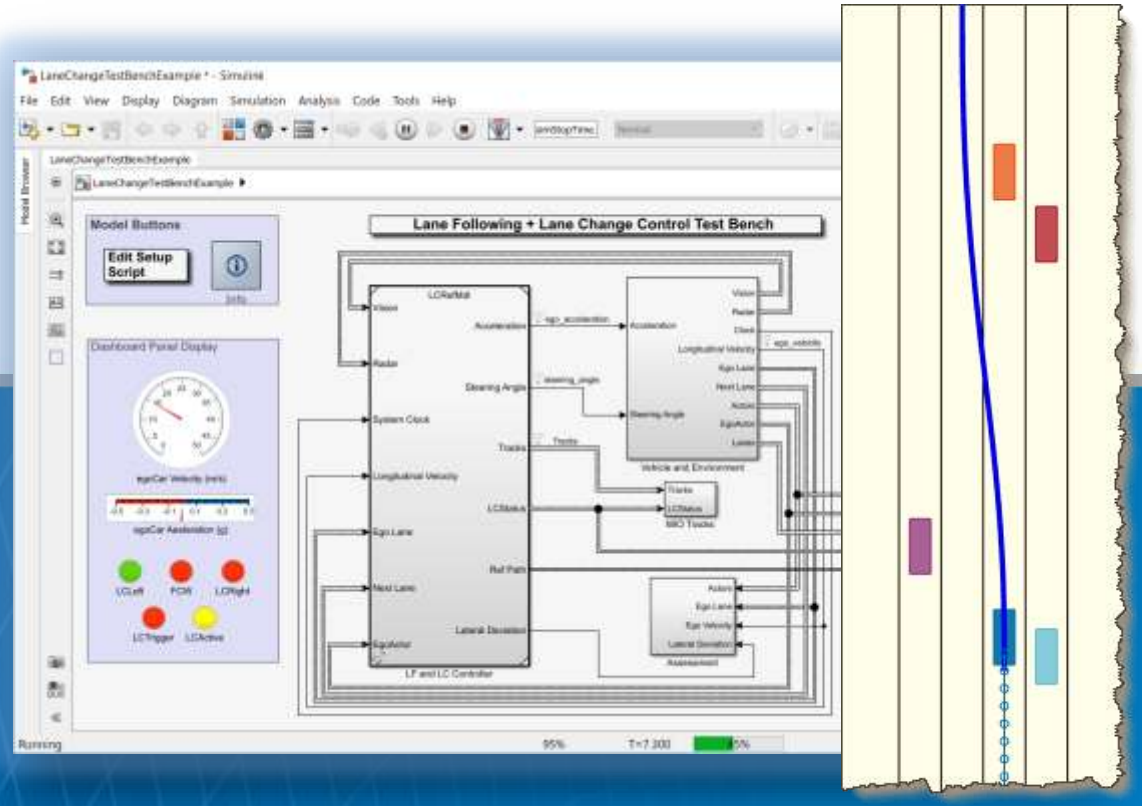
Case Study: Highway Lane Following + Lane Change

Design and test decision making, path planning, and control modules in traffic scenarios



