

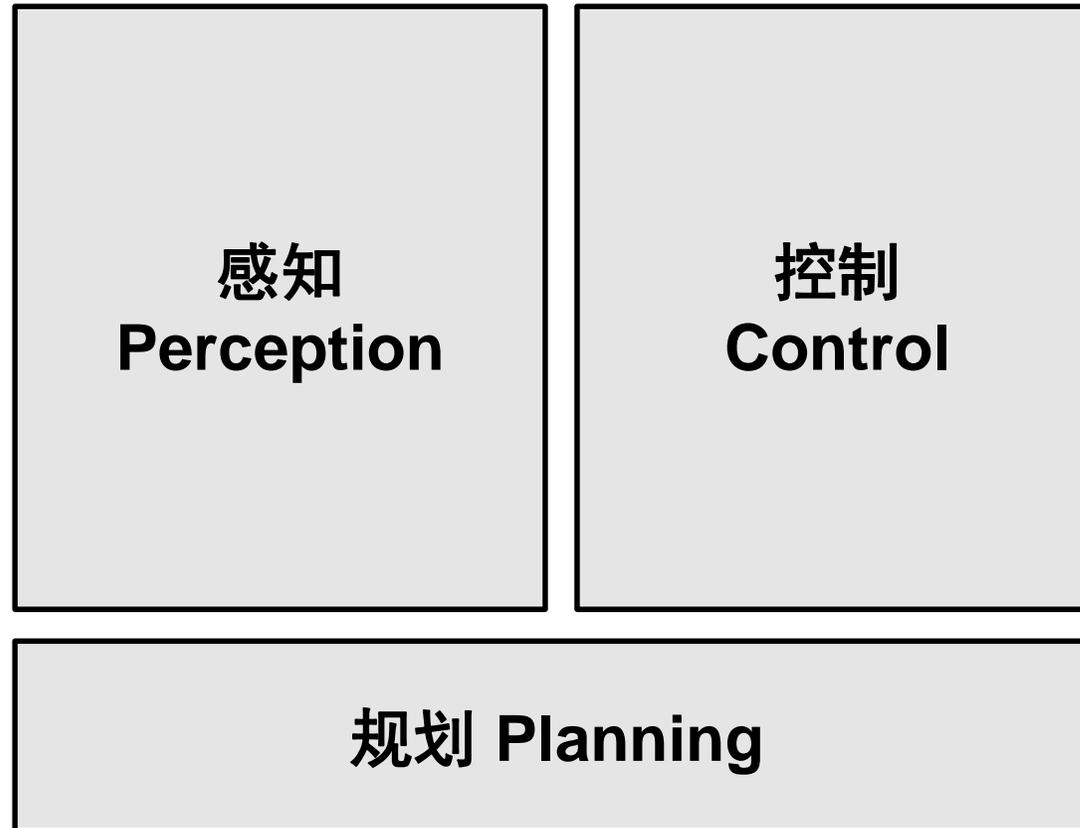
MATLAB EXPO 2018

使用 MATLAB 和 Simulink 开发 自动驾驶

王鸿钧, MathWorks 中国



您可以如何使用 MATLAB 和 Simulink 开发自动驾驶算法？



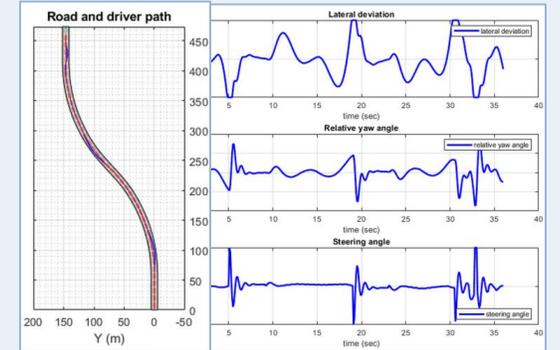
使用 MATLAB 和 Simulink 开发自动驾驶算法的案例

深度学习



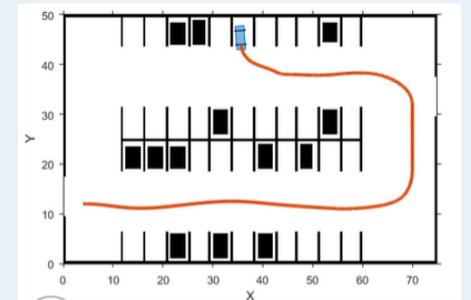
感知
Perception

传感器建模 &
模型预测控制



控制
Control

路径规划



对实时数据的
传感器融合



规划
Planning

使用 MATLAB 和 Simulink 开发自动驾驶算法的案例

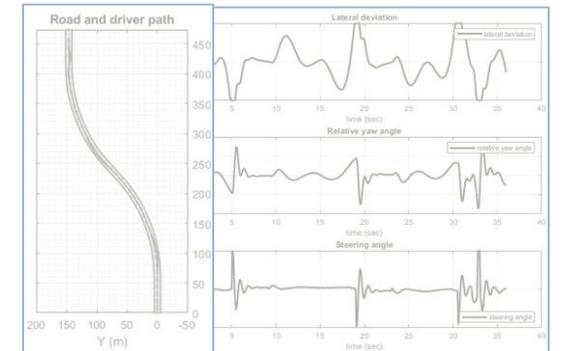
深度学习



感知
Perception

控制
Control

传感器建模 &
模型预测控制

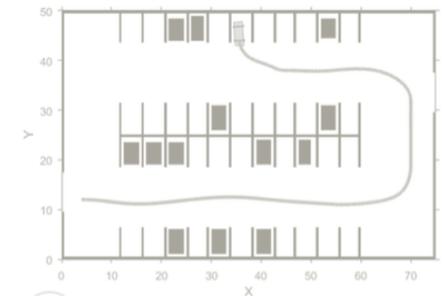


对实时数据的
传感器融合



规划
Planning

路径规划



自动驾驶系统工具箱 Automated Driving System Toolbox 为标注视频数据，引入真实值标注程序（Ground Truth Labeler）

R2017a

使用 Ground Truth Labeler 自动标注车道线

The screenshot displays the Ground Truth Labeler software interface. A yellow arrow points to the 'Run automation algorithm' button in the top toolbar. The interface includes a central video viewer showing a street scene with lane markings. On the left, there are panels for 'ROI Label Definition' (listing 'laneMarker', 'Road', and 'Sky') and 'Scene Label Definition' (listing 'Sunny' and 'Cloudy'). On the right, there is an 'Auto Lane Detection' panel with instructions and a 'Run the algorithm' button. At the bottom, a timeline shows the current frame at 01.30000, with start, end, and max time values.

Run automation algorithm

ROI Label Definition

- Label
- Sublabel
- Attribute
- laneMarker
- Road
- Sky

Scene Label Definition

- Sunny
- Cloudy

Auto Lane Detection

Load a MonoCamera configuration object from the workspace using the settings panel

Specify additional parameters in the settings panel

Run the algorithm

Manually inspect and modify results if needed

01.30000 01.30000 02.47726 08.33334
Start Time Current End Time Max Time

Zoom Out Time Interval

在 Ground Truth Labeler 中定义区域的子标签和属性

R2018a

The screenshot displays the Ground Truth Labeler interface. On the left, the 'ROI Label Definition' panel lists labels: 'cyclist' (green), 'bicycle' (green), and 'vehicle' (purple). The central video frame shows a cyclist and a car with bounding boxes and labels. The 'Attributes and Sublabels' panel on the right shows 'Attributes for cyclist' with 'bikeType' set to 'bicycle' and 'action' set to 'inMotion'. A bottom panel shows 'Scene Label Definition' options and a timeline with 'Start Time', 'Current', 'End Time', and 'Max Time' markers.

使用 Ground Truth Labeler 自动标注像素点

The screenshot displays the Ground Truth Labeler software interface. The main window shows a video frame from 'caltech_cordova1.avi' with a road scene. The road is labeled in blue, and the sky is labeled in orange. A yellow arrow points to the 'Automate pixel labeling' button in the top toolbar.

Automate pixel labeling

ROI Label Definition

Label	Sublabel	Attribute
laneMarker		
Road		
Sky		

Scene Label Definition

Define new scene label

Current Frame Add Label

Time Interval Remove Label

Sunny	
Cloudy	

Timeline

Start Time: 01.33334 | Current: 01.33334 | End Time: 02.47726 | Max Time: 08.33334

Zoom Out Time Interval

Review and Modify: Review automated labels over the interval using playback controls. Modify/delete/add ROIs that were not satisfactorily automated at this stage. If the results are satisfactory, click Accept to accept the automated labels.

Accept/Cancel: If results of automation are satisfactory, click Accept to accept all automated labels and return to manual labeling. If results of automation are not satisfactory, click Cancel to return to manual labeling without saving automated labels.

通过这个案例了解如何训练一个深度学习网络

R2017b



Semantic Segmentation Using Deep Learning

使用深度学习训练可行
驶区域检测网络

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 also uses:

[Neural Network Toolbox](#)

[vgg16](#)

Open Script

This example uses the [Computer Vision System Toolbox™](#) [Image Segmentation](#) block to process the street-level views obtained from the [Image Segmentation](#) block.

Learn more about [Image Segmentation](#)

Setup

This example creates the [Neural Network Toolbox™ Model for VGG-16 Network](#) block.

```
vgg16();
```

Manage Add-Ons

🏠

Search for add-ons 🔍

VGG-16

RETRAINED MODEL

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

载入和显示训练图像

```
% Create datastore for images  
imds = imageDatastore(imgDir);  
I = readimage(imds, 1);  
I = histeq(I);  
imshow(I)
```

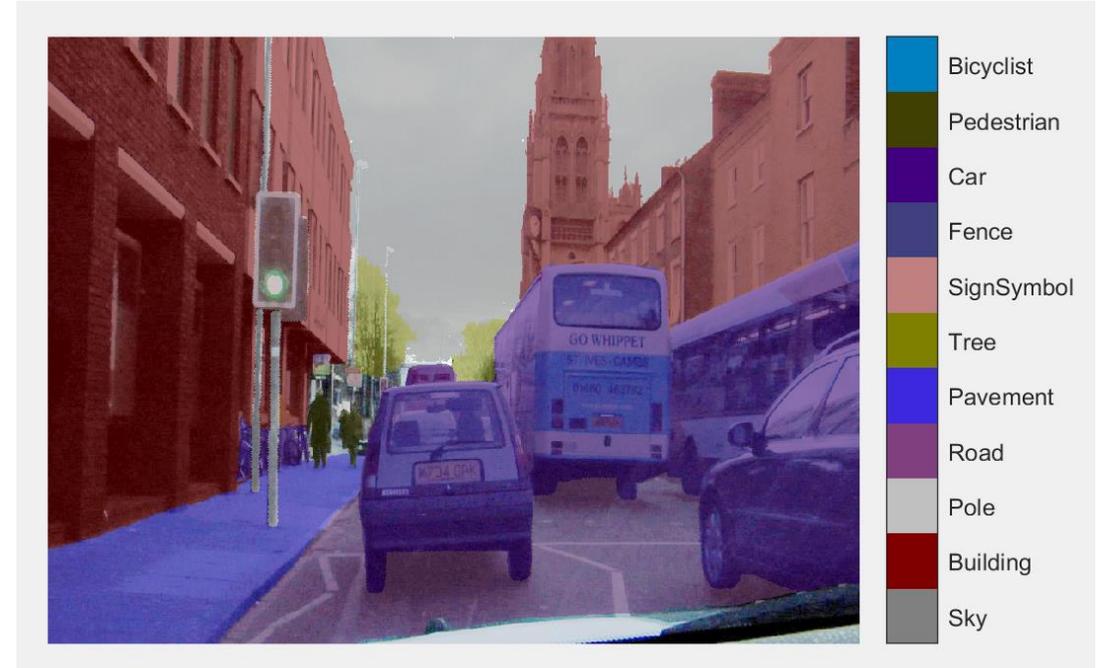


imageDatastore
可管理大型的图像数据集

载入和叠加显示被标注像素点

```
% 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)
```



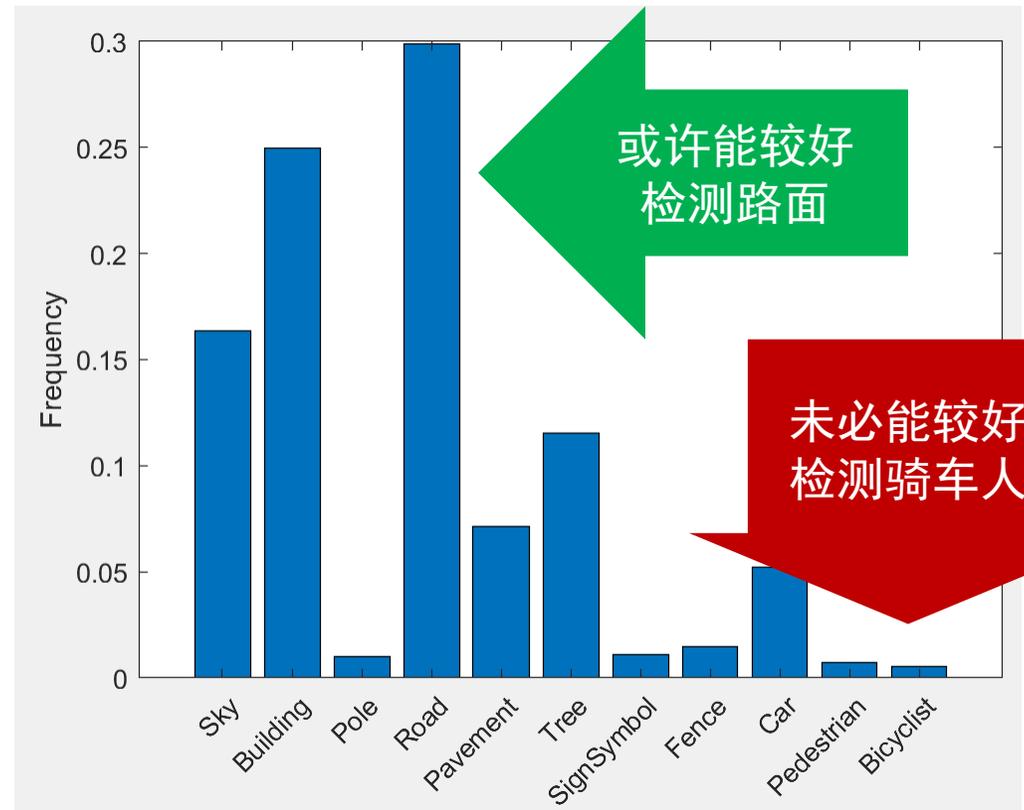
pixelLabelDatastore
可管理大型的像素点标签数据集

可视化被标注像素点的分布

```
% 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')
```



在这个集合中被标注像素点的分布是不均衡的

创建和可视化基础网络

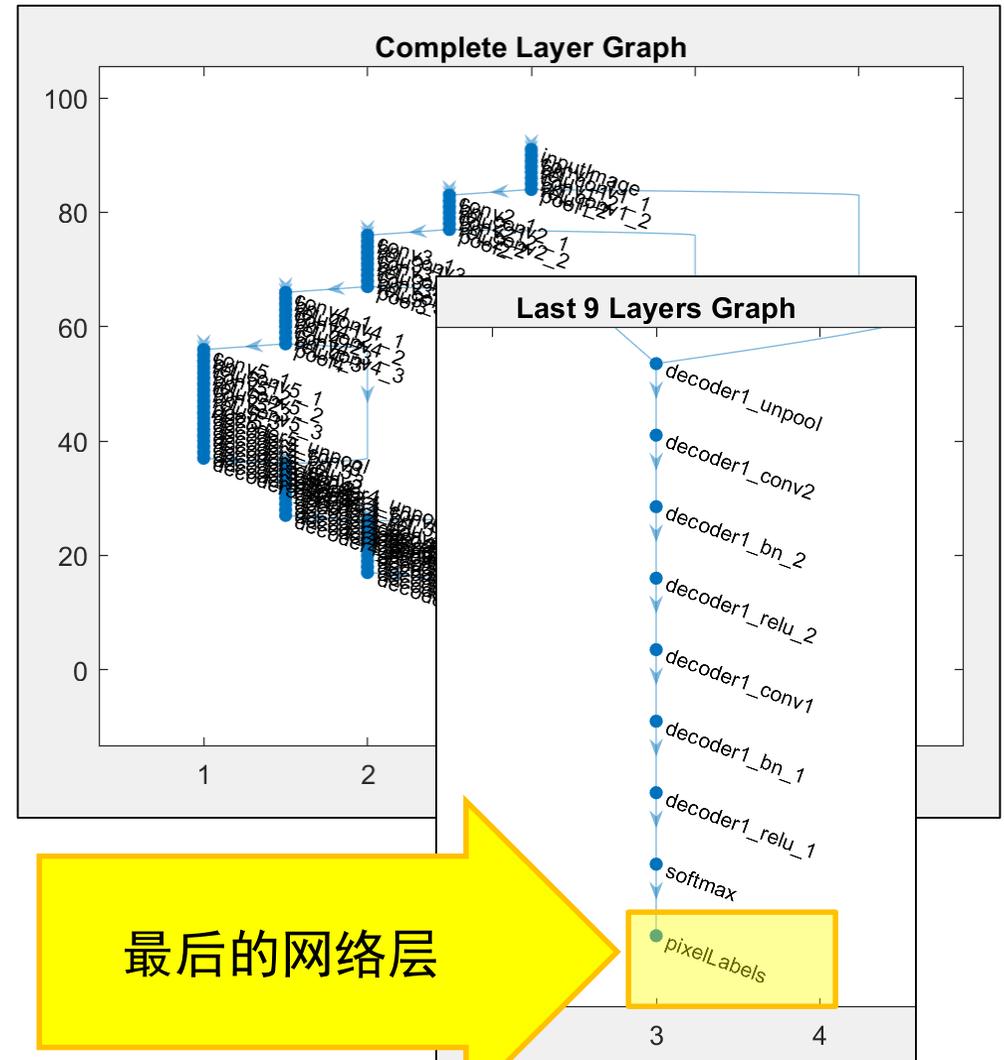
```

% Create SegNet architecture
lgraph = segnetLayers(...
    imageSize, numClasses,...
    'vgg16');

% Display network structure
plot(lgraph)
title('Complete Layer Graph')%

% Display last layers
plot(lgraph); ylim([0 9.5])
title('Last 9 Layers Graph')

