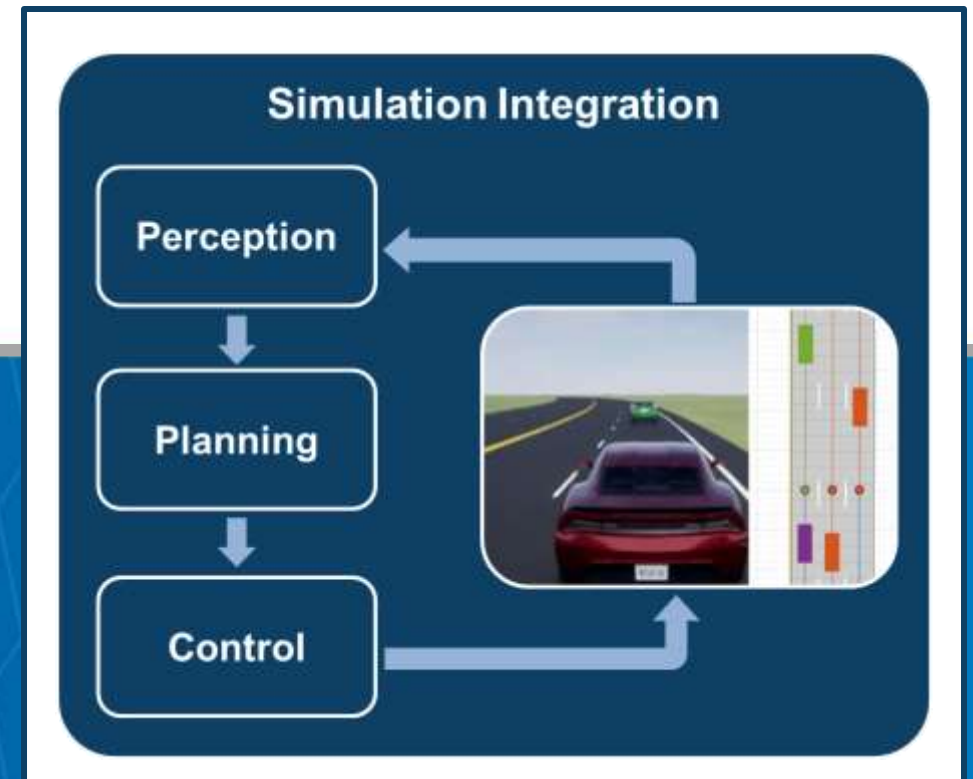


What's New in Automated Driving with MATLAB and Simulink

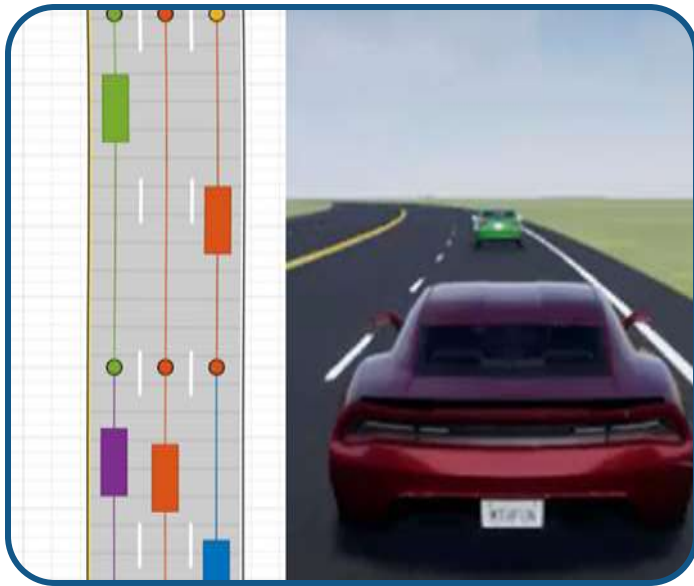
Mark Corless

Industry Marketing

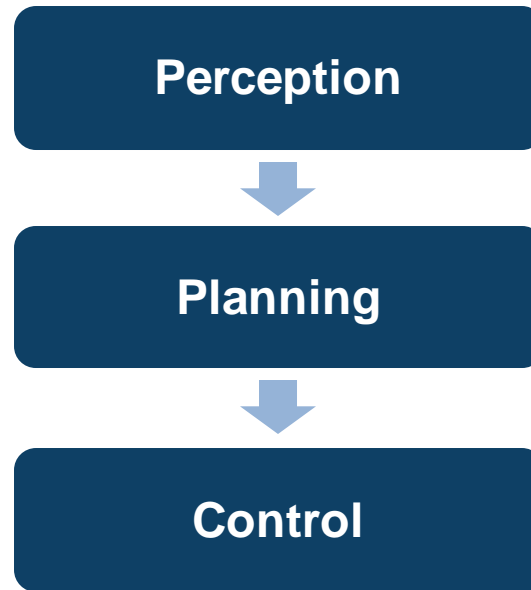
Automated Driving Segment Manager



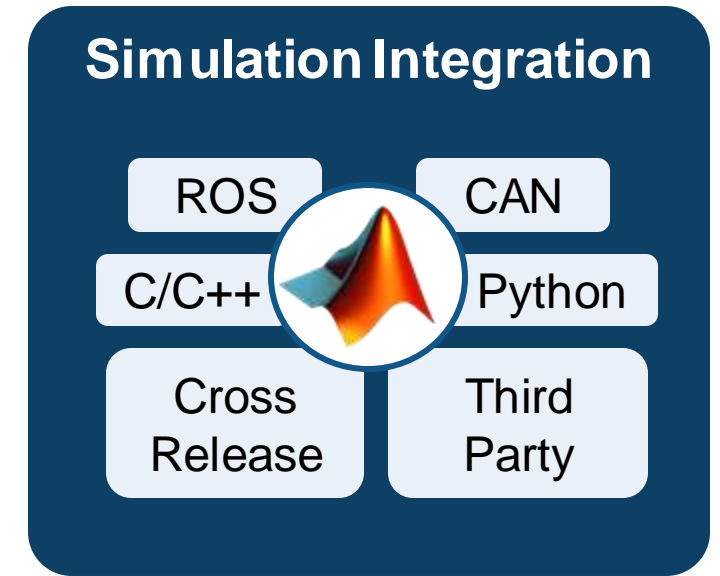
Some common questions from automated driving engineers



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

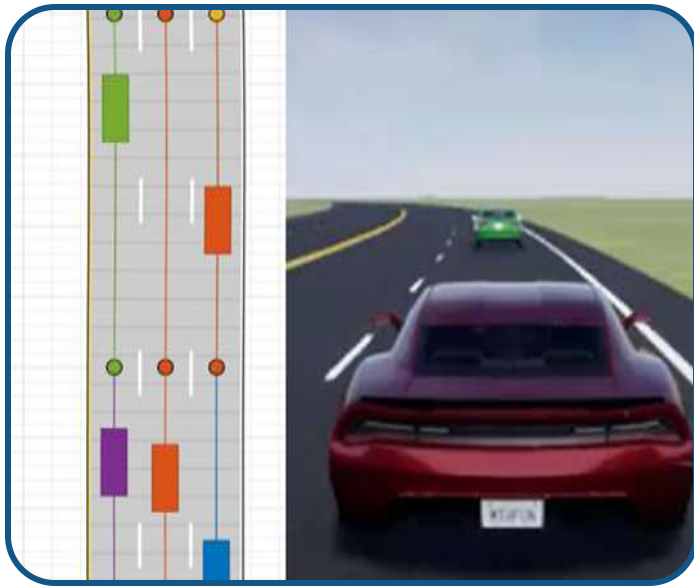


How can I **discover and design** in multiple domains?

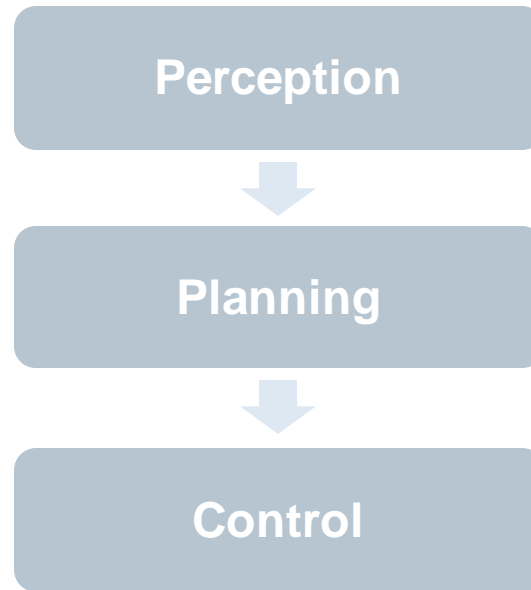


How can I **integrate** with other environments?

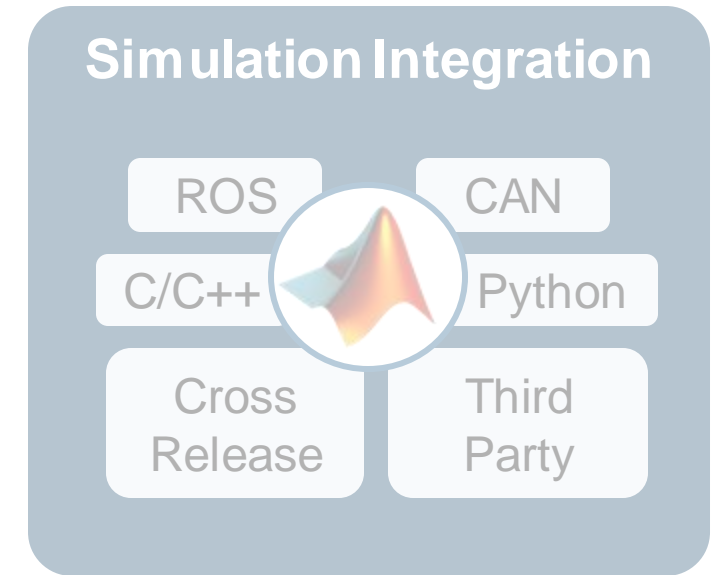
Some common questions from automated driving engineers



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



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



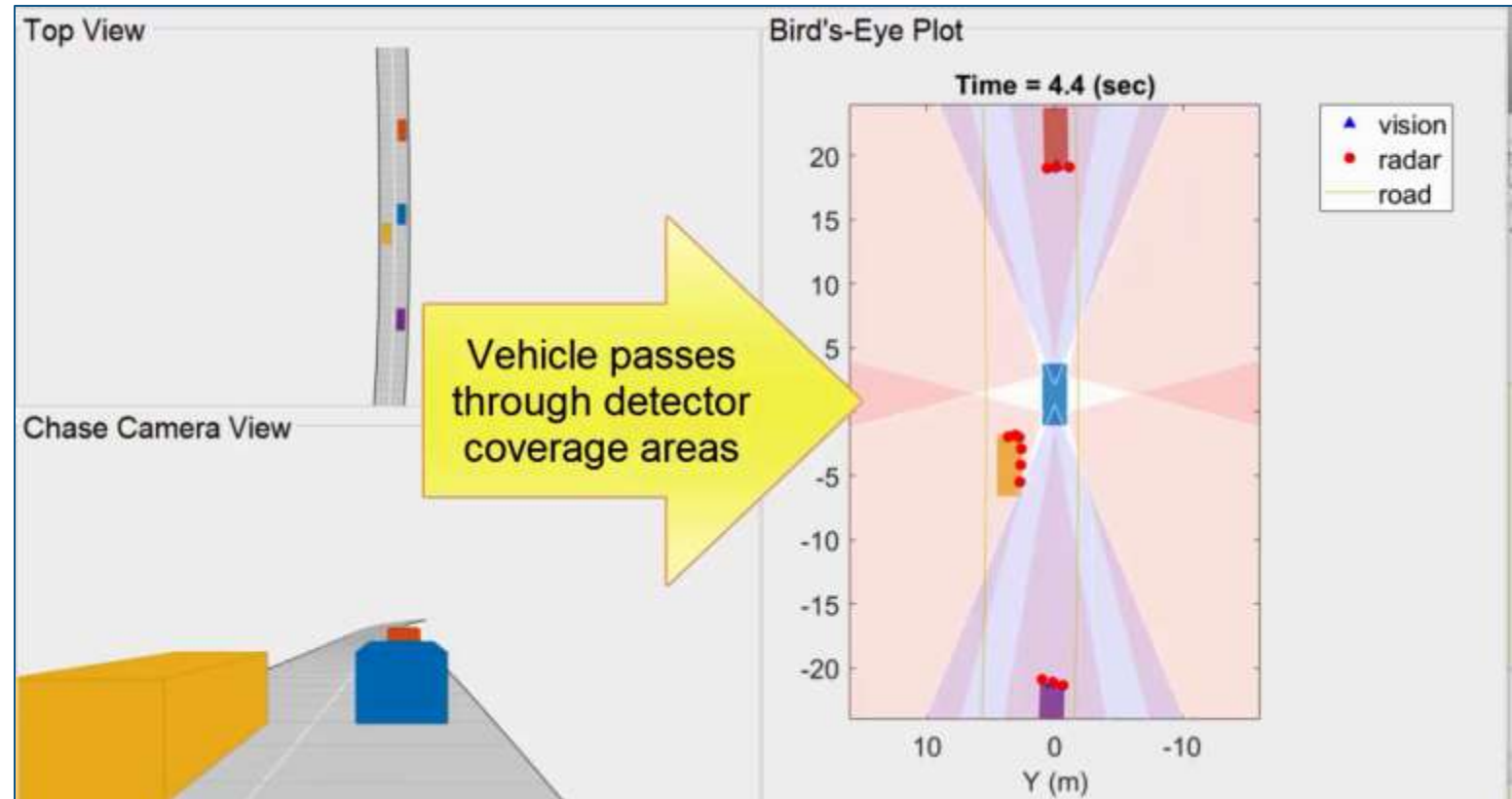
How can I **integrate** with other environments?

Synthesize scenarios to test sensor fusion algorithms

Sensor Fusion Using Synthetic Radar and Vision Data

- Synthesize road and vehicles
- Add probabilistic vision and radar detection sensors
- Fuse and track detections
- Visualize sensor coverage areas, detections, and tracks

Automated Driving Toolbox™
R2017a



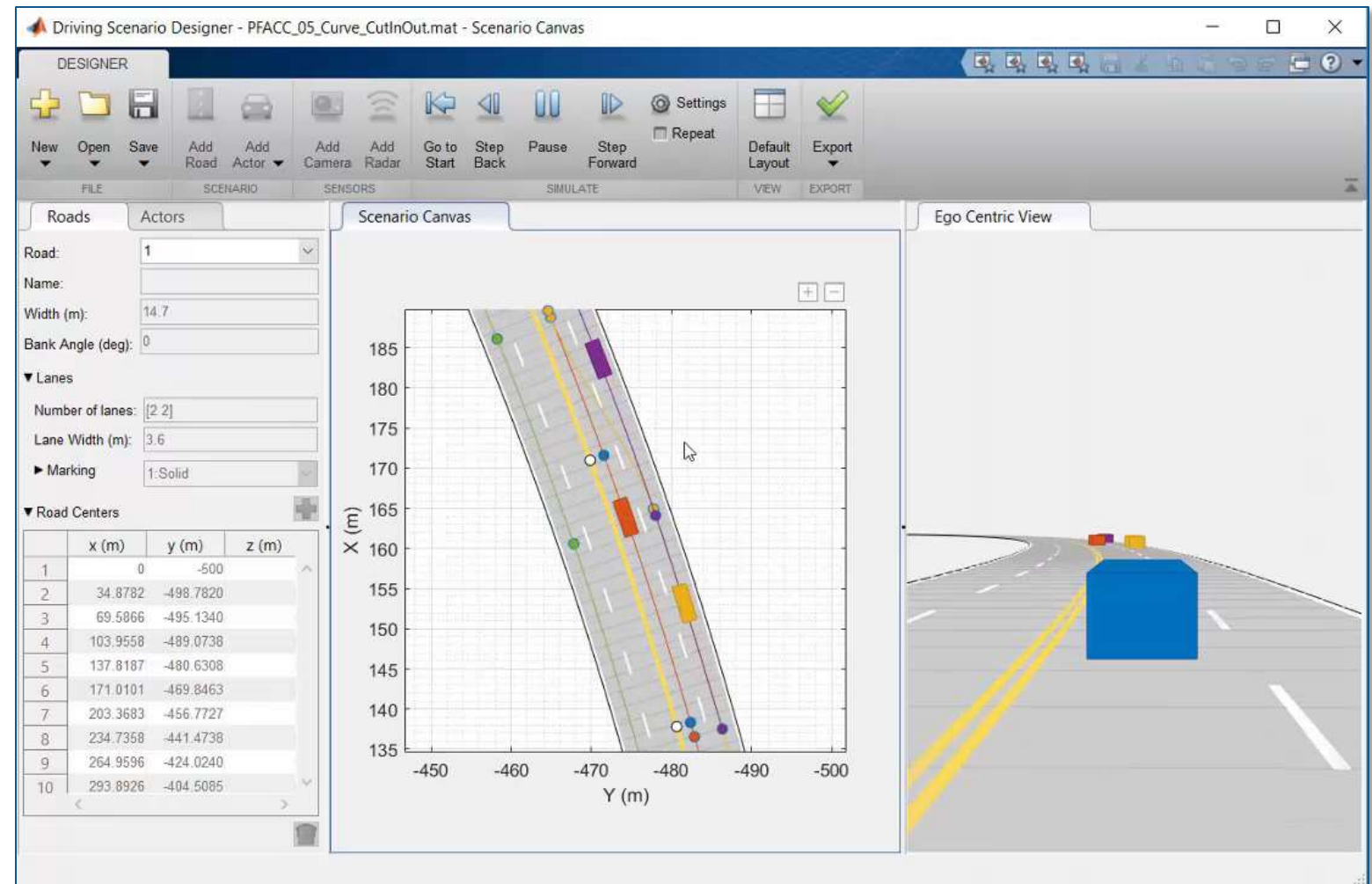
Graphically author driving scenarios

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™

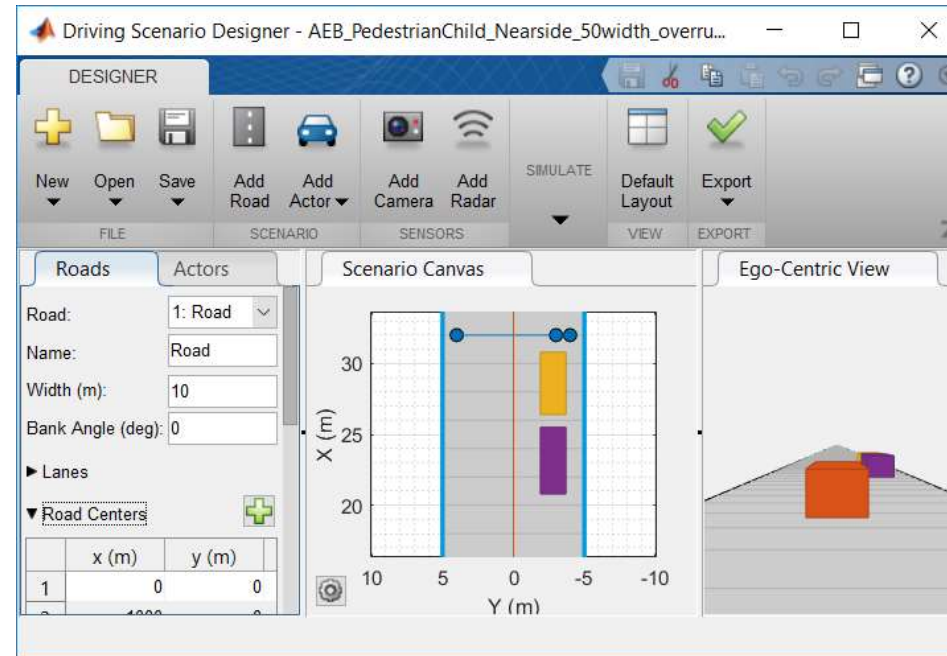
R2018a



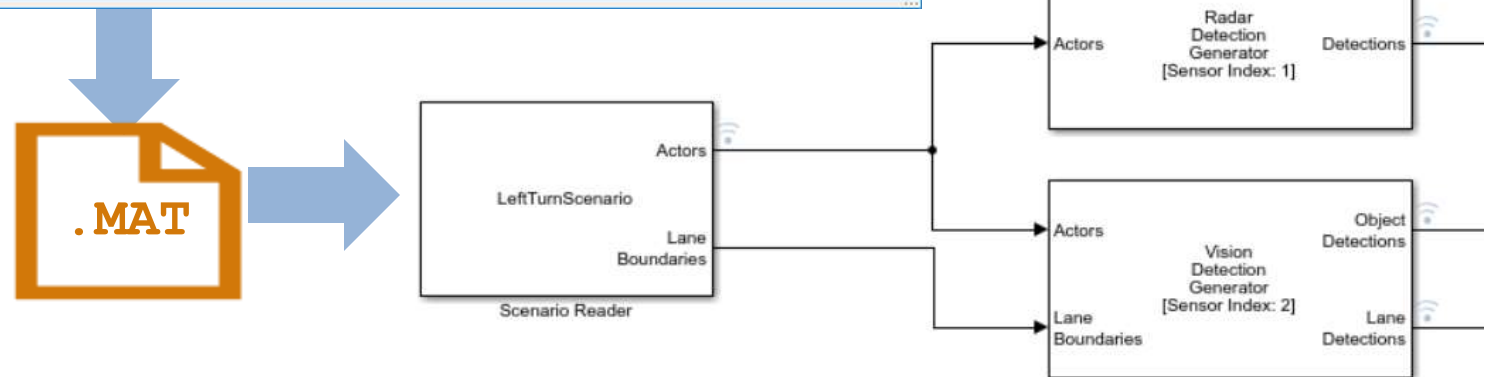
Integrate driving scenarios into Simulink simulations

