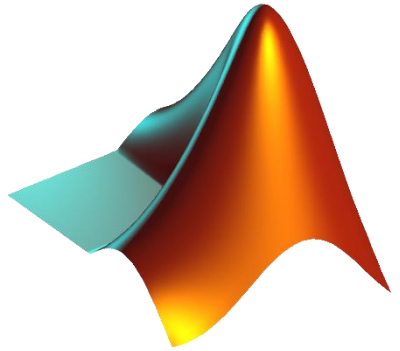
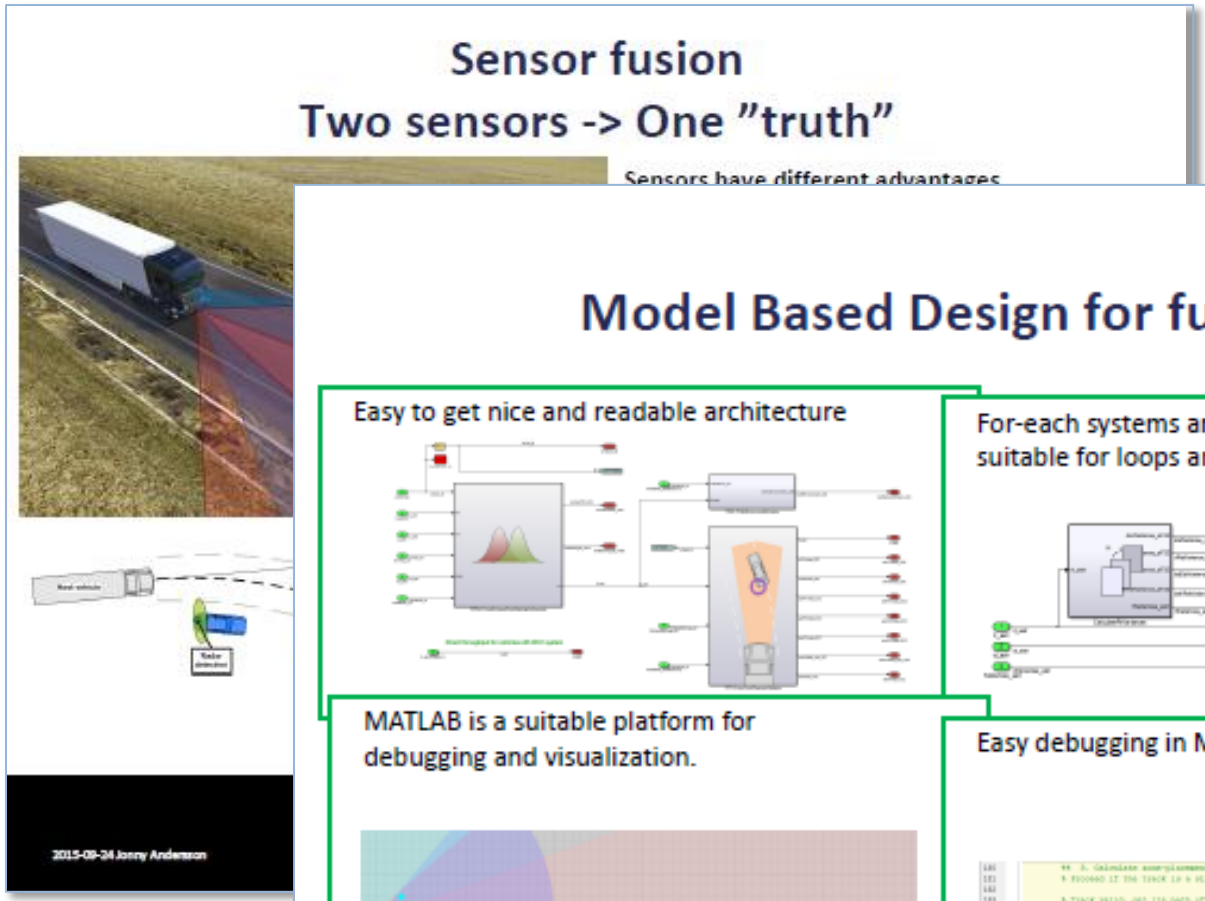
Automated Driving:
Design and Verify Perception Systems

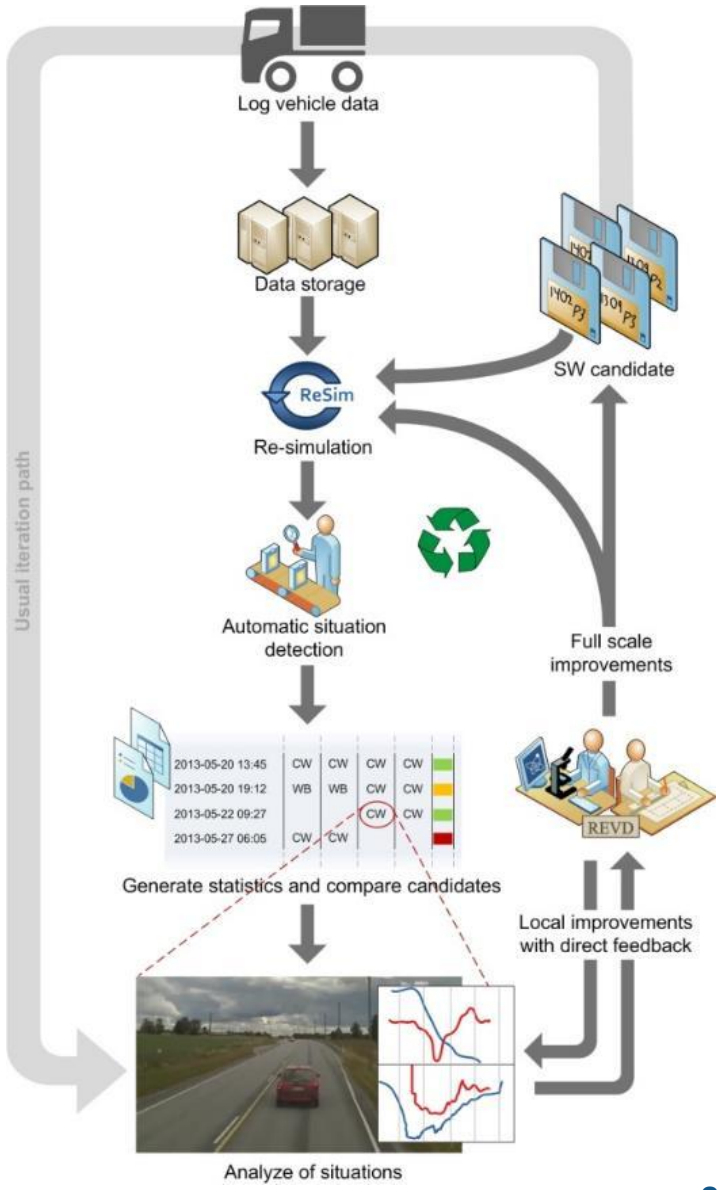


Abhijit Bhattacharjee
Senior Application Engineer, MathWorks Inc.

Scania: Model-Based Design for AEB Sensor Fusion Development

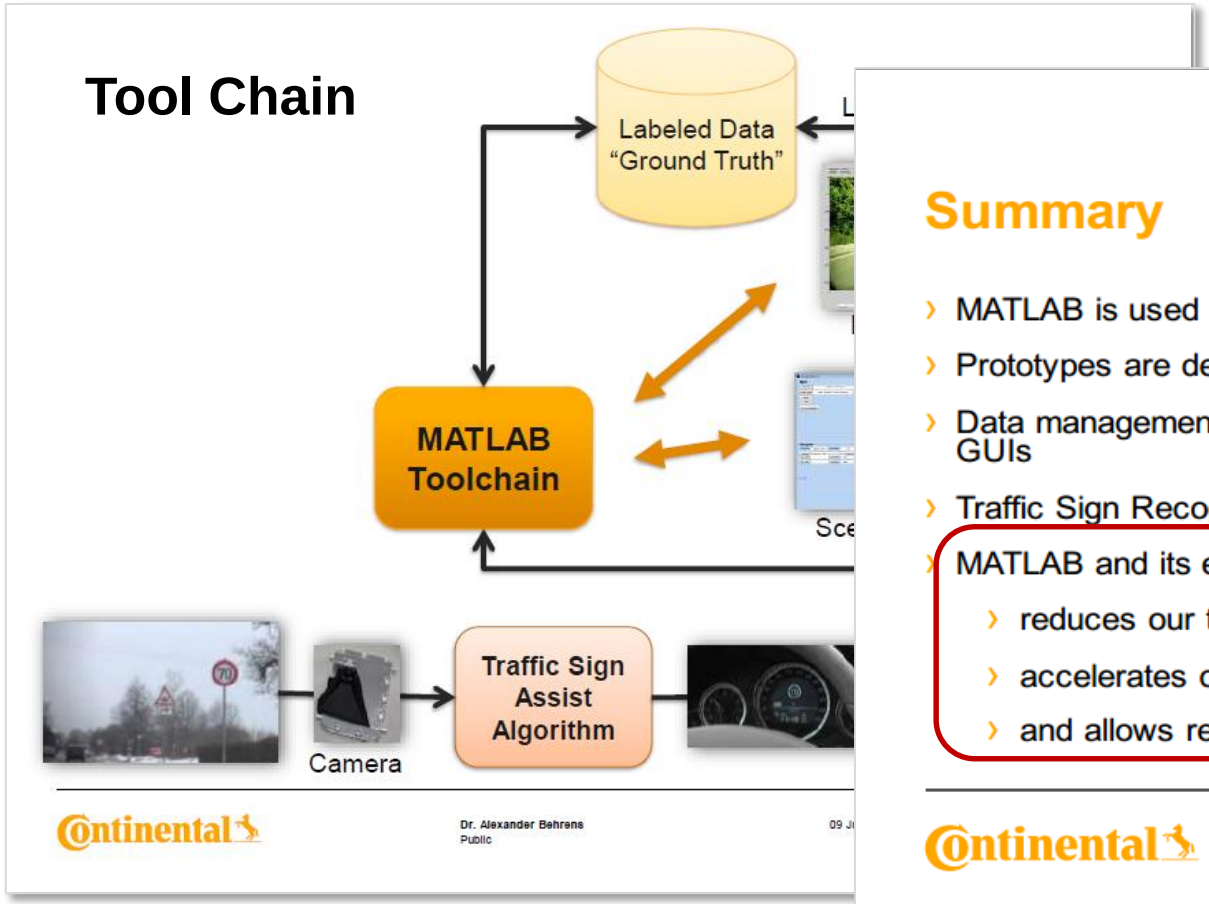
Scania: Model-Based Design for AEB Sensor Fusion Development