```



最后的网络层

添加权重层，补偿不均衡的数据集

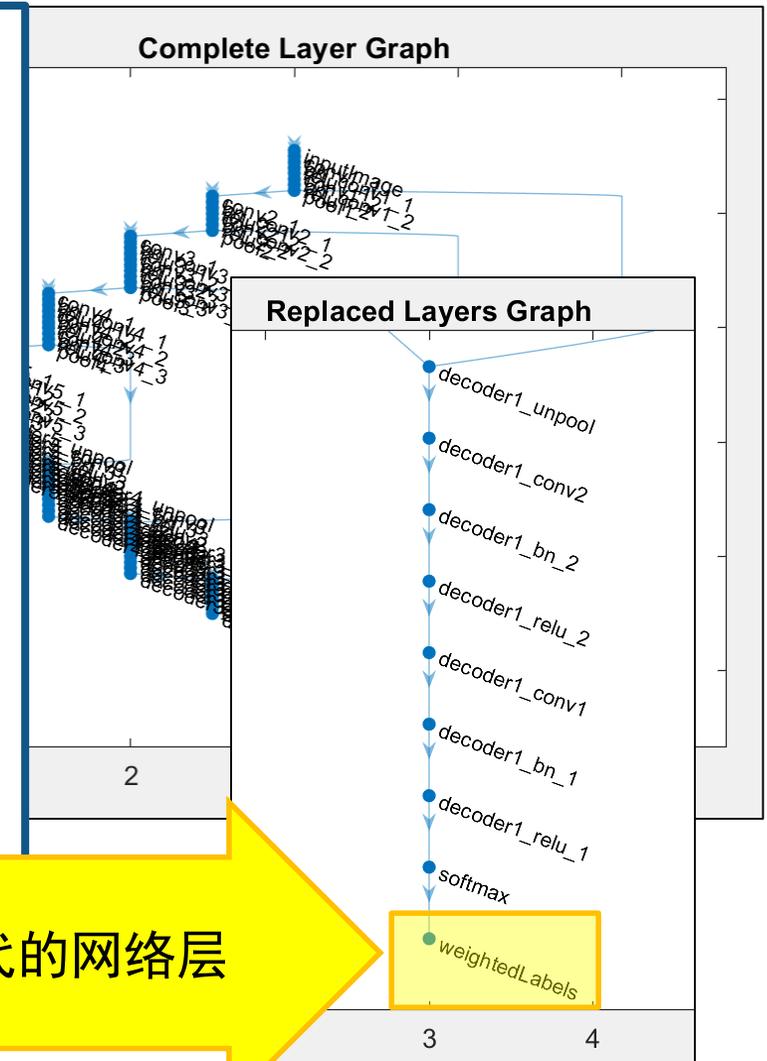
```

% 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');

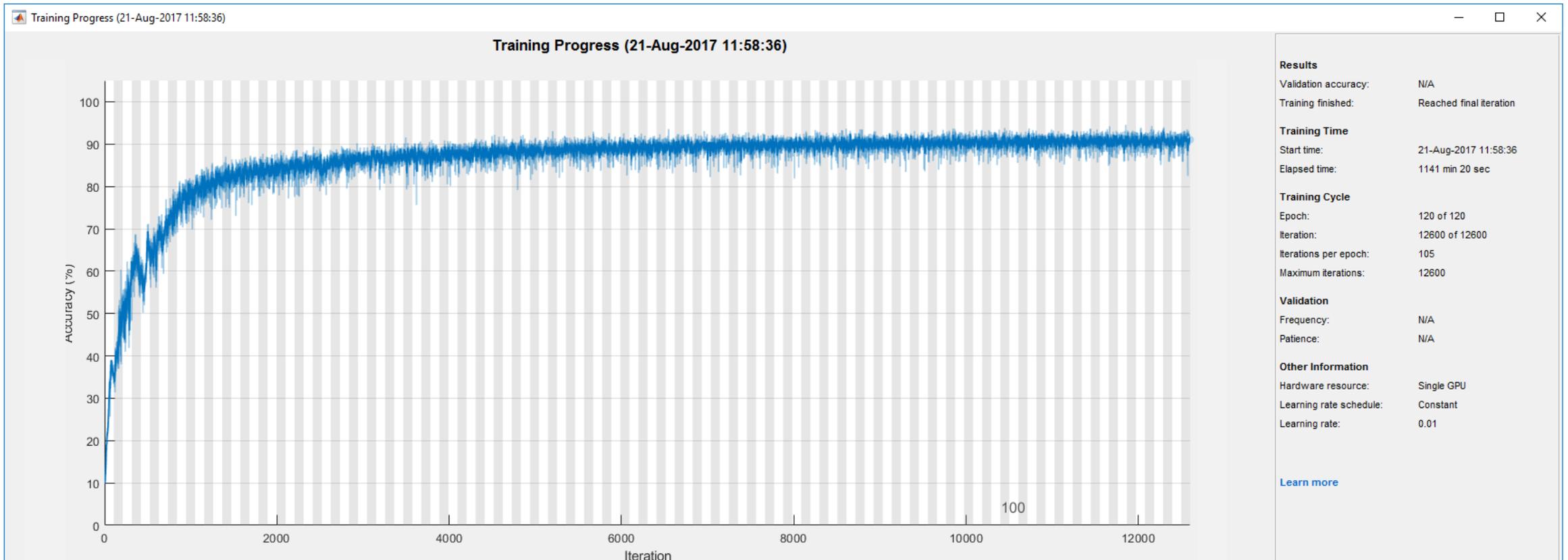
% Display network structure
plot(lgraph); ylim([0 9.5])
title('Replaced Layers Graph')

```



训练网络并观察训练过程

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

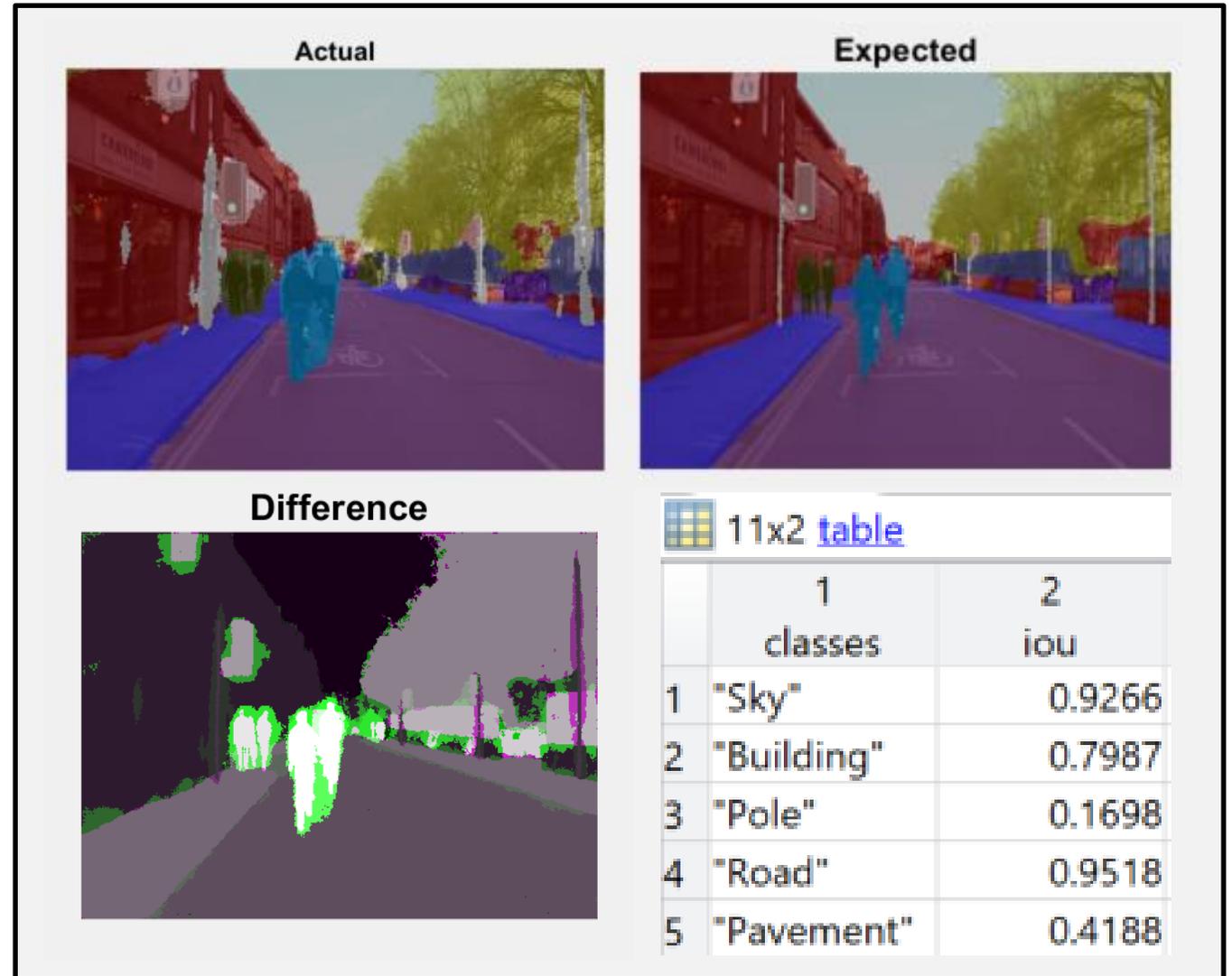


计量交并比 (IoU) , 评估相似度

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

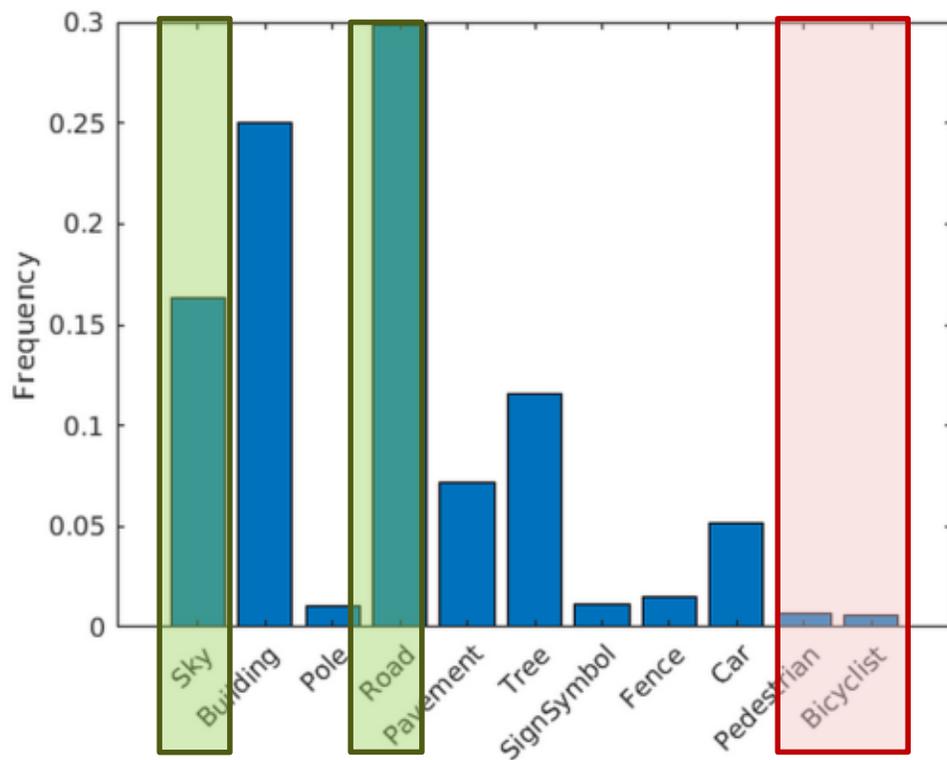
```
ans =
  11x2 table
    classes      iou
  _____  _____

  "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
```



数据中的标签分布对 IoU 的影响

在原始数据集中的标签分布



网络的计量评估结果

	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

未被充分表现的类例如行人、骑车人，没有像天空、道路一样获得良好的分割

使用语义分割检测可行驶区域



了解更多的深度学习感知算法开发案例

R2018a



Automate Ground Truth Labeling for Semantic Segmentation

R2017b



Semantic Segmentation Using Deep Learning

R2018a



Code Generation for Semantic Segmentation Network

- 添加语义分割自动算法到 Ground Truth Labeler 工具 Automated Driving System Toolbox™

- 训练深度学习网络可行驶空间检测应用 Computer Vision System Toolbox™

- 生成 CUDA® 代码在 NVIDIA GPU 上运行 DAG 网络 GPU Coder™

使用 MATLAB 和 Simulink 开发自动驾驶算法的案例

深度学习



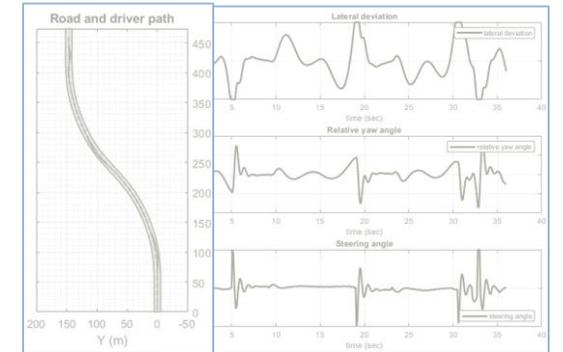
感知
Perception

对实时数据的
传感器融合



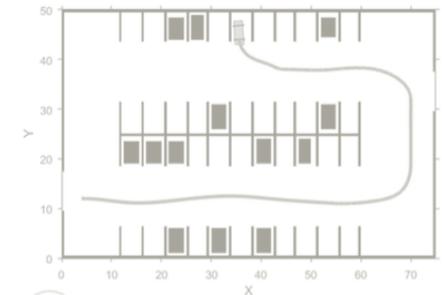
控制
Control

传感器建模 &
模型预测控制

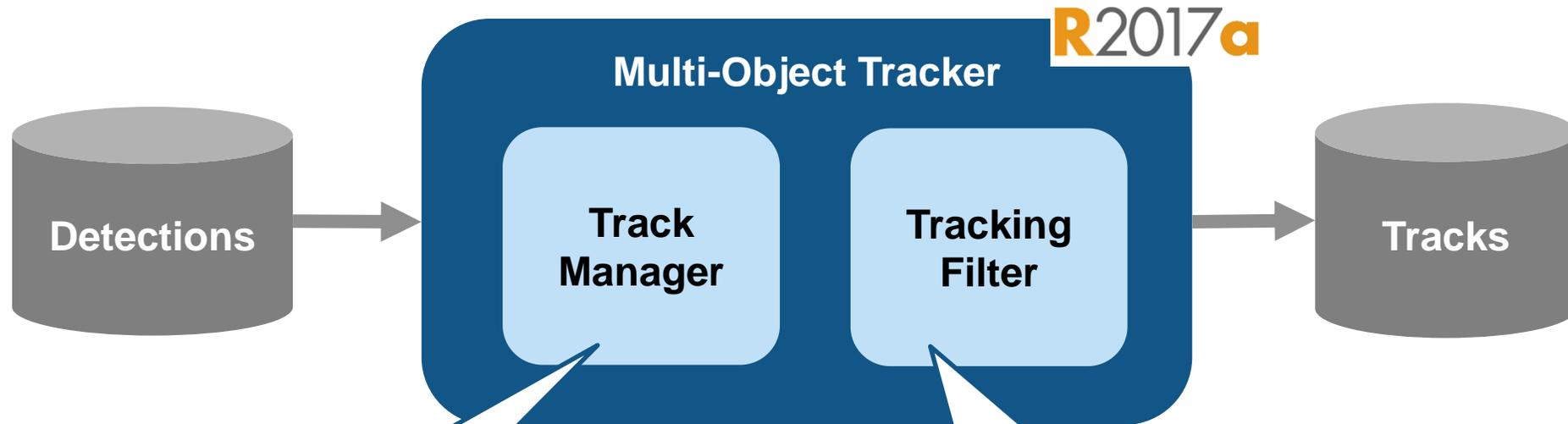


规划
Planning

路径规划



自动驾驶系统工具箱 Automated Driving System Toolbox 为开发传感器融合算法，引入多目标跟踪器（Multi-object tracker）



- 分配检测结果到目标跟踪
- 创建新的目标跟踪
- 更新已经存在的目标跟踪
- 删除过期的目标跟踪

- 目标跟踪的状态预测与更新
- 支持线性、扩展、无迹卡尔曼滤波

Videos and Webinars

Some common questions from automated driving engineers

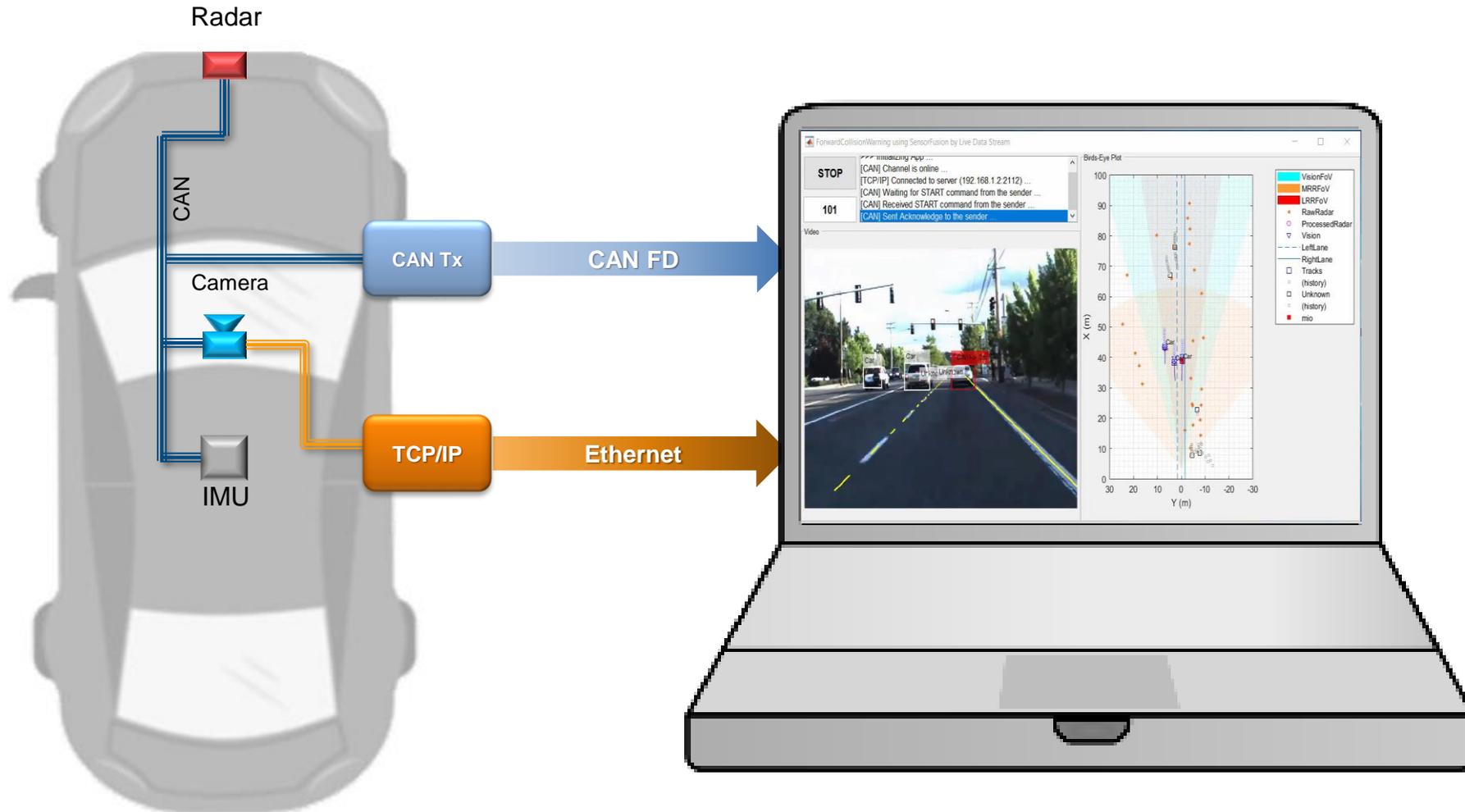
How can I visualize vehicle data?