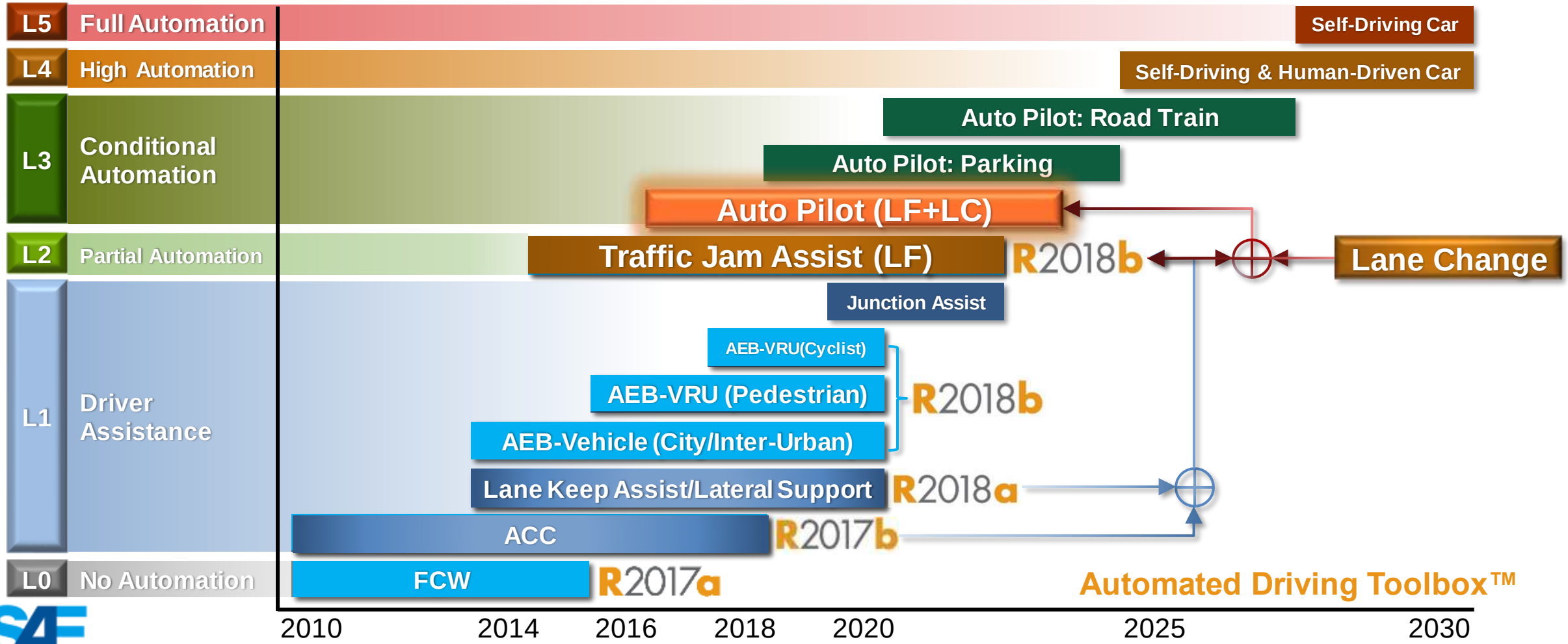
Seo-Wook Park

Application Engineering, MathWorks

Shusen Zhang

Application Engineering, MathWorks

SAE Levels of Driving Automation vs. Automated Driving Technologies

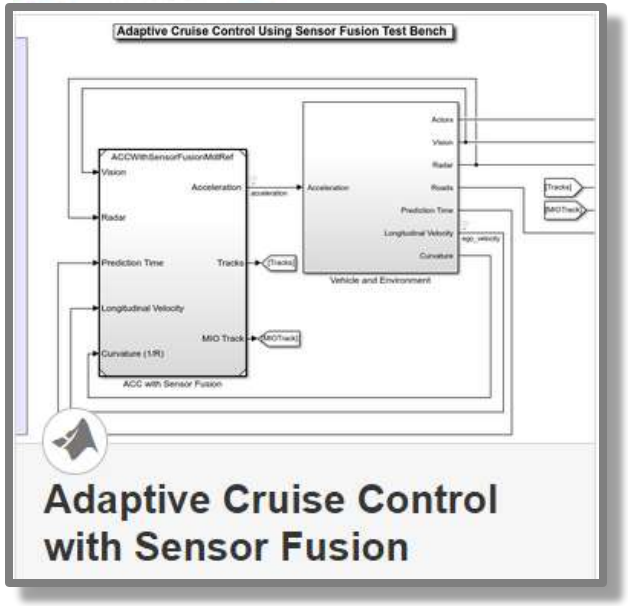


SAE INTERNATIONAL
J3016

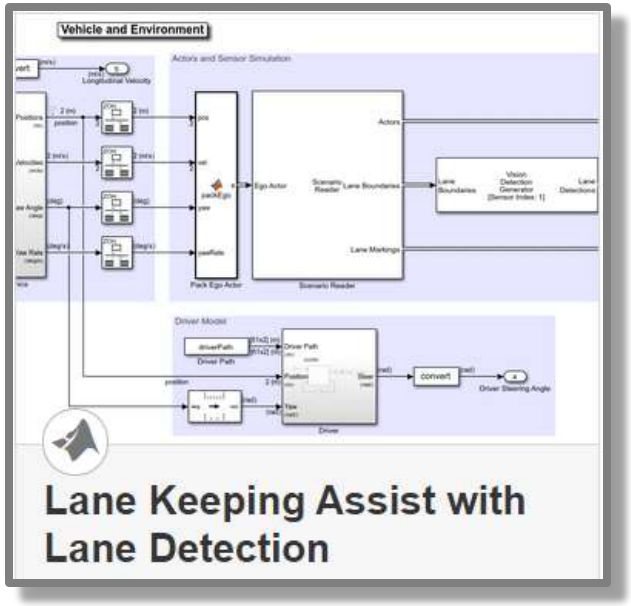
Traffic Jam Assist with ACC and Lane Following Control

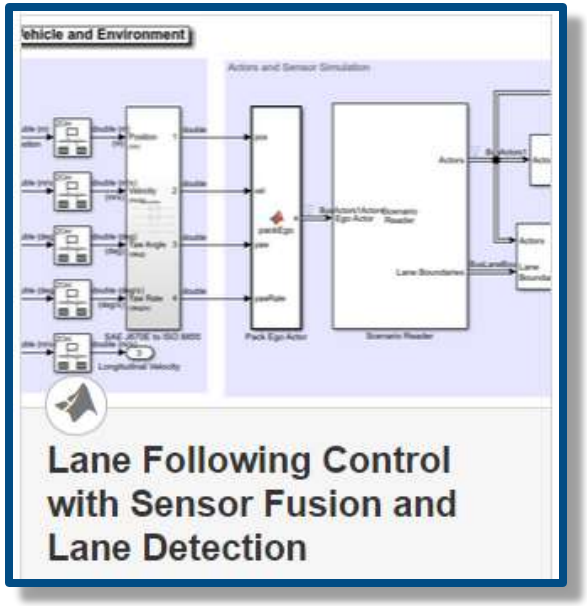
Automated Driving Toolbox™

R2017b



R2018a





ACC
(Longitudinal Control)

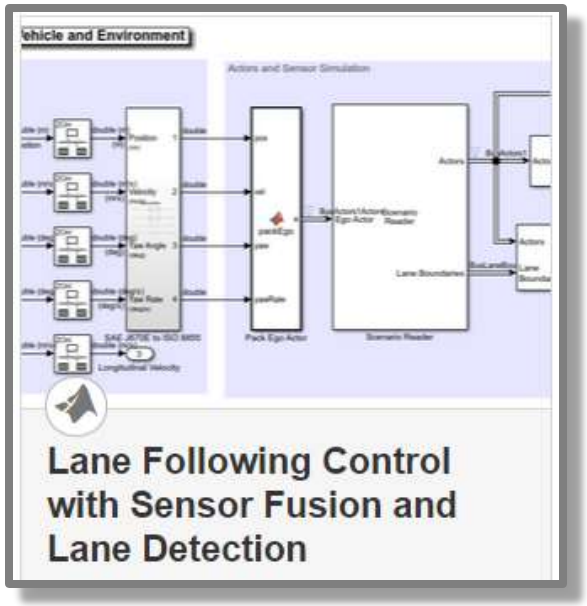
Lane Centering
(Lateral Control)

Traffic Jam Assist
(Longitudinal + Lateral Control)

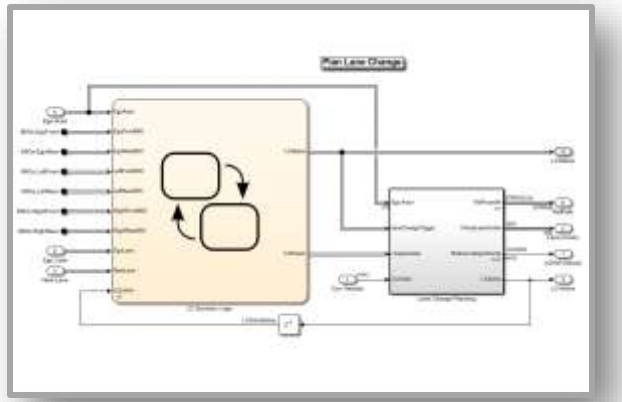
Auto Pilot: Lane Following plus Lane Change

