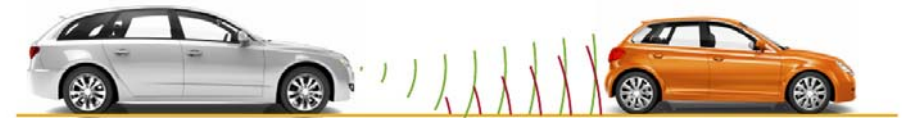


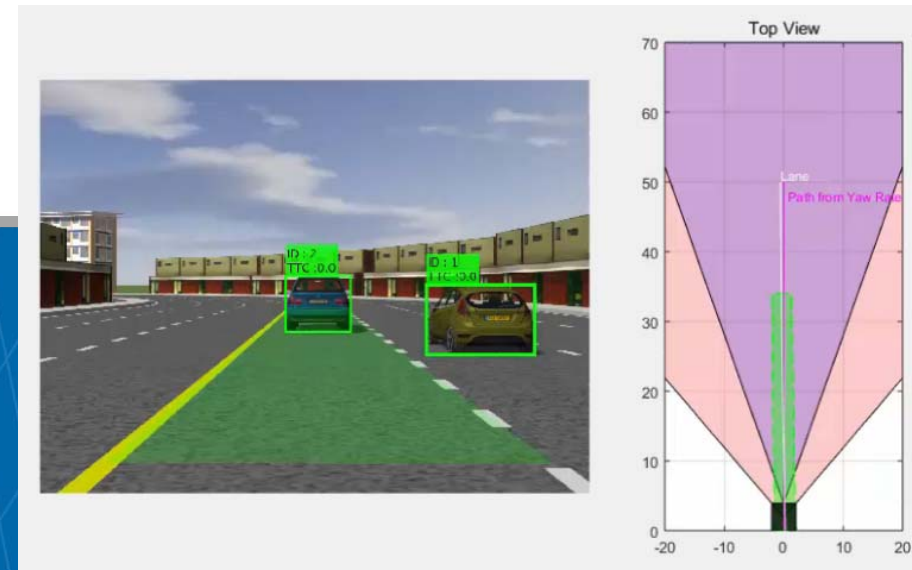
Developing Active Safety Systems Using MATLAB and Simulink



MathWorks
AUTOMOTIVE CONFERENCE 2015

Marco Roggero
Senior Application Engineer

marco.roggero@mathworks.de



Introduction & Motivation

- Active Safety Systems must operate consistent and robust also in unpredictable environments
- Testing these systems in a real world environment is dangerous and can cause serious damage
- Examples:
 - Lane Keeping Systems
 - Adaptive Cruise Control Systems
 - Automated Emergency Braking

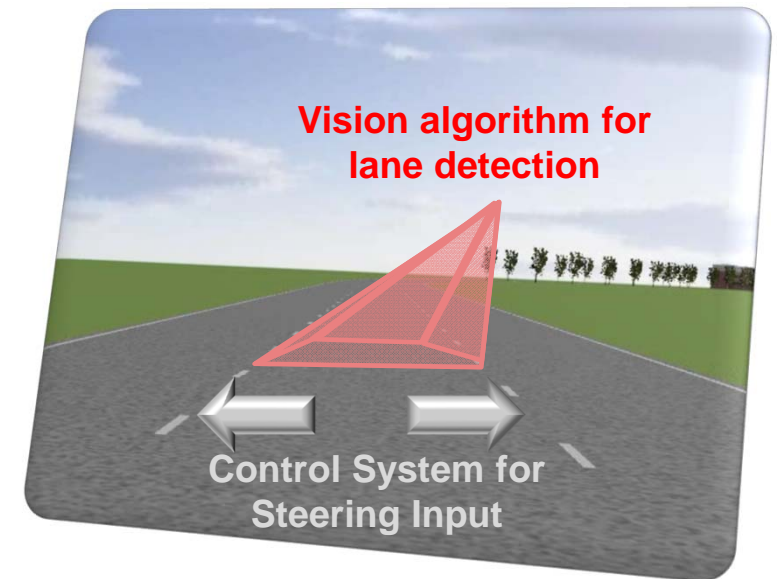
**Develop and verify your
active safety system functionality and
improve robustness using system level simulation.**

What is Active Safety?

- Safety systems that are active *prior* to an accident
 - Use an understanding of the state of the vehicle and its environment to avoid and minimize the effects of a crash.
 - Interpret signals from various sensors and decide how to help the driver to control the vehicle.