19:27

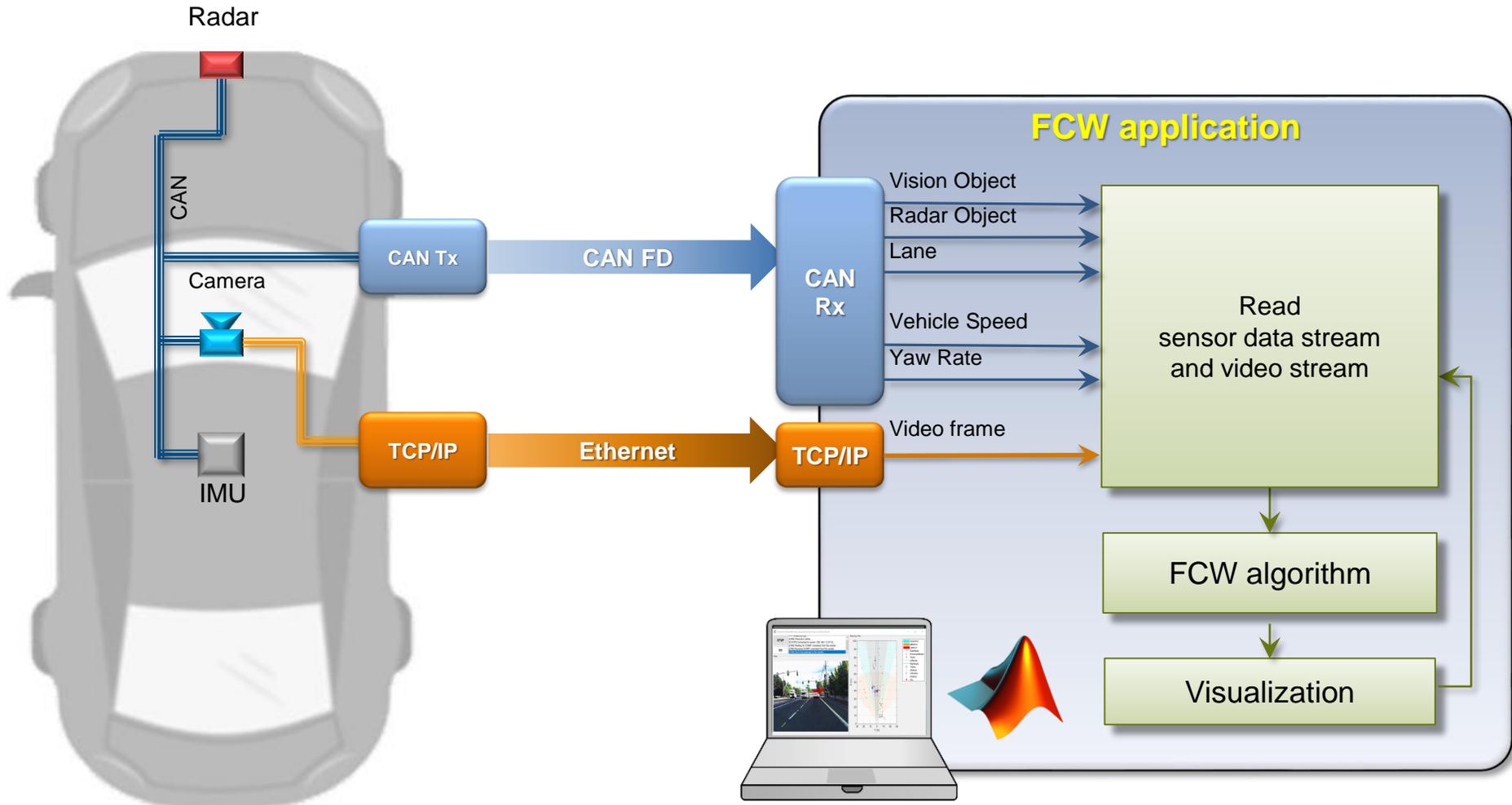
How can I fuse multiple detections?

Introduction to Automated Driving System Toolbox

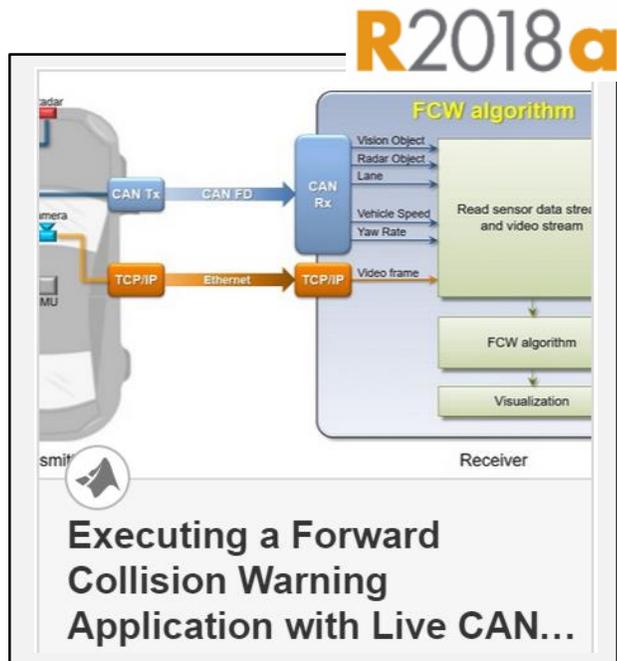
如何使用实时数据测试我的传感器融合算法？



使用来自车辆的实时数据测试前向碰撞预警算法



通过这个案例了解如何开发对实时数据的传感器融合算法

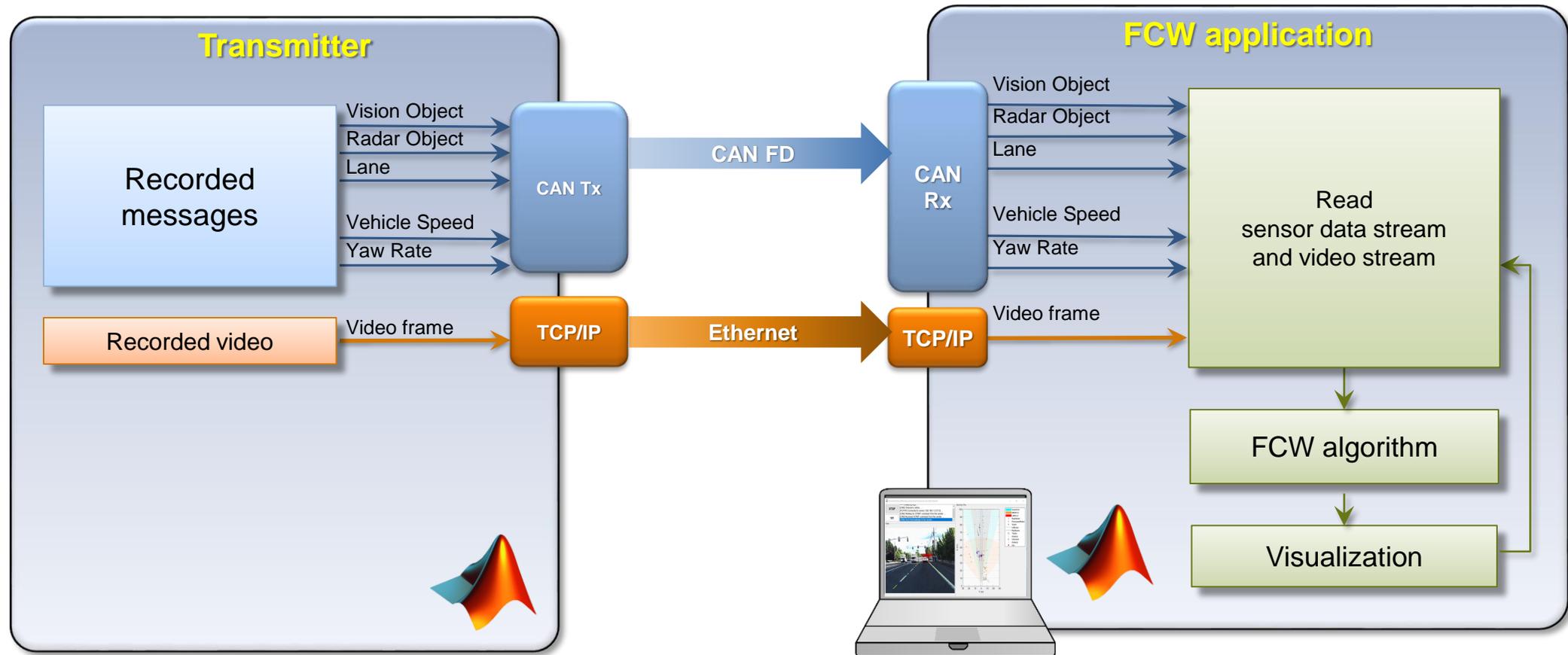


- 传输 CAN FD 数据流
在笔记本电脑上验证原型算法

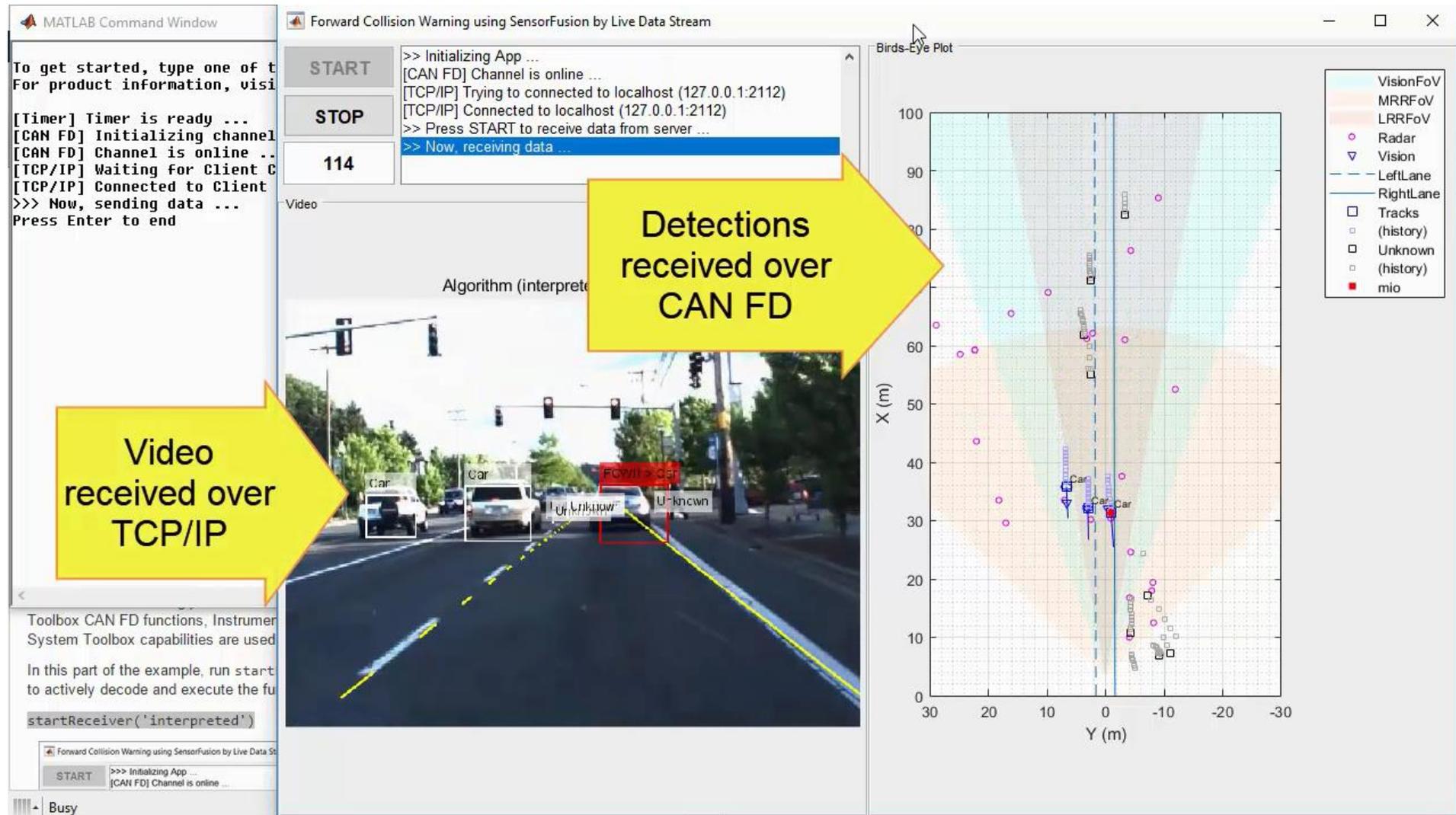
Vehicle Network Toolbox™



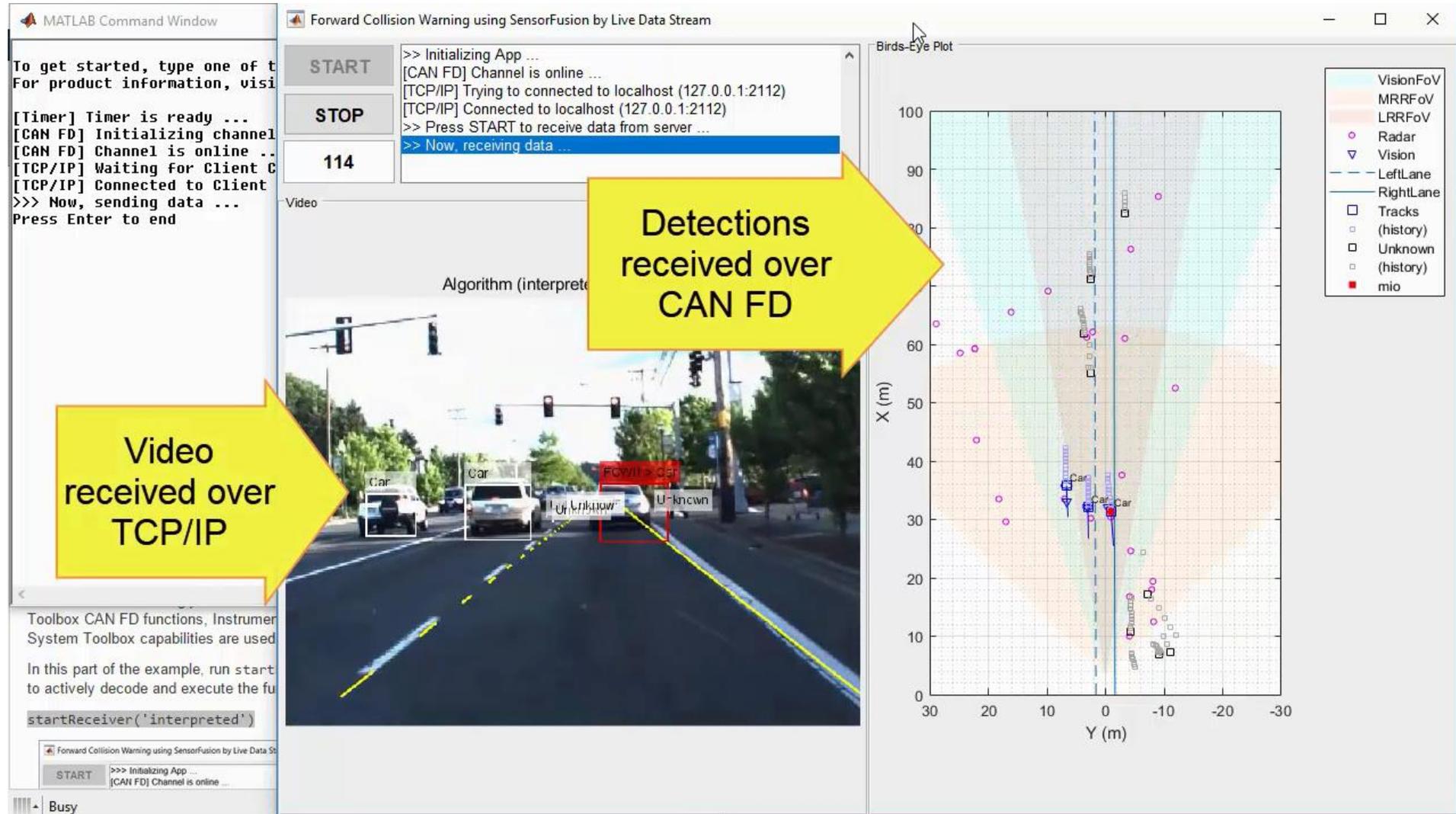
使用来自车辆替代者的实时数据测试前向碰撞预警算法



通过 CAN FD 和 TCP/IP 发送实时数据



通过 CAN FD 和 TCP/IP 接收实时数据



生成前向碰撞预警算法的 C/C++ 代码

MATLAB Coder Report Viewer - C:\MATLABExamples\VNT\codegen\lib\trackingForFCW_kernel\html\report.mldatx

REPORT

Back Forward Find Trace Code Edit In MATLAB Package Code

NAVIGATE TRACE EDIT SHARE

MATLAB SOURCE

Function List Call Tree

- trackingForFCW_kernel.m
 - trackingForFCW_kernel
 - fx calculateGroundSpeed
 - fx fcwmeas > 1
 - fx fcwmeas > 2
 - fx fcwmeasjac > 1
 - fx fcwmeasjac > 2
 - fx findMostImportantObject
 - fx findNonClutterRadarObjec
 - fx initConstantAccelerationFi
 - fx processDetections

GENERATED CODE

- trackingEKF.h
- trackingForFCW_kernel.c
- trackingForFCW_kernel.h
- trackingForFCW_kernel_e
- trackingForFCW_kernel_e
- trackingForFCW_kernel_e
- trackingForFCW_kernel_e
- trackingForFCW_kernel_e
- trackingForFCW_kernel_ir
- trackingForFCW_kernel_ir
- trackingForFCW_kernel_rt
- trackingForFCW_kernel_rt
- trackingForFCW_kernel_te

```

1565 *      const struct5_T *laneReports
1566 *      struct7_T *egoLane
1567 *      double time
1568 *      const double positionSelector[12]
1569 *      const double velocitySelector[12]
1570 *      emxArray_struct8_T *confirmedTracks
1571 *      double *numTracks
1572 *      struct10_T *mostImportantObject
1573 * Return Type : void
1574 */
1575 void trackingForFCW_kernel(const struct0_T *visionObjects, const struct2_T
1576 *radarObjects, const struct4_T *inertialMeasurementUnit, const struct5_T
1577 *laneReports, struct7_T *egoLane, double time, const double positionSelector
1578 [12], const double velocitySelector[12], emxArray_struct8_T *confirmedTracks,
1579 double *numTracks, struct10_T *mostImportantObject)
1580 {
1581     emxArray_objectDetection *detections;
1582     emxInit_objectDetection(&detections, 2);
1583     .....
1584

