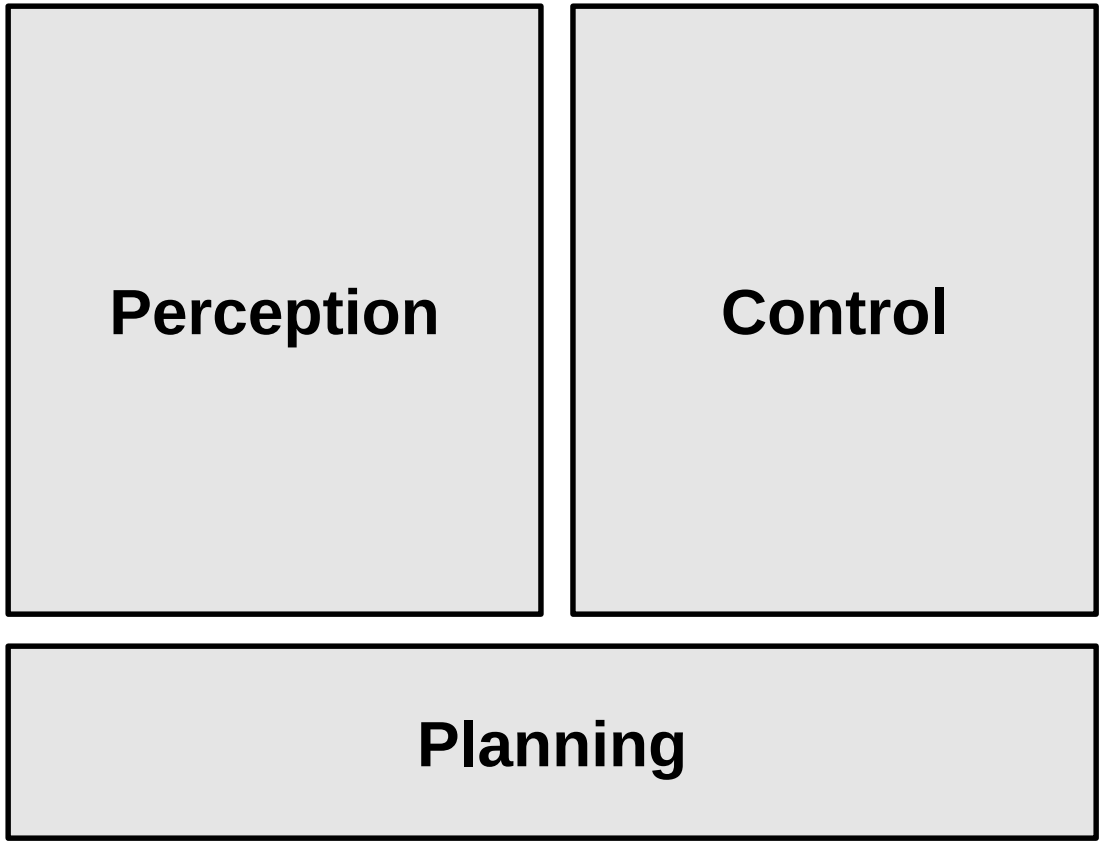
MATLAB EXPO 2018

Sensor Fusion and Tracking

for Automated Driving

Abhijit Bhattacharjee
Senior Application Engineer
MathWorks

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



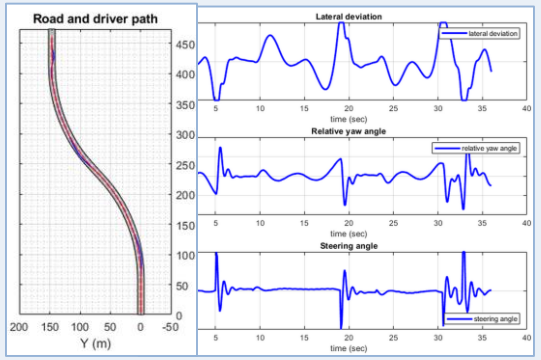
Examples of how you can use MATLAB and Simulink to develop automated driving algorithms

Deep learning

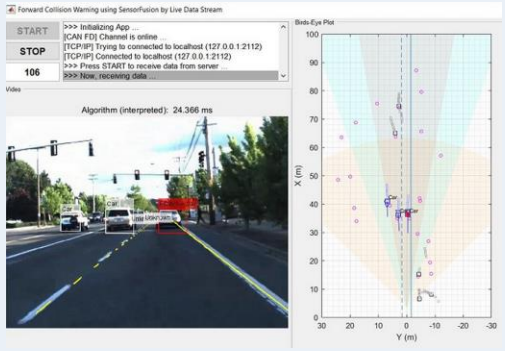
Perception

Control

Sensor models & model predictive control

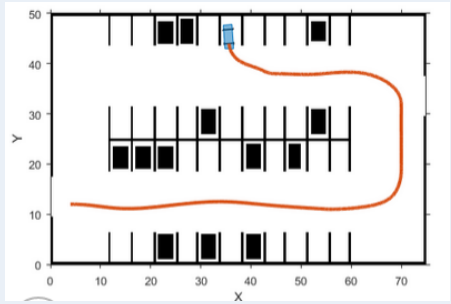


Sensor fusion



Planning

Path planning

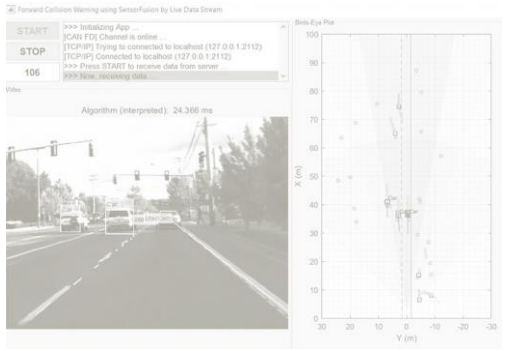


How can you use MATLAB and Simulink to develop perception algorithms?

Deep learning

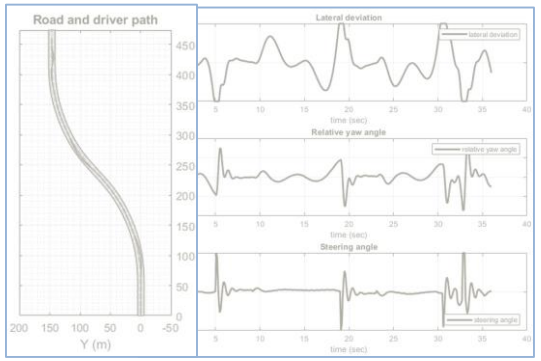
Perception

Sensor fusion



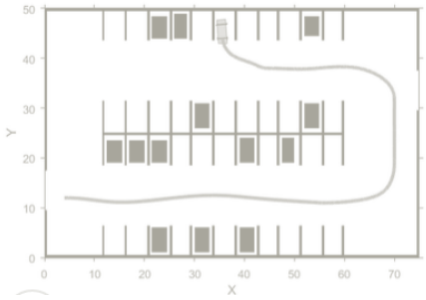
Control

Sensor models & model predictive control



Planning

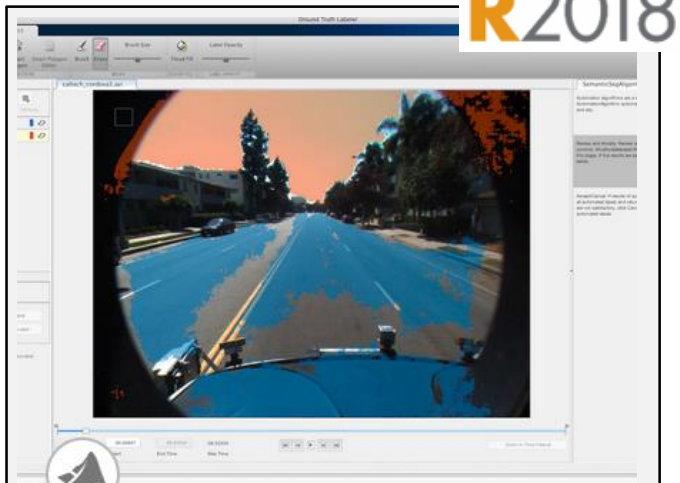
Path planning



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™

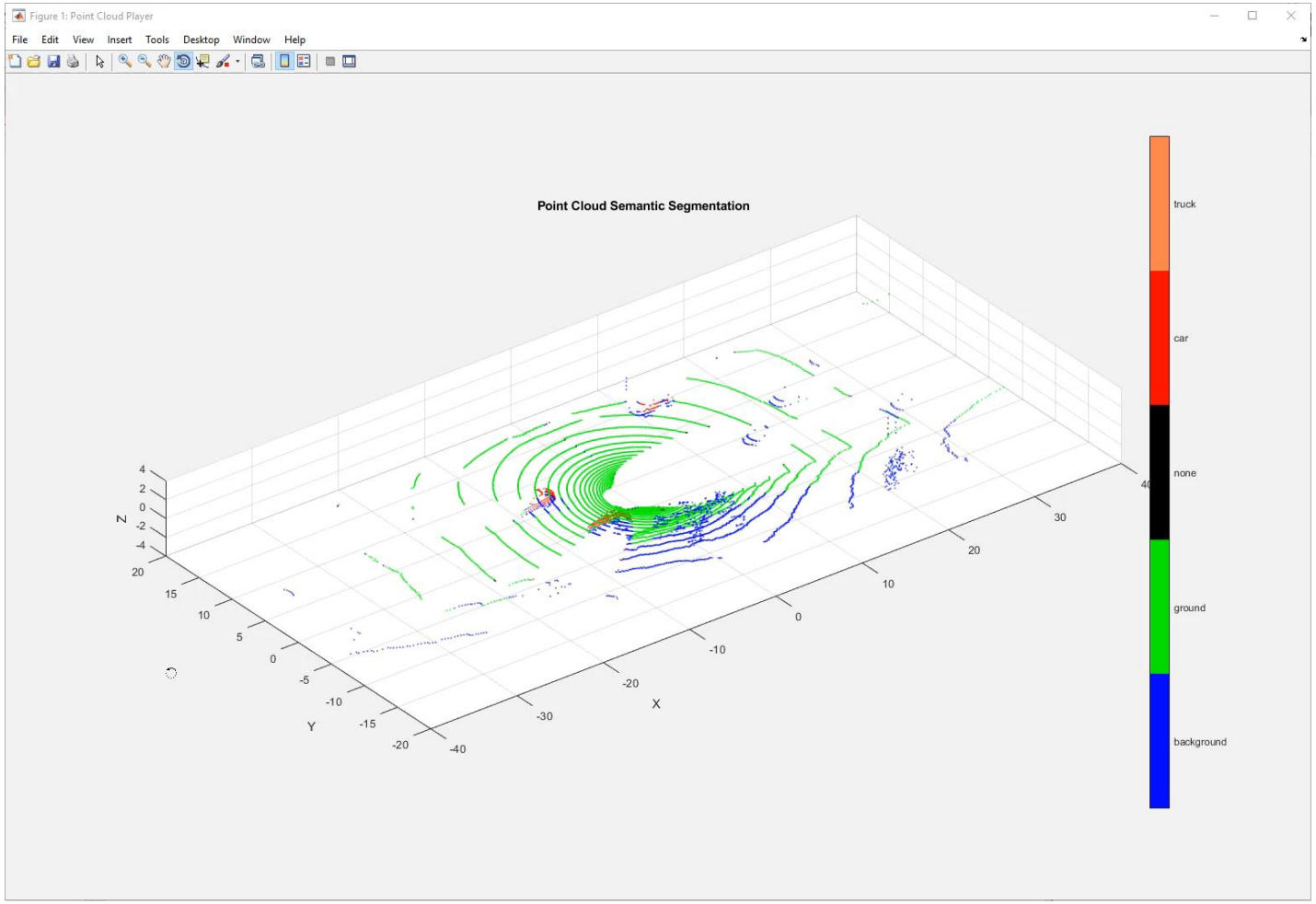
R2018a

Code Generation for Semantic Segmentation Network

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

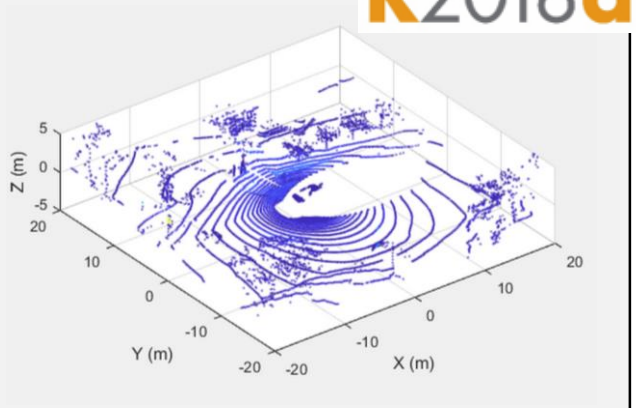
GPU Coder™

LIDAR Point Cloud Segmentation



Learn about developing lidar perception algorithms with these examples

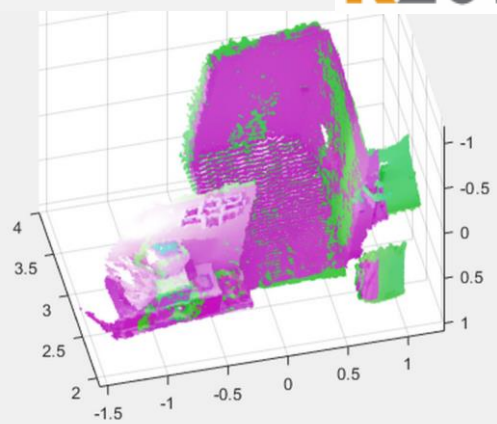
R2018a



Display Point Clouds from Velodyne PCAP File

- **Read Velodyne** files
  `velodyneFileReader`
  Automated Driving System Toolbox™

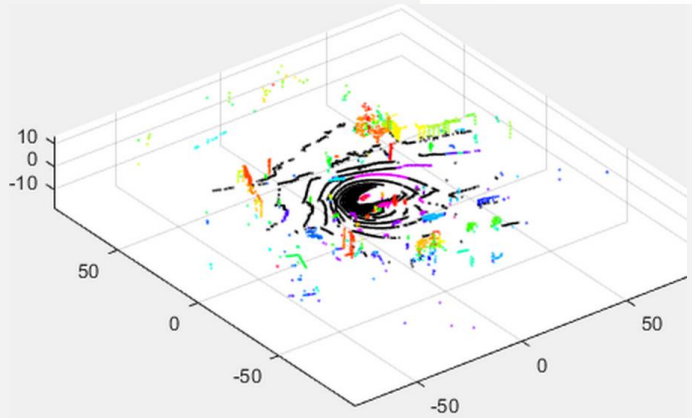
R2018a



Align Two Point Clouds Using NDT Algorithm

- **Register** point clouds with Normal Distributions Transform
  `pcregisterndt`
  Computer Vision System Toolbox™

R2018a



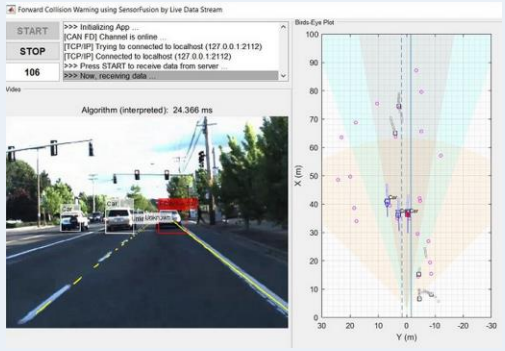
Cluster Organized Lidar Point Cloud

- **Segment** lidar point cloud
  `segmentLidarData`
  Automated Driving System Toolbox™

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

Deep learning

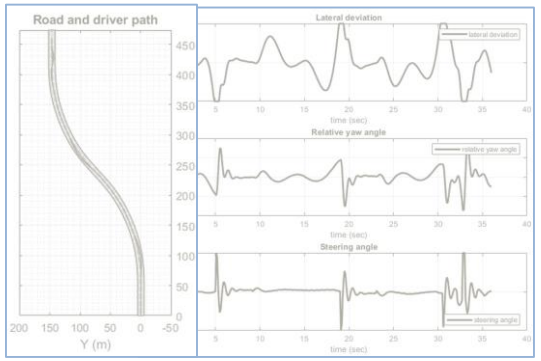
Sensor fusion



Perception

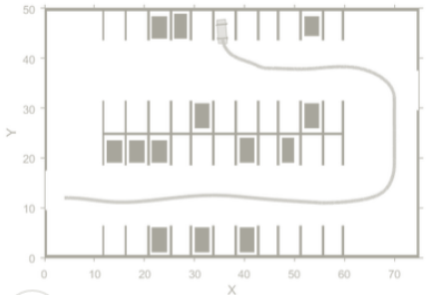
Control

Sensor models & model predictive control



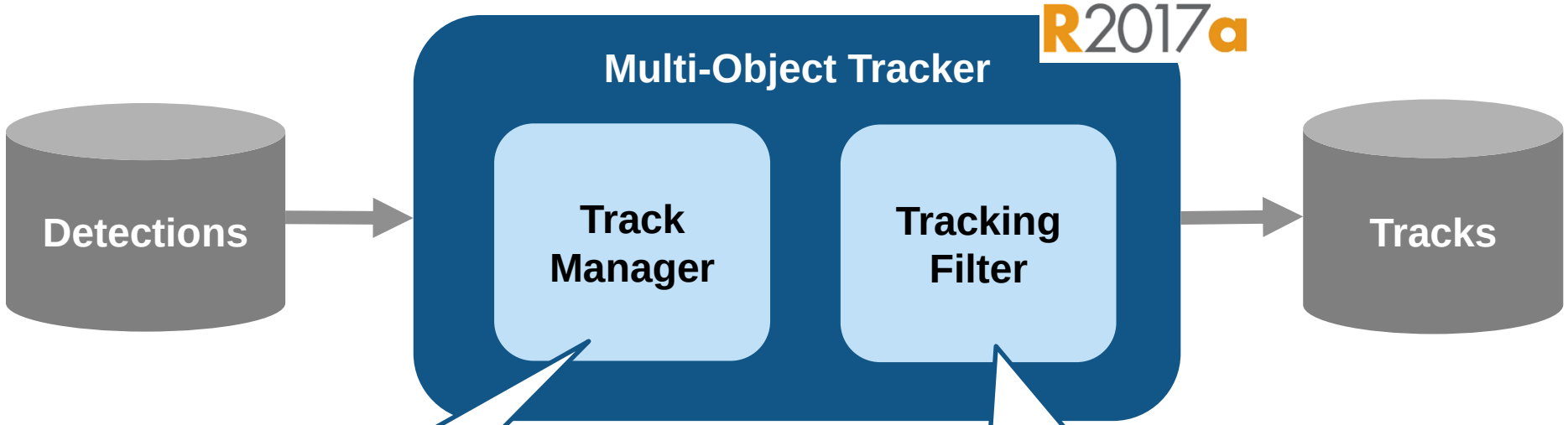
Planning

Path planning



Automated Driving System Toolbox introduced:

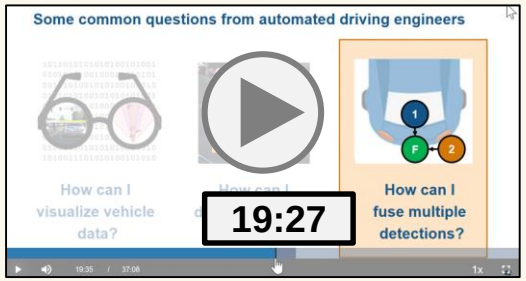
Multi-object tracker to develop sensor fusion algorithms



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

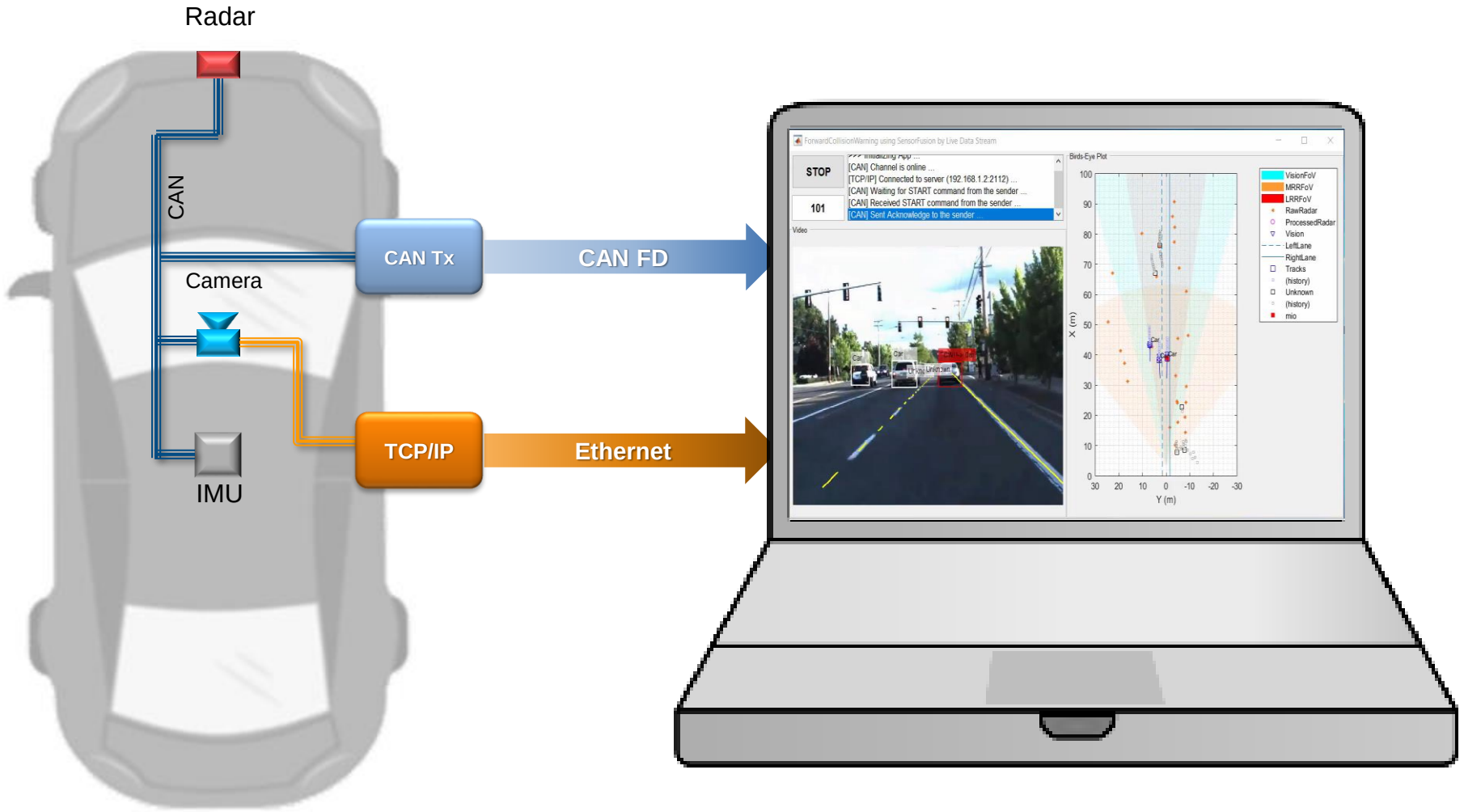
- Predicts and updates state of track
- Supports linear, extended, and unscented Kalman filters

Videos and Webinars

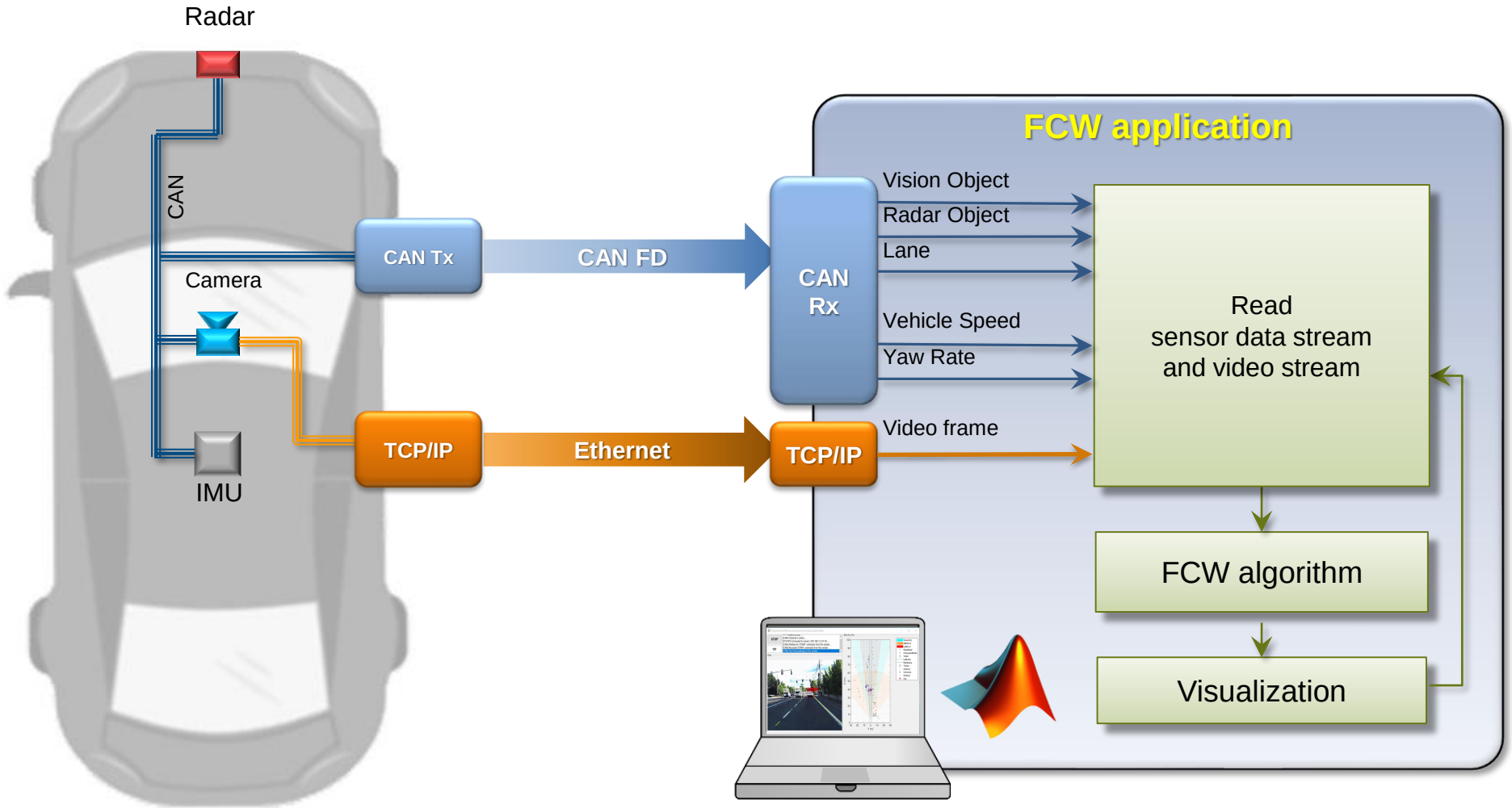


Introduction to Automated Driving System Toolbox

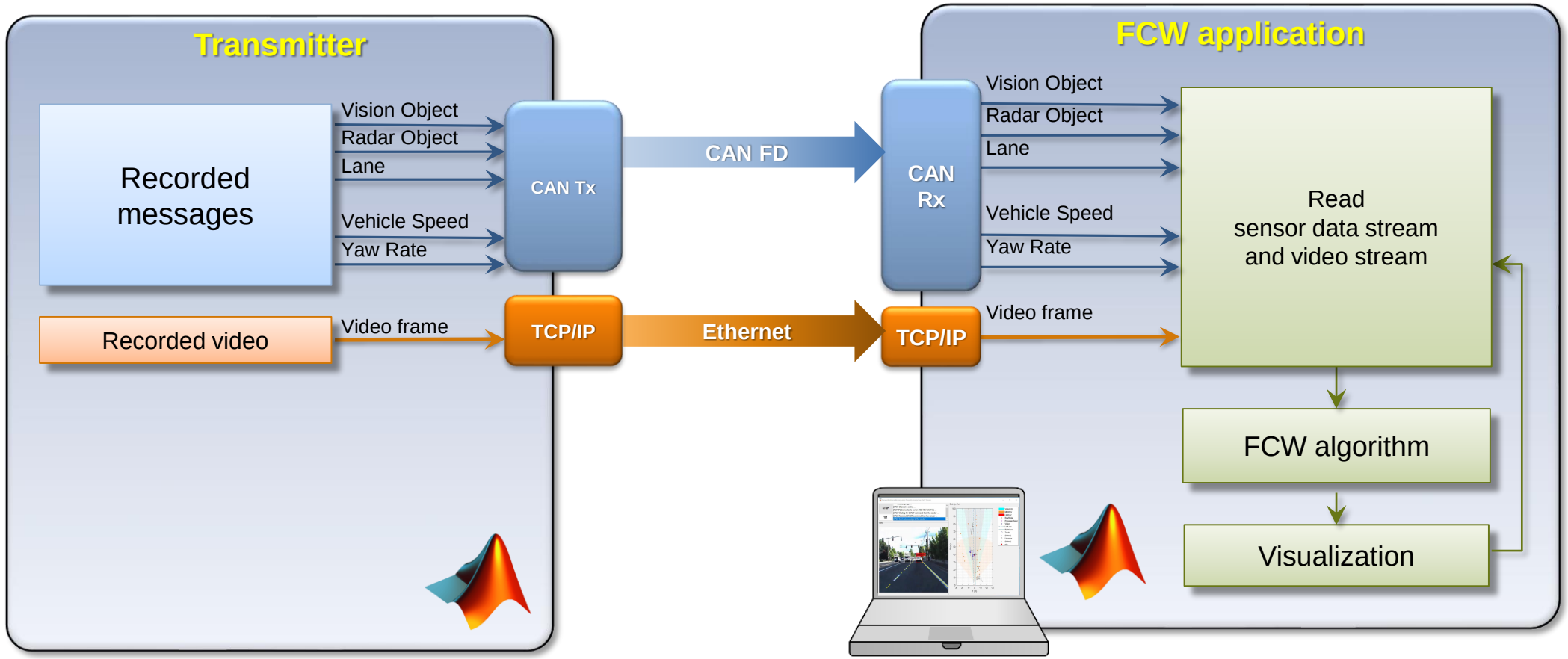
How can I test my sensor fusion algorithm with live data?



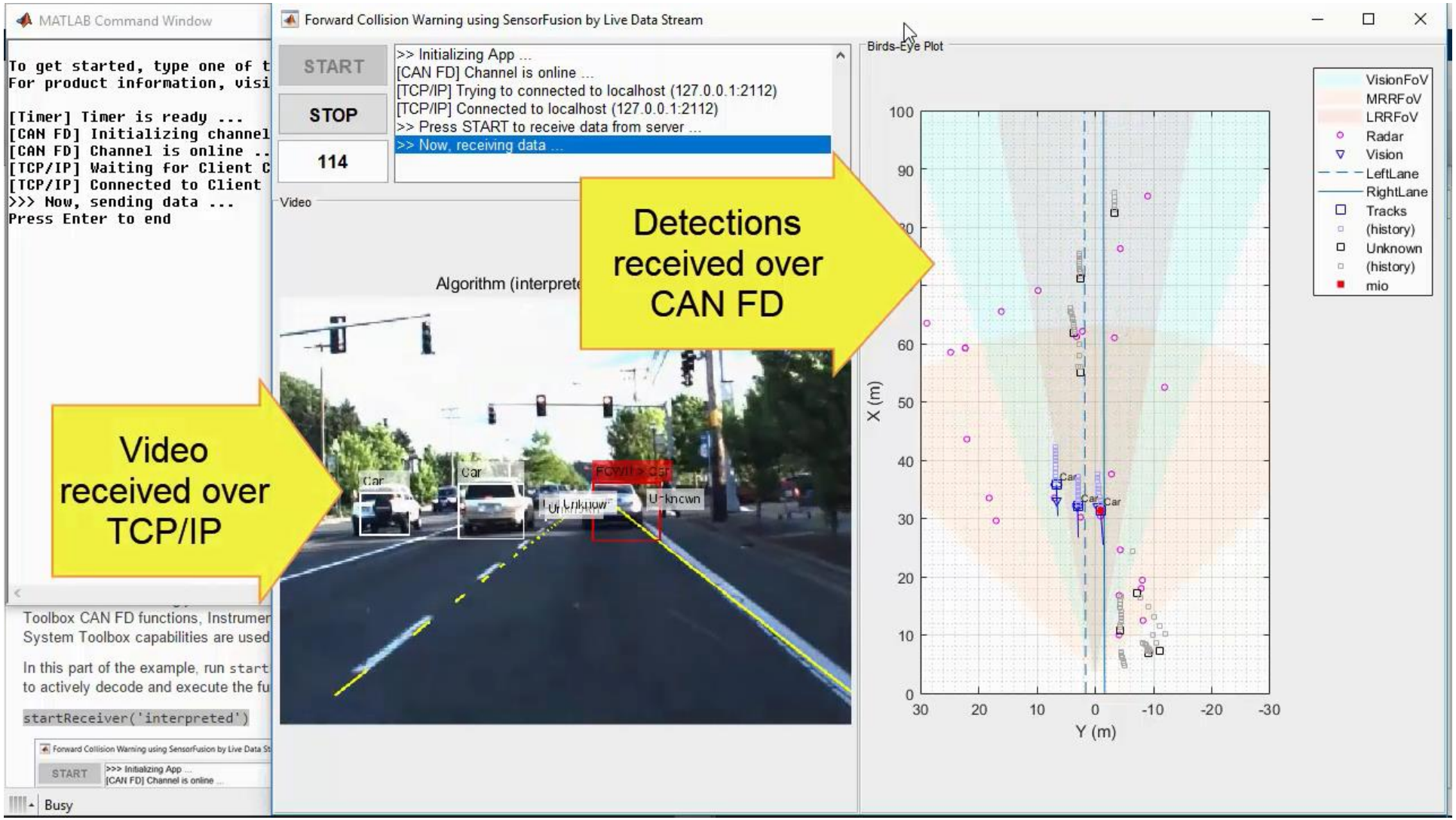
Test forward collision warning algorithm with live data from vehicle



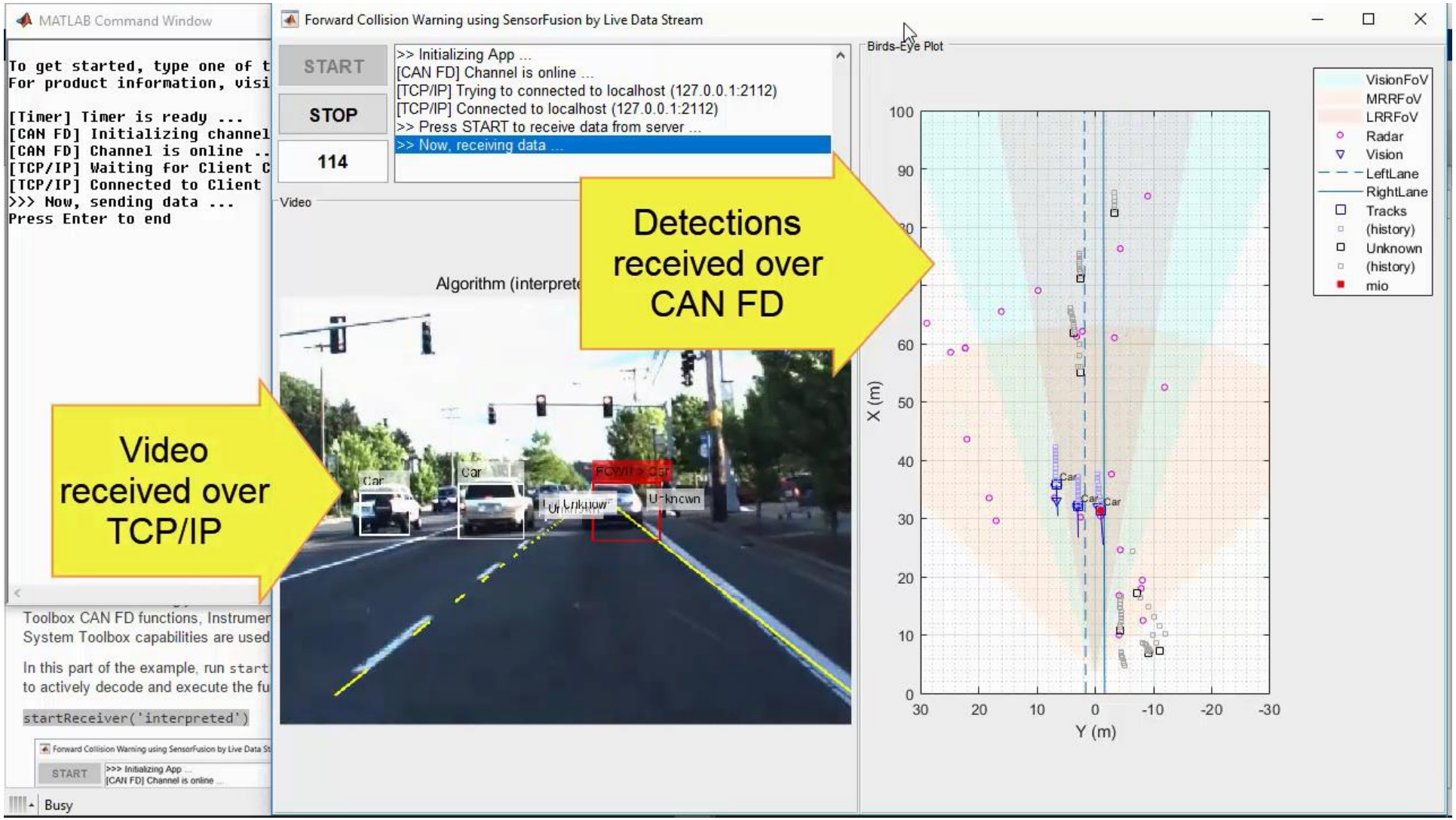
Test forward collision warning algorithm with live data from “surrogate” vehicle