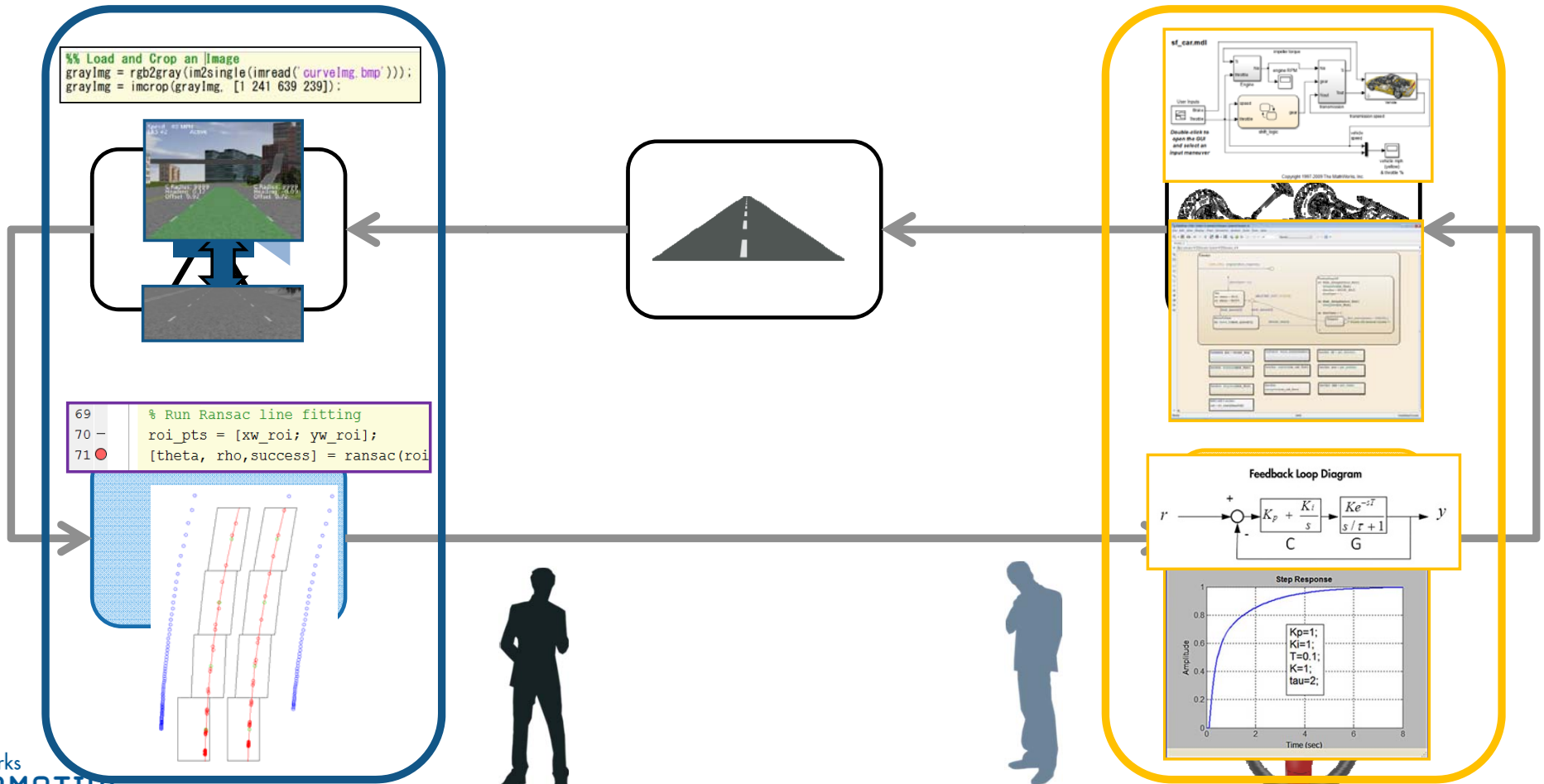
Case Study: Lane Keeping System

- Detect the vehicle's departure from its lane
- Warn the driver or actively steer the vehicle