```

Generated C function

SUMMARY ALL MESSAGES (1) BUILD LOGS CODE INSIGHTS (0) VARIABLES

Code generation successful

Generated on: 17-Mar-2018 19:07:16
 Build type: Static Library
 Output file: C:\MATLABExamples\VNT\codegen\lib\trackingForFCW_kernel\trackingForFCW_kernel.lib
 Processor: Generic->MATLAB Host Computer

通过 CAN FD 和 TCP/IP 传输实时数据流，并运行编译后的代码

MATLAB Command Window

```

To get started, type one of the following commands.
For product information, visit www.mathworks.com.

[Timer] Timer is ready ...
[CAN FD] Initializing channel ...
[CAN FD] Channel is online ...
[TCP/IP] Waiting for Client Connection ...
[TCP/IP] Connected to Client ...
>>> Now, sending data ...
Press Enter to end

38 - disp('Configur
39 - end
40 -
41 - % Generate code and m
42 - codegen -config codert
43 - disp('Completed gener
44 - end
45 -

```

Busy

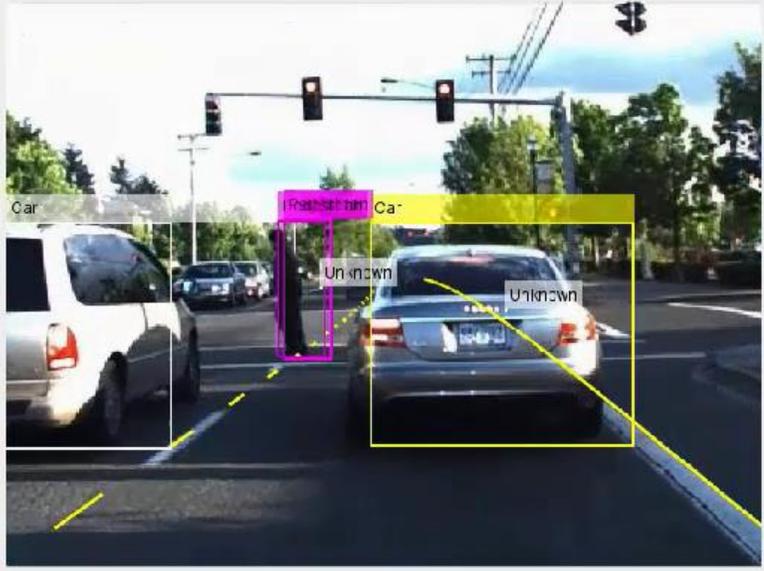
Forward Collision Warning using SensorFusion by Live Data Stream

START STOP 178

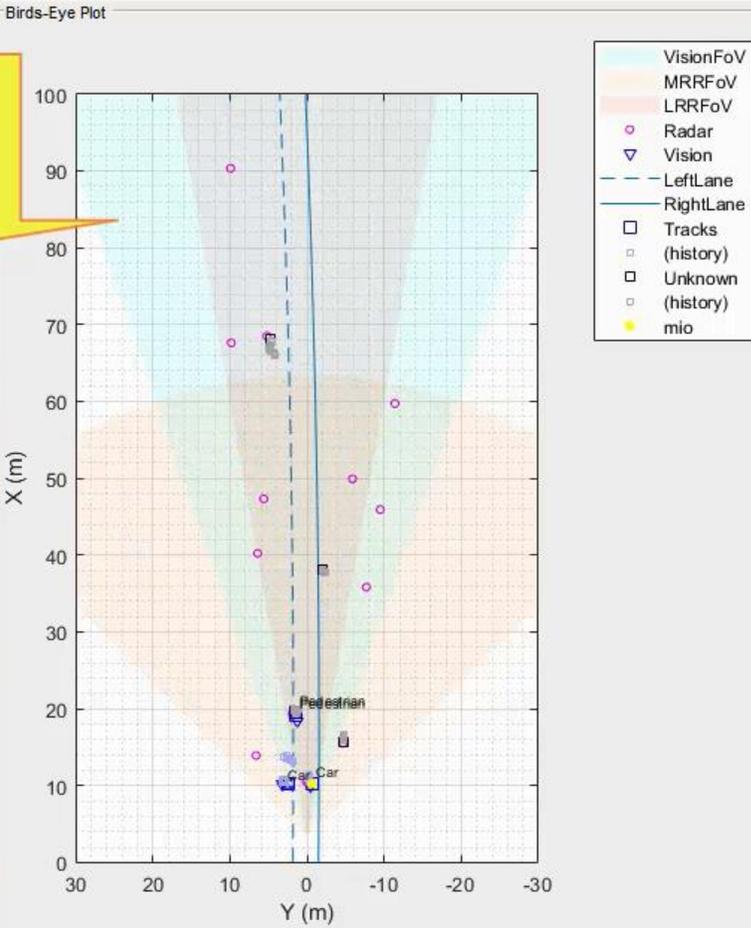
>> Initializing App ...
[CAN FD] Channel is online

Algorithm uses
1's of msec in
software-in-the-loop mode

Algorithm (sil): 1.808 ms

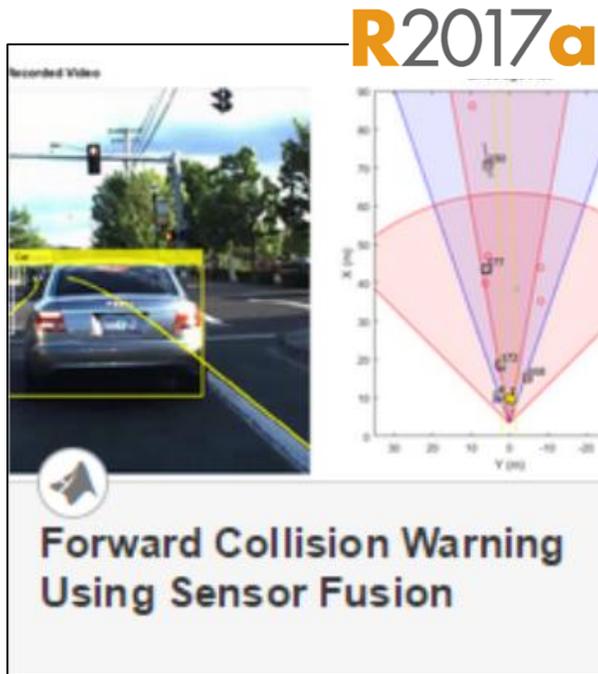


Birds-Eye Plot

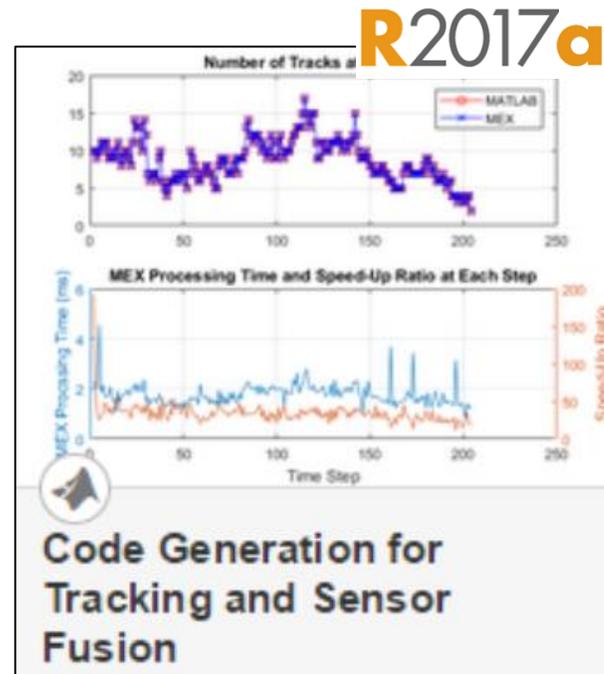


- VisionFoV
- MRRFoV
- LRRFoV
- Radar
- Vision
- LeftLane
- RightLane
- Tracks
- (history)
- Unknown
- (history)
- mio

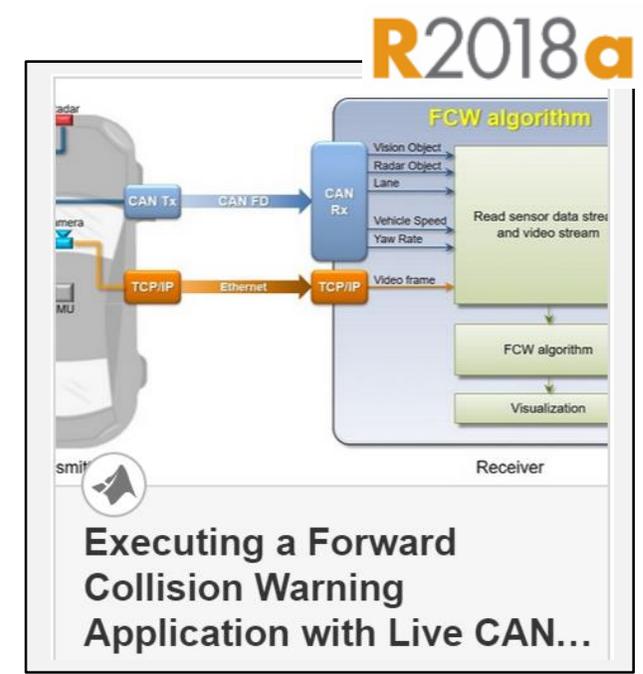
了解更多的传感器融合算法开发案例



- 使用记录的数据设计包含多目标跟踪器的算法
Automated Driving System Toolbox™



- 生成 C/C++ 代码
对包含多目标跟踪器的算法
MATLAB Coder™



- 传输 CAN FD 数据流
在笔记本电脑上验证原型算法

Vehicle Network Toolbox™

使用 MATLAB 和 Simulink 开发自动驾驶算法的案例

深度学习



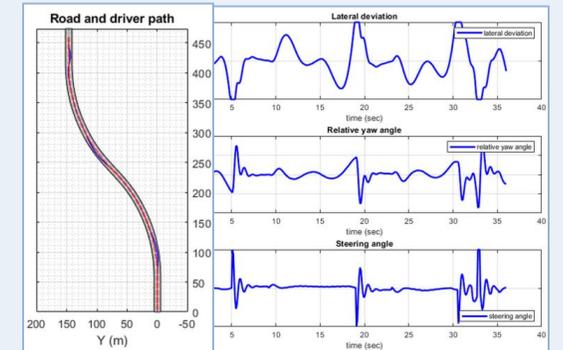
感知
Perception

对实时数据的
传感器融合



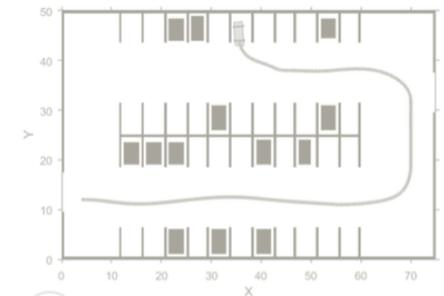
控制
Control

传感器建模 &
模型预测控制



规划
Planning

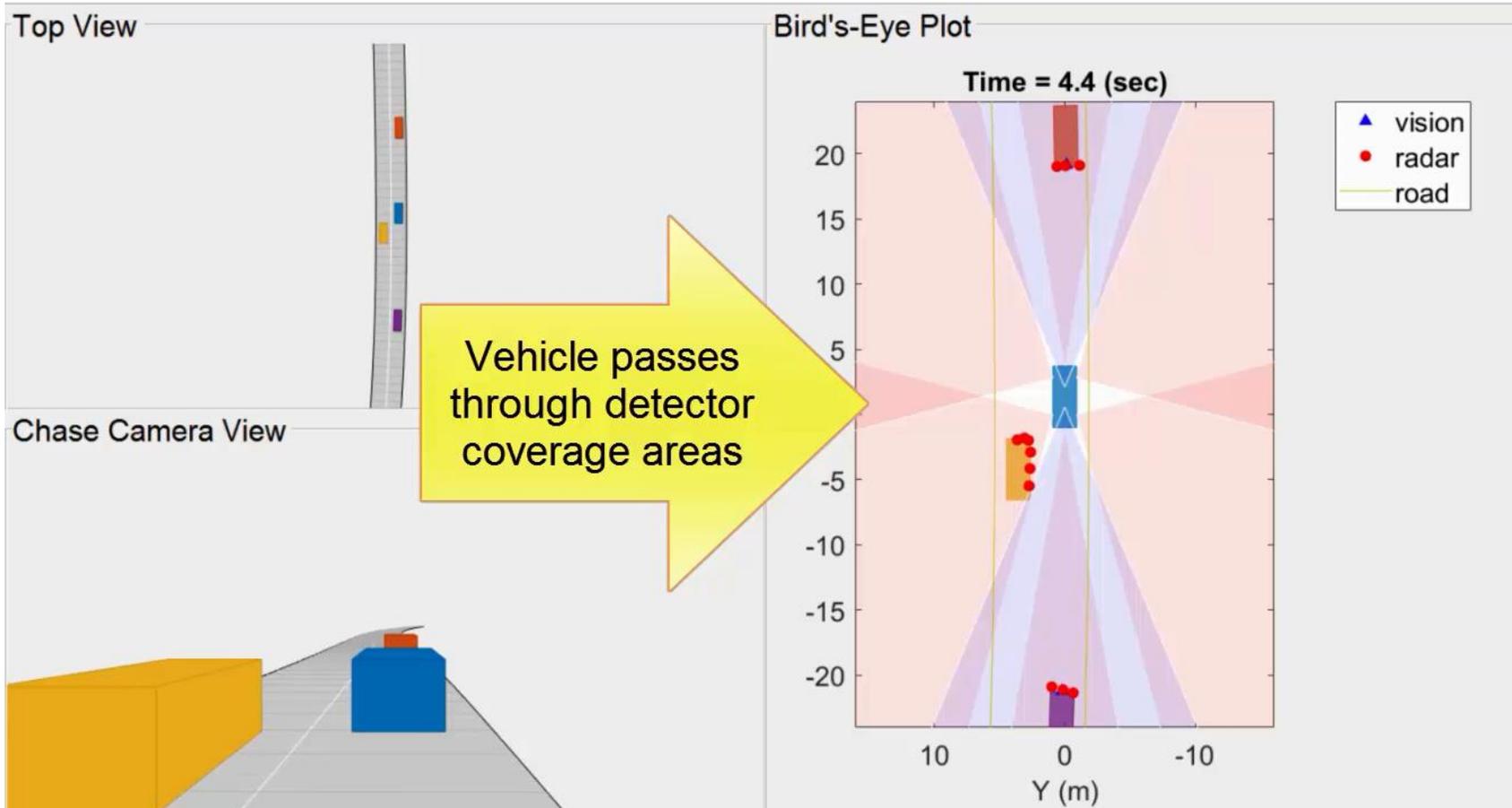
路径规划



自动驾驶系统工具箱 Automated Driving System Toolbox

人工构建场景，测试传感器融合算法

R2017a



Videos and Webinars

Play scenario with sensor models

```

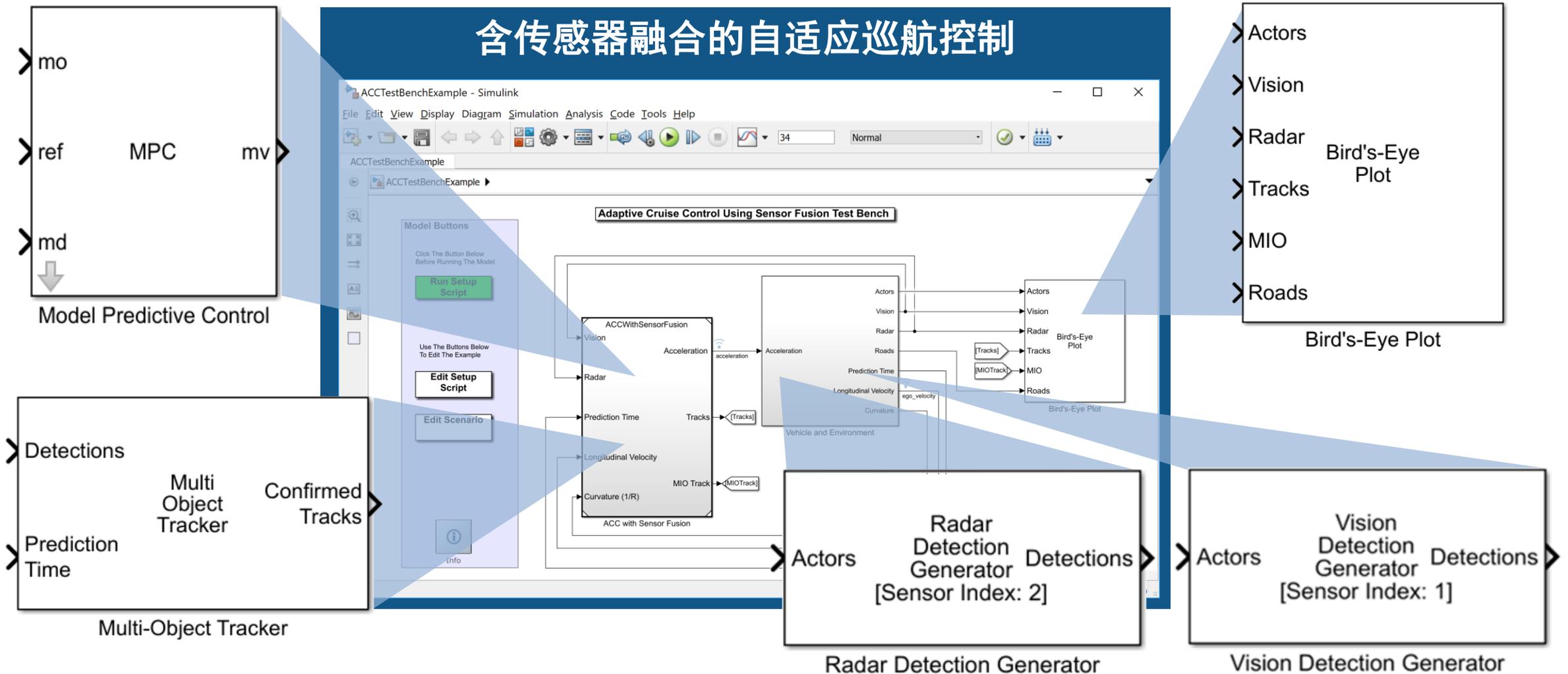
while advance(s)
    % get detections in ego vehicle coordinate
    det = sensor(targetPoses(egoCar), ...
                s.SimulationTime);
    % update plot area
    if isempty(det)
        clearData(detPlot);
    else % update measurements to position
        pos = callfun(@(d,d.Measurement(1:2) ...
                    det, 'UnitForOutput', false);
        vel = callfun(@(d,d.Measurement(4:5) ...
                    det, 'UnitForOutput', false);
        plotDetection(detPlot, ...
                    cell2mat(pos'), cell2mat(vel));
    end
    [p, y, l, w, oo, c] = targetOutline;
    plotOutline(truthPlot, p, y, l, w, ...
                'color', oo, 'strokeWidth', c);
end
    
```