Automated Driving Toolbox™

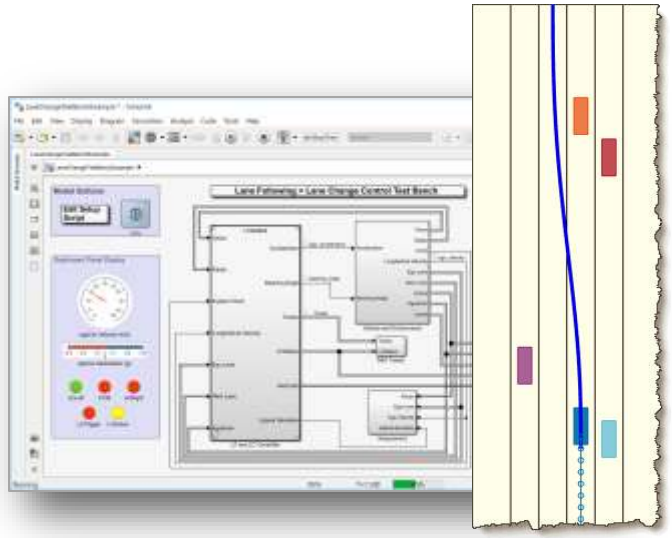
R2018b



+



=



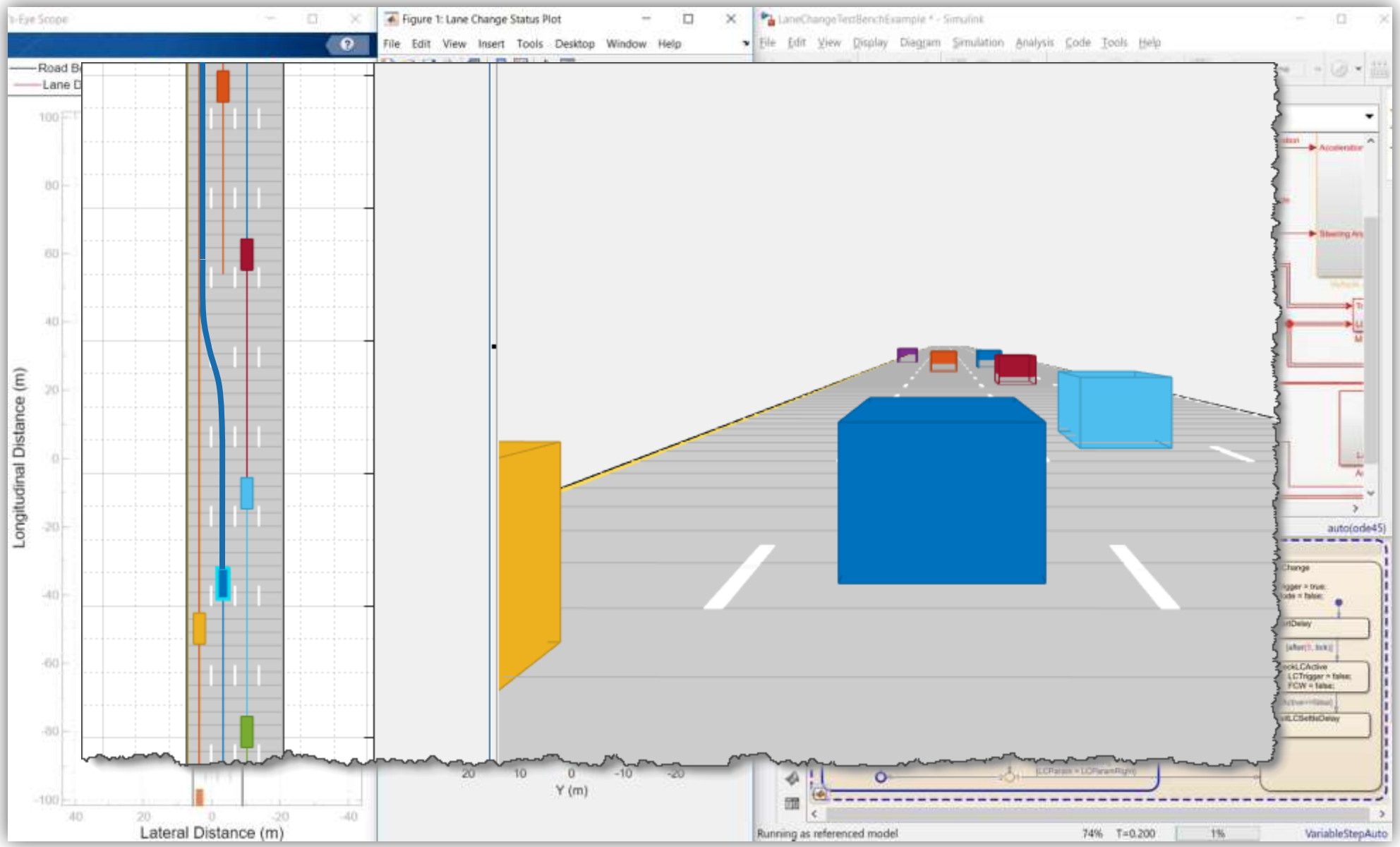
Traffic Jam Assist
(Longitudinal + Lateral Control)

Baseline example

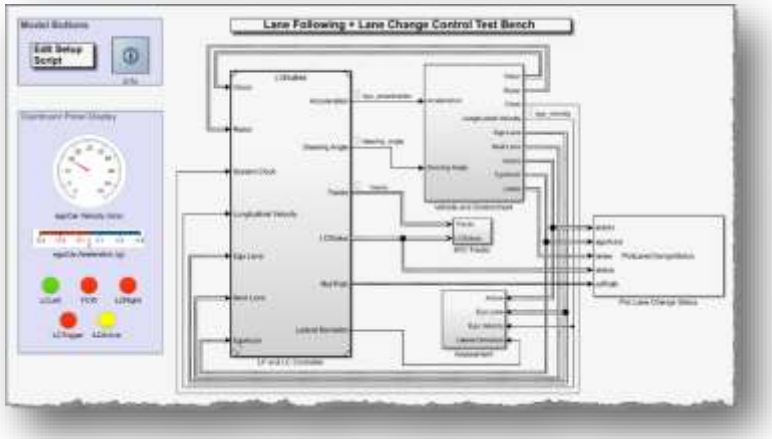
Auto Lane Change
(LC Decision Logic + Planning)

Auto Pilot
(Lane Following + Lane Change)