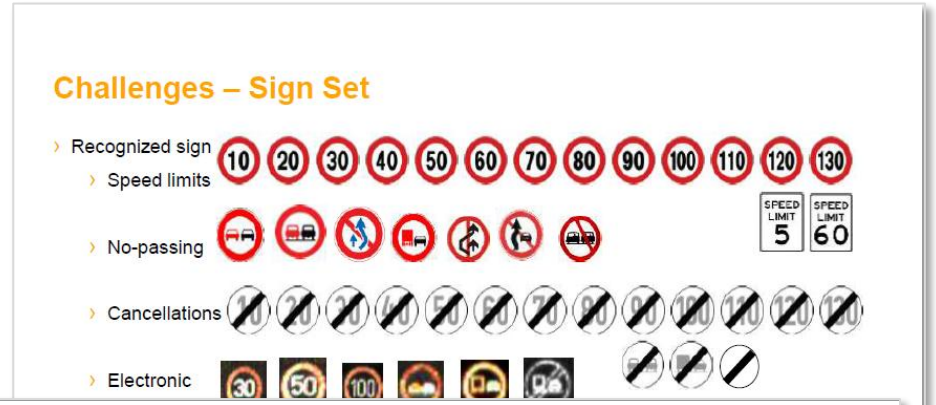




Continental

Ground Truth Labeling and Algorithm Training for Traffic Sign Recognition Systems





Summary

- MATLAB is used in daily work for development and evaluation of driver assistance functions
- Prototypes are designed with MATLAB for predevelopment and proof of concept
- Data management, evaluation, and interactive analysis are supported by MATLAB tools and GUIs
- Traffic Sign Recognition and other functions make high use of MATLAB tools
- MATLAB and its established features
  - reduces our tool development efforts,
  - accelerates our simulation cost,
  - and allows reliable, repeatable and accurate parameter optimizations

Continental | Dr. Alexander Behrens, Public | 24 September 2015 | 18

Source: Traffic sign recognition for driver assistance systems (MAC Germany 2015)

BMW Autonomous Driving R&D

ROSCon 2015
Hamburg, Germany

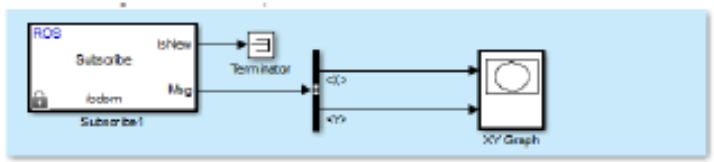
AUTOMATED DRIVING WITH ROS AT BM

MICHAEL AEBERHARD, THOMAS KÜHBECK, BERNHARD SEIDL, MARTIN F
JULIAN THOMAS, OLIVER SCHEICKL.

BMW GROUP

USING MATLAB/SIMULINK WITH ROS.

- MathWorks released the Robotics System Toolbox this year for ROS integration with Matlab/Simulink.
- Easily read and analzye data from ROS Bags → useful for evaluating the system.
- Some of our software is implemented as a Simulink model.
  - Use the Toolbox to easily integrated this software into the ROS eco-system:

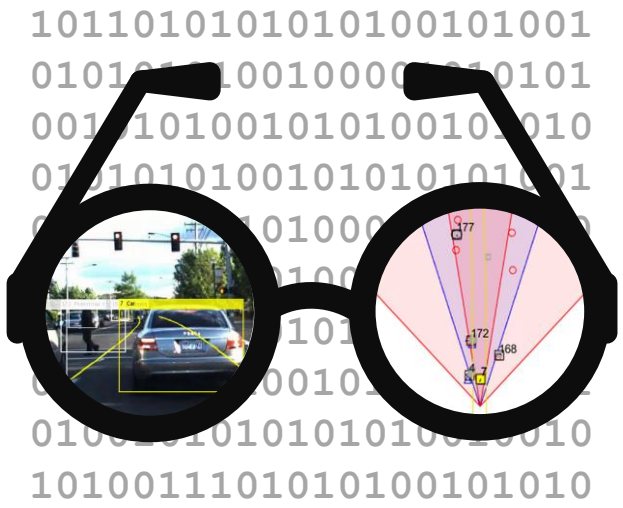


http://www.mathworks.com/products/robotics/

Michael Aeberhard, BMW Group Research and Technology

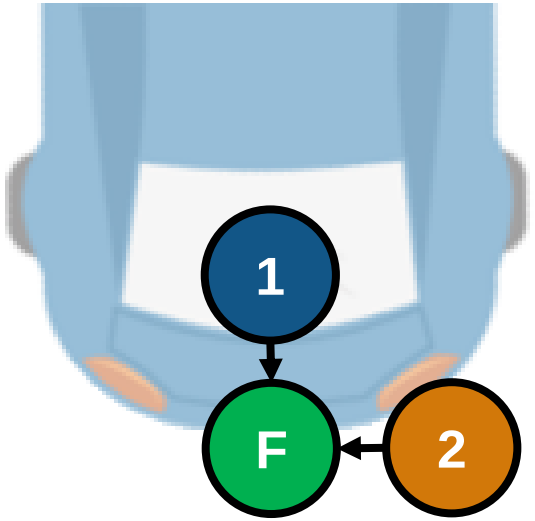
Page 15

Common Questions from Automated Driving Engineers



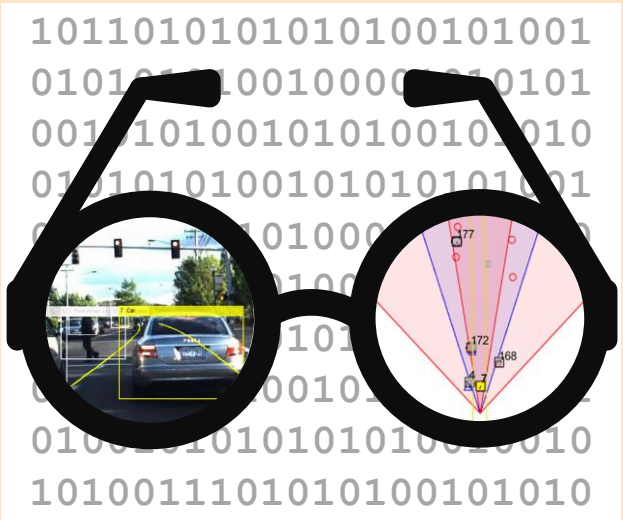
How can I visualize **vehicle data?**

How can I design and verify **perception algorithms?**



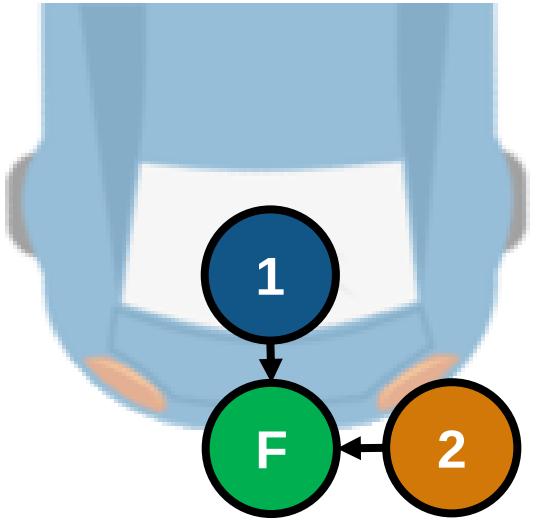
How can I design and verify **sensor fusion?**

Common Questions from Automated Driving Engineers



How can I visualize **vehicle data?**

How can I design and verify **perception algorithms?**



How can I design and verify **sensor fusion?**

Examples of Automated Driving Sensors

Camera

Radar-based object detector

Vision-based object detector

Lidar

Lane detector

Inertial measurement unit

Examples of automated driving sensor data

Camera (640 x 480 x 3)

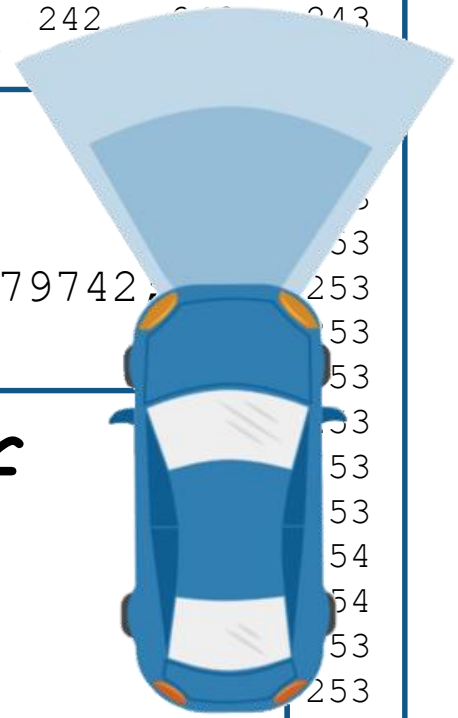
239 239 237 238 241 241 241 242
252 252 251 252 252 253 253

Vision Detector

```
SensorID       = 1;
Timestamp      = 1461634696379742;
NumDetections = 6;
Dete
  Tr
  Cl
  Po
  Ve
  Si
Detec
  Tr
  Cl
  Po
  Ve
  Si
```

Lane Detector

```
Left
  IsValid:          1
  Confidence:       3
  BoundaryType:     3
  Offset:           1.68
  HeadingAngle:     0.002
  Curvature:        0.000
Right
  IsValid:          1
```



Radar Detector

```
SensorID       = 2;
Timestamp      = 1461634696407521;
NumDetections = 23;
Detection
  TrackID
  TrackSt
  Positio
  Velocit
  Amplitu
Detection
  TrackID
  TrackSt
```

Lidar (47197 x 3)

-12.2911	1.4790	-0.59
-14.8852	1.7755	-0.64
-18.8020	2.2231	-0.73
-25.7033	3.0119	-0.92
-0.0632	0.0815	1.25
-0.0978	0.0855	1.25
-0.2814	0.1064	1.25