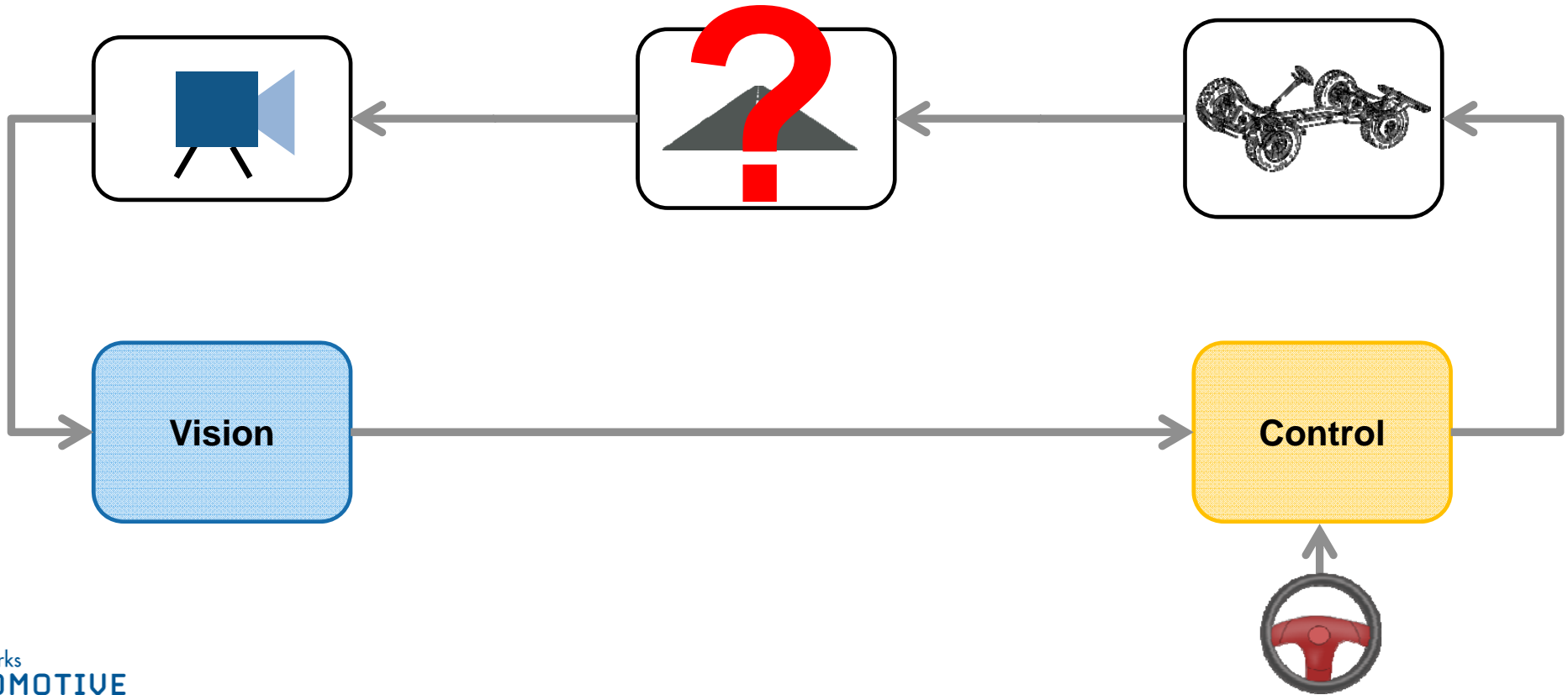
Developing Active Safety Systems

A Multi-Domain Problem



Developing Active Safety Systems

The Challenge: Closed Loop with Environment



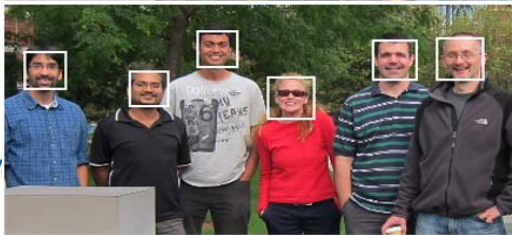
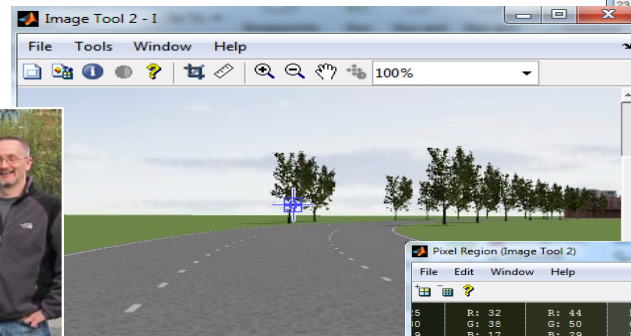
Developing Active Safety Systems

Developing the Vision Algorithm Part

- MATLAB based workflow provides
 - Easy to debug scripting environment
 - Pixel level, 2D, 3D visualization
 - Easy-to-use powerful image processing, and computer vision algorithms

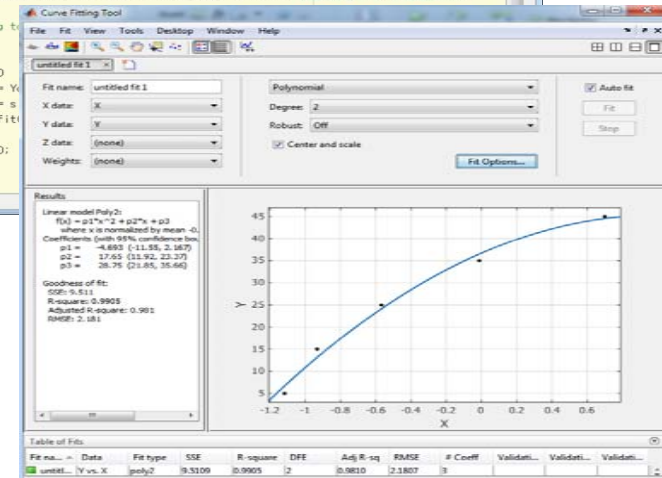
```

8  Eq = zeros(2,4);
9  Valid = [1,1];
10 if isempty(LaneDetectBefore)
11     LaneDetectBefore = 0;
12     ycenter_init = [2, -2];
13     slope_init = [0, 0];
14     Count = 0;
15 end
16
17 % Detect Edges
18 edgeImg = findLaneEdges(grayImg, filter_kernel);
19
20 % Find lane points
21 if LaneDetectBefore == 0
22     ycenter_init = [2, -2];
23     slope_init = [0, 0];
24 end
25
26 [Xcenter, Ycenter, slope, NSupportPts] = findLanePts2(edgeImg, invtform, ycenter_init, slope_init);
27
28 % adjust xmin_orig according to
29 LaneDetectBefore = 1;
30 for lane = 1:2
31     if NSupportPts(lane) > 0
32         ycenter_init(lane) = Ycenter;
33         slope_init(lane) = slope;
34         Eq(lane,2:4) = polyfit(Xcenter, Ycenter, 2);
35     else
36         LaneDetectBefore = 0;
37     end
38 end
    
```



Pixel Region (Image Tool 2)