Example for Single Lane Change in dense traffic conditions

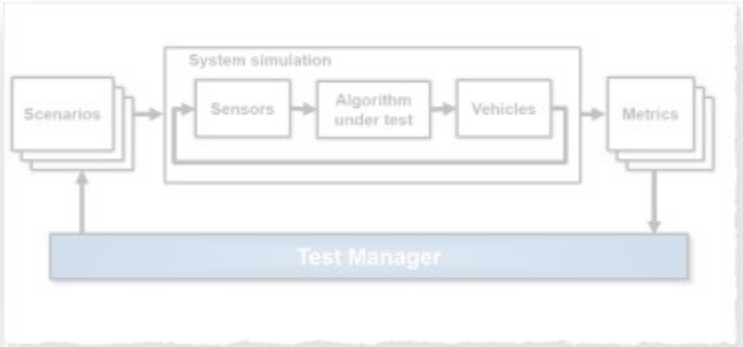


Case Study for Lane Following plus Lane Change



Design lane following + lane change controller

- Review baseline LF example
- Design sensor configuration
- Design additional MIO detectors
- Design safety zone calculation
- Design lane change logic
- Design trajectory planner



Automate regression testing

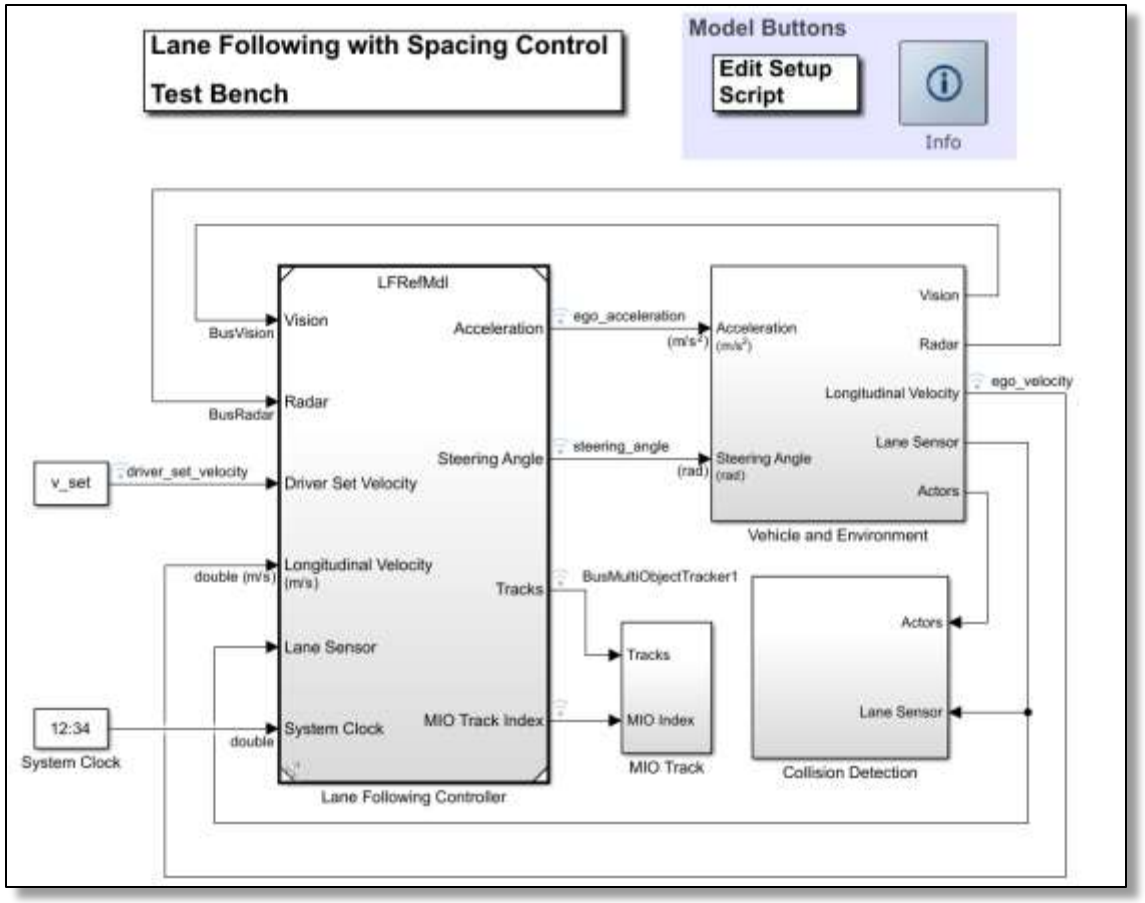
- Define assessment metrics
- Add predefined scenarios
- Run Simulink test

Test robustness with traffic agents

- Specify driver logic for traffic agents
- Randomize scenarios using traffic agents
- Identify and assess unexpected behavior