26:10

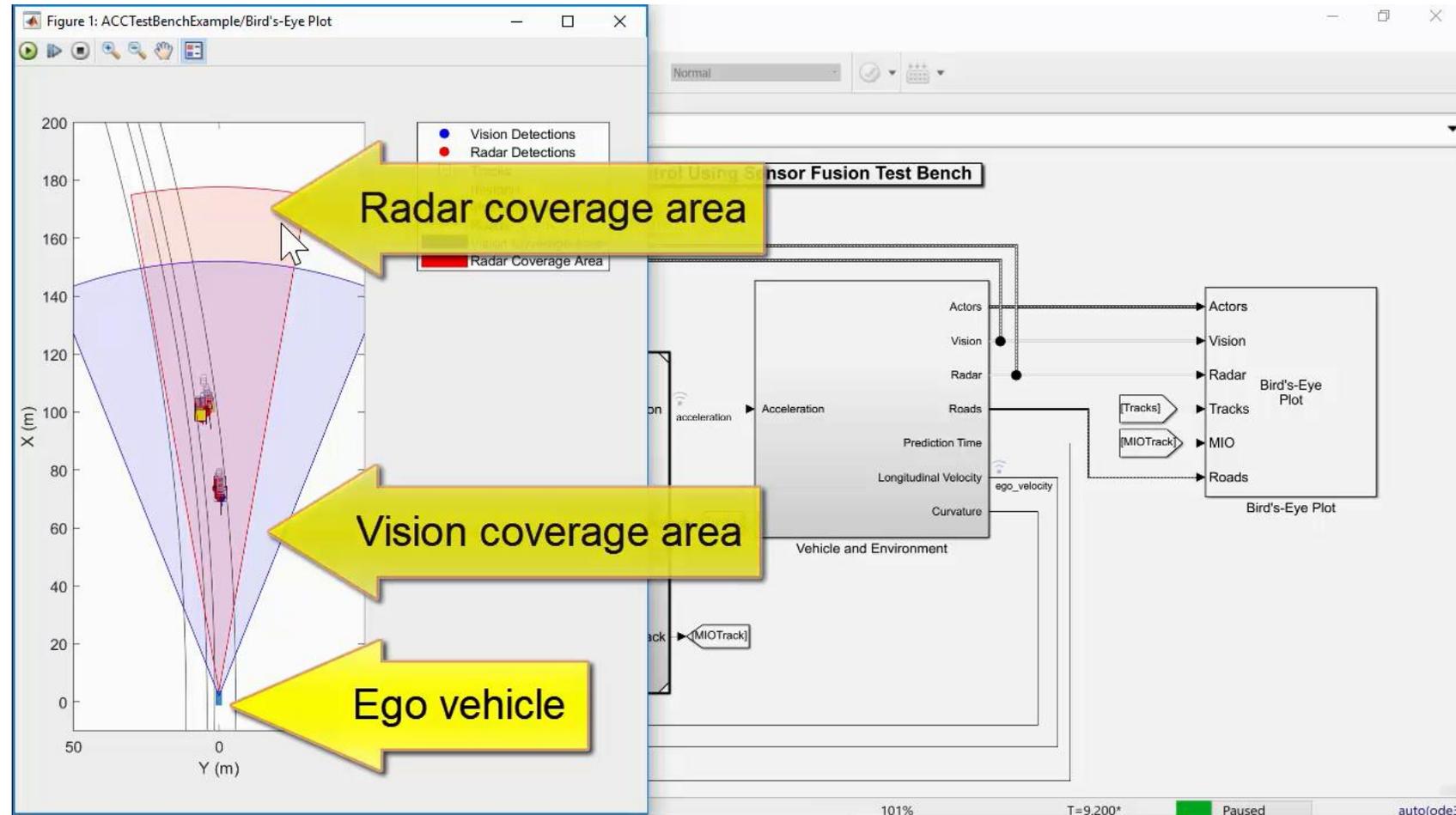
Introduction to Automated Driving System Toolbox

使用雷达和视觉检测器、传感器融合、模型预测控制等模块，进行系统的闭环仿真

R2017b

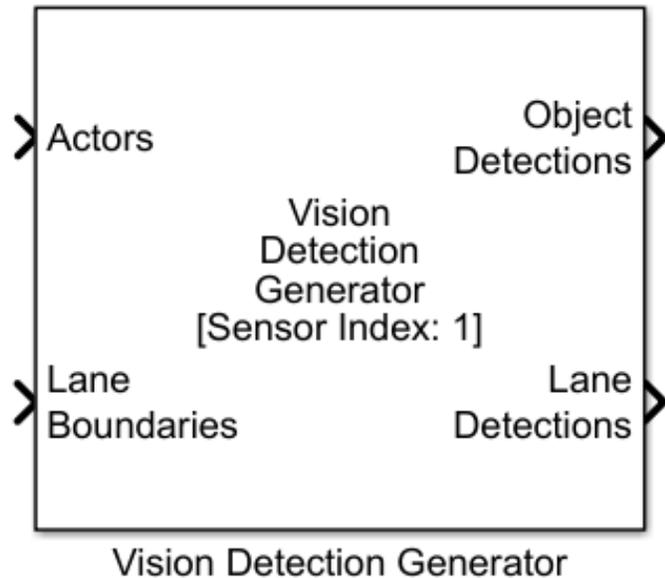


模拟传感器检测结果，测试传感器融合与模型预测控制算法



使用视觉检测器模拟车道线检测

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 Intrinsic

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): Lanes and objects

Sensor Extrinsic

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

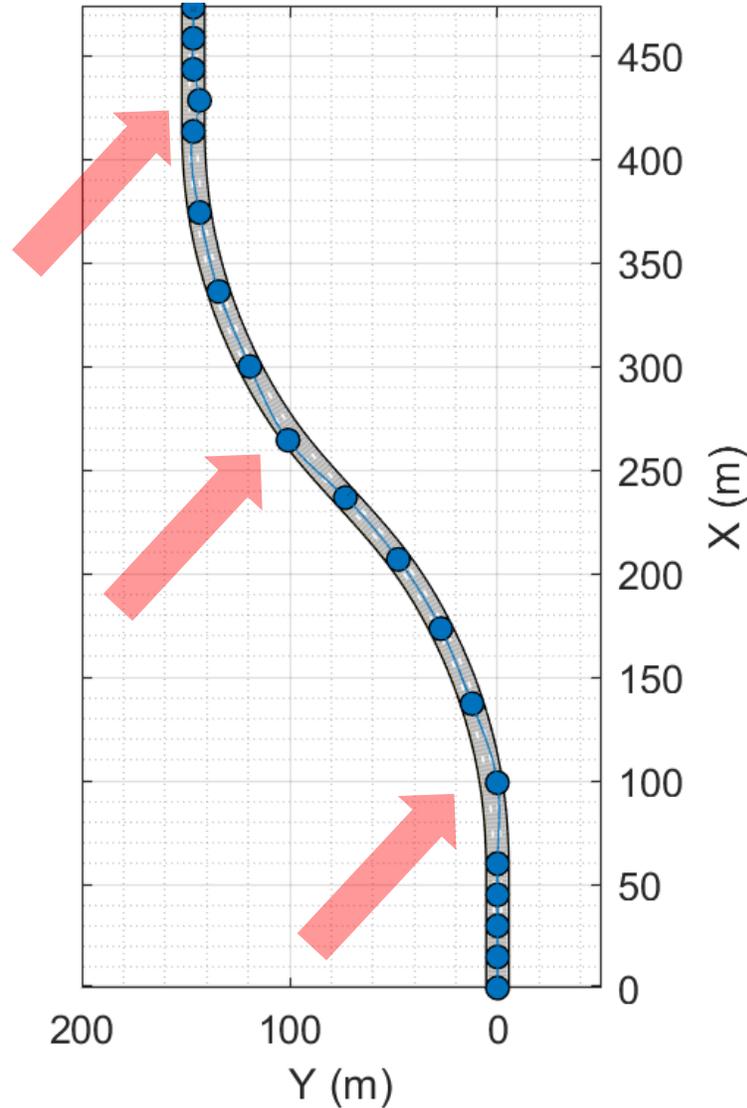
Pitch angle of sensor mounted on ego vehicle (deg): 1

