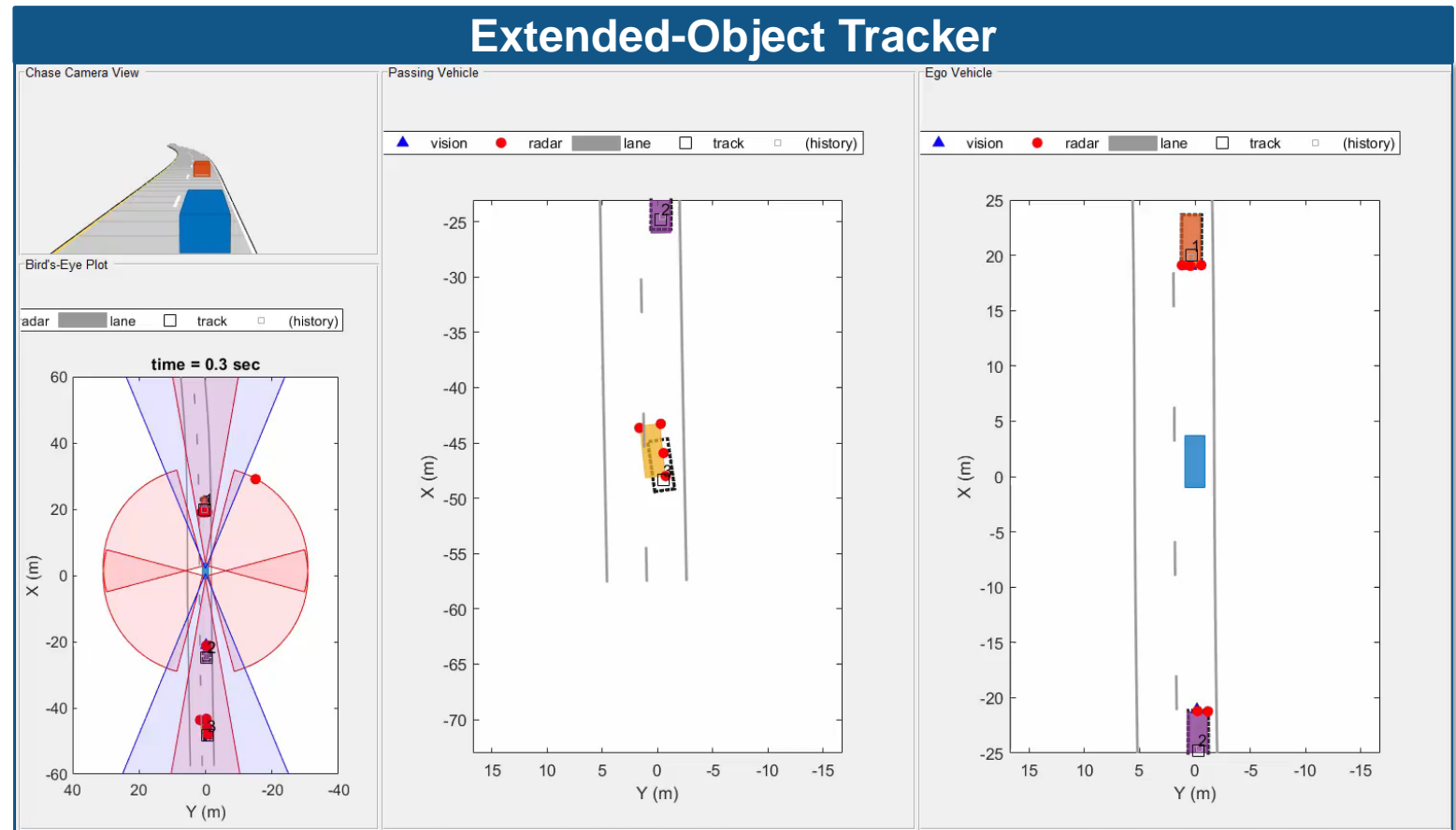
Extended Object Tracking

- 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

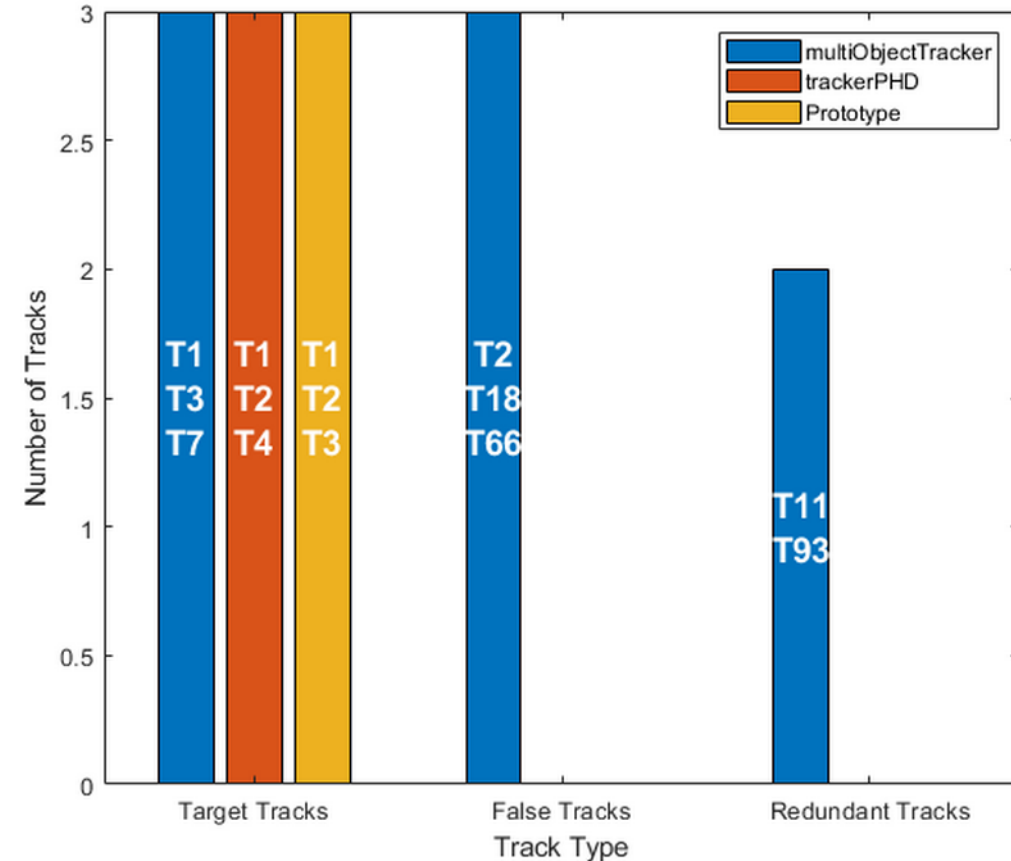
Extended Object Tracking

- 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

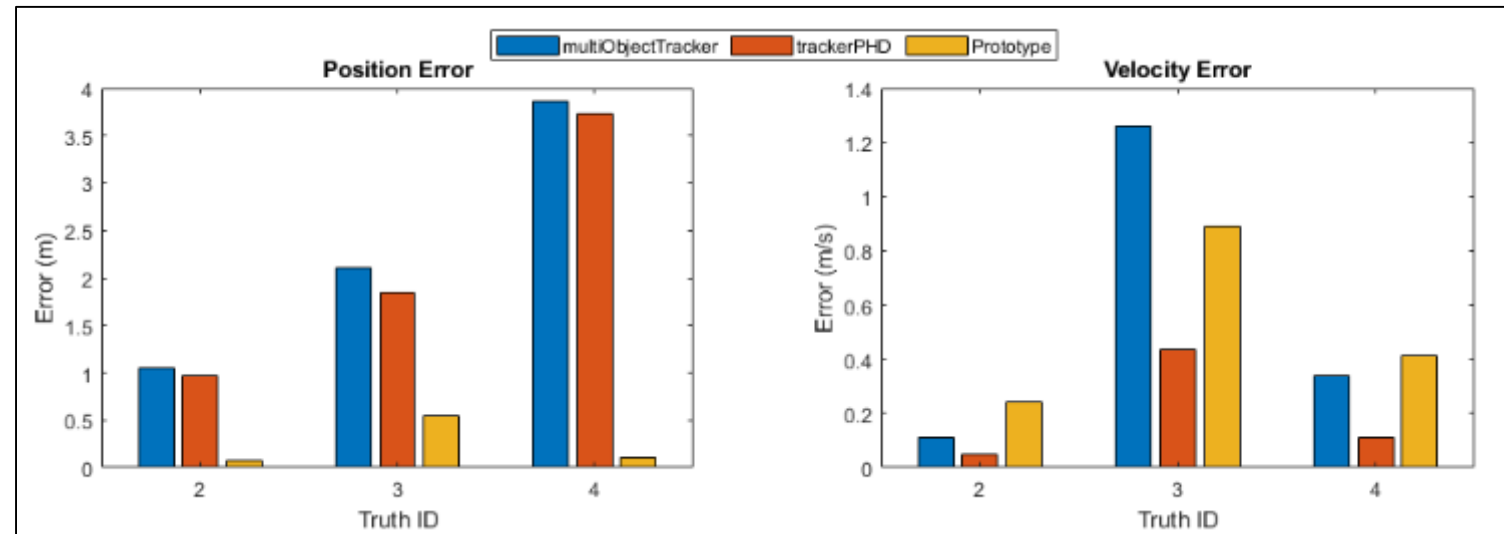
Extended Object Tracking

- 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

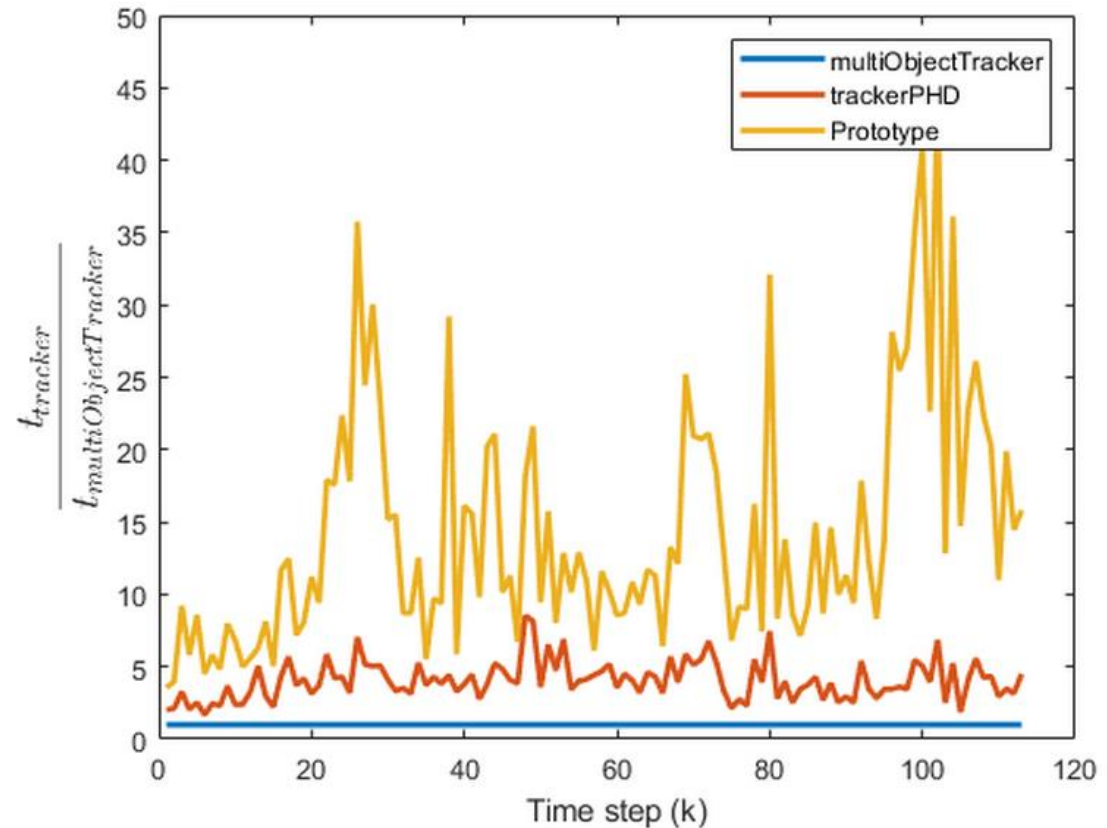
Extended Object Tracking

- 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

Design detector for lidar point cloud data

Track Vehicles Using Lidar: From Point Cloud to Track List

- Design 3-D bounding box detector
- Design tracker (target state and measurement models)
- Generate C/C++ code for detector and tracker

*Sensor Fusion and Tracking
Toolbox™*

Computer Vision Toolbox™

R2019a

The screenshot shows the MATLAB R2019a environment. The Editor window displays the code for the `HelperBoundingBoxDetector.m` file. The code includes comments for processing a point cloud: cropping, removing the ground plane, forming clusters, and getting bounding boxes. The Command Window shows the execution of `pcshow(pcObstacles)`. A yellow callout box points to a 3D visualization of the point cloud with bounding boxes, listing the parameters for each: X-Center, Y-Center, Z-Center, Length, Width, and Height.

```

methods (Access = protected)
function [bboxDets, obstacleI] = HelperBoundingBoxDetector(pcObstacles)
% Crop point cloud
[pcSurvived, survivedIndices] = cropPointCloud(pcObstacles);
% Remove ground plane
[pcObstacles, obstacleIndices] = removeGroundPlane(pcSurvived);
% Form clusters and get bounding boxes
detBBBoxes = getBoundingBoxes(pcObstacles);
% Assemble bounding boxes
bboxDets = detBBBoxes;
end
end
end

```

Command Window:

```

K>> pcshow(pcObstacles)
fx K>>

```

Output:

```

detBBBoxes: 6x10 single matrix =
Columns 1 through 4
12.8921 -22.6758 -46.8280 -21.1414
-3.9148 -3.7233 -3.5872 0.0260
0.7299 0.6966 -0.6705 0.7558
2.8747 2.6390 0.0816 2.2517
1.7510 1.7391 0.8562 1.6446
1.0838 0.5916 0.0068 0.5503

```

Callout box labels: X-Center, Y-Center, Z-Center, Length, Width, Height.