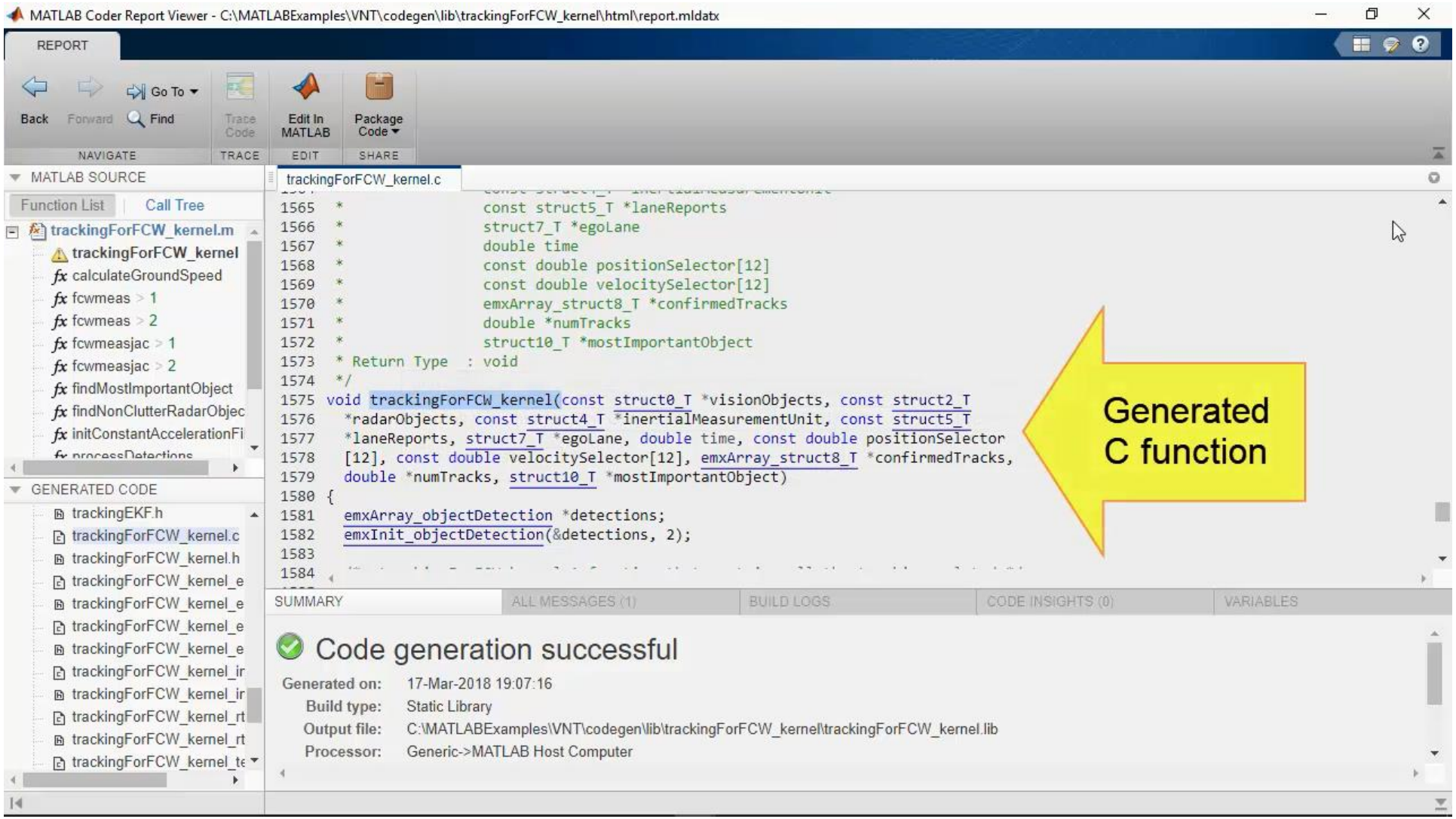
Send live CAN FD and TCP/IP data



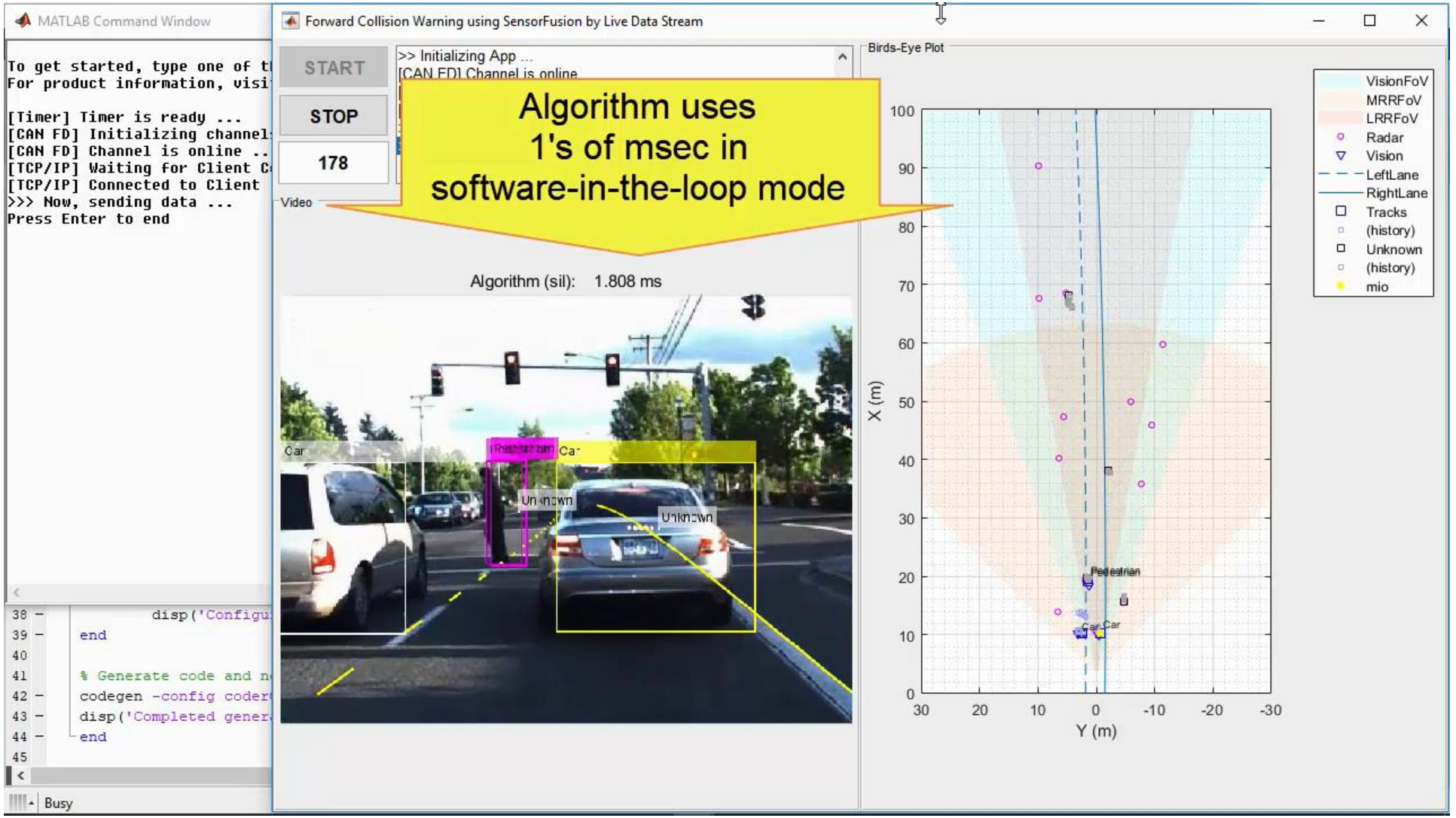
Receive live CAN FD and TCP/IP data



Generate C/C++ code for algorithm



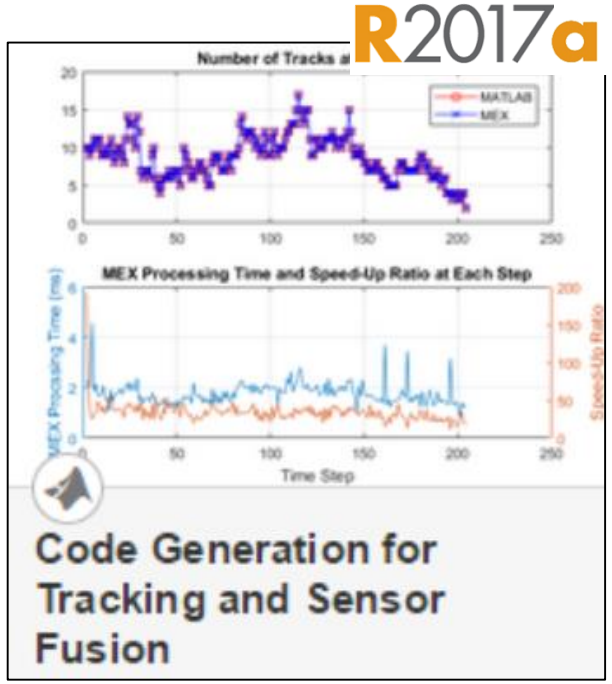
Stream live CAN FD and TCP/IP data into compiled algorithm code



Learn more about developing sensor fusion algorithms

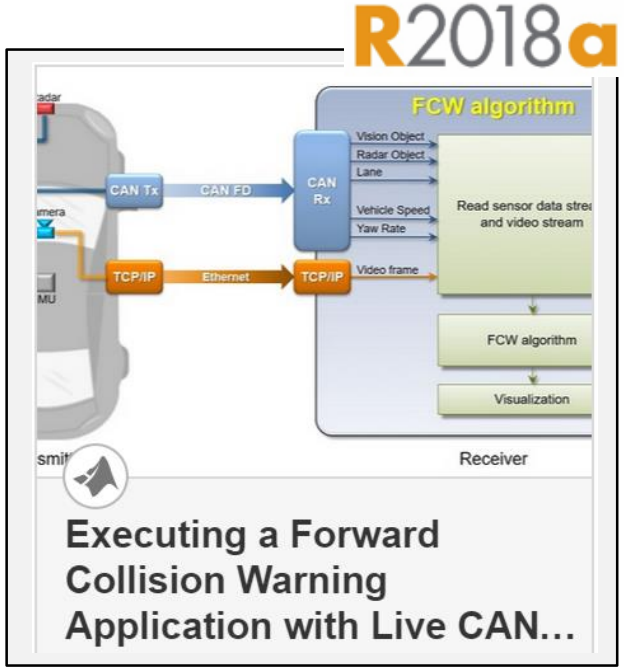
- **Design** algorithm with multi-object tracker and recorded vehicle data

  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?

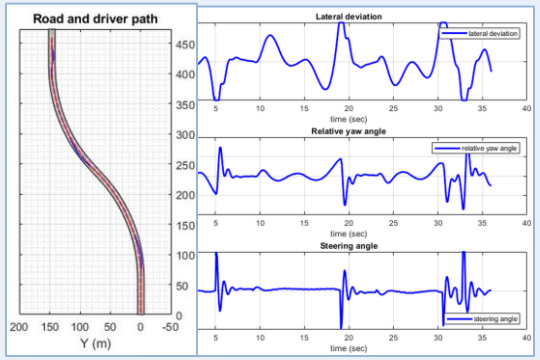
Deep learning

Perception

Sensor fusion

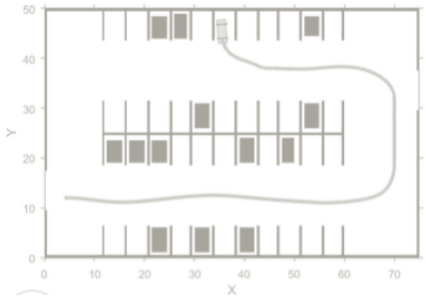
Control

Sensor models & model predictive control



Planning

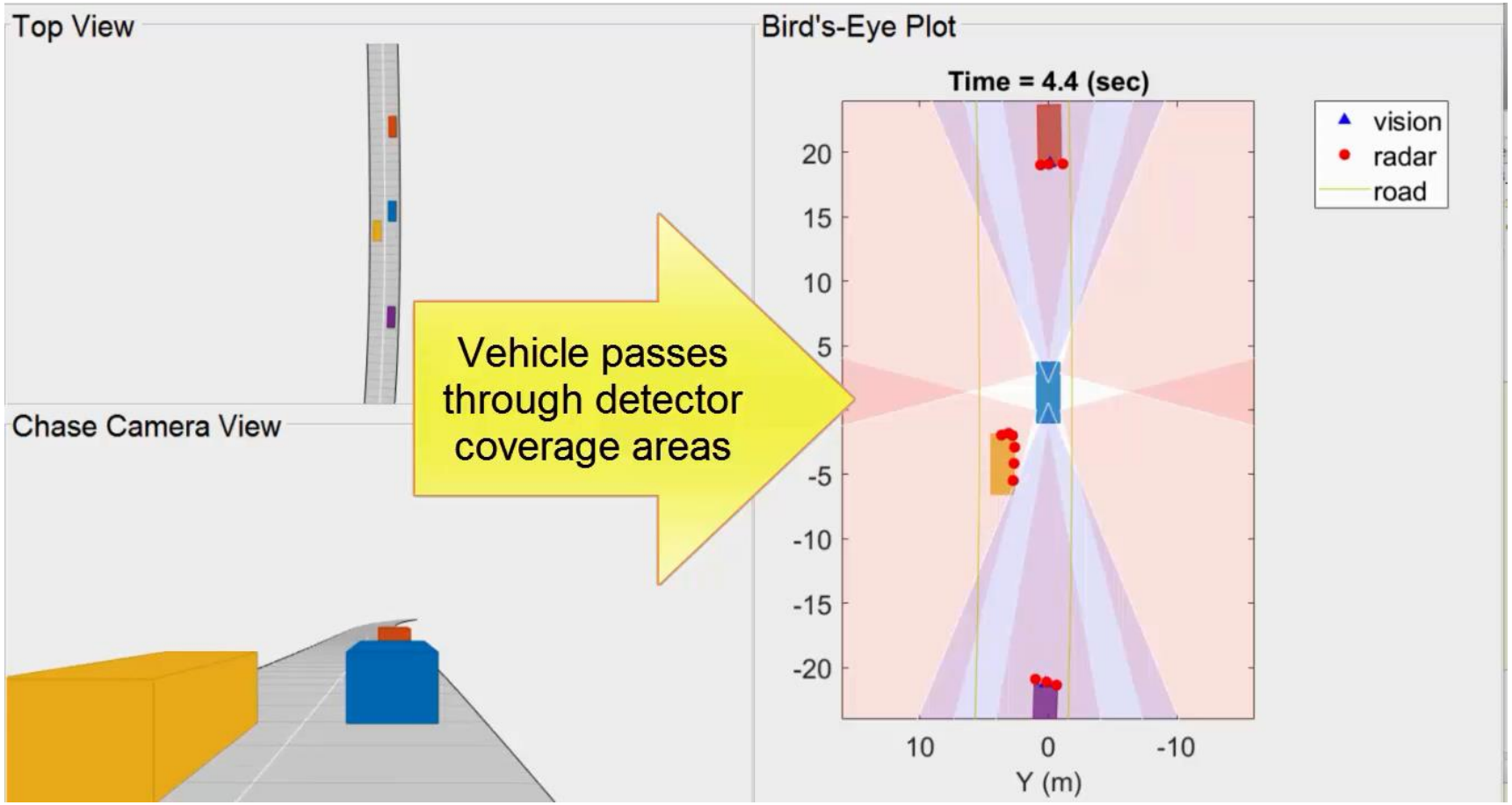
Path planning

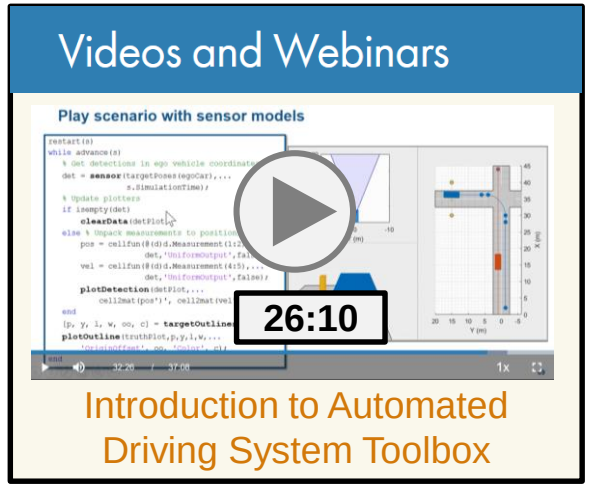


Automated Driving System Toolbox introduced:

Synthesizing scenarios to test sensor fusion algorithms

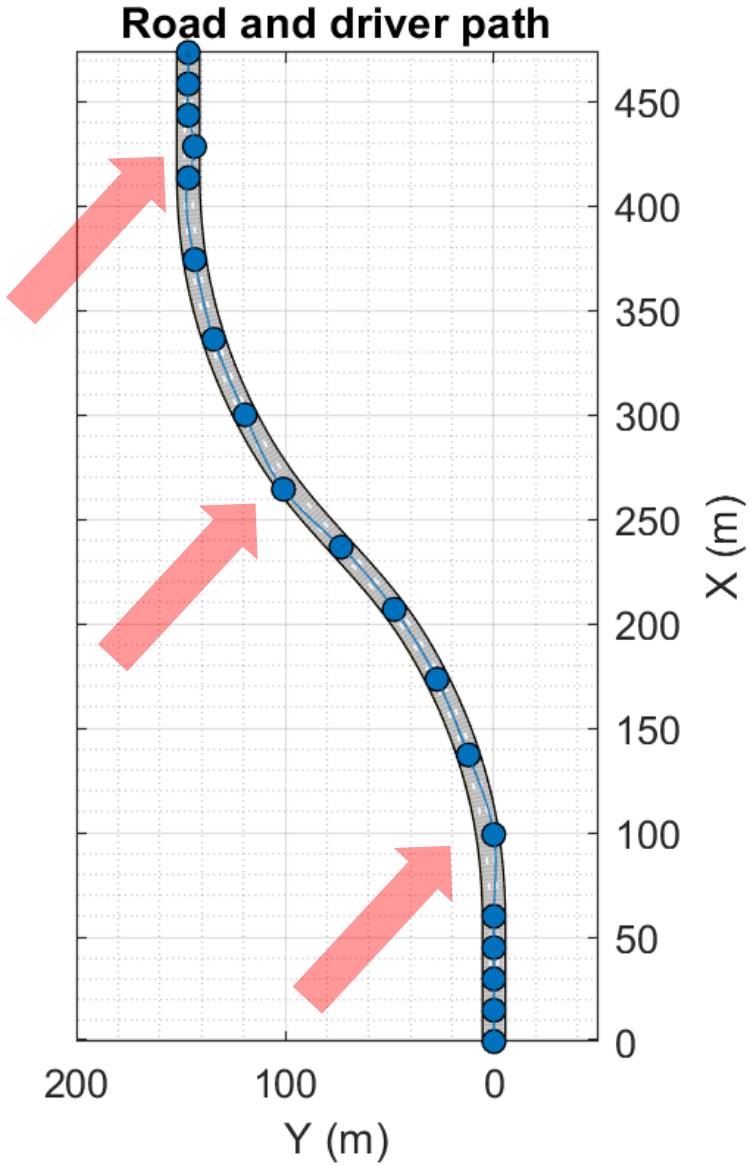
R2017a

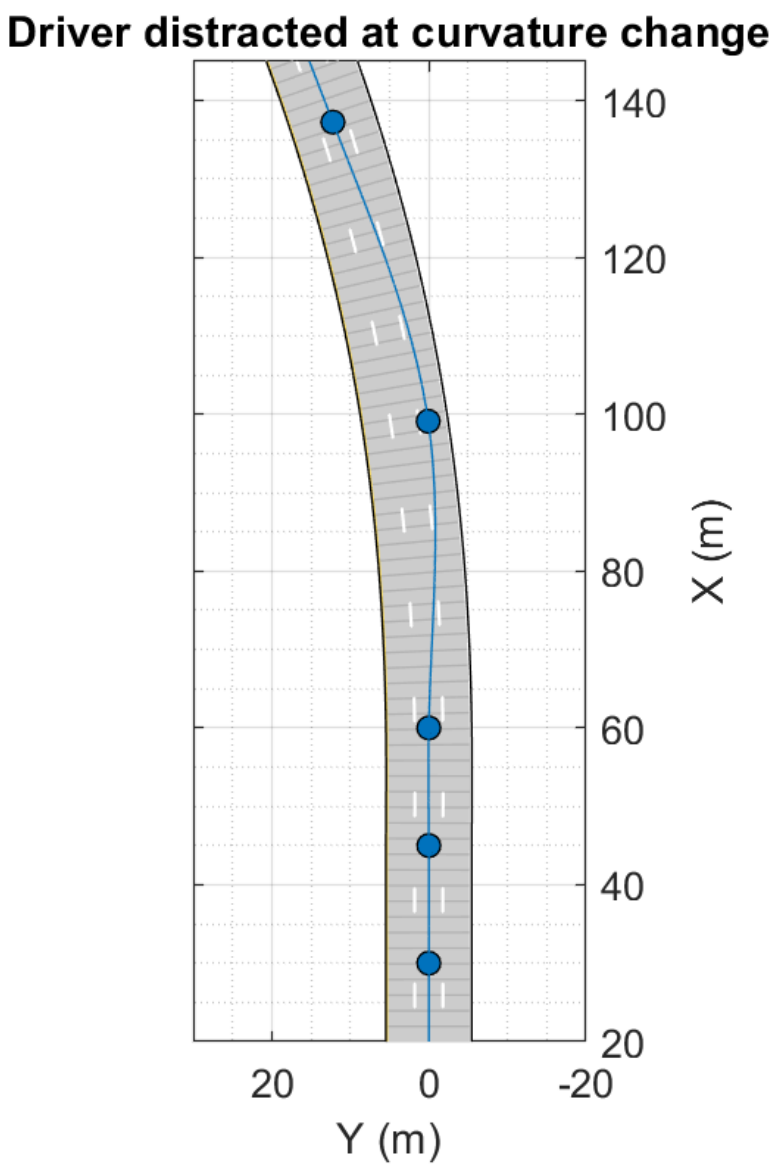




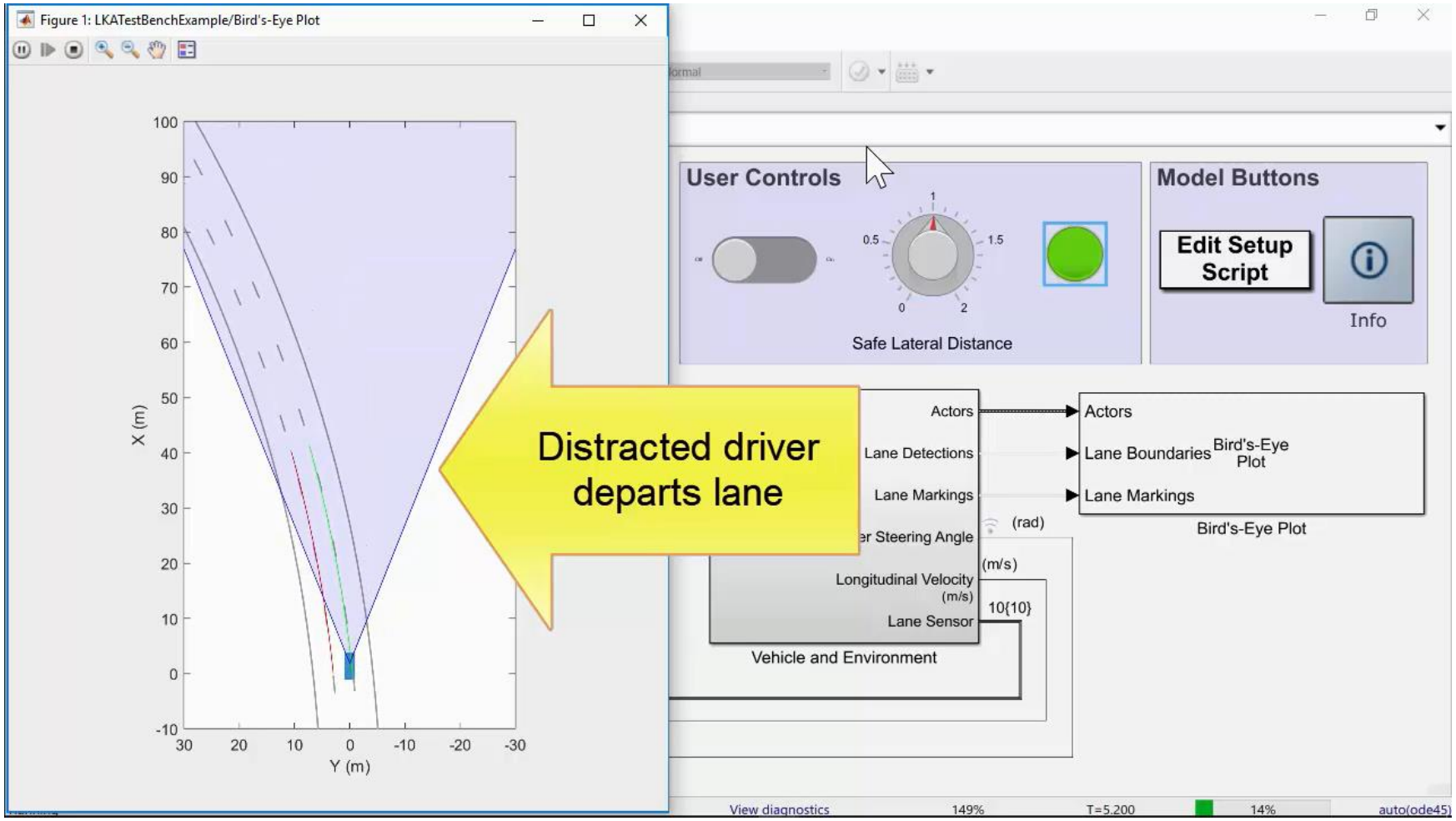
Create highway double curve with drivingScenario

- Driver waypoints simulate distraction at curvature changes

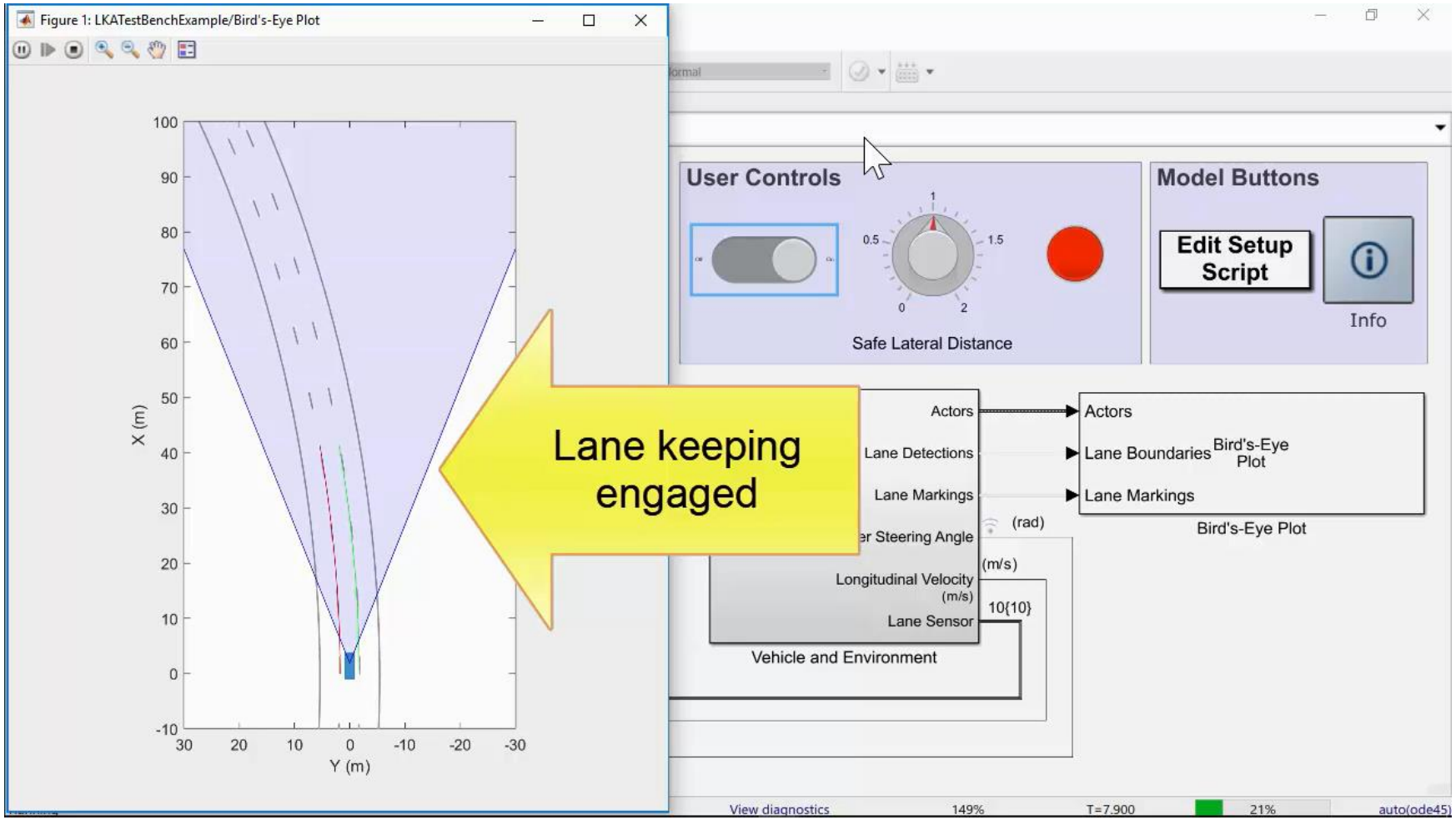




Simulate distracted driver

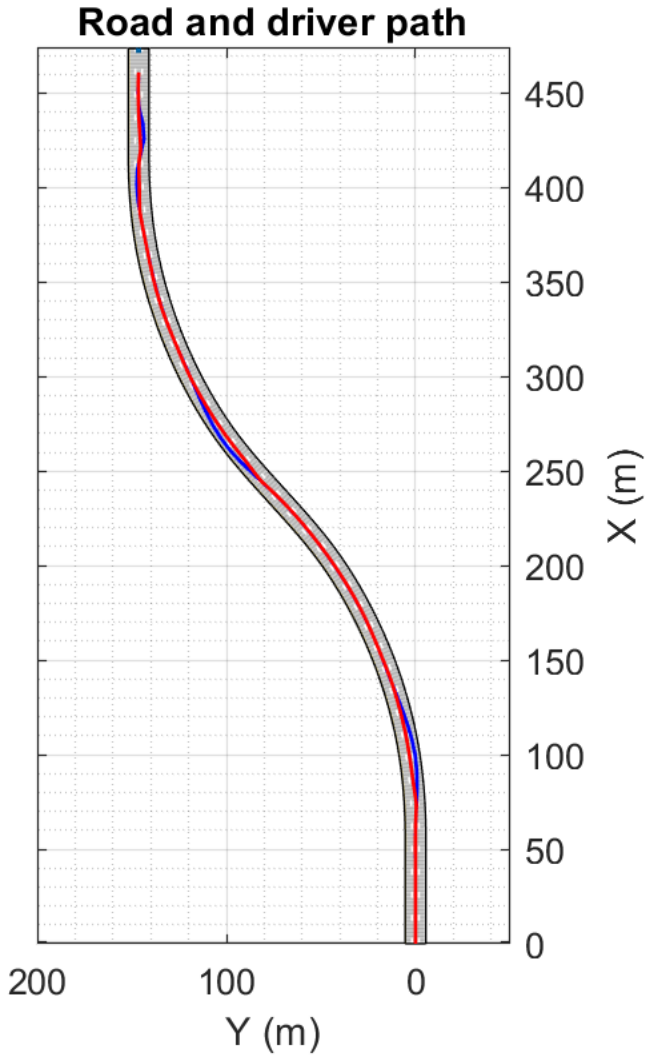


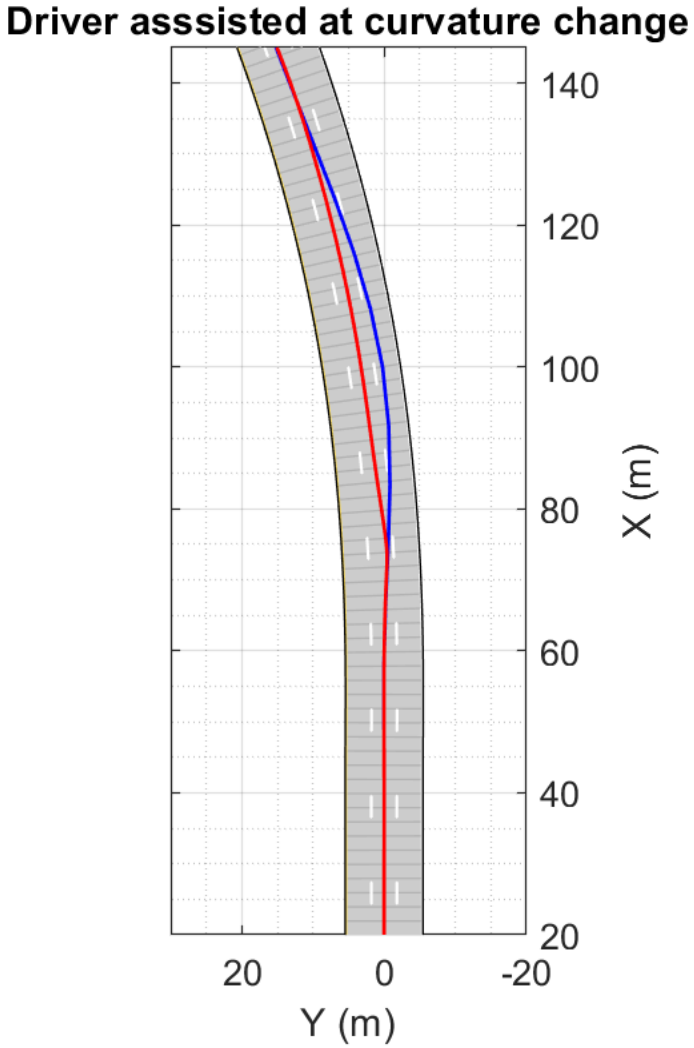
Simulate lane keep assist at distraction events