Learn about developing a lane following controller

R2018b
Automated Driving Toolbox™



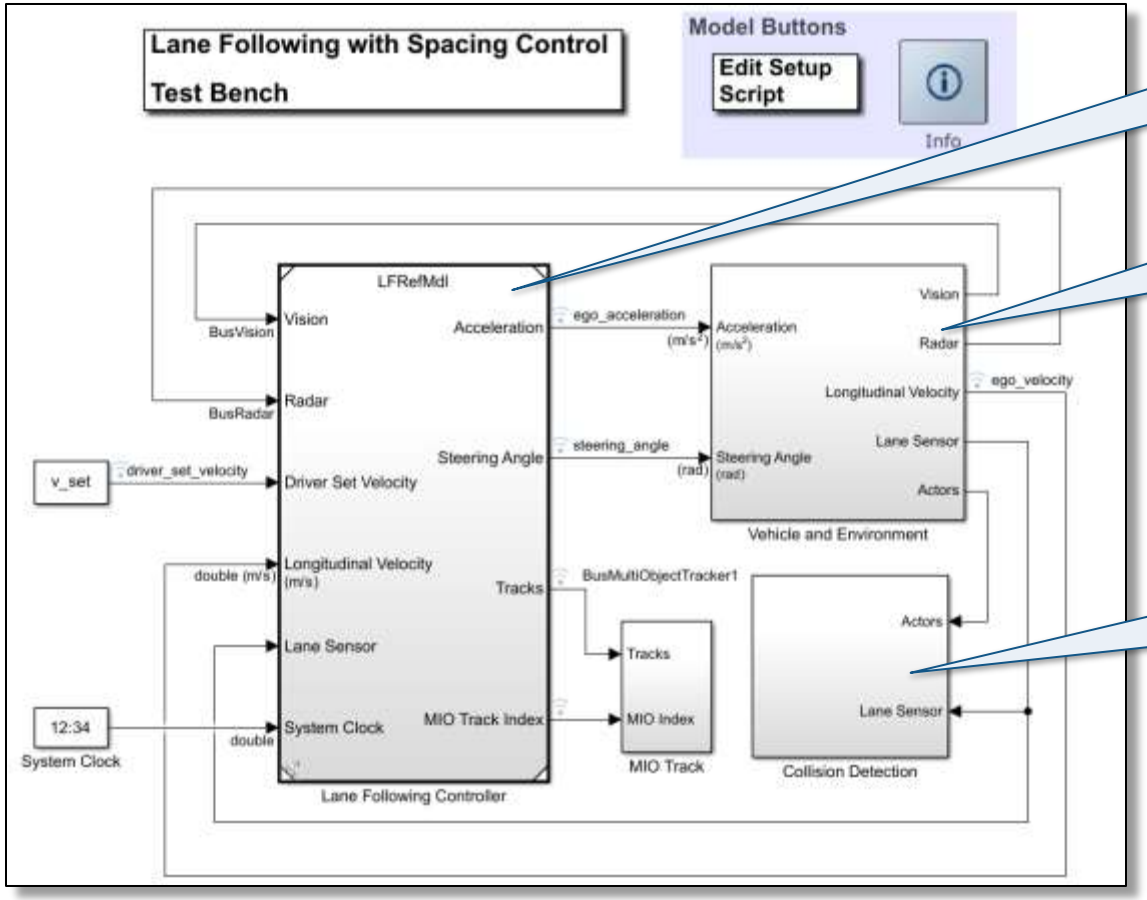
Lane Following Control with Sensor Fusion

- Specify scenario and sensors
- Design lateral (lane keeping) and longitudinal (lane spacing) model predictive controllers
- Integrate sensor fusion
- Generate C/C++ code
- Test with software in the loop (SIL) simulation