Design tracker for lidar point cloud data

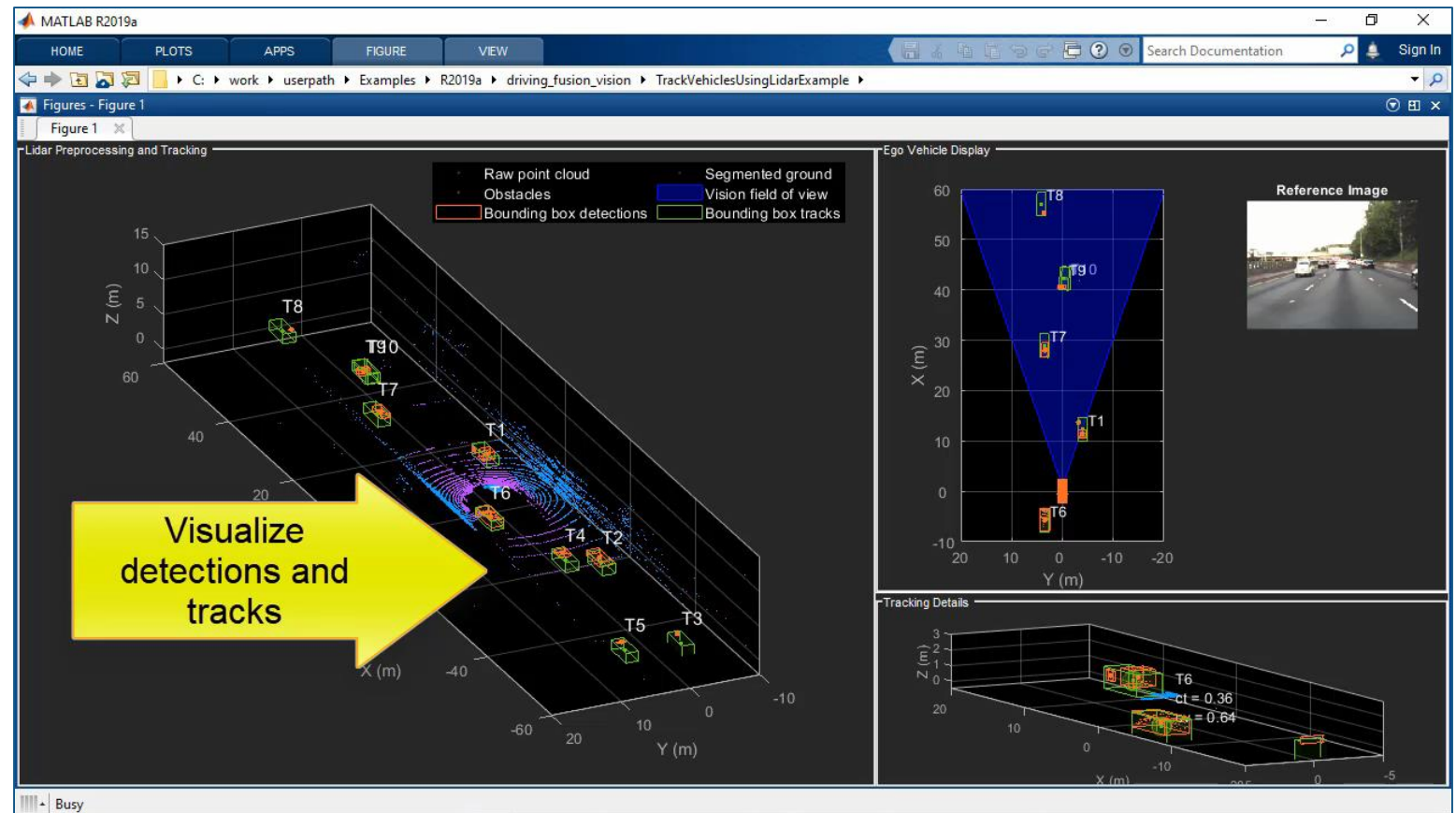
Track Vehicles Using Lidar: From Point Cloud to Track List

- Design 3-D bounding box detector
- Design tracker (target state and measurement models)
- Generate C/C++ code for detector and tracker

*Sensor Fusion and Tracking
Toolbox™*

Computer Vision Toolbox™

R2019a



Generate C/C++ code for lidar detector and tracker

Track Vehicles Using Lidar: From Point Cloud to Track List

- Design 3-D bounding box detector
- Design tracker (target state and measurement models)
- Generate C/C++ code for detector and tracker

*Sensor Fusion and Tracking
Toolbox™*

Computer Vision Toolbox™

R2019a

MATLAB Coder Report Viewer - C:\work\userpath\Examples\R2019a\driving_fusion_vision\TrackVehiclesUsingLidarExample\codegen\lib\mexLidarTracker\html\report.mldat

REPORT

Back Forward Find Trace Code Edit In MATLAB Package Code Export Report Information

NAVIGATE TRACE EDIT SHARE

MATLAB SOURCE

Function List Call Tree

- mexLidarTracker.m
 - fx mexLidarTracker
- HelperBoundingBoxDetector.m
 - fx HelperBoundingBoxDetector
 - fx assembleDetections
 - fx cropPointCloud
 - fx getBoundingBoxes
 - fx removeGroundPlane
 - fx stepImpl
- helperCalcDetectability.m
 - fx helperCalcDetectability

GENERATED CODE

Source Files

- AssignmentCostCalculator
- AssignmentCostCalculator.h
- ExtendedKalmanFilter.cpp
- ExtendedKalmanFilter.h
- HelperBoundingBoxDetector
- HelperBoundingBoxDetector.h
- KFDistance.cpp
- KFDistance.h
- KalmanLikelihood.cpp
- KalmanLikelihood.h
- ObjectTrack.cpp

Code

```

mexLidarTracker.m
5 persistent detectorModel tracker detectabilityTracks input current
6
7 if isempty(detectorModel) || isempty(tracker) || isempty(dete
8
9 % A bounding box detector model.
10 detectorModel = HelperBoundingBoxDetector(...
11     'XLimits', [-50 75],... % min
12     'YLimits', [-5 5],... % min
13     'ZLimits', [-2 5],... % min
14     'SegmentationMinDistance', 1.6,... % min
15     'MinDetectionsPerCluster', 1,... % min
16     'MeasurementNoise', eye(6),... % mea
17     'GroundMaxDistance', 0.3); % max
18
19 assignmentGate = [10 100]; % Assignment threshold;
20 confThreshold = [7 10]; % Confirmation threshold for h
21 delThreshold = [8 10]; % Deletion threshold for histo
22 Kc = 1e-5; % False-alarm rate per unit vo
23
356 // % measurement noise in detection
357 // 'mexLidarTracker:10' detectorModel = He
358 // 'mexLidarTracker:11'
359 // 'mexLidarTracker:12'
360 // 'mexLidarTracker:13'
361 // 'mexLidarTracker:14'
362 // 'mexLidarTracker:15'
363 // 'mexLidarTracker:16'
364 // 'mexLidarTracker:17'
365 detectorModel.GroundReferenceVector[0] =
366 detectorModel.GroundReferenceVector[1] =
367 detectorModel.GroundReferenceVector[2] =
368 detectorModel.GroundMaxAngularDistance =
369 detectorModel.MaxZDistanceCluster = 3.0;
370 detectorModel.MinZDistanceCluster = -3.0;
371 detectorModel.EgoVehicleRadius = 3.0;
372 detectorModel.isInitialized = 0;
373
374
SUMMARY ALL MESSAGES (0) BUILD LOGS CODE INSIGHTS (0) VARIABLES
Code generation successful
Generated on: 24-Mar-2019 14:59:22
Build type: Static Library
Toolchain: Microsoft Visual C++ 2017 v15.0 | nmake (64-bit Windows)
Build Configuration: Faster Builds
    
```

