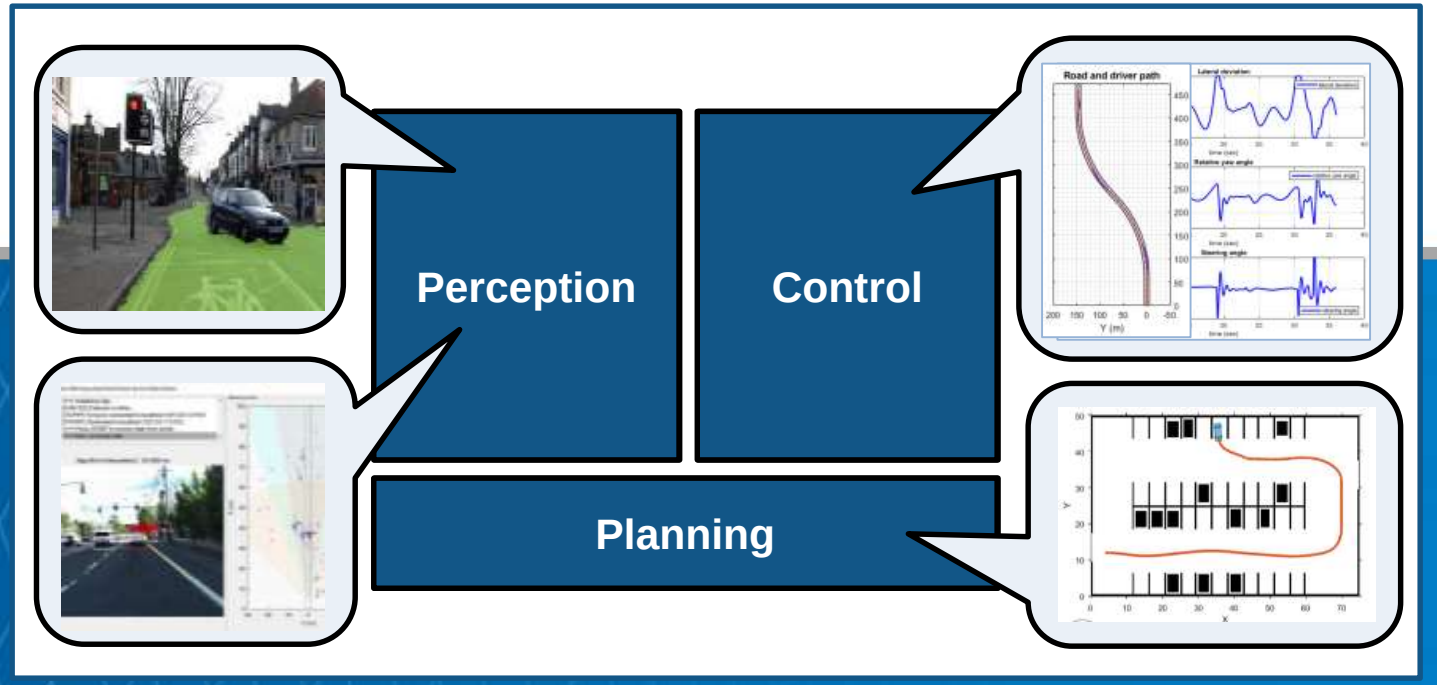
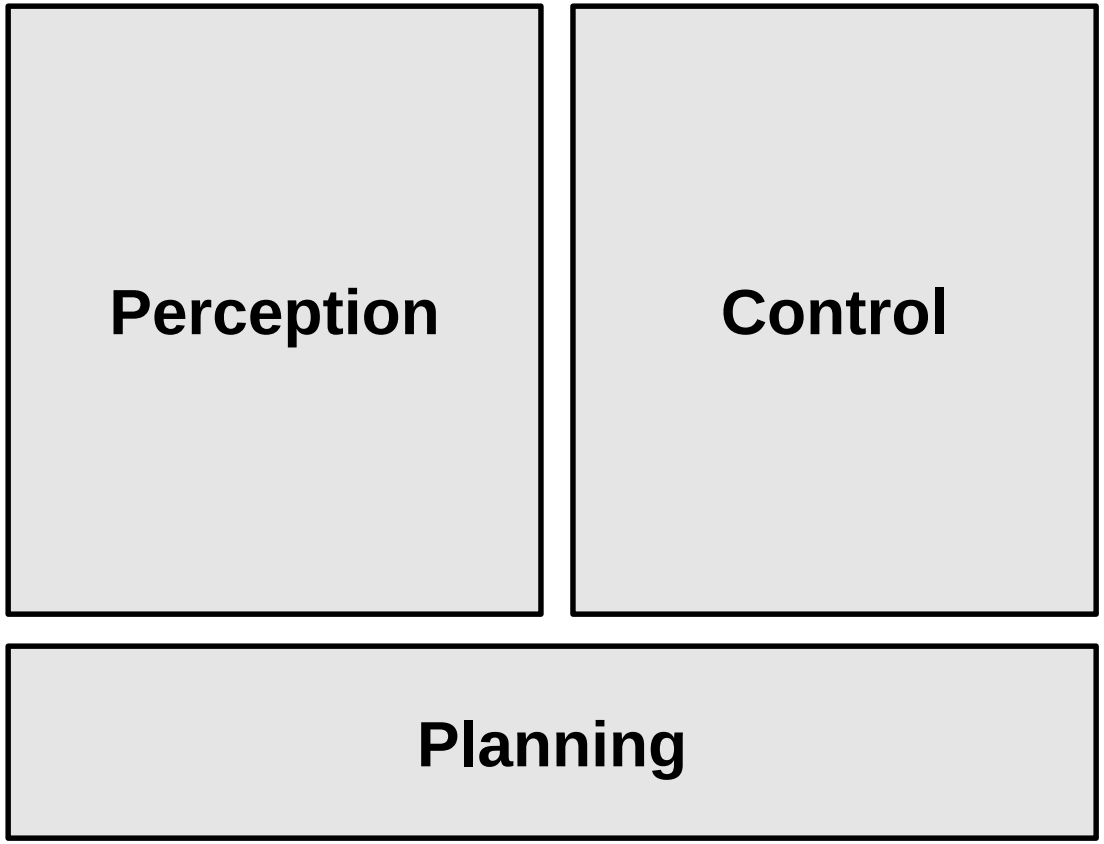
What's new in MATLAB® and Simulink® for

Automated Driving

Mark Corless
Automated Driving Segment Manager
Industry Marketing
2018-05-02



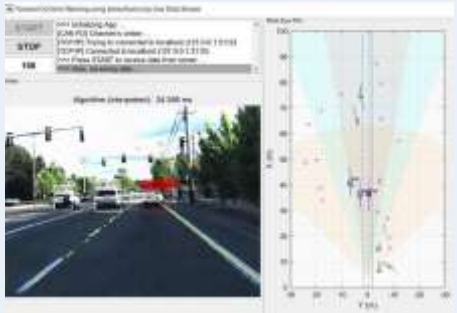
How can you use MATLAB and Simulink to develop automated driving algorithms?



Examples of how you can use MATLAB and Simulink to develop automated driving algorithms

Deep learning

Sensor fusion with live data

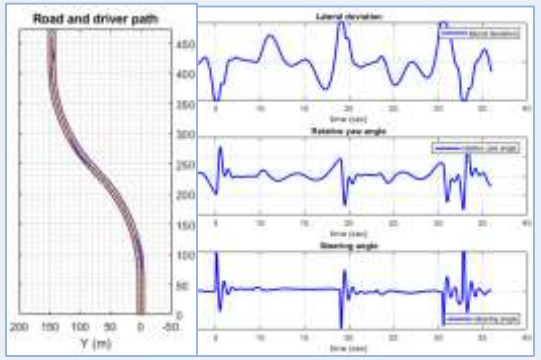


Perception

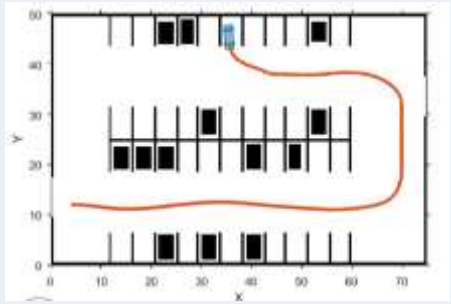
Control

Planning

Sensor models & model predictive control



Path planning

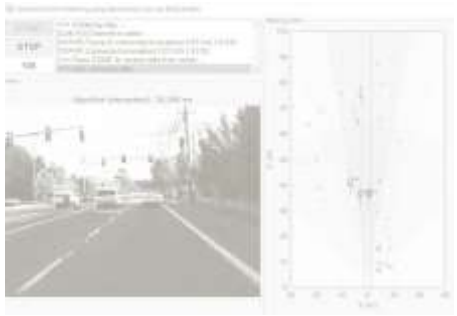


How can you use MATLAB and Simulink to develop perception algorithms?

Deep learning

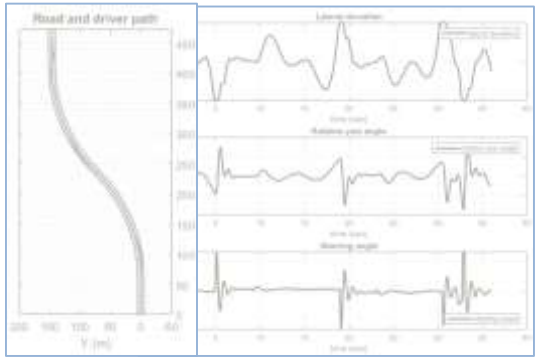
Perception

Sensor fusion with live data



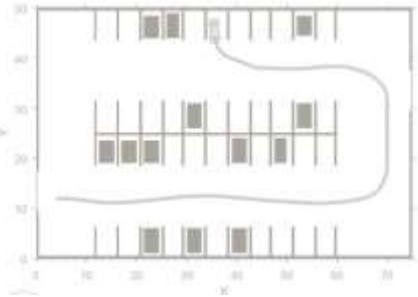
Control

Sensor models & model predictive control



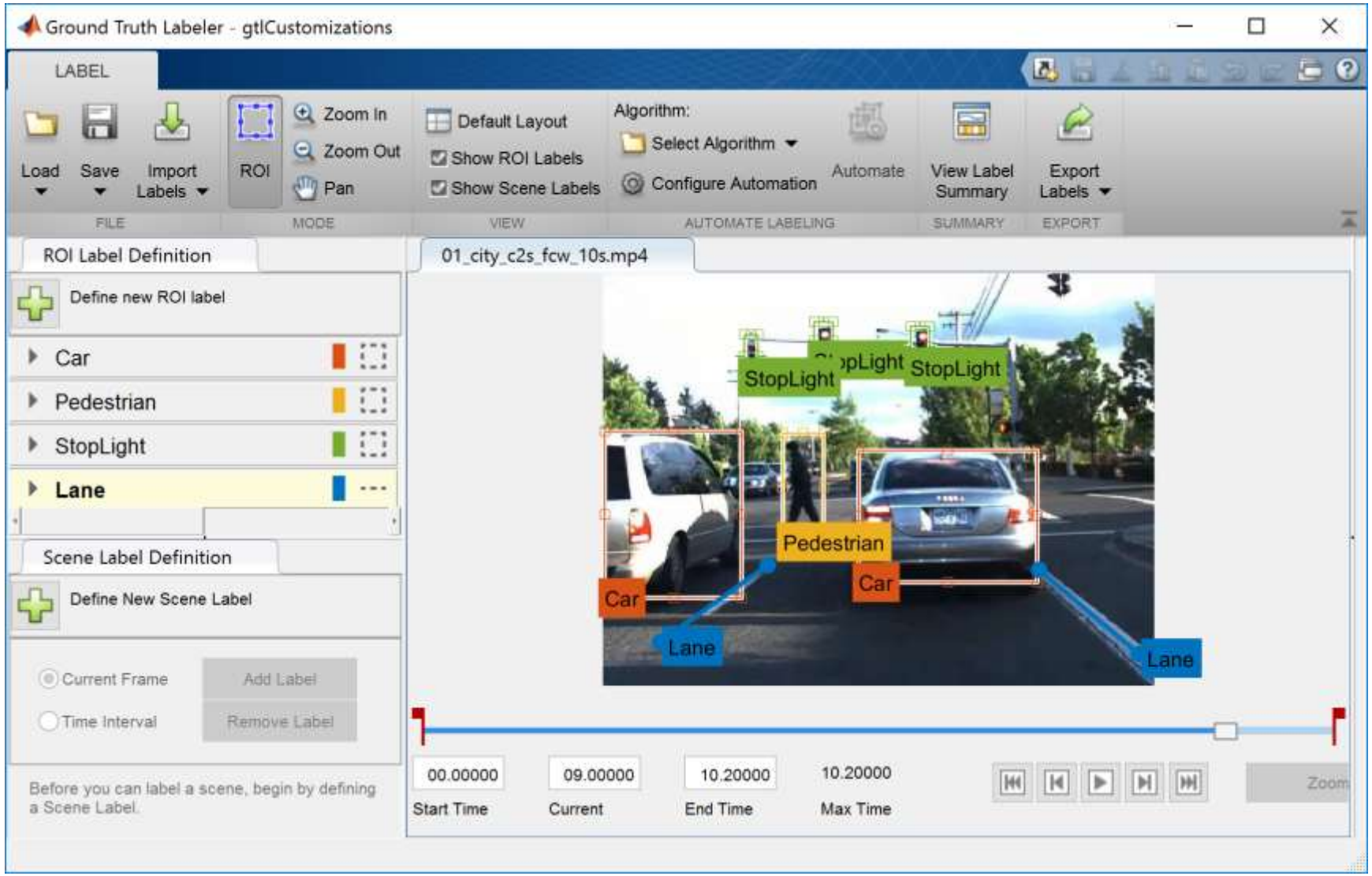
Planning

Path planning



Automated Driving System Toolbox introduced:

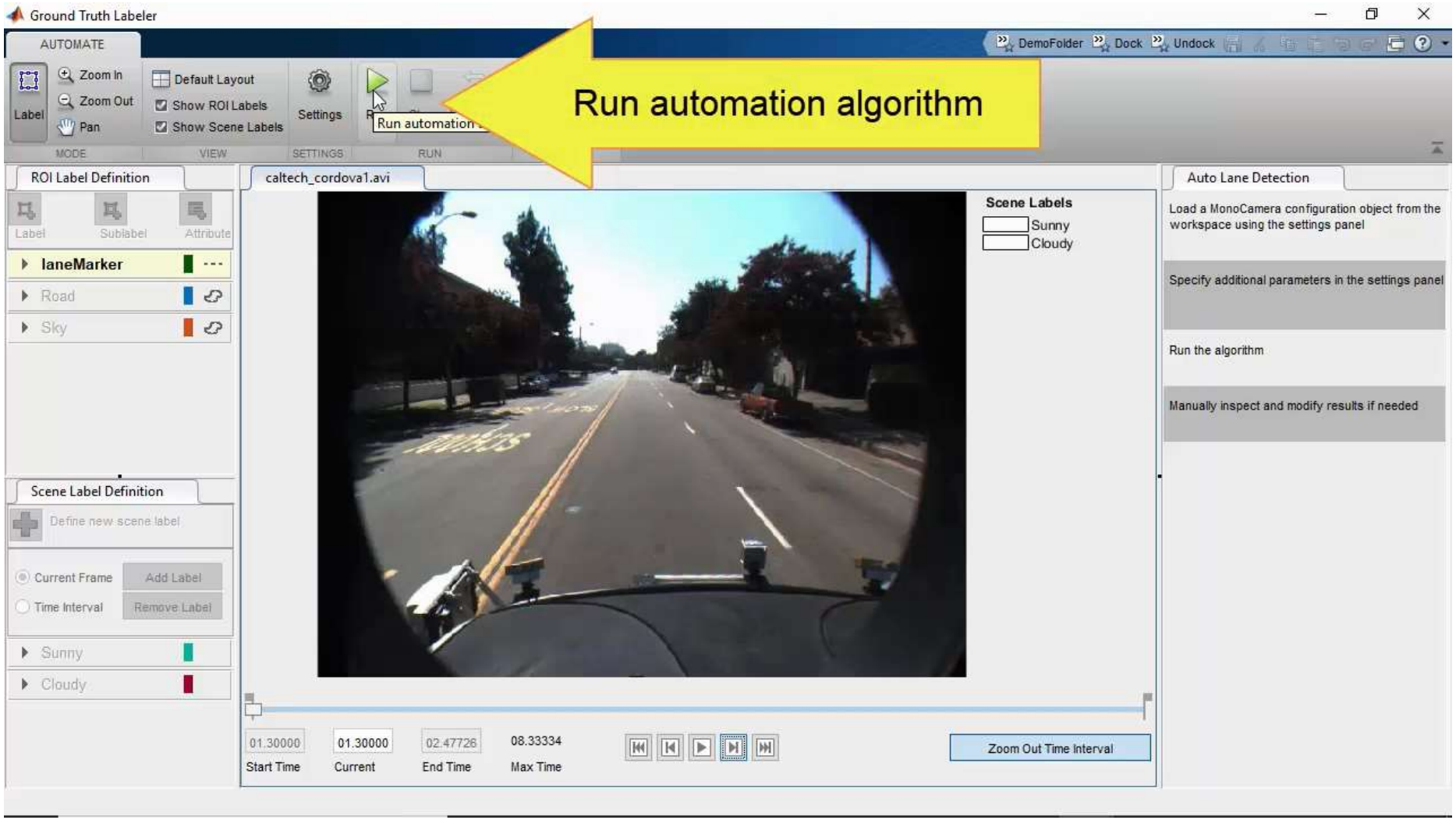
Ground Truth Labeling App to label video data