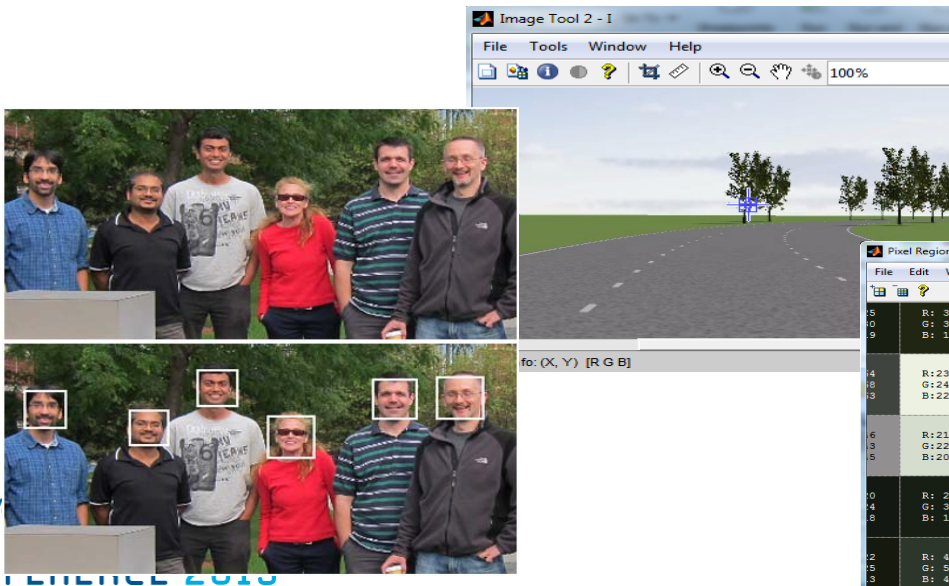
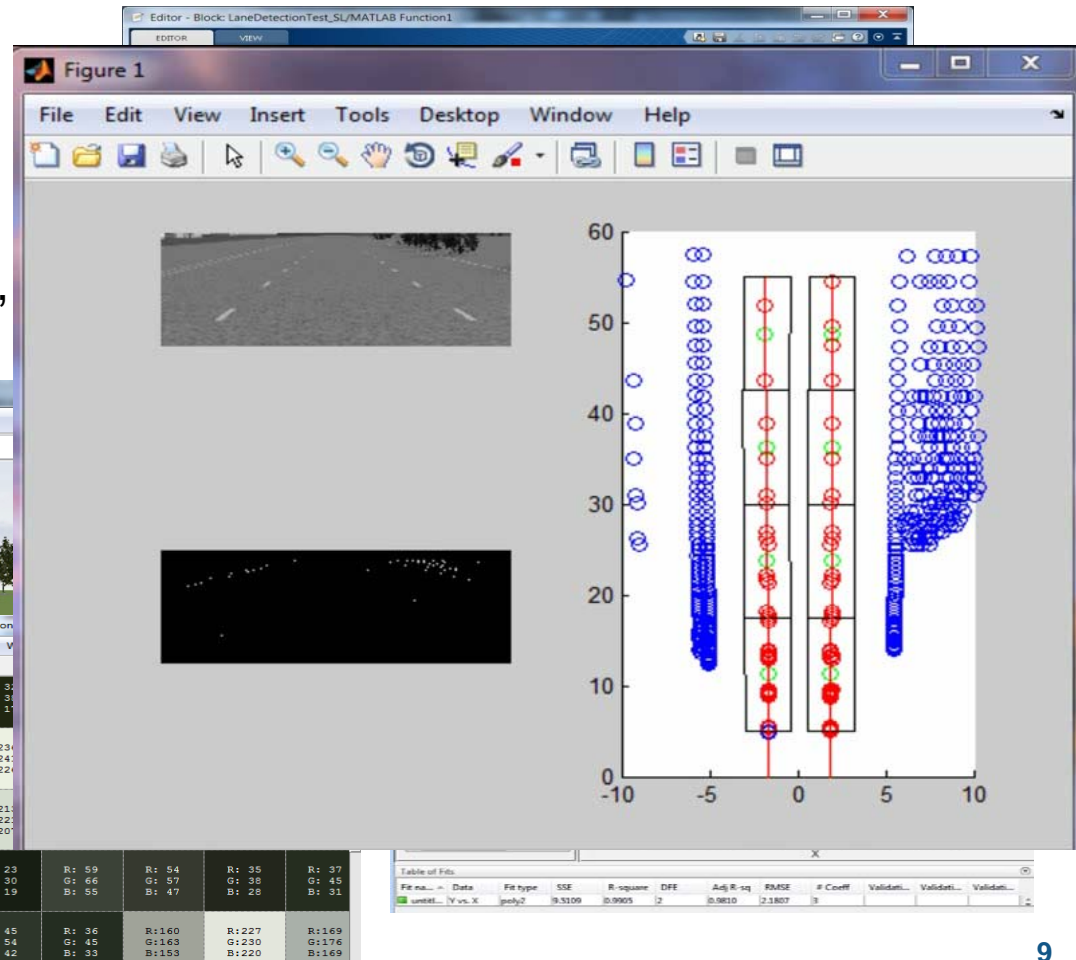
| | | | | | |
|----|--------|--------|--------|--------|--------|
| 5 | R: 32 | R: 44 | R: 66 | R: 78 | R: 238 |
| 6 | G: 36 | G: 50 | G: 71 | G: 83 | G: 239 |
| 7 | B: 17 | B: 29 | B: 58 | B: 70 | B: 225 |
| 8 | R: 236 | R: 127 | R: 55 | R: 238 | R: 56 |
| 9 | G: 241 | G: 132 | G: 56 | G: 239 | G: 59 |
| 10 | B: 226 | B: 117 | B: 44 | B: 227 | B: 49 |
| 11 | R: 213 | R: 59 | R: 160 | R: 232 | R: 184 |
| 12 | G: 221 | G: 67 | G: 159 | G: 231 | G: 187 |
| 13 | B: 207 | B: 53 | B: 146 | B: 218 | B: 175 |
| 14 | R: 23 | R: 59 | R: 54 | R: 35 | R: 37 |
| 15 | G: 30 | G: 66 | G: 57 | G: 38 | G: 45 |
| 16 | B: 19 | B: 55 | B: 47 | B: 28 | B: 31 |
| 17 | R: 45 | R: 36 | R: 160 | R: 227 | R: 169 |
| 18 | G: 54 | G: 45 | G: 163 | G: 230 | G: 176 |
| 19 | B: 42 | B: 33 | B: 153 | B: 220 | B: 169 |