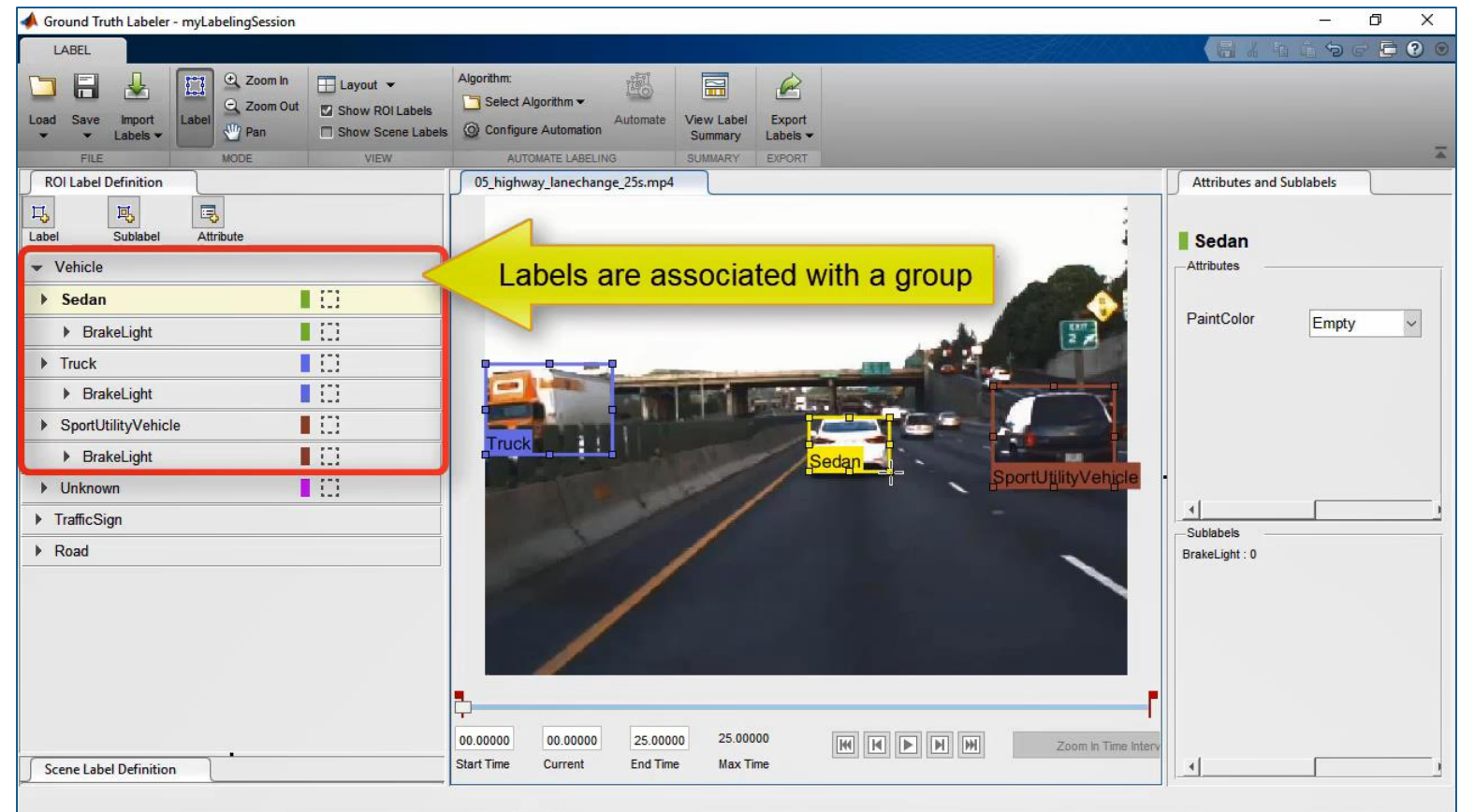

Create region of interest labels and groups

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**



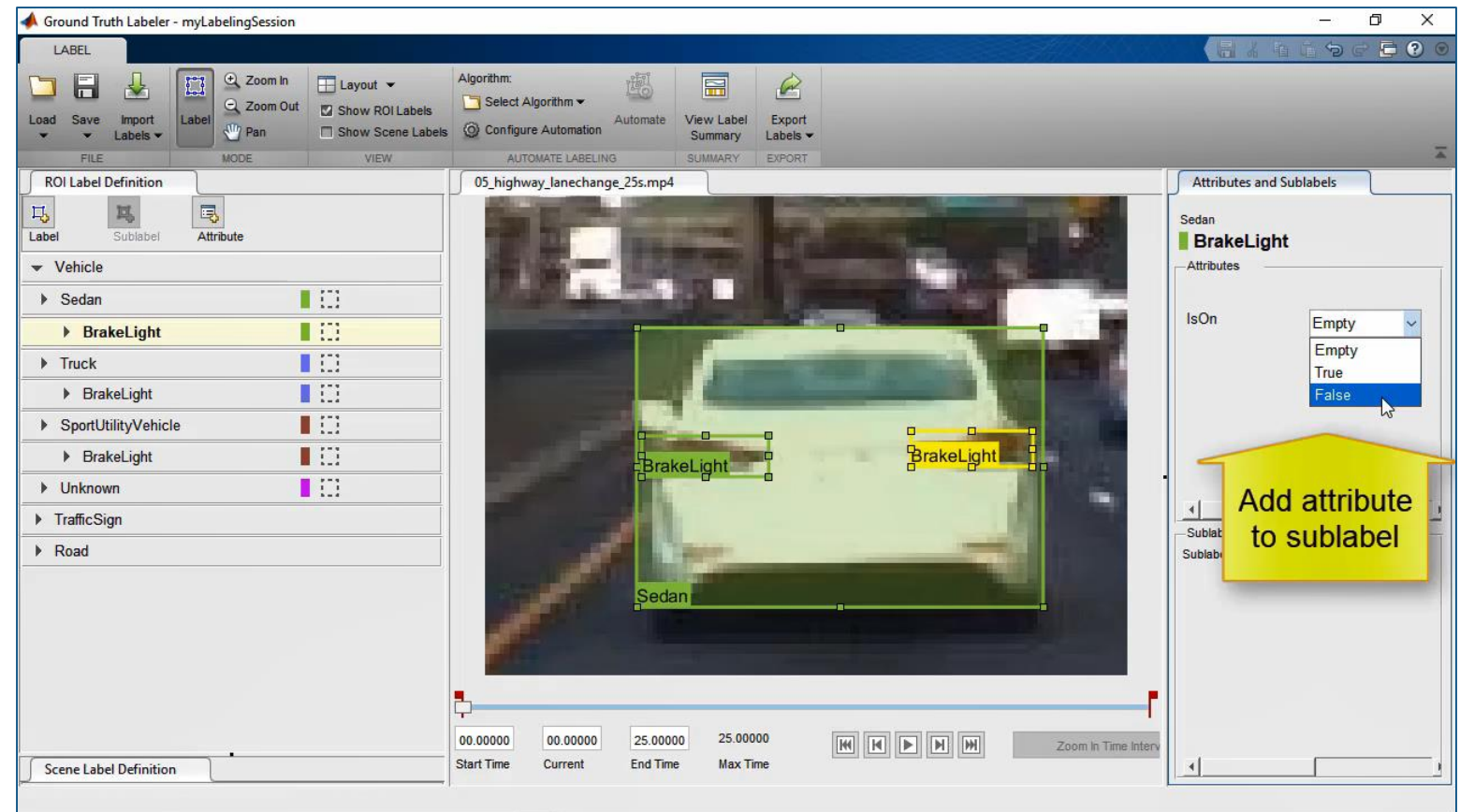
Create sublabels and add attributes

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**



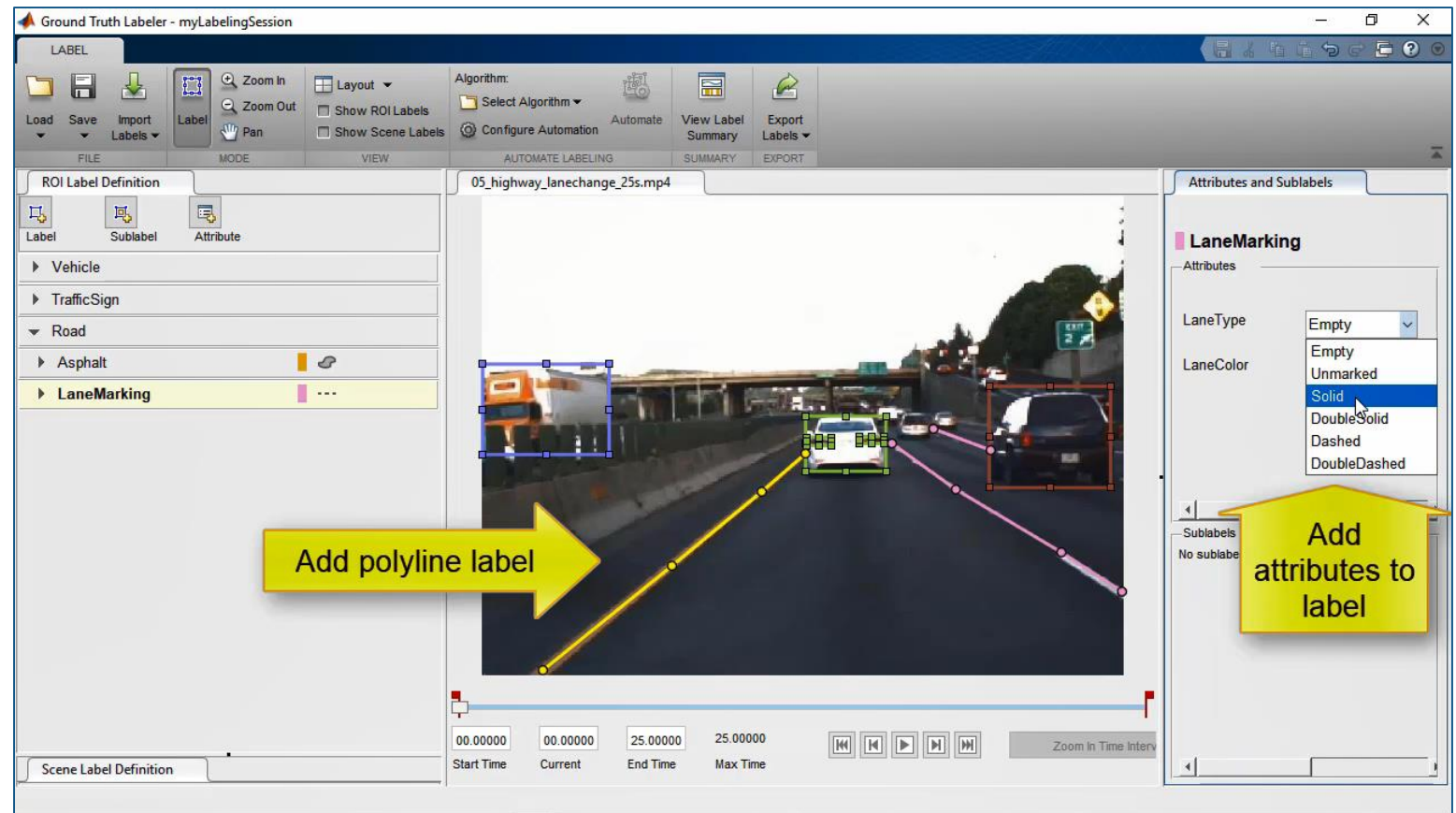
Create polyline labels and add attributes