R2017a

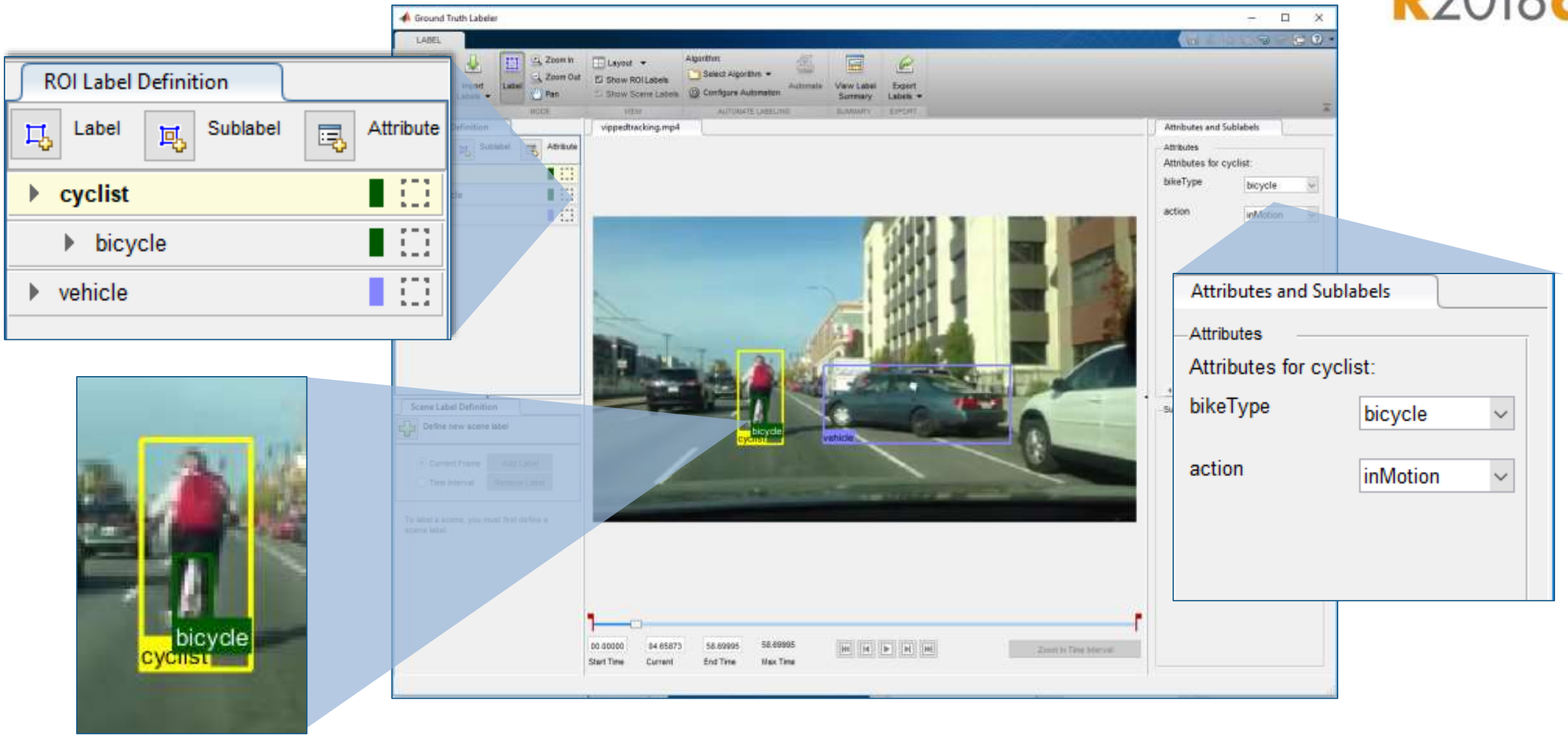
Introduction to Automated Driving System Toolbox

Automate labeling lanes with Ground Truth Labeler

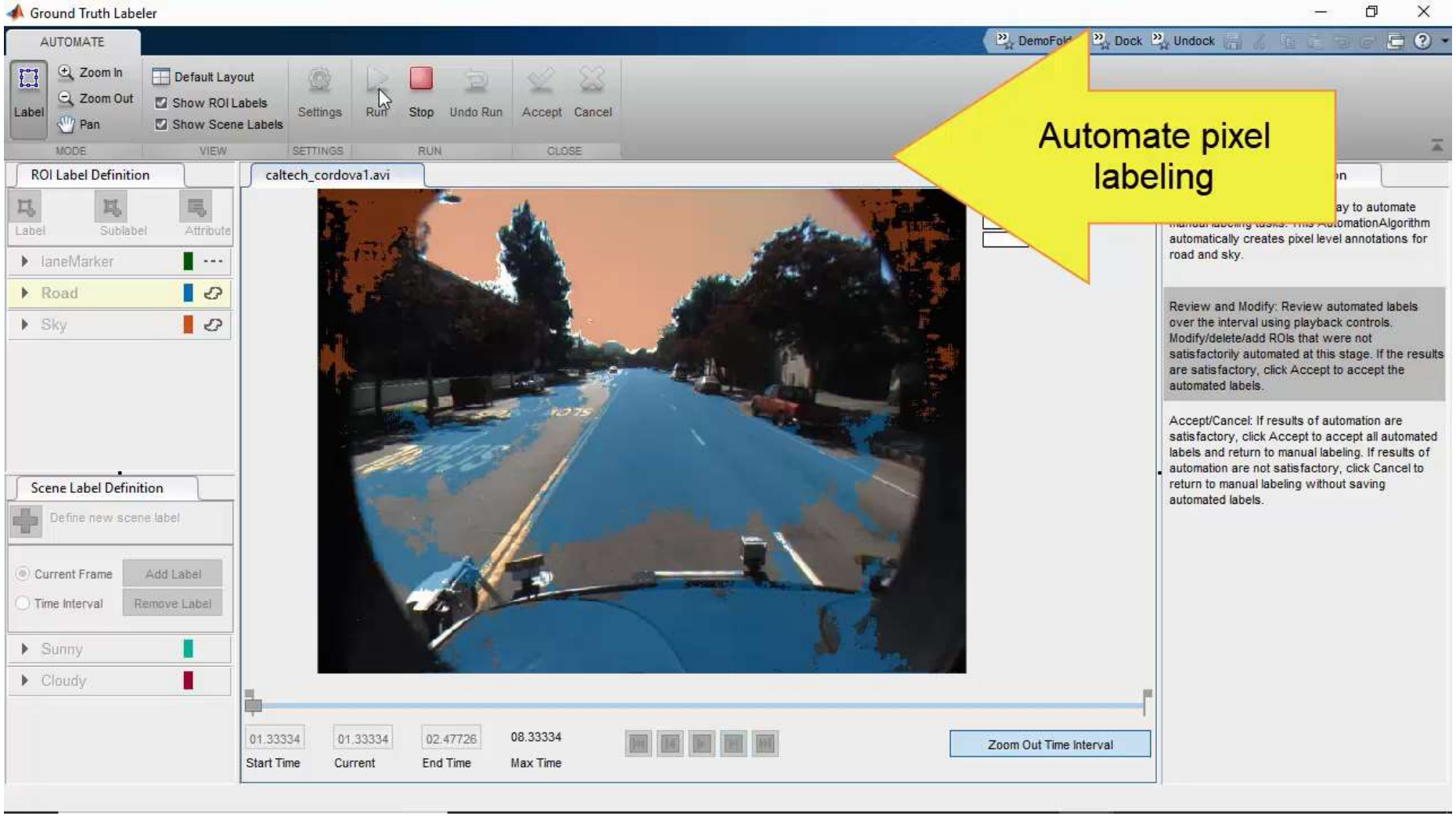


Specify sublabels and attributes in Ground Truth Labeler App

R2018a



Automate labeling pixels with Ground Truth Labeler



Learn how to train a deep learning network using this example

R2017b

Semantic Segmentation Using Deep Learning

- **Train free space detection network** using deep learning

  Computer Vision System Toolbox™

Examples

Search Help

Semantic Segmentation Using Deep Learning

This example shows how to train a semantic segmentation network using deep learning.

A semantic segmentation network classifies every pixel in an image, resulting in an image that is segmented by class. Applications for semantic segmentation include road segmentation for autonomous driving and cancer cell segmentation for medical diagnosis. To learn more, see Semantic Segmentation Basics.

To illustrate the training procedure, this example trains SegNet [1], one type of convolutional neural network (CNN) designed for semantic image segmentation. Other types networks for semantic segmentation include fully convolutional networks (FCN) and U-Net. The training procedure shown here c

This example uses the C
street-level views obtain

Learn more about

Setup

This example creates the
Toolbox™ Model for VG

```
vgg16();
```

This example also uses:
Neural Network Toolbox
vgg16

Open Script

Add-On Explorer

Manage Add-Ons

Search for add-ons

VGG-1
RETRAINED MODE

Neural Network Toolbox Model for VGG-16 Network

version 17.2.0.0 by MathWorks Neural Network Toolbox Team

Pretrained VGG-16 network model for image classification

MathWorks Feature

4 Ratings
125 Downloads
Updated 14 Jun 2017

Install

Overview

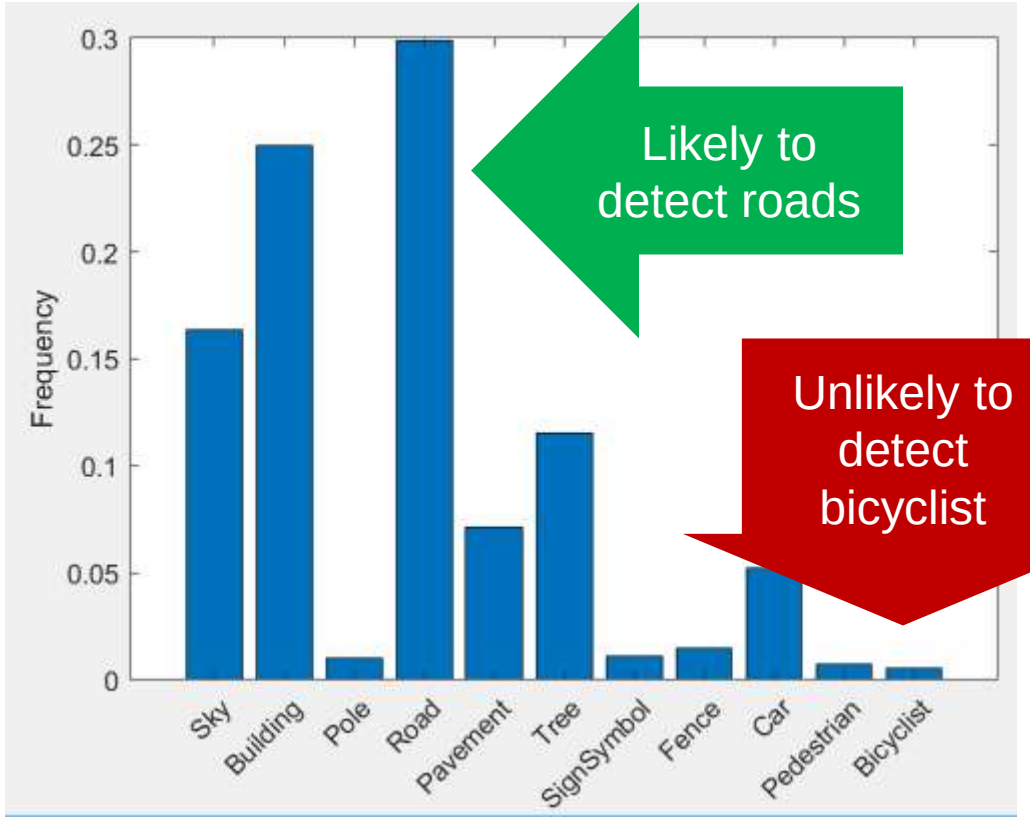
Load and overlay pixel labels

```
% Load pixel labels
classes = ["Sky";  "Building";...
  "Pole";  "Road"; "Pavement"; "Tree";...
  "SignSymbol";    "Fence";    "Car";...
  "Pedestrian";    "Bicyclist"];
pxds = pixelLabelDatastore(...
    labelDir,classes,labelIDs);

% Display labeled image
C = readimage(pxds, 1);
cmap = camvidColorMap;
B = labeloverlay(I,C,'ColorMap',cmap);
imshow(B)
```

`pixelLabelDatastore` manages large collections of pixel labels

Visualize distribution of labeled pixels

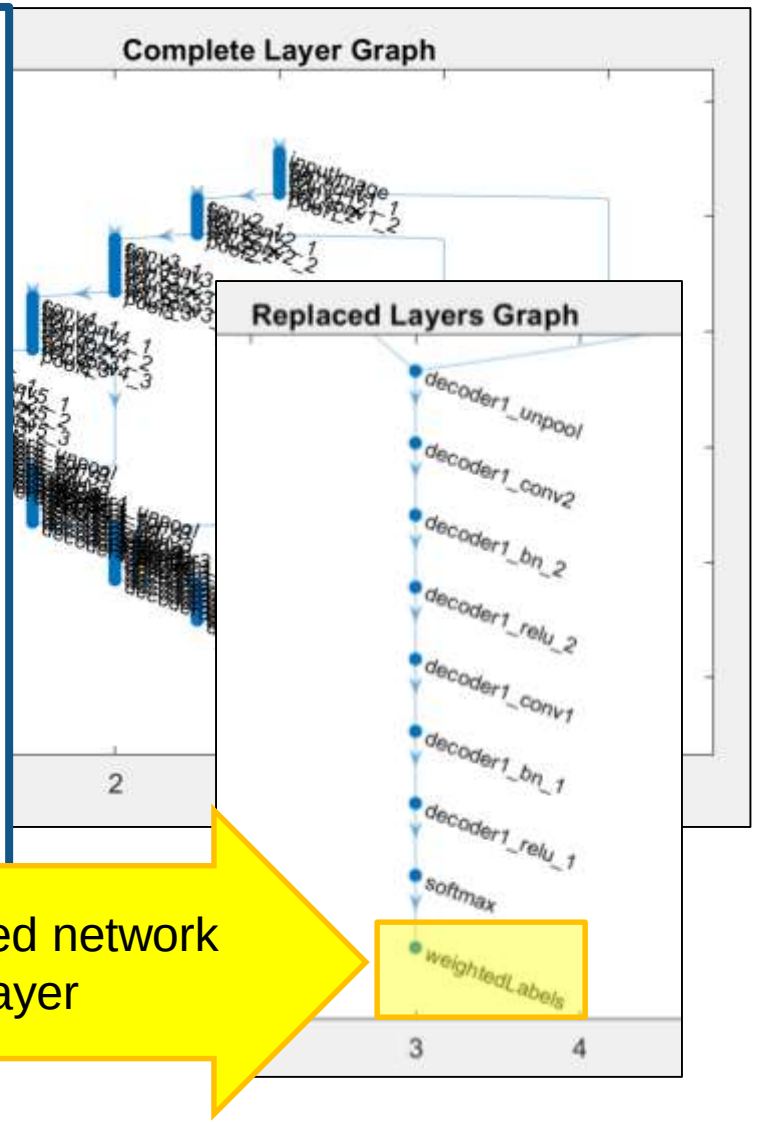
```
% Visualize label count by class
tbl = countEachLabel(pxds)

frequency = tbl.PixelCount / ...
            sum(tbl.PixelCount);

bar(1:numel(classes),frequency)
xticks(1:numel(classes))
xticklabels(tbl.Name)
xtickangle(45)
ylabel('Frequency')
```



Labeled pixels in this set are imbalanced

Add weighted layer to compensate for imbalanced data set

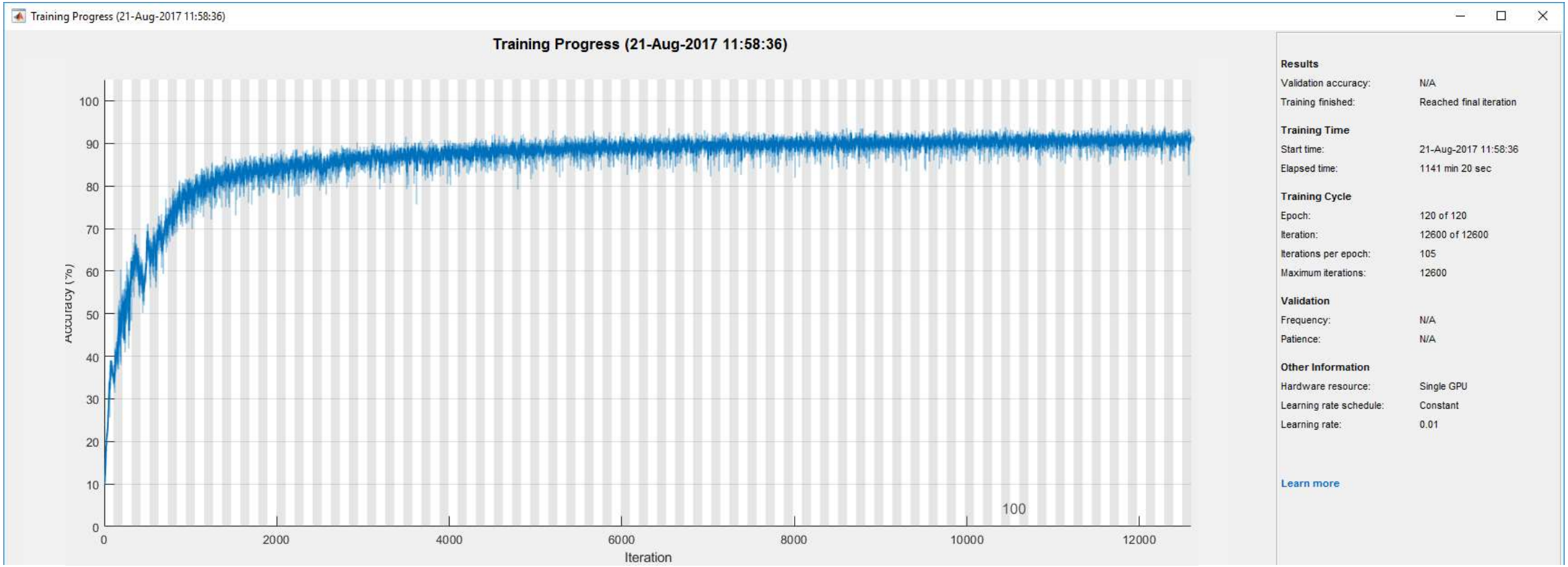
```
% Create weighted layer
pxLayer = pixelClassificationLayer(...
    'Name','weightedLabels', 'ClassNames',tbl.Name,...
    'ClassWeights',classWeights)

% Replace layer
lgraph = removeLayers(lgraph, 'pixelLabels');
lgraph = addLayers(lgraph, pxLayer);
lgraph = connectLayers(lgraph,...
    'softmax', 'weightedLabels');

% Display network structure
plot(lgraph); ylim([0 9.5])
title('Replaced Layers Graph')
```