Compare distracted and assisted results

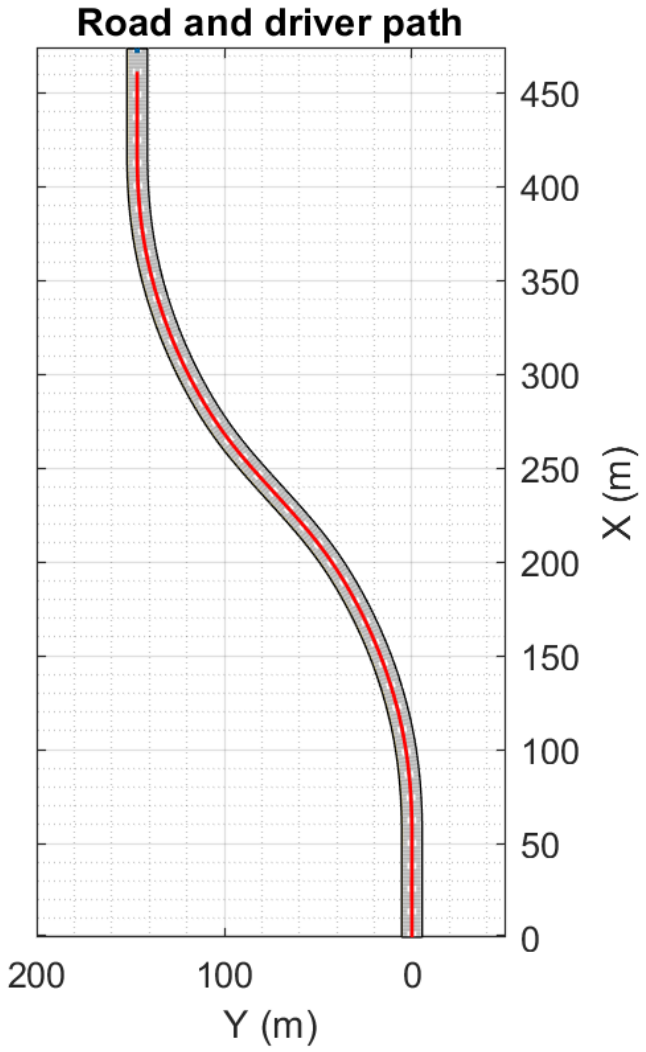
- Detect lane departure and maintain lane during distraction

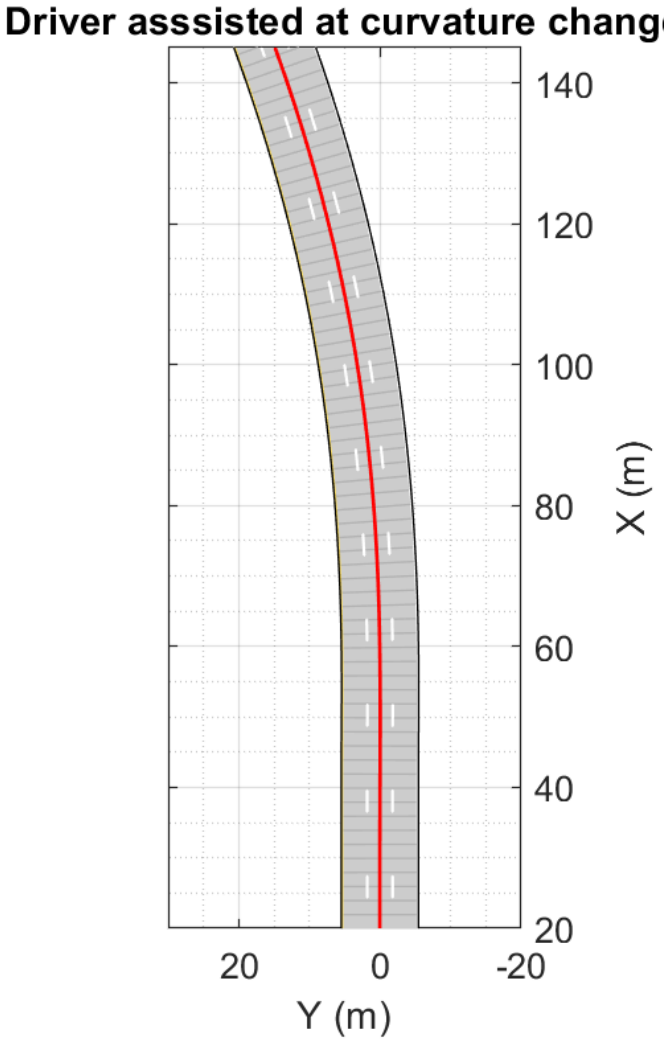




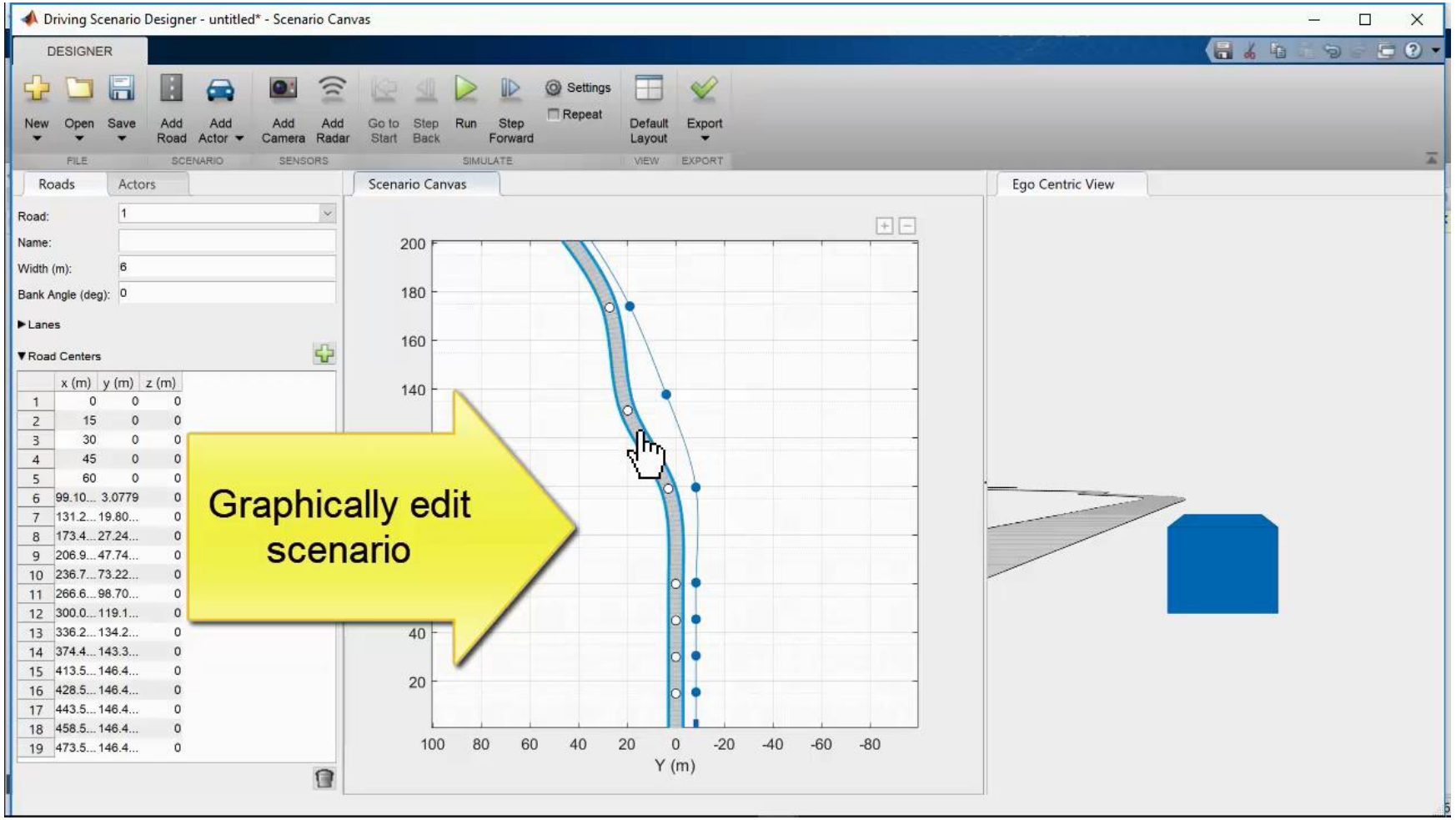
Explore lane following results

- Vehicle stays within lane boundaries

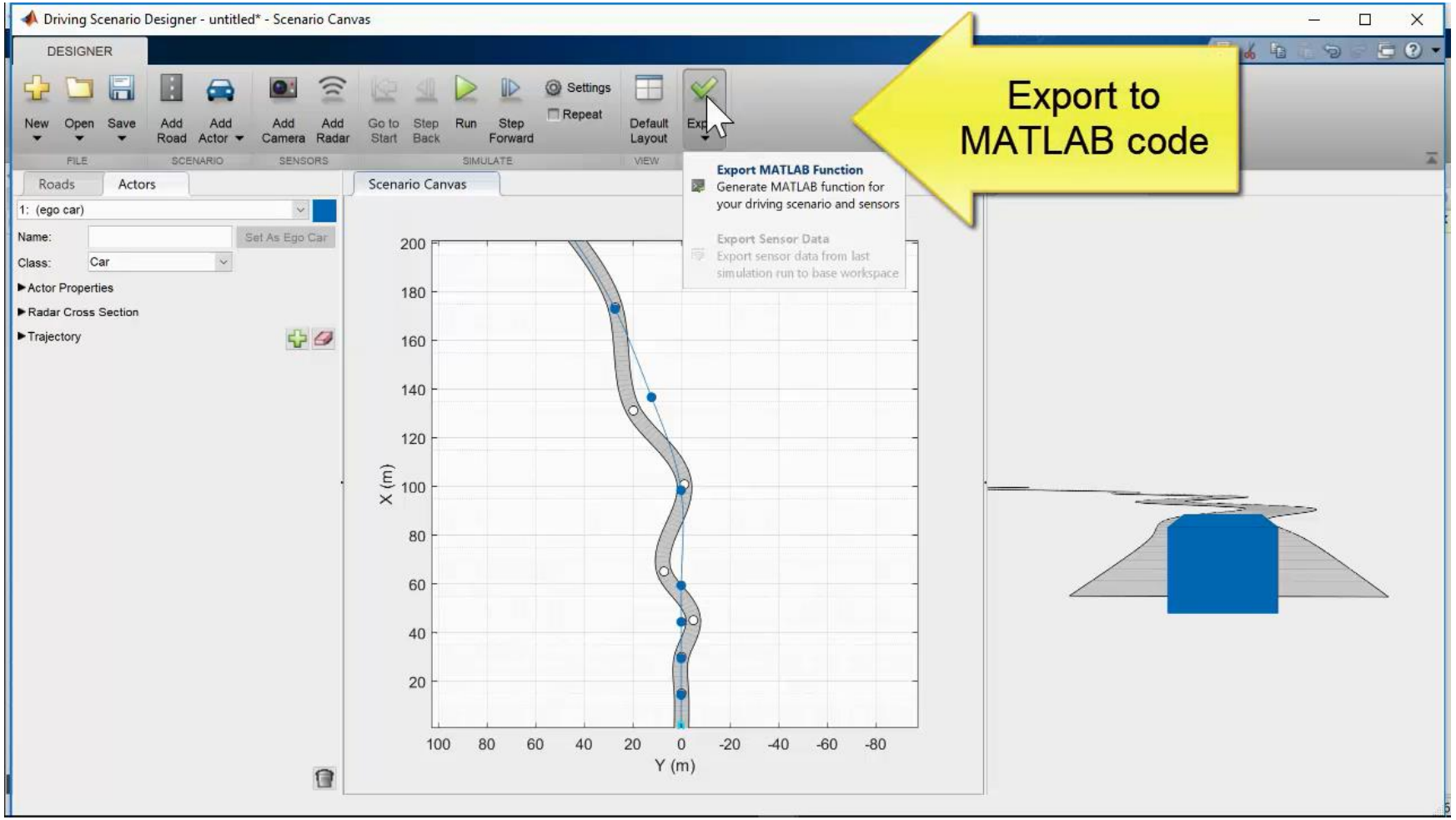




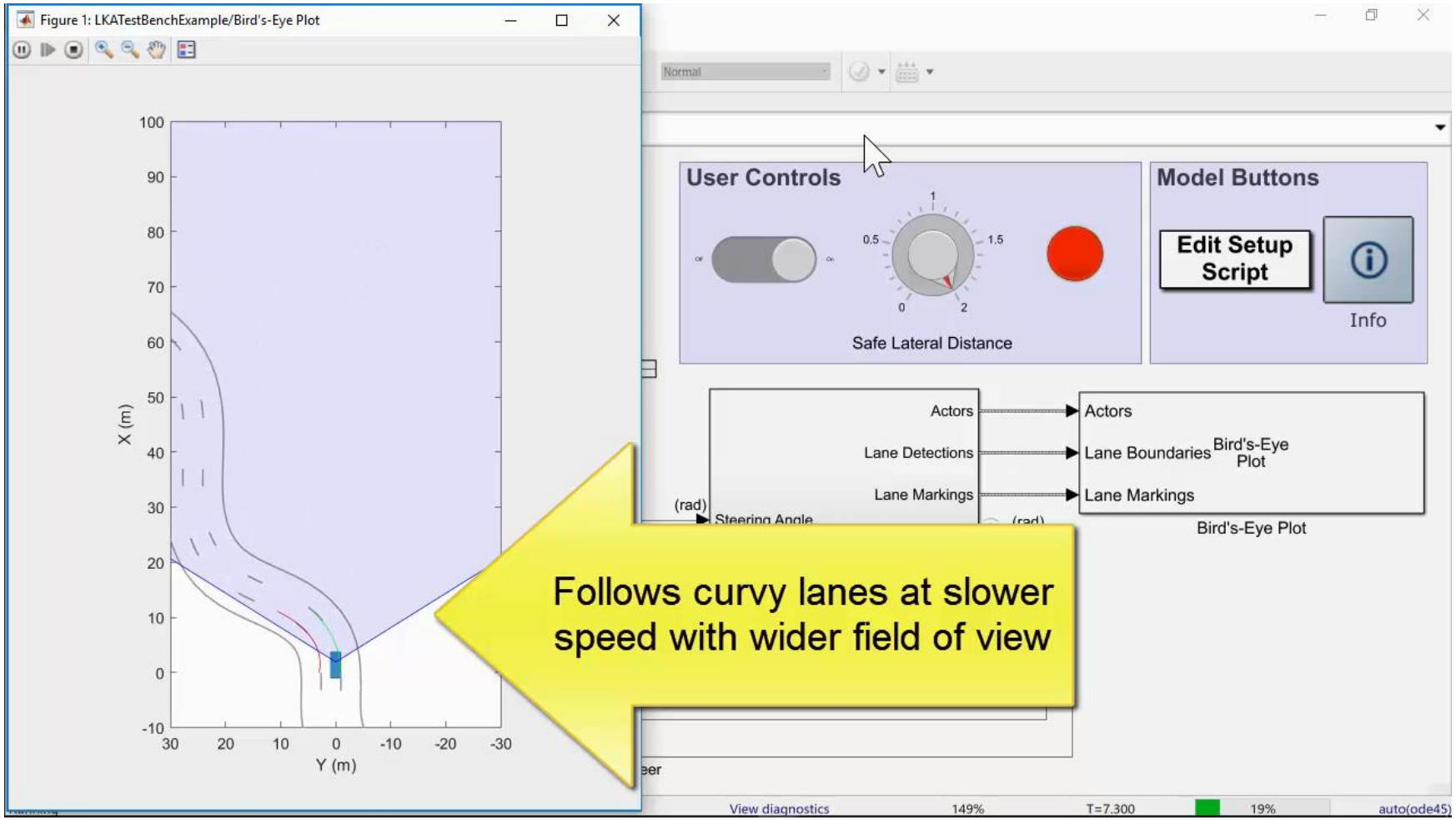
Graphically edit scenarios with Driving Scenario Designer