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**



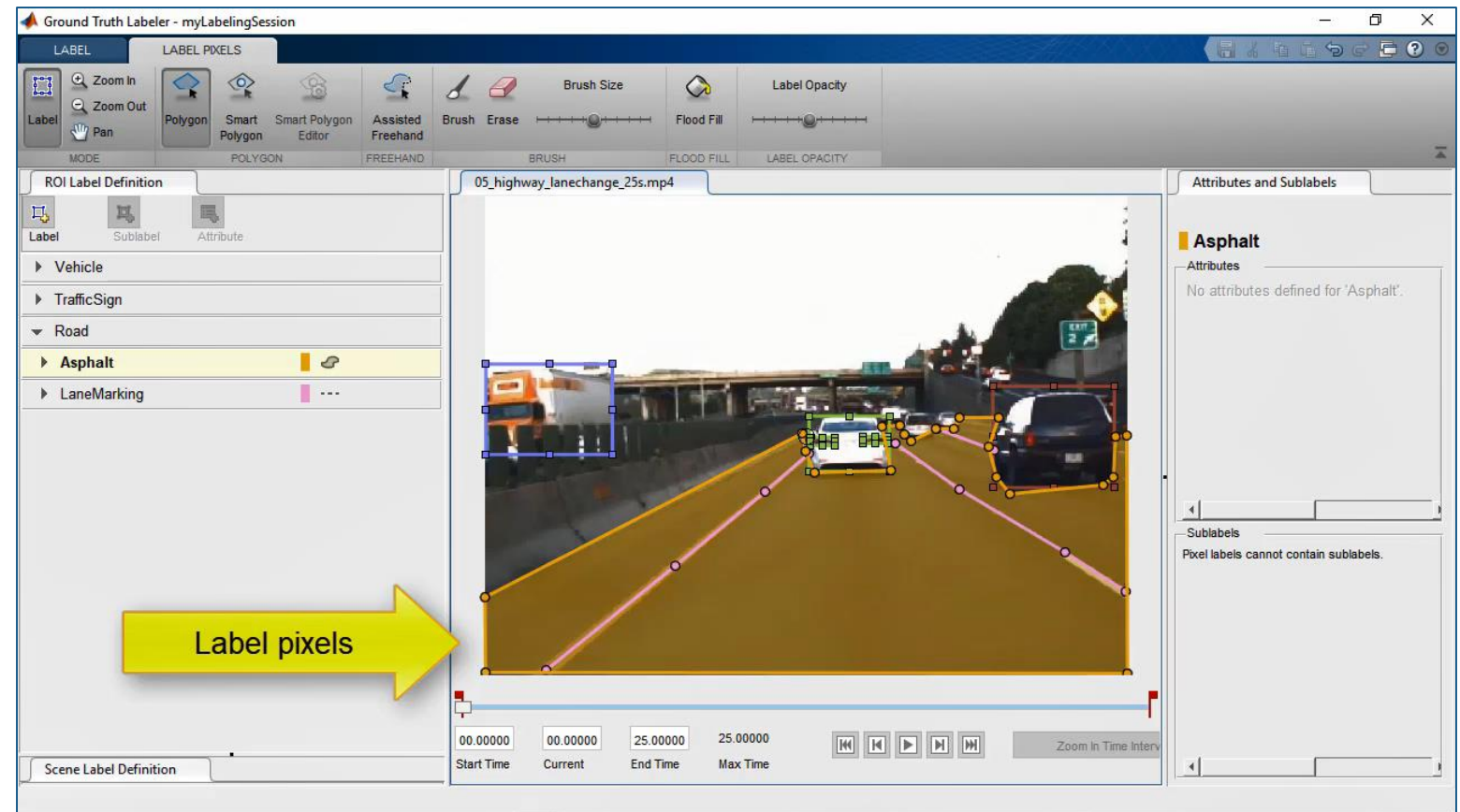
Create pixel labels

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**



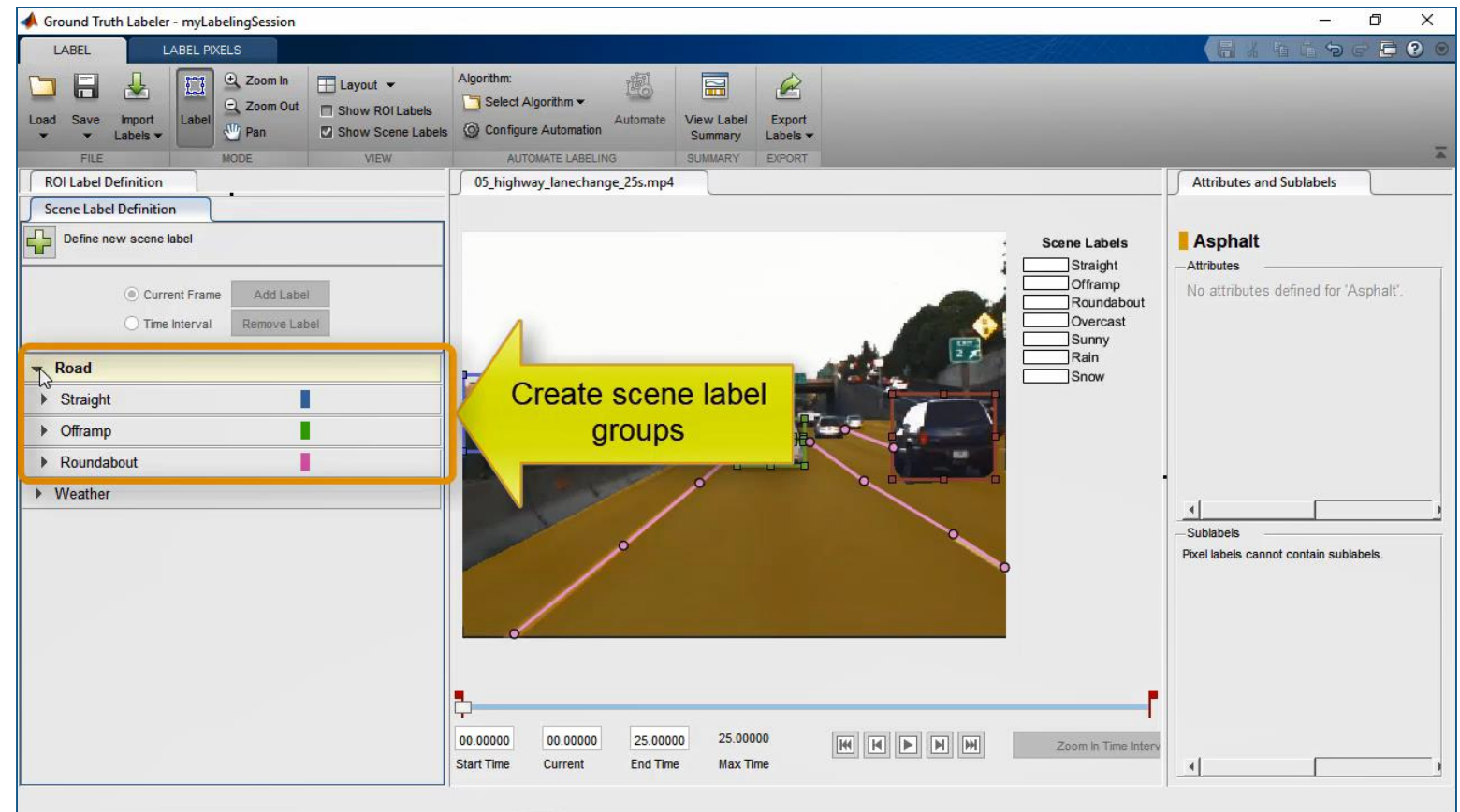
Create scene labels and groups

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**



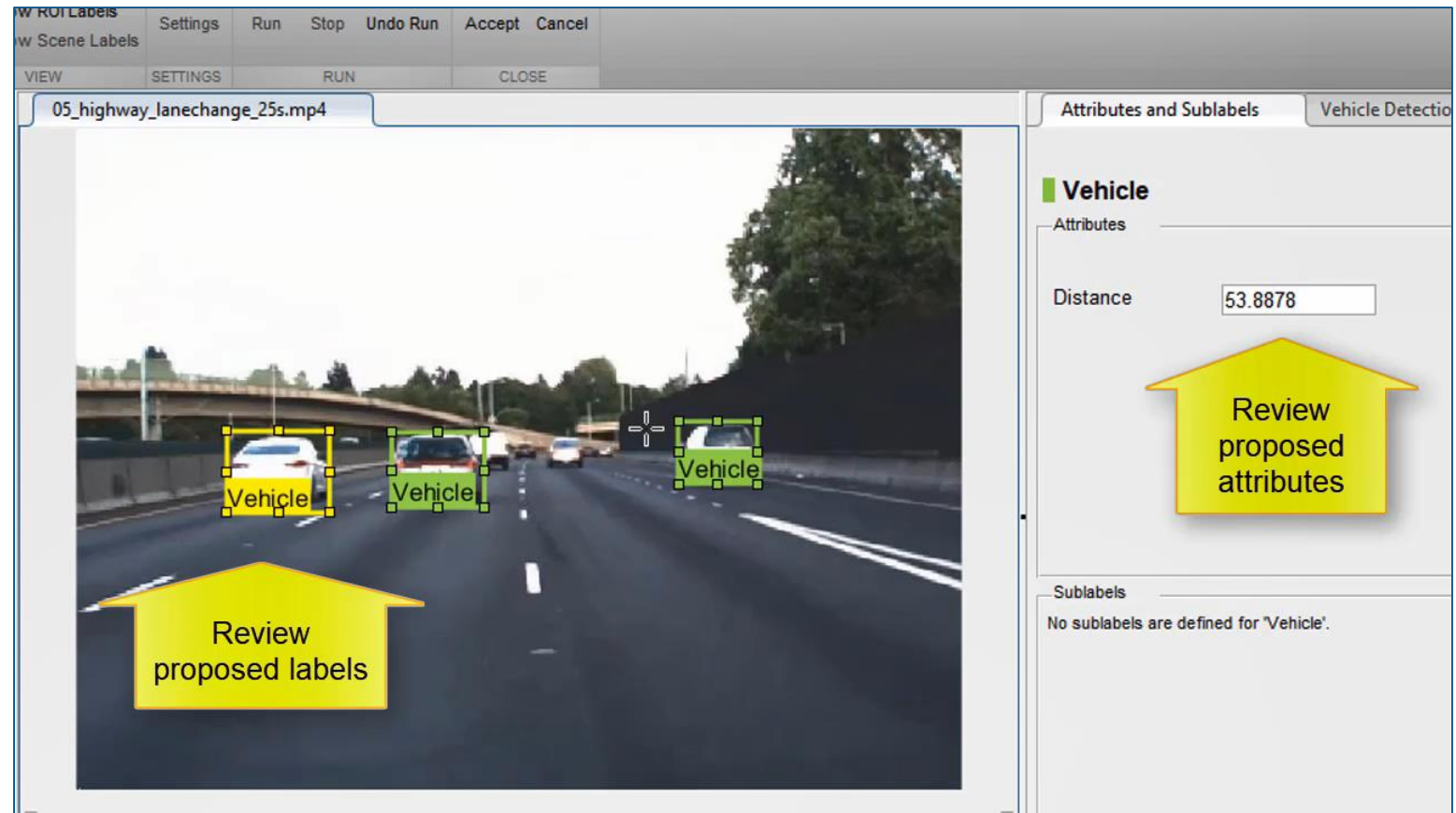
Import custom automation algorithms

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

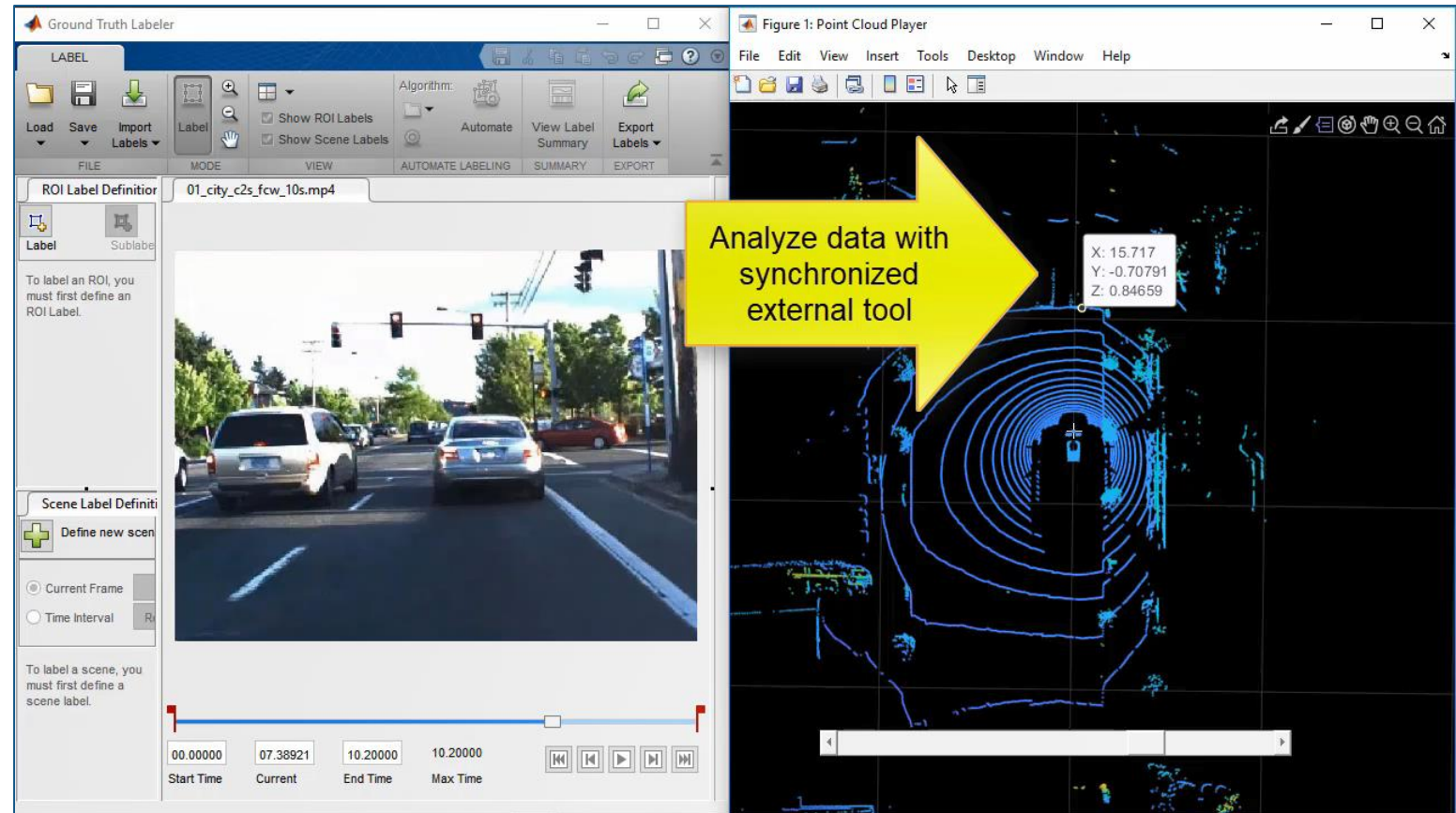


Add custom visualizations for multi-sensor data

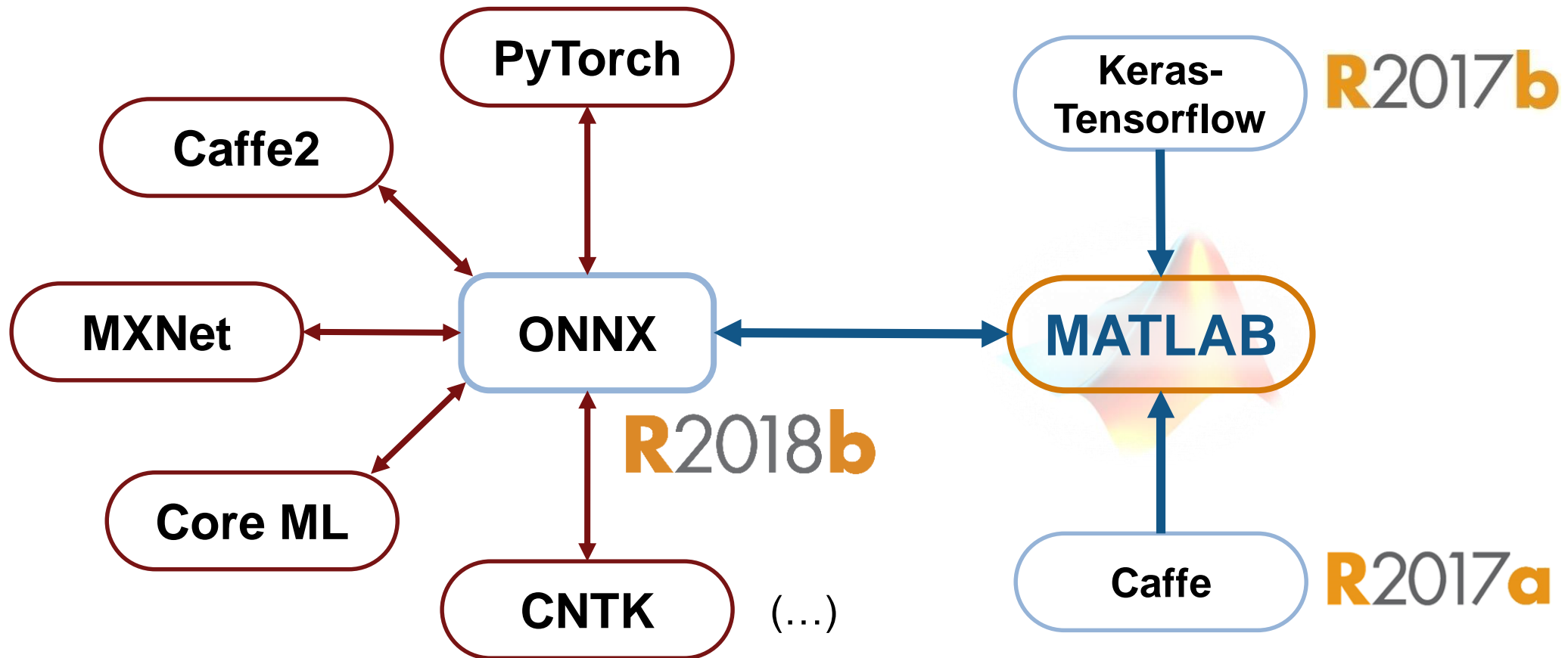
Connect Lidar Display to Ground Truth Labeler