Replaced network layer

Train network and view progress

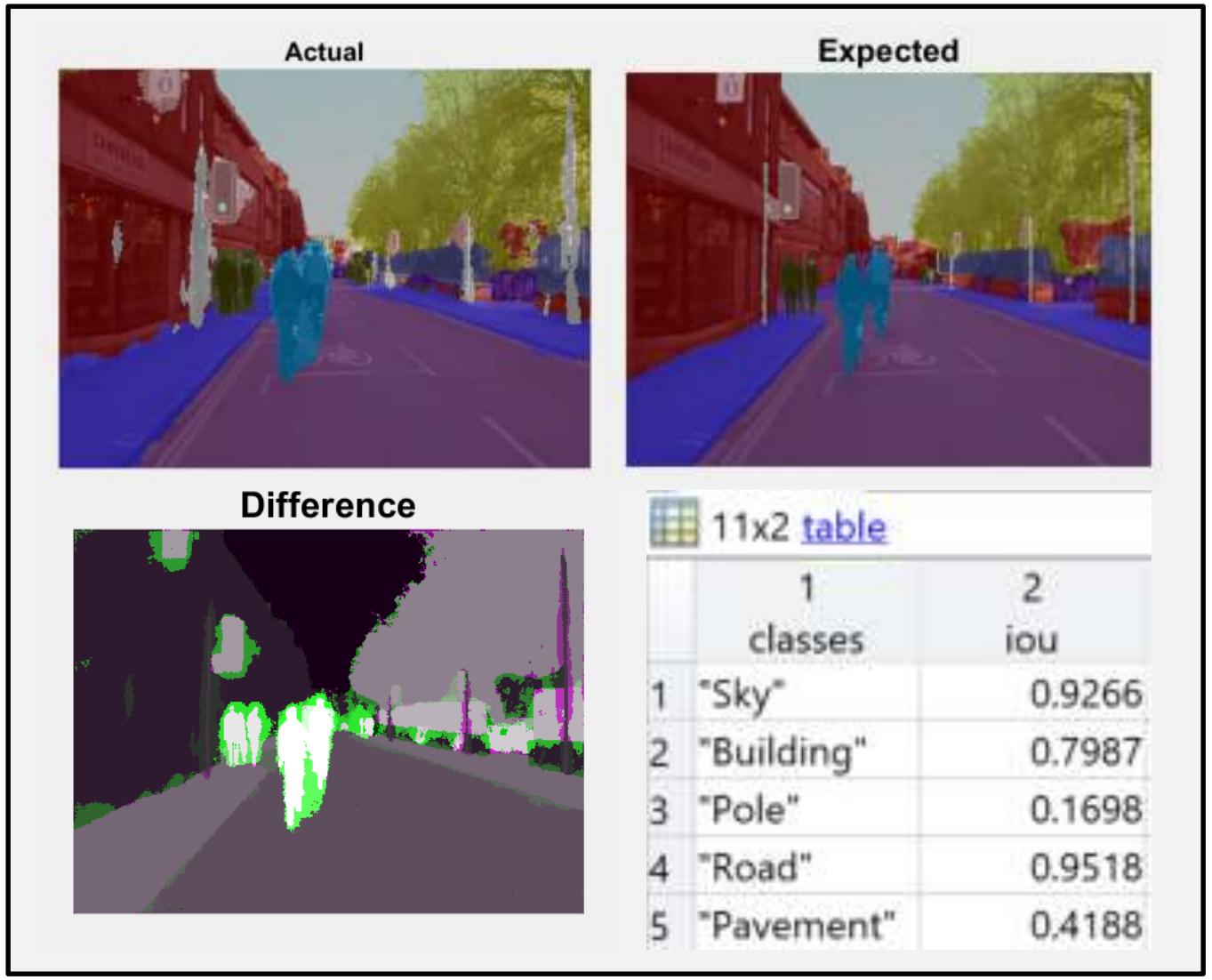
```
[net, info] = trainNetwork(datasource, lgraph, options);
```



Assess similarity using intersection-over-union (IoU) metric

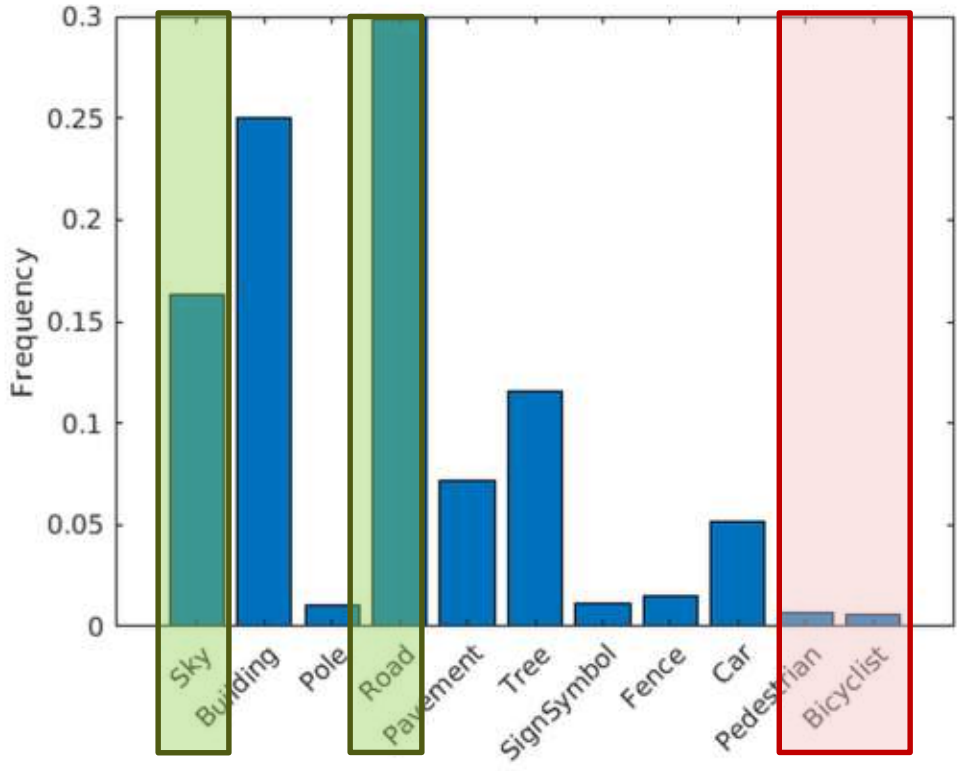
```
iou = jaccard(actual,...
                  expected);
table(classes,iou)
```

```
ans =
  11×2 table
      classes          iou
    ___________     ________

    "Sky"            0.92659
    "Building"        0.7987
    "Pole"           0.16978
    "Road"           0.95177
    "Pavement"       0.41877
    "Tree"           0.43401
    "SignSymbol"     0.32509
    "Fence"            0.492
    "Car"          0.068756
    "Pedestrian"          0
    "Bicyclist"           0
```



Distribution of labels in data affects intersection-over-union (IoU)

Distribution of labels in original data set



Evaluation metrics of network

	Accuracy	IoU	MeanBFScore
Sky	0.93544	0.89279	0.88239
Building	0.79978	0.75543	0.59861
Pole	0.73166	0.18361	0.51426
Road	0.93644	0.90663	0.7086
Pavement	0.90624	0.72932	0.70585
Tree	0.86587	0.73694	0.67097
SignSymbol	0.76118	0.35339	0.44175
Fence	0.83258	0.49648	0.50265
Car	0.90961	0.75263	0.64837
Pedestrian	0.83751	0.35409	0.46796
Bicyclist	0.84156	0.5472	0.46933

Underrepresented classes such as Pedestrian and Bicyclist are not segmented as well as classes such as Sky and Road

Detection drivable space using semantic segmentation

Learn more about developing deep learning perception algorithms with these examples

R2018a

Automate Ground Truth Labeling for Semantic Segmentation

- **Add semantic segmentation** automation algorithm to Ground Truth Labeler App

Automated Driving System Toolbox™

R2017b

Semantic Segmentation Using Deep Learning

- **Train free space detection network** using deep learning

Computer Vision System Toolbox™

R2018a

Code Generation for Semantic Segmentation Network

- **Generate CUDA® code** to execute directed acyclic graph network on an NVIDIA GPU

GPU Coder™

How can you use MATLAB and Simulink to develop perception algorithms?

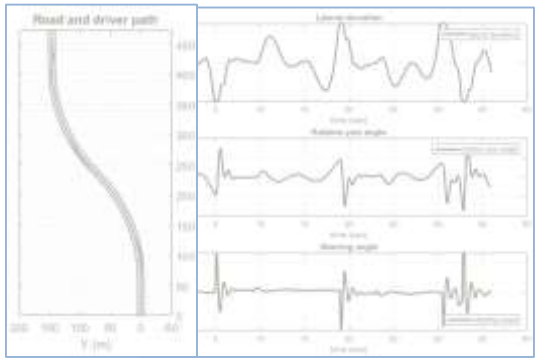
Deep learning

Sensor fusion with live data

Perception

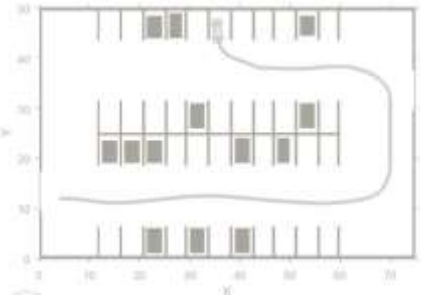
Control

Sensor models & model predictive control



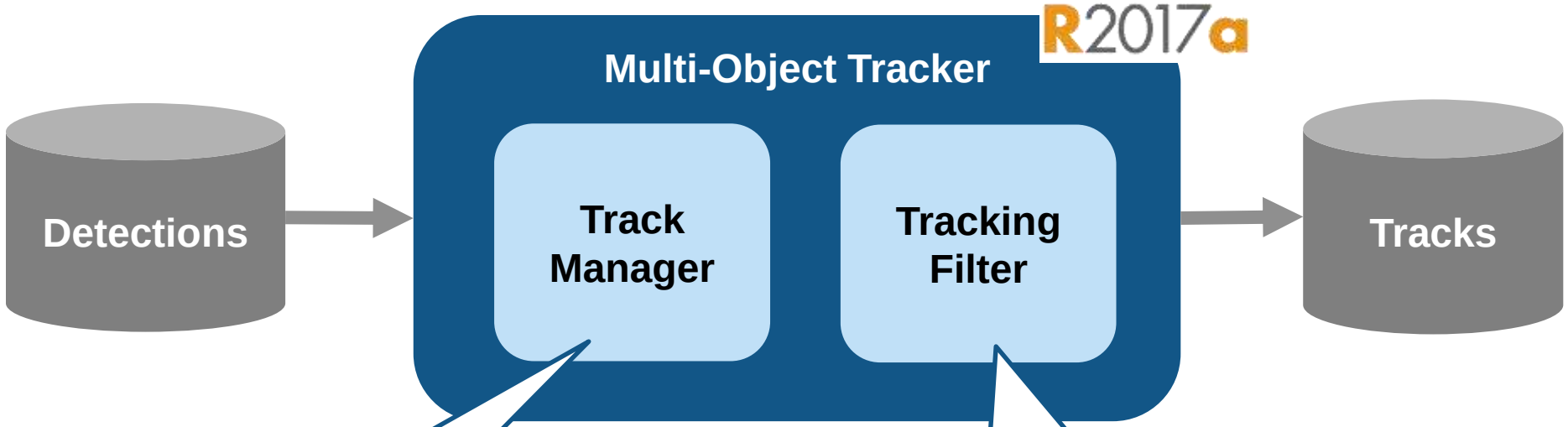
Planning

Path planning



Automated Driving System Toolbox introduced:

Multi-object tracker to develop sensor fusion algorithms

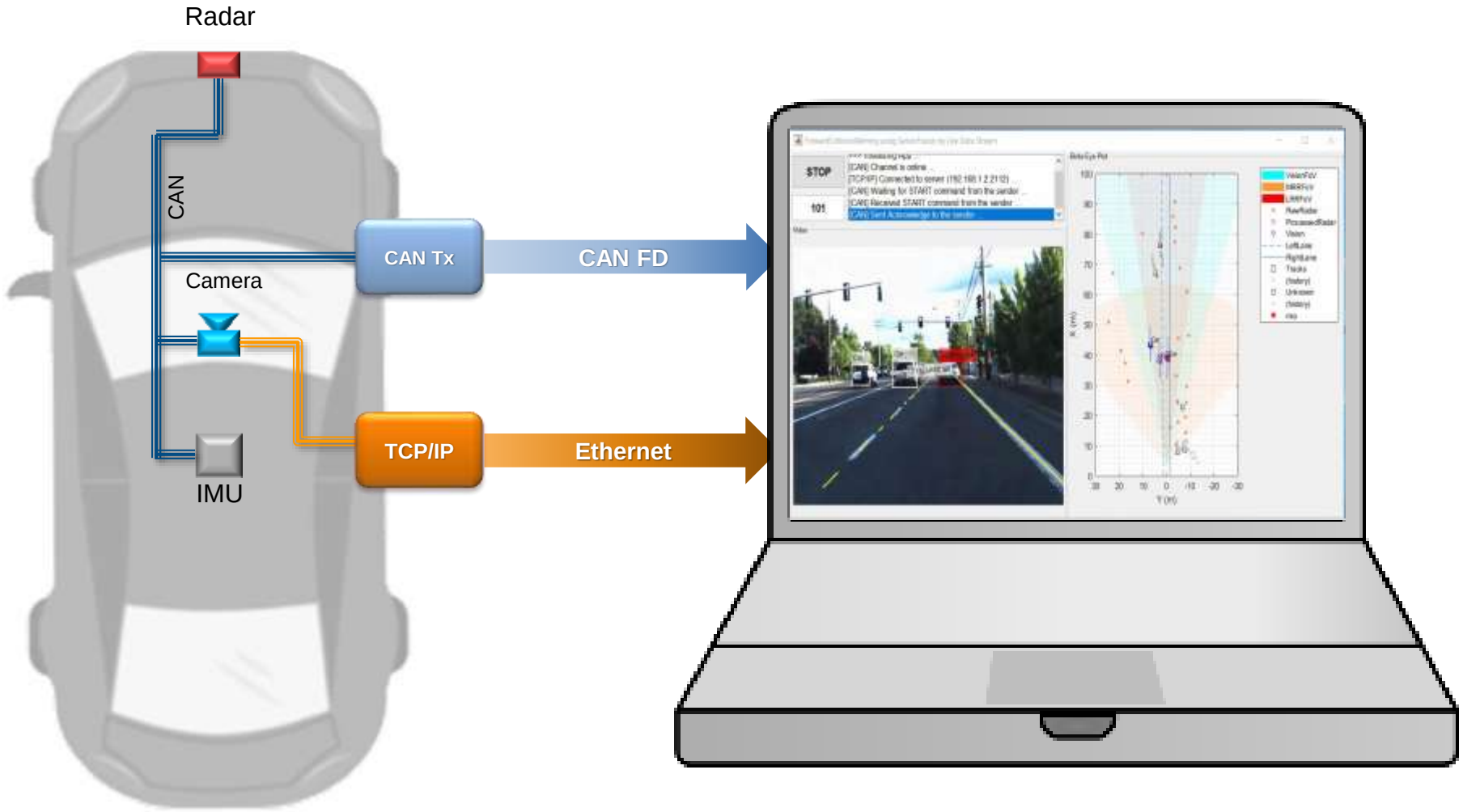


- Assigns detections to tracks
- Creates new tracks
- Updates existing tracks
- Removes old tracks

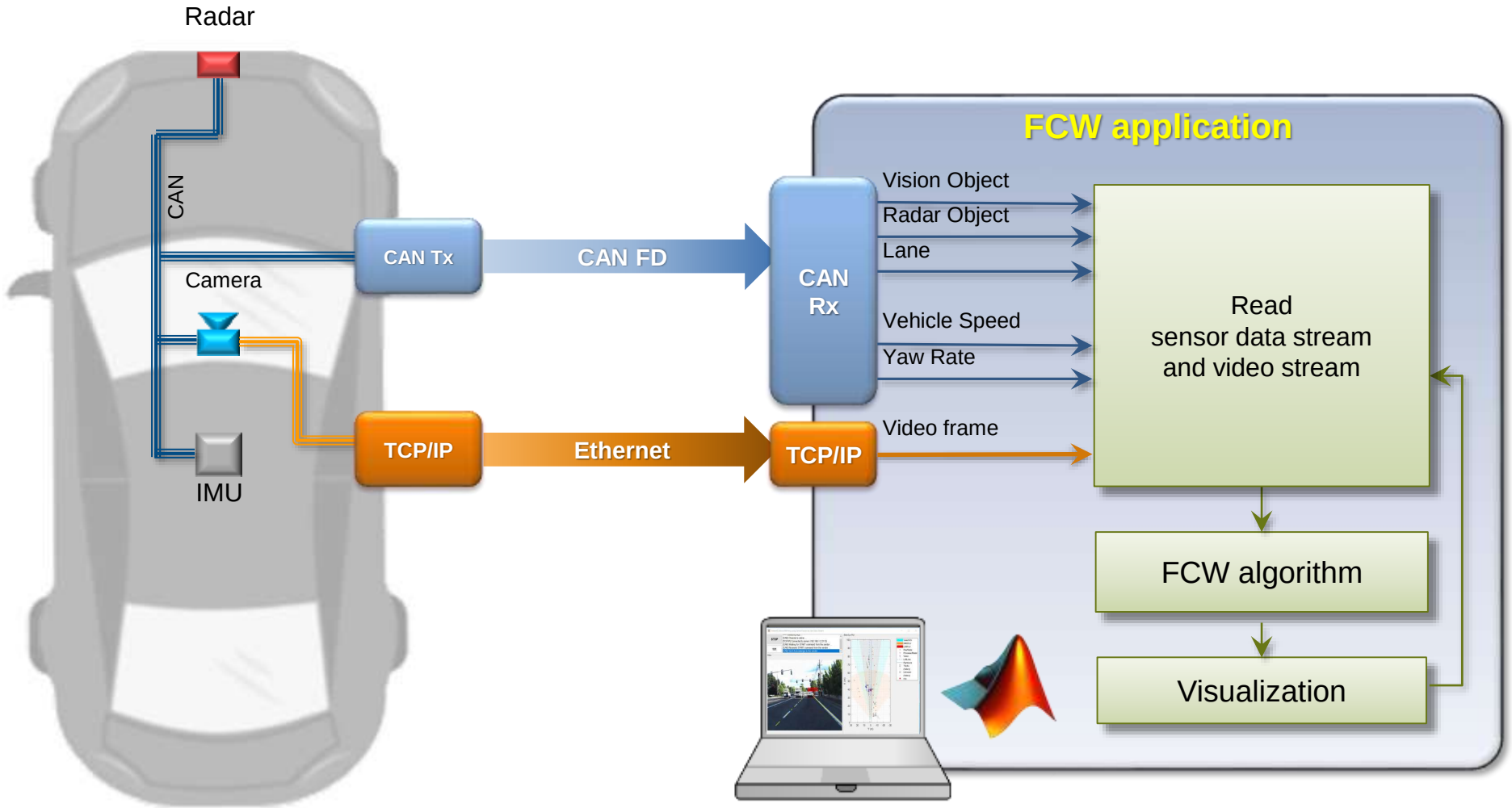
- Predicts and updates state of track
- Supports linear, extended, and unscented Kalman filters

Introduction to Automated Driving System Toolbox

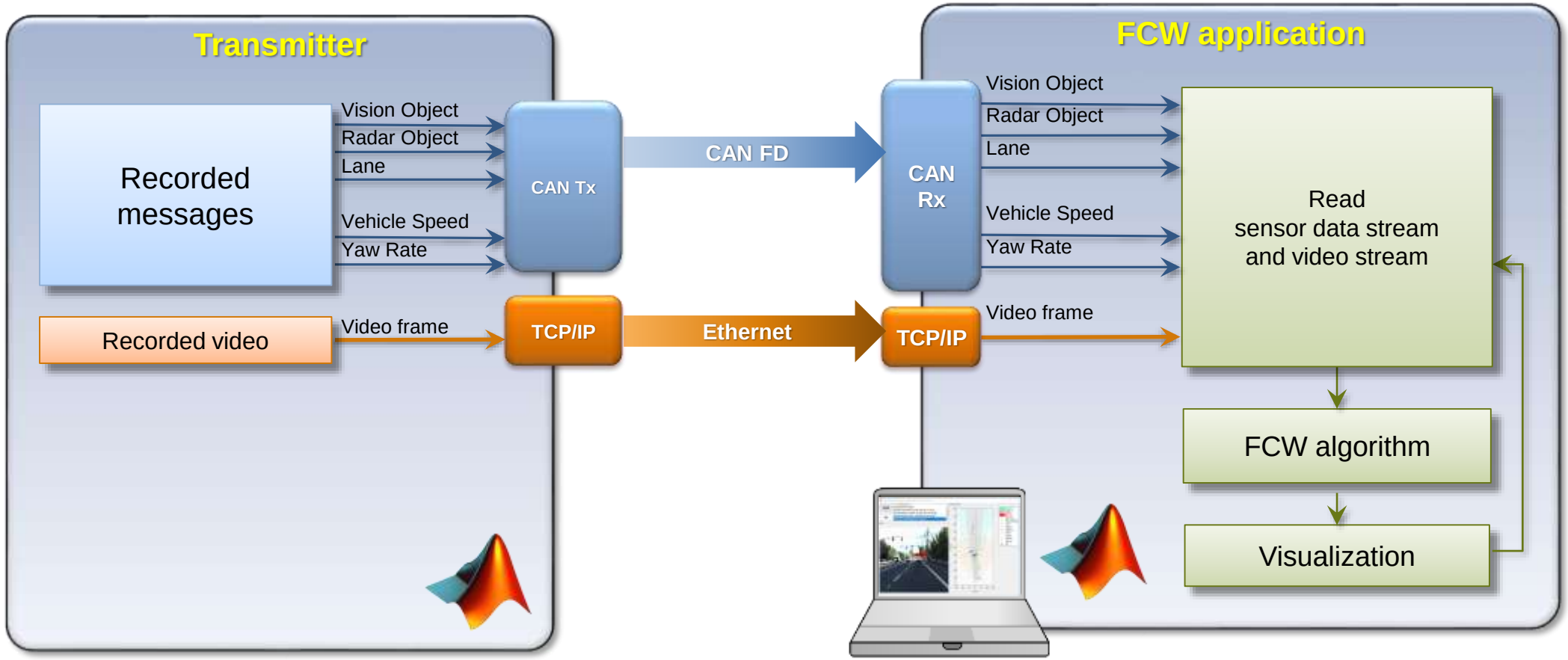
How can I test my sensor fusion algorithm with live data?