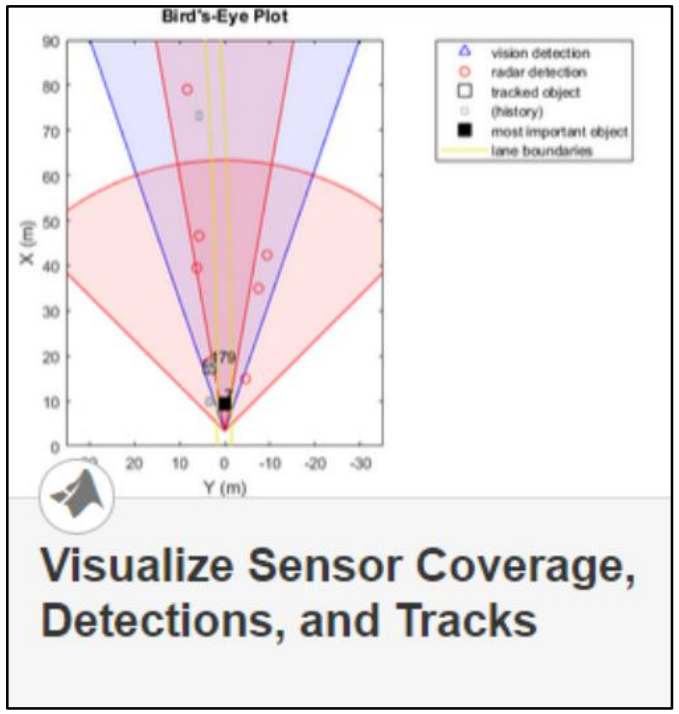
Inertial Measurement Unit

```
Timestamp:   1461634696379742
Velocity:    9.2795
YawRate:     0.0040
```

Visualize sensor data

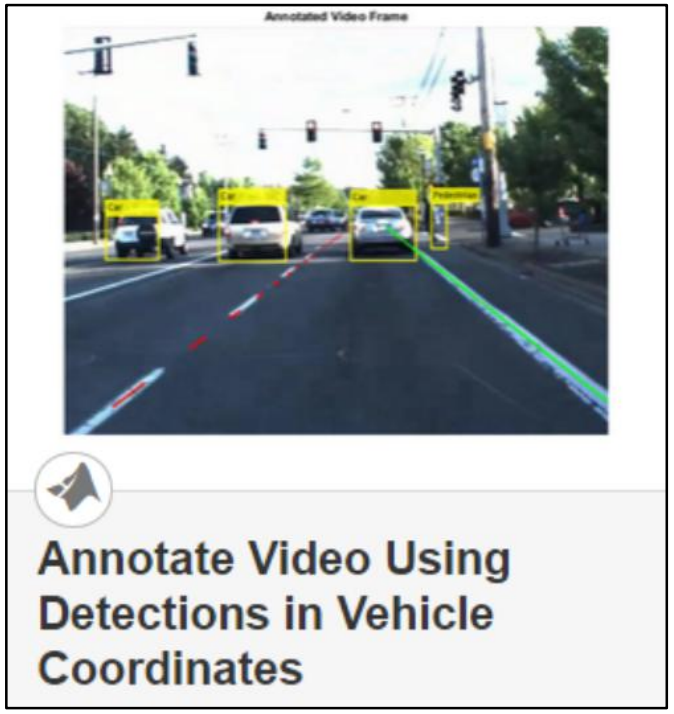
Learn More About Visualizing Vehicle Data

Explore Examples in Automated Driving System Toolbox



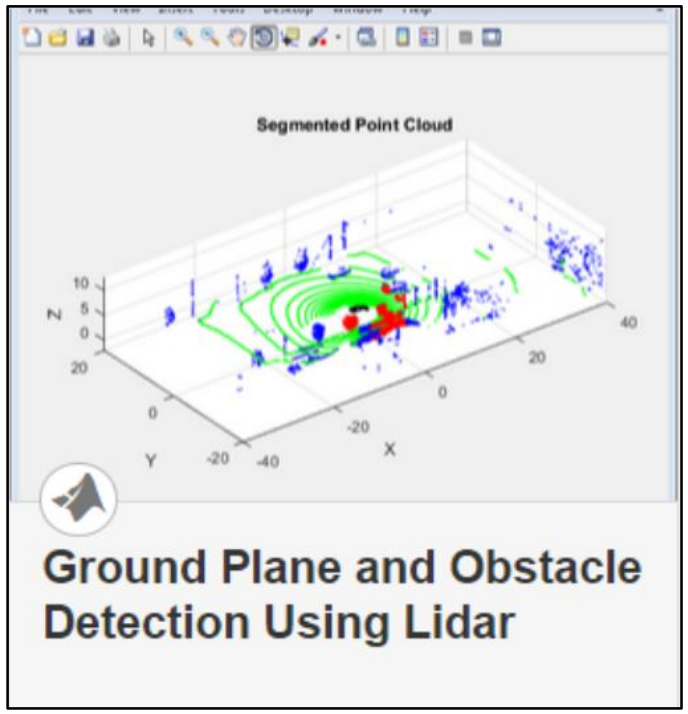
Visualize Sensor Coverage, Detections, and Tracks

- **Plot object detectors in vehicle coordinates**
  - Vision & radar detector
  - Lane detectors
  - Detector coverage areas



Annotate Video Using Detections in Vehicle Coordinates

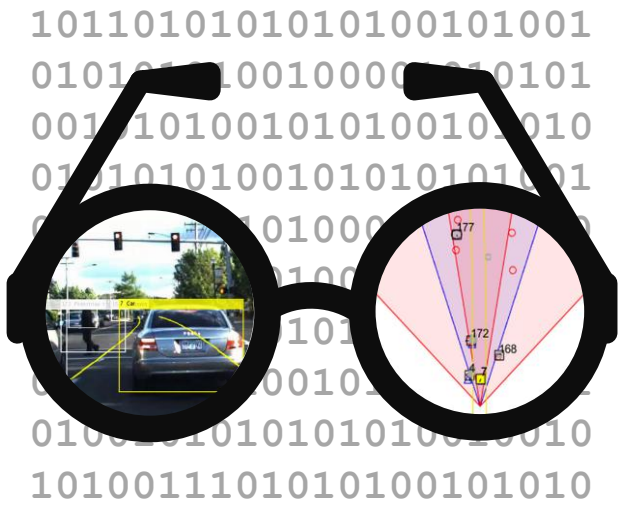
- **Transform between vehicle and image coordinates**



Ground Plane and Obstacle Detection Using Lidar

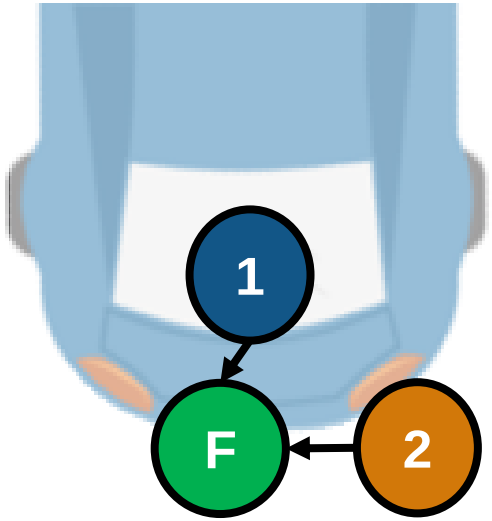
- **Plot lidar point cloud**

Common Questions from Automated Driving Engineers



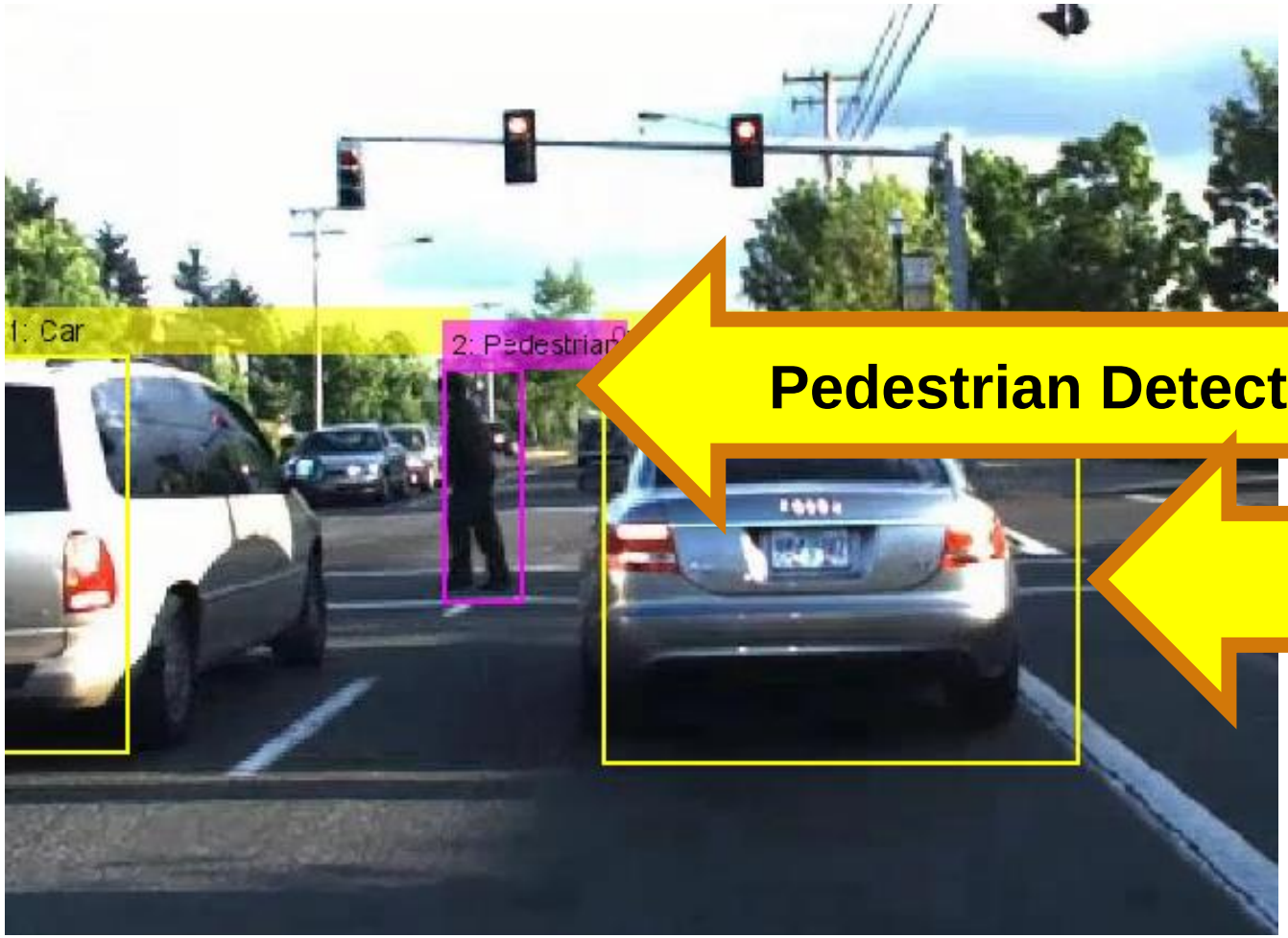
How can I visualize **vehicle data?**

How can I design and verify **perception algorithms?**



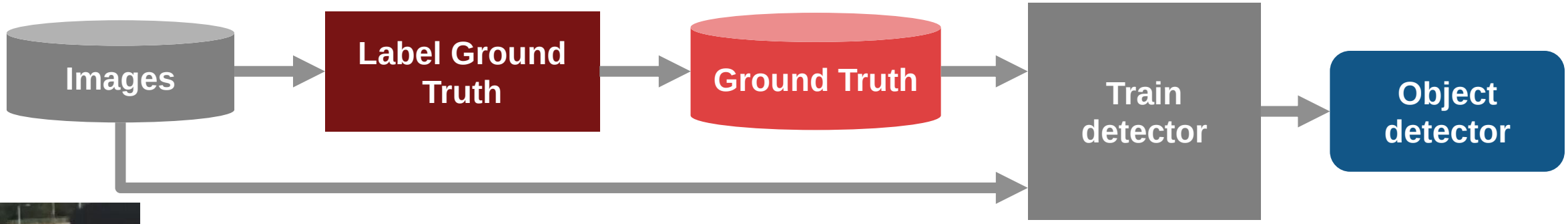
How can I design and verify **sensor fusion?**

Examples of Perception Algorithms



Object Detection: Locate and classify object in image or video.