Export MATLAB code to generate scenarios

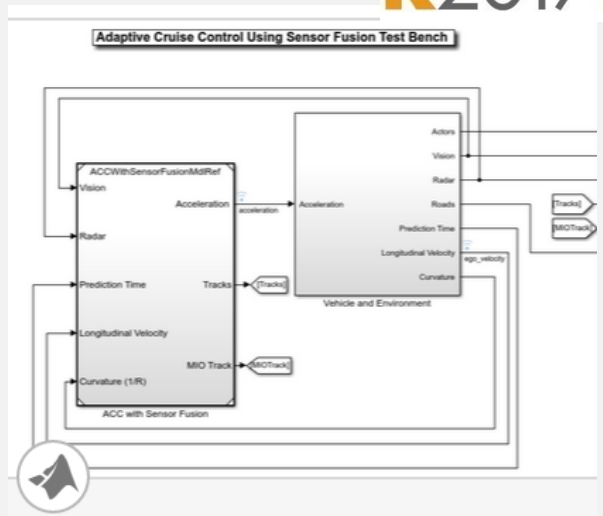


Explore what is required to follow high curvature paths



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

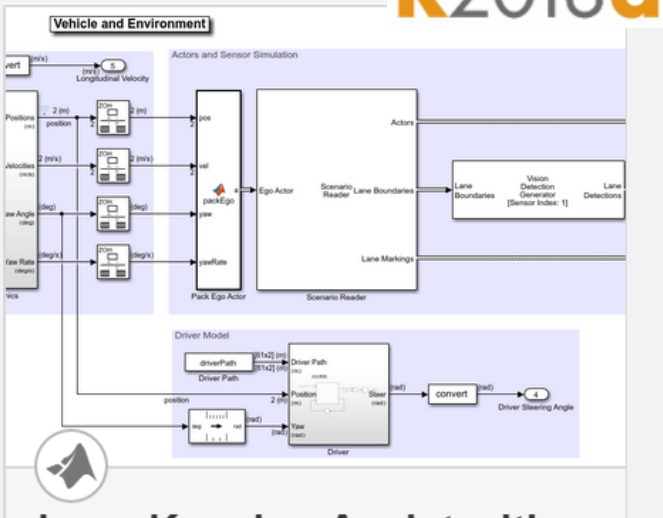
R2017b



Adaptive Cruise Control with Sensor Fusion

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

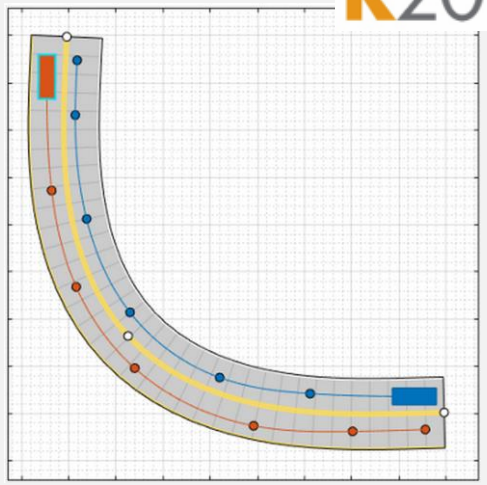
R2018a



Lane Keeping Assist with Lane Detection

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

R2018a



Generate Synthetic Detections from an Interactive Driving Scenario

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

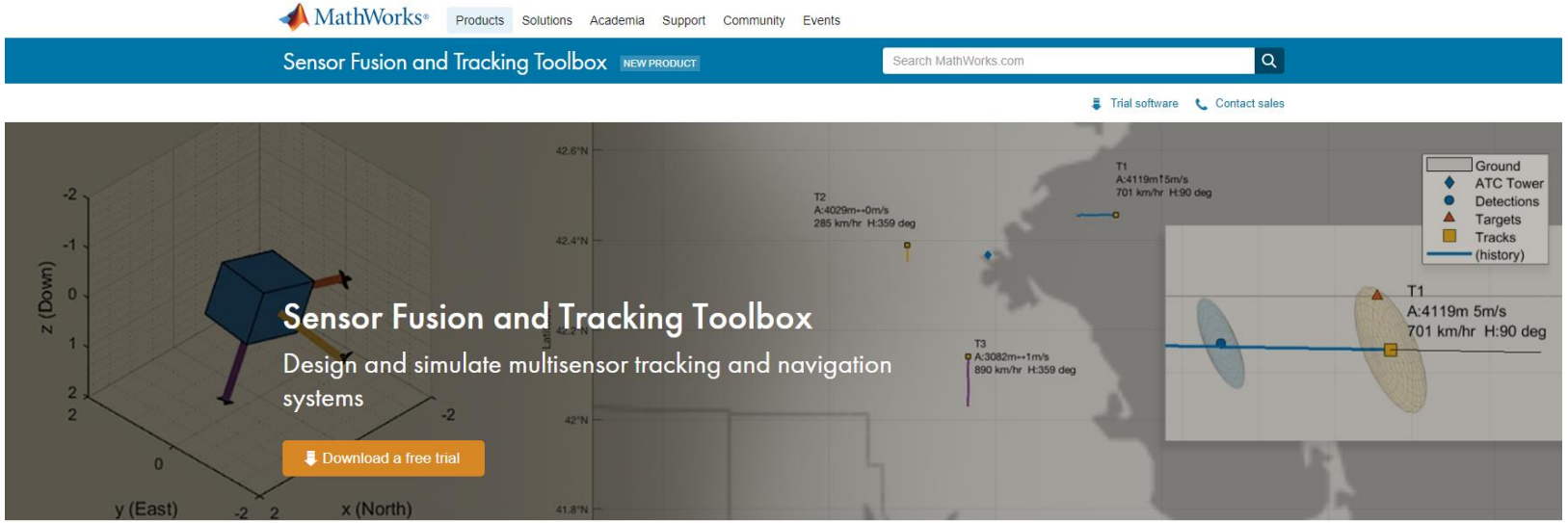
Sensor Fusion and Tracking Toolbox

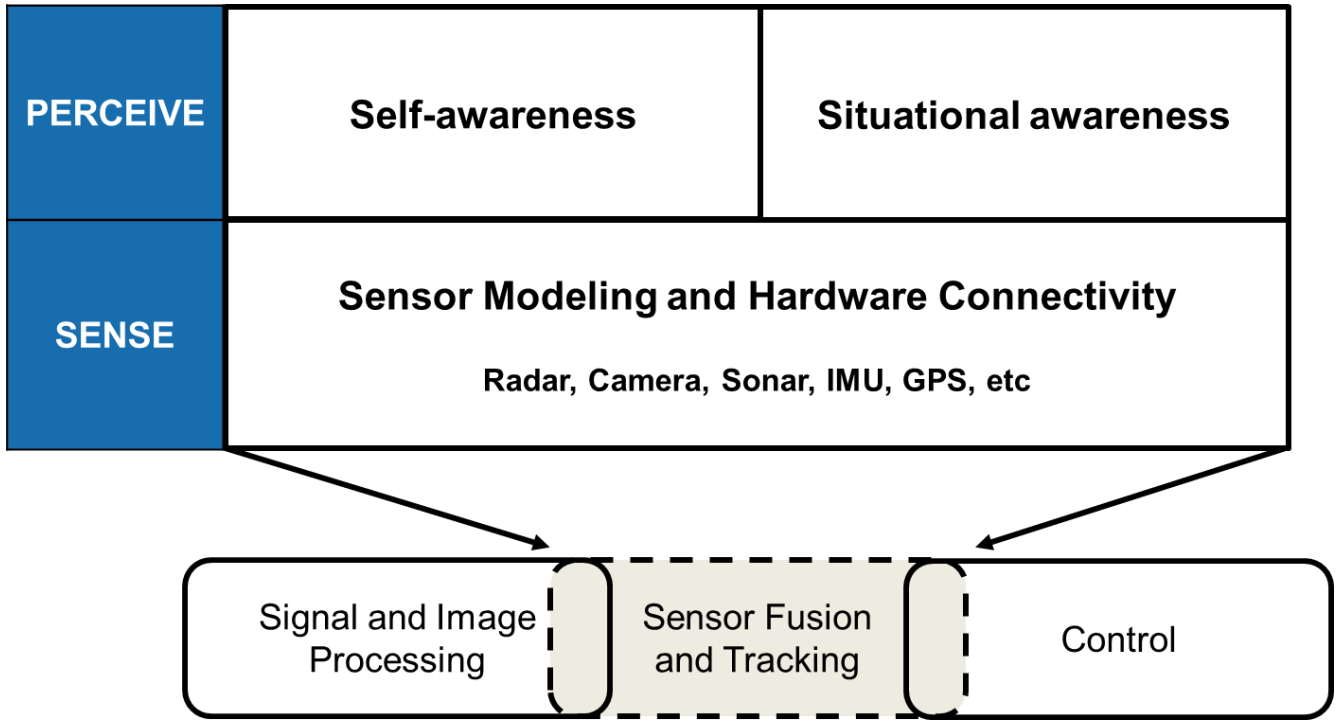
Extending the Automated Driving Workflow

- Localization for pose, orientation, and position
- Tracker and fusion algorithm development
- Metrics – accuracy and track assignment

In this part:

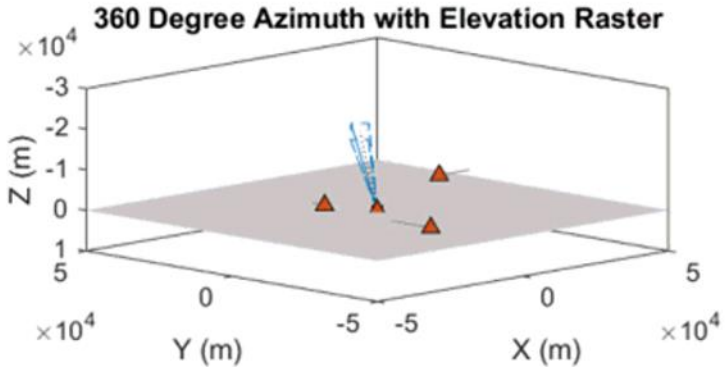
- Scope
- Overview
- Applications

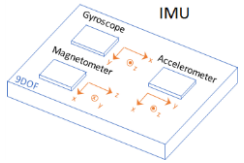


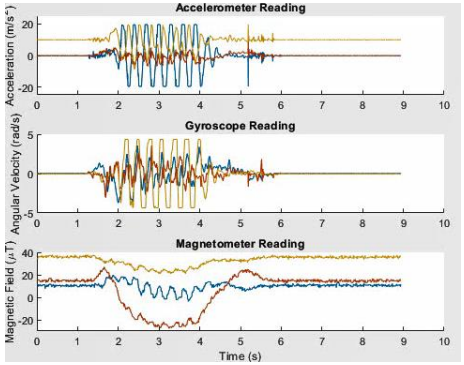


Overview: Sensor Fusion and Tracking Toolbox

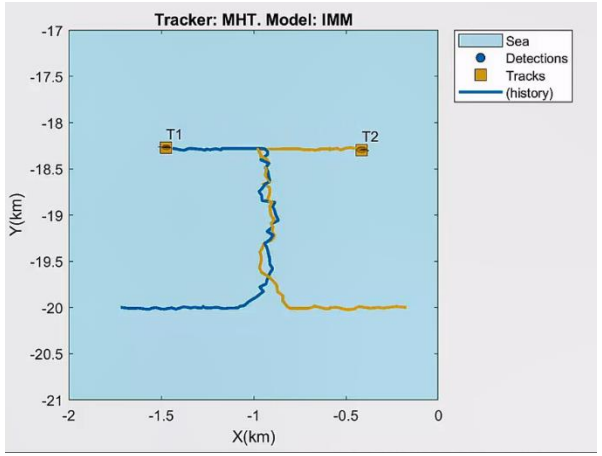
R2018b

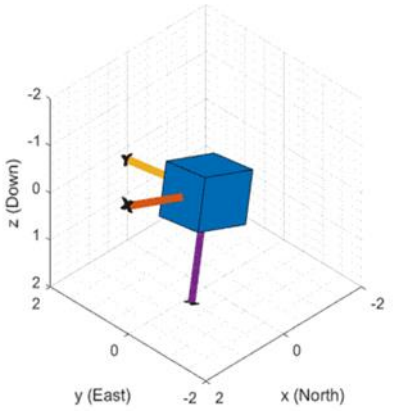




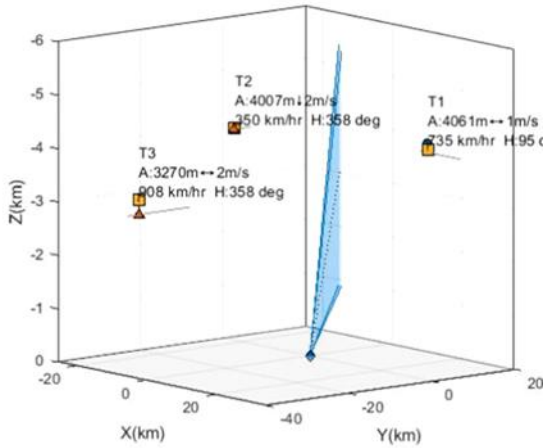


Scenarios and Sensors Simulation



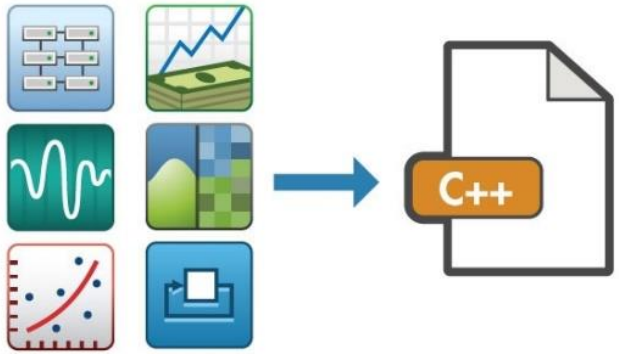


Tracking and Localization Algorithms



TrackID	AssignedTruthID	TotalLength
2	NaN	1
3	NaN	1
4	8	55
5	7	56
6	NaN	2

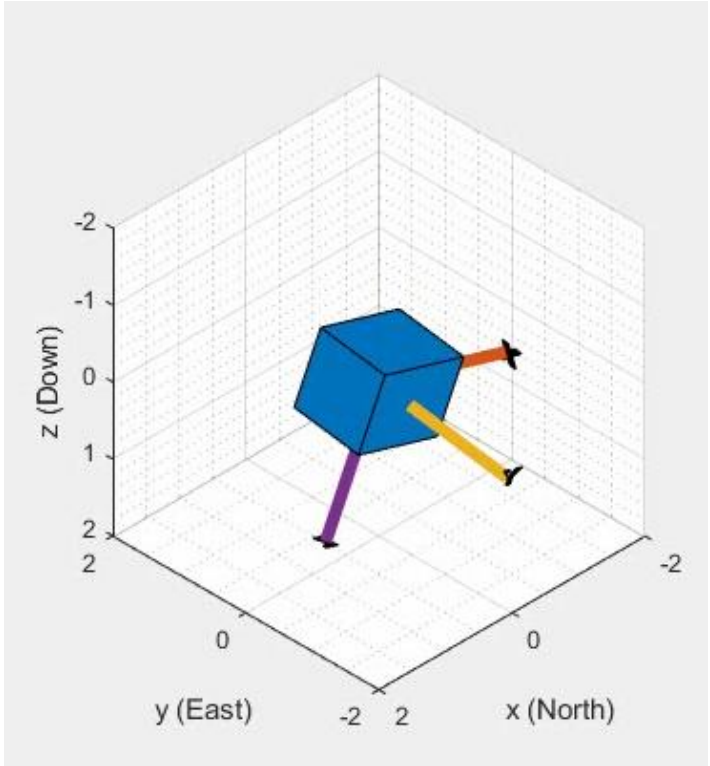
Visualization and Metrics



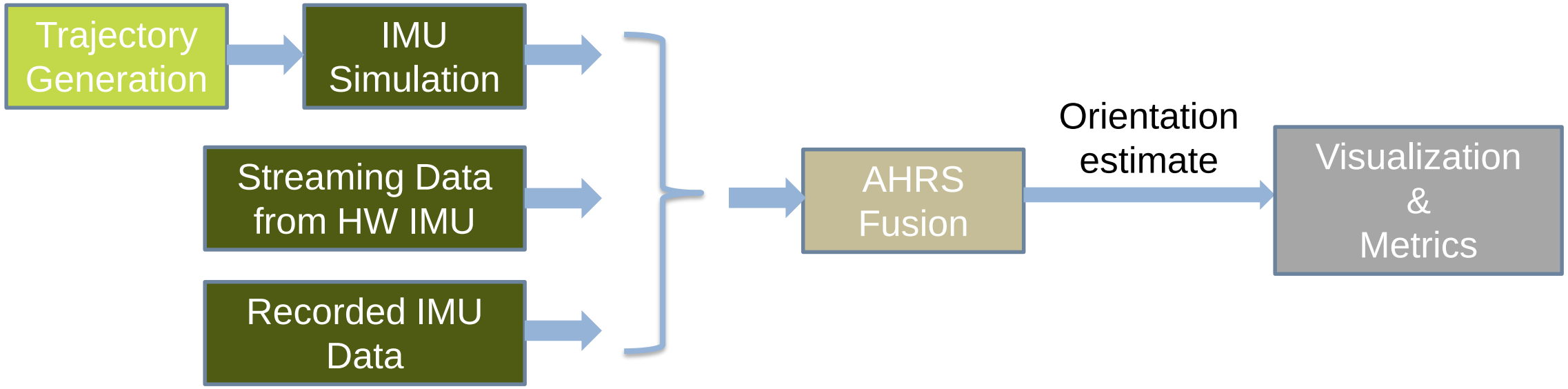
Code Generation

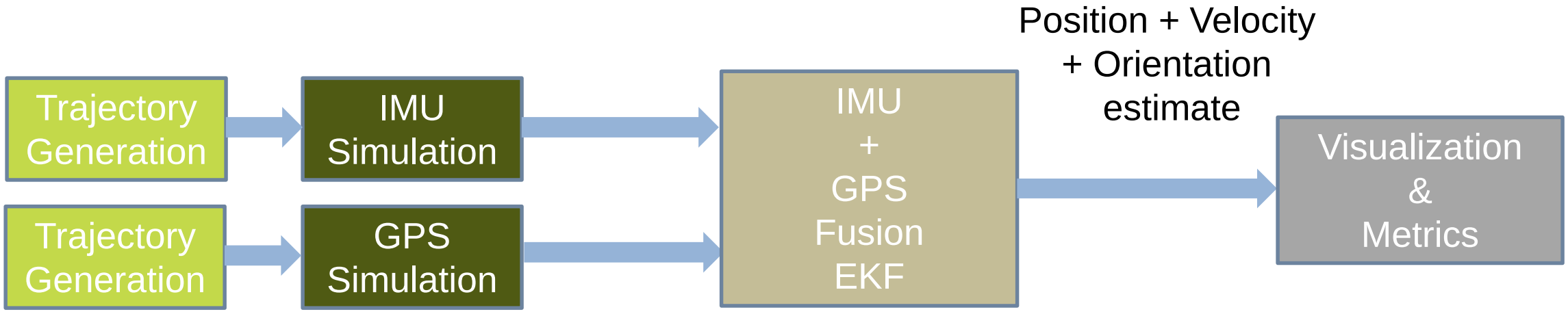
Localization

- **Workflows**
- **IMU and GPS Sensor Models**
- **Orientation estimation** ***(6-DOF, 9-DOF, 10-DOF)***
- **Pose estimation** ***(IMU+GPS, IMU+Visual Odometry)***