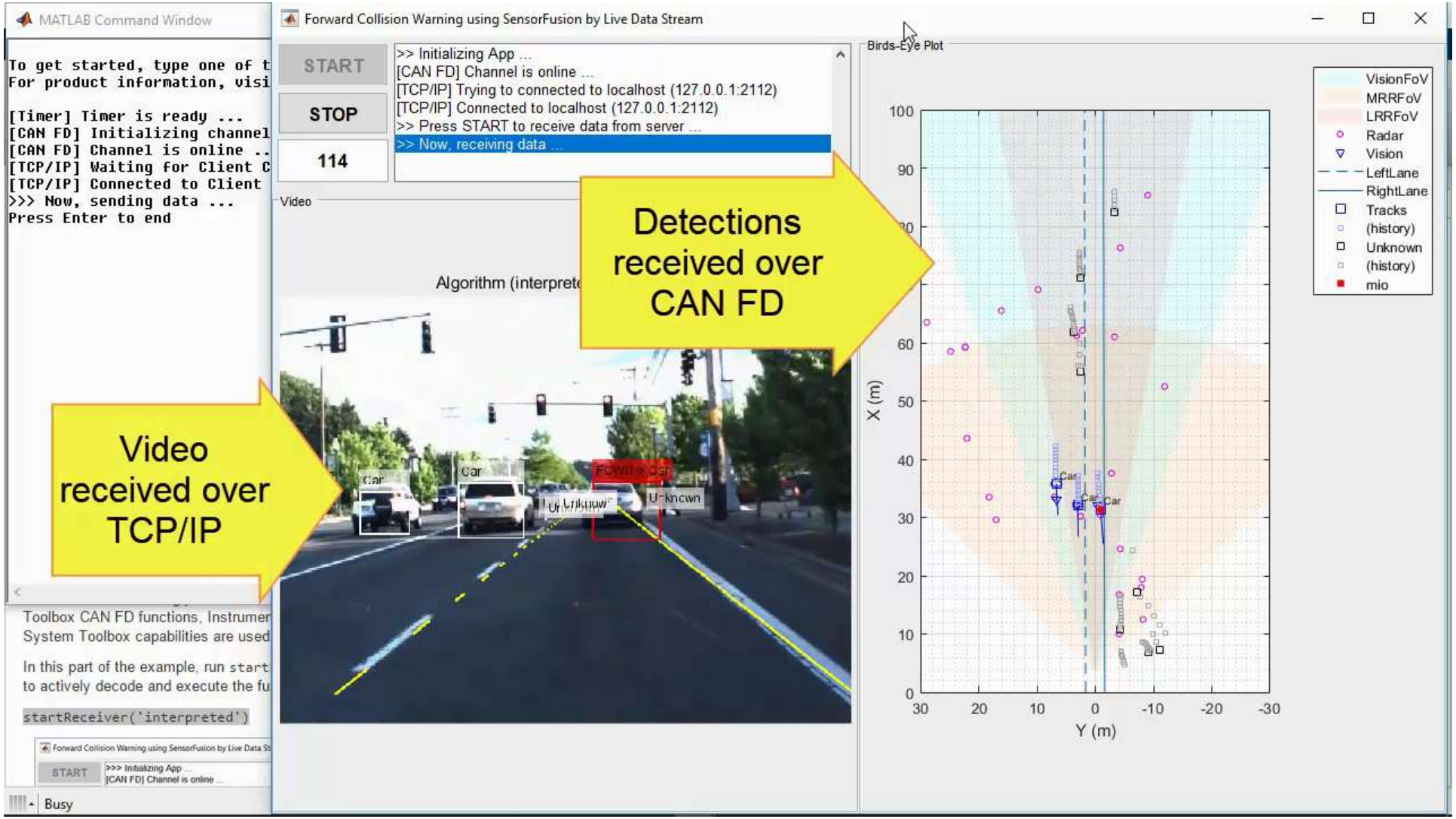
Test forward collision warning algorithm with live data from vehicle



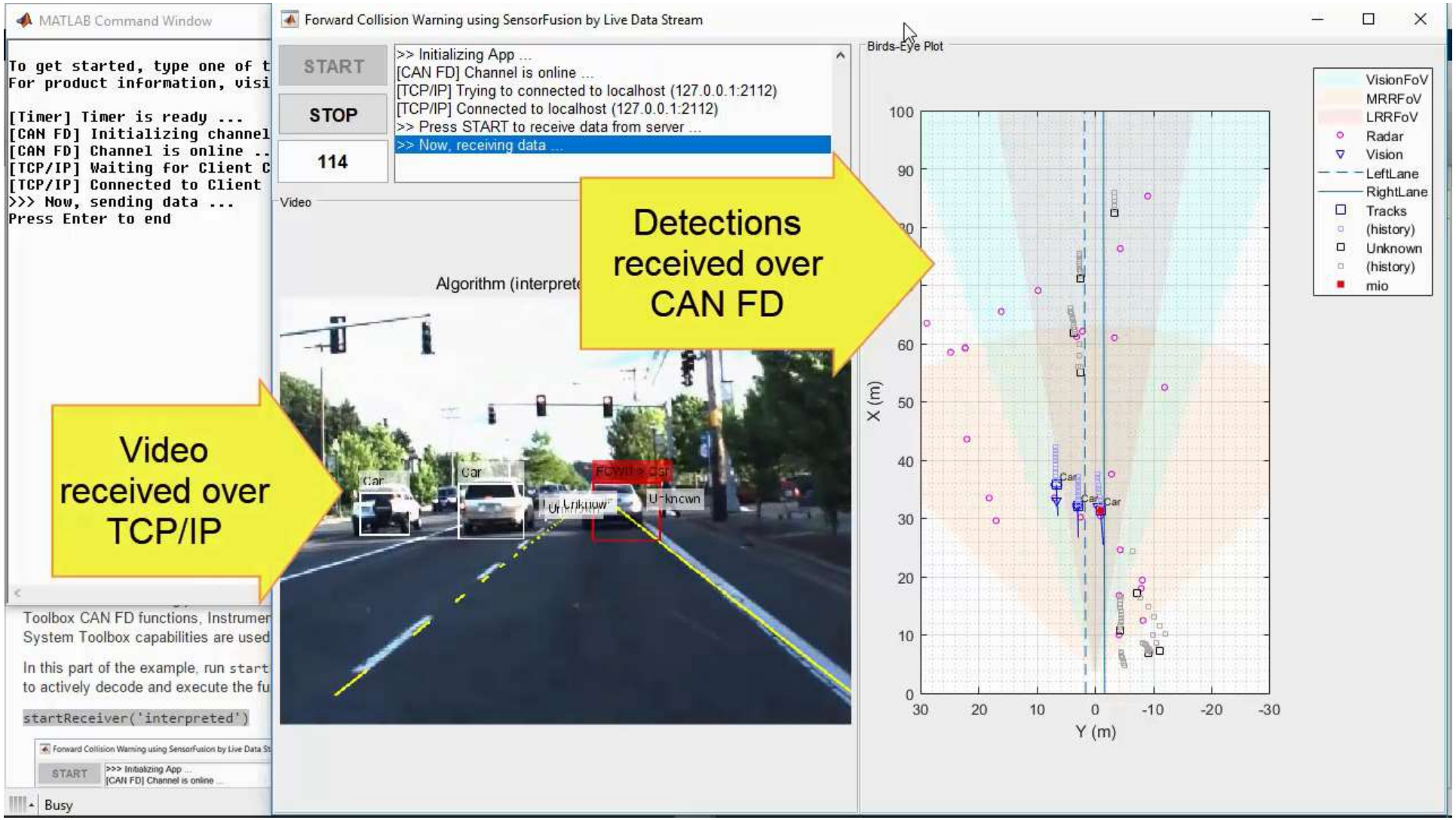
Test forward collision warning algorithm with live data from “surrogate” vehicle



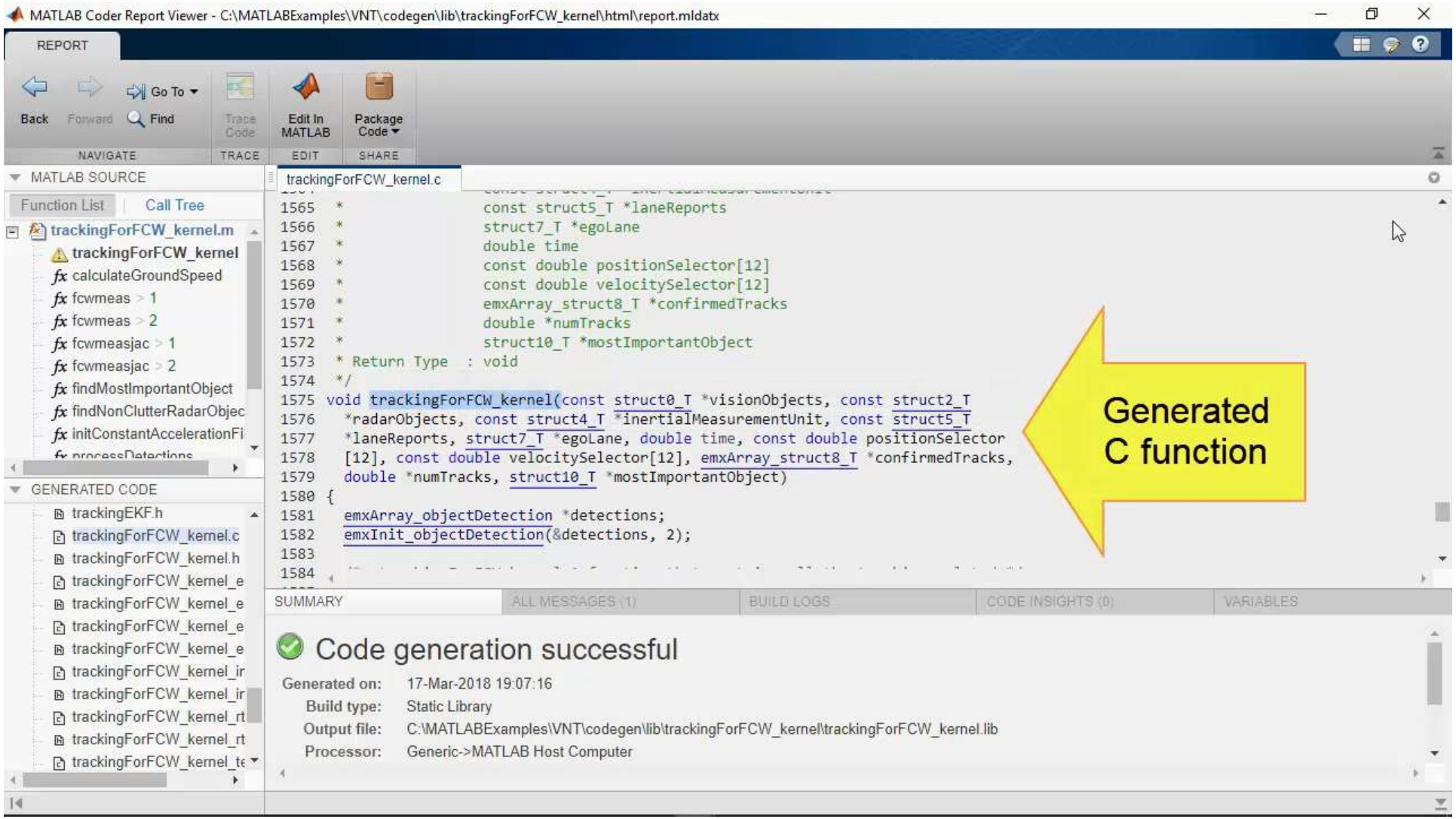
Send live CAN FD and TCP/IP data



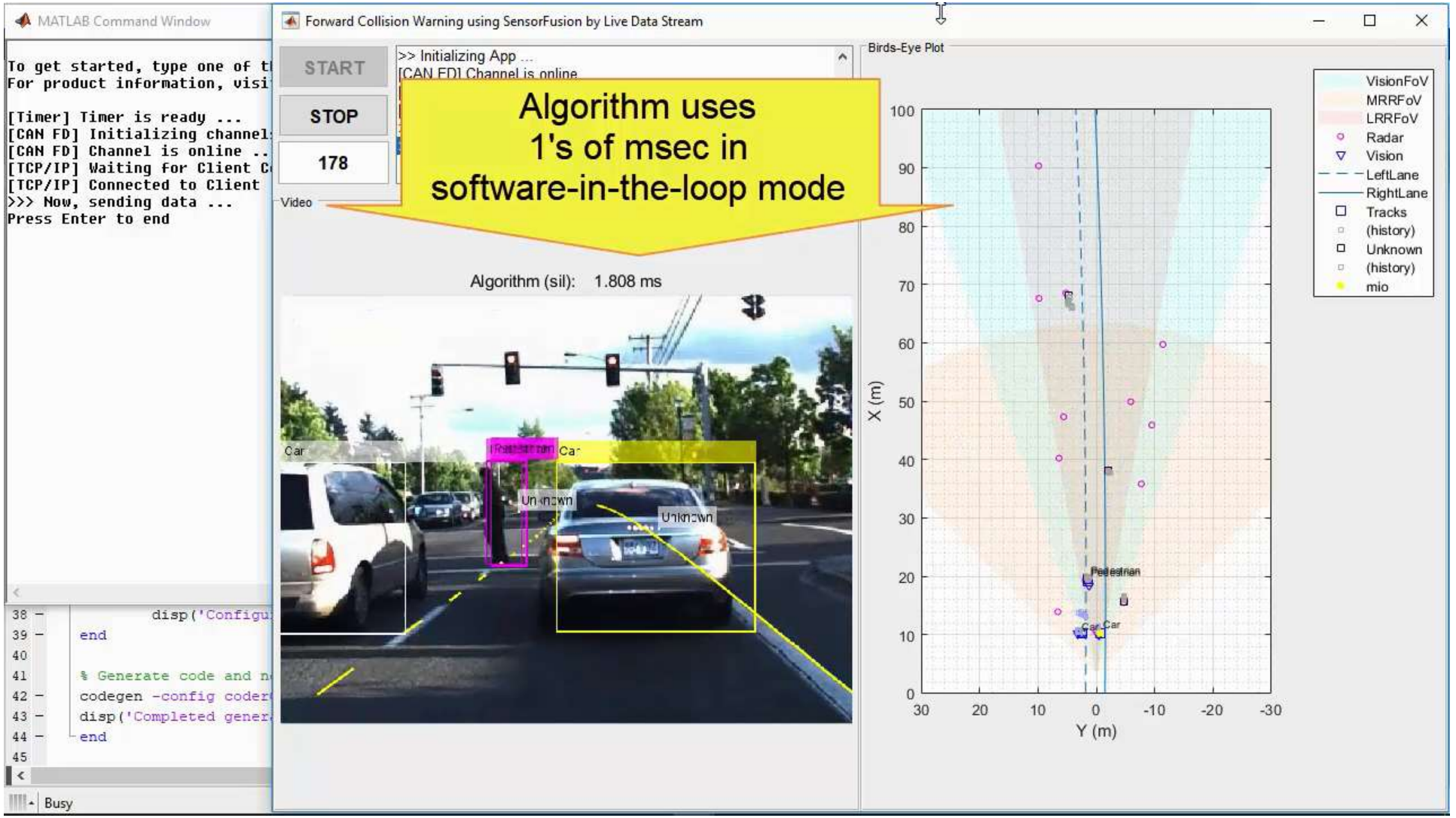
Receive live CAN FD and TCP/IP data



Generate C/C++ code for algorithm



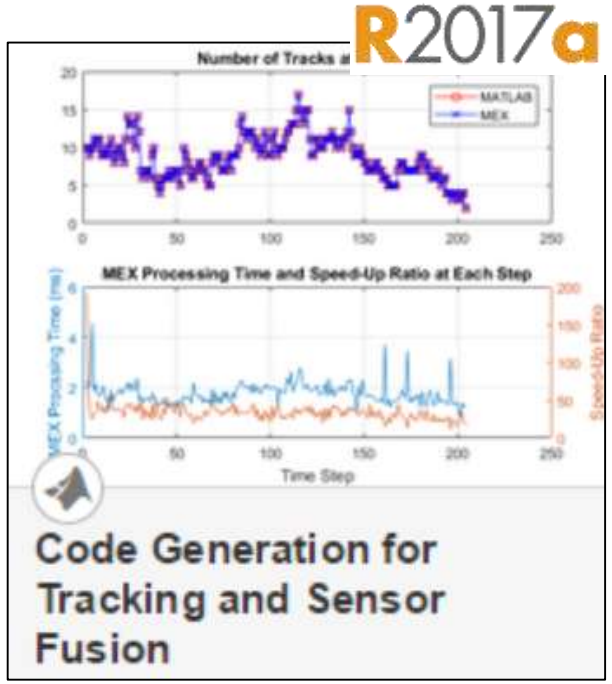
Stream live CAN FD and TCP/IP data into compiled algorithm code



Learn more about developing sensor fusion algorithms

- **Design** algorithm with multi-object tracker and recorded vehicle data

  Automated Driving System Toolbox™



- **Generate C/C++** code from algorithm which includes a multi-object tracker

  MATLAB Coder™

- **Stream CAN FD** data to prototype algorithm on your laptop

  Vehicle Network Toolbox™

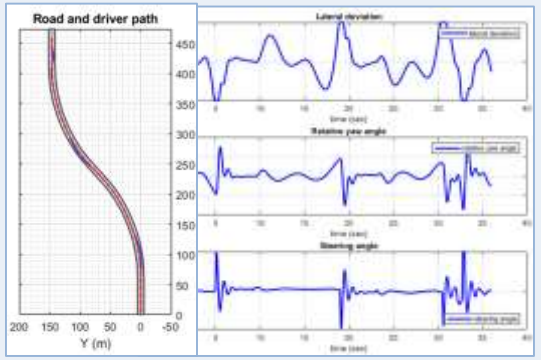
How can you use MATLAB and Simulink to develop control algorithms?