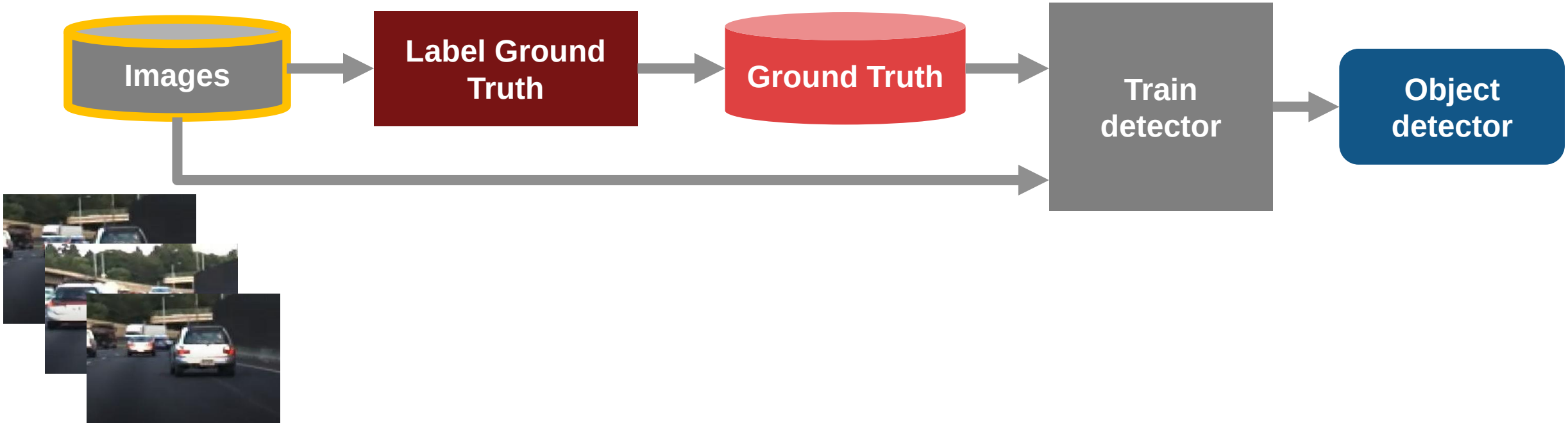
Workflow to Train Detectors



1 imageFilename	2 StopSign	3 chimney	4 car
...toolbox\vision\visiondata\stopSignImages\image001.jpg'	[848,281,5...	[]	[415,39...
...toolbox\vision\visiondata\stopSignImages\image002.jpg'	[385,501,1...	[]	[293,61...
...toolbox\vision\visiondata\stopSignImages\image003.jpg'	[459,368,6...	[]	[]
...toolbox\vision\visiondata\stopSignImages\image004.jpg'	[935,387,6...	[]	*3x4 do...*
...toolbox\vision\visiondata\stopSignImages\image005.jpg'	[960,385,8...	[431,83,1...	*3x4 do...*
...toolbox\vision\visiondata\stopSignImages\image006.jpg'	[1021,337,...	[]	[]
...toolbox\vision\visiondata\stopSignImages\image007.jpg'	[]	[]	[]

Machine Learning
Deep Learning

MATLAB Tools to Train Detectors

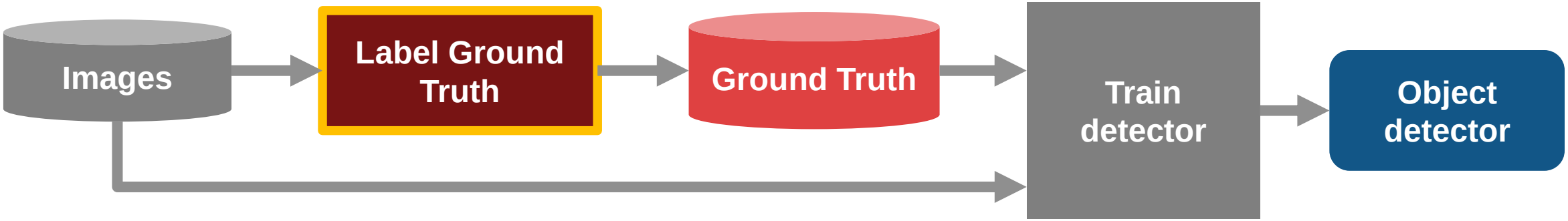



```
imageDS = imageDatastore(dir)
```

Easily manage large sets of images

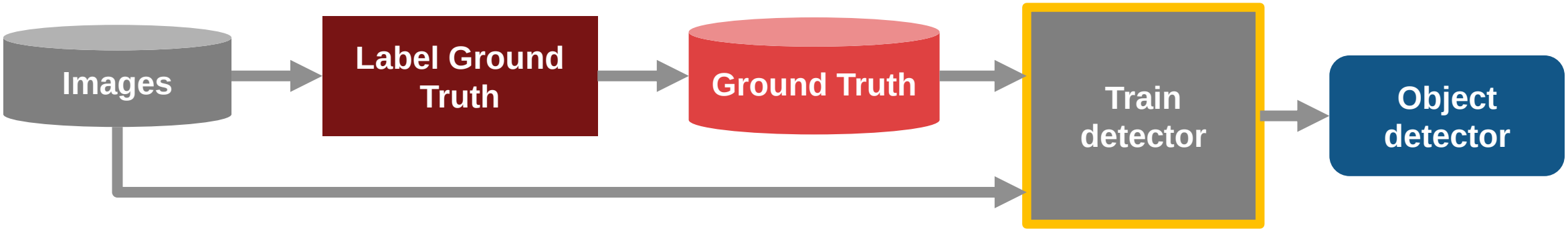
- Single line of code to access images
- Operates on disk, database, big-data file system

MATLAB Tools to Train Detectors



Automate Labeling of Ground Truth

MATLAB Tools to Train Detectors



Use same ground truth to try different detectors.

Machine Learning
- Cascade Object Detector
- Aggregate Channel Features

Deep Learning
- R-CNN
- Fast R-CNN
- Faster R-CNN

Single Line of Code to Train Each Detector

E.g.
`trainCascadeObjectDetector`
`trainFasterRCNNObjectDetector`

Designing Perception Systems
Computer Vision Algorithms for Automated Driving

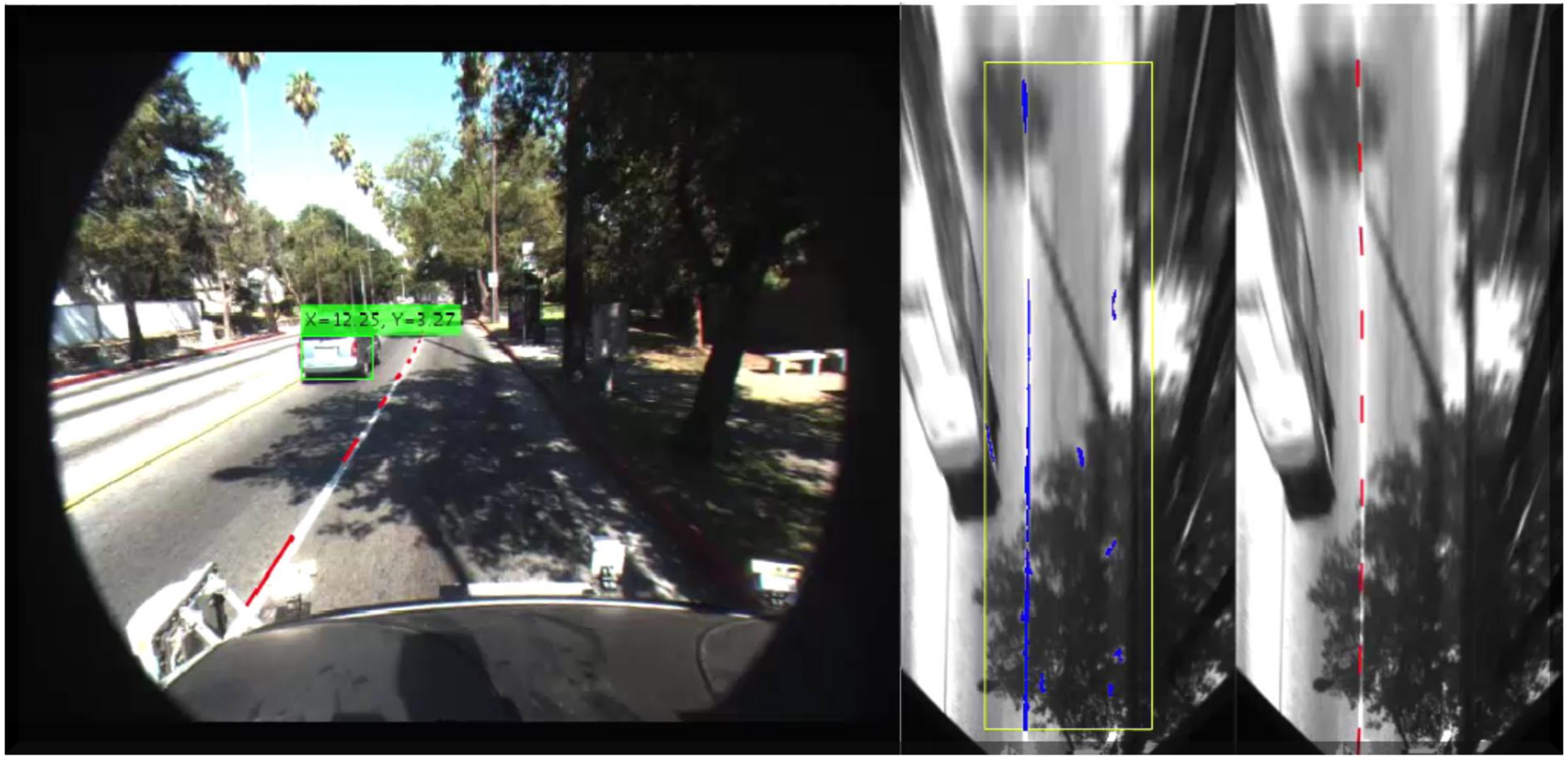
Vehicle Detection

Deep learning and ACF based (pre-trained)