Model Predictive Control Toolbox™
Automated Driving Toolbox™
Embedded Coder®

Baseline example

Review lane following test bench model architecture



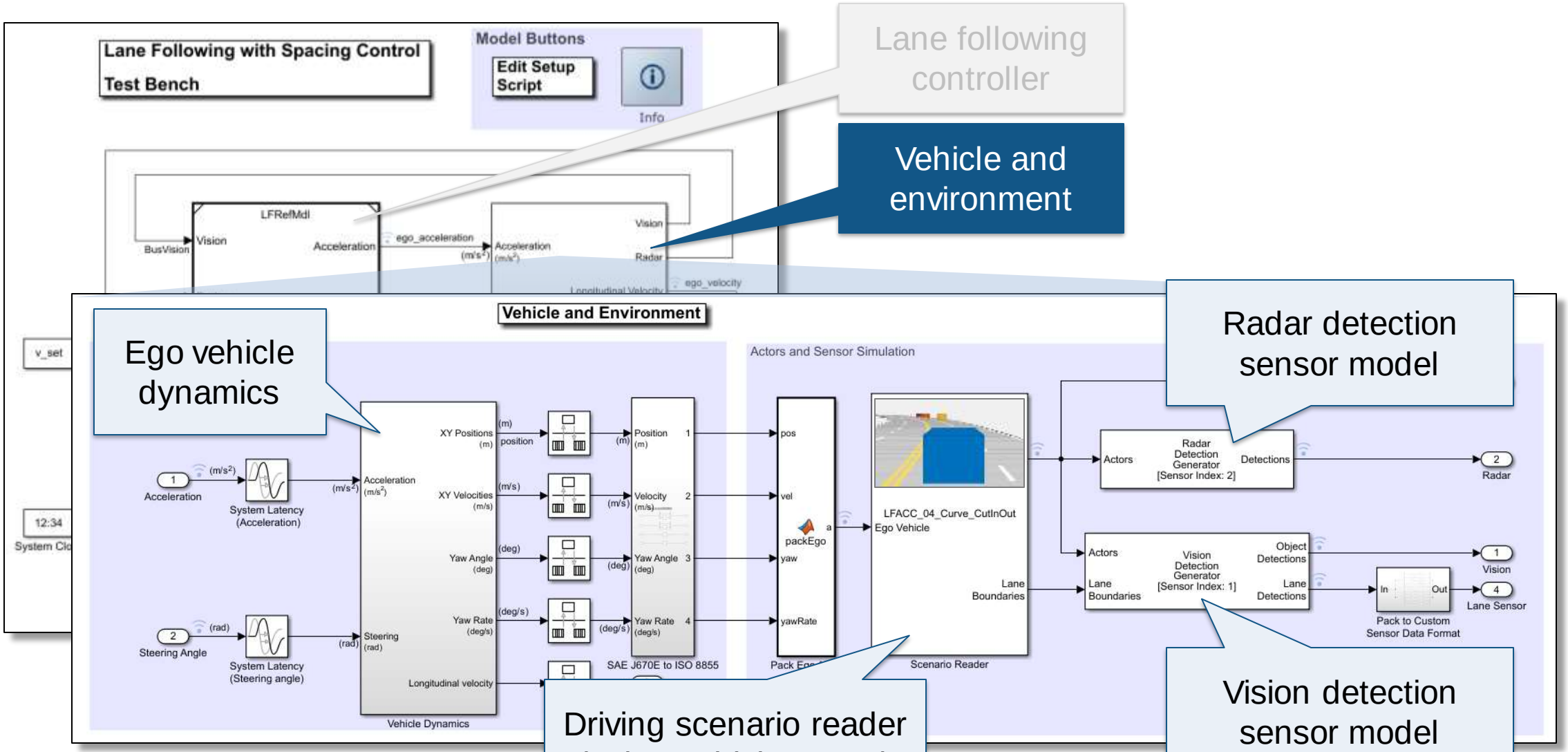
Lane following controller

Vehicle and environment

Collision detection

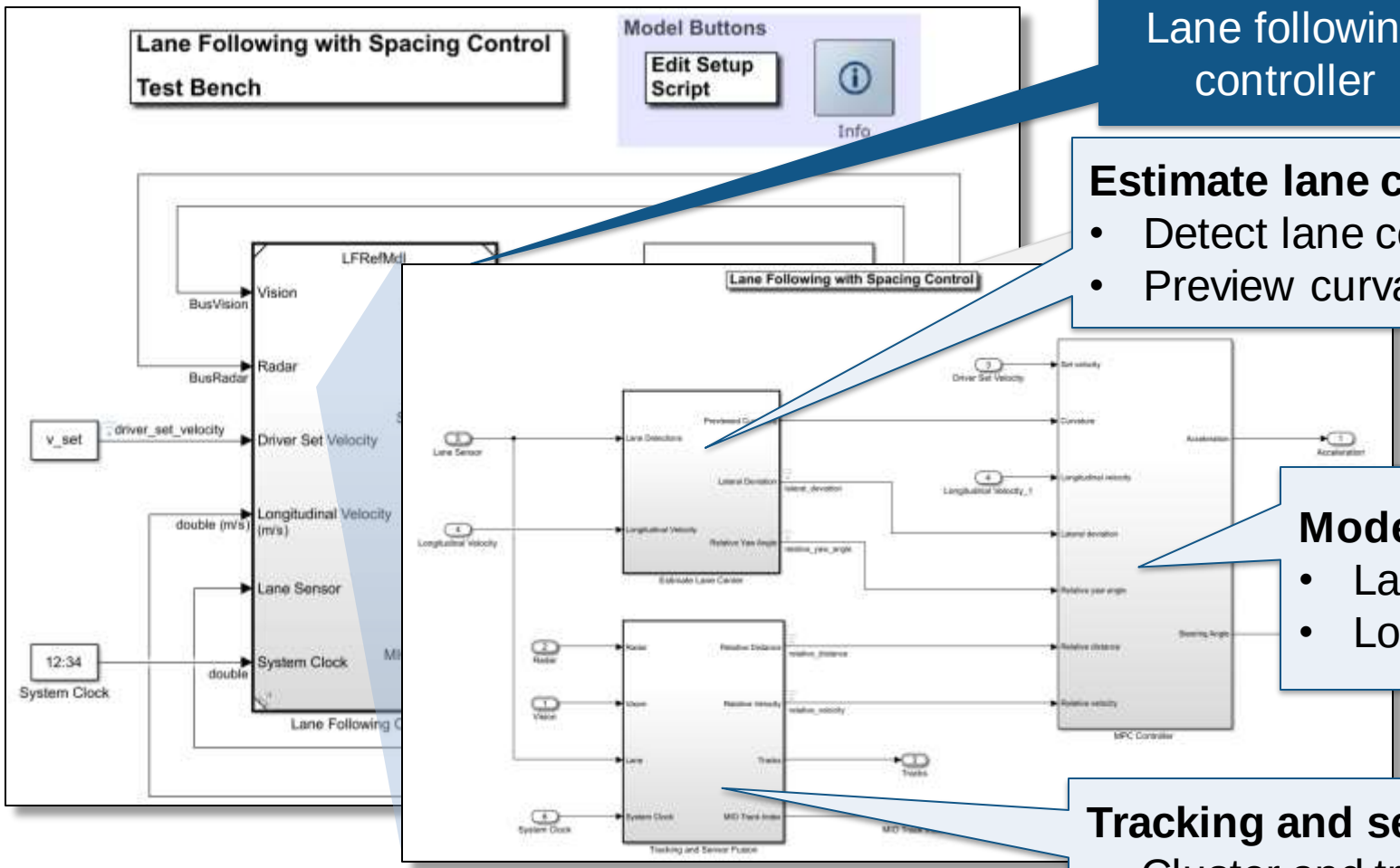
Baseline example

Review lane following test bench model architecture



Baseline example

Review lane following test bench model architecture



Lane following controller

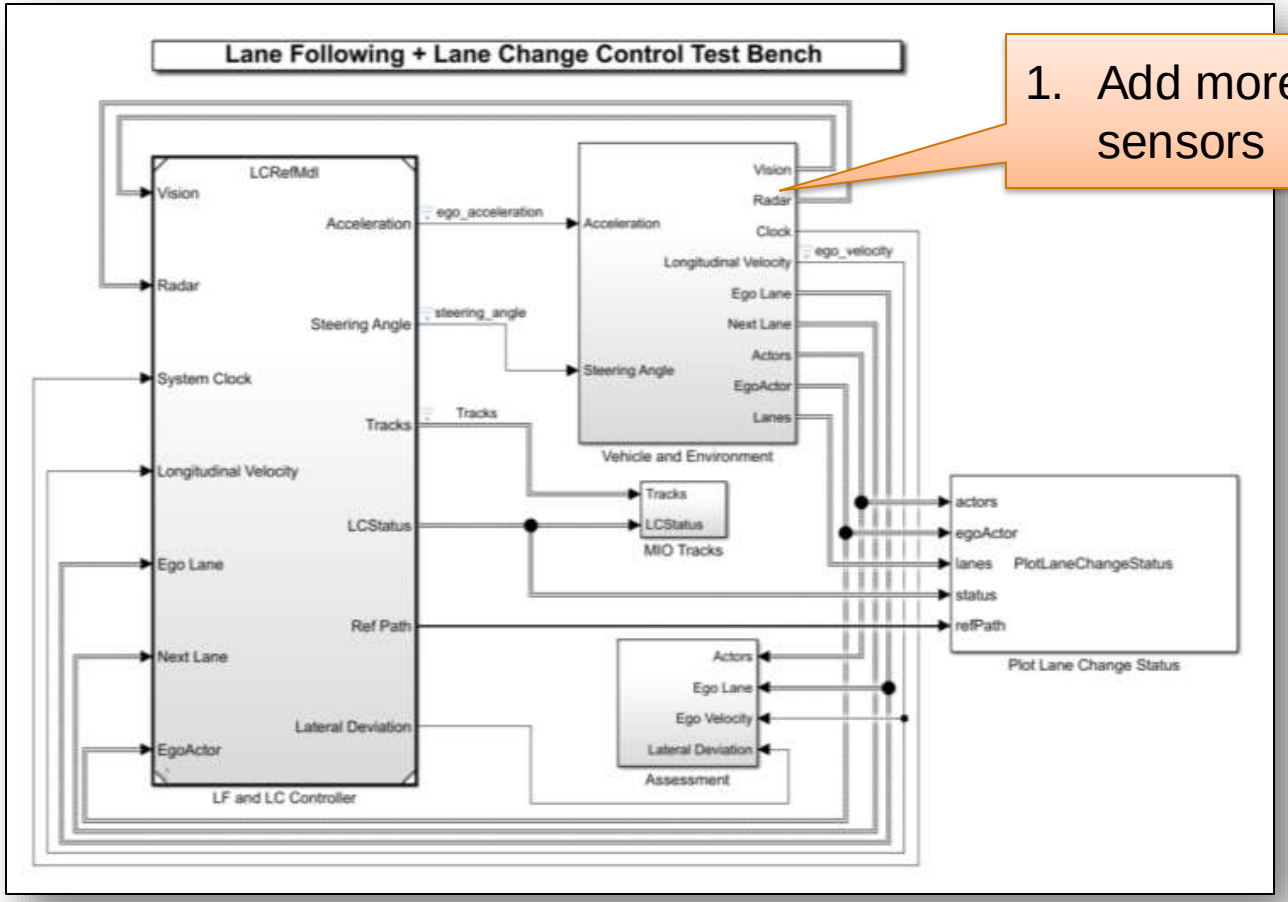
Estimate lane center
- Detect lane center
- Preview curvature