Developing Active Safety Systems

Developing the Vision Algorithm Part

- MATLAB based workflow provides
 - Easy to debug scripting environment
 - Pixel level, 2D, 3D visualization
 - Easy-to-use powerful image processing, computer vision algorithms



Developing Active Safety Systems

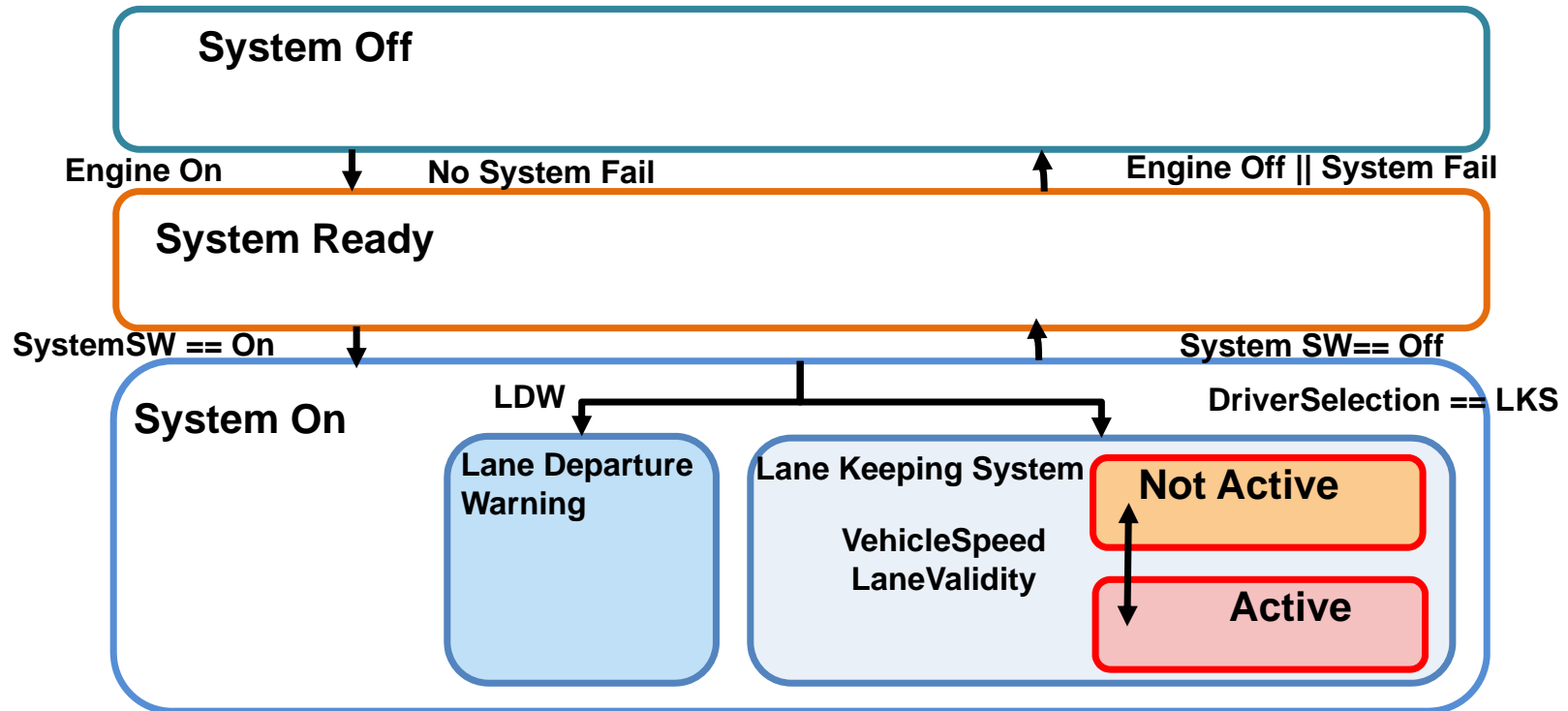
Developing the Controller Part

- Goal
 - Keep the vehicle within a lane by controlling steering input
- Controller configuration
 - Mode Selector (Stateflow)
 - Risk Assessment (Stateflow)
 - Steering angle compensator (Simulink)
 - Feed-back steer angle using heading & lateral offset
 - Feed-forward steer angle using curvature & vehicle speed
- Control Parameter Tuning