Pedestrian Detection

ACF and HOG/SVM based (pre-trained)

Designing Perception Systems
Additional Computer Vision Algorithms for Automated Driving



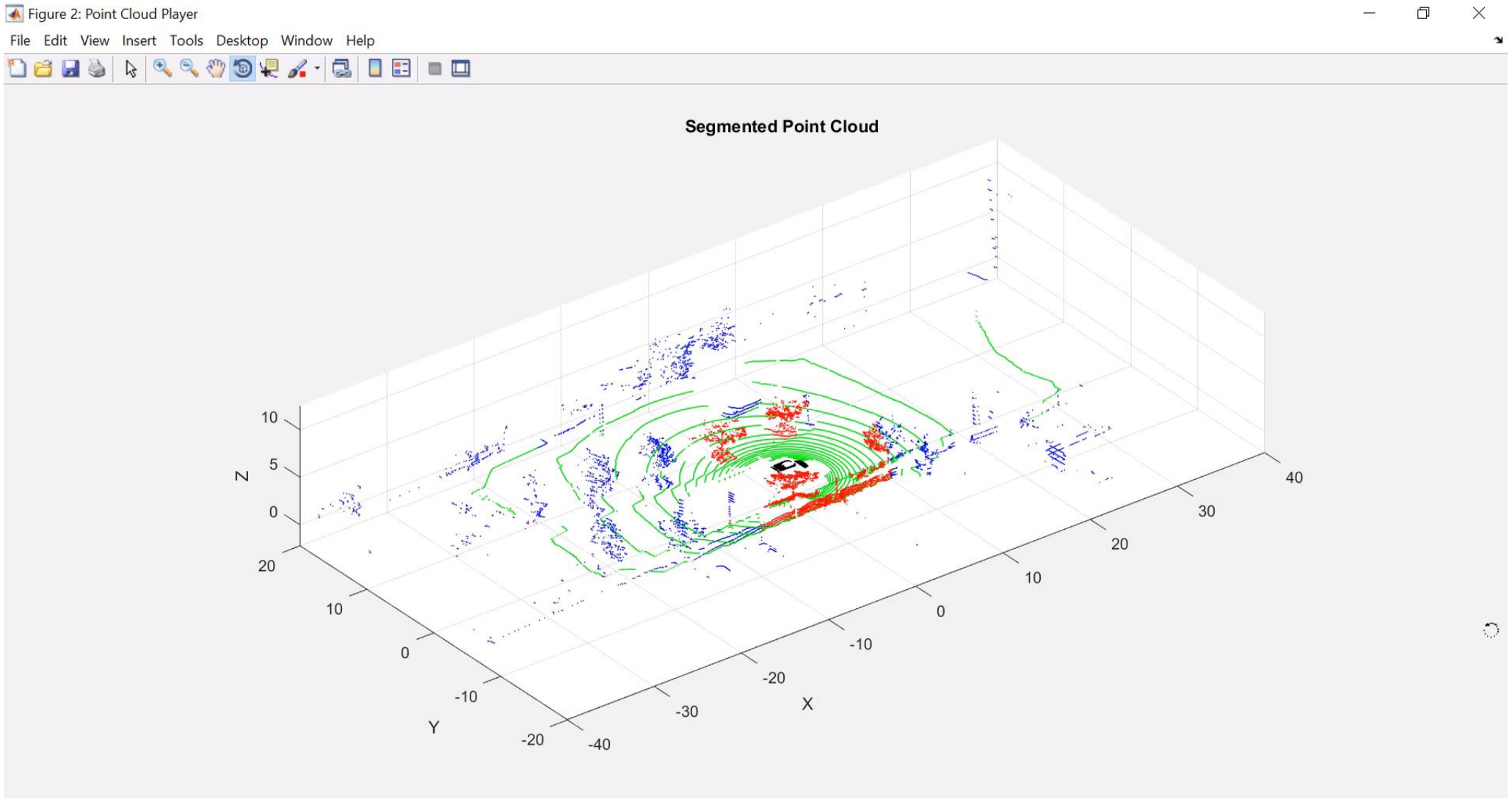
Vehicle detection

with distance estimation
using mono-camera

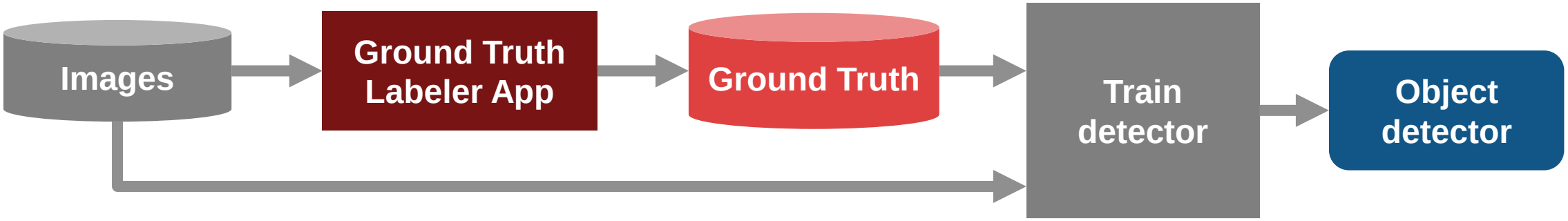
Lane Detection and Classification

- RANSAC-based lane boundary fitting
- Lane boundary visualization

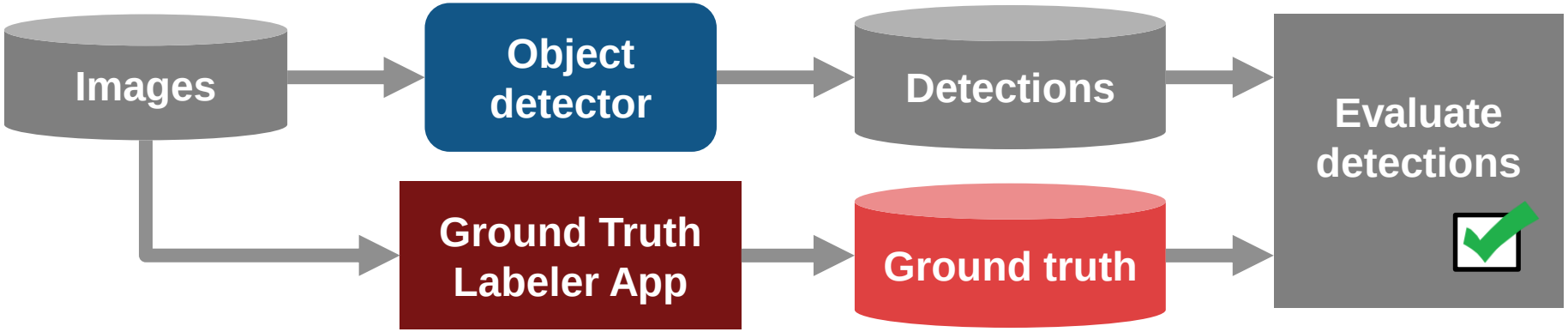
Designing LiDAR Processing Algorithms



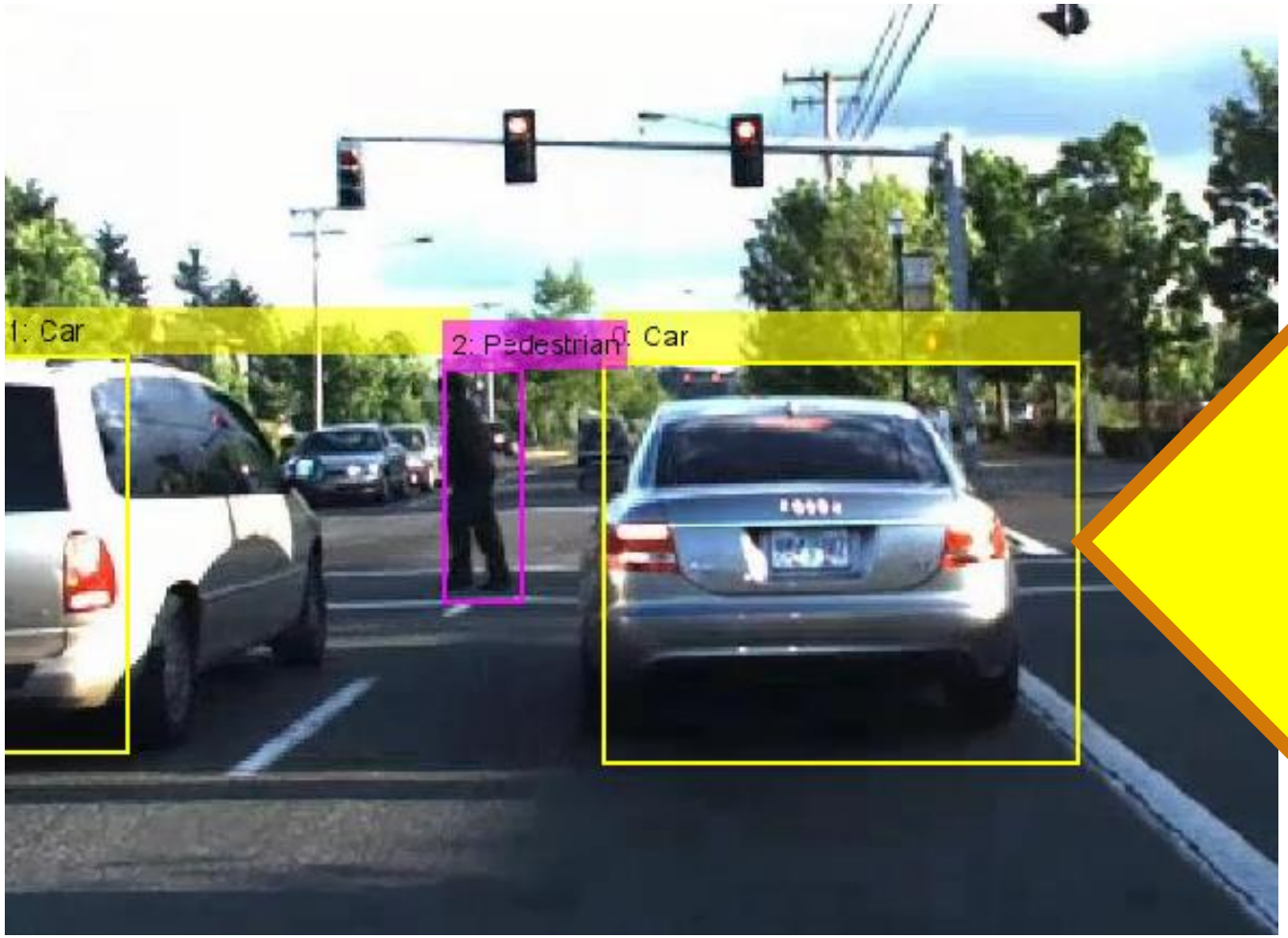
Ground truth labeling to train detectors