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



Integrate driving scenario into closed loop simulation

Lane Following Control with Sensor Fusion

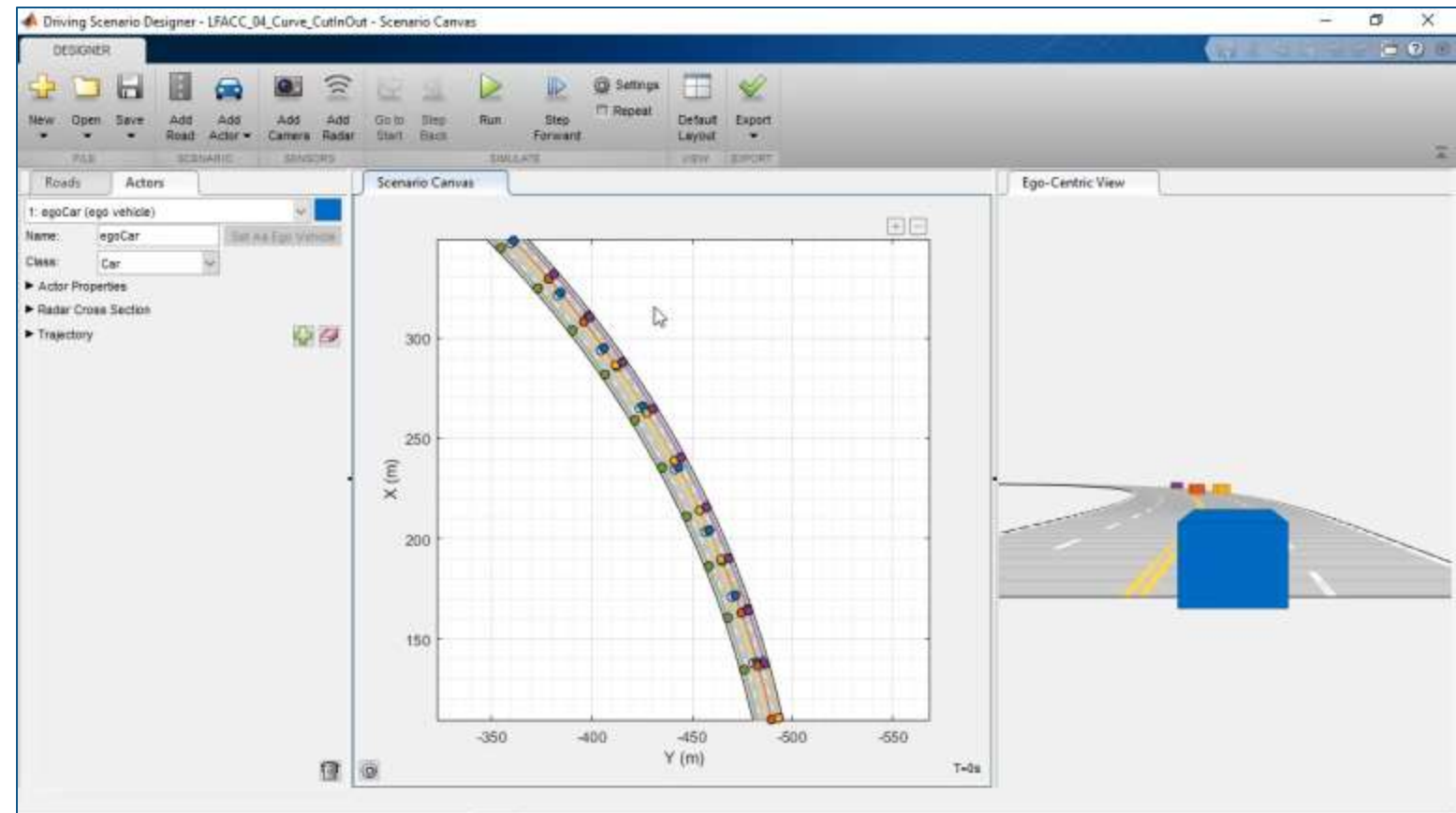
- Integrate scenario into system
- Design lateral (lane keeping) and longitudinal (lane spacing) model predictive controllers
- Visualize sensors and tracks
- Generate C/C++ code
- Test with software in the loop (SIL) simulation

Model Predictive Control Toolbox™

Automated Driving Toolbox™

Embedded Coder®

R2018b



Design lateral and longitudinal controls

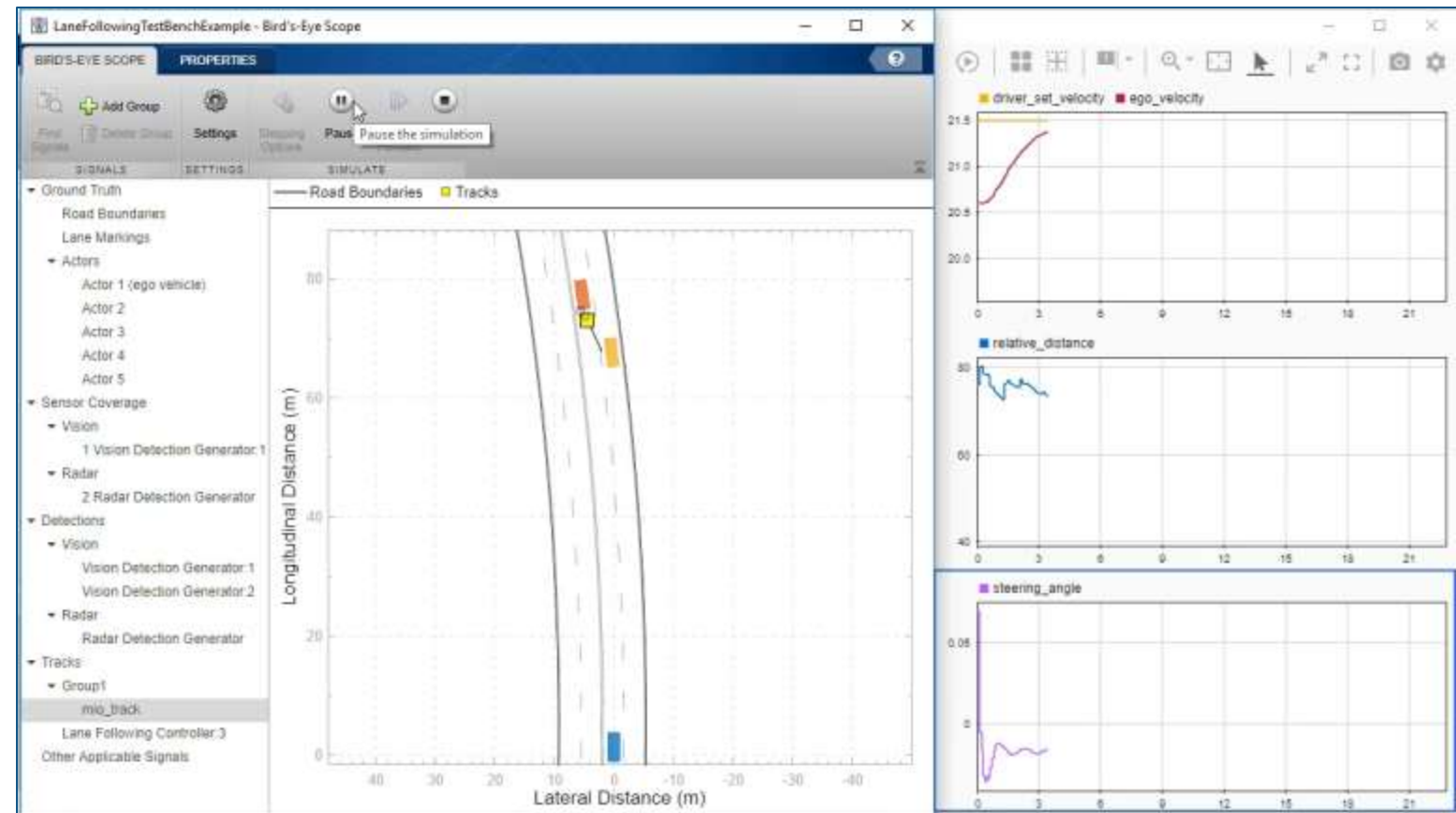
Lane Following Control with Sensor Fusion

- Integrate scenario into system
- Design lateral (lane keeping) and longitudinal (lane spacing) model predictive controllers
- Visualize sensors and tracks
- Generate C/C++ code
- Test with software in the loop (SIL) simulation

Model Predictive Control Toolbox™

Automated Driving Toolbox™

Embedded Coder®



R2018b

Visualize sensor detections and tracks

Lane Following Control with Sensor Fusion

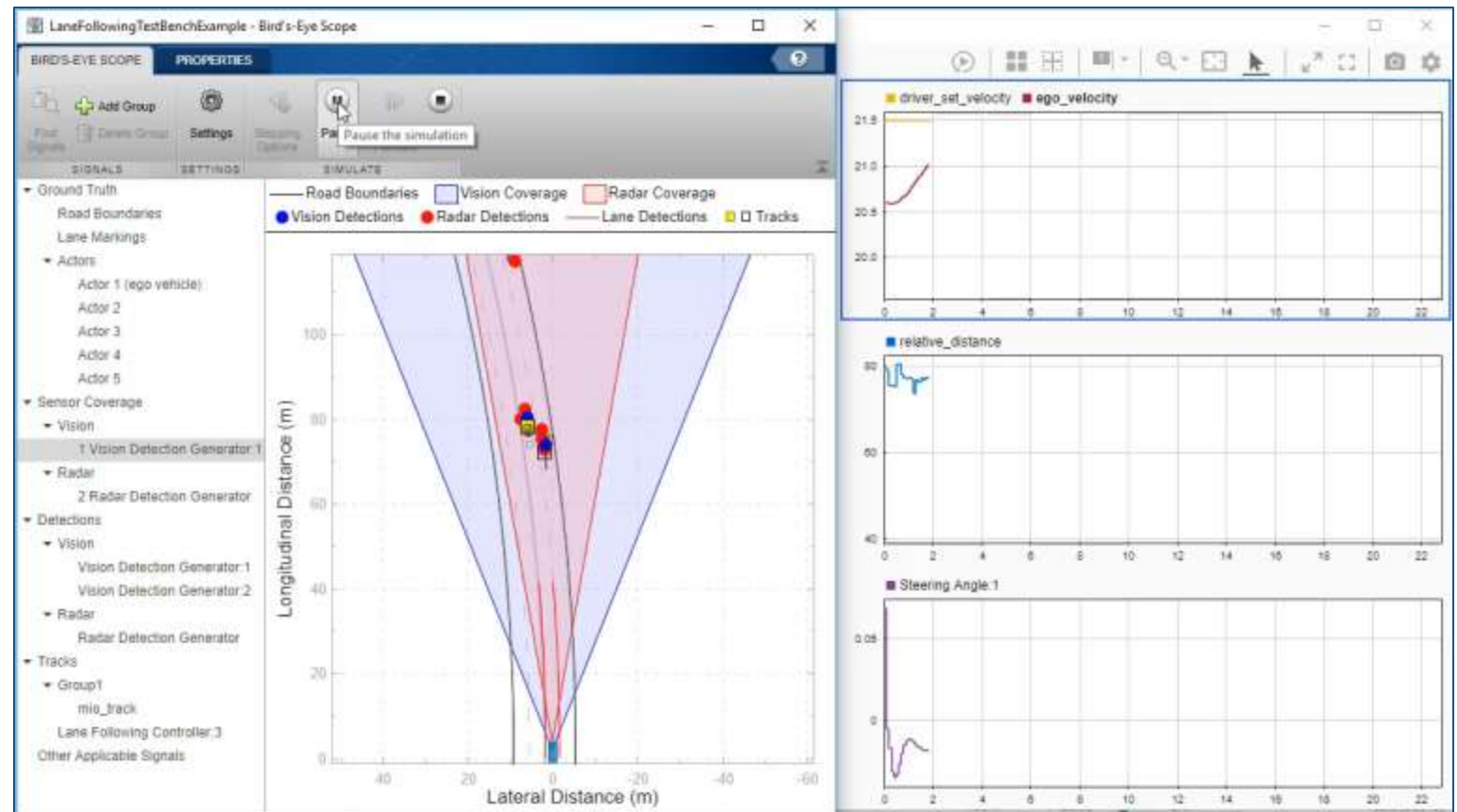
- Integrate scenario into system
- Design lateral (lane keeping) and longitudinal (lane spacing) model predictive controllers
- Visualize sensors and tracks
- Generate C/C++ code
- Test with software in the loop (SIL) simulation

Model Predictive Control Toolbox™

Automated Driving Toolbox™

Embedded Coder®

R2018b



Automate testing against driving scenarios

Testing a Lane Following Controller with Simulink Test

- Author high level requirements
- Synthesize driving scenarios
- Specify assessment criteria
- Run interactive simulation
- Automate regression testing
- Review verification status

Simulink Test™

Automated Driving Toolbox™

Model Predictive Control Toolbox™

R2018b

The screenshot shows the Simulink Test Manager interface. The 'TESTS' toolbar at the top has a red box around the 'Run' button. The 'Test Browser' on the left shows a tree view of test scenarios, with 'ACC_ISO_TargetDiscriminationTest' selected and highlighted by a blue box. Below this tree is a 'PROPERTY' table:

PROPERTY	VALUE
Name	ACC_ISO_TargetDiscriminationTest
Type	Simulation Test
Model	LaneFollowingTestBenchExample
Simulation Mode	Normal
Location	C:\02_ADST\2018b\Demos\...
Enabled	<input checked="" type="checkbox"/>
Hierarchy	LaneFollowingTestScenario...
Tags	Type comma or space separat

The main configuration pane on the right shows the 'DESCRIPTION*' section with a 'REQUIREMENTS*' link (circled in blue). Below it, the 'SYSTEM UNDER TEST*' section shows the 'Model' set to 'LaneFollowingTestBenchExample' (circled in blue). The 'CALLBACKS*' section contains a 'POST-LOAD*' callback with the following code:

```

1 scenarioId = 1;
2 helperLFSetUp;
    
```

A blue callout points to this code with the text 'Define scenario ID and data initialization'. At the bottom, the 'CLEANUP*' section contains the following code:

```

1 plotLFResults(sltest_simout.logout);
    
```

A blue callout points to this code with the text 'Plot the results'. Other callouts point to the 'Requirements link' and 'Simulink Model' fields.

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 Live Editor interface. The main window shows a script titled "PlaybackScenarioExample.mlx" with the following content:

```
Summary
This example shows how to automatically generate a virtual driving scenario from vehicle data recorded using the GPS and lidar sensors.

Helper Functions
helperGetEgoData
This function reads the ego vehicle data from a text file and converts into a structure.

108 function [egoData] = helperGetEgoData(egoFile)
109 %Read the ego vehicle data from text file
110 fileID = fopen(egoFile);
111 content = textscan(fileID, '%f %f %f');
112 fields = {'lat', 'lon', 'Time'};
113 egoData = cell2struct(content, fields, 2);
114 fclose(fileID);
115 end

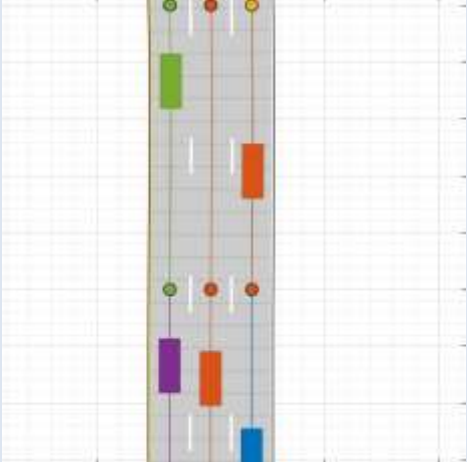

helperGetNonEgoData
This function reads the processed lidar data from a text file and converts into a structure. The processed lidar data contains information about the
```

The right side of the interface features a yellow header "Simulate synthesized scenario" above a 3D visualization of a road scene. The scene includes a blue ego vehicle, a green vehicle, and a purple vehicle on a road with lane markings. A 2D plot of the road geometry is visible in the background, showing X (m) on the horizontal axis (ranging from 270 to 320) and Y (m) on the vertical axis (ranging from 80 to 120).

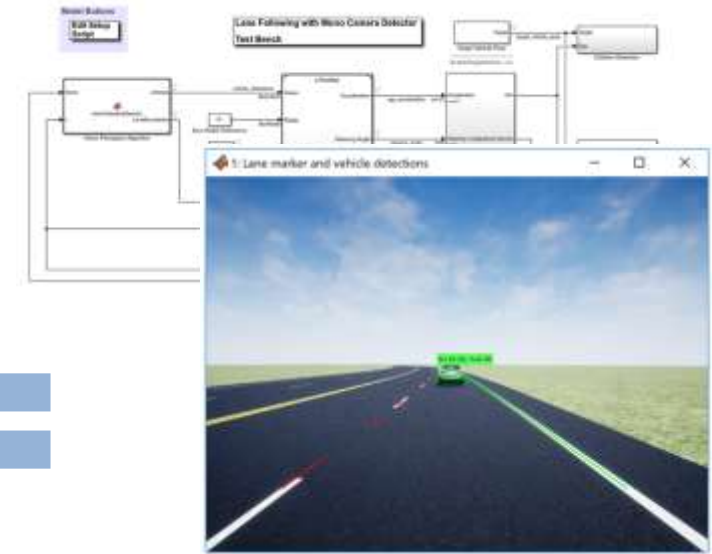
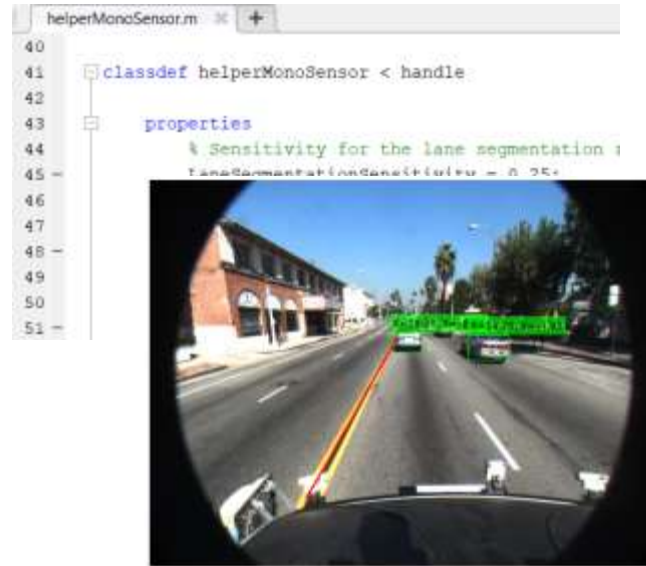
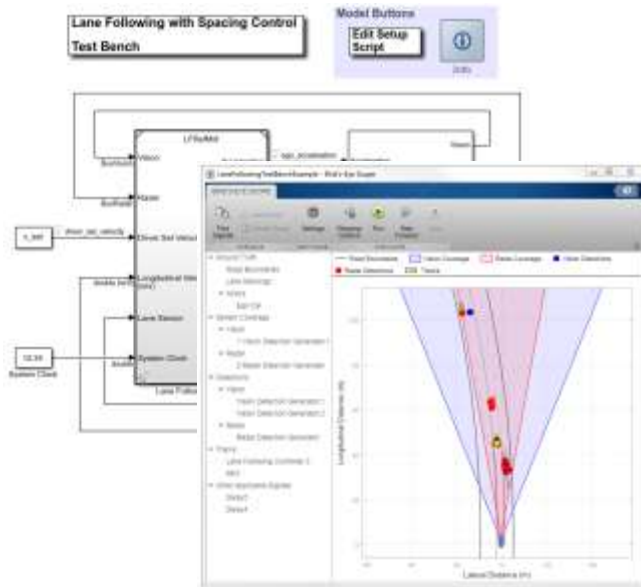
How can I design with virtual scenarios?