Deep learning

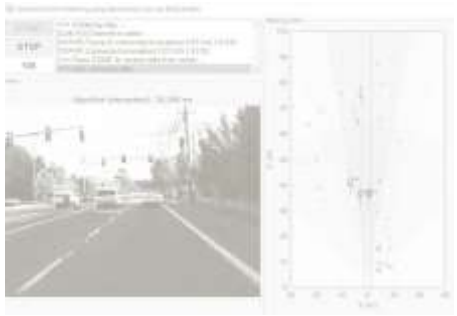
Perception

Control

Sensor models & model predictive control

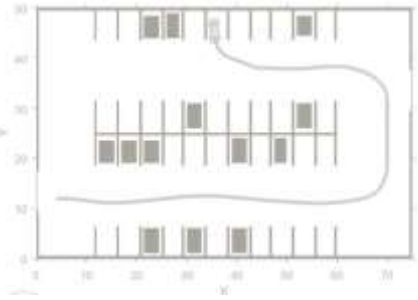


Sensor fusion with live data



Planning

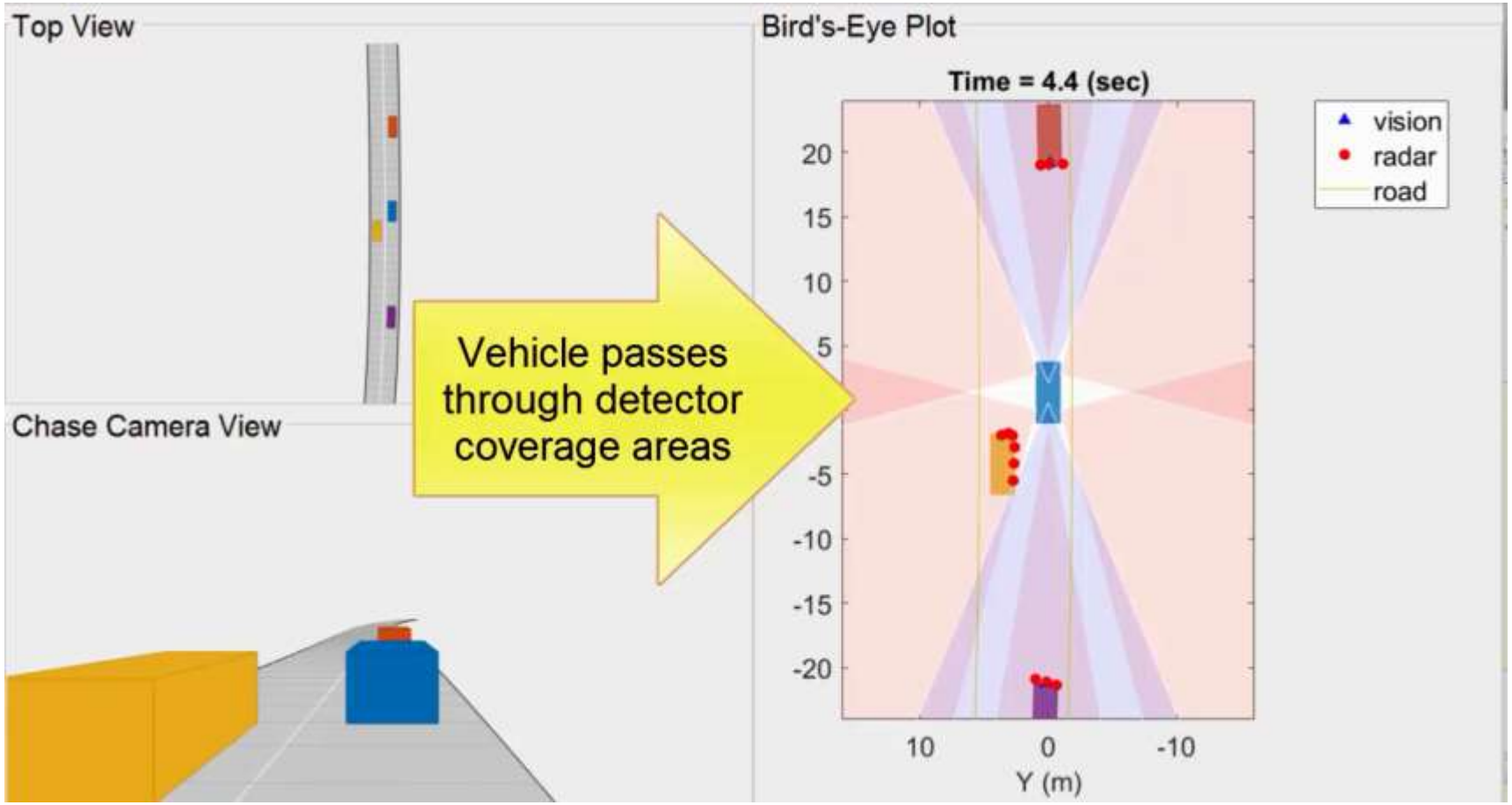
Path planning



Automated Driving System Toolbox introduced:

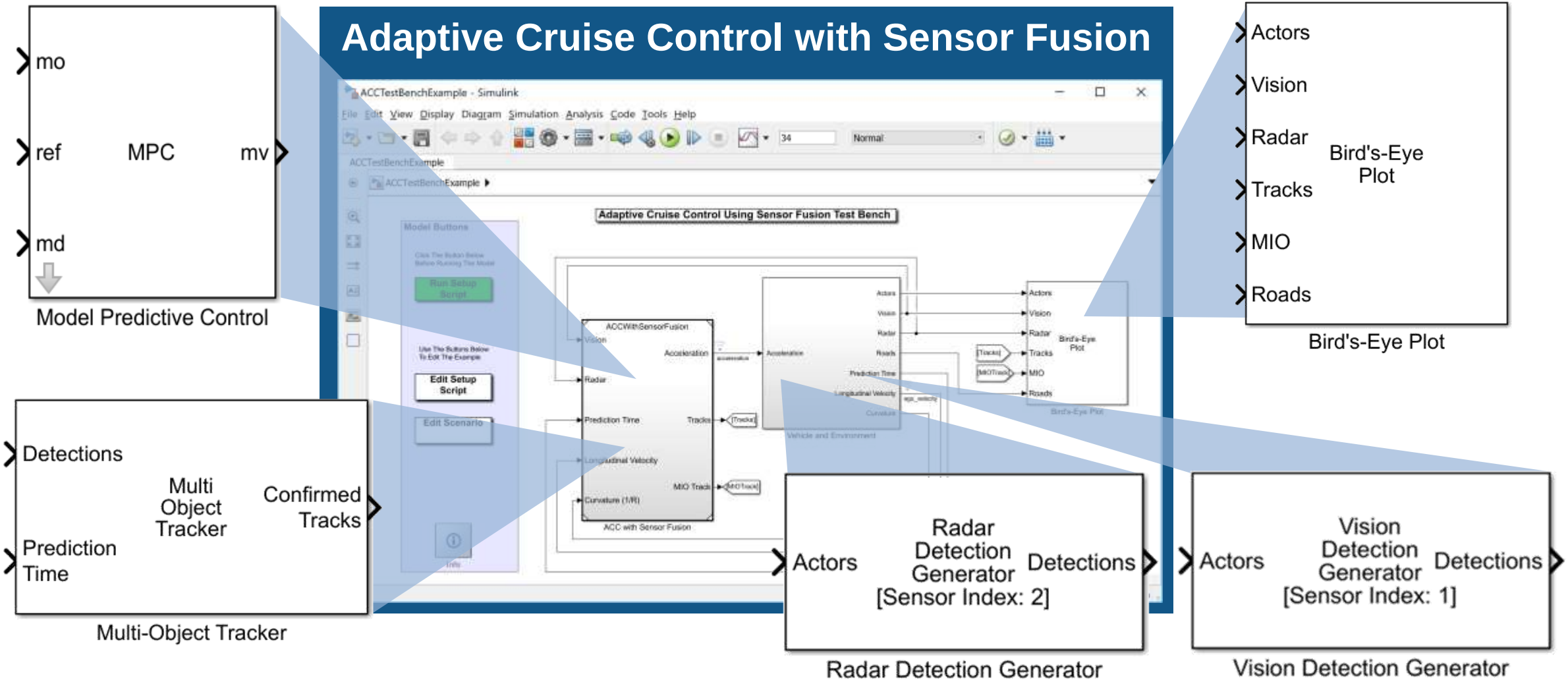
Synthesizing scenarios to test sensor fusion algorithms

R2017a

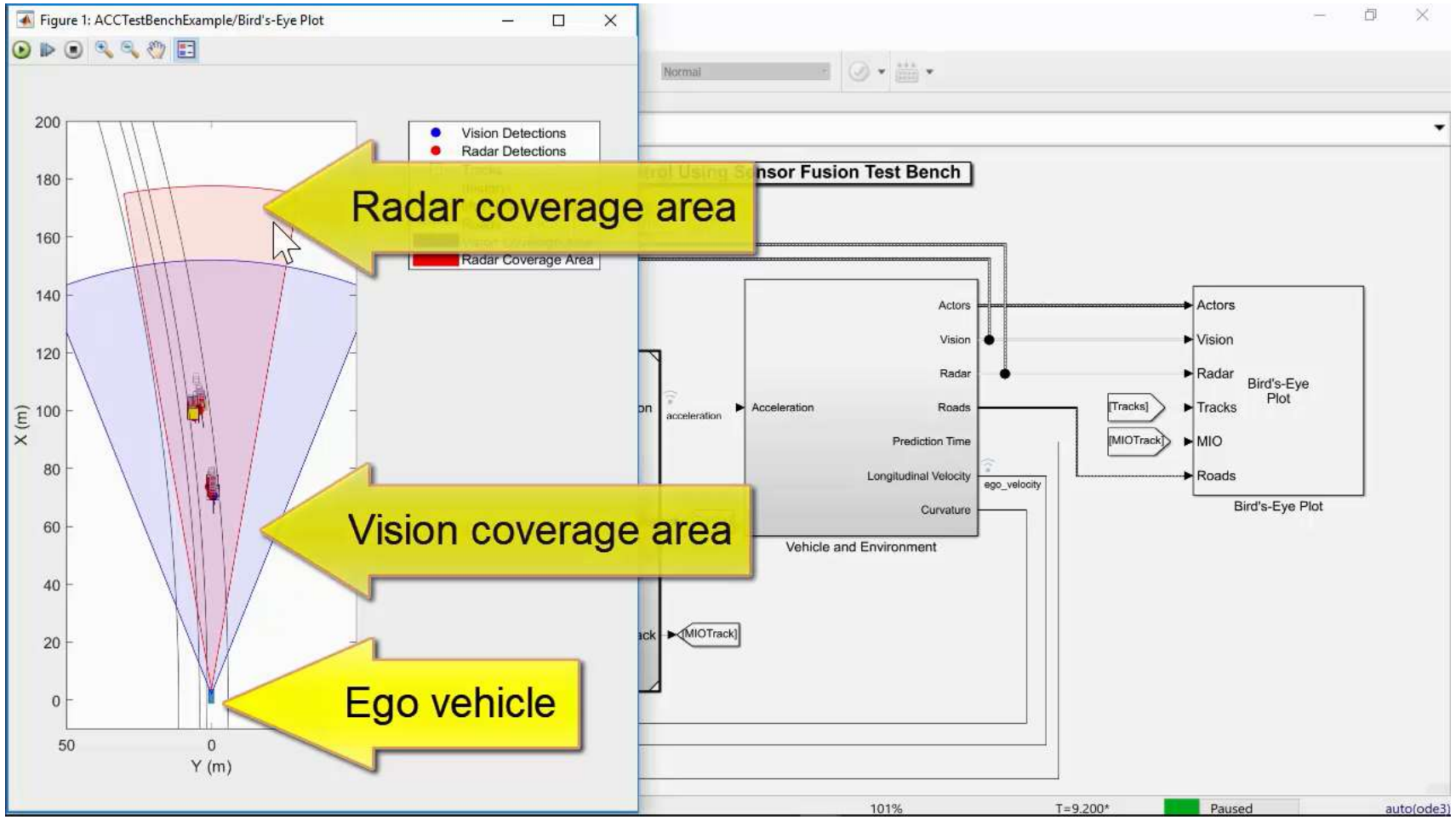


Simulate closed loop system with radar/vision detections, sensor fusion, and model-predictive control

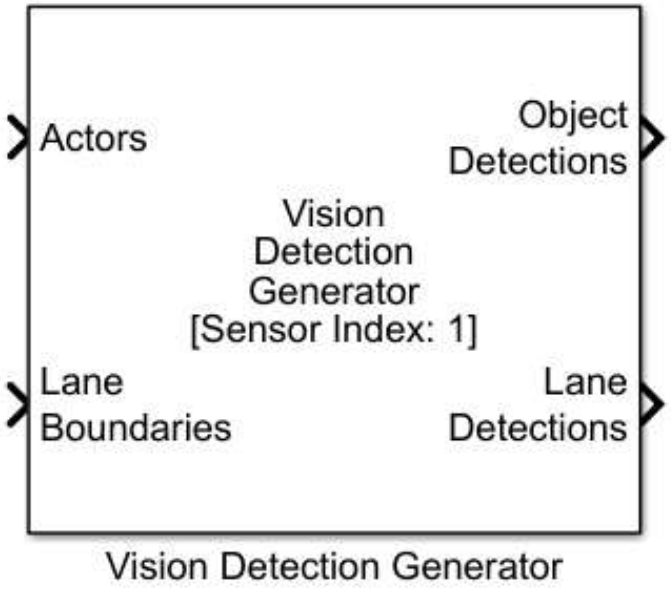
R2017b



Synthesize detections to test sensor fusion and model-predictive controller



Synthesize lane detection with Vision Detection Generator



Vision Detection Generator

R2018a

Block Parameters: Vision Detection Generator

Vision Detection Generator

Sensor simulation block used to generate vision detections from simulated actor poses. Detections are generated at intervals of the sensor's update interval. A statistical model generates measurement noise, true detections, and false positives. The random numbers used by the statistical model are controlled by the random number generator settings on the Measurements tab.

Source code

Parameters | Measurements | Actor Profiles | Camera Intrinsics

Sensor Identification

Unique identifier of sensor: 1

Types of detections generated by sensor: Lanes and objects

Required interval between sensor updates (s):

Required interval between lane detection updates (s):

- Objects only
- Lanes only
- Lanes with occlusion
- Lanes and objects

Sensor Extrinsics

Sensor's (x,y) position (m): [1.9, 0]

Sensor's height (m): 1.1

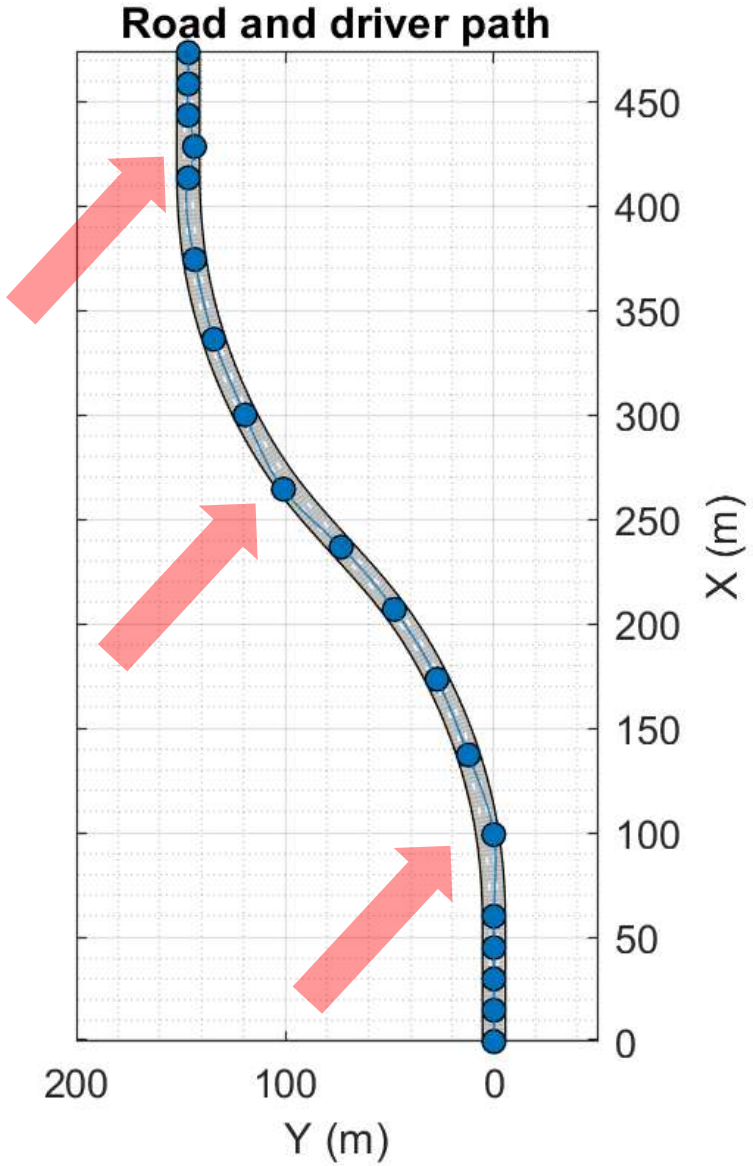
Yaw angle of sensor mounted on ego vehicle (deg): 0