Ground truth labeling to evaluate detectors



Example of Vision System Detection

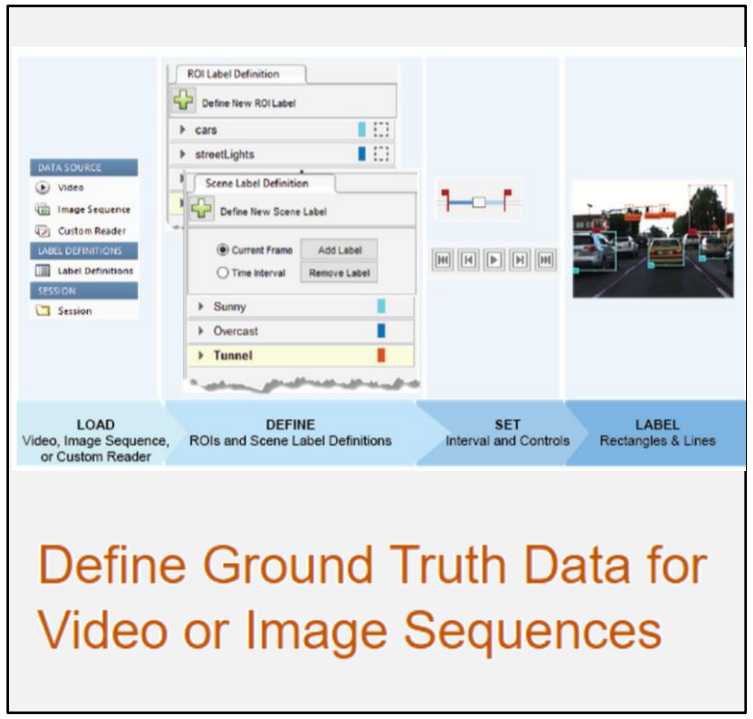


How can I verify this detection is correct?

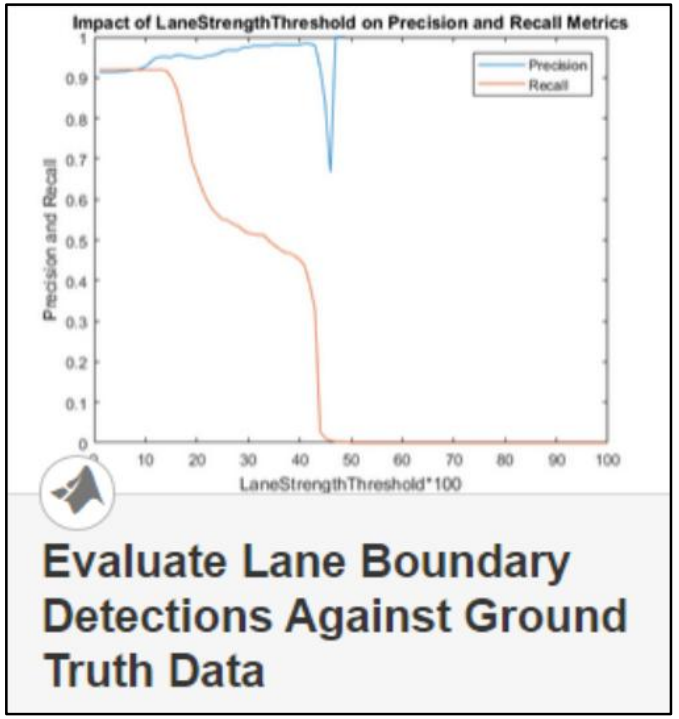
Evaluate detections against ground truth

Learn More About Verifying Perception Algorithms

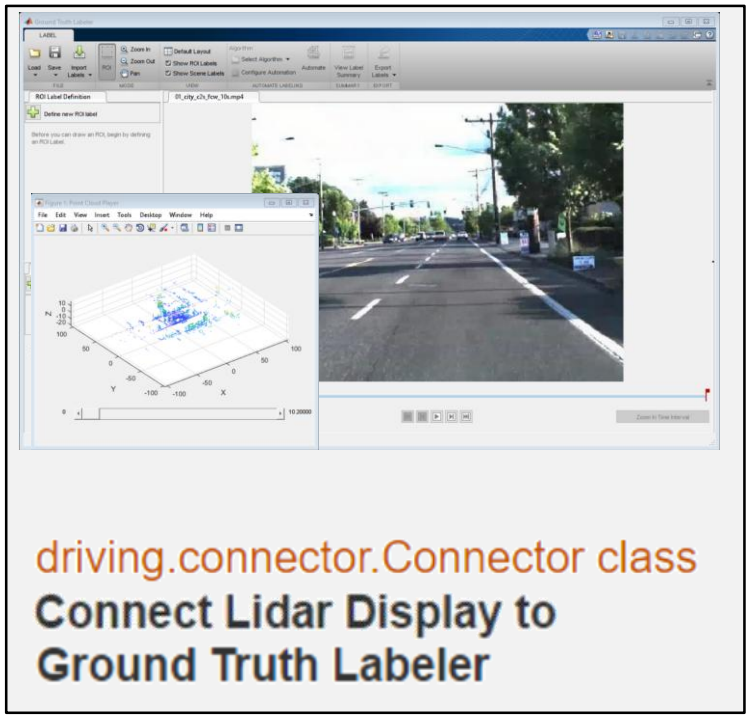
Explore Examples in Automated Driving System Toolbox



- **Label detections** with Ground Truth Labeler App

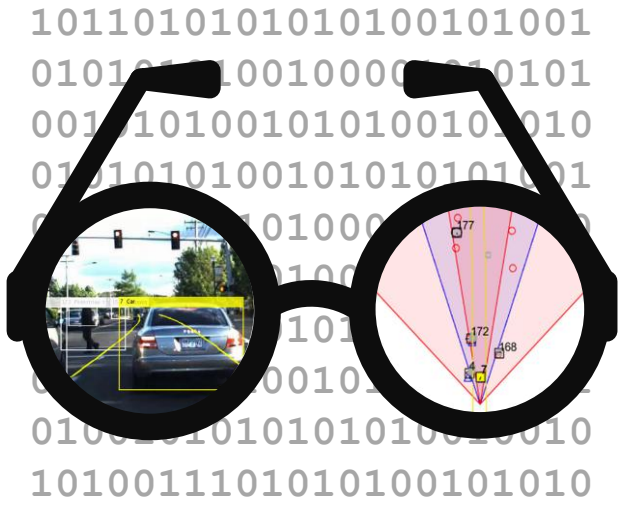


- **Evaluate detections** against ground truth



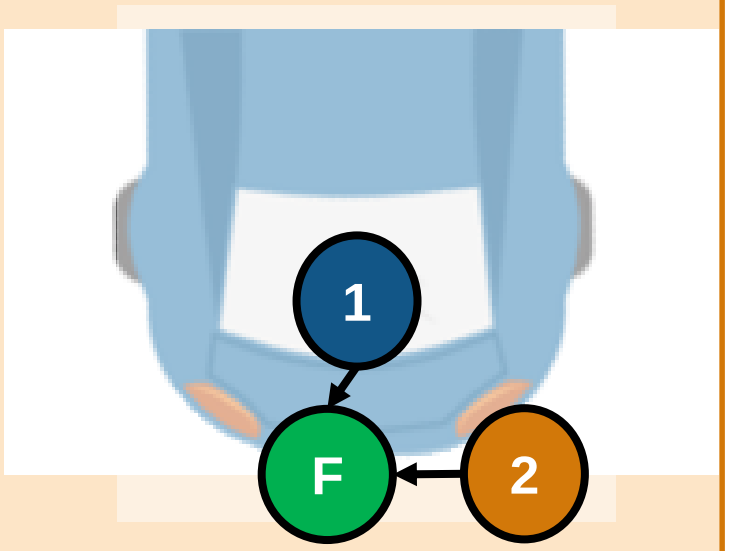
- **Extend connectivity** of Ground Truth Labeler App