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) 	3D Simulation (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



[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

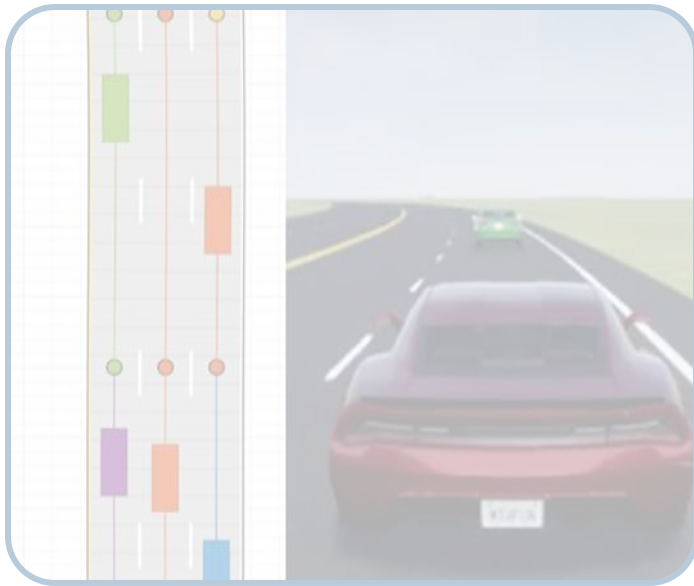
Model Predictive Control Toolbox™

Automated Driving Toolbox™

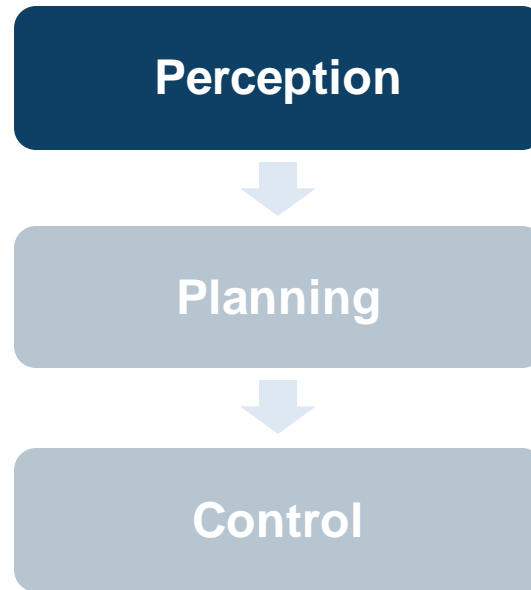
Vehicle Dynamics Blockset™



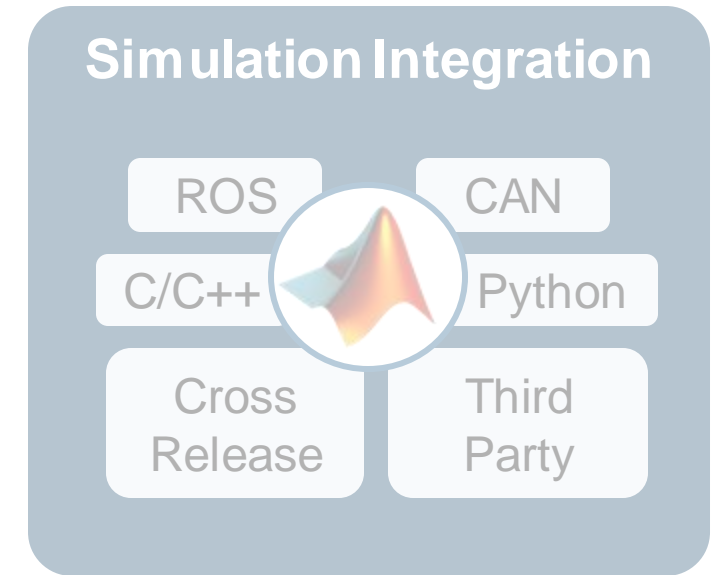
Some common questions from automated driving engineers



How can I
synthesize scenarios
to test my designs?

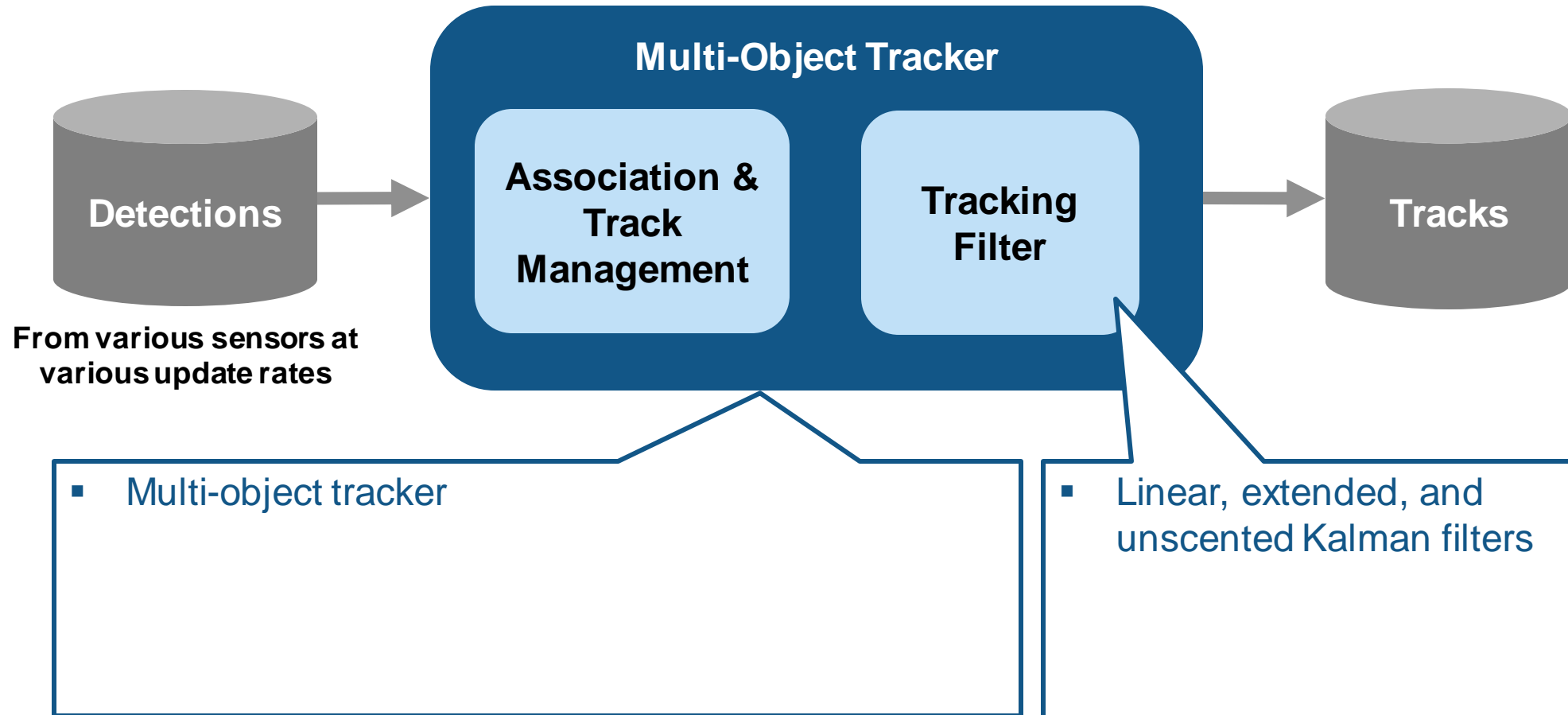


How can I
discover and design
in multiple domains?

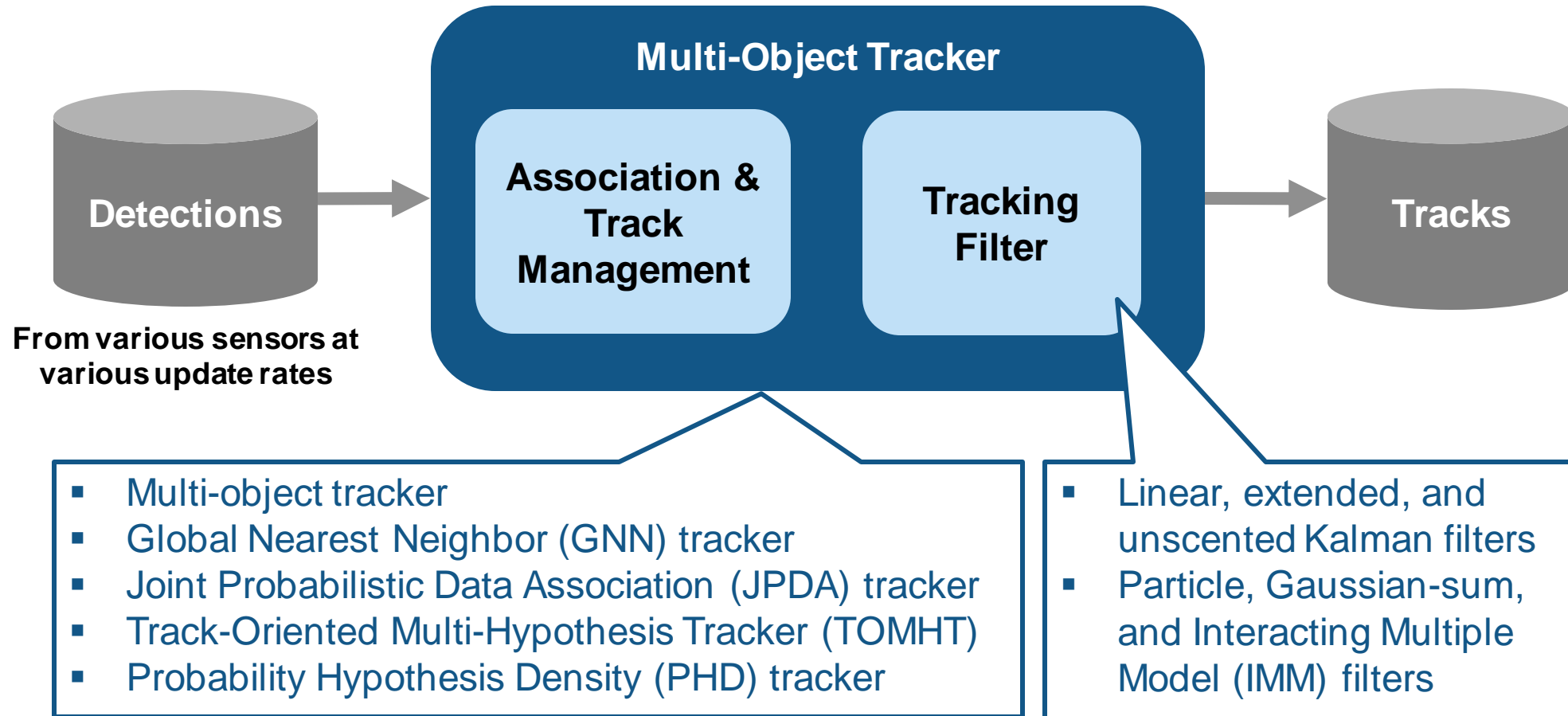


How can I
integrate
with other environments?