Model predictive controller
- Lateral: Follow lane center
- Longitudinal: Adaptive cruise control

Tracking and sensor fusion
- Cluster and track detections
- Identify most important object (MIO) ahead

Baseline example

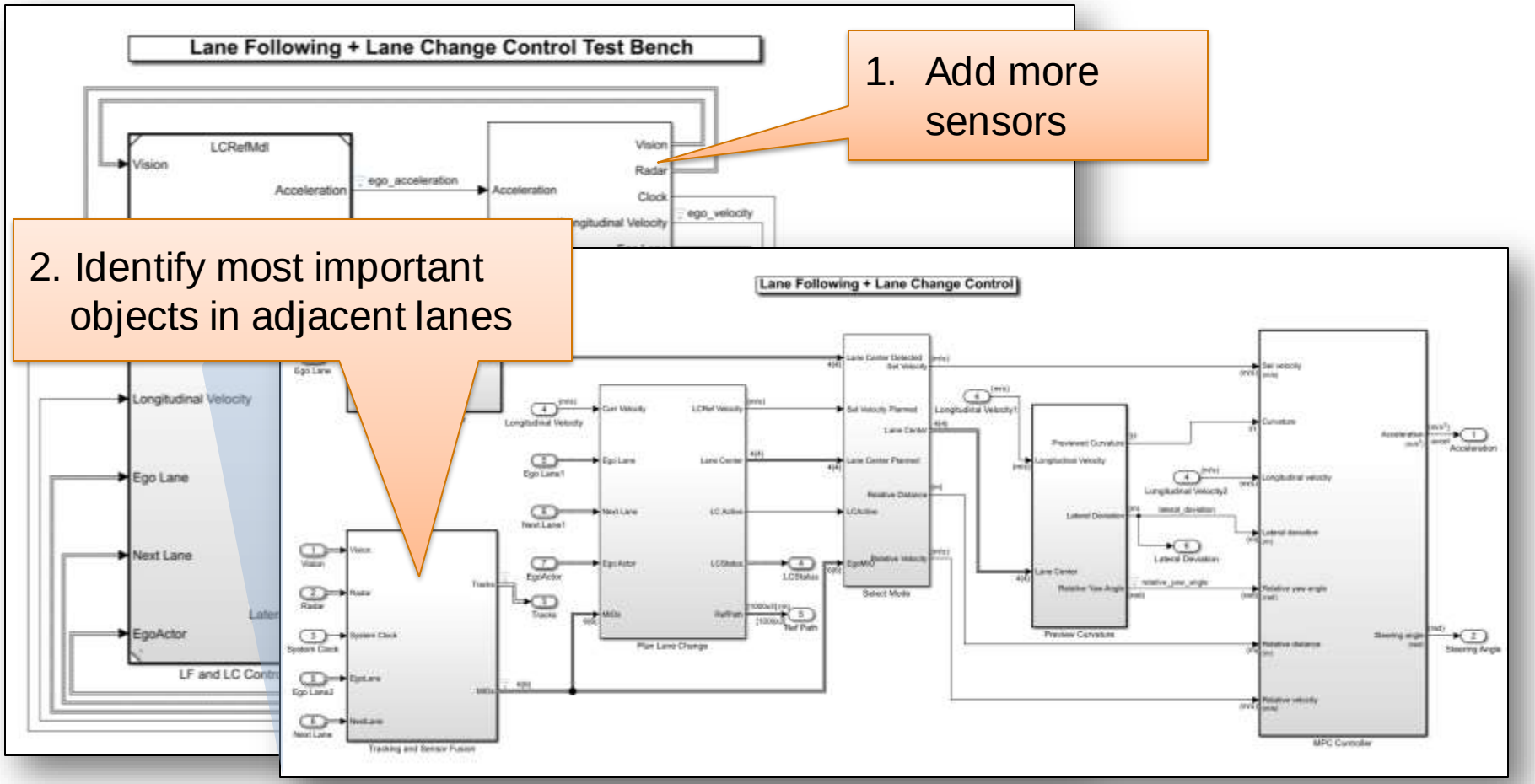
Add lane change functionality to lane following controller