Common Questions from Automated Driving Engineers



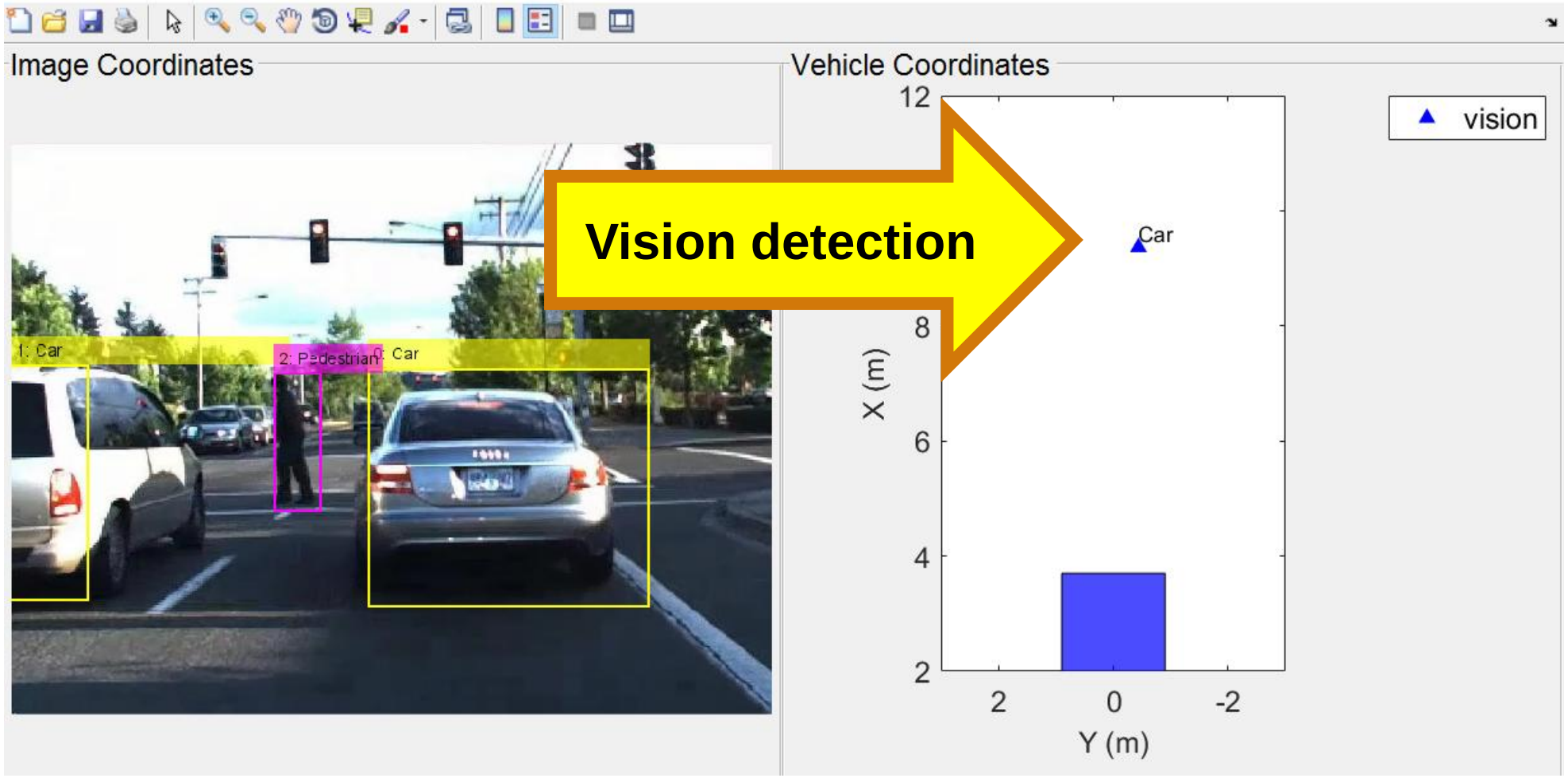
How can I visualize **vehicle data?**

How can I design and verify **perception algorithms?**

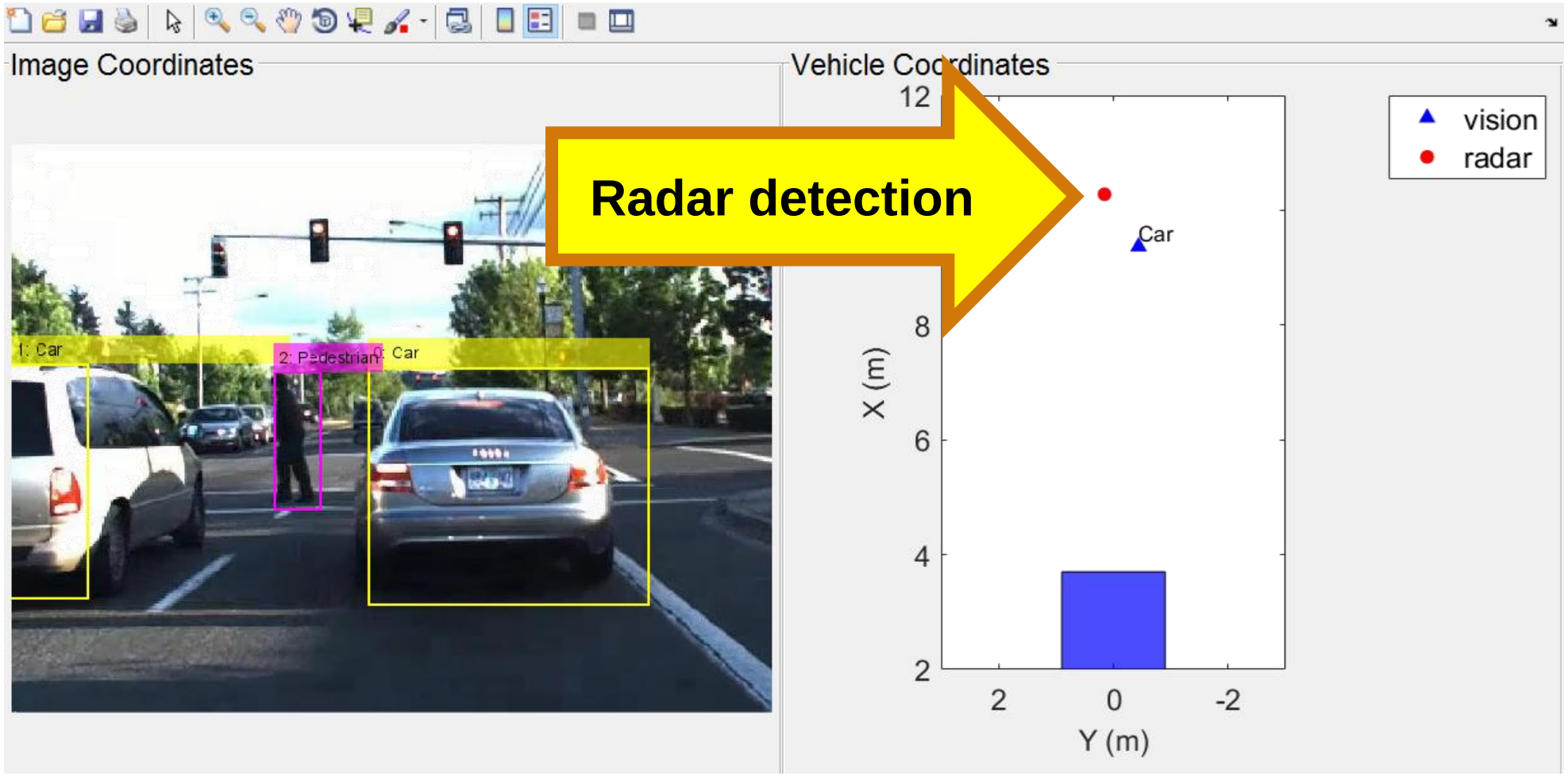


How can I design and verify **sensor fusion?**

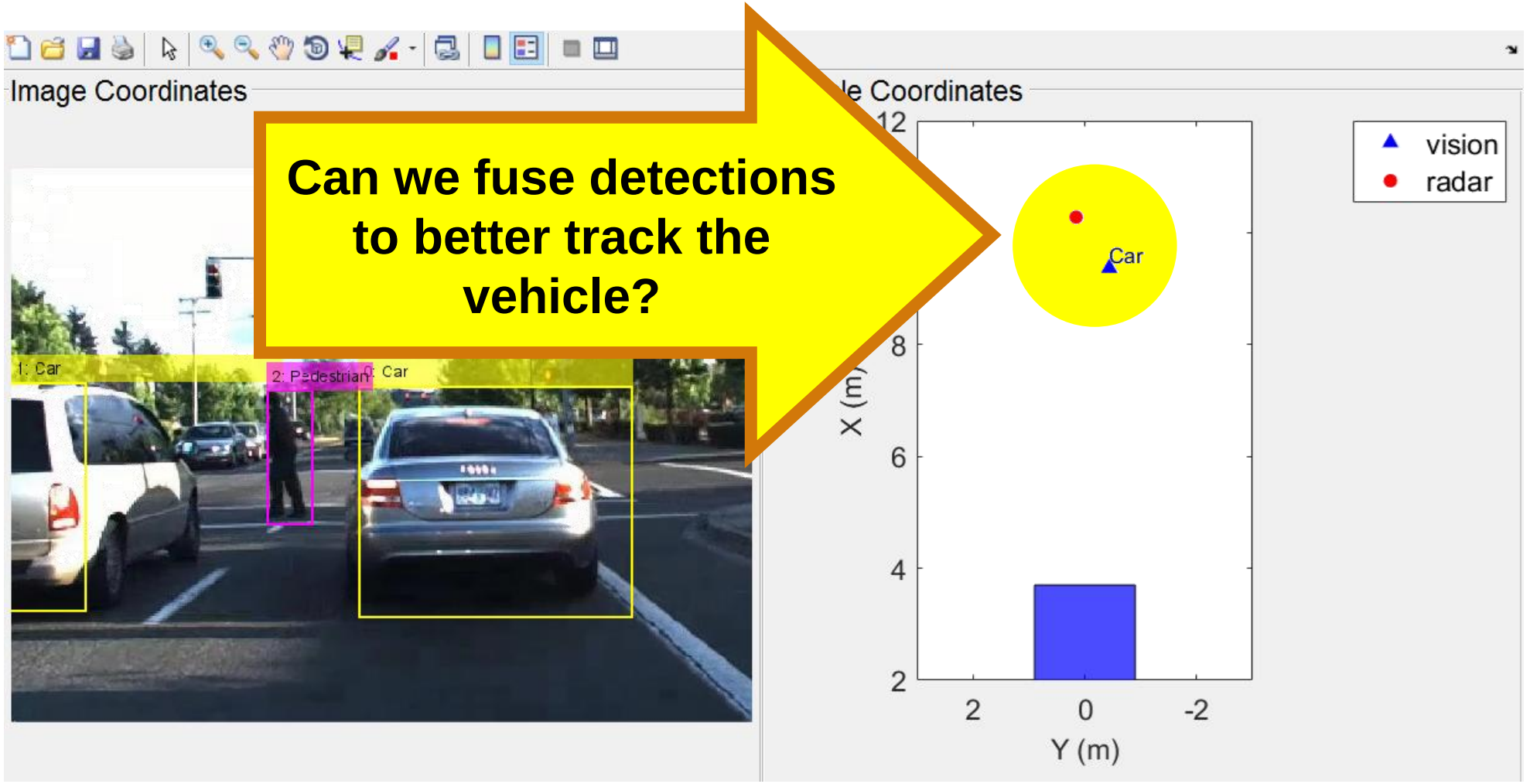
Example of radar and vision detections of a vehicle