Design trackers



Design trackers



Automated Driving Toolbox™

Sensor Fusion and Tracking Toolbox™

R2019a

Design multi-object trackers

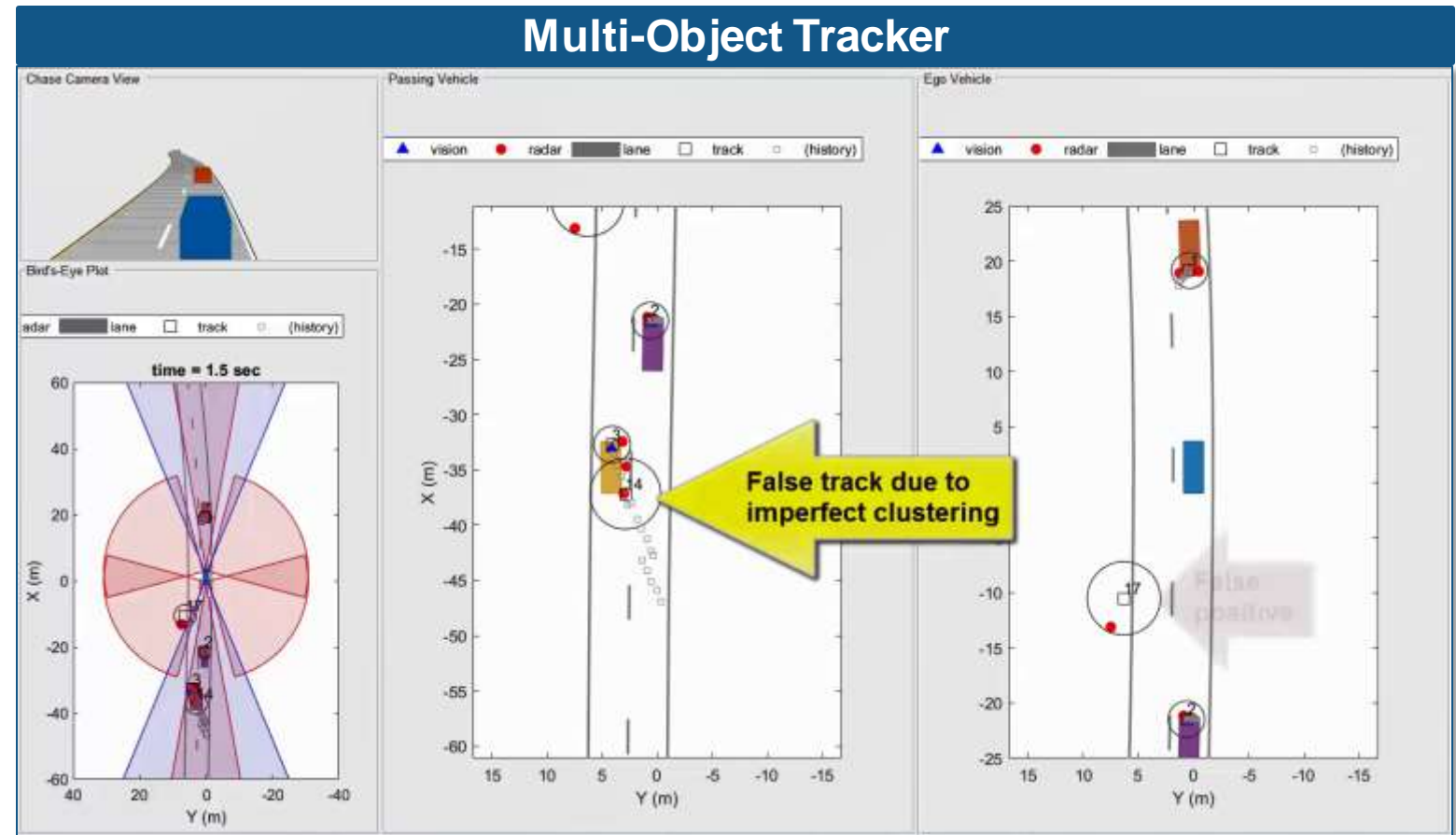
Extended Object Tracking

- 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

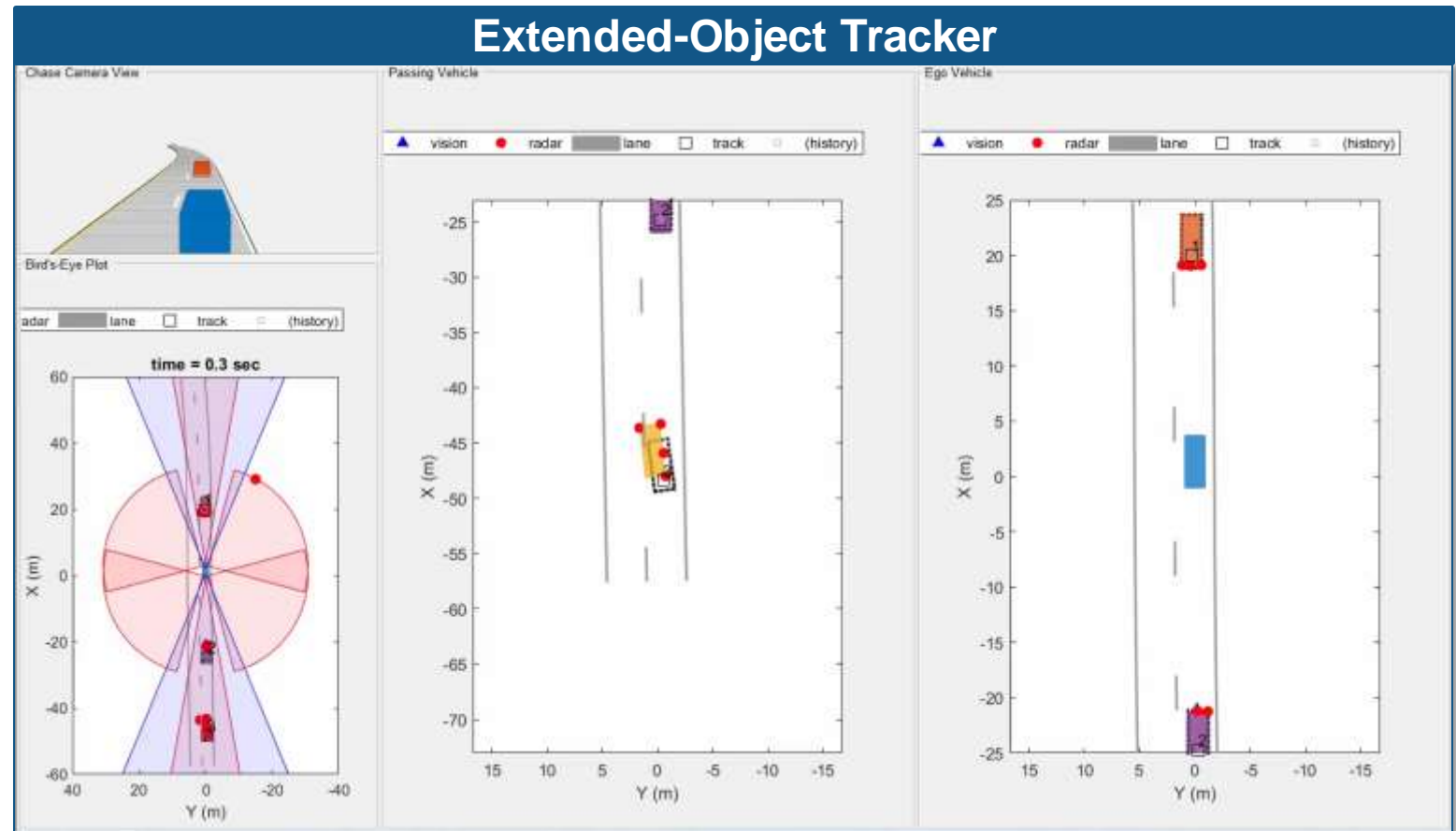
Extended Object Tracking

- 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

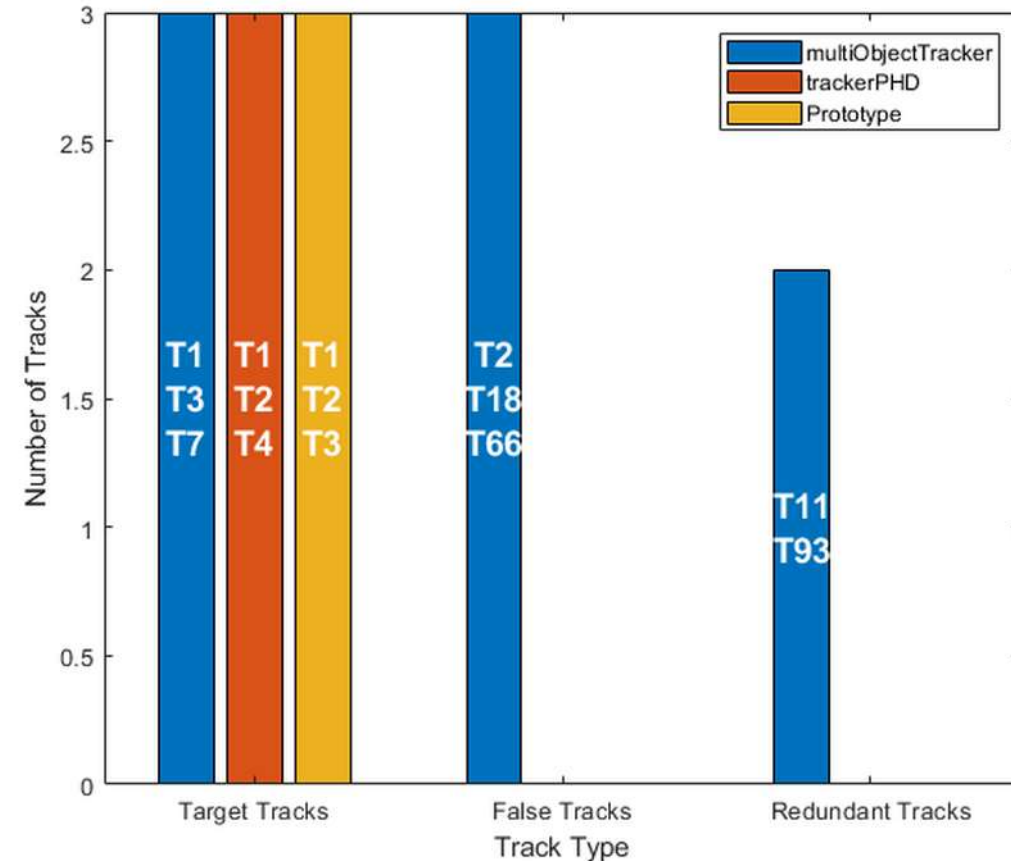
Extended Object Tracking

- 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

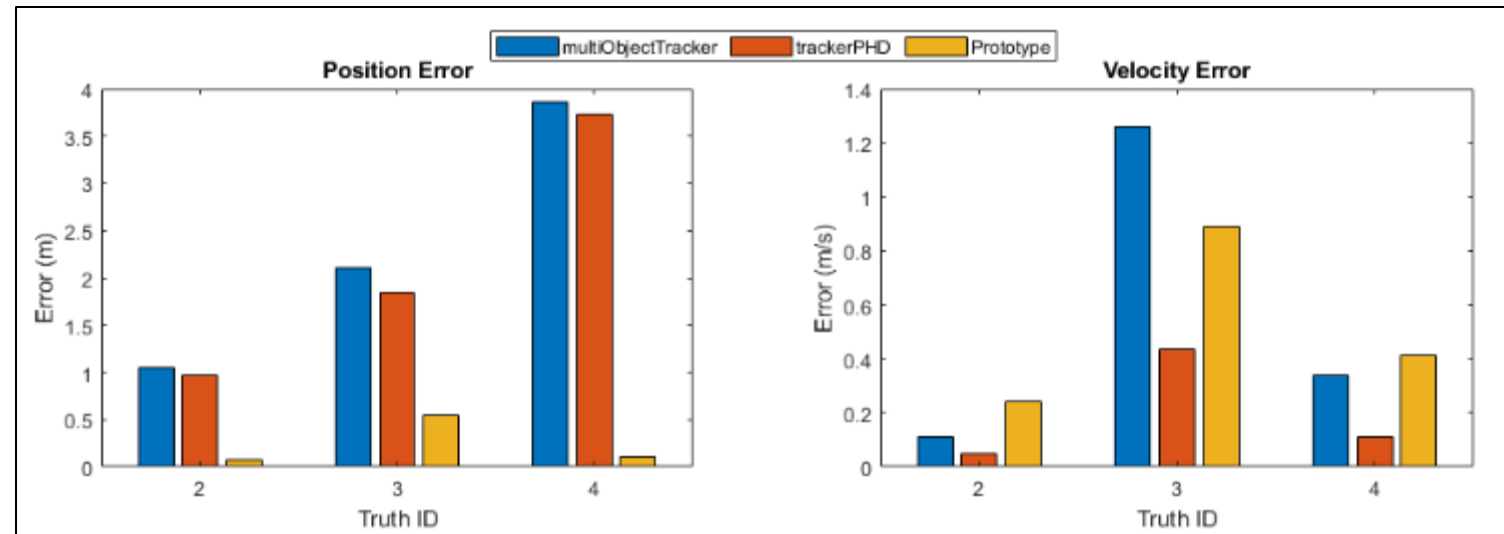
Extended Object Tracking

- 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

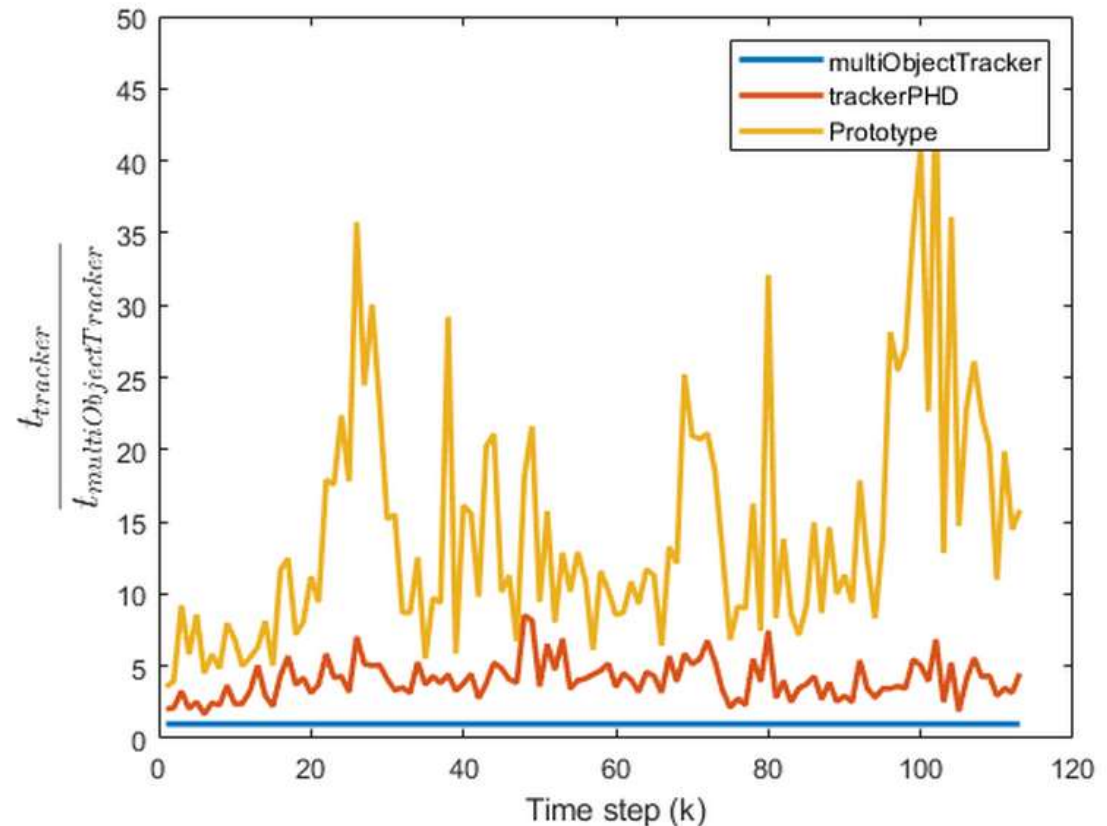
Extended Object Tracking

- 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

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` class. The Command Window shows the execution of `pcshow(pcObstacles)` and the resulting output of the `detBBboxes` property, which is a 6x10 matrix. A yellow callout box points to the first four columns of this matrix, identifying them as X-Center, Y-Center, Z-Center, Length, Width, and Height.

```

methods (Access = protected)
function [bboxDets, obstacleI
% Crop point cloud
[pcSurvived, survivedIn
% Remove ground plane
[pcObstacles, obstacleI
% Form clusters and ge
detBBboxes = getBoundin
% Assembl
bboxDets
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

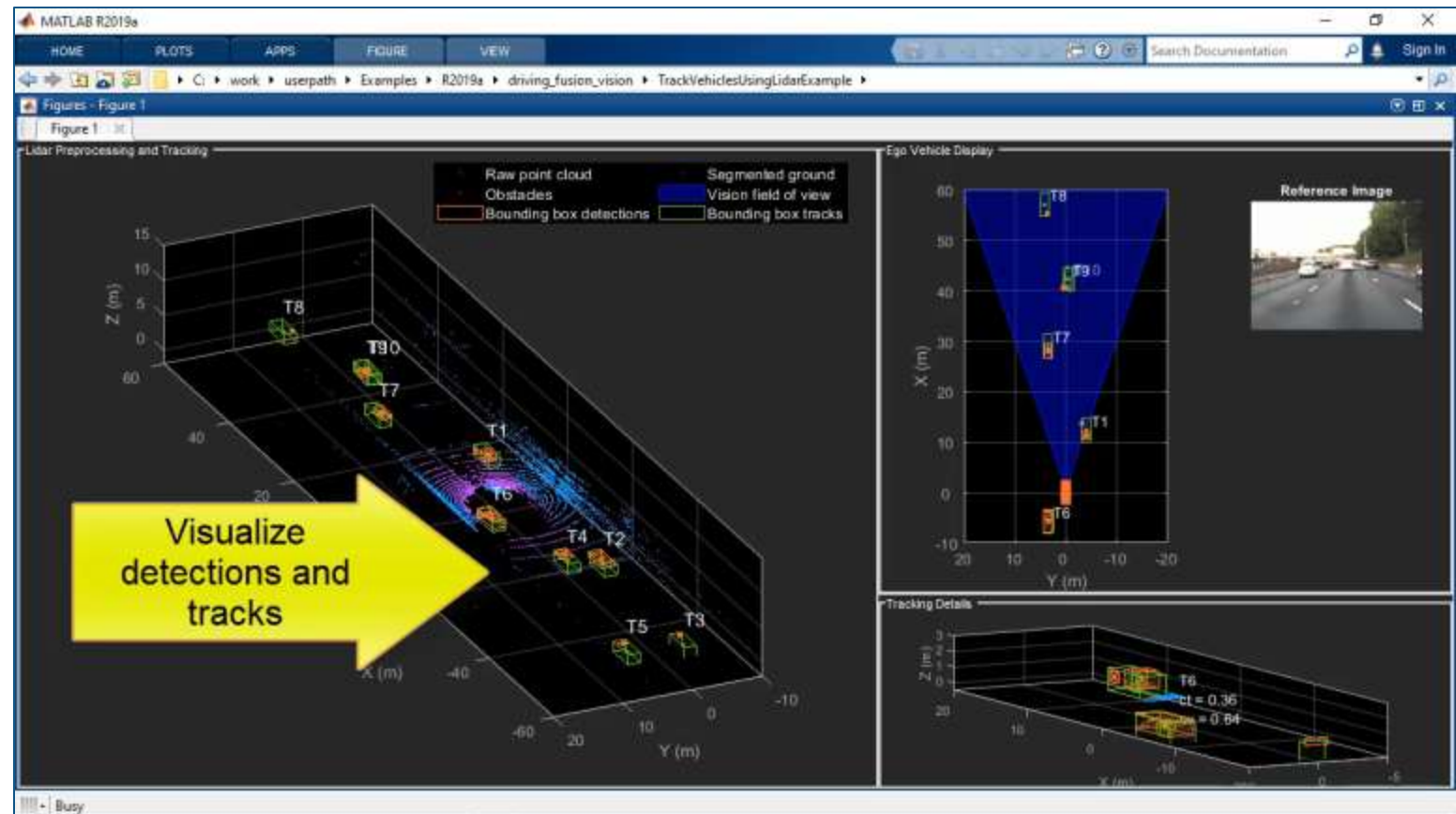
Track Vehicles Using Lidar: From Point Cloud to Track List

- 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.mldats

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

Code