Localization Algorithm Development Workflow



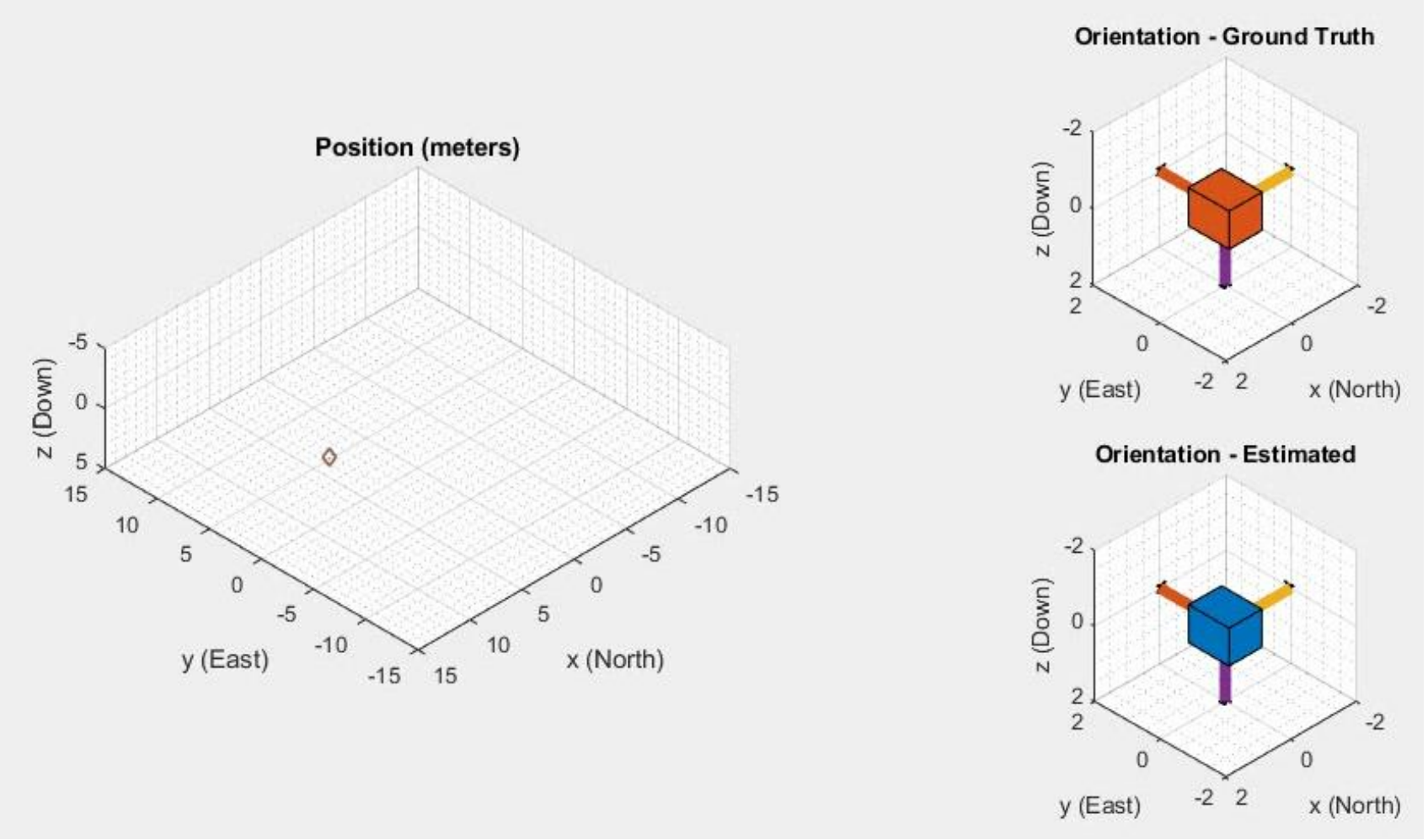


Estimate Orientation, Position and Velocity of Ground Vehicles

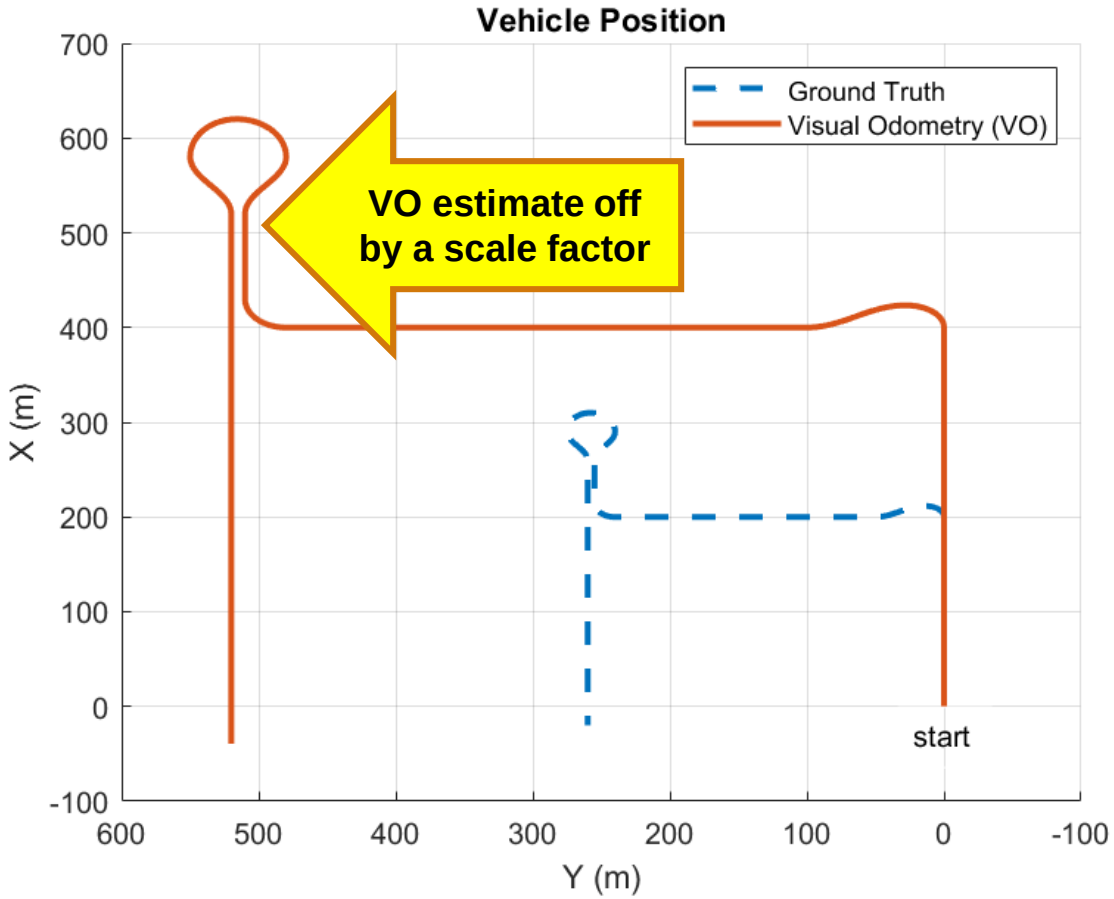
MathWorks R2018b

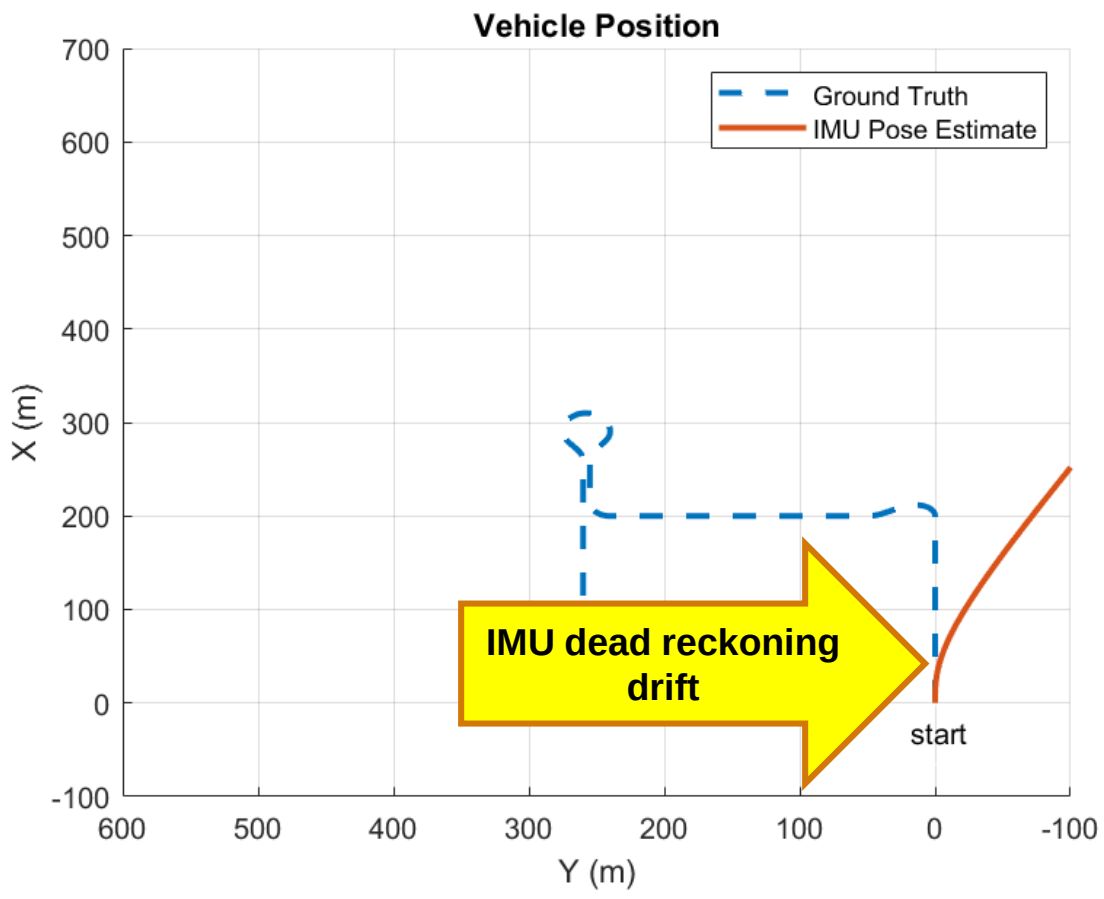
IMU + GPS Fusion

Non-holonomic Motion Constraint



Estimate Position of Ground Vehicle without GPS





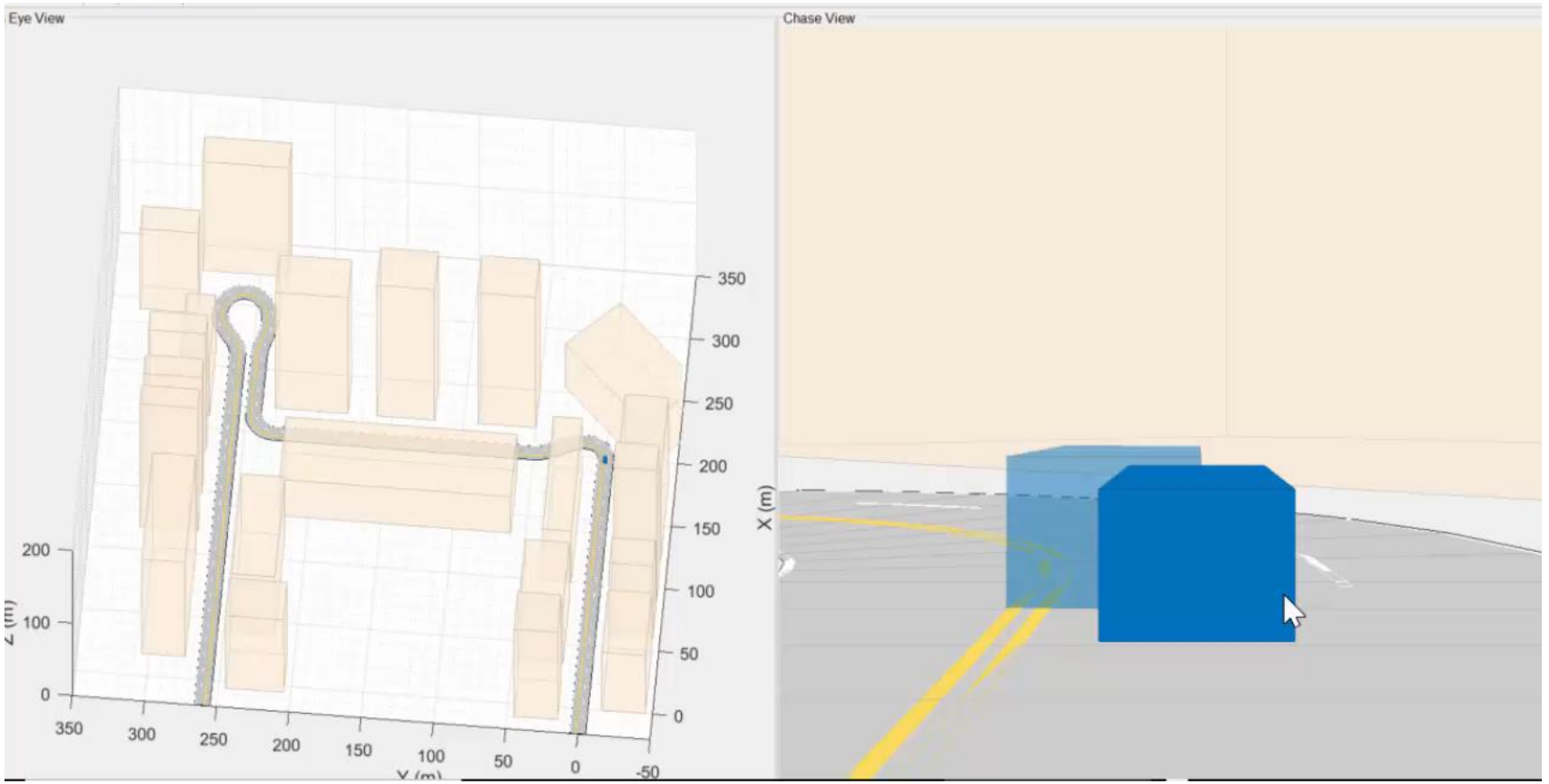
Estimate Position of Ground Vehicle without GPS

MathWorks R2018b

IMU + Camera Fusion

Fuse:
Accelerometer
Gyroscope
Camera (Visual Odometry)

Unconstrained Motion



Multi-Target Tracking

Tracking Algorithm Development Workflow

MathWorks R2018b

Recorded Sensor Data

Scenario Definition and Sensor Simulation

Ownship Trajectory Generation → INS Sensor Simulation

Actors/ Platforms → Radar, IR, & Sonar Sensor Simulation

objectDetection

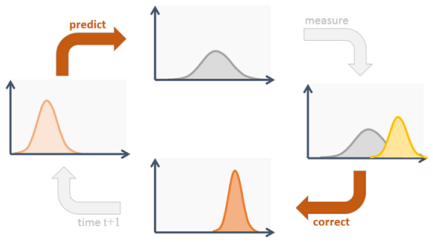
Tracking Algorithms

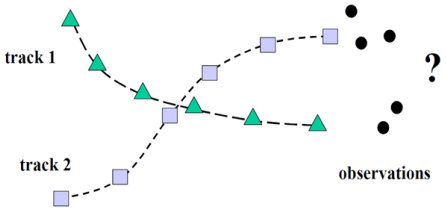
GNN, MHT, etc..

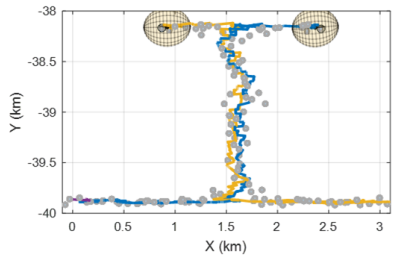
tracks

Visualization & Metrics

A rich library of tracking algorithms







Filters

- Alpha Beta filter
- Kalman filters
  - Linear, EKF, UKF, CKF, MSCEKF
- Particle filter
- Multiple models
  - GSF, IMM

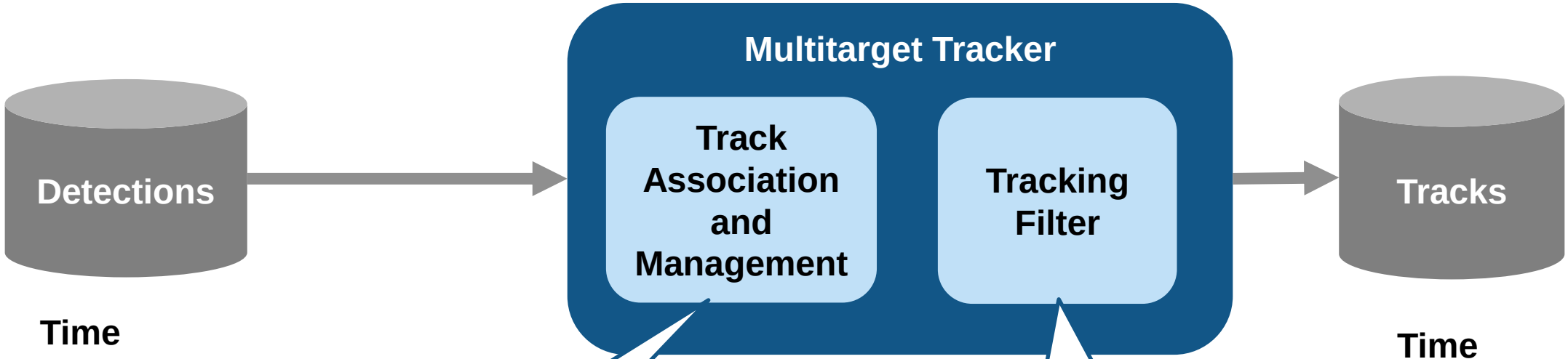
Data Association

- 2D assignment
- S-D assignment
- K-best assignmemt

Trackers

- GNN
- MHT (track-oriented)
- Trackers components
  - History and score logic
  - etc……

Difference Between Multitarget Tracker and Kalman Filter



Time
Measurement
Measurement Noise
…

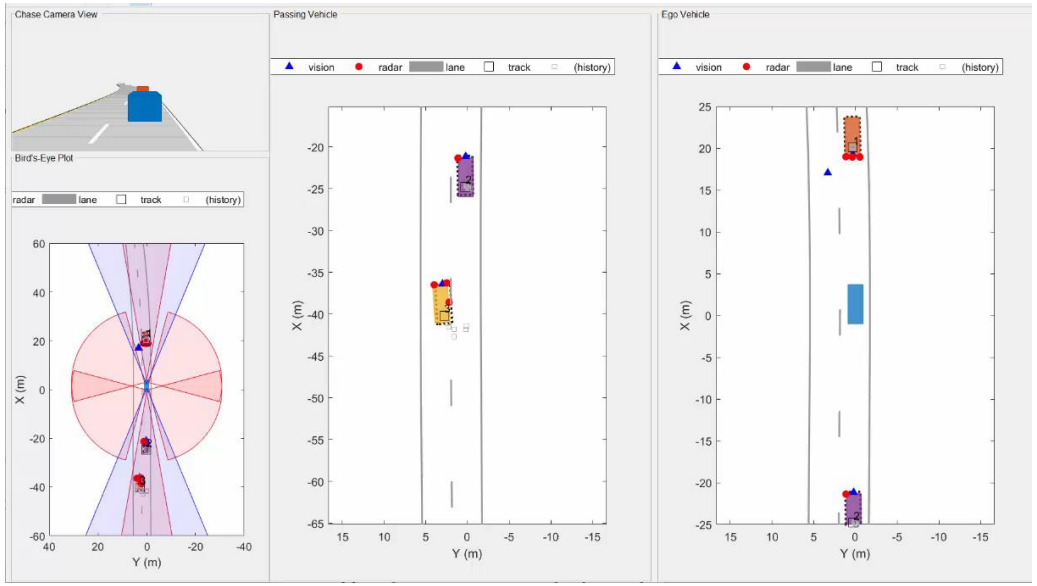
- Assigns detections to tracks
- Creates new tracks
- Updates existing tracks
- Removes old tracks

- Predicts and updates state of track
- Linear, extended, and unscented Kalman filters
- Particle, Gaussian-sum, IMM filters