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

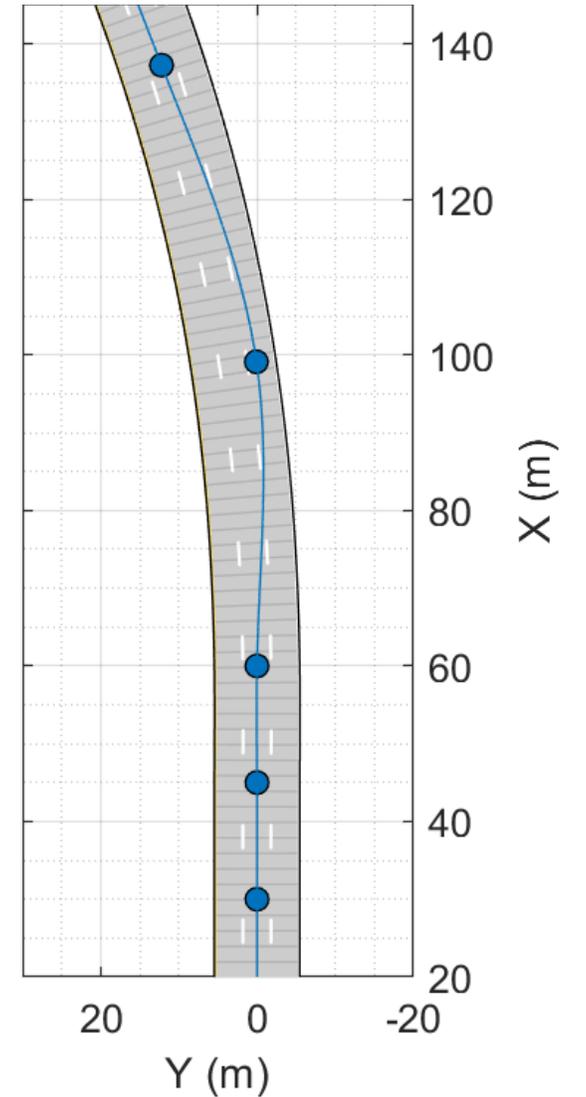
使用驾驶场景工具（drivingScenario）创建双曲线道路

- 模拟驾驶路径
- 驾驶员在曲率变化处出现分心

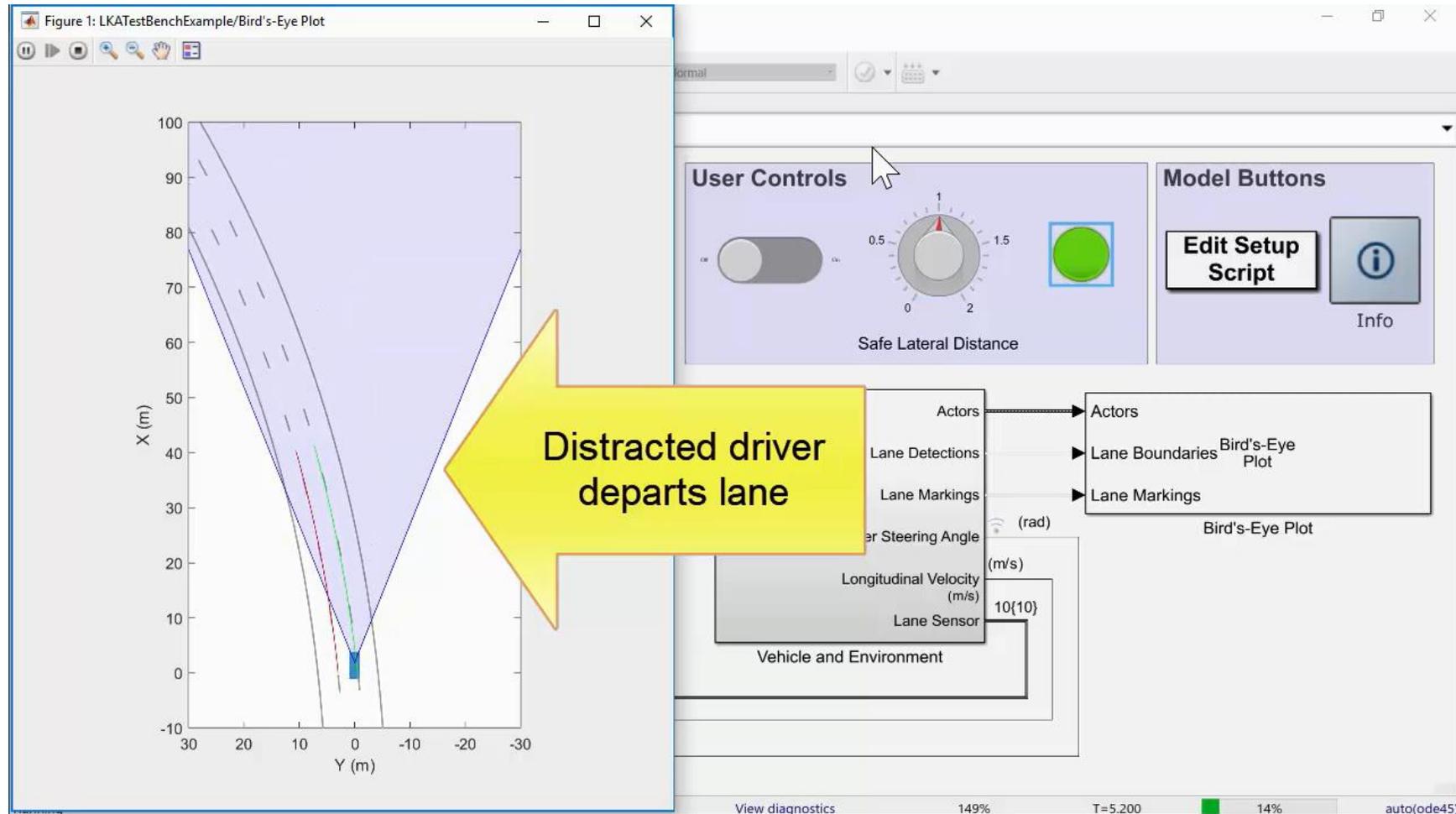
道路与驾驶路径



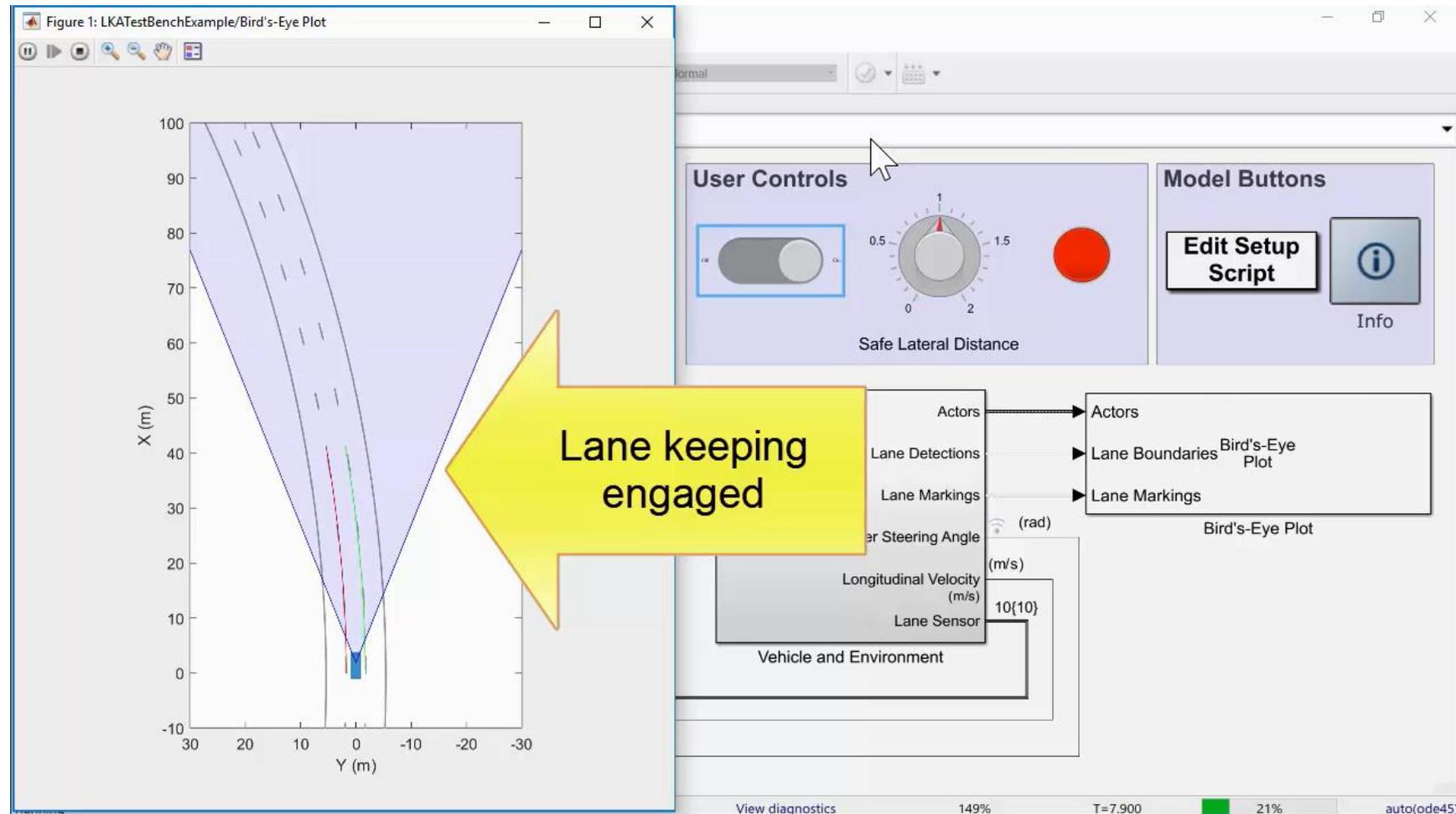
驾驶员在道路曲率变化处出现分心



分心驾驶员的仿真结果

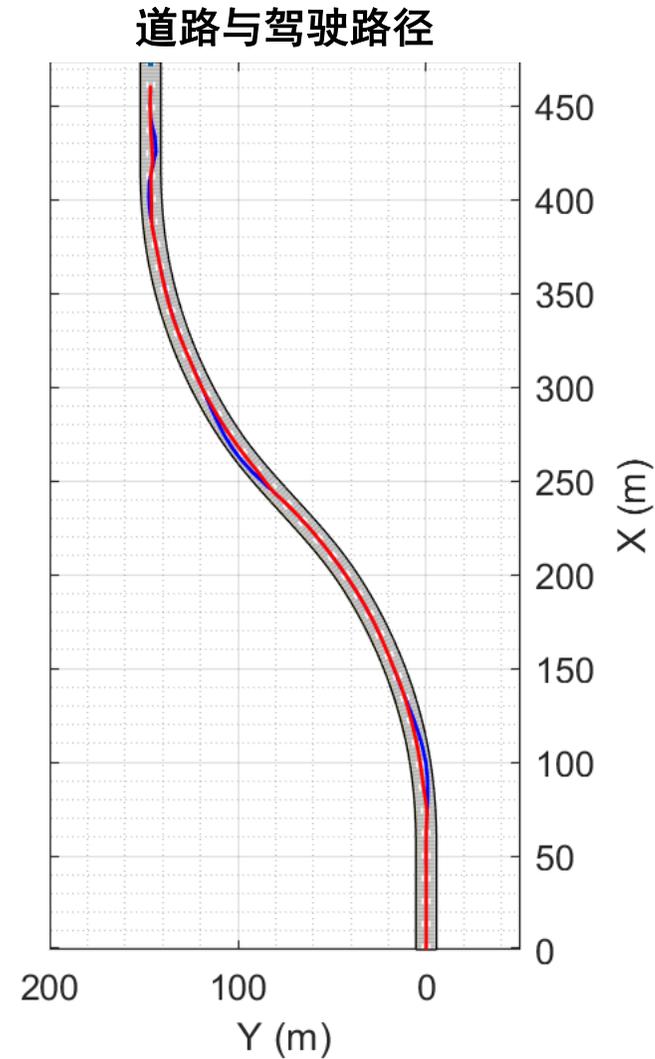


车道保持辅助介入后的仿真结果

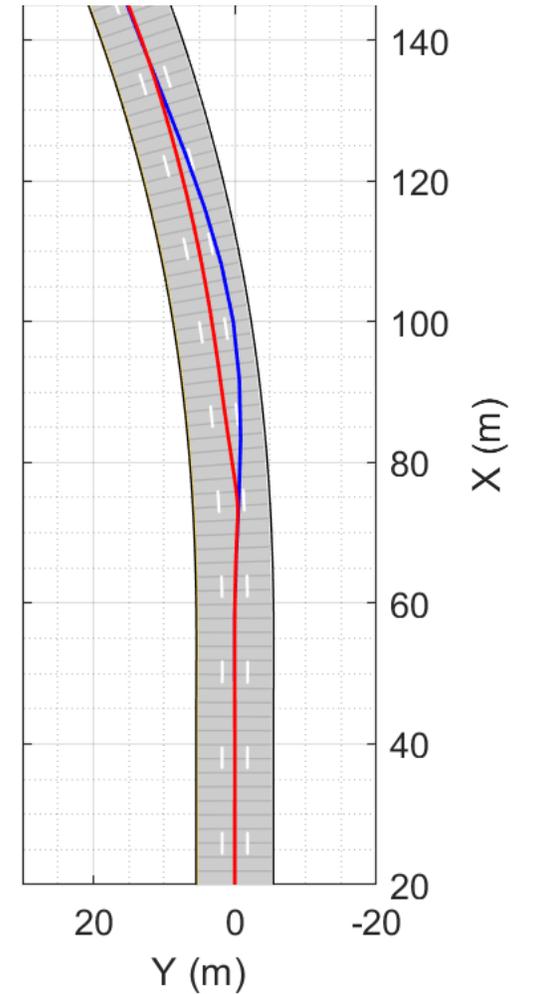


对比分心与辅助后的结果

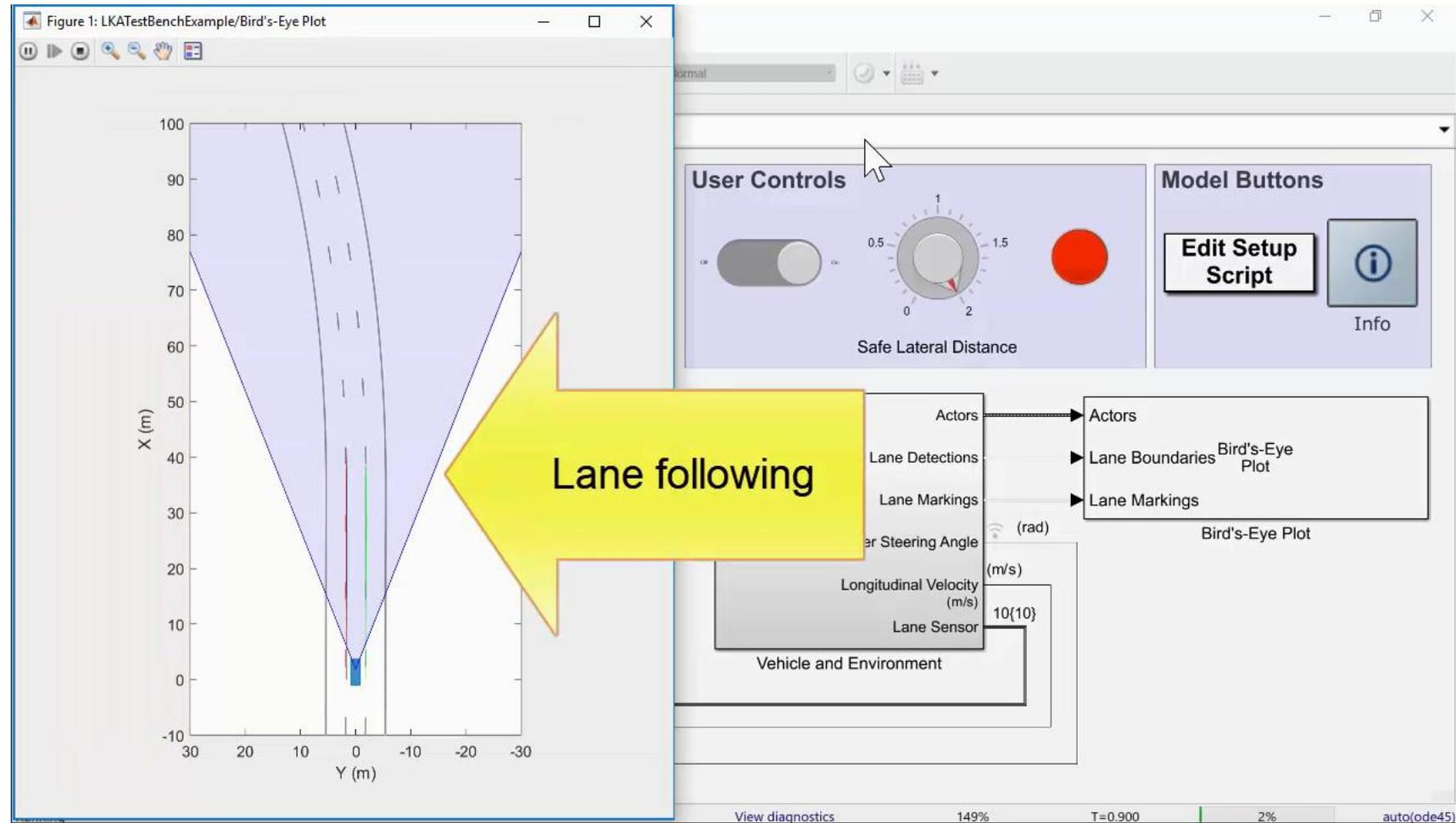
- 检测车道偏离
- 在驾驶员分心时保持车道



驾驶员在道路曲率变化处获得辅助

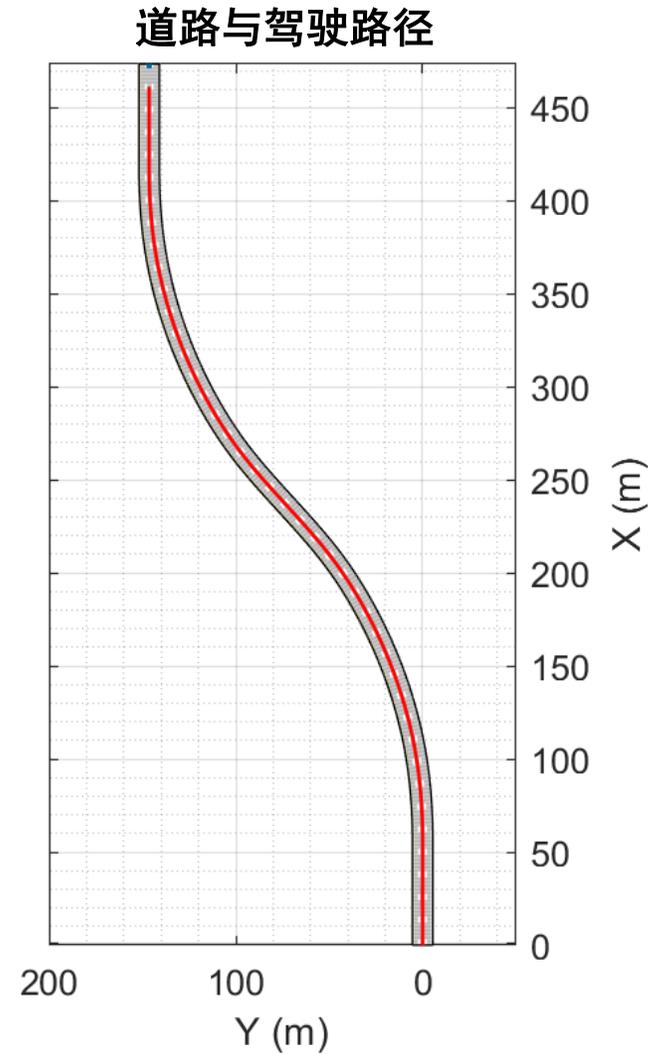


最大化横向安全距离后的车道跟随

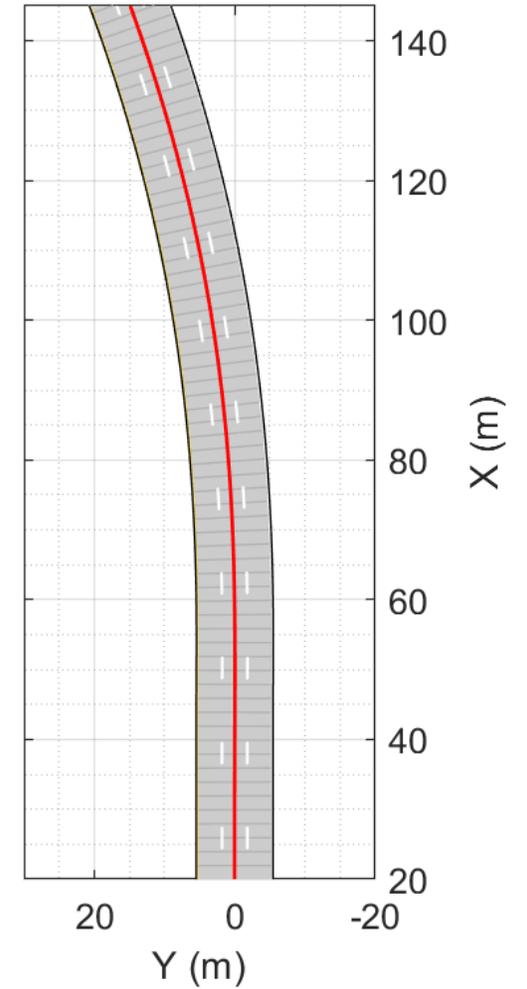


观察车道跟随结果

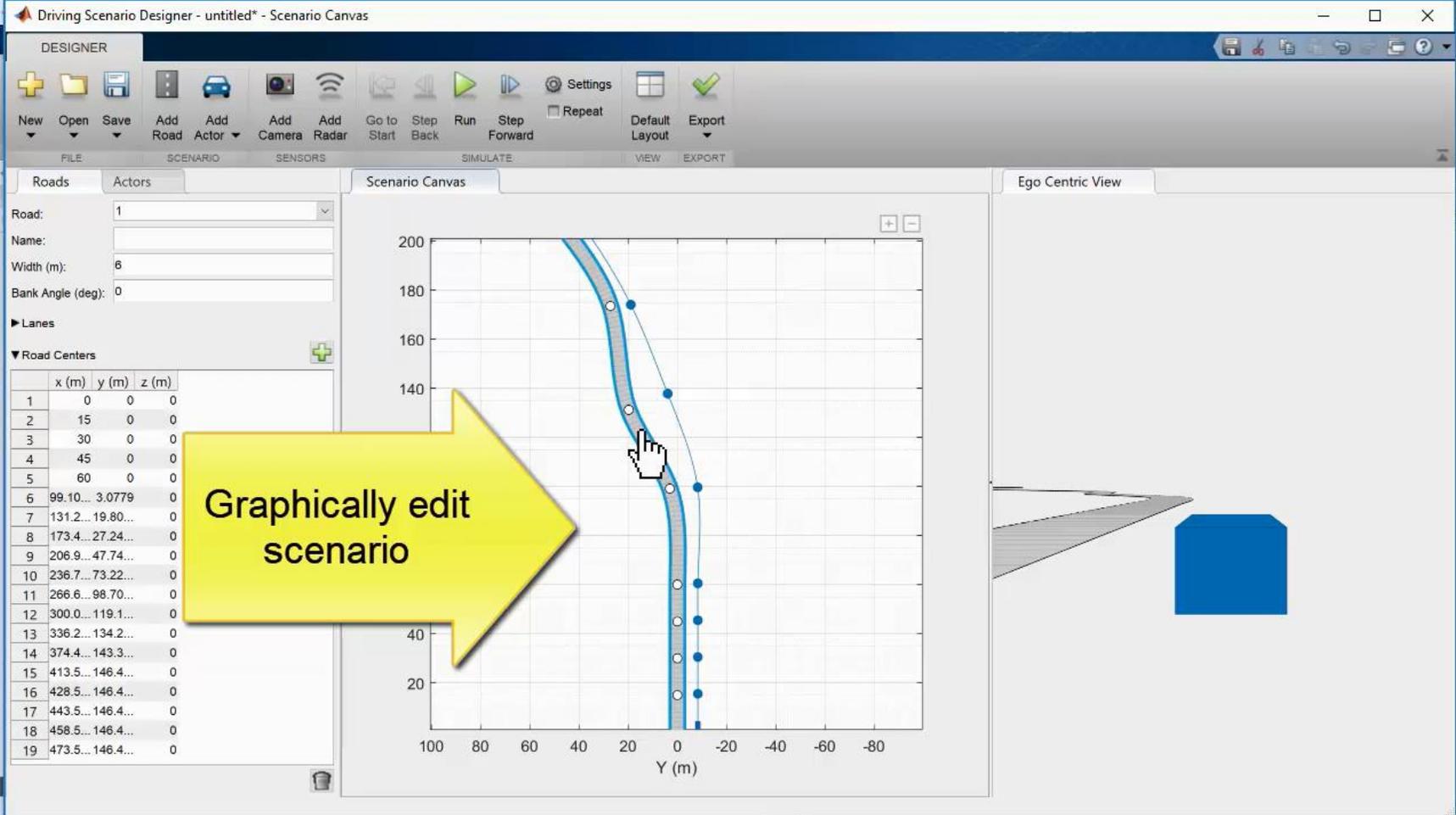
- 车辆始终保持在车道边界线内



驾驶员在道路曲率变化处获得辅助



图形化的驾驶场景设计器 (Driving Scenario Designer)



DESIGNER

FILE SCENARIO SENSORS SIMULATE VIEW EXPORT

New Open Save Add Road Add Actor Add Camera Add Radar Go to Start Step Back Run Step Forward Settings Repeat Default Layout Export

Roads Actors Scenario Canvas Ego Centric View

Road: 1
Name:
Width (m): 6
Bank Angle (deg): 0

Lanes

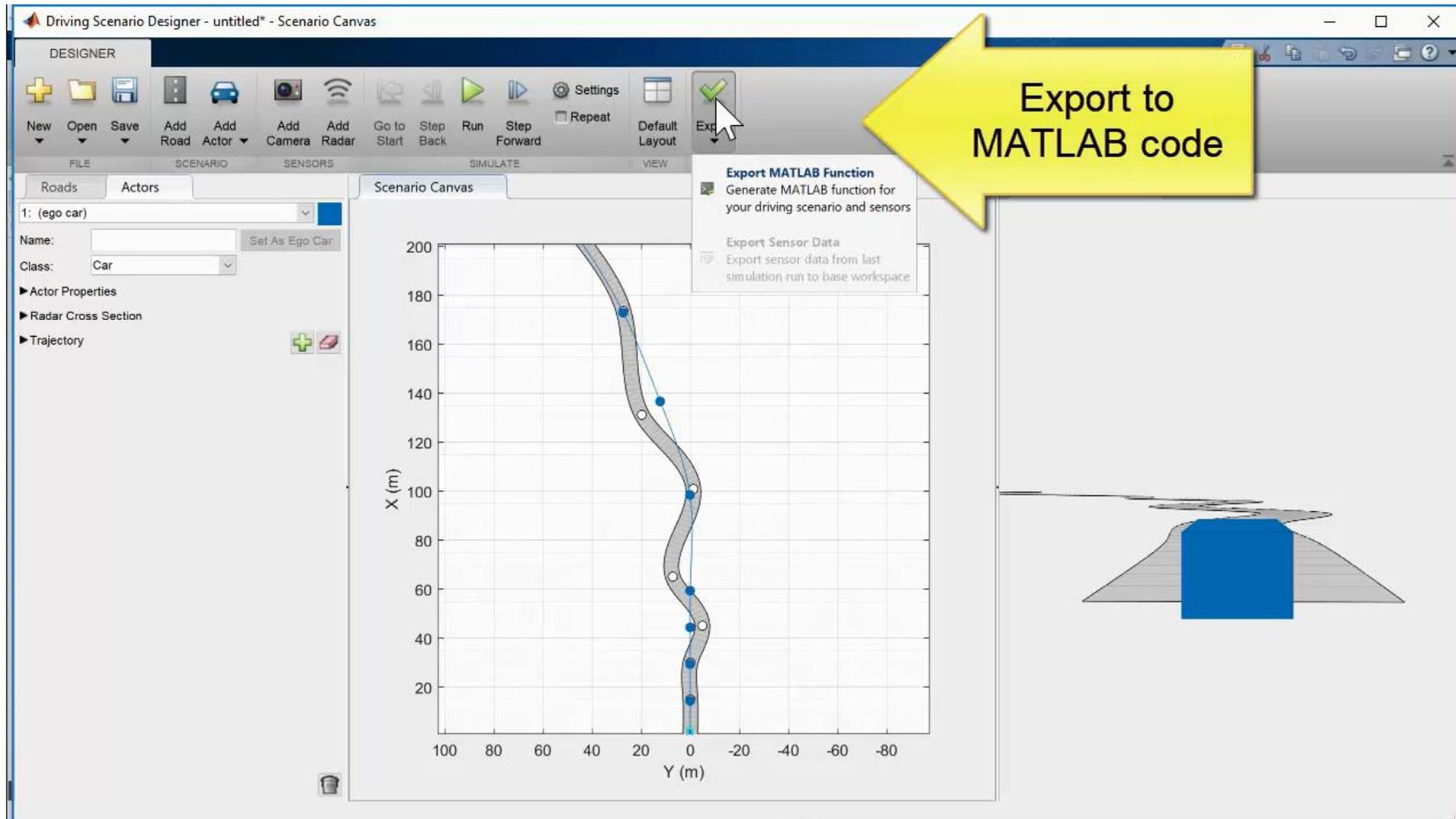
Road Centers

	x (m)	y (m)	z (m)
1	0	0	0
2	15	0	0
3	30	0	0
4	45	0	0
5	60	0	0
6	99.10...	3.0779	0
7	131.2...	19.80...	0
8	173.4...	27.24...	0
9	206.9...	47.74...	0
10	236.7...	73.22...	0
11	266.6...	98.70...	0
12	300.0...	119.1...	0
13	336.2...	134.2...	0
14	374.4...	143.3...	0
15	413.5...	146.4...	0
16	428.5...	146.4...	0
17	443.5...	146.4...	0
18	458.5...	146.4...	0
19	473.5...	146.4...	0