Developing Active Safety Systems

Developing the Controller Part

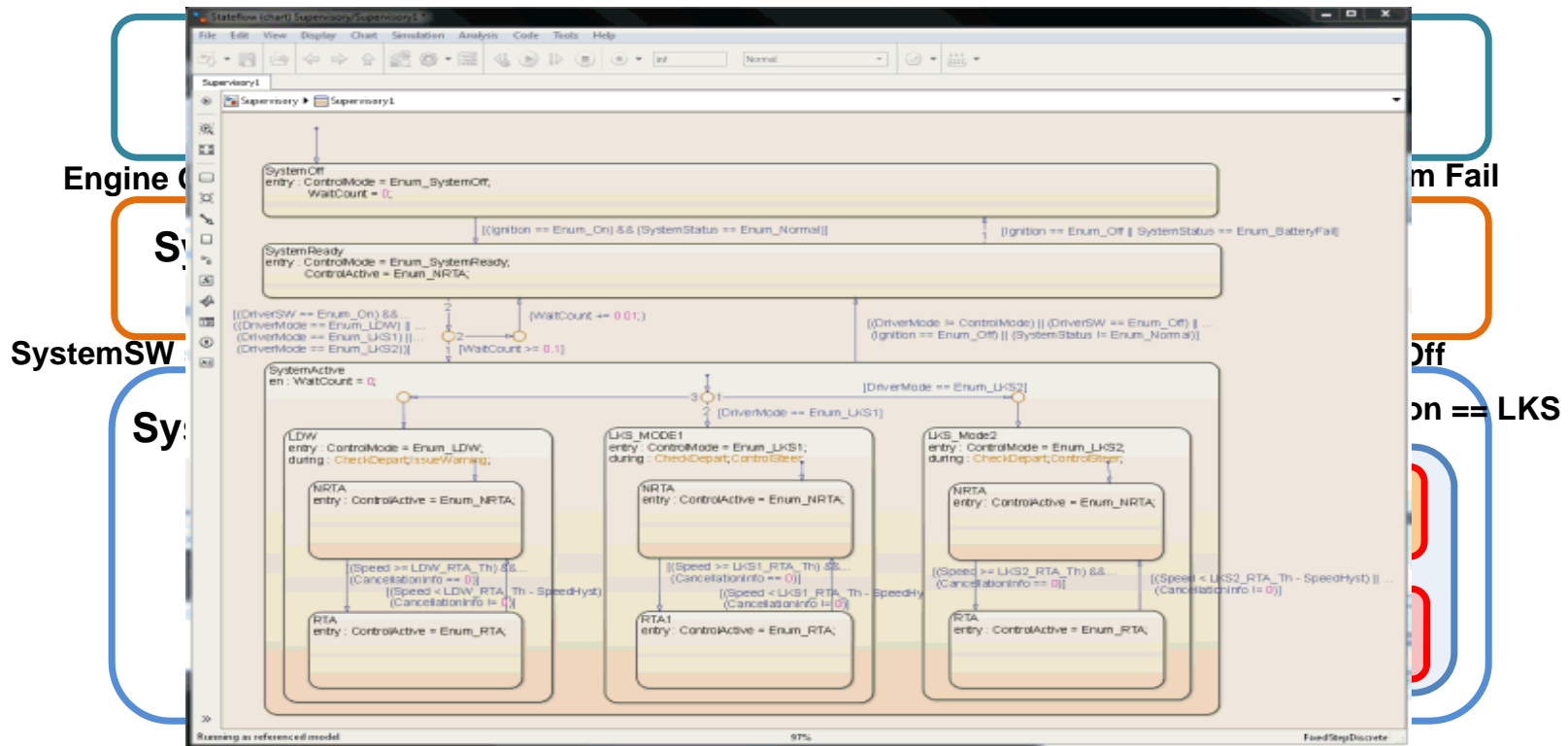
- Determining Control Mode – creating the model



Developing Active Safety Systems

Developing the Controller Part

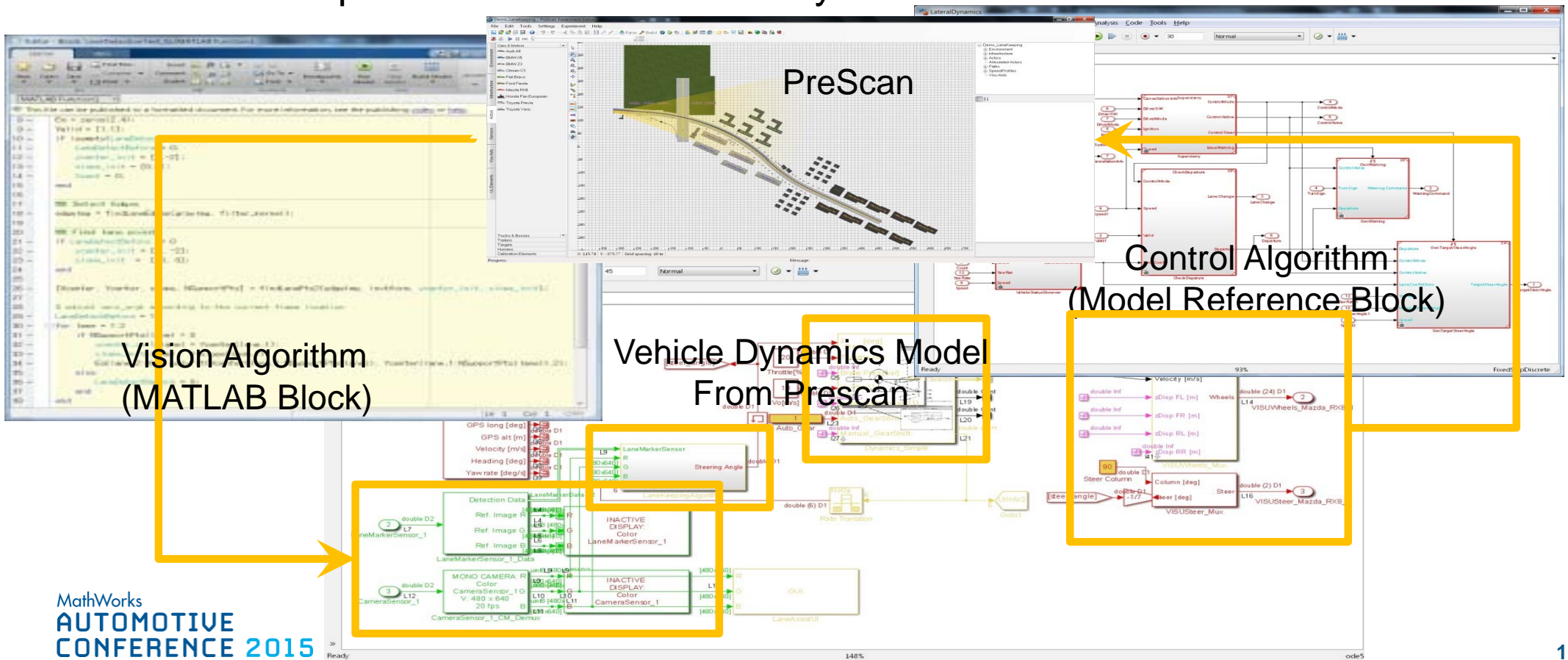
- Determining Control Mode – creating the model