Example of radar and vision detections of a vehicle

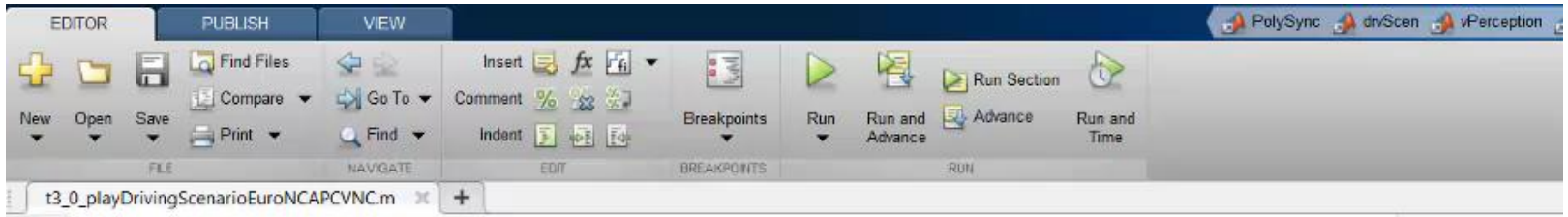


Example of radar and vision detections of a vehicle



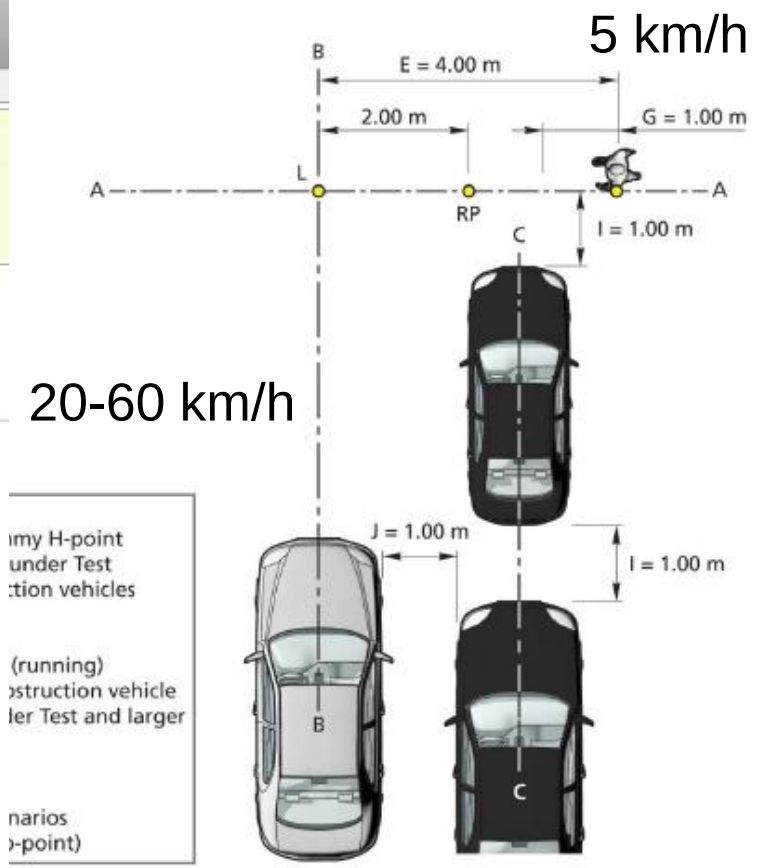
Design
multi-object tracker

Euro NCAP TEST PROTOCOL Scenario Generation



t3_0_playDrivingScenarioEuroNCAPCVNC.m

```
28    %% Create a new scenario
29 -  s = drivingScenario;
30 -  s.SampleTime = 0.05;
31
32    %% Create road
33 -  RoadCenters = [0 0 0; 50 0 0];
34 -  road(s, RoadCenters, 10);
35
36    %% Add actors
37    % --- moving ego vehicle towards a child pedestrian crossing
38 -  egoCar = vehicle(s,'Position',[0.4 -1 0],'Yaw',180);
39 -  Waypoints = [0.4 -1; 36 -1]; % in meters
40 -  Speed = 13.89; % egoCar speed = 13.89 m/s = 50 km/hr
41 -  path(egoCar, Waypoints, Speed); % create egoCar path
42
43    % --- two stationary cars
44 -  vehicle(s,'Position',[35.3 -3.8 0]);
45 -  vehicle(s,'Position',[29.6 -3.8 0]);
46
47    % --- child pedestrian crossing it's path running from behind of stationary
48    % cars
49 -  child = actor(s,'Length',0.24,'Width',0.45,'Height',1.7,...
50        'Position',[40 -5 0],'Yaw',180);
51 -  Waypoints = [40 -5; 40 10]; % in meters
52 -  Speed = 1.39; % child speed = 1.39 m/s = 5 km/hr
53 -  path(child, Waypoints, Speed); % create child path
```

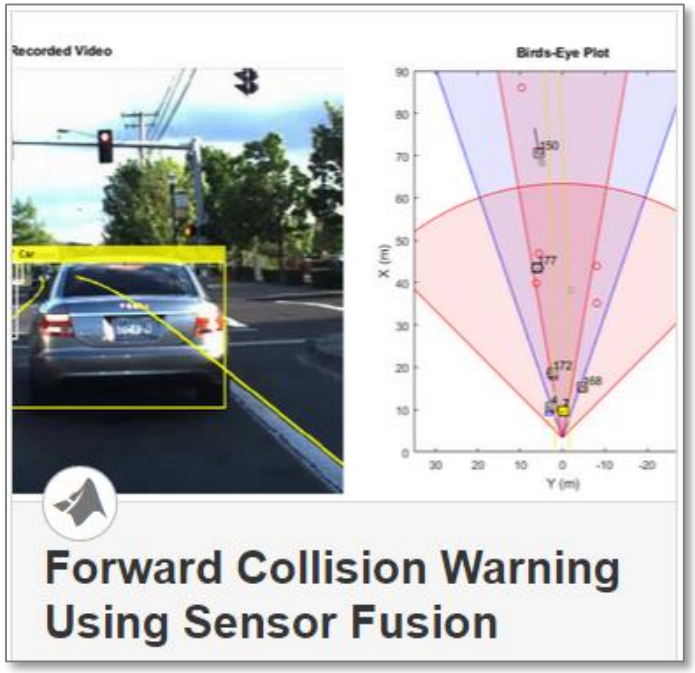


C scenario, Running Child from Nearside
from Obstruction vehicles (see Annex B)

Learn More About Sensor Fusion

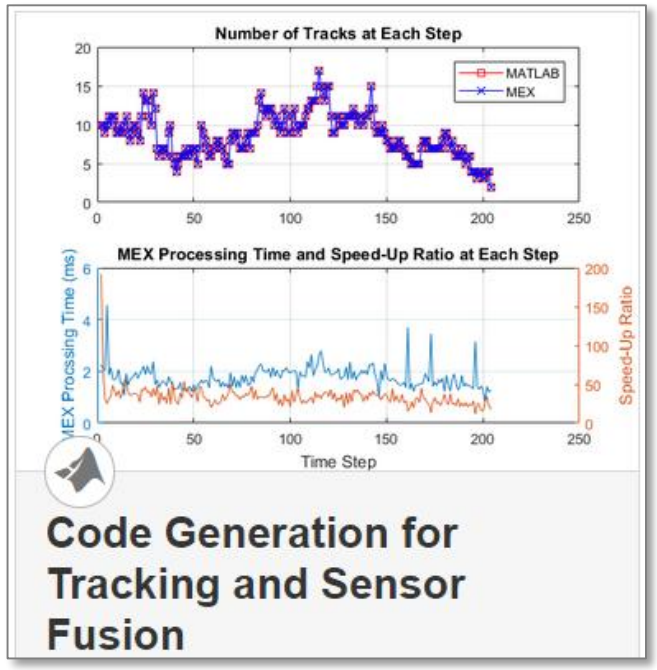
Explore examples in the Automated Driving System Toolbox

R2017a



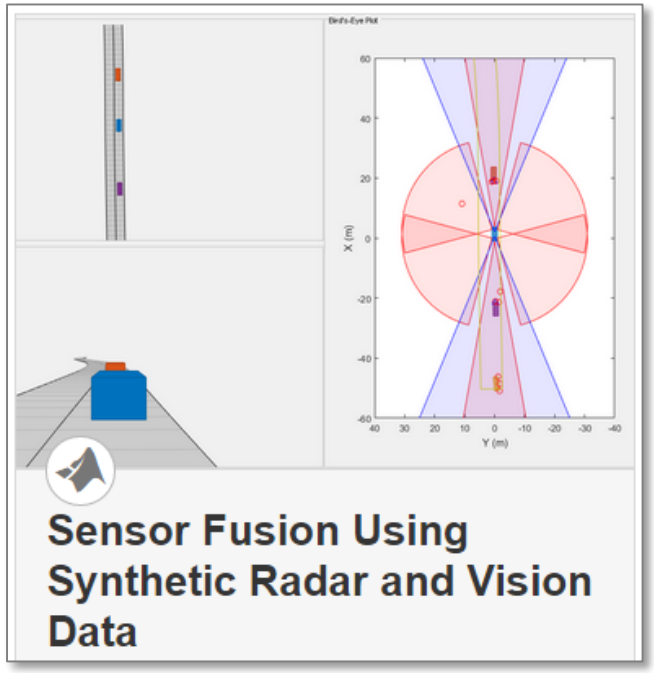
Forward Collision Warning Using Sensor Fusion

- **Design** multi-object tracker based on logged vehicle data



Code Generation for Tracking and Sensor Fusion

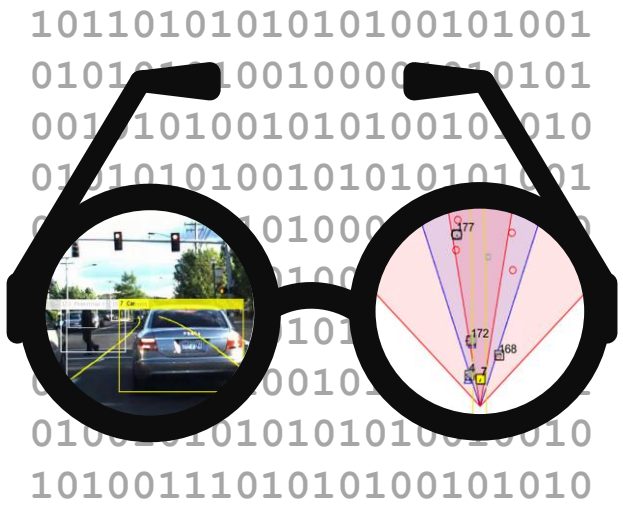
- **Generate C/C++** code from algorithm which includes a multi-object tracker



Sensor Fusion Using Synthetic Radar and Vision Data

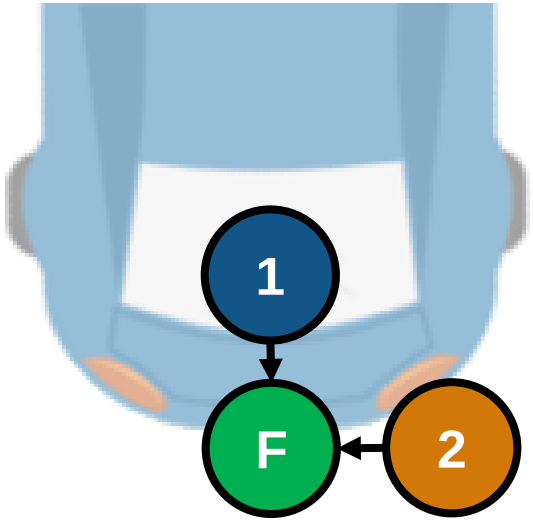
- **Synthesize driving scenario** to test multi-object tracker

Automated Driving System Toolbox



How can I visualize **vehicle data?**

How can I design and verify **perception algorithms?**



How can I design and verify **sensor fusion?**