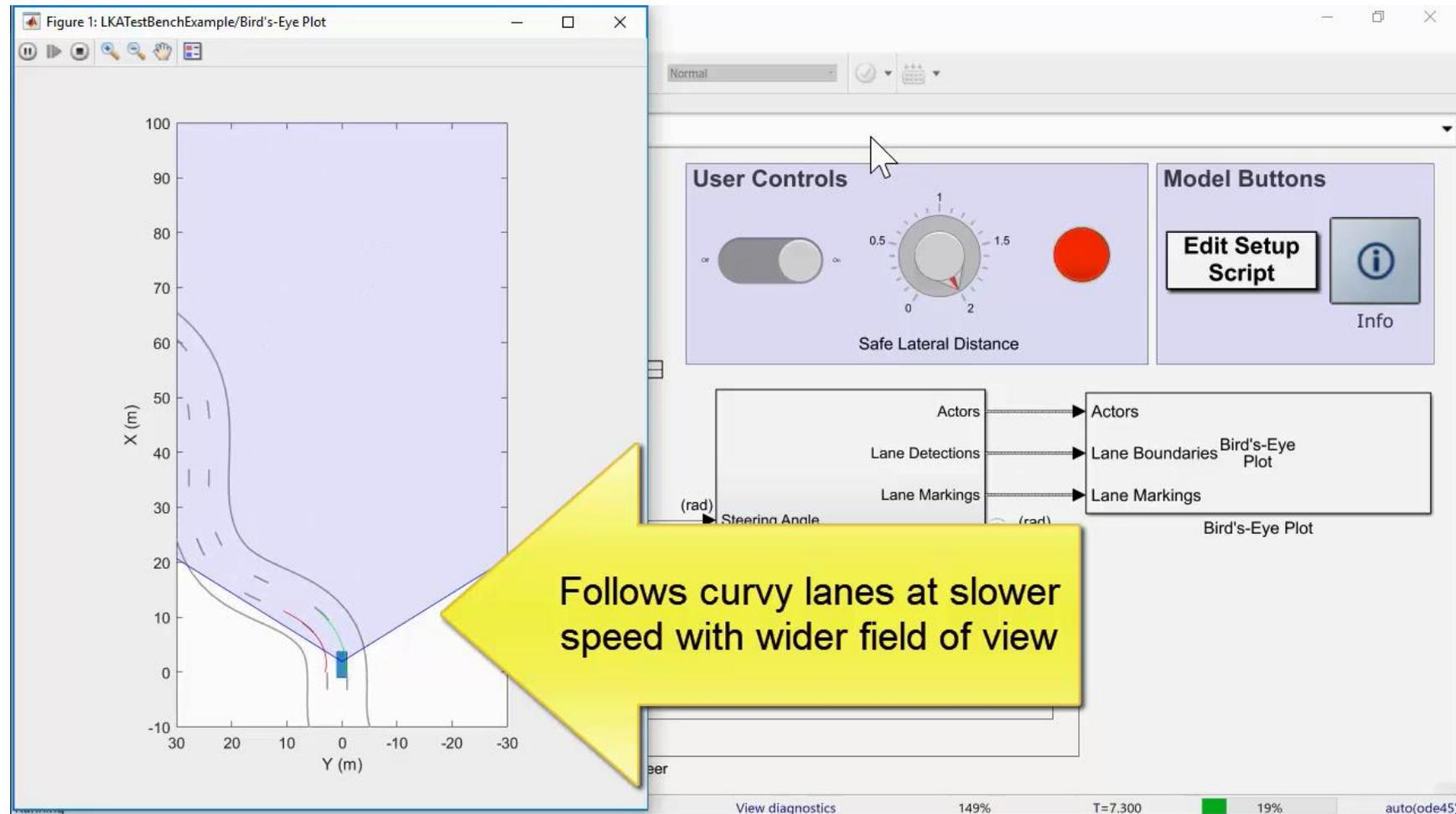
Graphically edit scenario

Y (m)

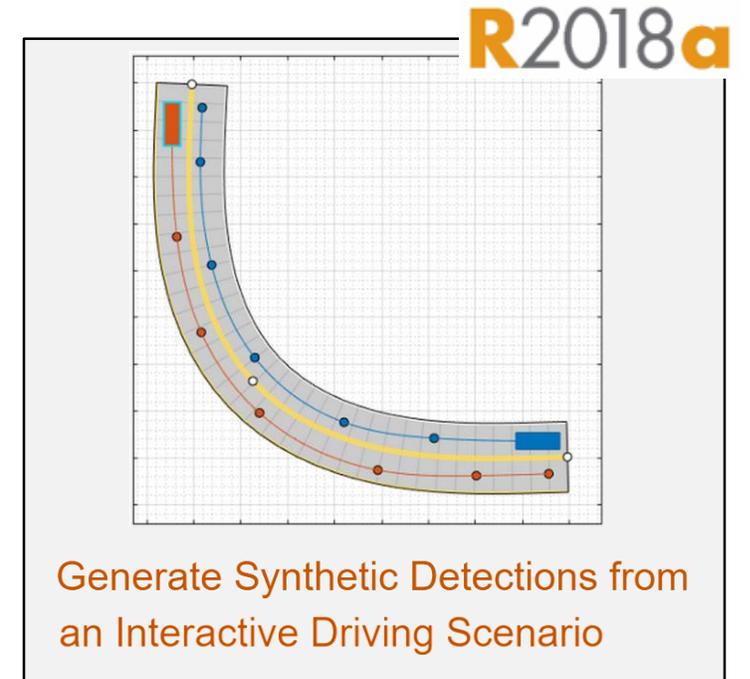
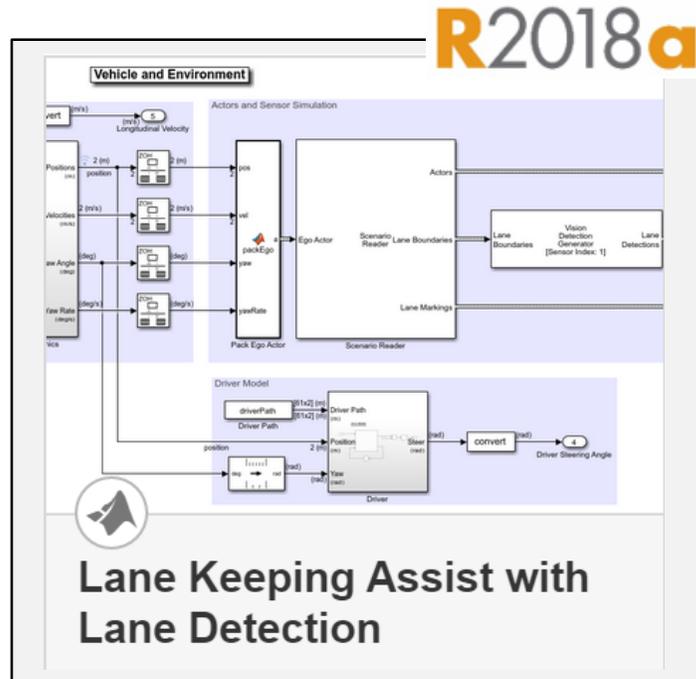
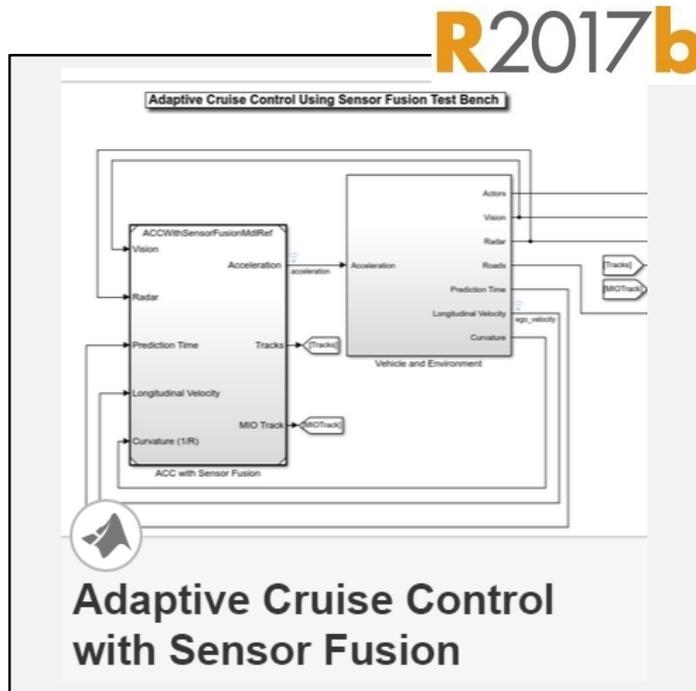
导出用于生成场景的 MATLAB 代码



进一步调整参数，适应高曲率道路



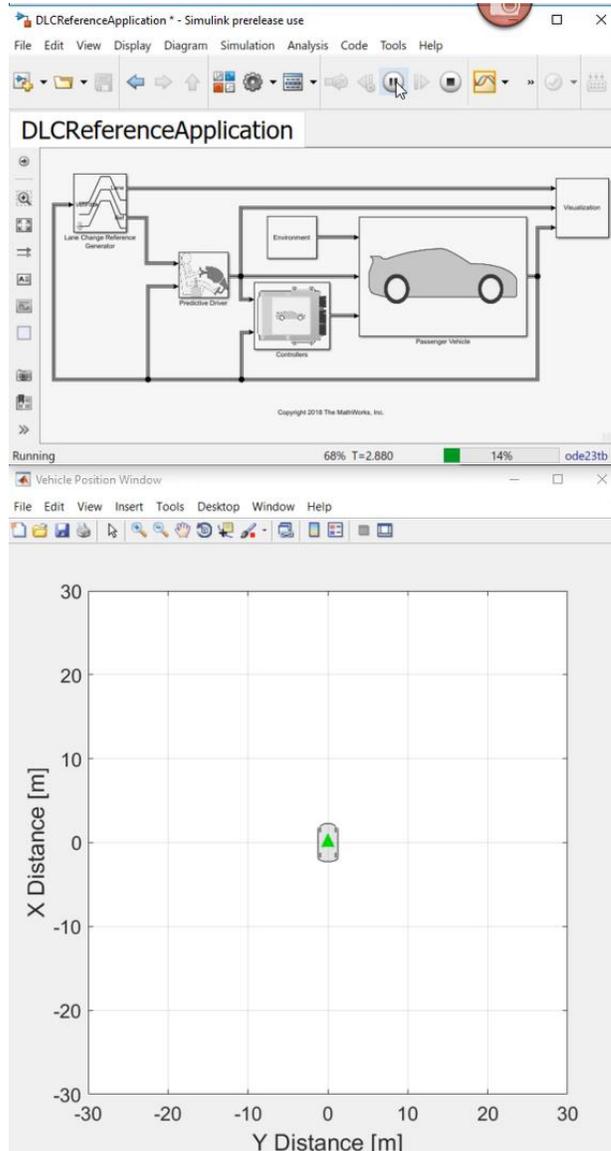
通过这些案例了解如何模拟传感器和驾驶场景，用于开发控制算法



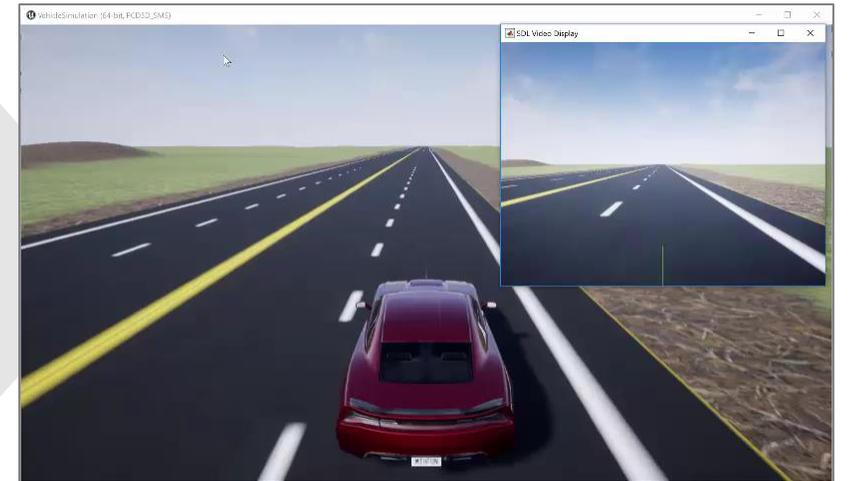
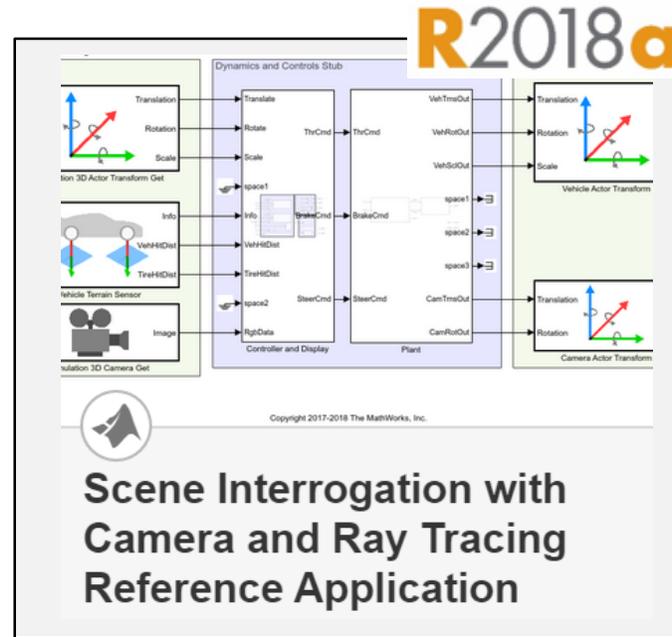
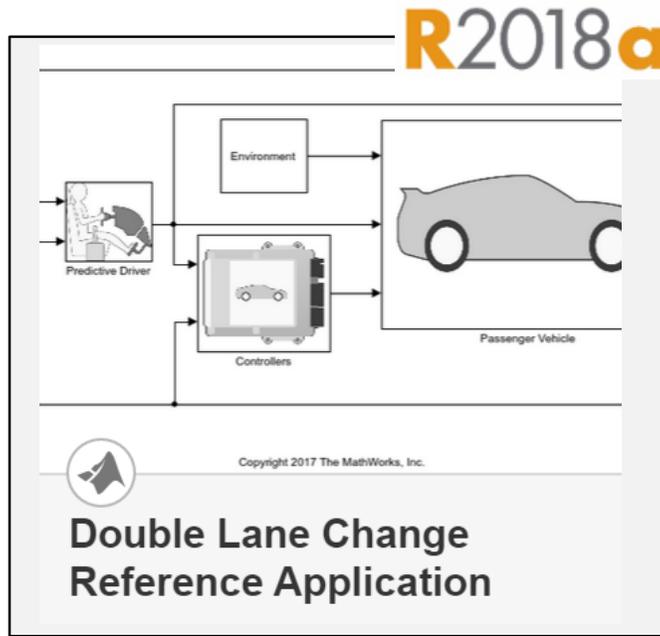
- 仿真与生成 C/C++ 代码
模型预测控制与传感器融合算法
- 仿真与生成 C/C++ 代码
模型预测控制与含车道线检测的视觉传感器
- 编辑道路、交通参与者及传感器
使用驾驶场景设计器
drivingScenarioDesigner

车辆动力学模块库 Vehicle Dynamics Blockset

R2018a



通过这些案例了解如何对车辆动力学建模，用于开发控制算法



- **车辆动力学仿真**
用于闭环控制设计
Vehicle Dynamics Blockset™

- **与 Unreal 引擎协同仿真**
设置交通参与者位置及获取
摄像机图像
Vehicle Dynamics Blockset™

使用 MATLAB 和 Simulink 开发自动驾驶算法的案例

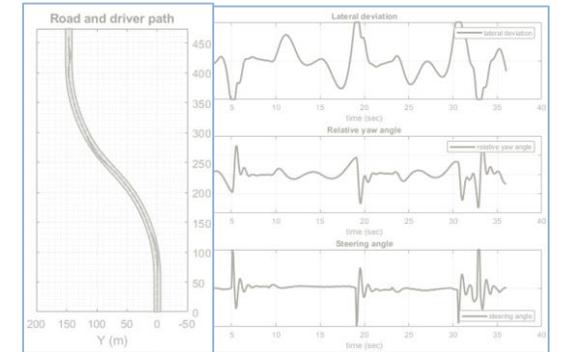
深度学习



感知
Perception

控制
Control

传感器建模 &
模型预测控制

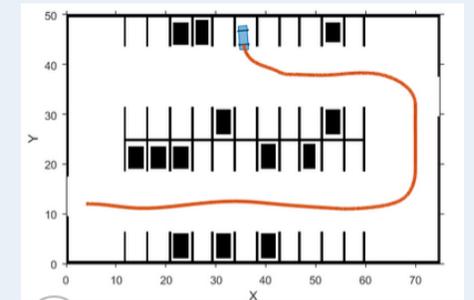


对实时数据的
传感器融合

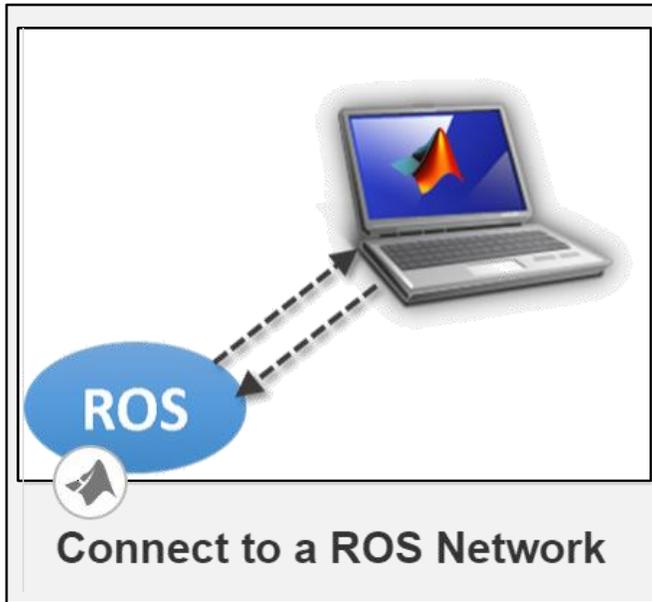


规划
Planning

路径规划



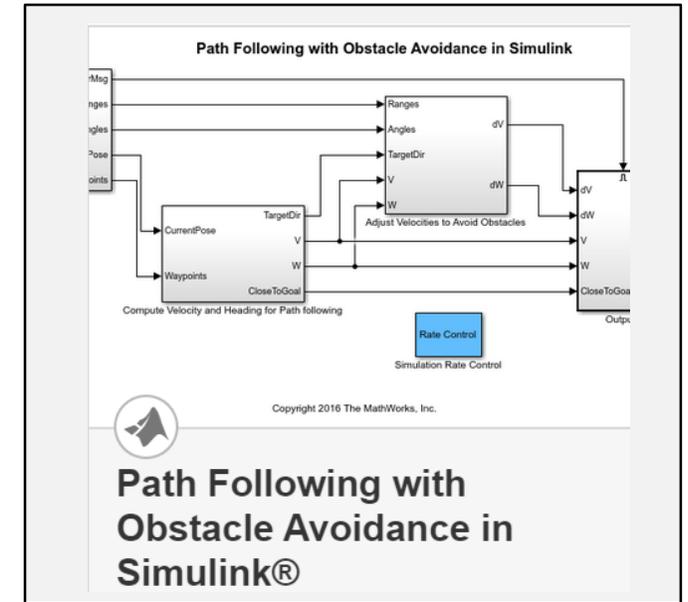
机器人系统工具箱 Robotics System Toolbox 提供与 ROS 生态系统的连接性