```

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

```

assignmentGate = [10 100]; % Assignment threshold;
confThreshold = [7 10]; % Confirmation threshold for h
delThreshold = [8 10]; % Deletion threshold for high
Kc = 1e-5; % False-alarm rate per unit time

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

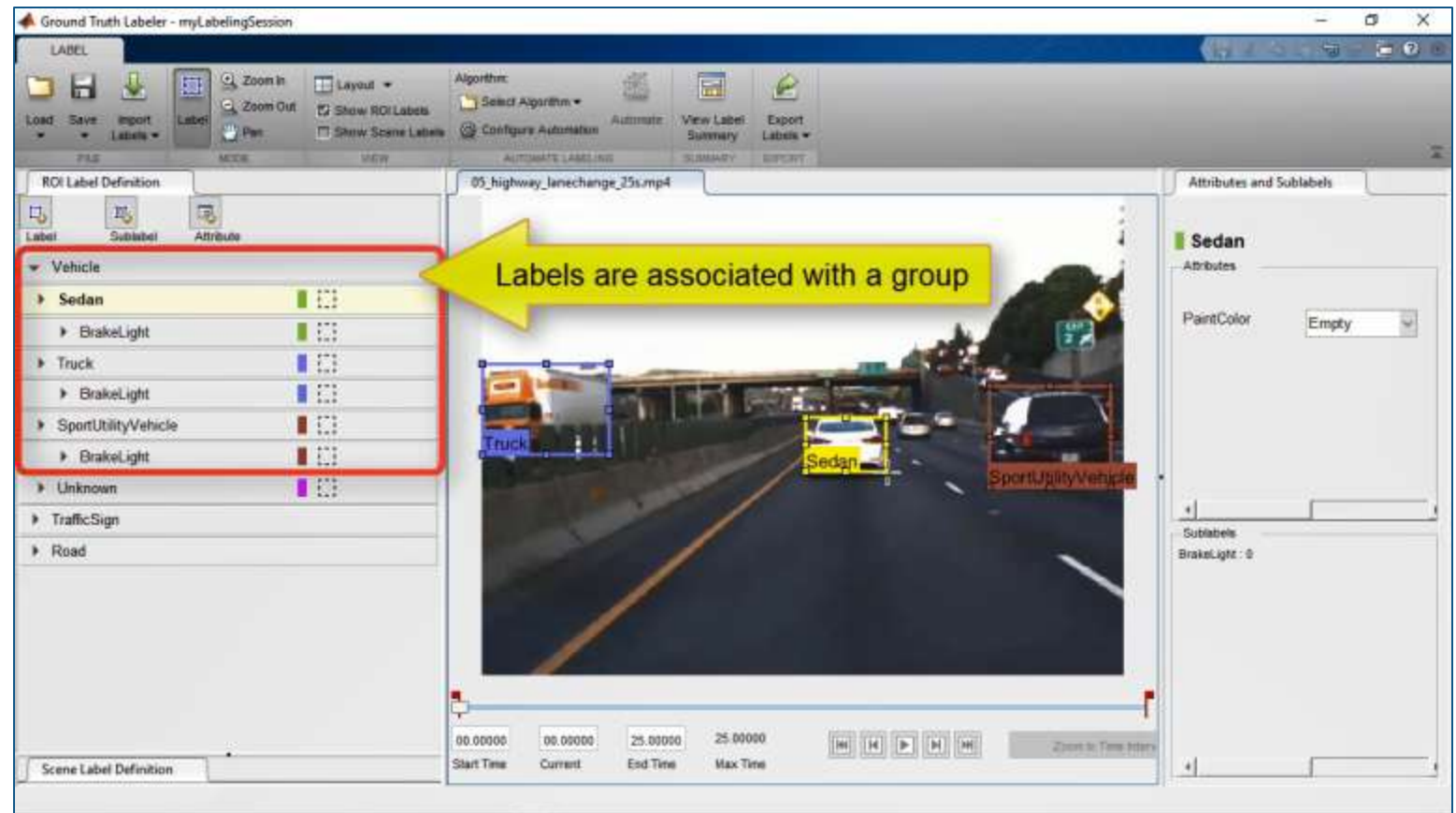
Create region of interest labels and groups

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**



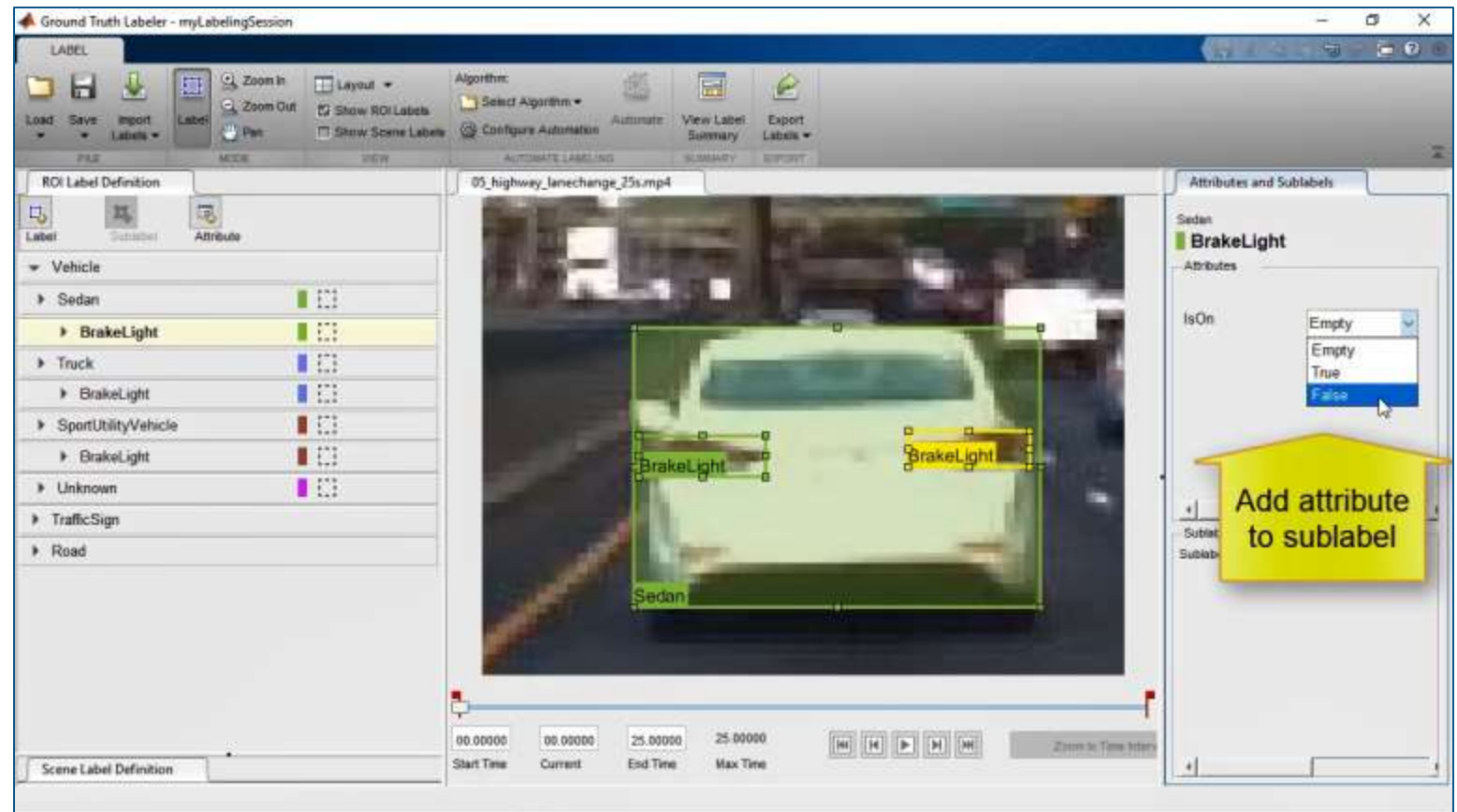
Create sublabels and add attributes

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**



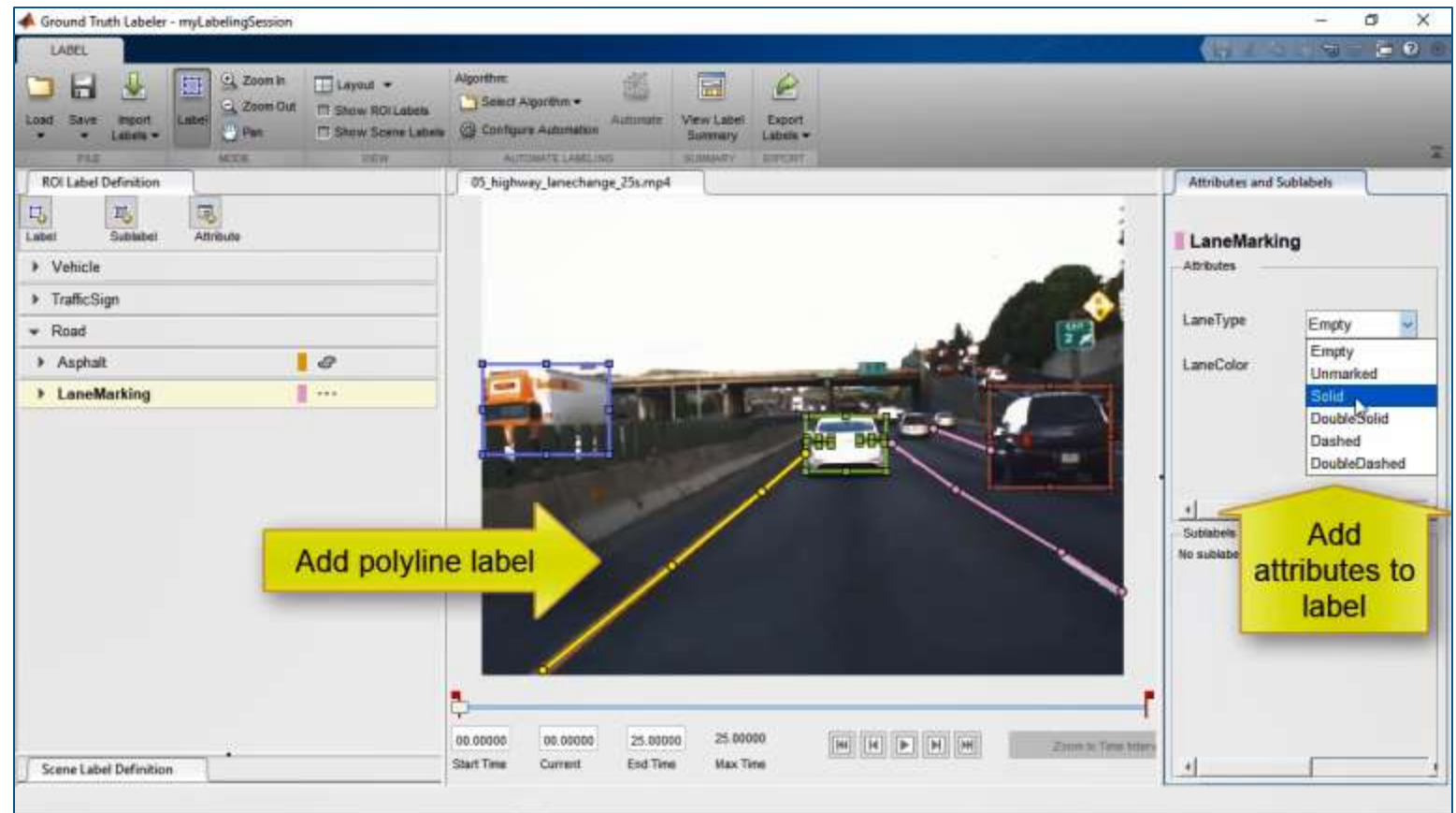
Create polyline labels and add attributes

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**



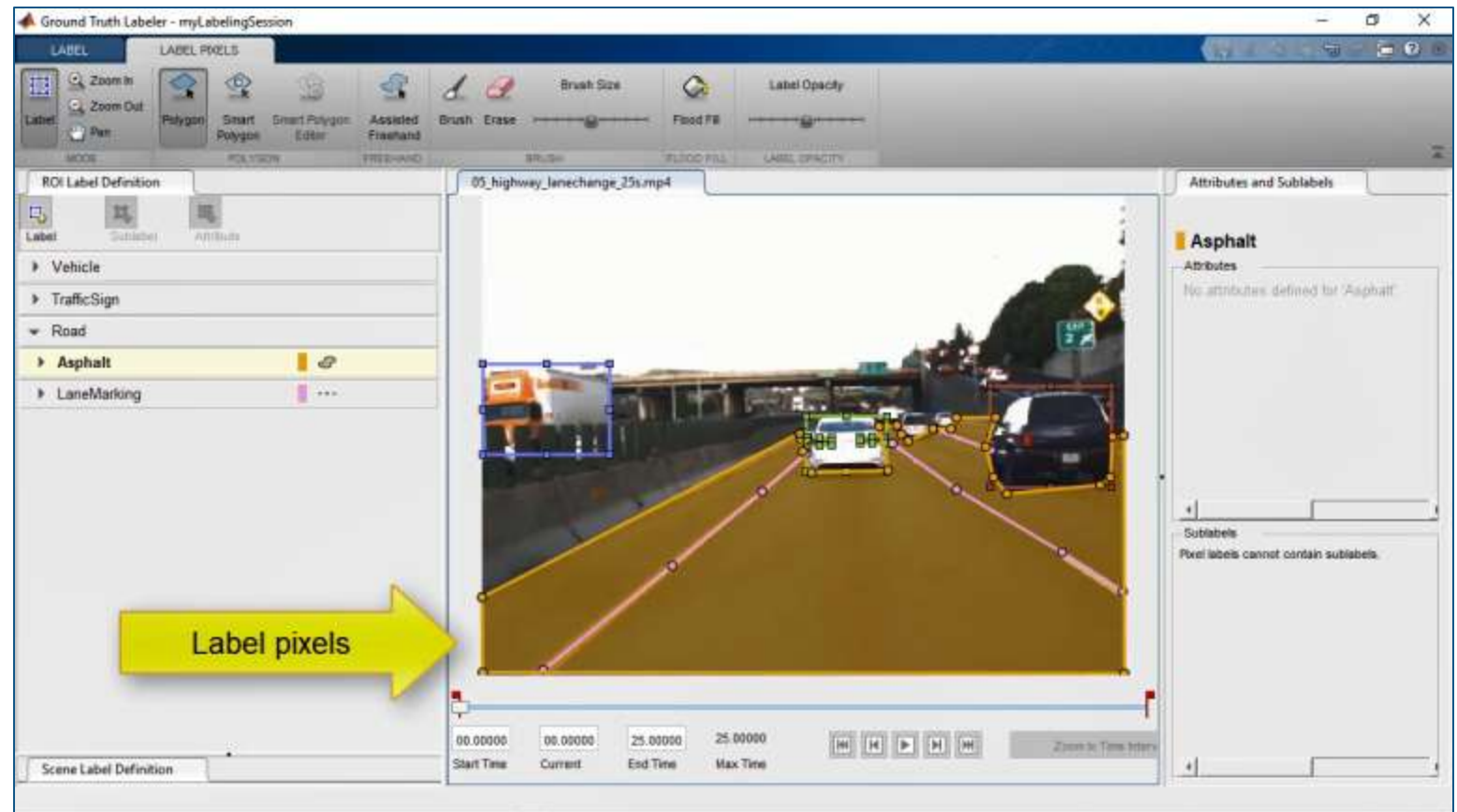
Create pixel labels

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**



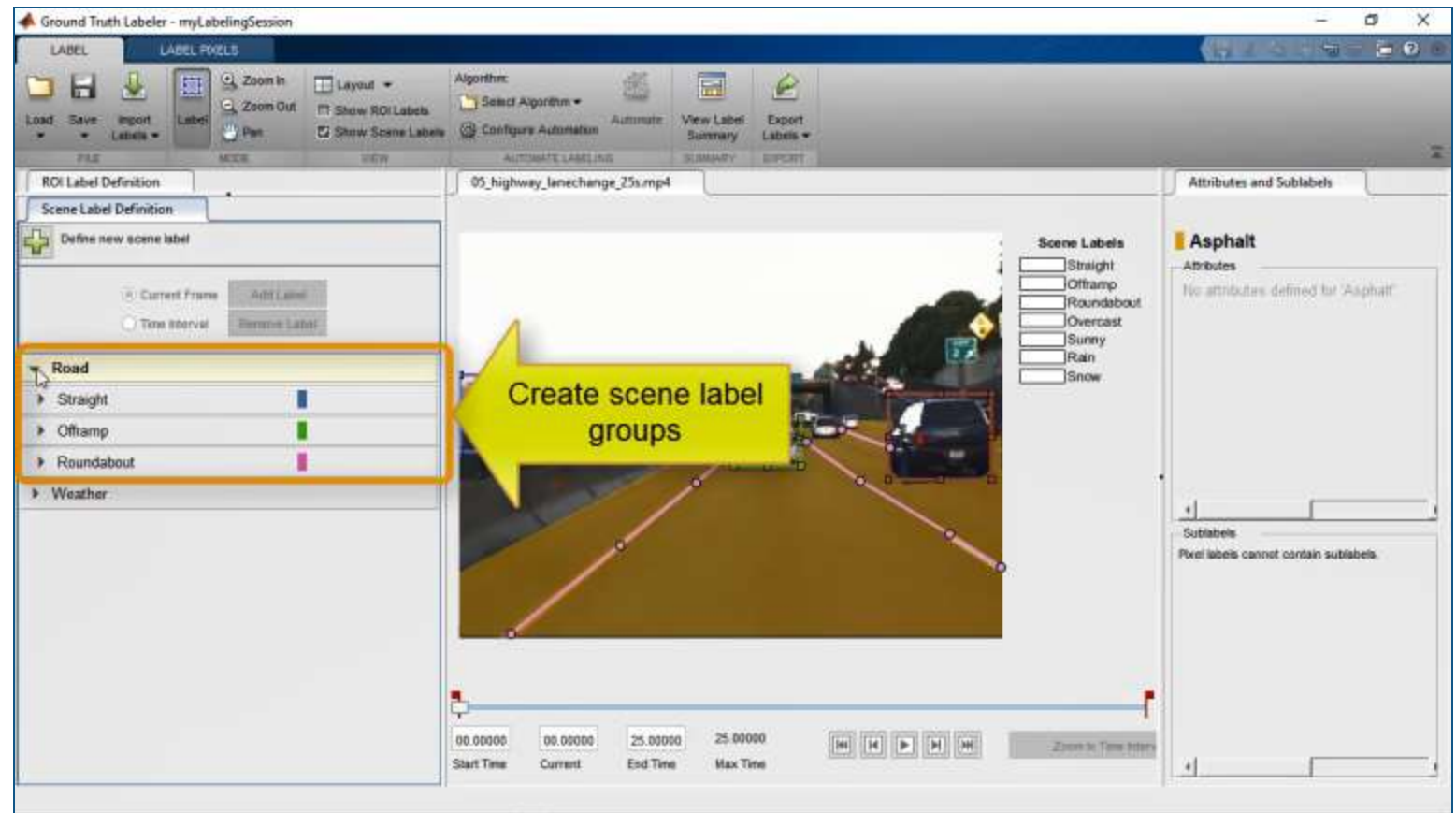
Create scene labels and groups

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**



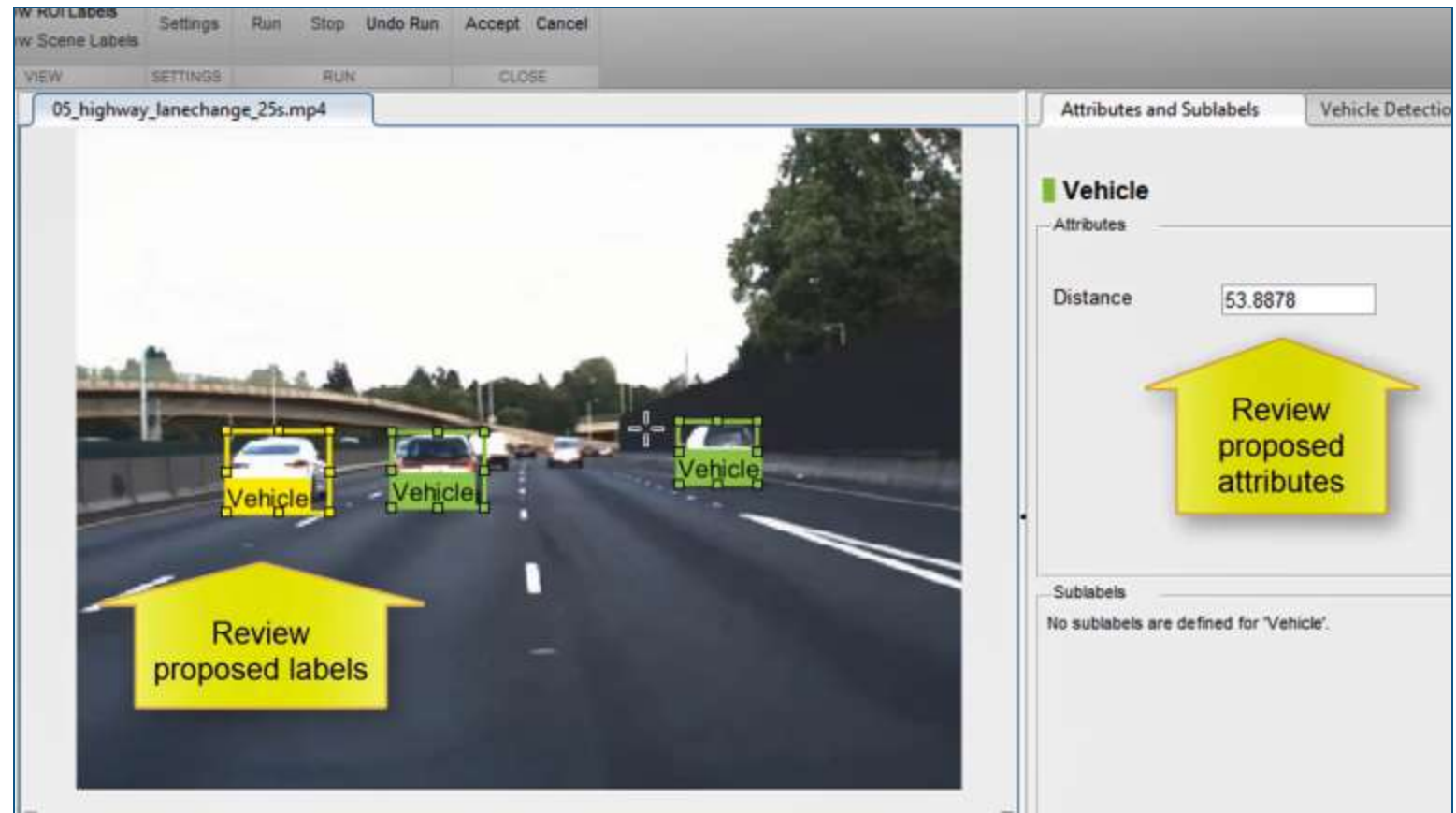
Import custom automation algorithms

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

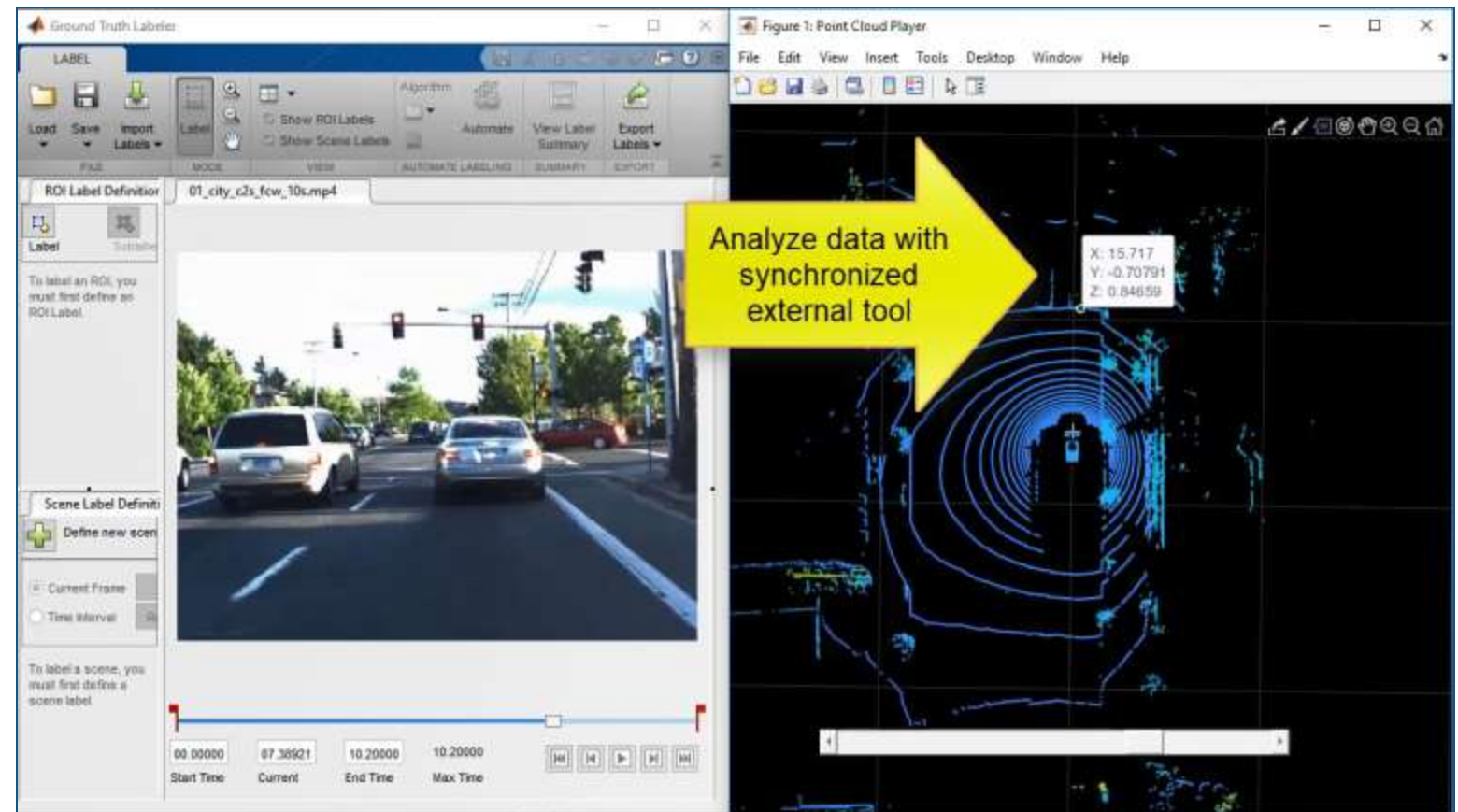


Add custom visualizations for multi-sensor data

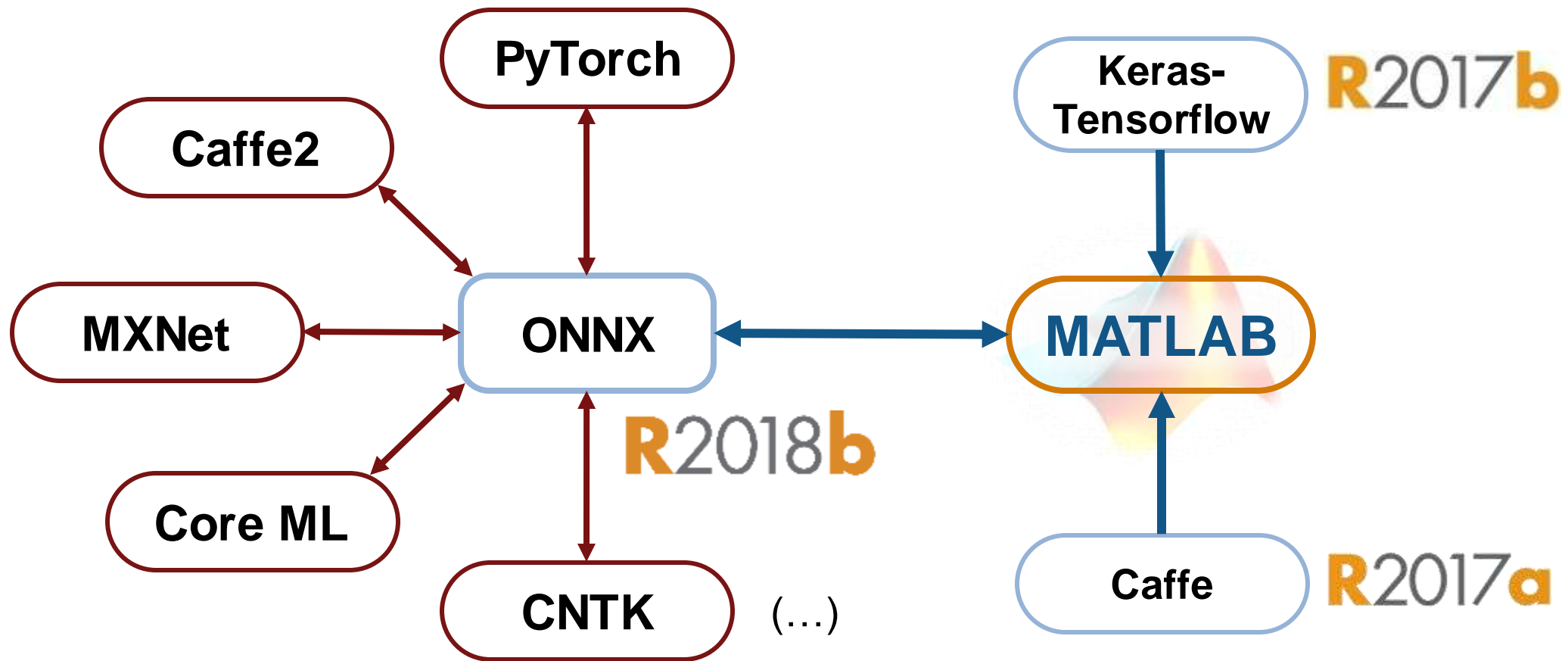
Connect Lidar Display to Ground Truth Labeler

- 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

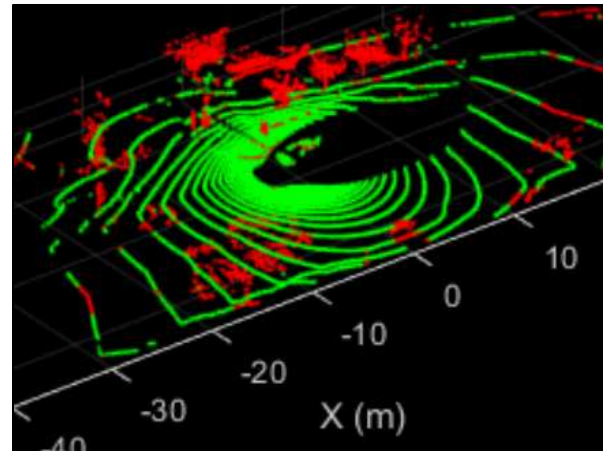


[Object Detection Using YOLO v2 Deep Learning](#)

*Computer Vision Toolbox™
Deep Learning Toolbox™*

R2019a

Detect ground with lidar

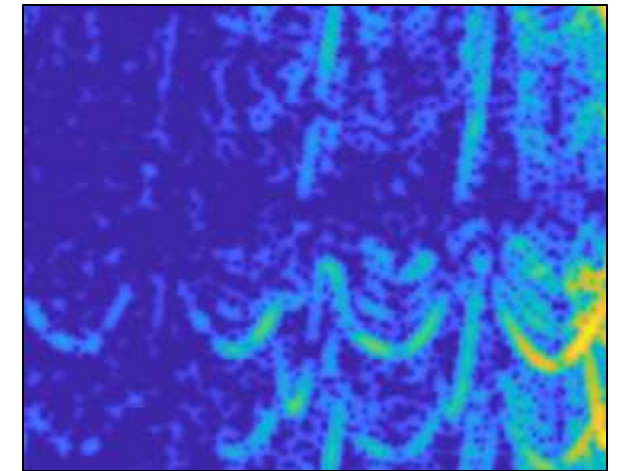


[Segment Ground Points from Organized Lidar Data](#)

Computer Vision Toolbox™

R2018b

Detect pedestrian with radar

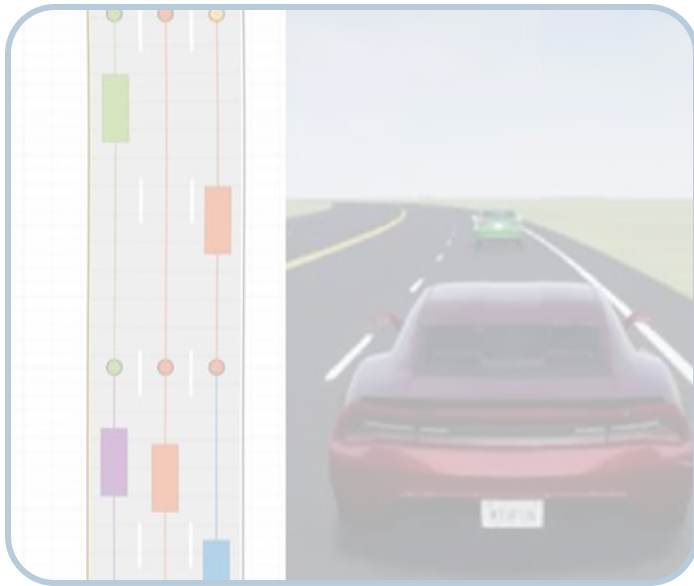


[Introduction to Micro-Doppler Effects](#)

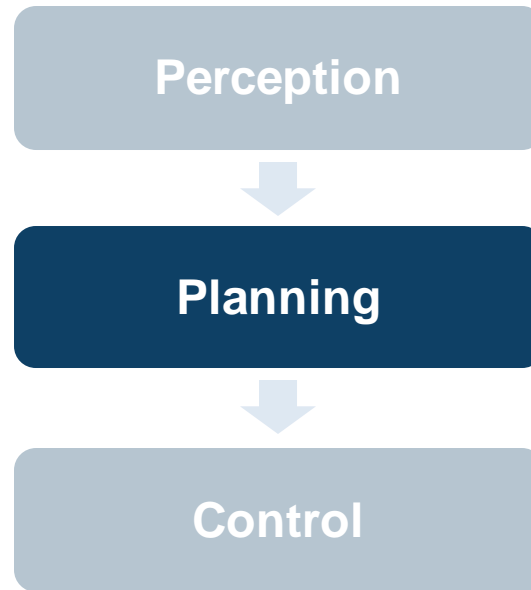
Phased Array System Toolbox™

R2019a

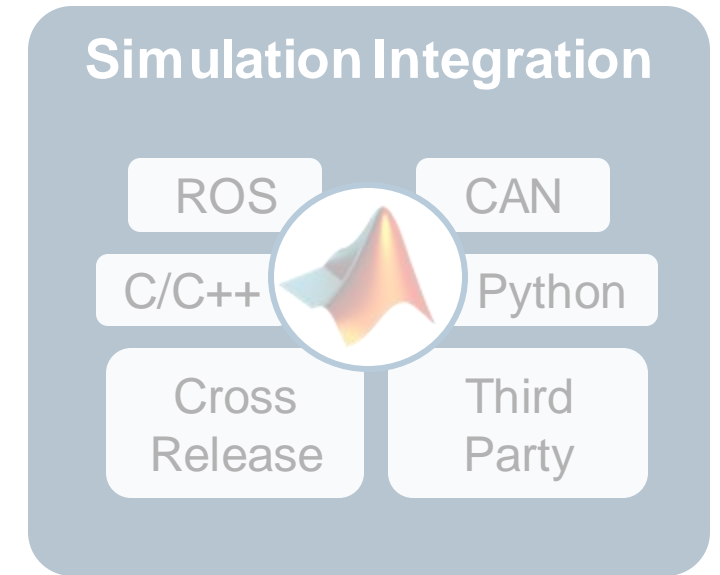
Some common questions from automated driving engineers



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?

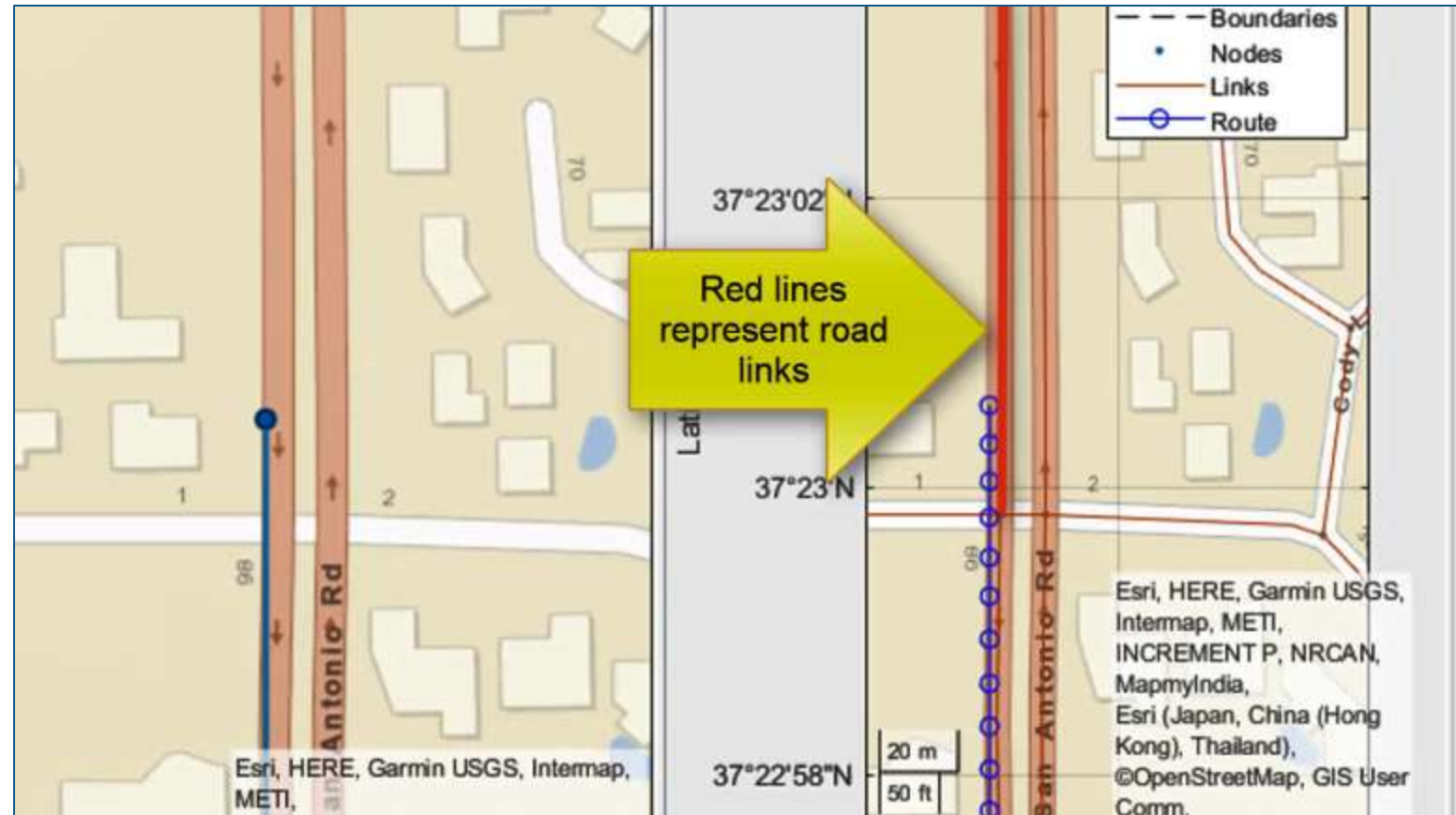
Read road and speed attributes from HERE HD Live Map data

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



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 screenshot displays the MATLAB R2019a environment. The main window shows a live video feed of a road with traffic. To the right, a map view shows the same road with lane configurations overlaid. The map includes a legend for 'Link' and 'Lane Group', a scale bar (20 m, 50 ft), and a copyright notice for HERE, Garmin, USGS, etc. Below the map, a 'Timestamp' field shows 22:11:25 and a 'Speed Limit' field shows 35. At the bottom, a 'Lane Types and Boundaries' section shows a diagram of lane configurations with arrows and a 'BICYCLE' lane. The MATLAB editor window on the left shows a script with the following code:

```

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

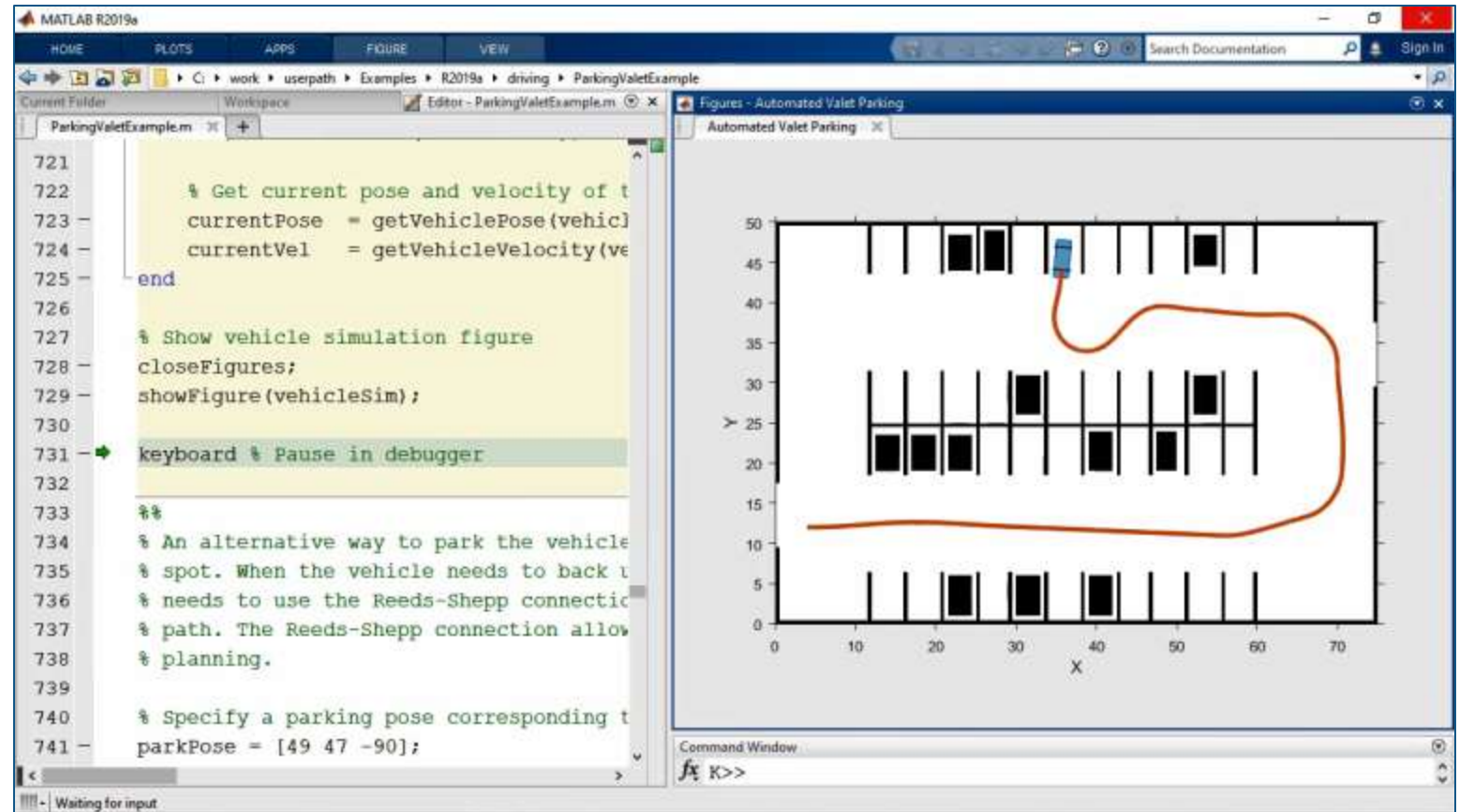
Design path planner

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



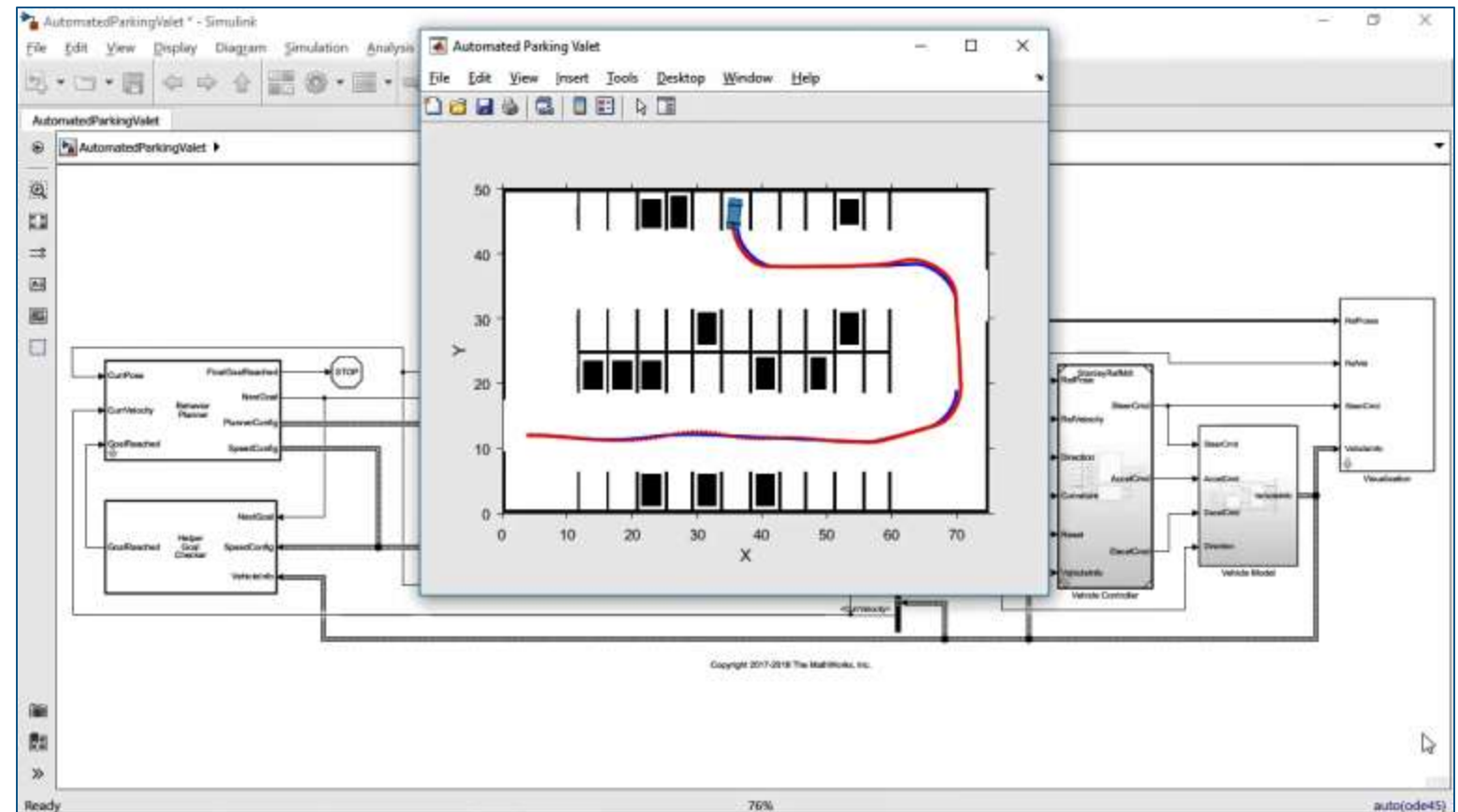
Design path planner and controller

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

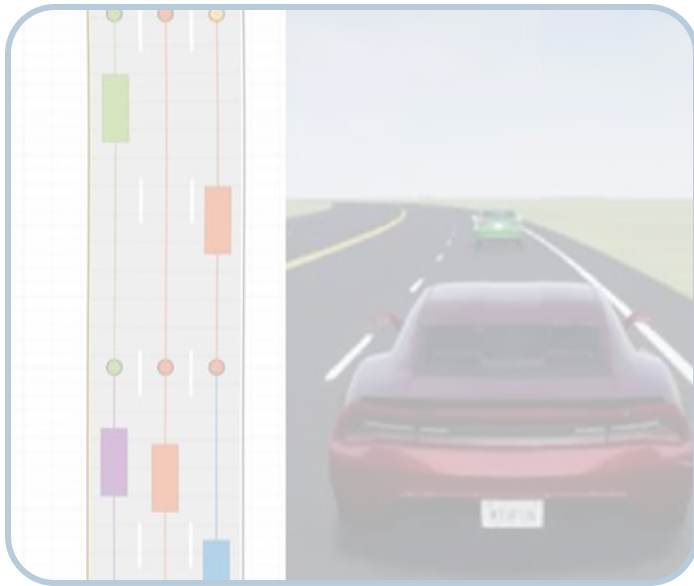
Automated Driving Toolbox™

Embedded Coder

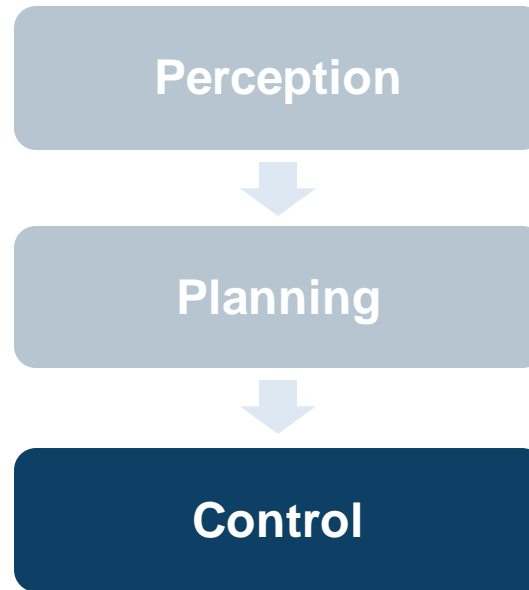
R2019a

```
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
```

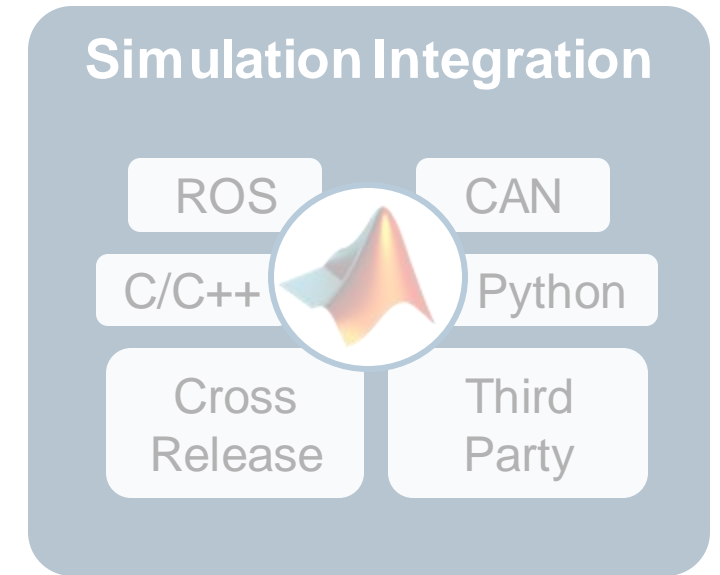

Some common questions from automated driving engineers



How can I
synthesize scenarios
to test my designs?