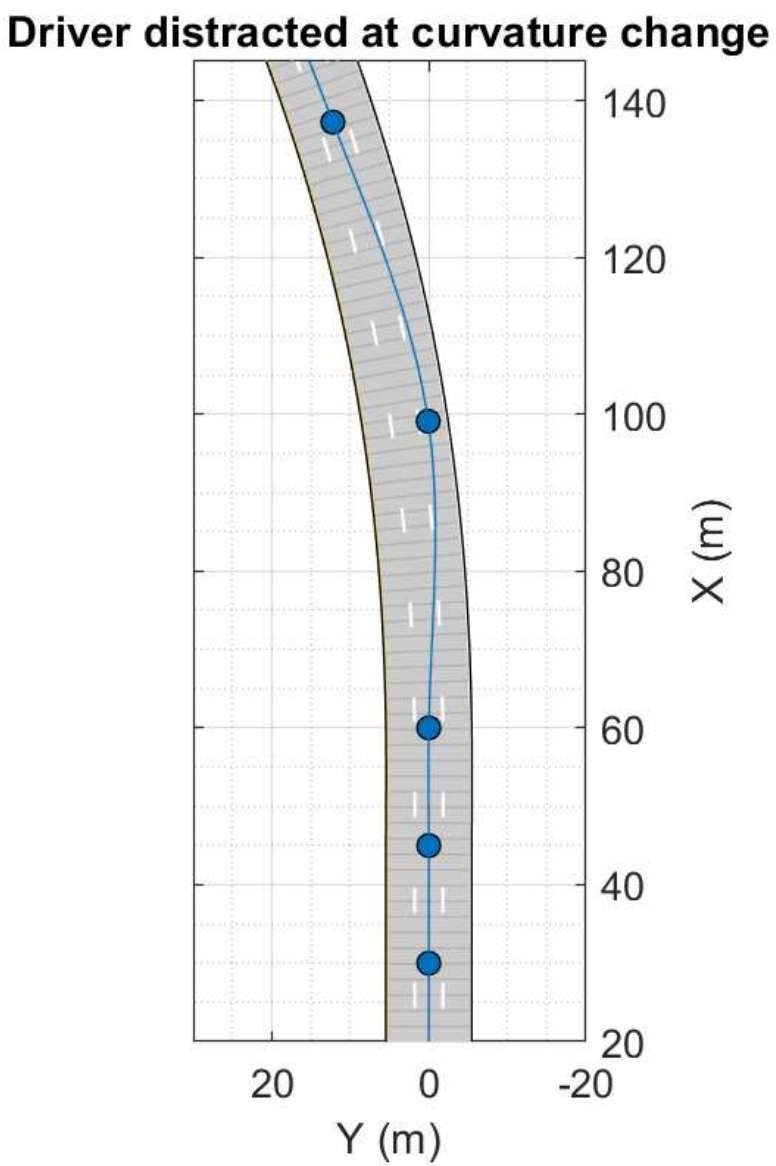
Pitch angle of sensor mounted on ego vehicle (deg): 1

Roll angle of sensor mounted on ego vehicle (deg): 0

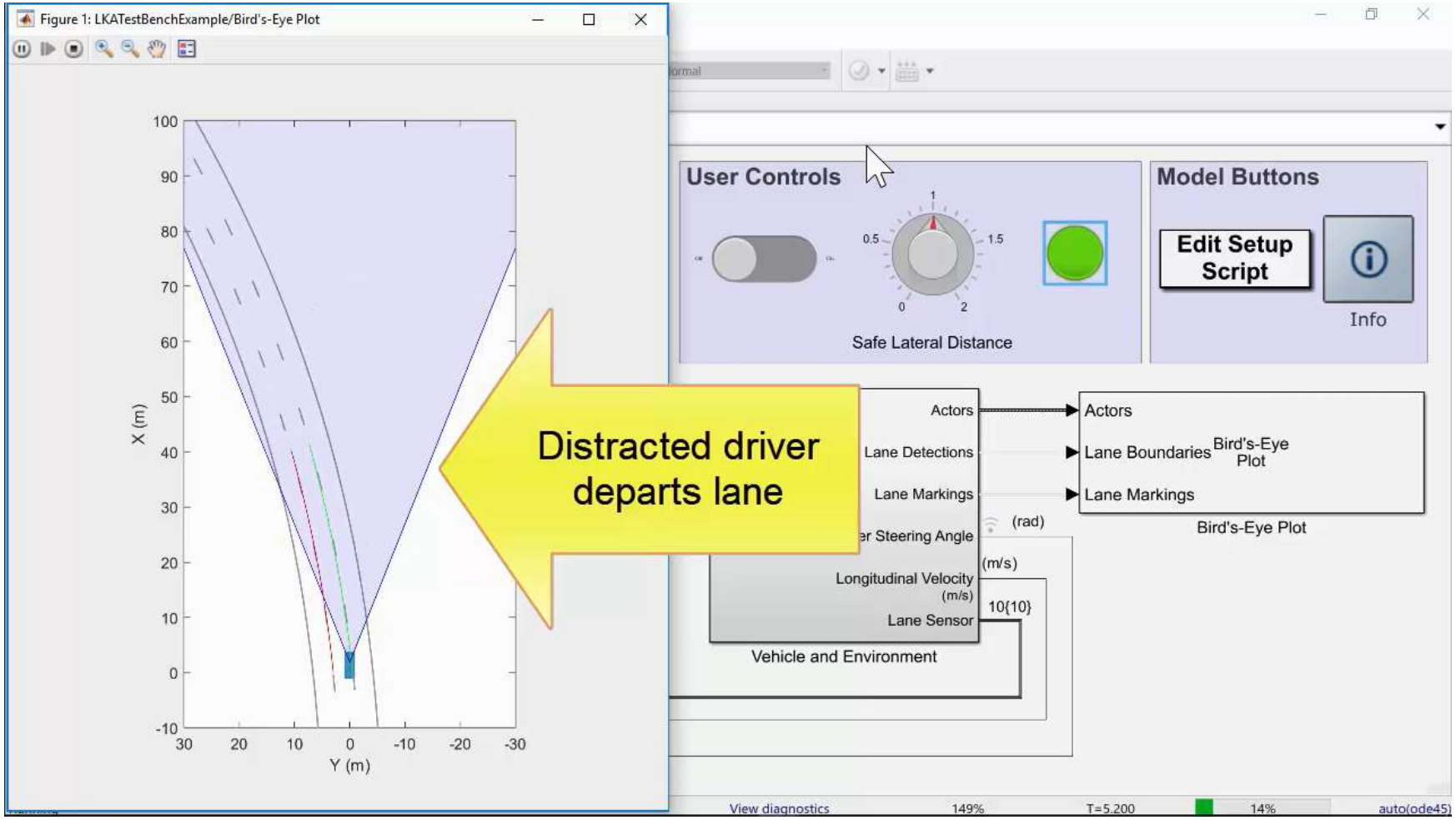
Create highway double curve with drivingScenario

- Driver waypoints simulate distraction at curvature changes

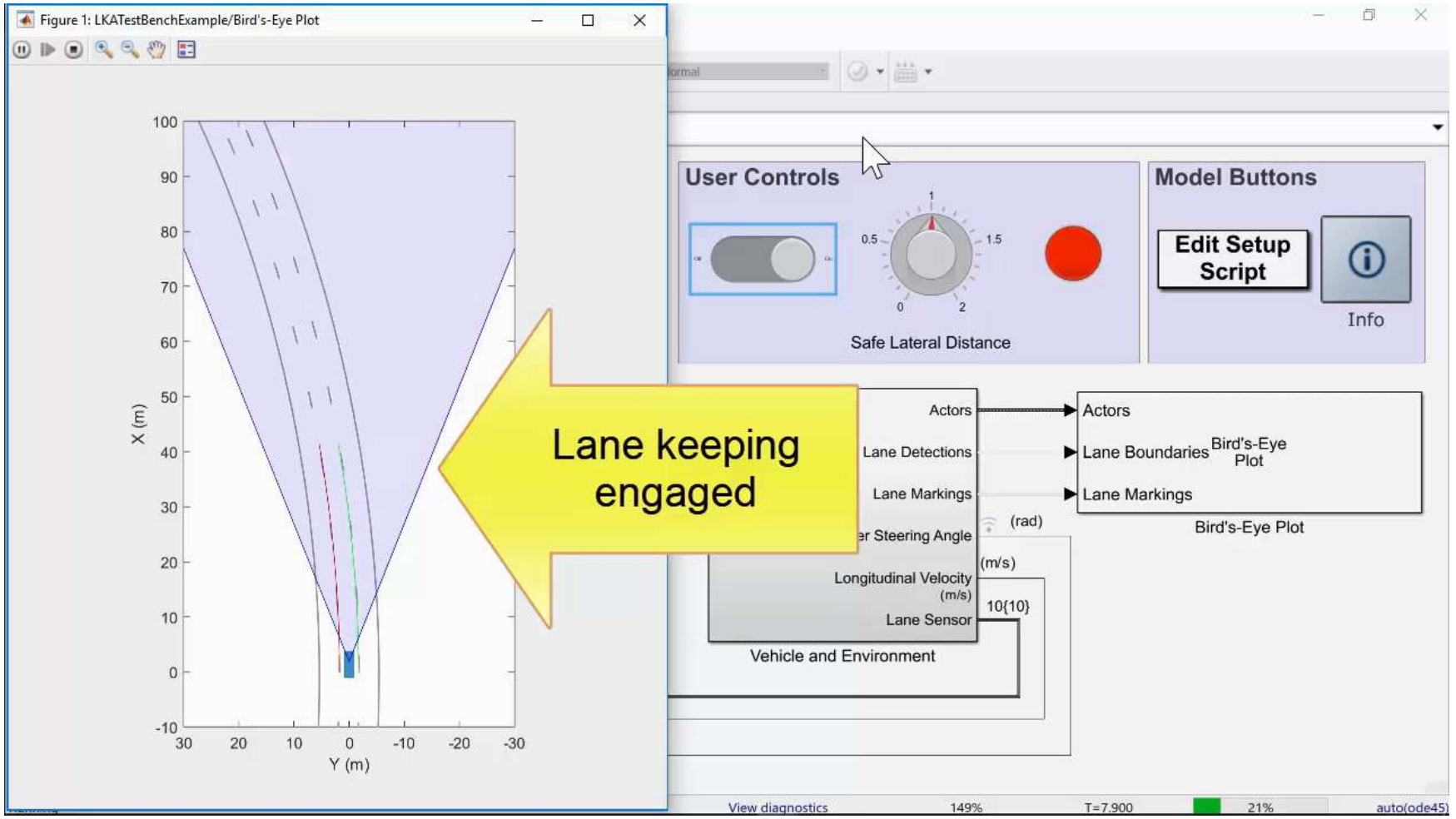




Simulate distracted driver

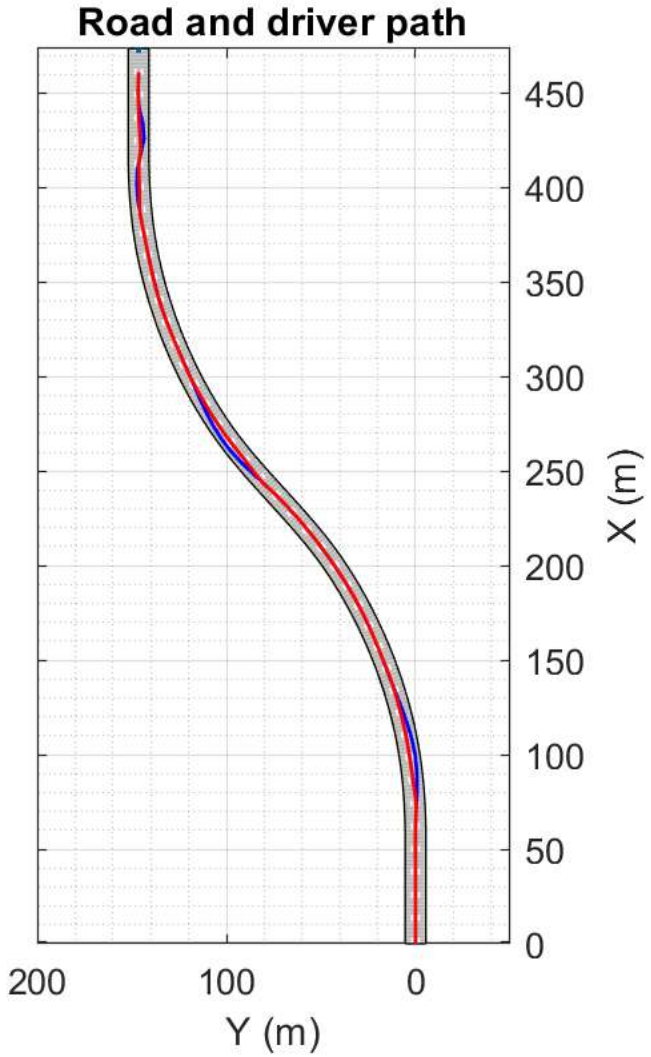


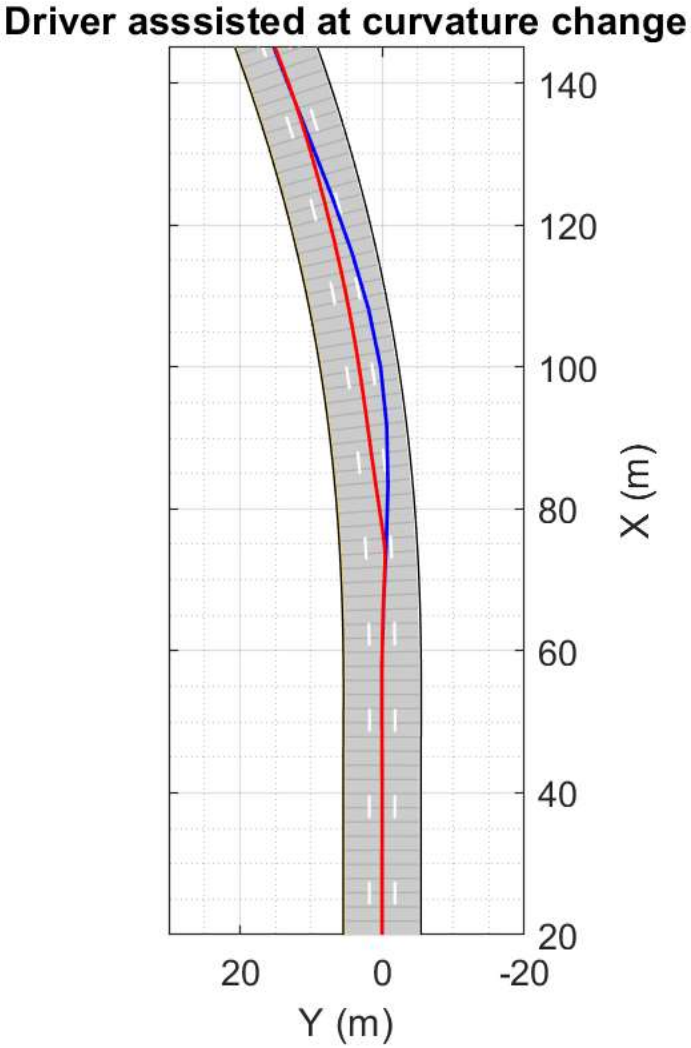
Simulate lane keep assist at distraction events