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

Automated Driving Toolbox™
R2017a



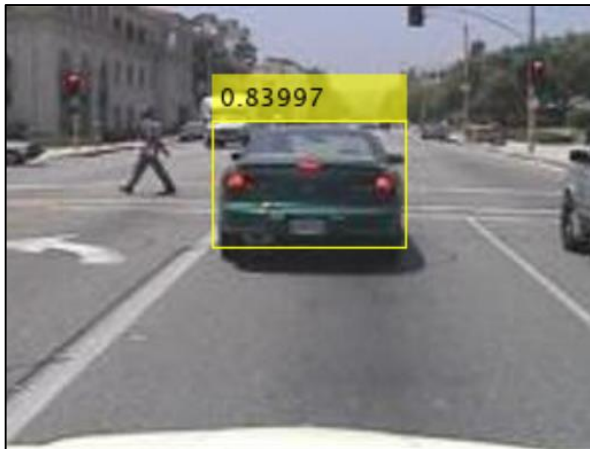
Interoperate with neural network frameworks



Open Neural Network Exchange

Design camera, lidar, and radar perception algorithms

Detect vehicle with camera

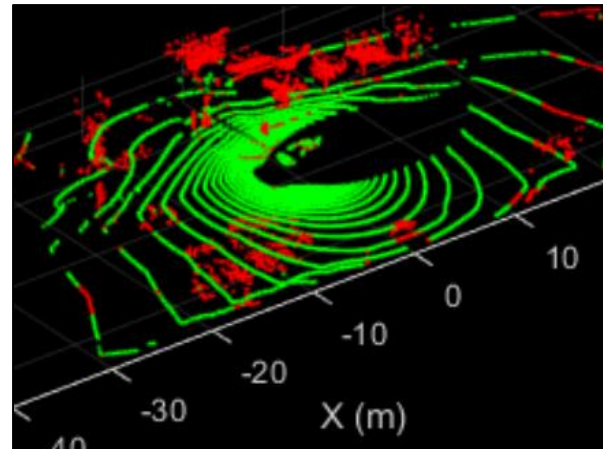


[Object Detection Using YOLO v2 Deep Learning](#)

*Computer Vision Toolbox™
Deep Learning Toolbox™*

R2019a

Detect ground with lidar

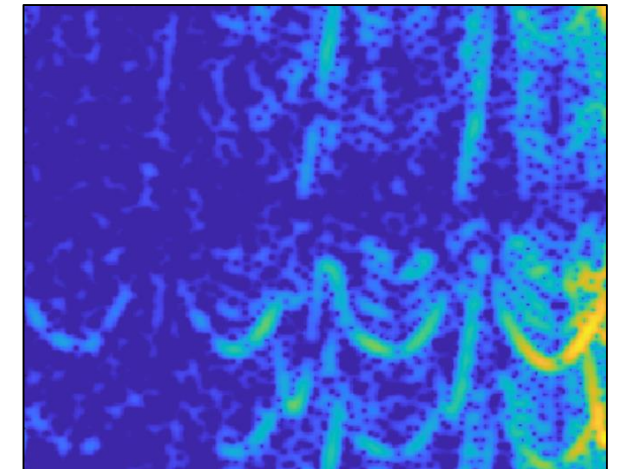


[Segment Ground Points from Organized Lidar Data](#)

Computer Vision Toolbox™

R2018b

Detect pedestrian with radar



[Introduction to Micro-Doppler Effects](#)

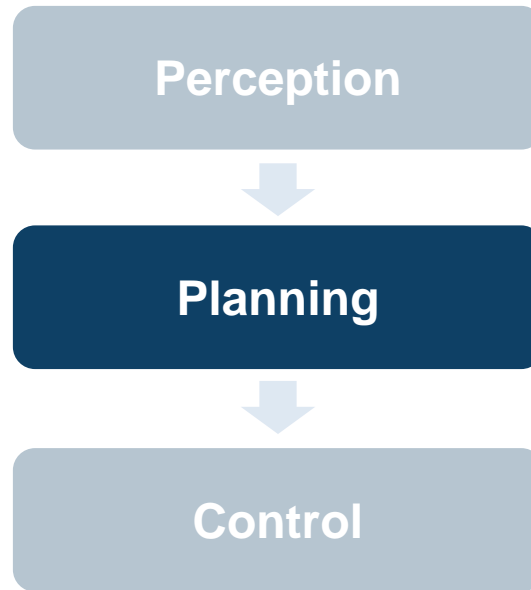
Phased Array System Toolbox™

R2019a

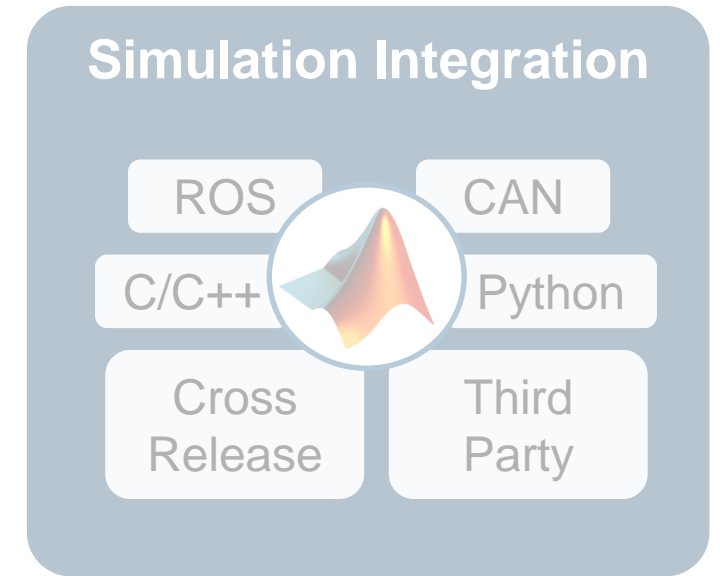
Some common questions from automated driving engineers



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?

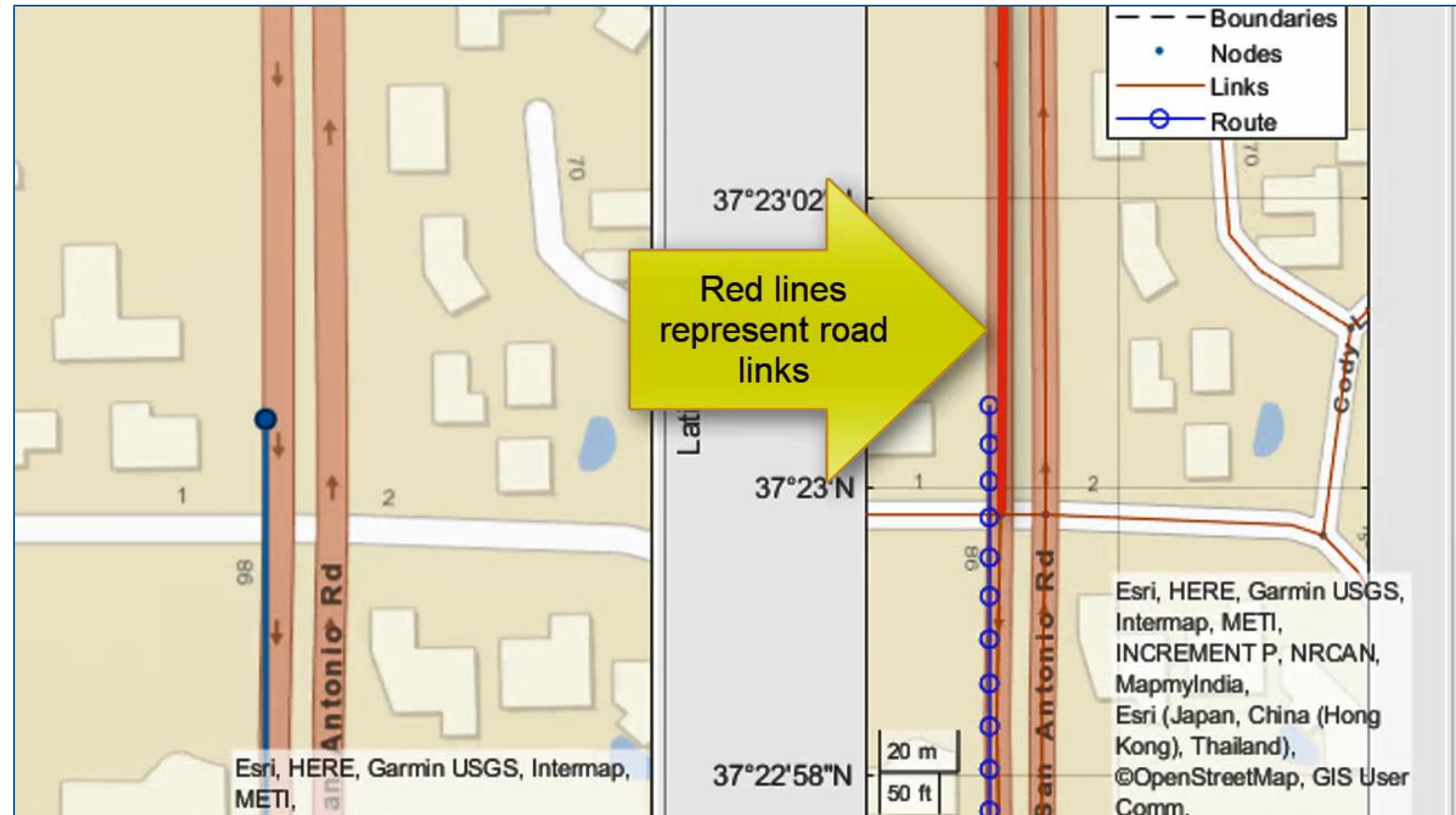
Read road and speed attributes from HERE HD Live Map data

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



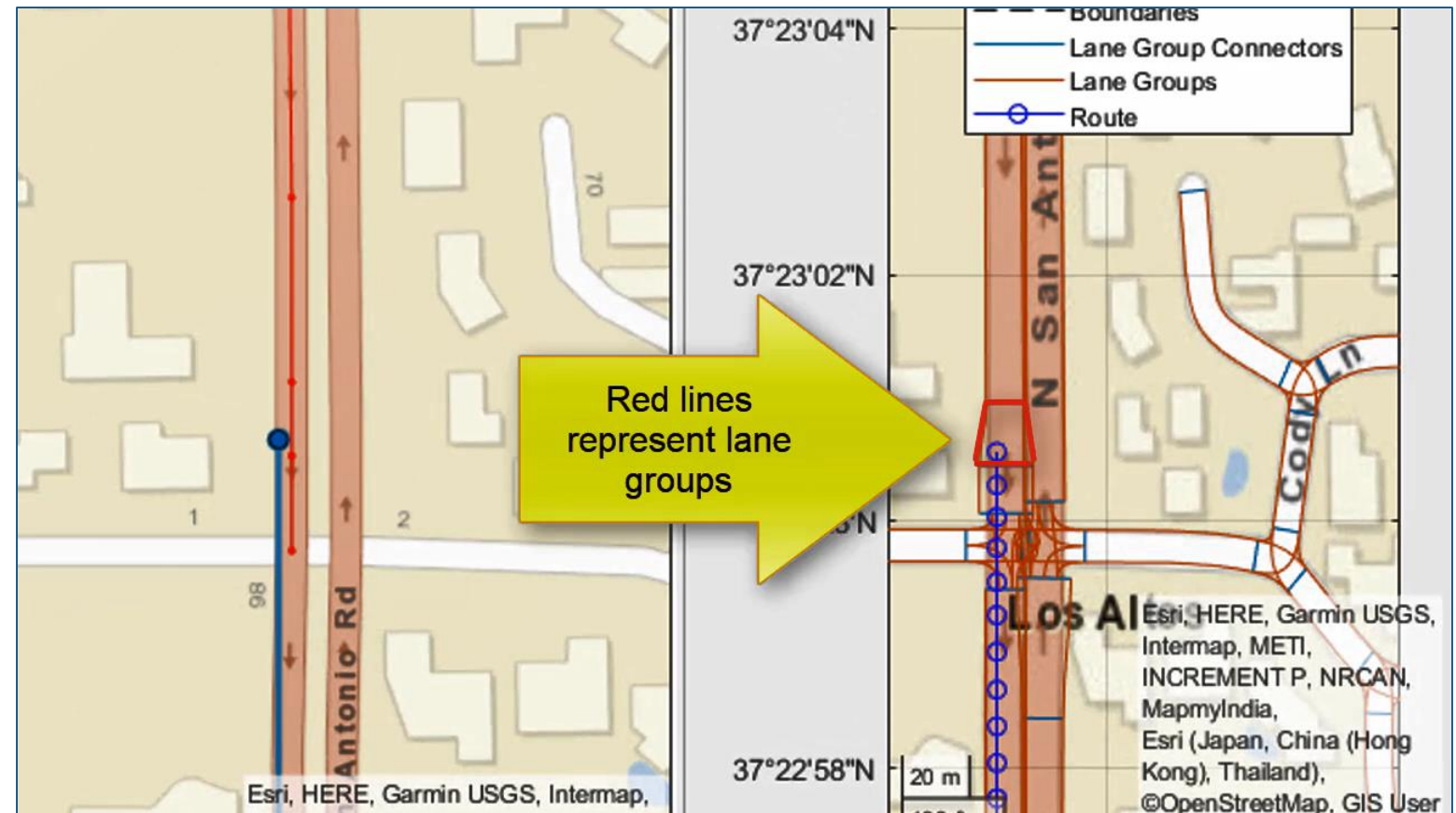
Read lane attributes from HERE HD Live Map data

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



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

The figure window displays a camera view of a road with a traffic jam, a map overlay of the same road, and a data panel with a timestamp of 22:11:25 and a speed limit of 35. The map overlay shows lane configurations and boundaries.