- 使用 ROS 通信
与外部的 ROS 组件集成

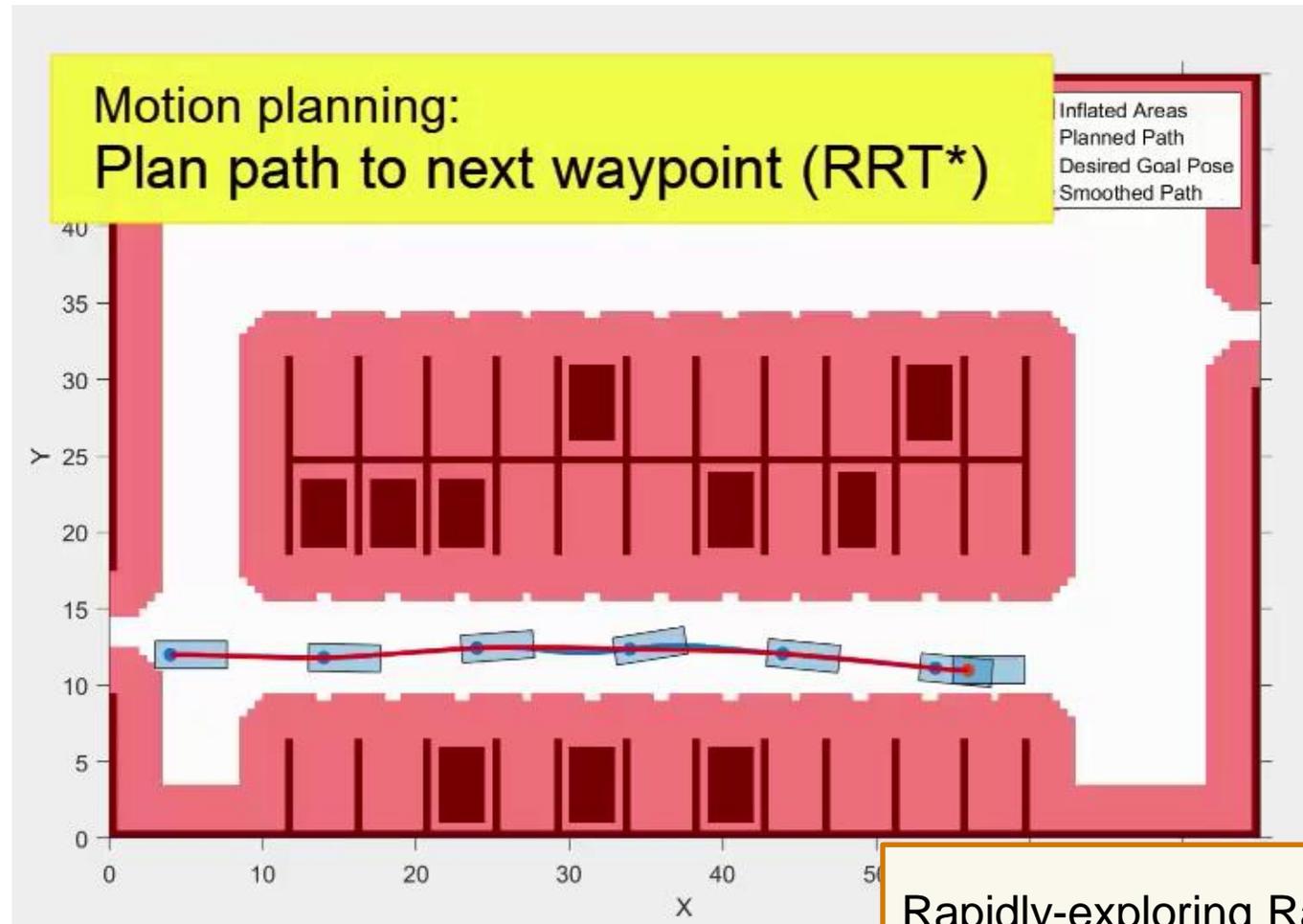


- 与 Gazebo 通信
机器人系统的可视化仿真



- 路径跟随
基于 ROS 的差分驱动机器人仿真

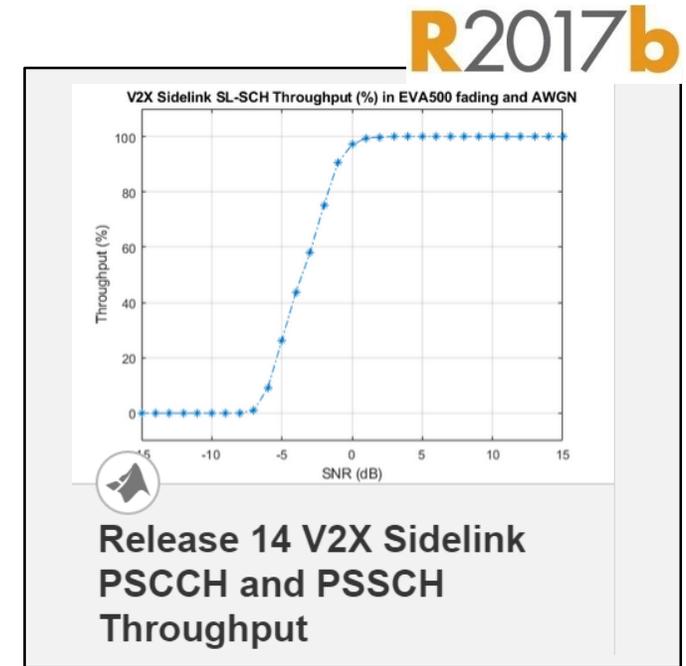
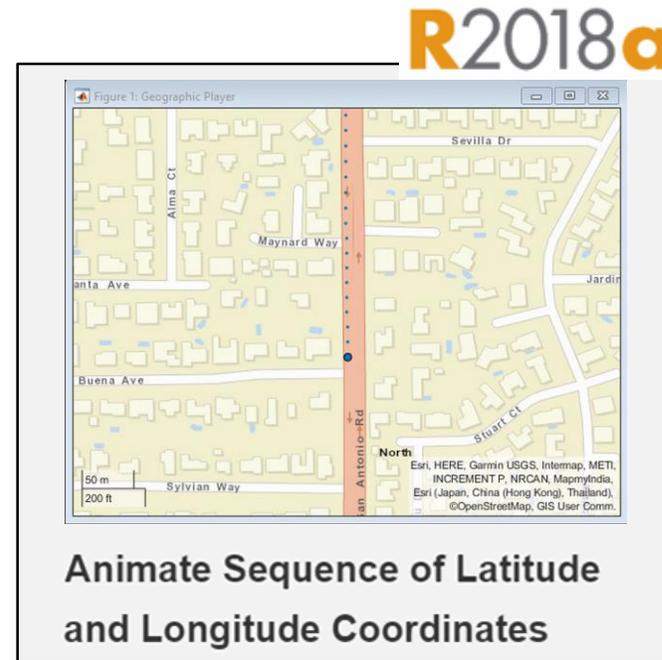
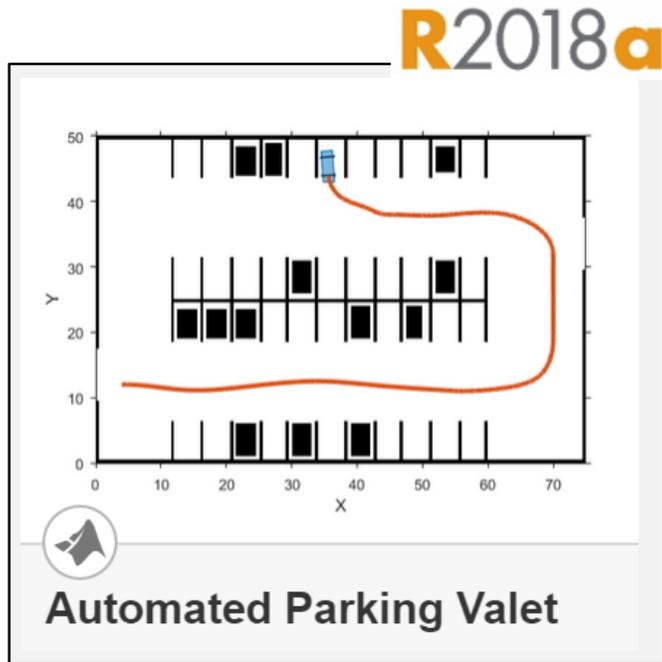
车辆路径规划算法的设计和仿真



R2018a

Rapidly-exploring Random Tree (RRT*)

通过这些案例了解如何开发路径规划算法



- 规划车辆路径
预定义地图的应用
Automated Driving
System Toolbox™

- 绘制地图瓦片
使用 World Street Map (ESRI)
Automated Driving
System Toolbox™

- 仿真 V2X 通信
评估信道吞吐量
LTE System Toolbox™

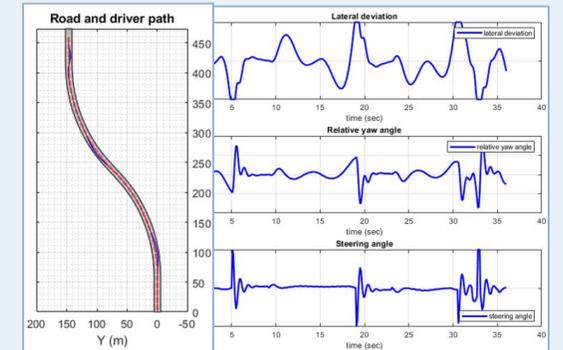
使用 MATLAB 和 Simulink 开发自动驾驶算法的案例

深度学习



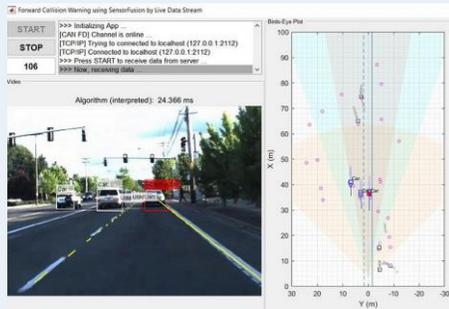
感知
Perception

传感器建模 &
模型预测控制



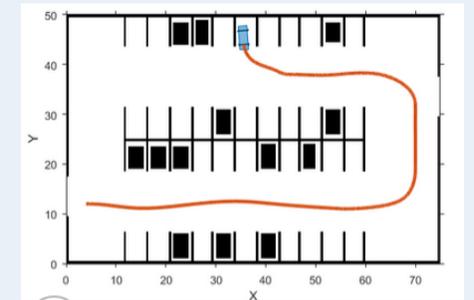
控制
Control

对实时数据的
传感器融合



规划
Planning

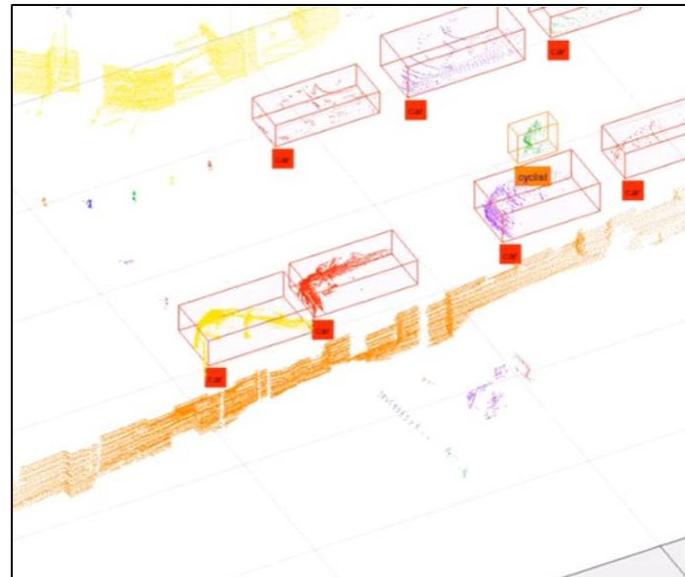
路径规划



MathWorks 能够帮助您自定义 MATLAB 和 Simulink, 建立专门的自动驾驶开发工具



- **基于 Web 的真实值标注**
- 与 Caterpillar 合作的咨询项目
- [2017年 MathWorks 北美汽车年会视频](#)



- **激光雷达真实值标注**
- 与 Autoliv 联合发表论文 (SAE 论文 2018-01-0043)



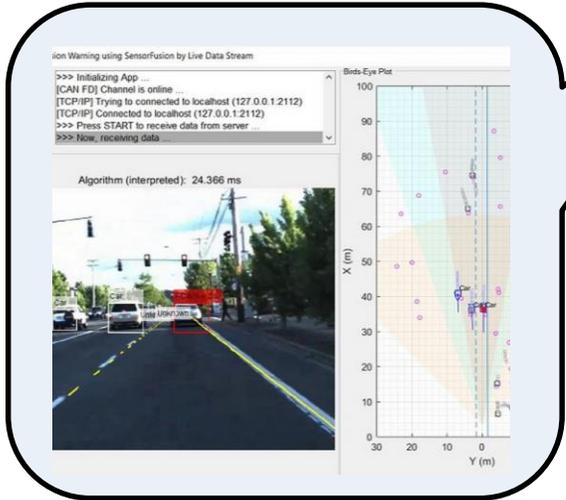
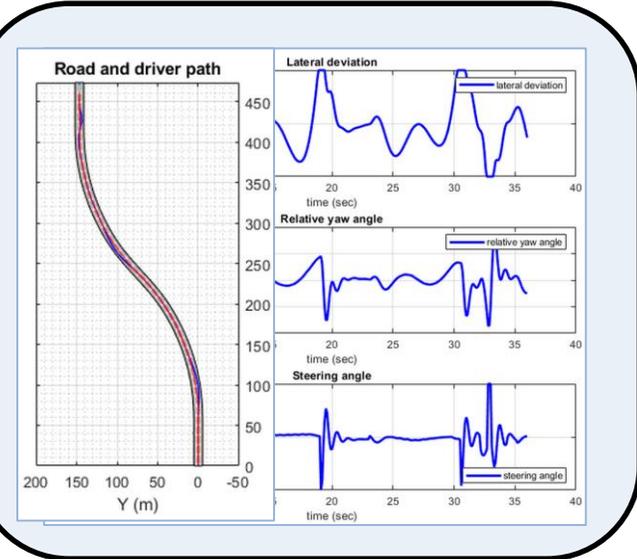
- **用于 Unreal 引擎的激光雷达传感器模型**
- 与 Ford 联合发表论文 (SAE 论文 2017-01-0107)

您可以使用 MATLAB 和 Simulink 开发自动驾驶算法 ...



感知
Perception

控制
Control



规划
Planning



... 同时获得完善的设计验证工具支持

R2017b

Polyspace

```

typedef int int32_t;

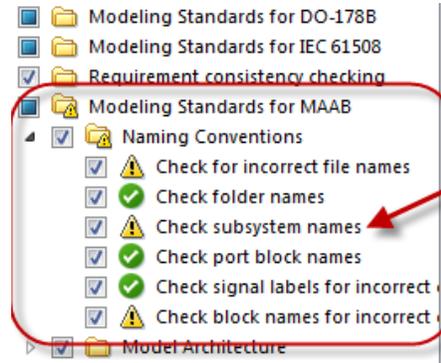
int32_t test1(void)
{
    char *data;
    data = (char *)malloc(100*sizeof(char));
    memset(data, 'A', 100-1);

    return(0);
}

int32_t test2(void)

```

Simulink Verification & Validation

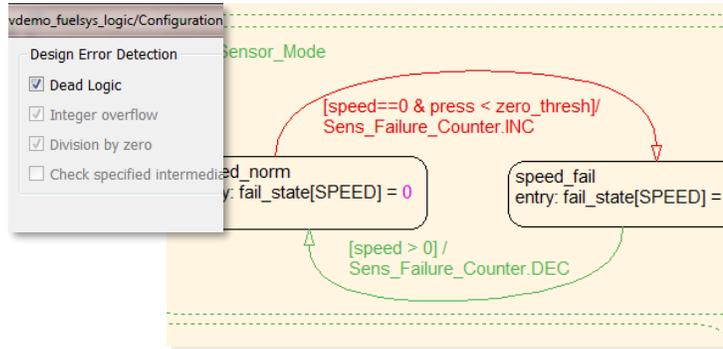


Simulink Requirements

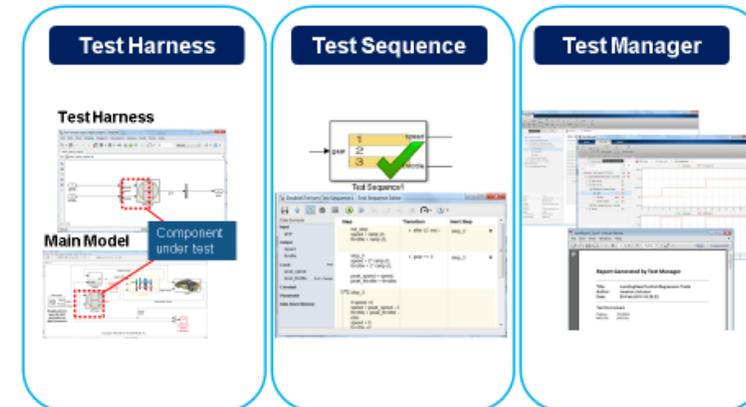
Simulink Coverage

Simulink Check

Simulink Design Verifier



Simulink Test



Q&A