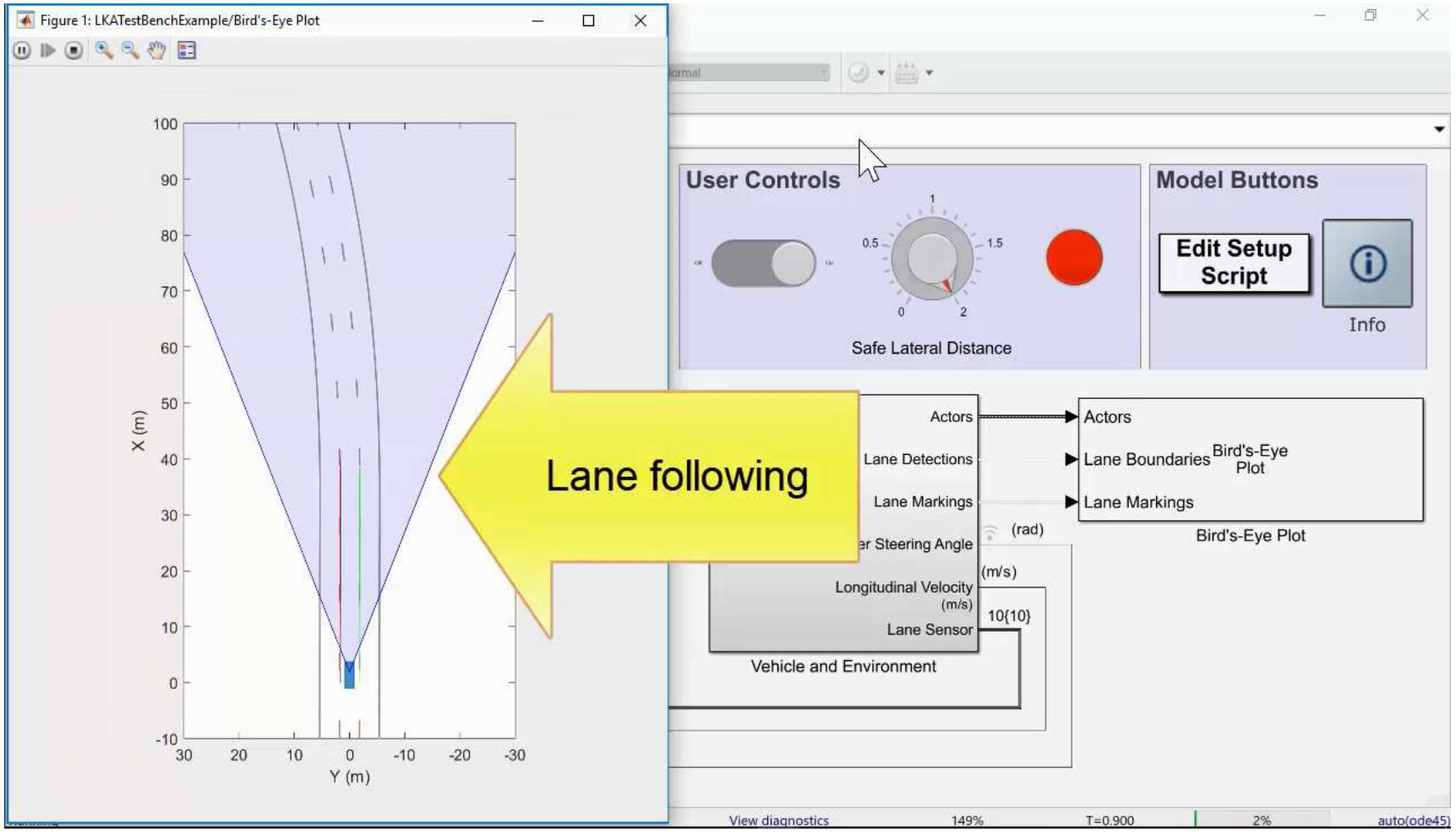
Compare distracted and assisted results

- Detect lane departure and maintain lane during distraction

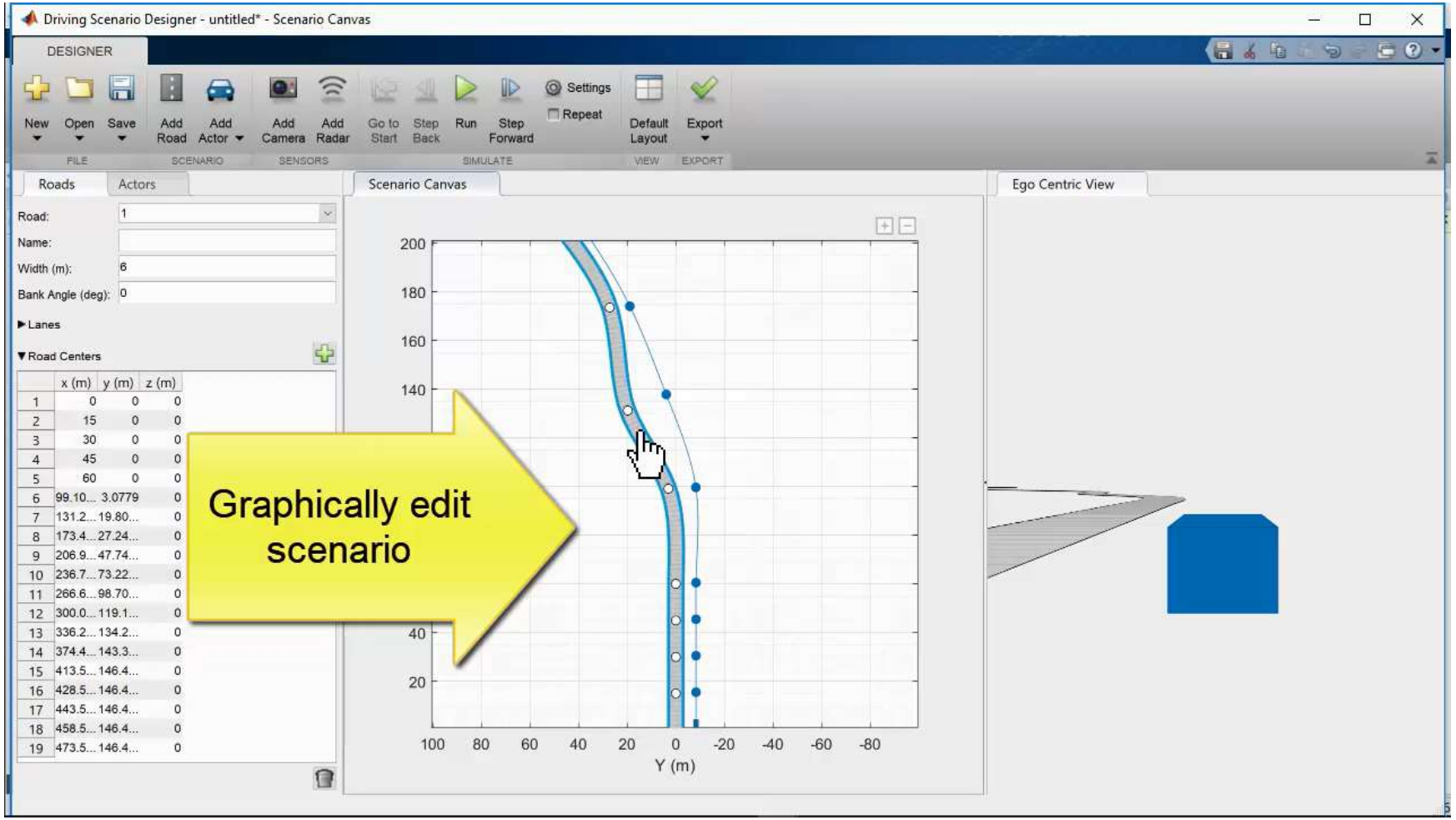




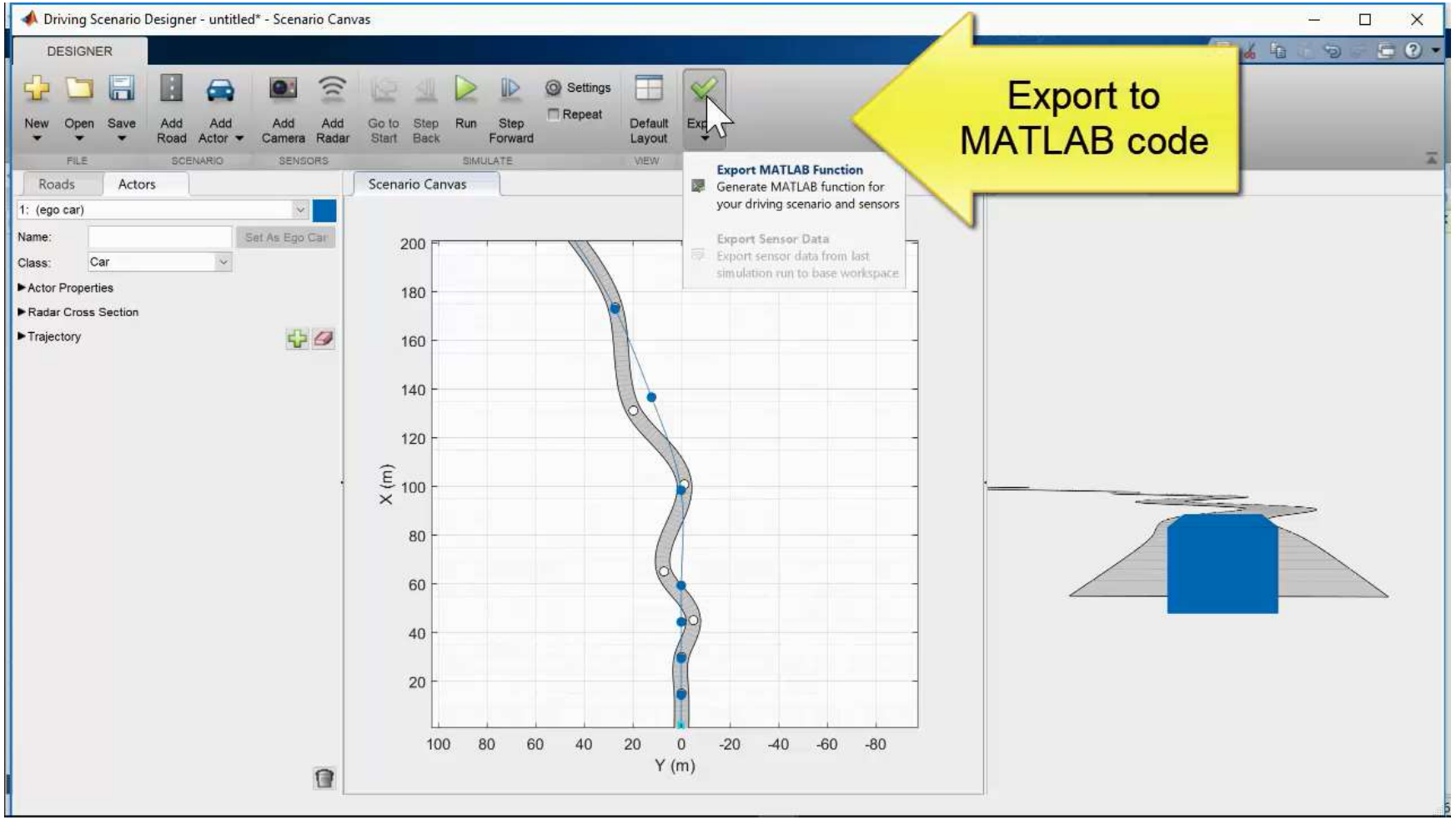
Simulate lane following by increasing minimum safe distance



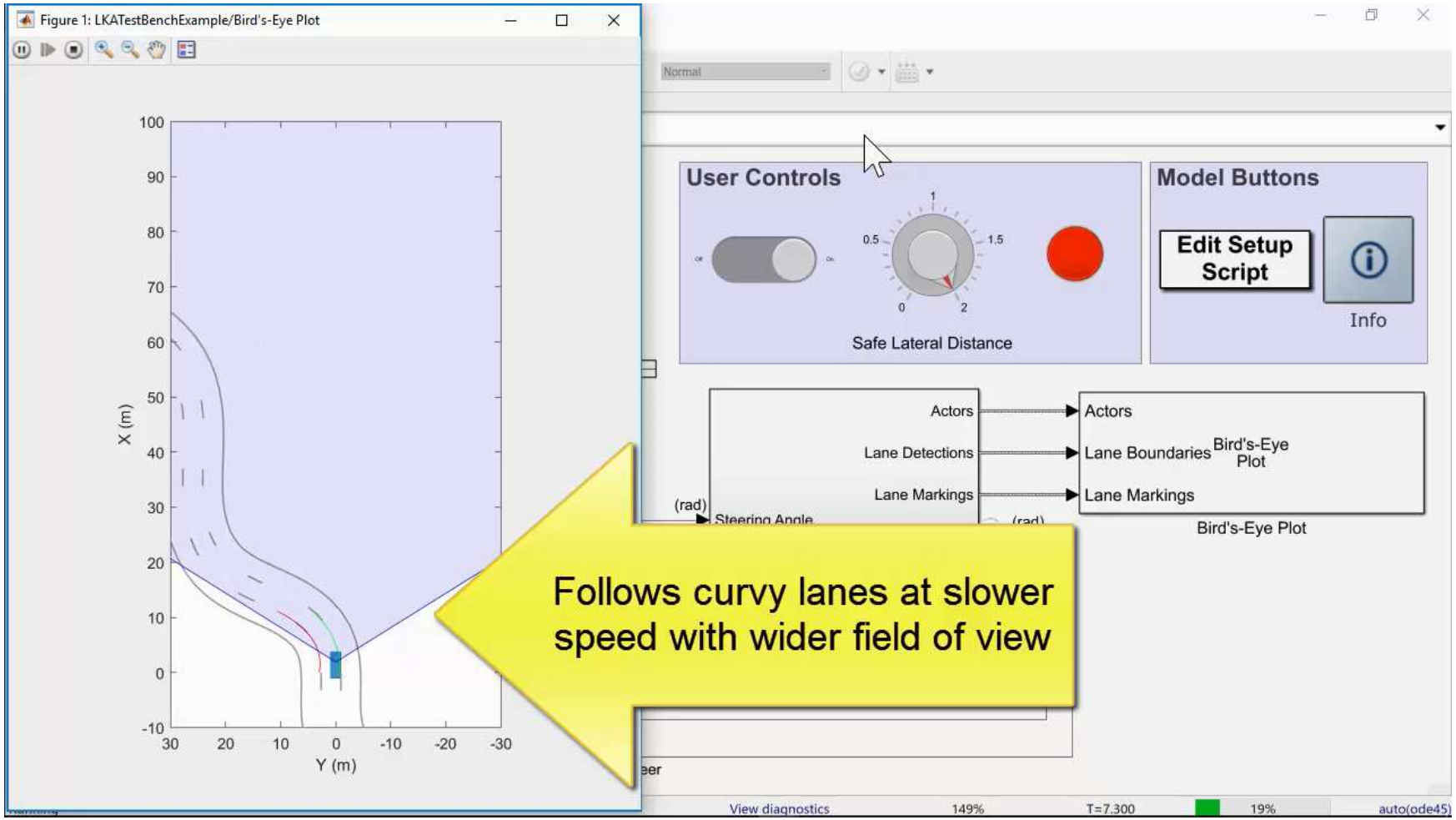
Graphically edit scenarios with Driving Scenario Designer



Export MATLAB code to generate scenarios

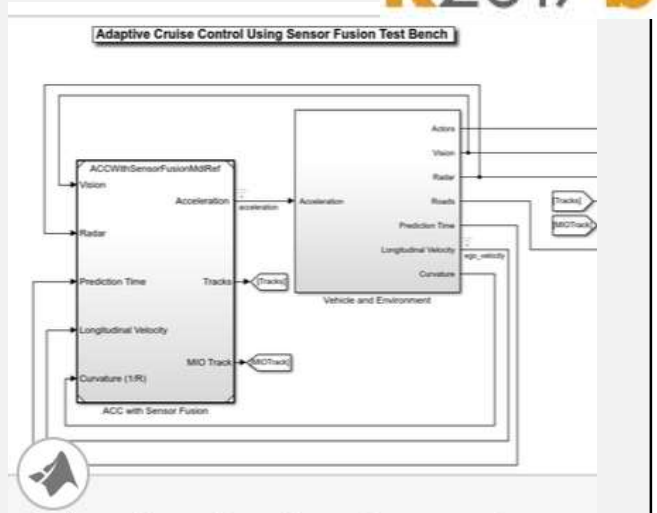


Explore what is required to follow high curvature paths



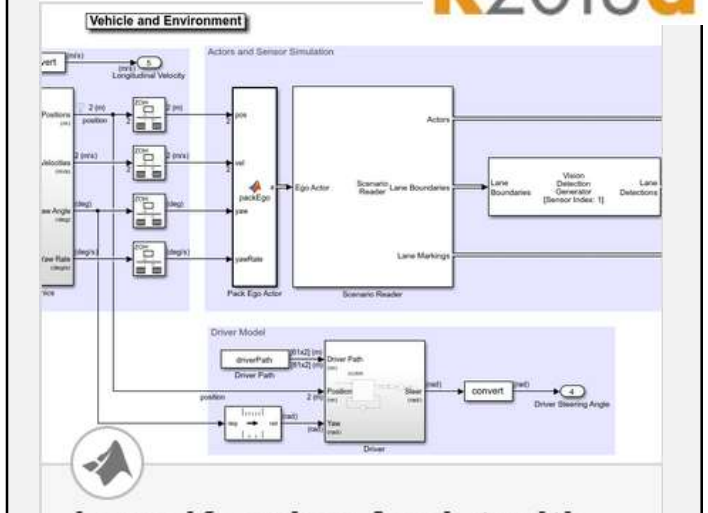
Learn about synthesizing sensor detections to develop control algorithms with these examples

R2017b



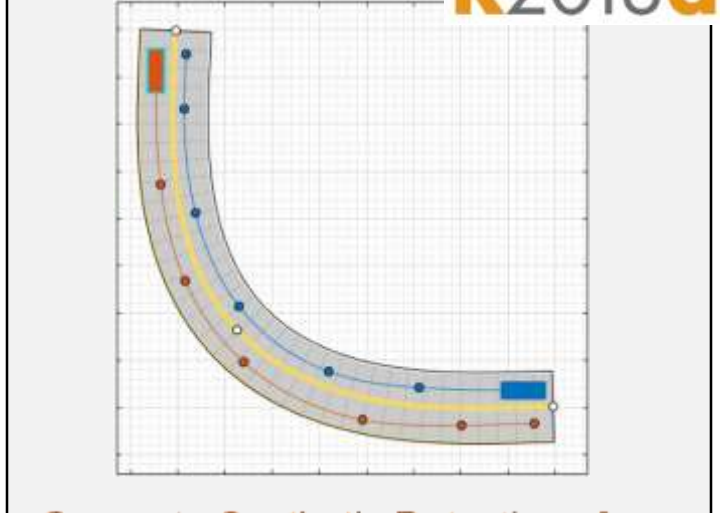
- **Simulate and generate C++** for model-predictive control and sensor fusion algorithms

R2018a



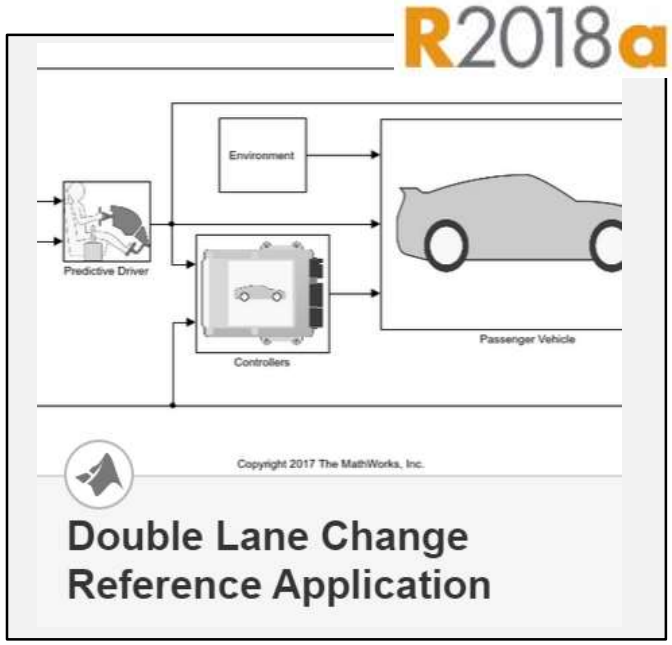
- **Simulate and generate C++** for model-predictive control with lane detections

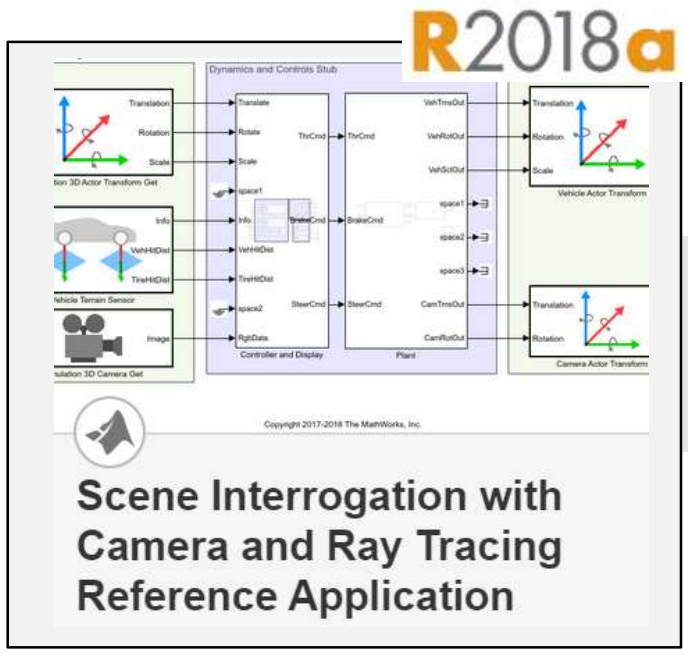
R2018a



- **Edit roads, cuboid actors, and sensors** with Driving Scenario Designer App `drivingScenarioDesigner`

Learn about modeling vehicle dynamics to develop control algorithms with these examples





- **Simulate vehicle dynamics** for closed loop design

  Vehicle Dynamics Blockset™

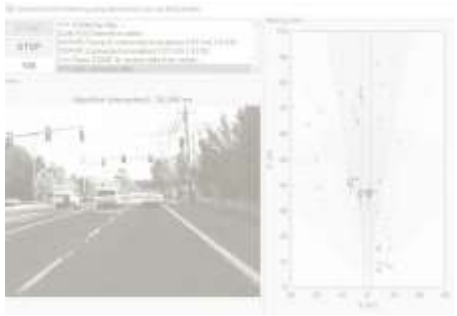
- **Co-simulate with Unreal Engine** and to set actor positions get camera image

  Vehicle Dynamics Blockset™

How can you use MATLAB and Simulink to develop planning algorithms?