Script Editor Content (lines 286-303):

```

286 % Visu
287 % The m
288 % lane
289 % coord
290 % link
291 % confi
292 %
293 % The
294 % |<mat
295 % Help
296 % a rec
297 % HD Li
298 hdlmUI :
299
300
301 % Synch
302 synchro:
303 videoRe
  
```

Figure Window Data Panel:

- Timestamp: 22:11:25
- Speed Limit: 35
- Lane Types and Boundaries: [Diagram showing lane configurations]

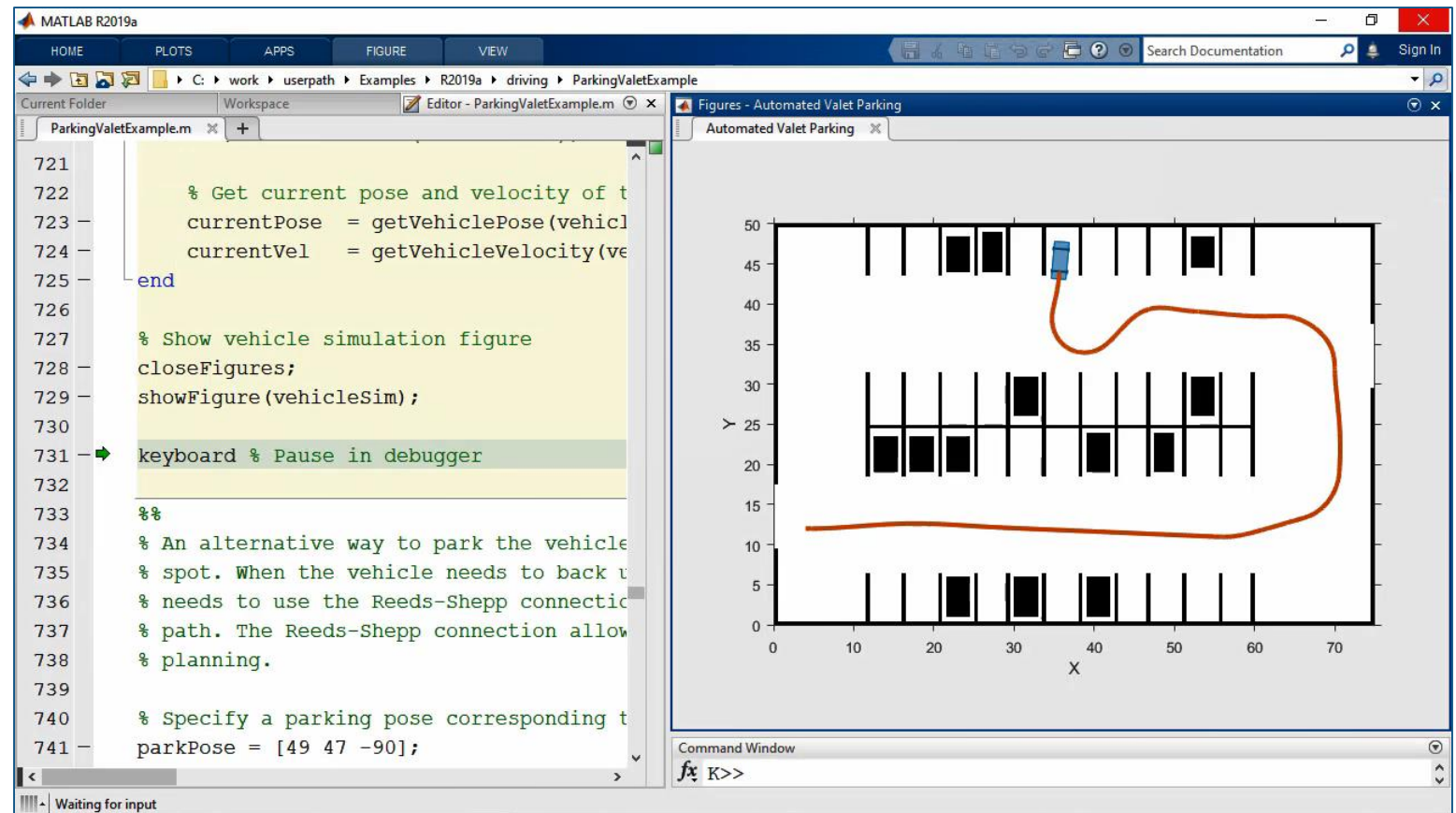
Design path planner

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



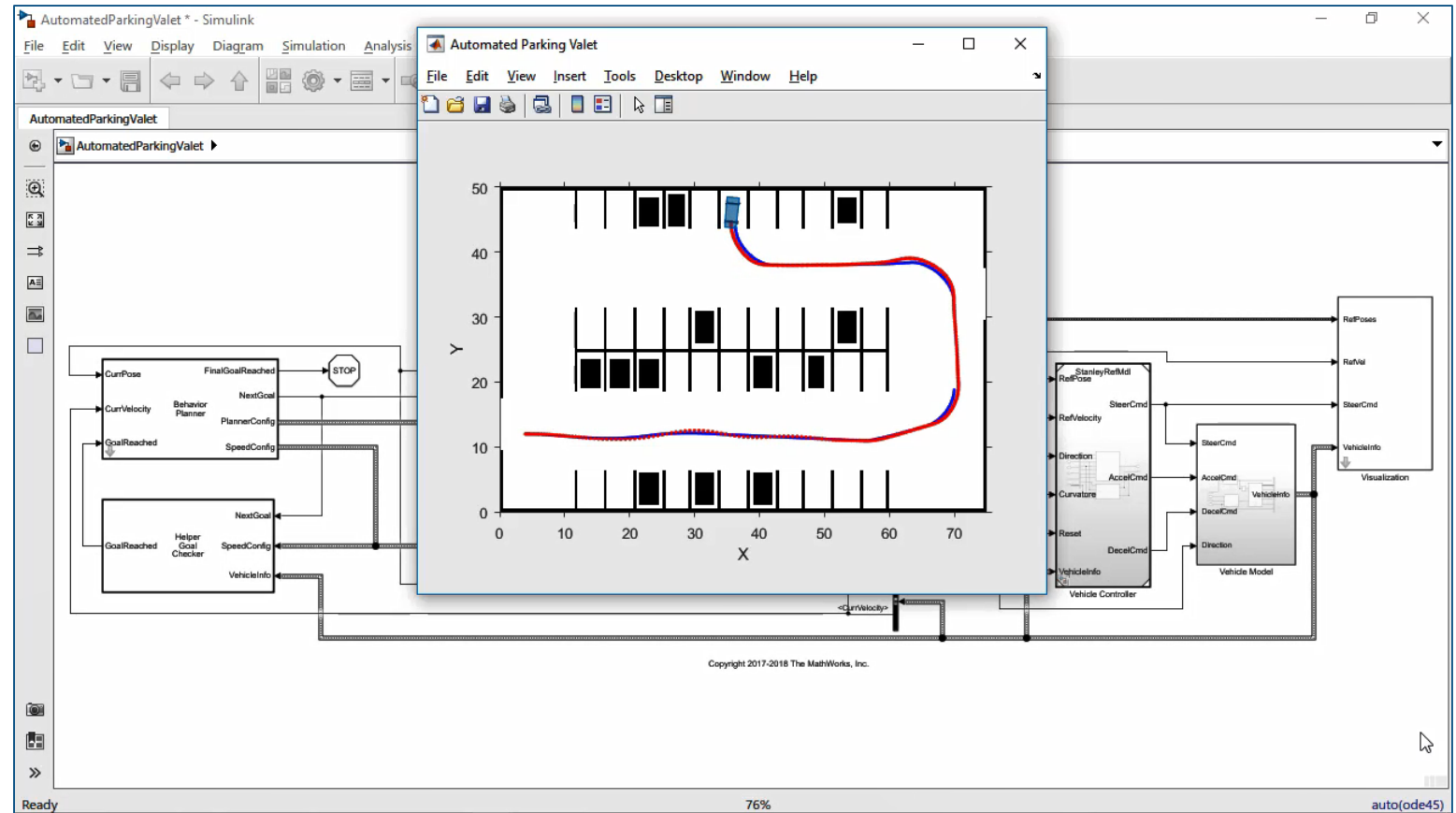
Design path planner and controller

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

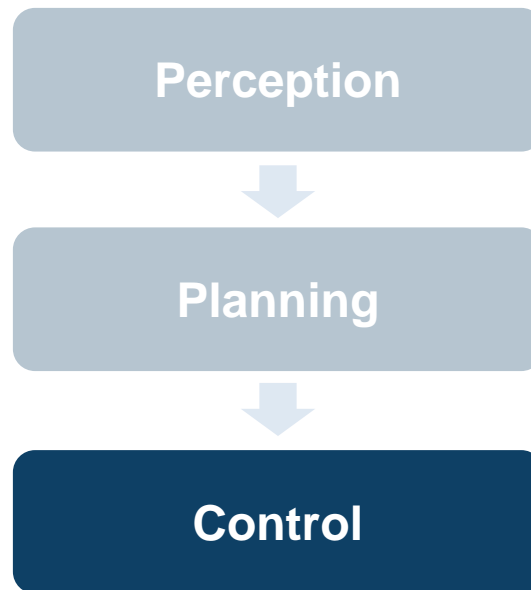
R2019a

```
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 inport: '<Root>/Costmap' set method
203 void setCostmap(costmapBus localArgInput);
204
205 // Root inport: '<Root>/GoalPose' set method
206 void setGoalPose(real_T localArgInput[3]);
207
```

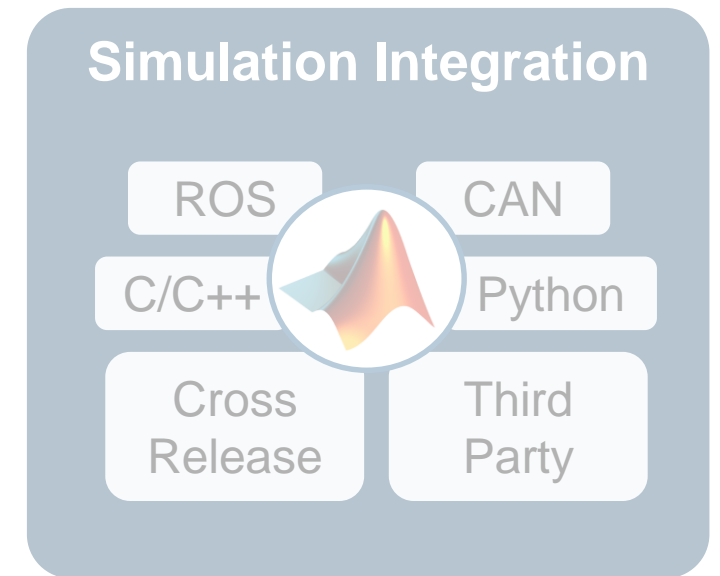
Some common questions from automated driving engineers



How can I
synthesize scenarios
to test my designs?



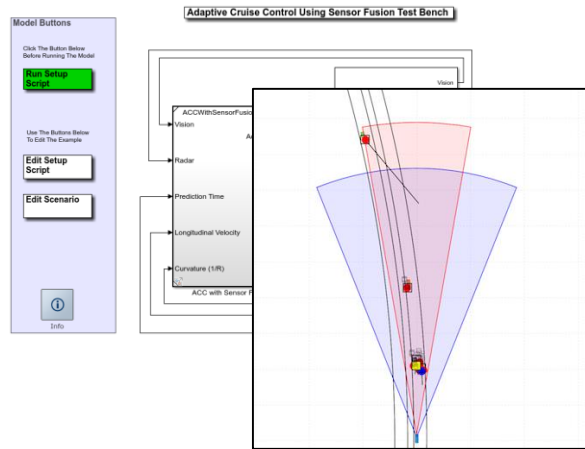
How can I
discover and design
in multiple domains?



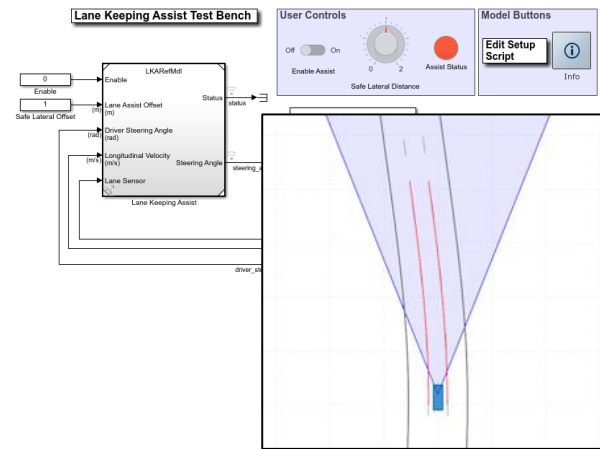
How can I
integrate
with other environments?