Developing Active Safety Systems

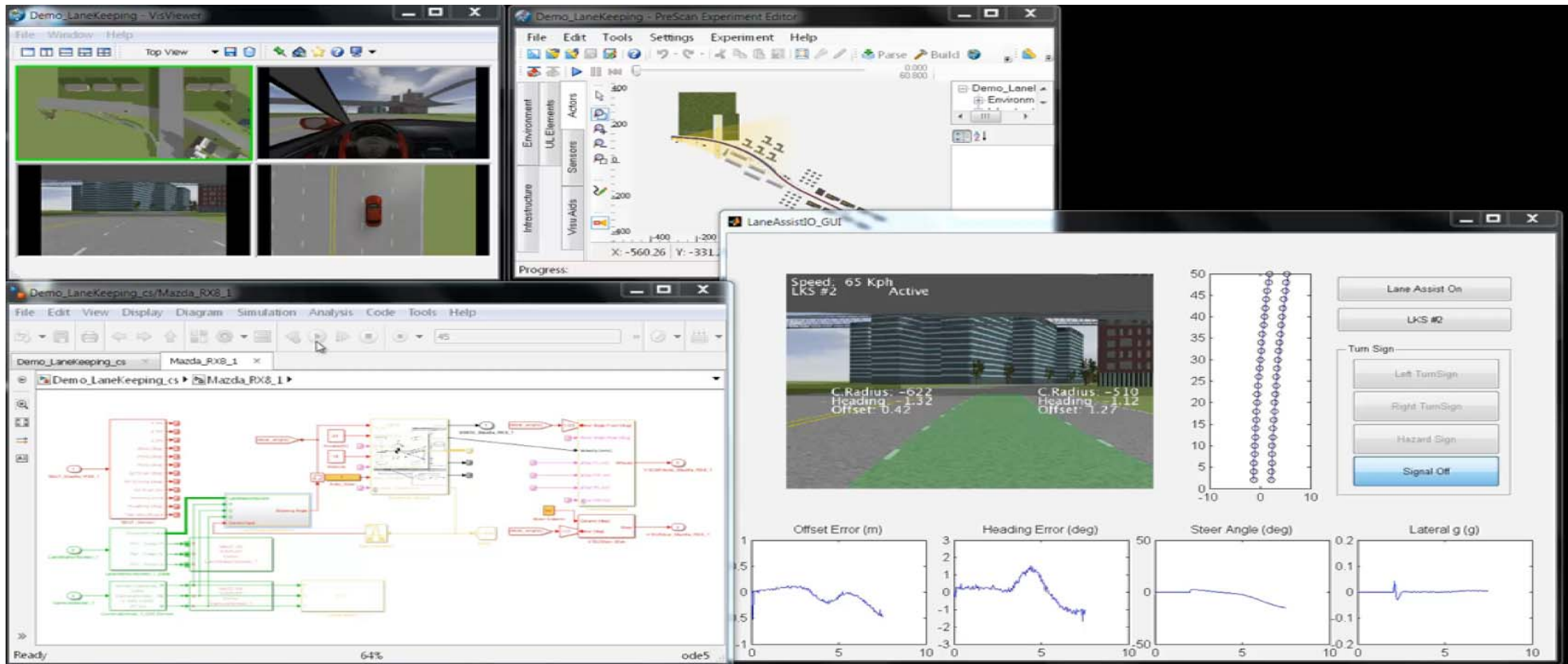
System Level Simulation

- Closed-loop test harness model for system level validation



Developing Active Safety Systems

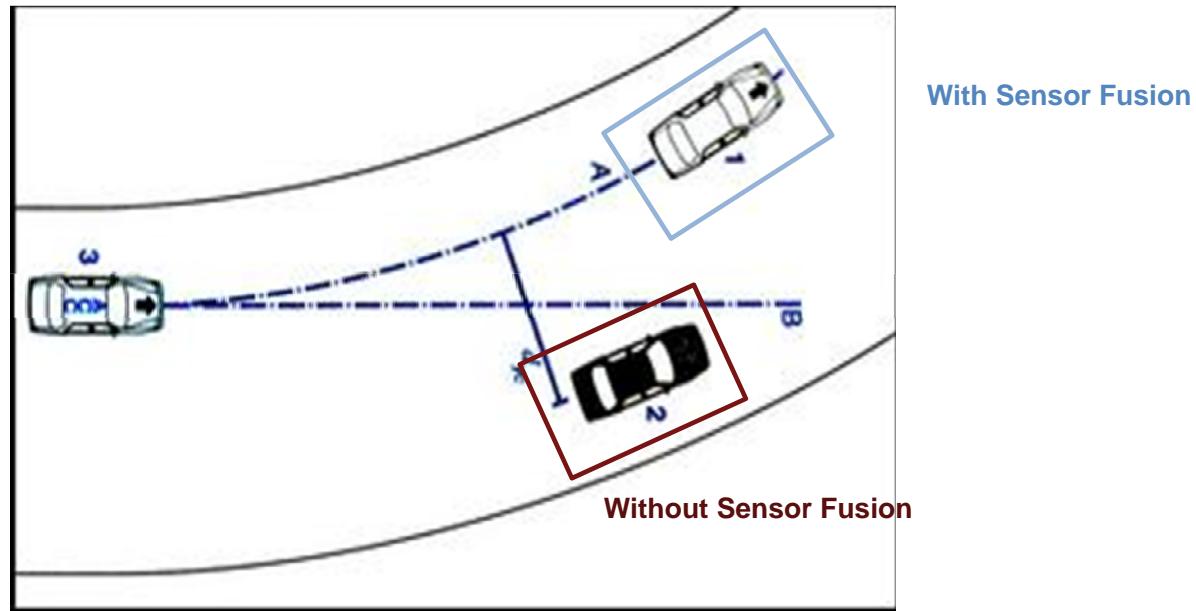
System Level Simulation



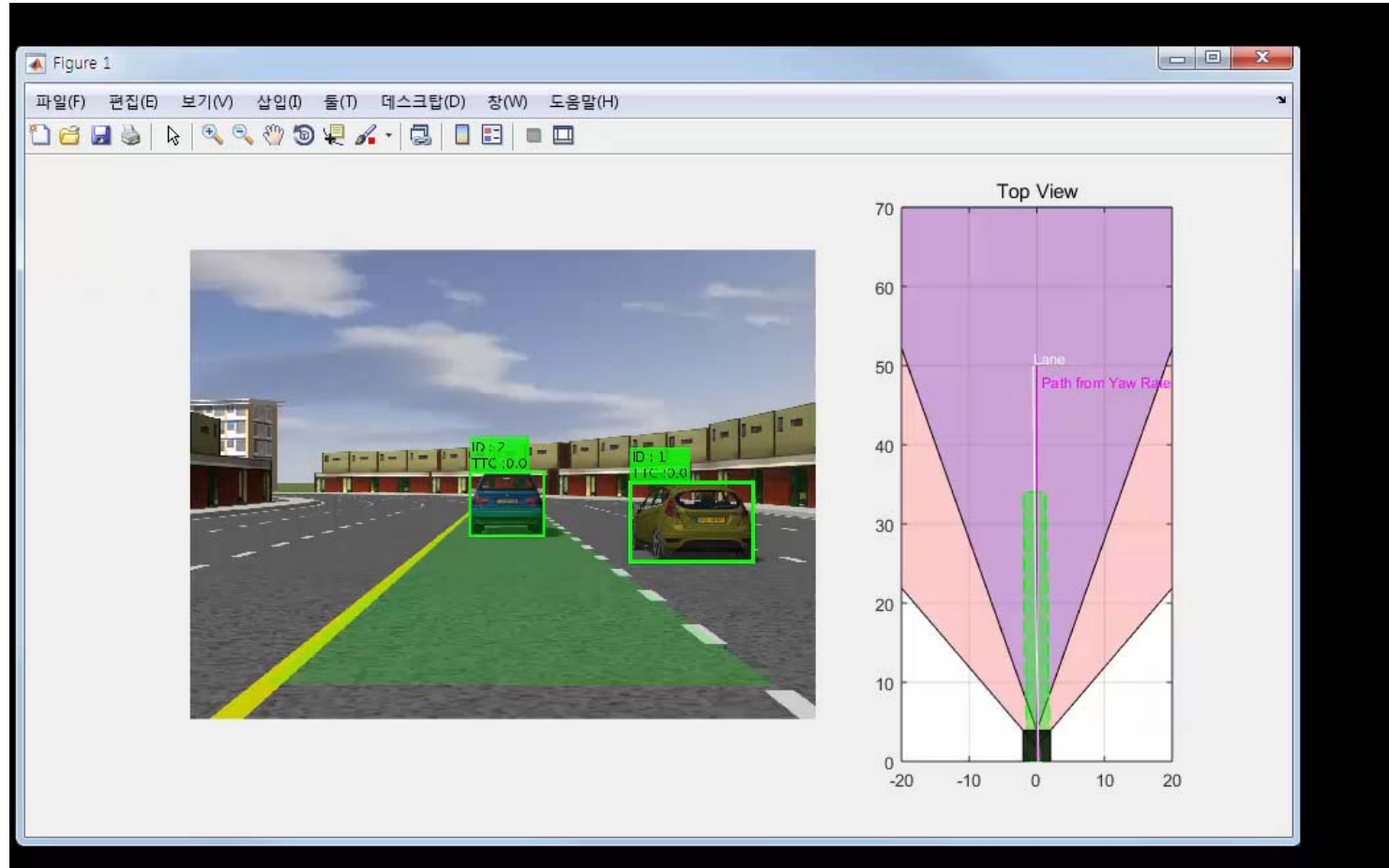
Need for Sensor Fusion

Case study: Automated Emergency Braking

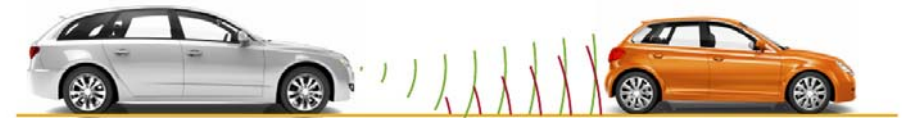
Accurate CIPV (Critical In-Path Vehicle) selection



Example: Radar and Camera Data Fusion



Developing Active Safety Systems Using MATLAB and Simulink



MathWorks
AUTOMOTIVE CONFERENCE 2015

Marco Roggero
Senior Application Engineer

marco.roggero@mathworks.de