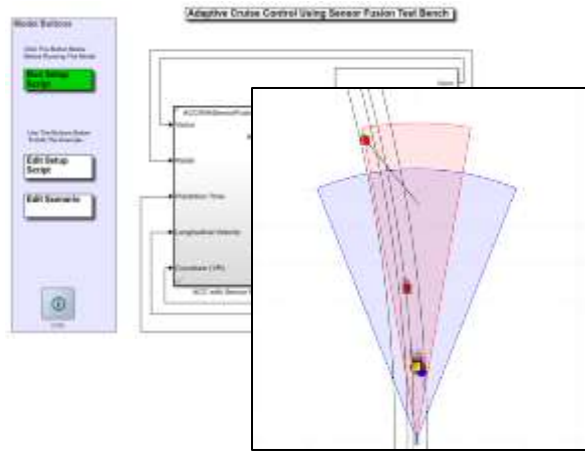
How can I
discover and design
in multiple domains?



How can I
integrate
with other environments?

Design lateral and longitudinal Model Predictive Controllers

Longitudinal Control



[Adaptive Cruise Control with Sensor Fusion](#)

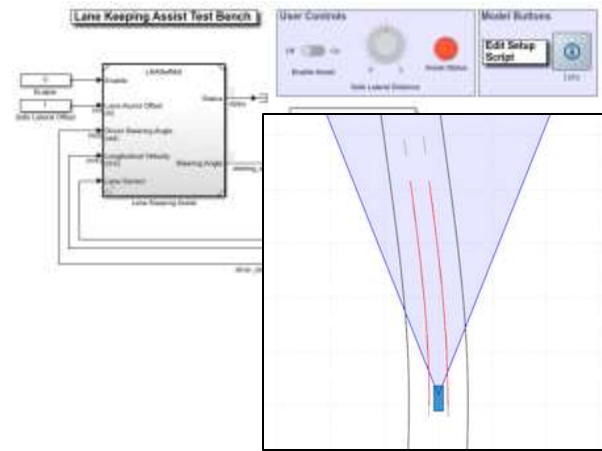
Automated Driving Toolbox™

Model Predictive Control Toolbox™

Embedded Coder®

R2017b

Lateral Control



[Lane Keeping Assist with Lane Detection](#)

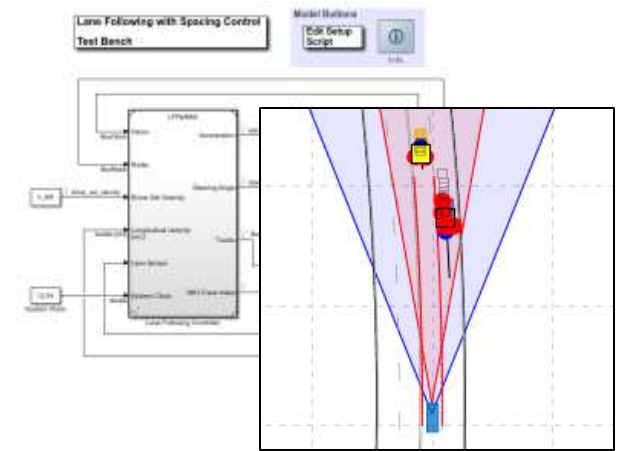
Automated Driving Toolbox™

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

Develop automatic emergency braking application

Automatic Emergency Braking (AEB) with Sensor Fusion

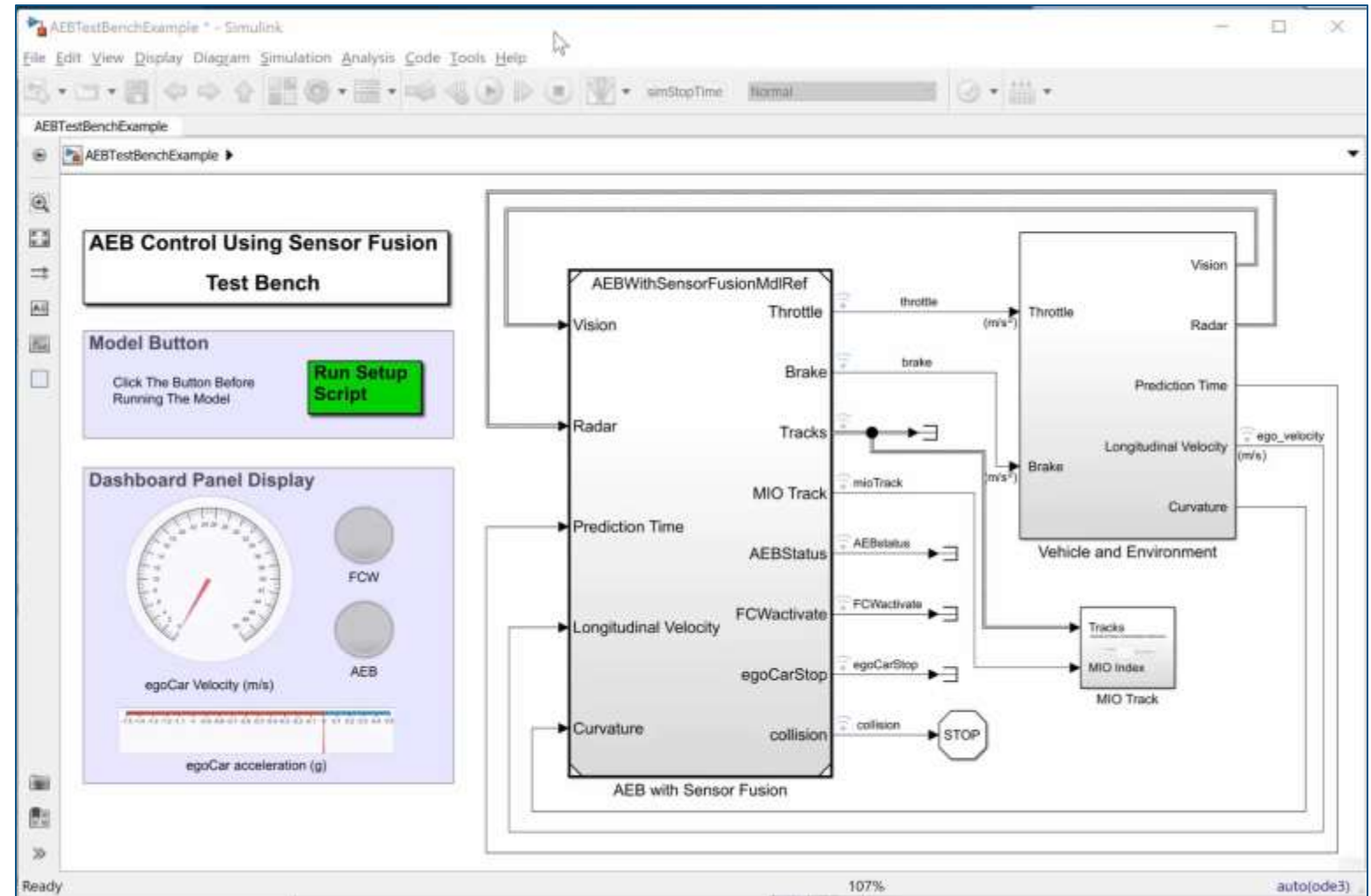
- Specify driving scenario
- Design AEB logic
- Integrate sensor fusion
- Visualize sensors and tracks
- Generate C/C++ code
- Test with software in the loop (SIL) simulation

Automated Driving Toolbox™

Stateflow®

Embedded Coder®

R2018b



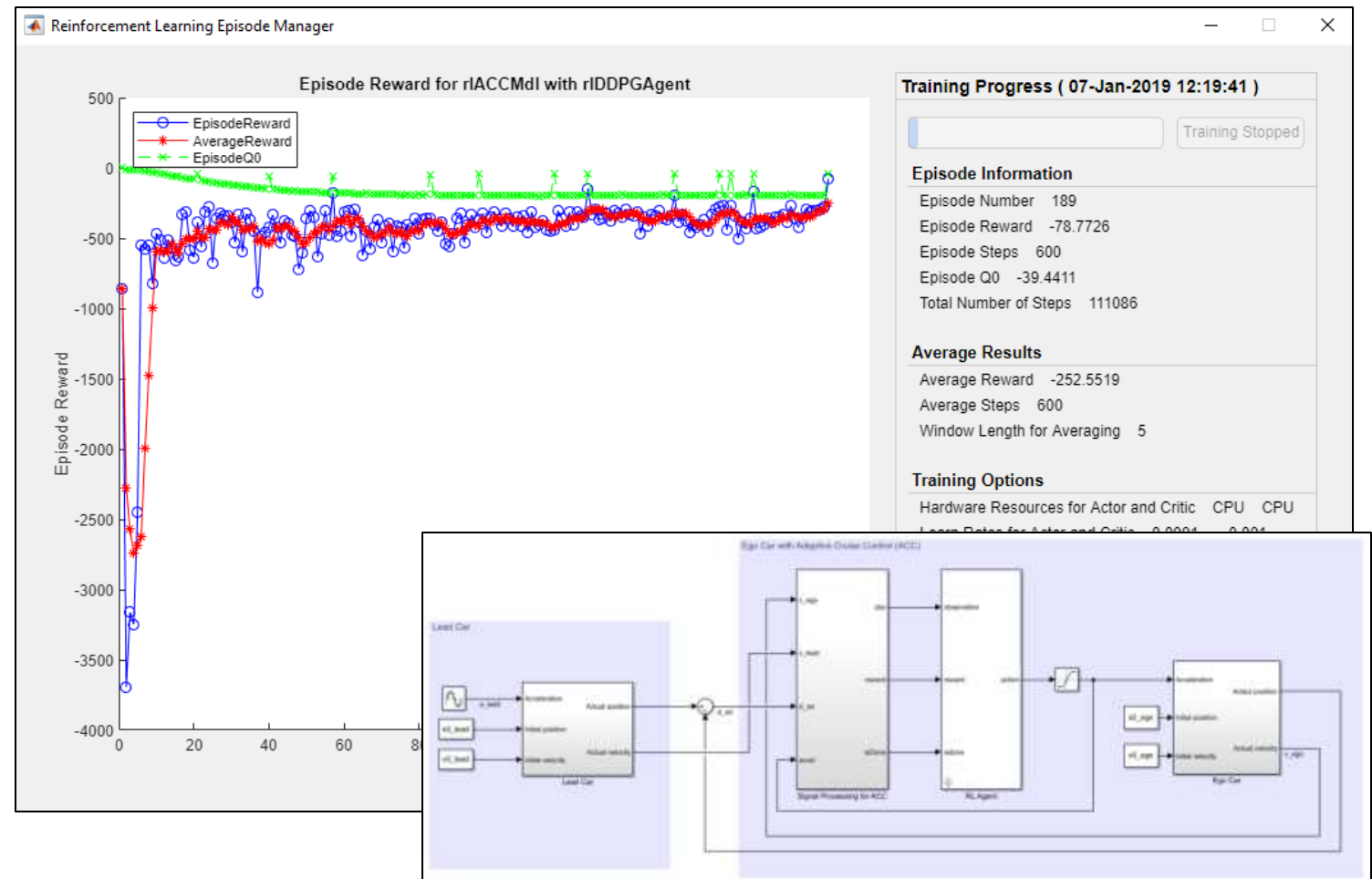
Train reinforcement learning networks for ADAS controllers

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

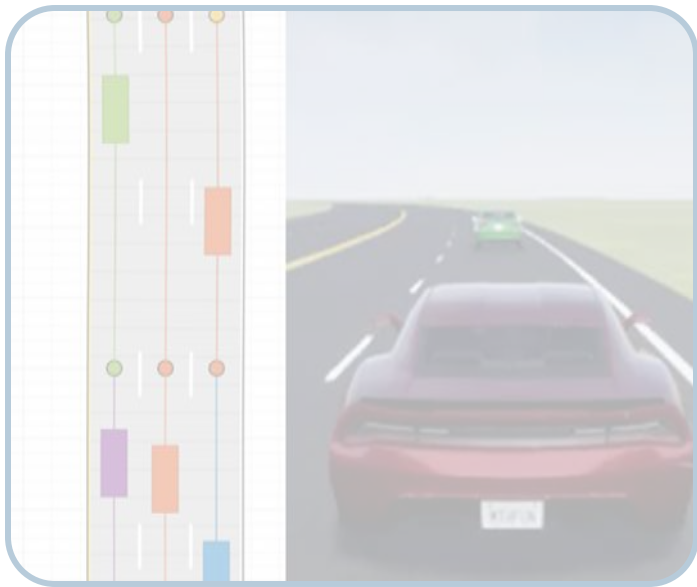
- Create environment interface
- Create agent
- Train agent
- Simulate trained agent

Reinforcement Learning Toolbox™

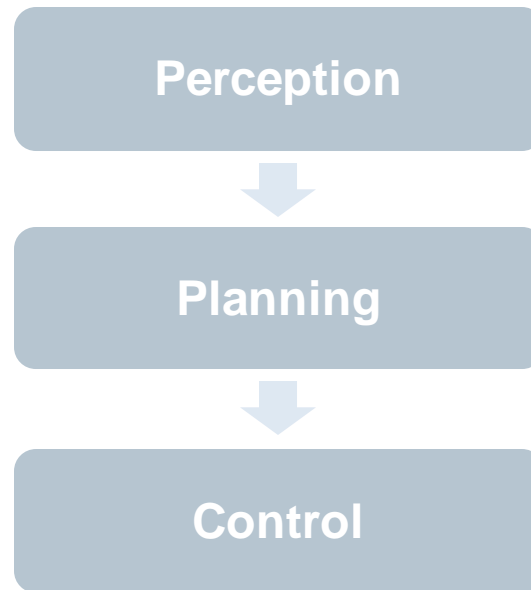
R2019a



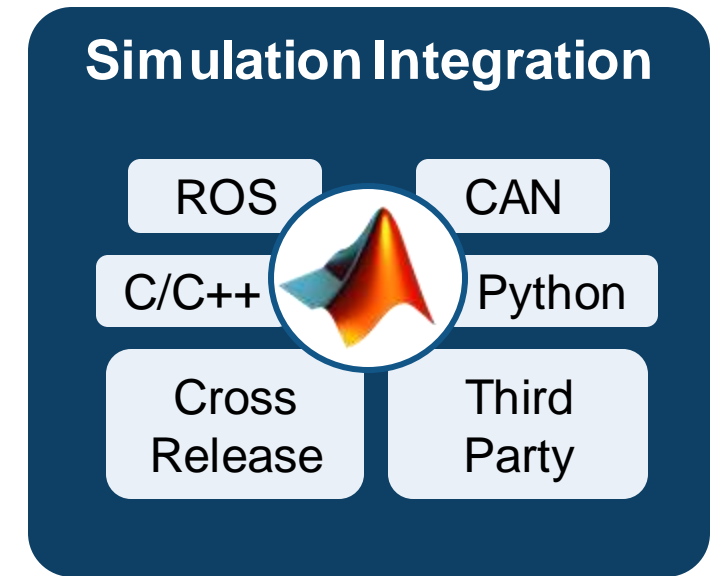
Some common questions from automated driving engineers



How can I
synthesize scenarios
to test my designs?



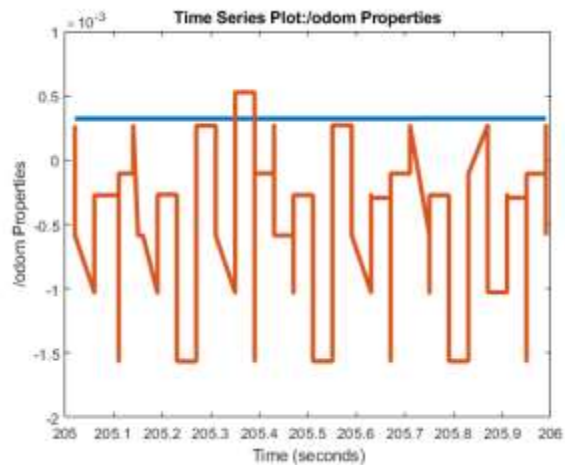
How can I
discover and design
in new domains?



How can I
integrate
with other environments?

Integrate with ROS

Replay logged ROS data

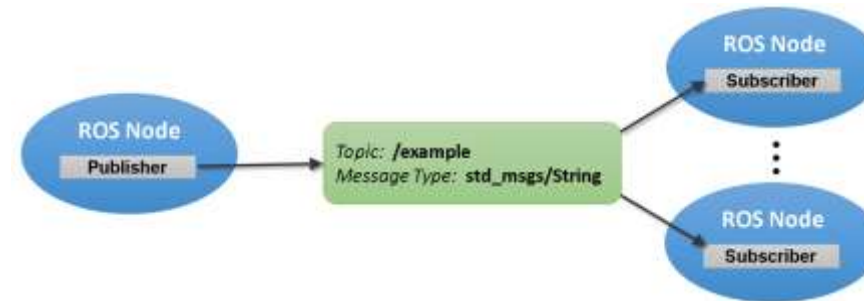


[Work with rosbag Logfiles](#)

Robotic System Toolbox™

Updated **R2018a**

Connect to live ROS data

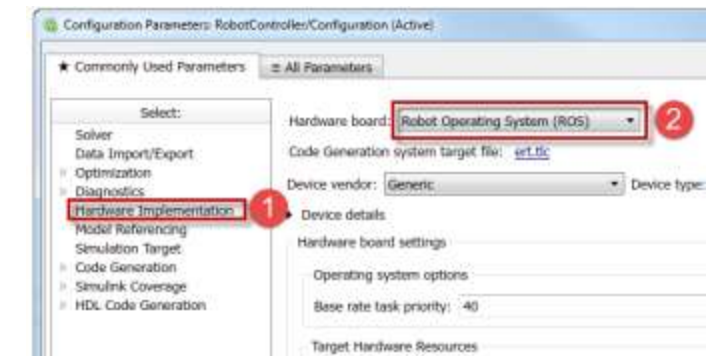


[Exchange Data with ROS Publishers and Subscribers](#)

Robotic System Toolbox™

R2016b

Generate standalone ROS node



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

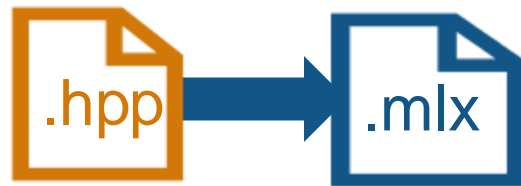
Robotic System Toolbox™

Simulink Coder™

R2016b

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](#)