Design lateral and longitudinal Model Predictive Controllers

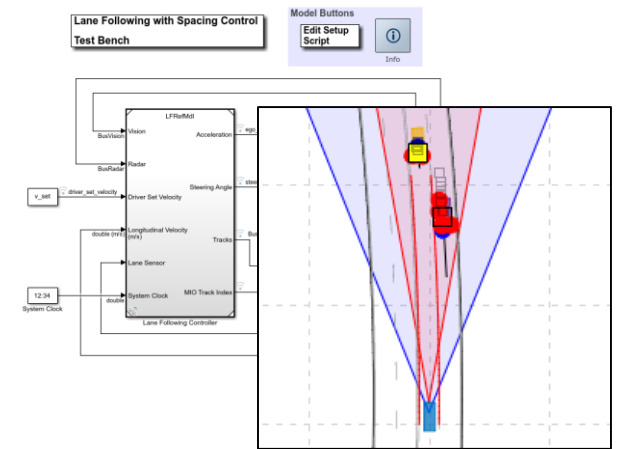
Longitudinal Control



Lateral Control



Longitudinal + Lateral



[Adaptive Cruise Control with Sensor Fusion](#)

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

R2017b

[Lane Keeping Assist with Lane Detection](#)

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

R2018a

[Lane Following Control with Sensor Fusion and Lane Detection](#)

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

R2018b

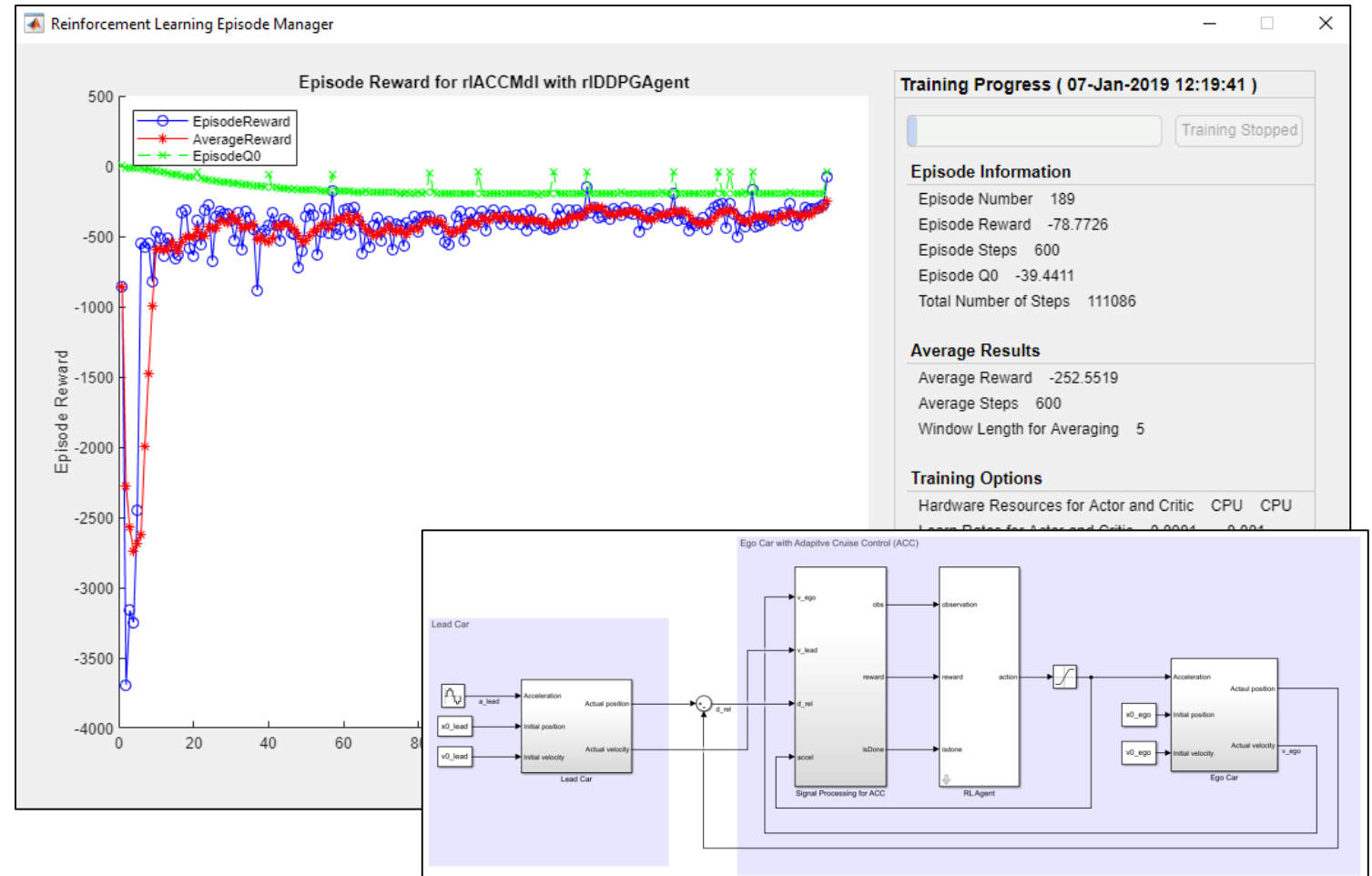
Train reinforcement learning networks for ADAS controllers

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

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

Reinforcement Learning Toolbox™

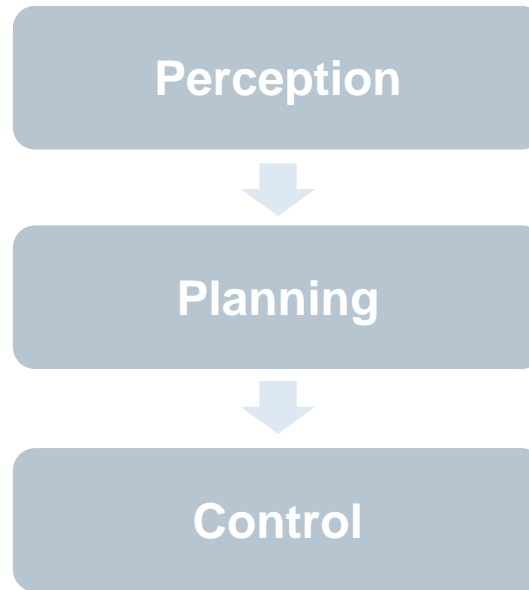
R2019a



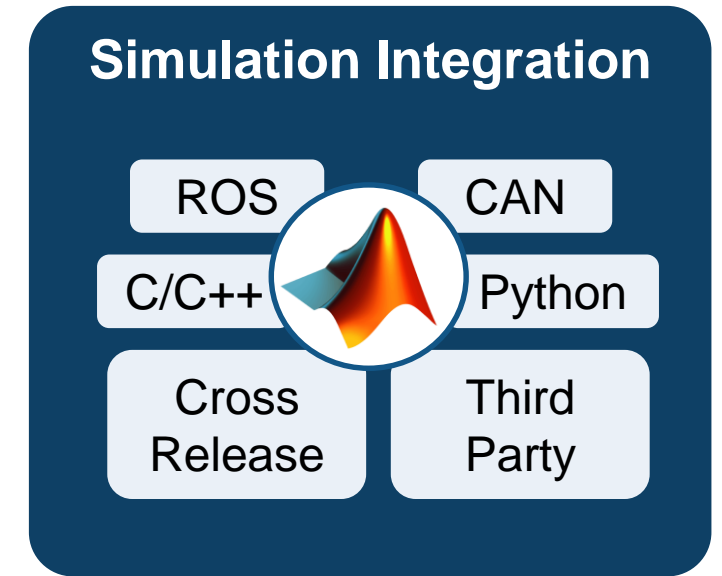
Some common questions from automated driving engineers



How can I
synthesize scenarios
to test my designs?



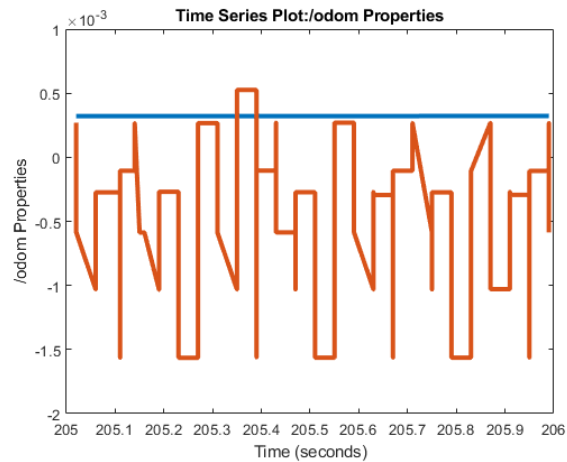
How can I
discover and design
in new domains?



How can I
integrate
with other environments?

Integrate with ROS

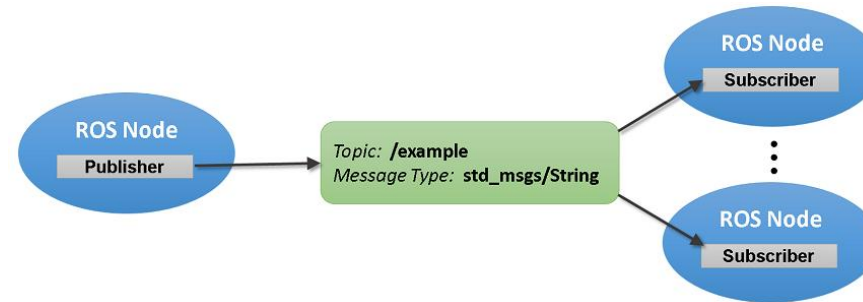
Replay logged ROS data



[Work with rosbag Logfiles](#)

Robotic System Toolbox™

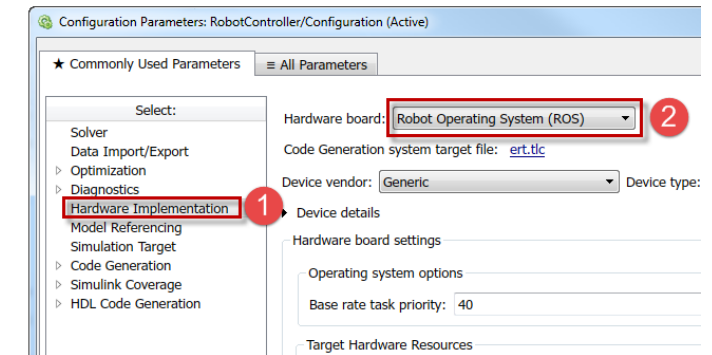
Connect to live ROS data



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

Robotic System Toolbox™

Generate standalone ROS node



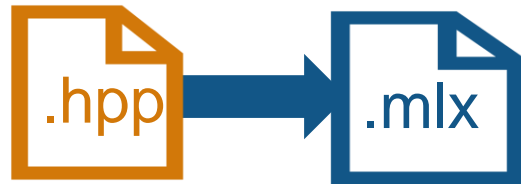
[Generate a Standalone ROS Node from Simulink](#)

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](#)