Time
State
State Covariance
Track ID
Age
Is Confirmed
Is Coasted
…

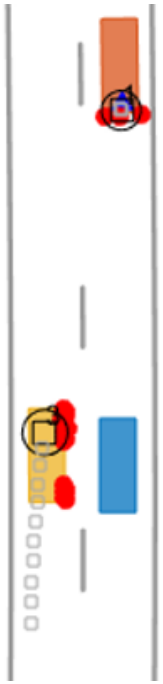
Tracking extended objects

- **High resolution sensors (radar, lidar)**
- **Close-range targets**

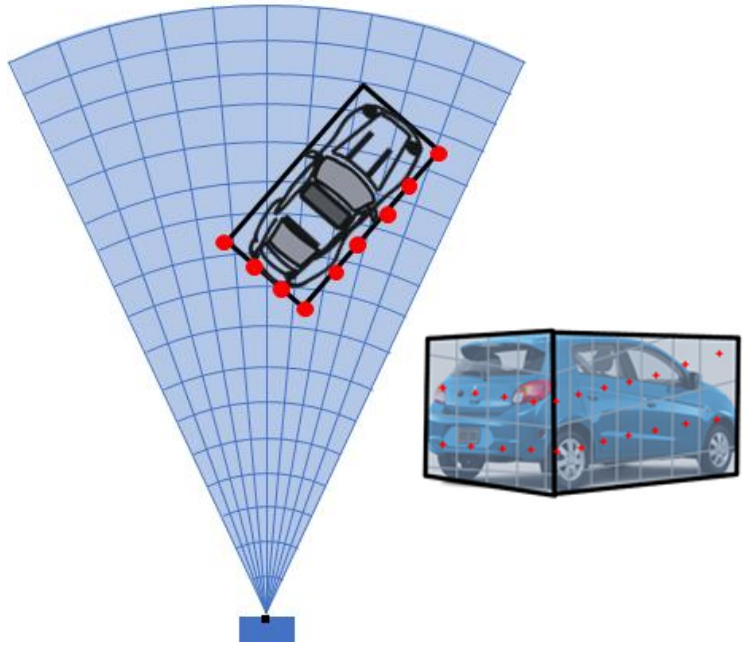


Point object vs. Extended object

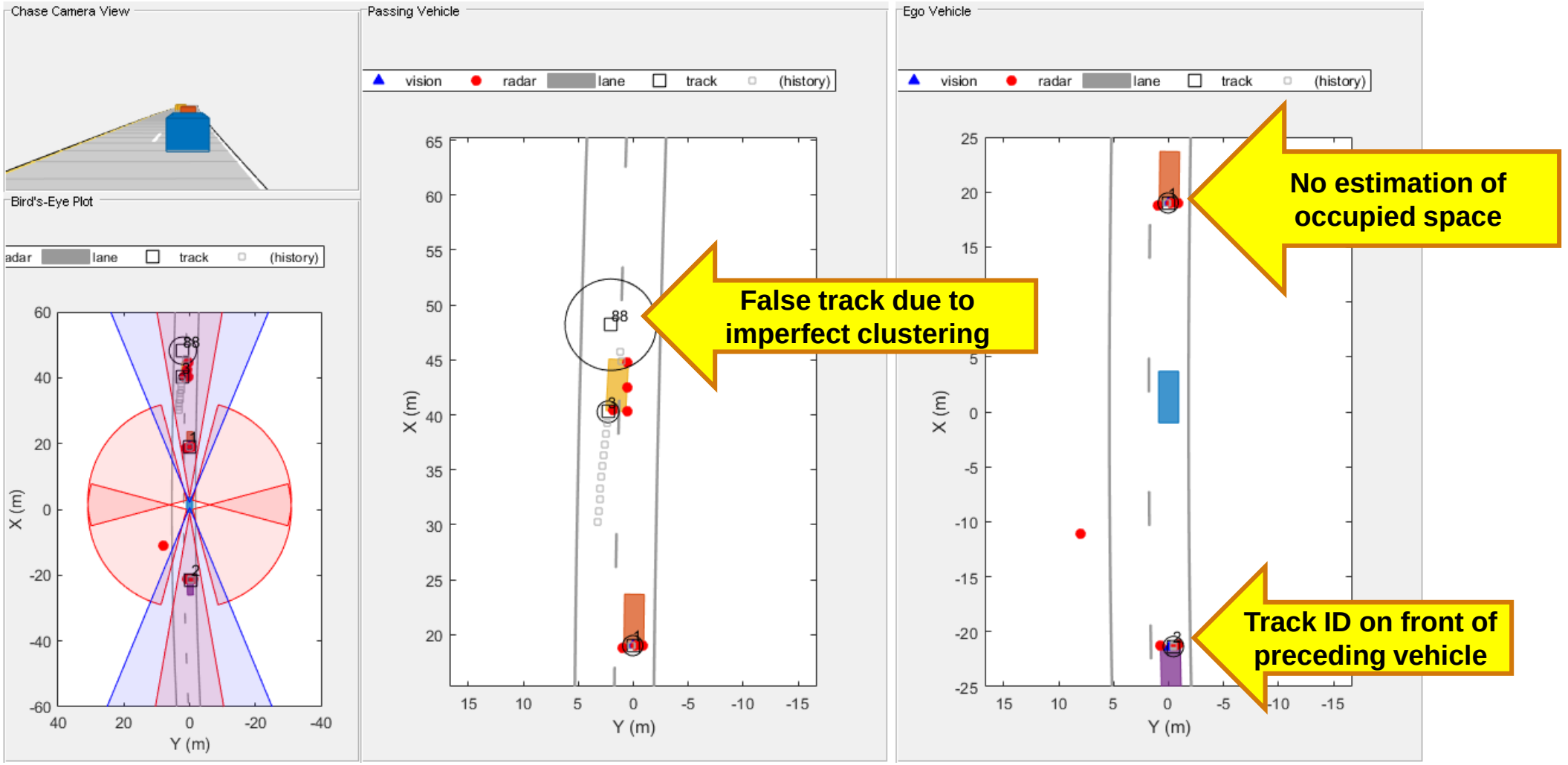
- **Point object**
  - Distant object represented as a single point
  - One detection per object per scan



- **Extended object**
  - High resolution sensors generate multiple detections per object per scan



Tracking Extended Objects with Point Trackers

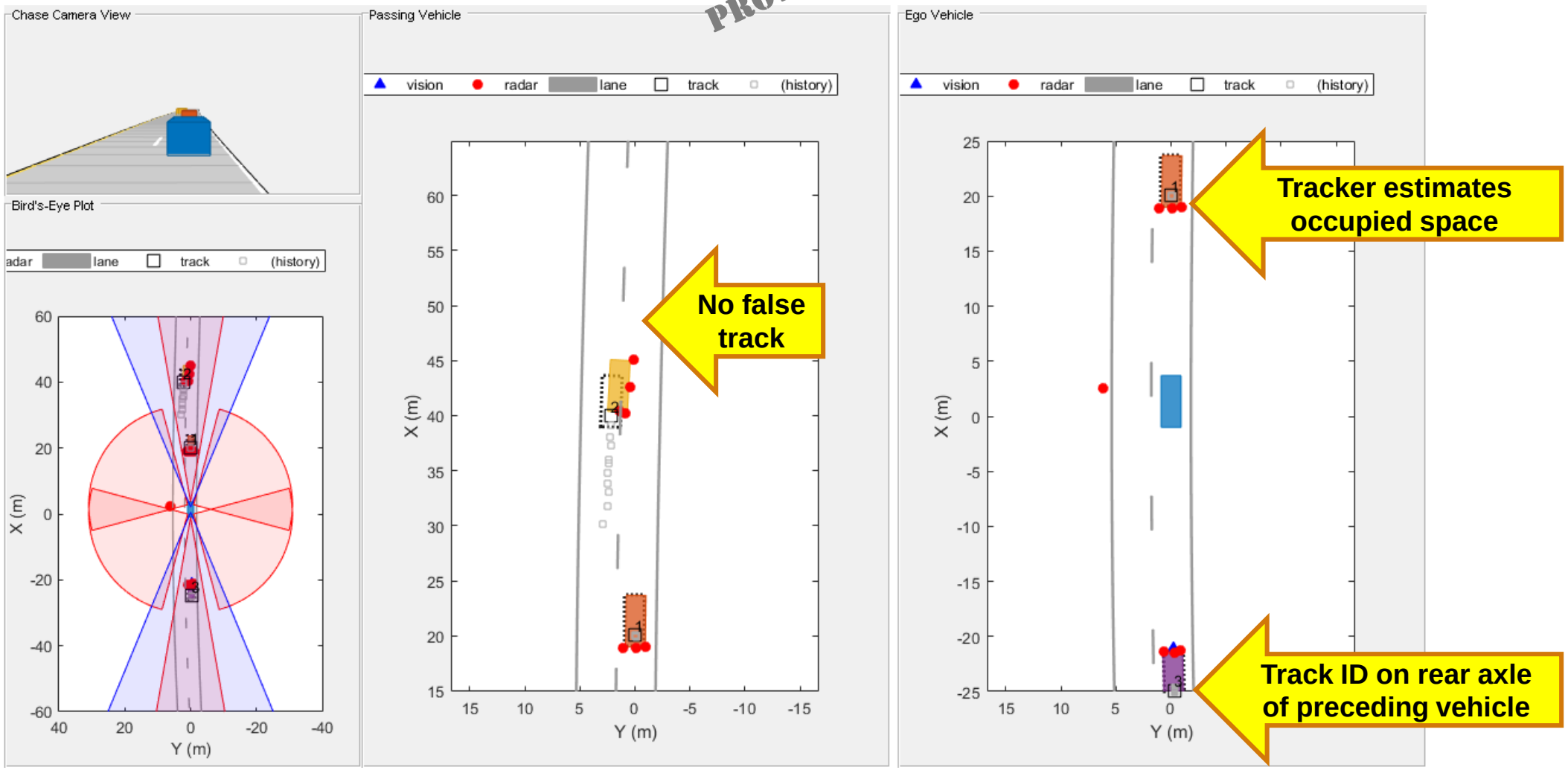


Point cloud changing with aspect angle -> challenging clustering problem

Tracking Extended Objects: Estimate Position, Velocity, and Shape

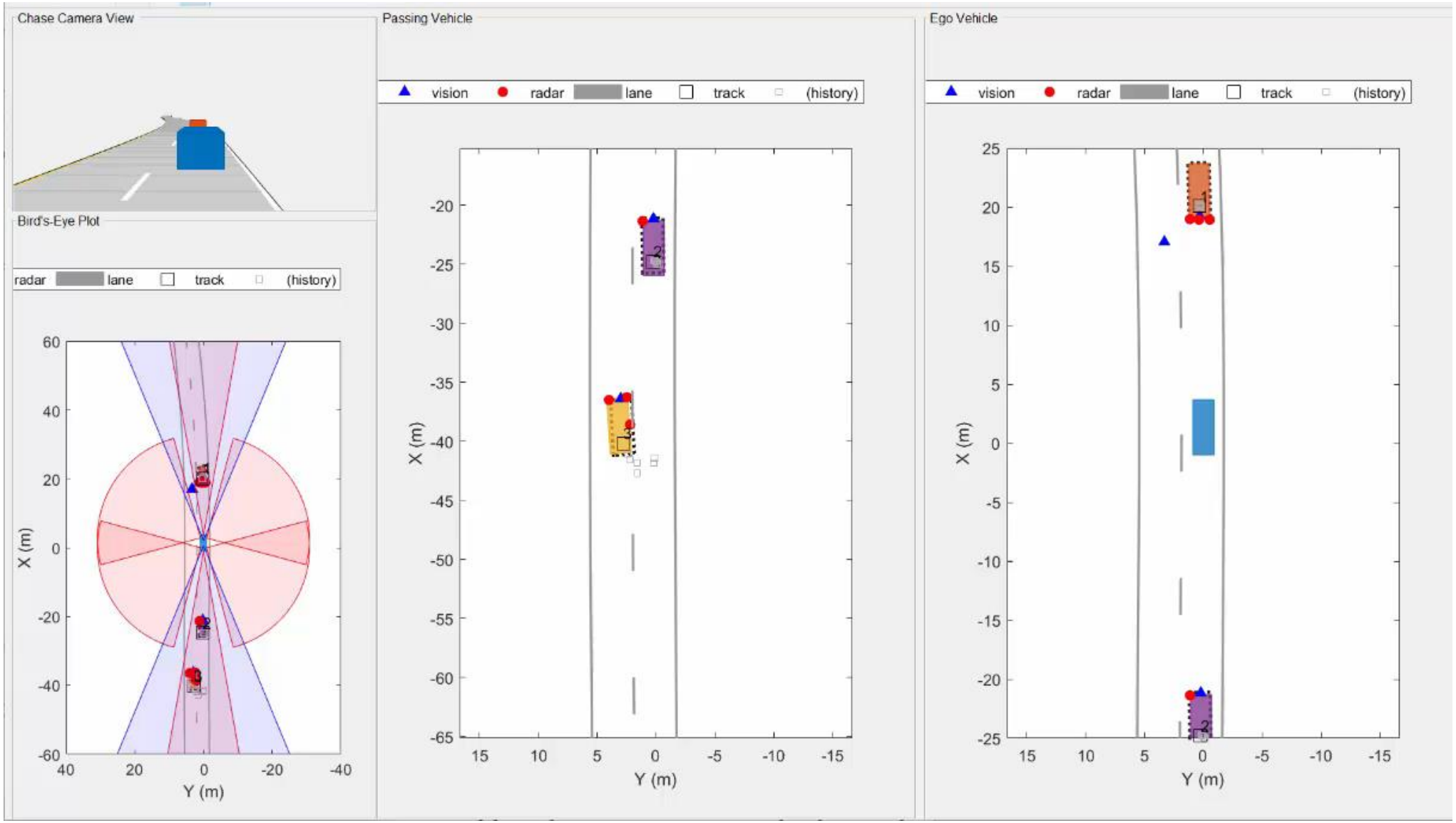
MathWorks R2018b

PROTOTYPE



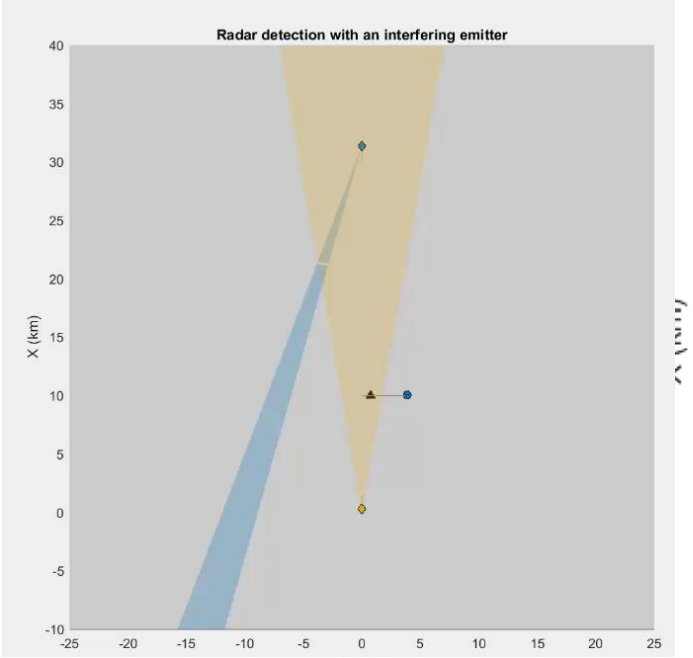
No clustering needed, Consistent tracks

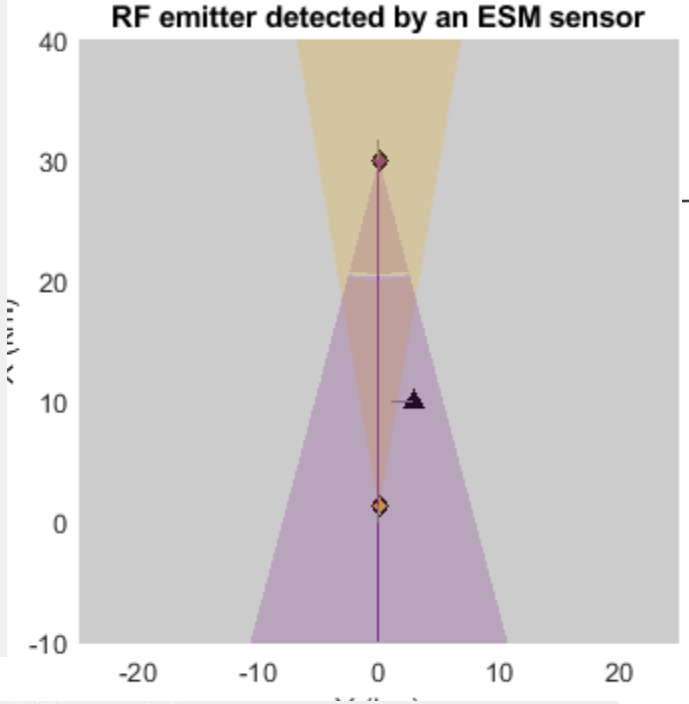
Tracking Extended Objects: Estimate Position, Velocity, and Shape

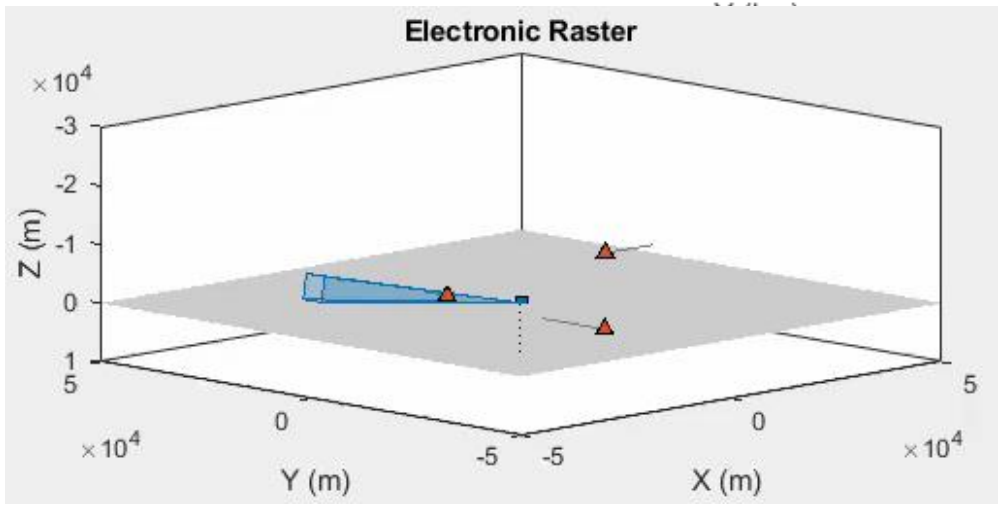


Sensor Models Beyond Automated Driving

Sensor Models Beyond Automated Deriving







Sensor Models

IMU, GPS, RADAR, ESM, and EO/IR

Model various sensors, including: IMU (accelerometer, gyroscope, magnetometer), GPS receivers, radar, sonar, ar
and channels as separate objects.

Functions

- GPS
- IMU
- Infrared
- INS
- Radar
- Sonar

Getting Started