Deep learning

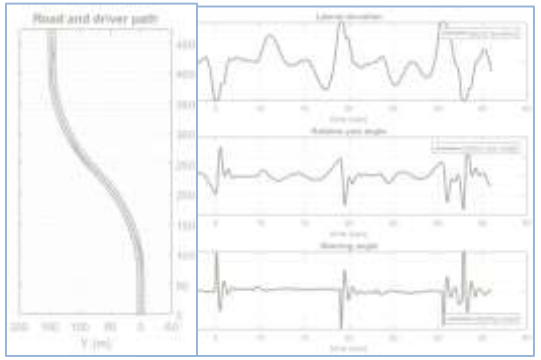
Sensor fusion with live data



Perception

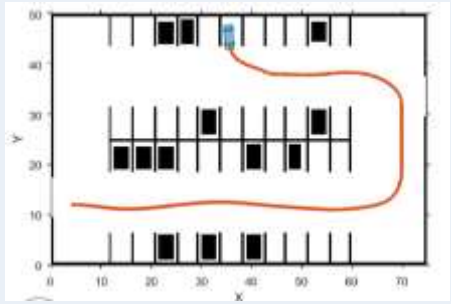
Control

Sensor models & model predictive control



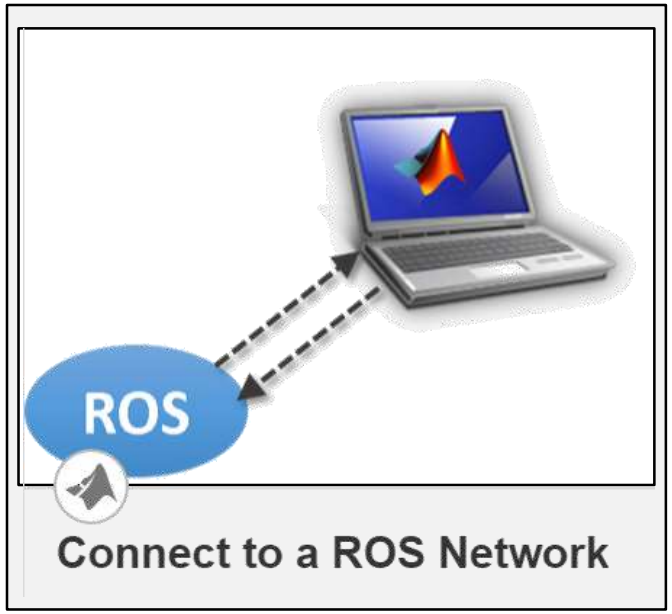
Planning

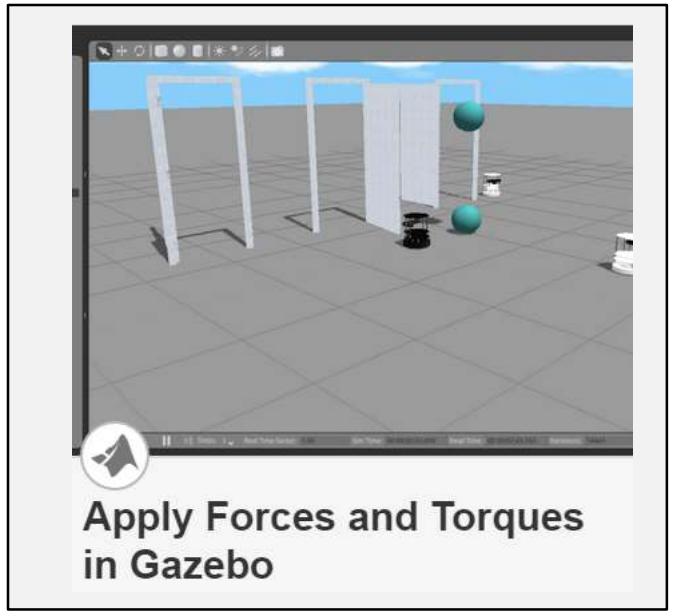
Path planning

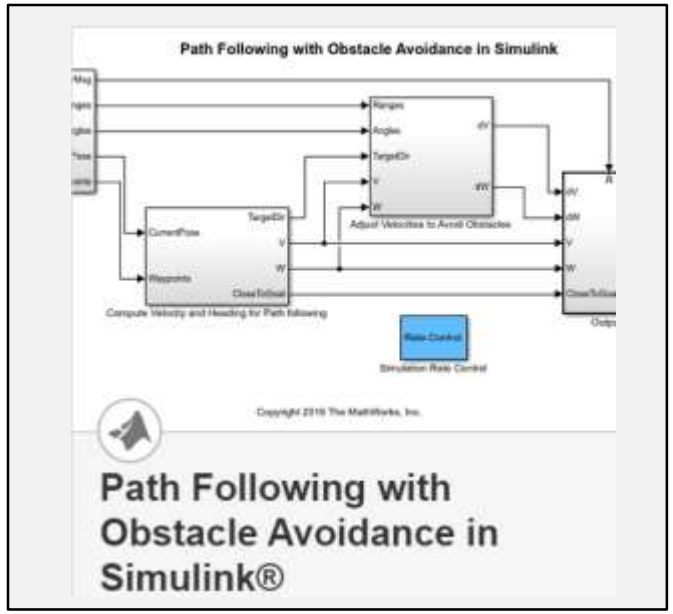


Robotics System Toolbox introduced:

Connectivity with the ROS ecosystem

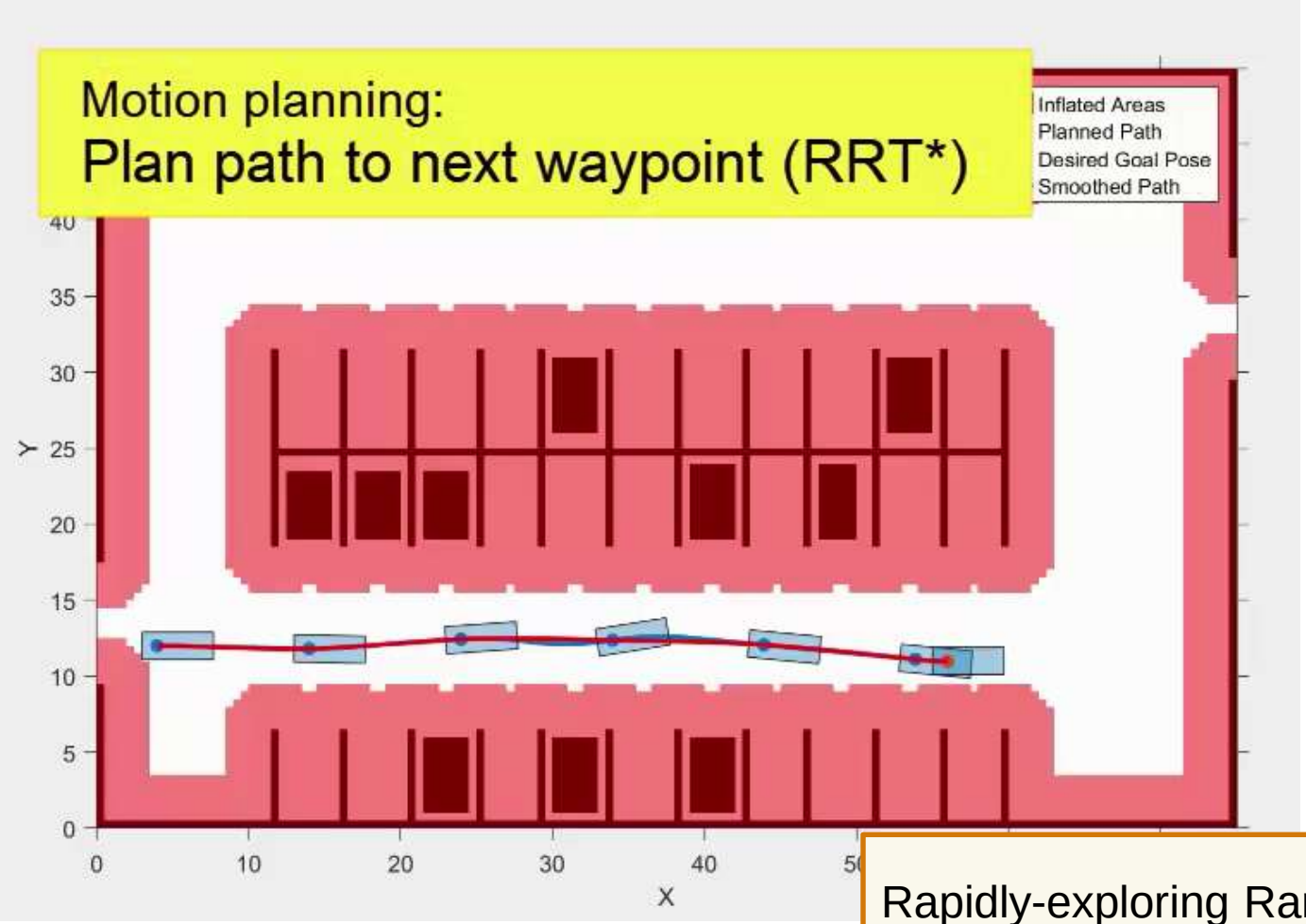






- **Communicate via ROS** to integrate with externally authored ROS components
- **Communication with Gazebo** to visualize and simulated system
- **Follow path** for differential drive robot with ROS based simulator

We are investing in design and simulation of path planning for automobiles

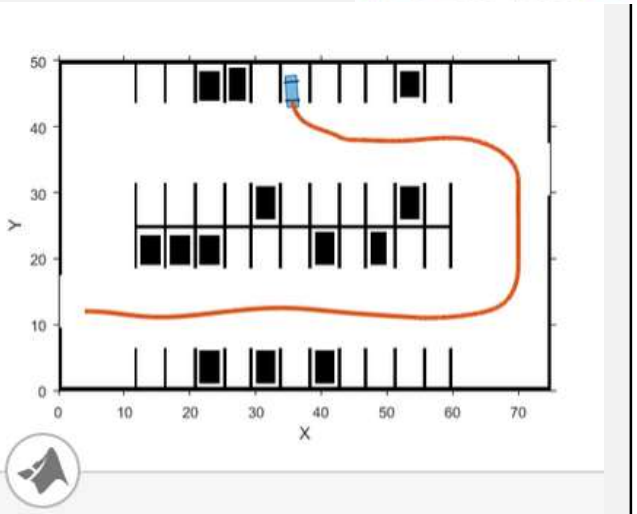


R2018a

Rapidly-exploring Random Tree (RRT*)

Learn about developing path planning algorithms with these examples

R2018a

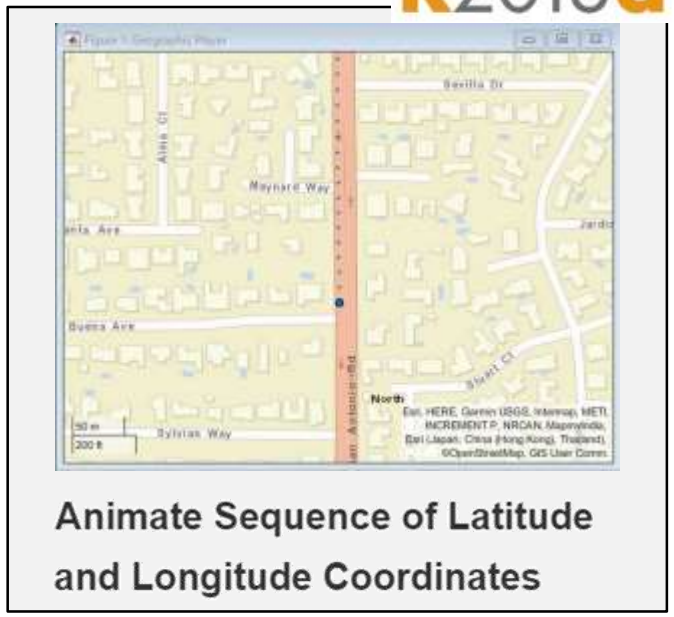


- **Plan path** for automobile given pre-defined map

  Automated Driving

  System Toolbox™

R2018a

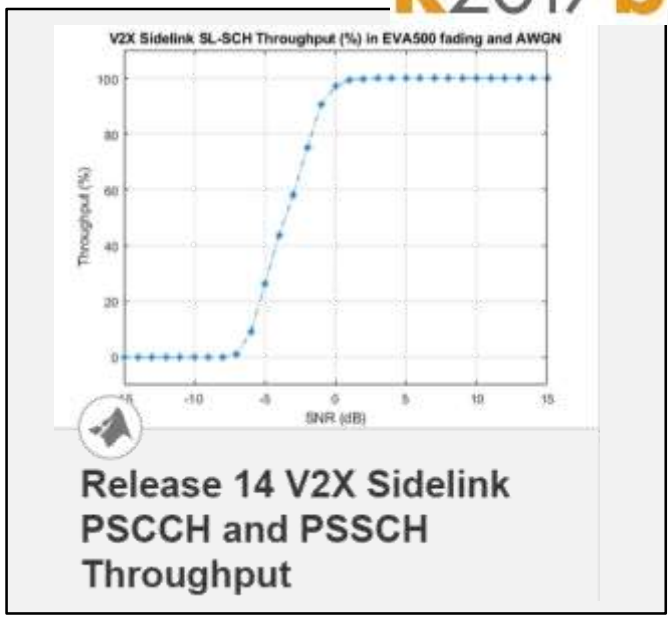


- **Plot map tiles** using World Street Map (Esri)

  Automated Driving

  System Toolbox™

R2017b



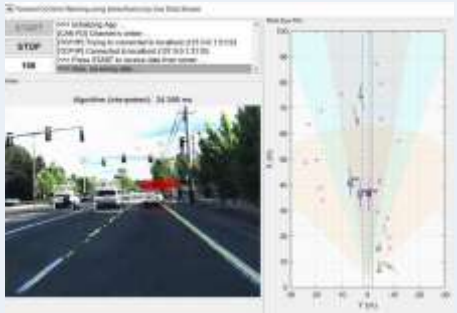
- **Simulate V2X communication** to assess channel throughput

  LTE System Toolbox™

Examples of how you can use MATLAB and Simulink to develop automated driving algorithms

Deep learning

Sensor fusion with live data

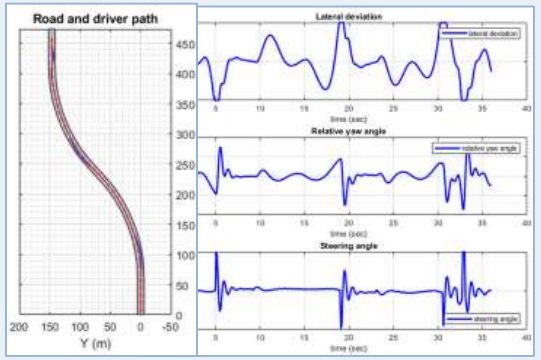


Perception

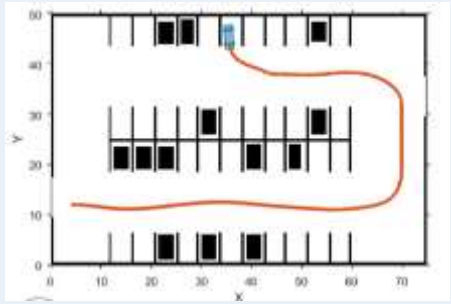
Control

Planning

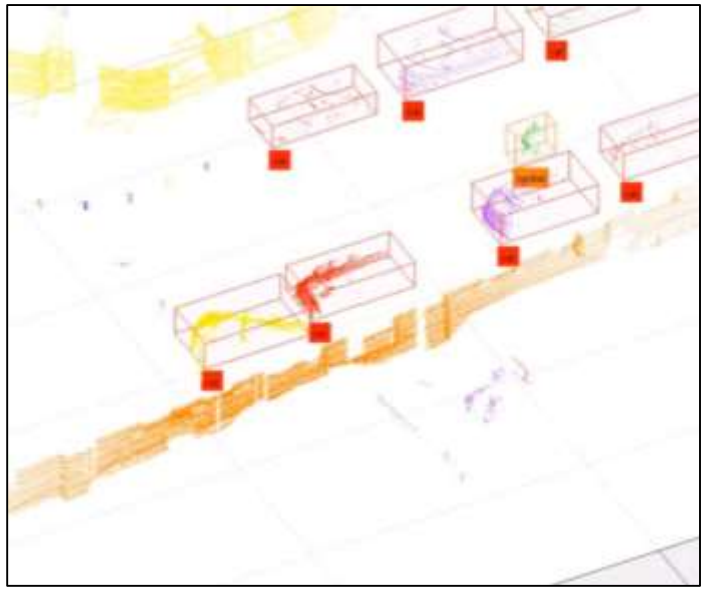
Sensor models & model predictive control

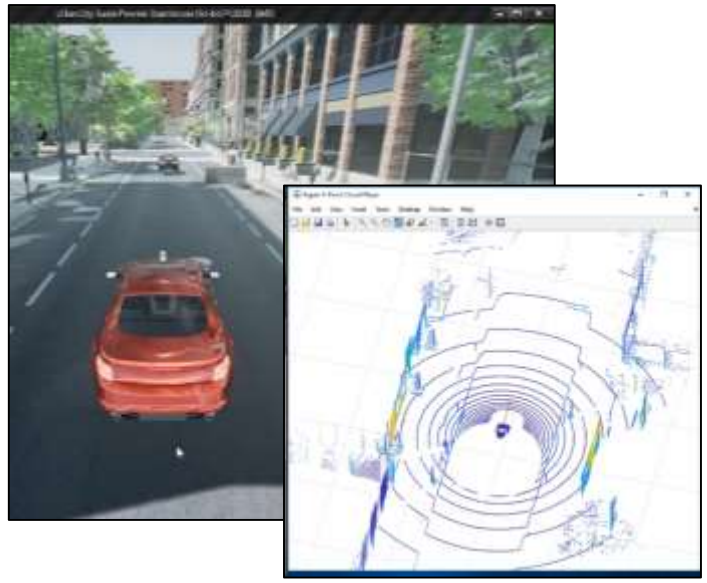


Path planning



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



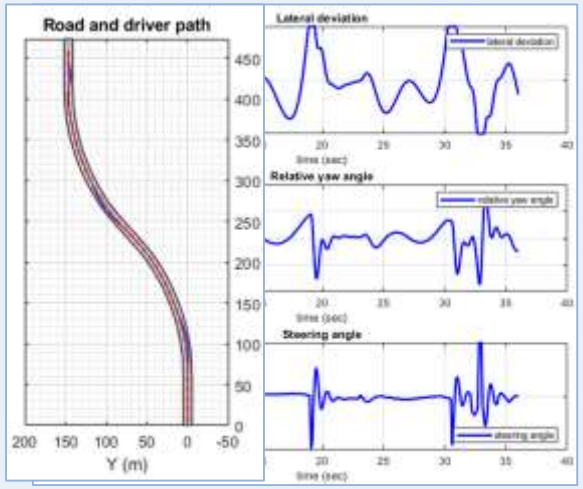


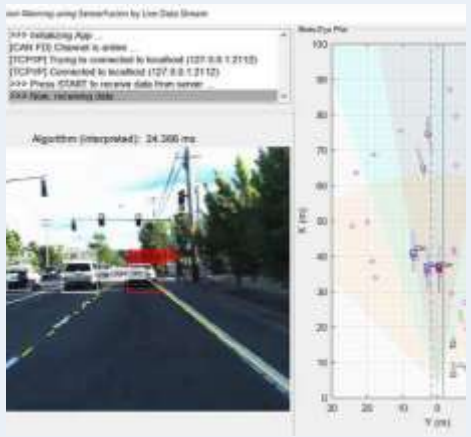
- **Web based ground truth labeling**
  - Consulting project with Caterpillar
  - 2017 MathWorks Automotive Conference
- **Lidar ground truth labeling**
  - Joint presentation with Autoliv
  - SAE Paper 2018-01-0043
  - 2018 MathWorks Automotive Conference
- **Lidar sensor model for Unreal Engine**
  - Joint paper with Ford
  - SAE Paper 2017-01-0107

How can we help you can use MATLAB and Simulink to develop automated driving algorithms?

Perception

Control





Planning