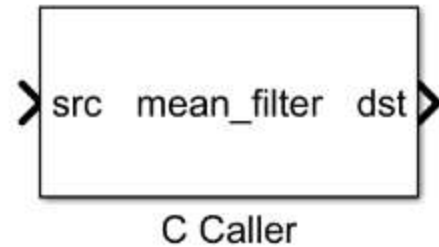
Computer Vision System Toolbox™

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;
```

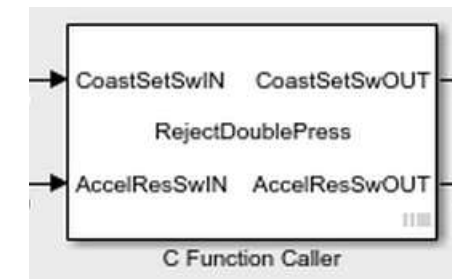
Name	DataType
-coeff	double
-init	double
-fault	Enum: fault_T

[Import Structure and Enumerated Types](#)

Simulink®

R2017a

Test and verify C code



ANALYZED MODEL	DECISION	CONDITION	MCDC
RejectDoublePress.c	100%	100%	100%

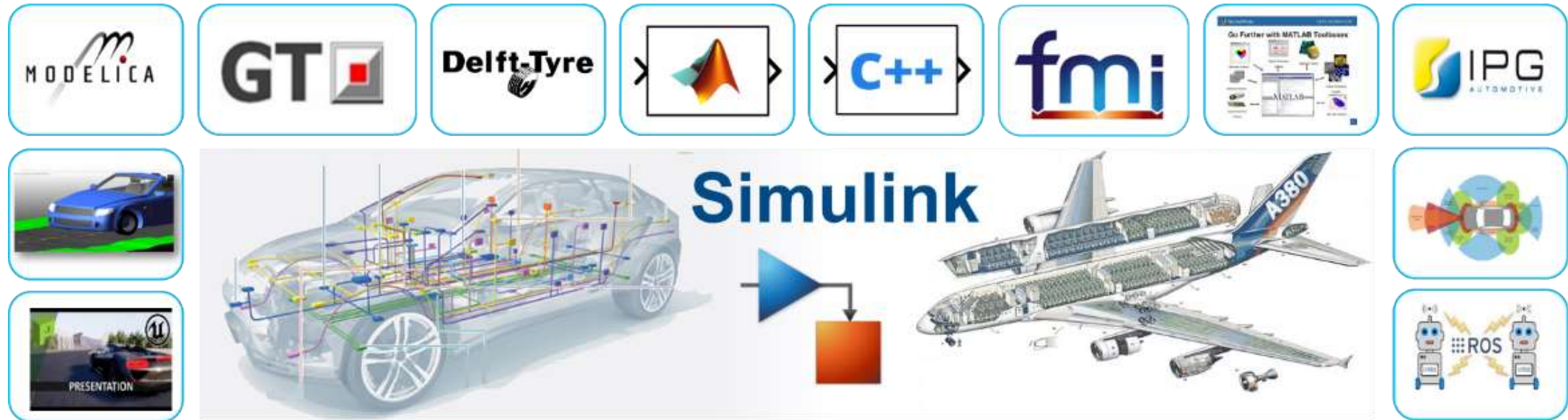
[Custom C Code Verification with Simulink Test](#)

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

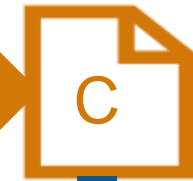
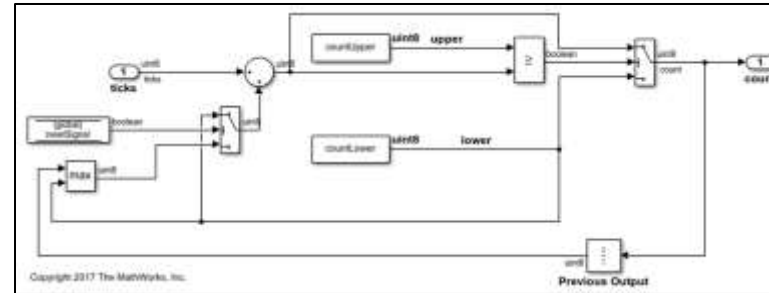
Integrate Generated Code by Using Cross-Release Workflow

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

Embedded Coder

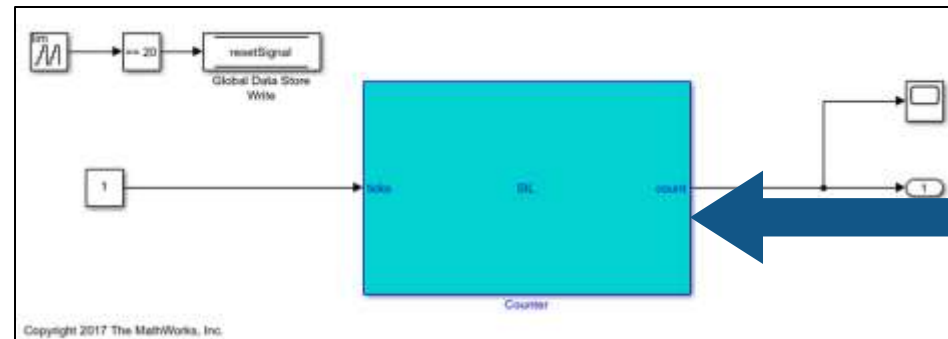
R2016a

Previous Release

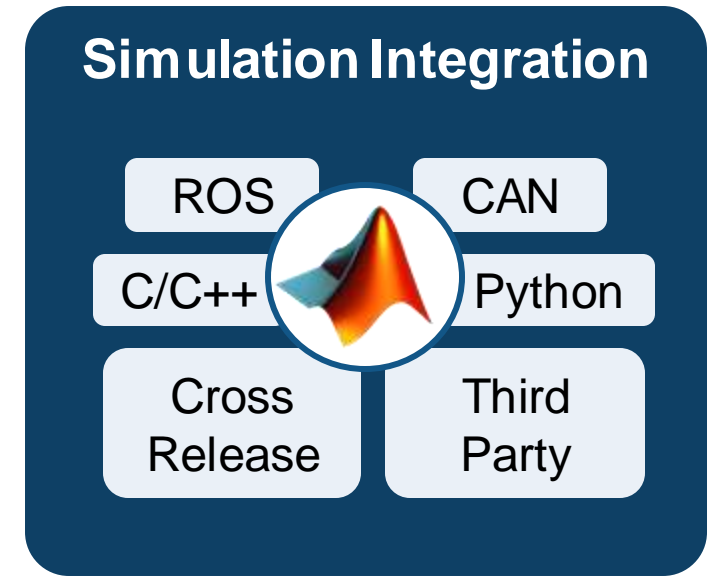
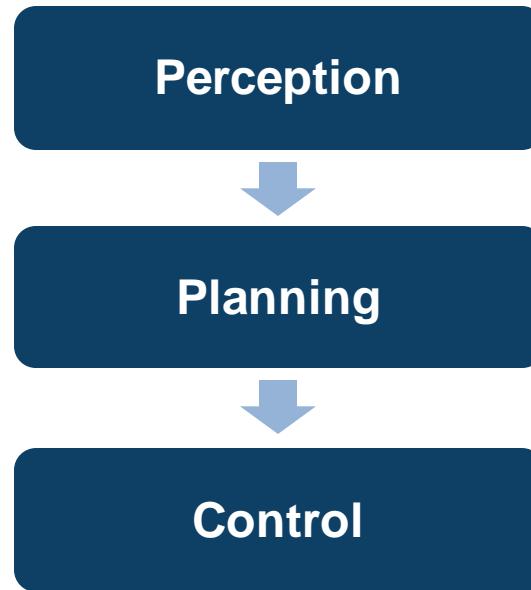
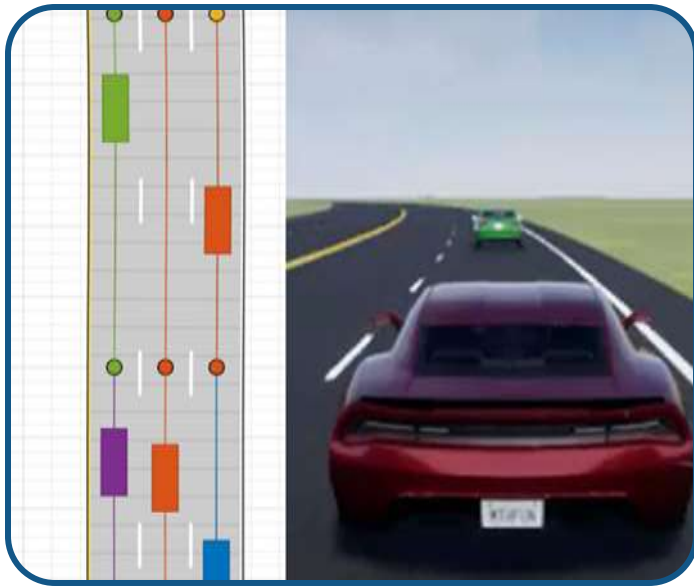


crossReleaseImport

Current Release



Some common questions from automated driving engineers

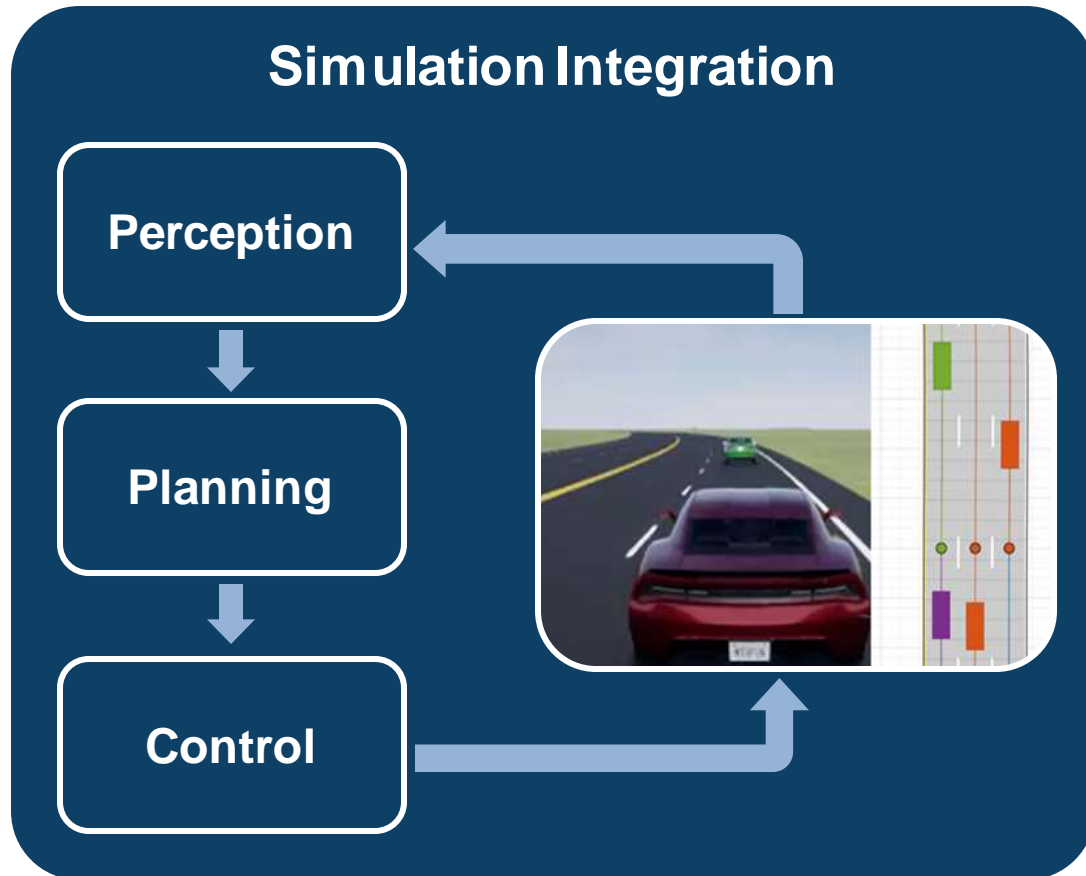


Synthesize scenarios
to test my designs

Discover and design
in multiple domains

Integrate
with other environments

Get started on your own with documented examples



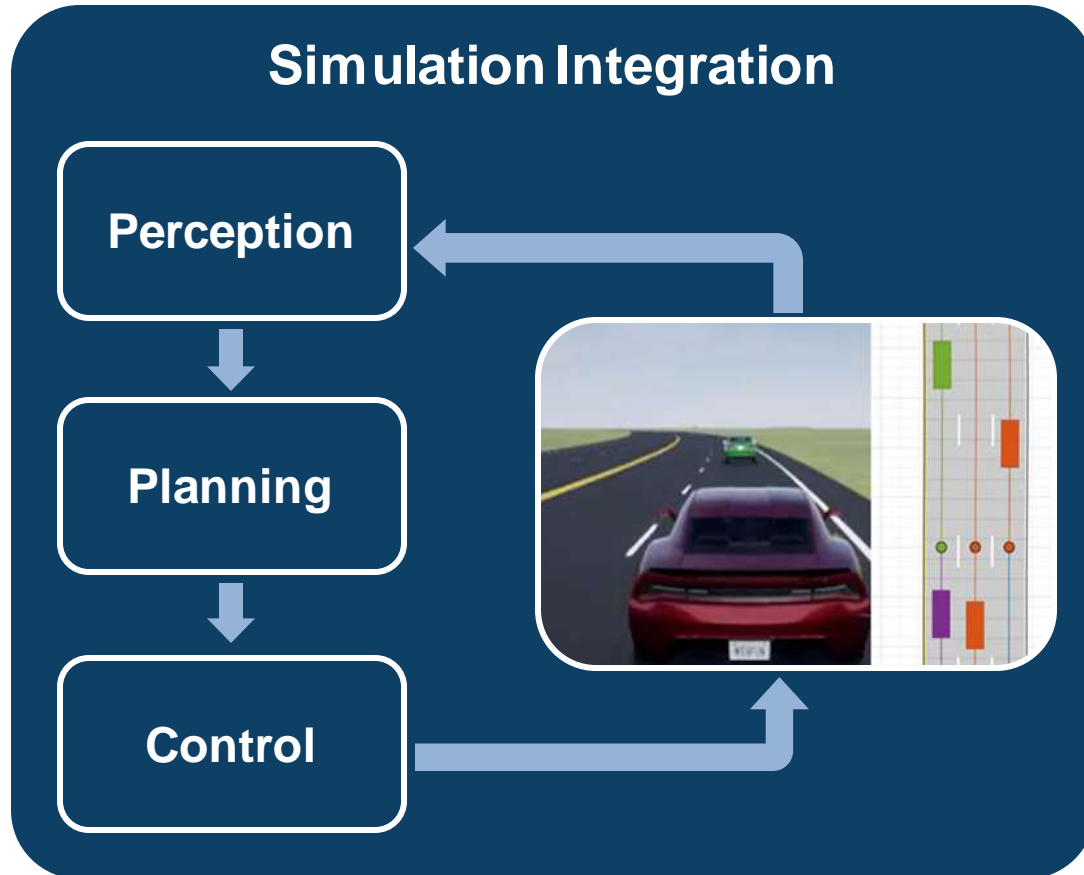
Documentation

All

Examples

- [Automated Driving Toolbox](#)
 - Labeling, perception, sensor fusion, path planning, synthetic sensor data
- [Model Predictive Control Toolbox](#)
(Section: [Automated Driving Applications](#))
 - Adaptive cruise control, lane keeping, lane following with spacing control
- [Simulink Test](#)
(Section: [Systematic Testing and Reporting](#))
 - Test lane following controller with sensor fusion

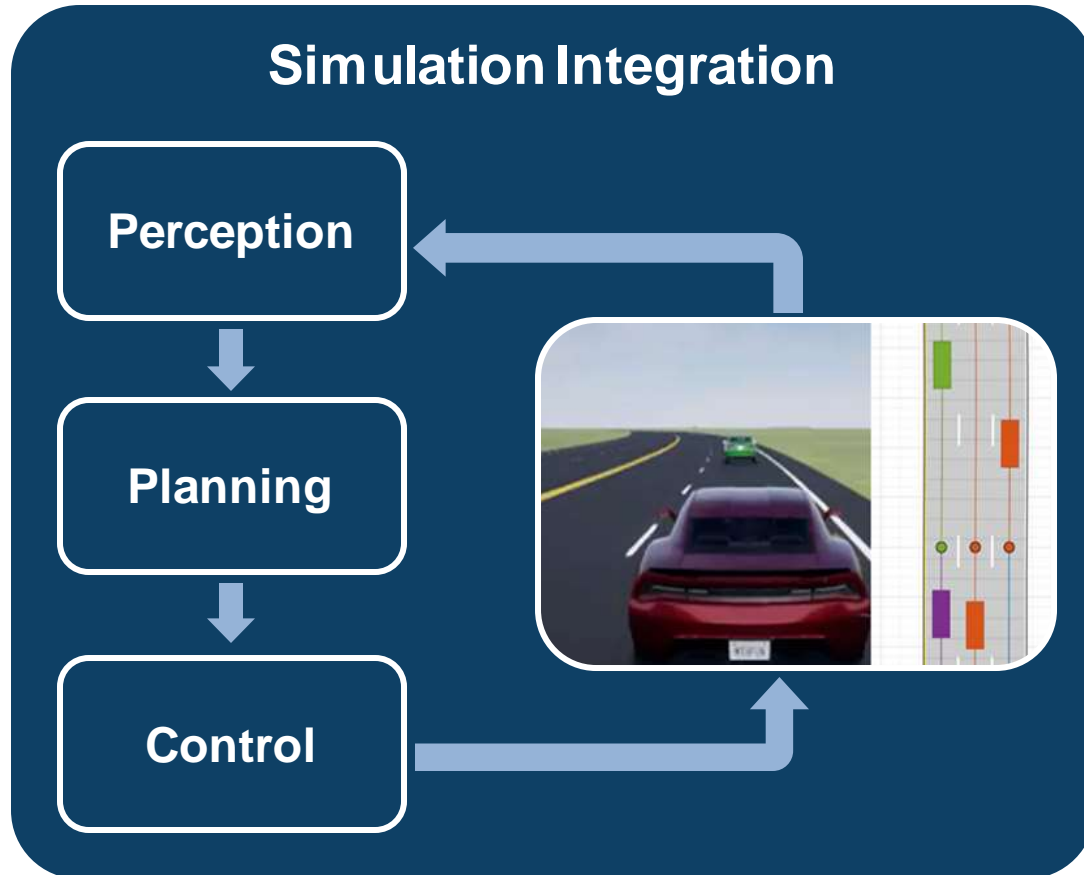
Gain tool experience with Training Services



MATLAB and Simulink Training

- [Automated Driving with MATLAB](#)
- [Deep Learning with MATLAB](#)
- [Computer Vision with MATLAB](#)
- [Simulink for System and Algorithm Modeling](#)
- [Integrating Code with Simulink](#)
- [Code Generation for AUTOSAR Software](#)
- [Verification and Validation of Simulink Models](#)
- [Polyspace Bug Finder for C/C++ Code Analysis](#)
- Ask about customizing training courses for your needs ([contact training](#))

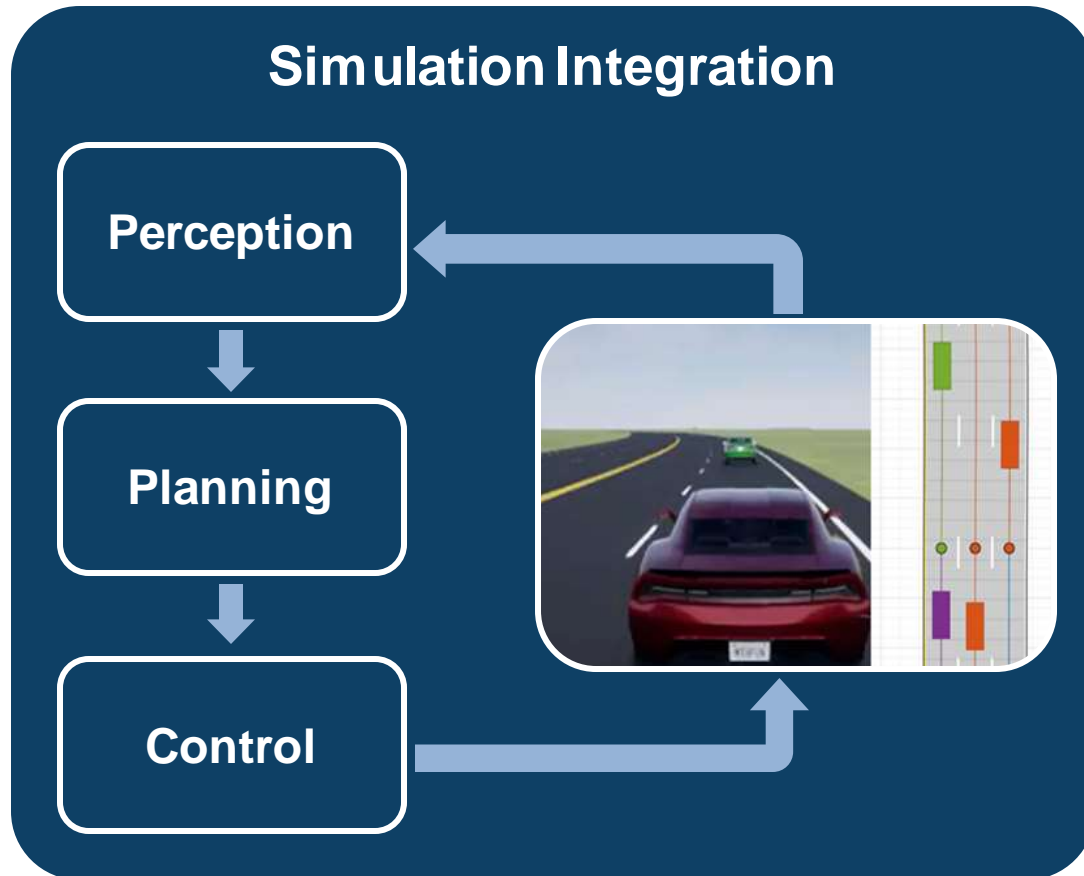
Partner on your projects with Consulting Services



MATLAB and Simulink Consulting Services

- [Image Processing and Computer Vision](#)
- [MATLAB with Hadoop and Spark](#)
- [Tools Integration](#)
- [ISO 26262 Process Deployment Advisory Service](#)
- [Model-Based Design Process Establishment](#)
- [Model-Based Design Process Assessment and Maturity Framework](#)
- Ask about extending tools for labeling or synthesizing sensor data ([contact consulting](#))

Get started developing automated driving systems with MATLAB and Simulink



Discuss your application with me (mark.corless@mathworks.com) or a MathWorks field engineer to help you structure an evaluation

- Understand your goals
- Recommend tasks
- Answer questions