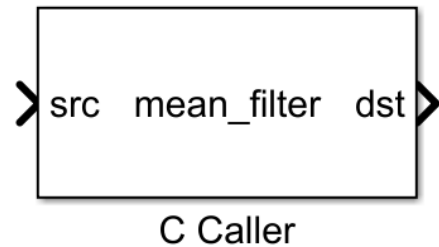
Computer Vision System Toolbox™

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;
```

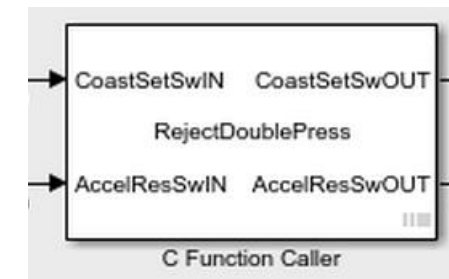
Name	DataType
-coeff	double
-init	double
-fault	Enum: fault_T

[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%

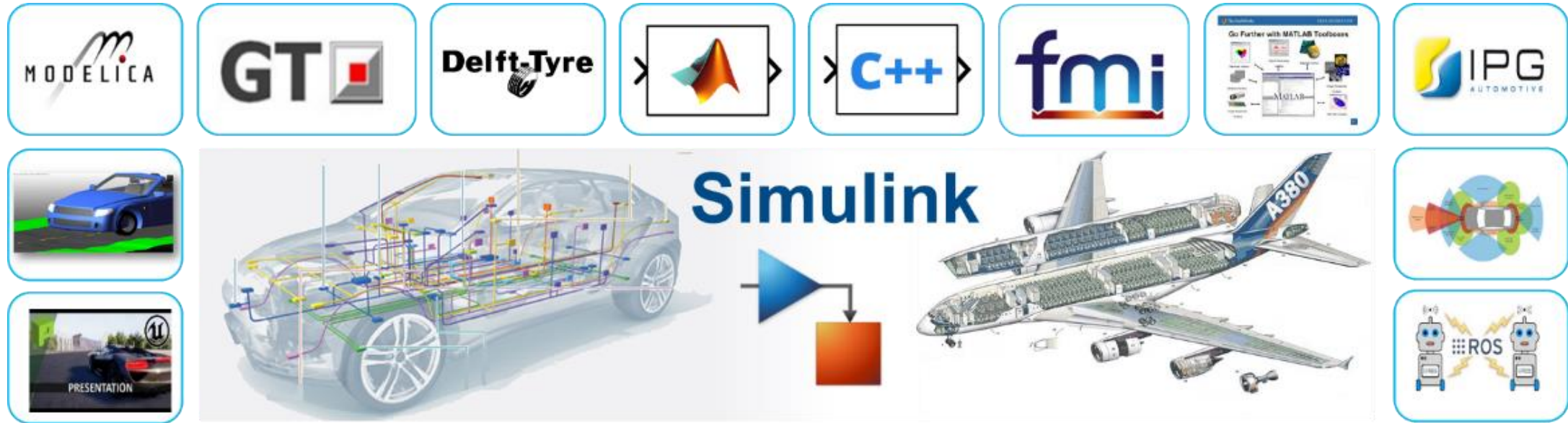
[Custom C Code Verification with Simulink Test](#)

Simulink Test™

Simulink Coverage™

R2019a

Connect to third party tools



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



Cross-release simulation through code generation

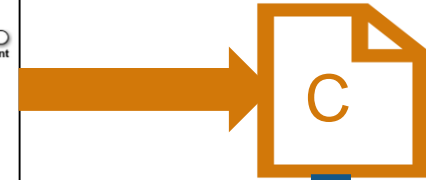
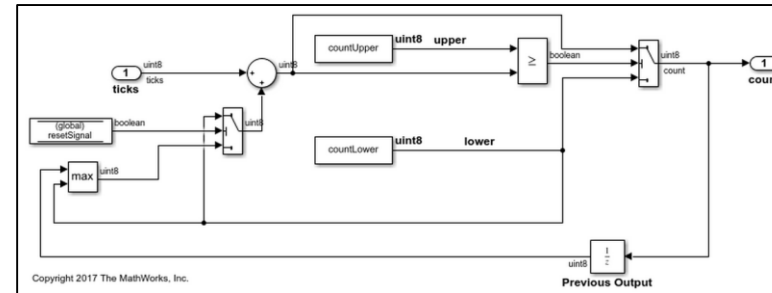
Integrate Generated Code by Using Cross-Release Workflow

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

Embedded Coder

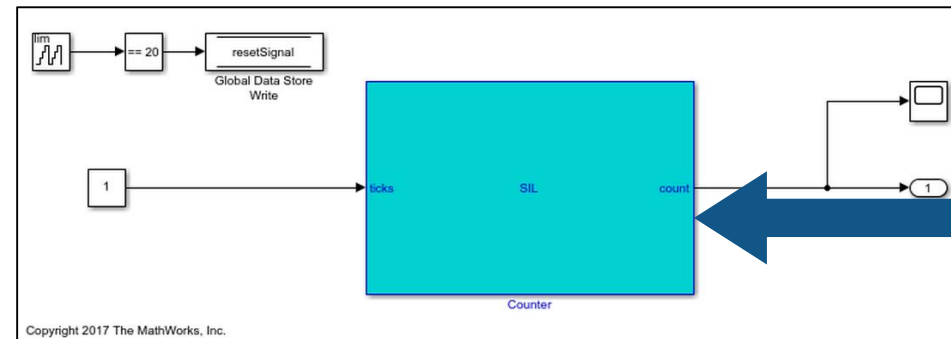
R2016a

Previous Release

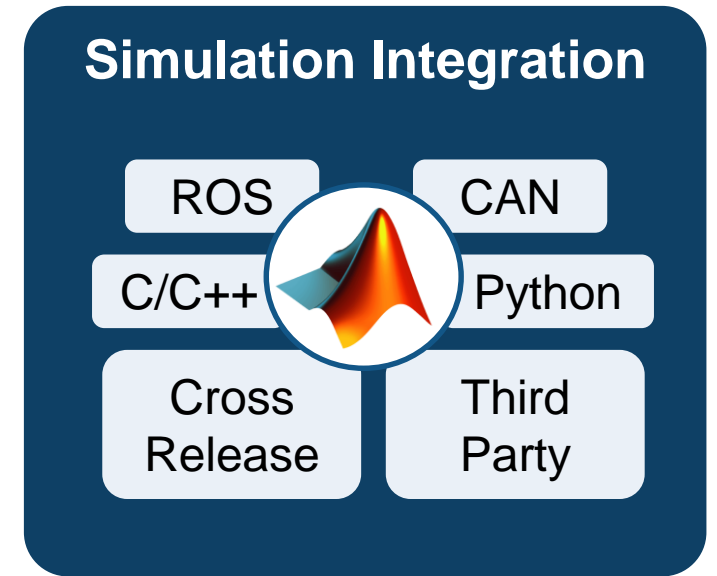
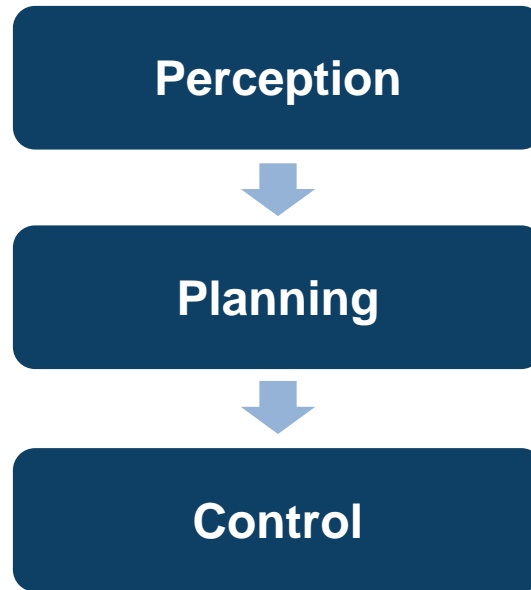
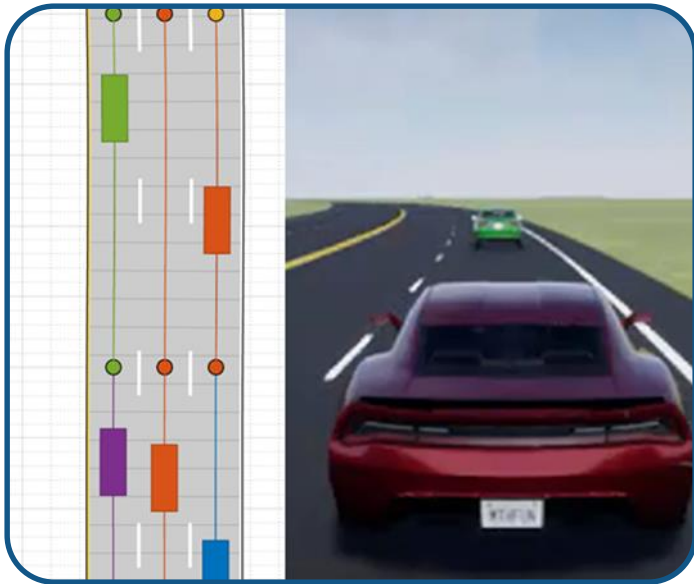


crossReleaseImport

Current Release



Some common questions from automated driving engineers

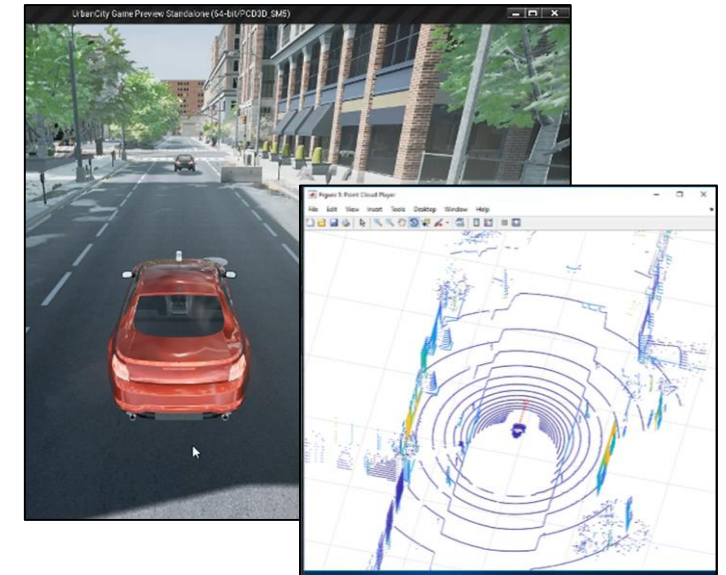
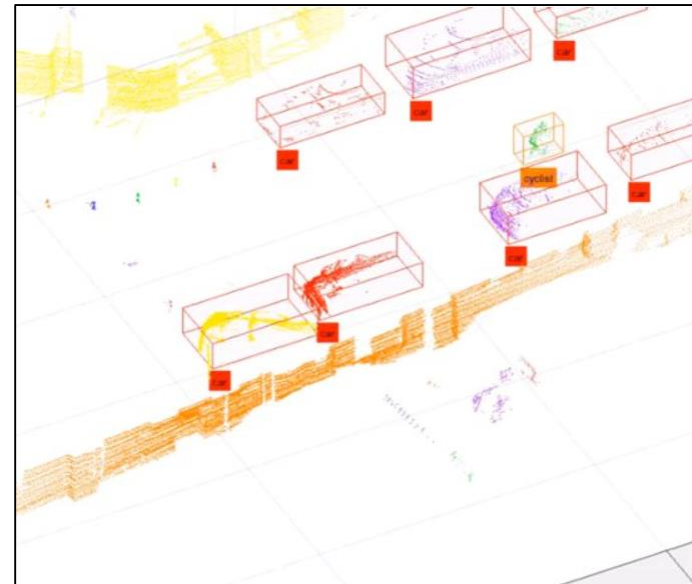


Synthesize scenarios
to test my designs

Discover and design
in multiple domains

Integrate
with other environments

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

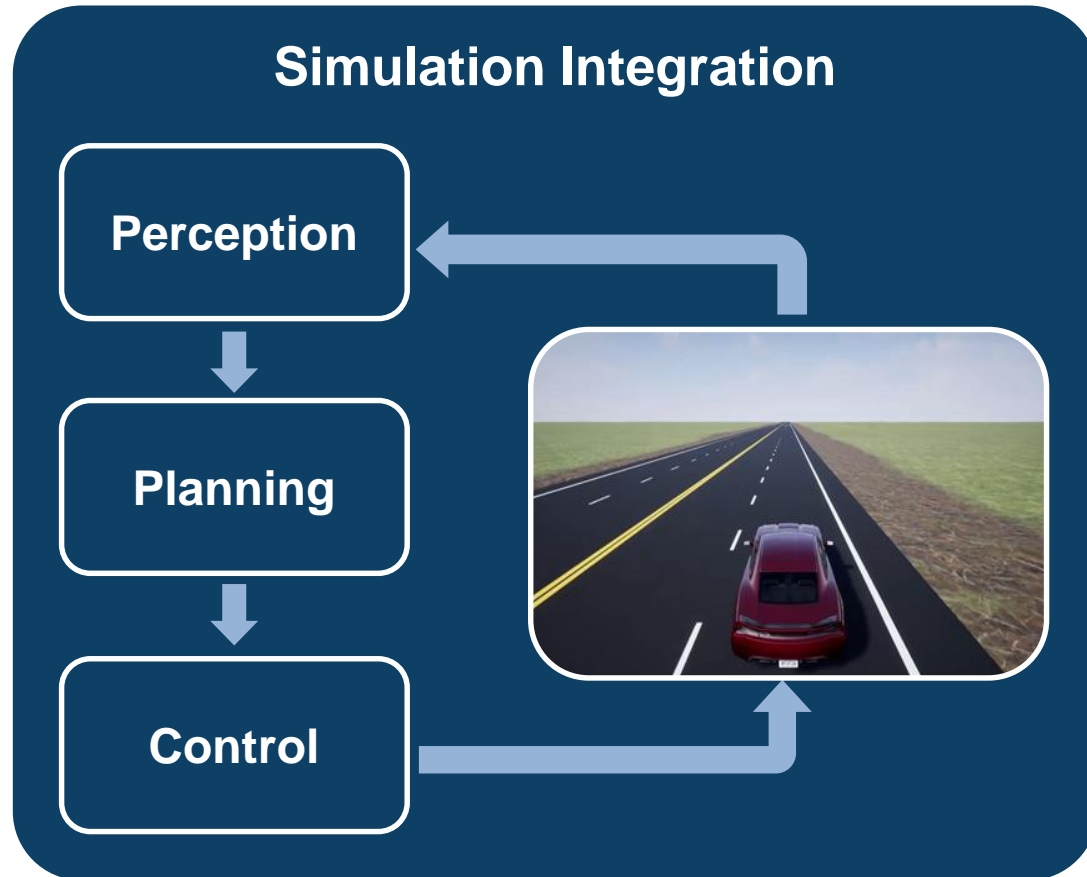
Autoliv labels ground truth lidar data

- 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

Develop Automated Driving Systems with MATLAB and Simulink



Discuss your application with a MathWorks field engineer to help you structure your evaluation

- Understand your goals
- Recommend tasks
- Answer questions