1. Add more sensors

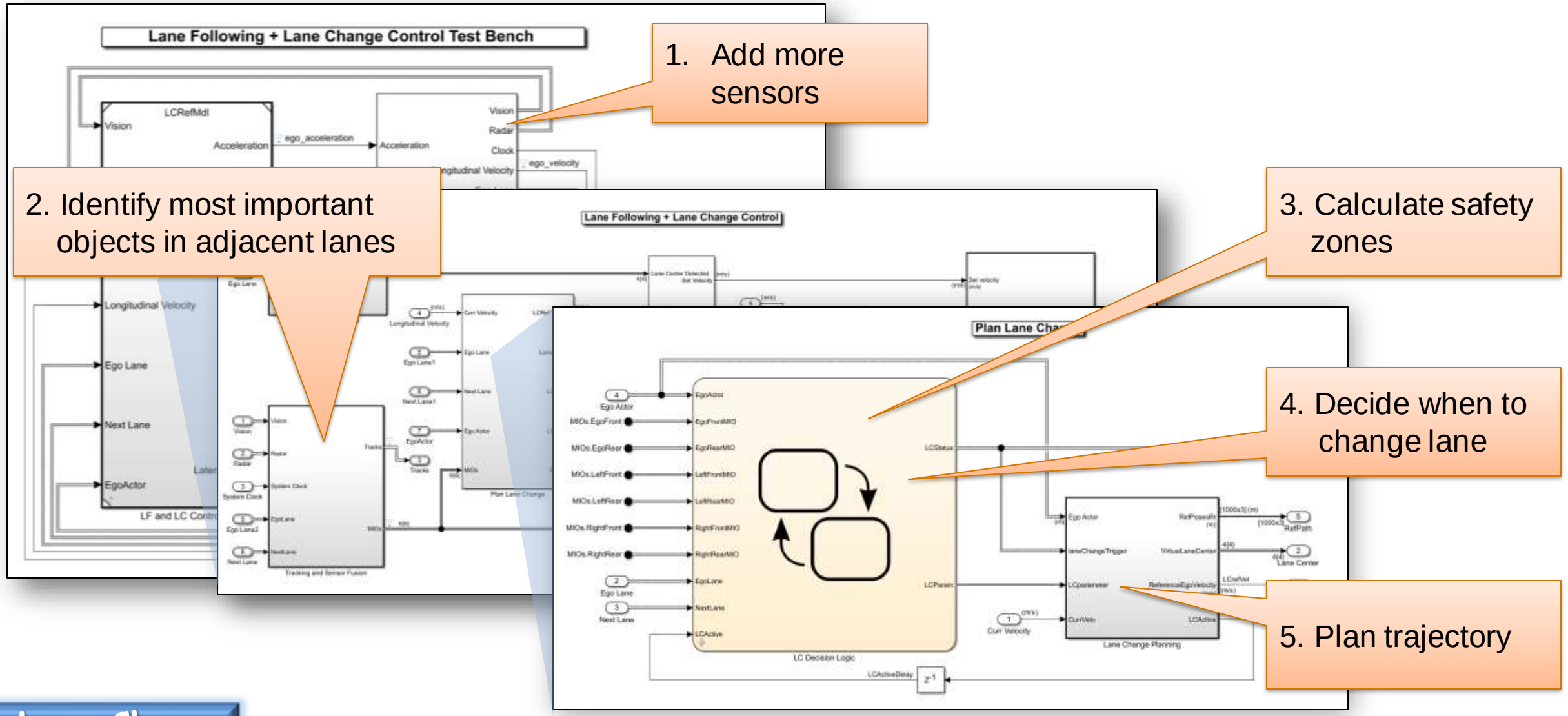
+ Lane Change

Add lane change functionality to lane following controller

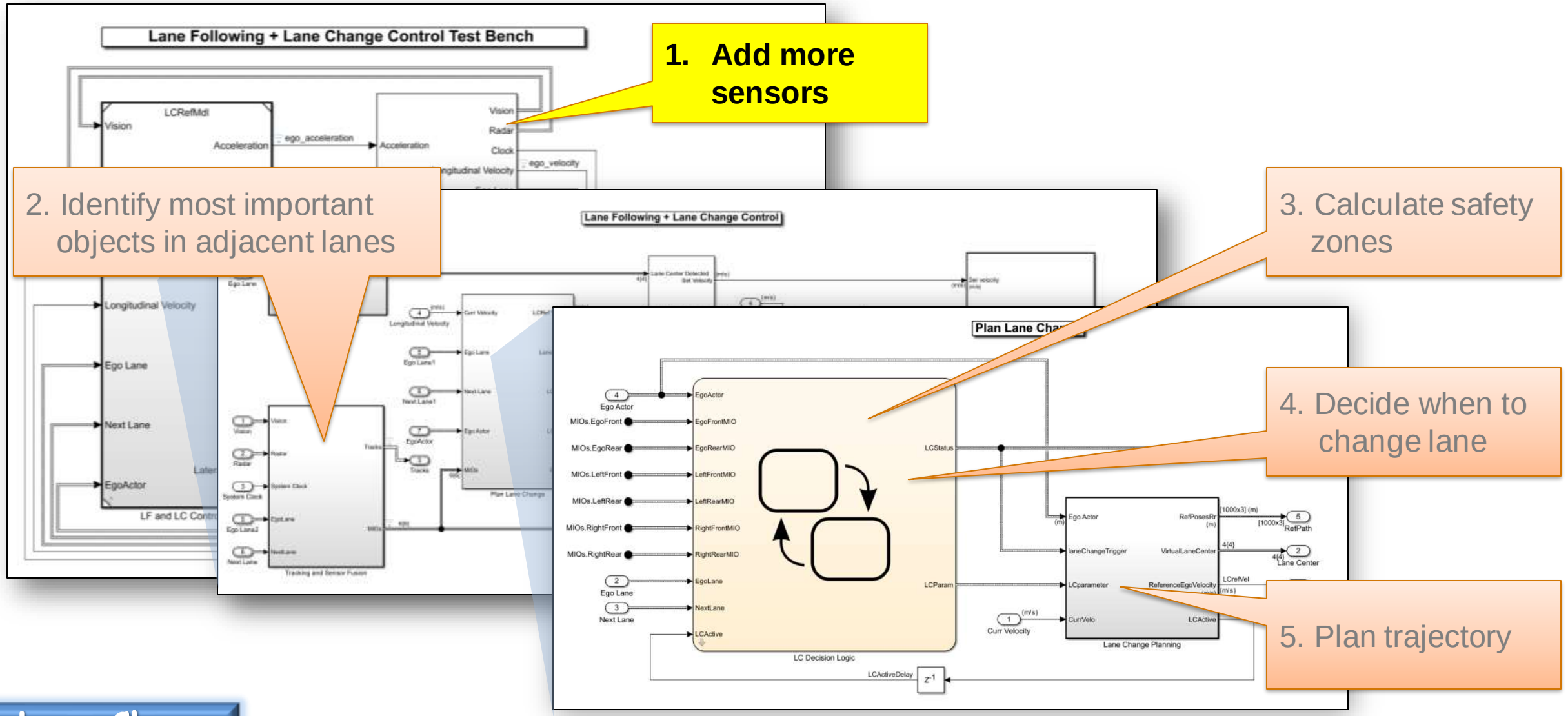


+ Lane Change

Add lane change functionality to lane following controller



Add lane change functionality to lane following controller



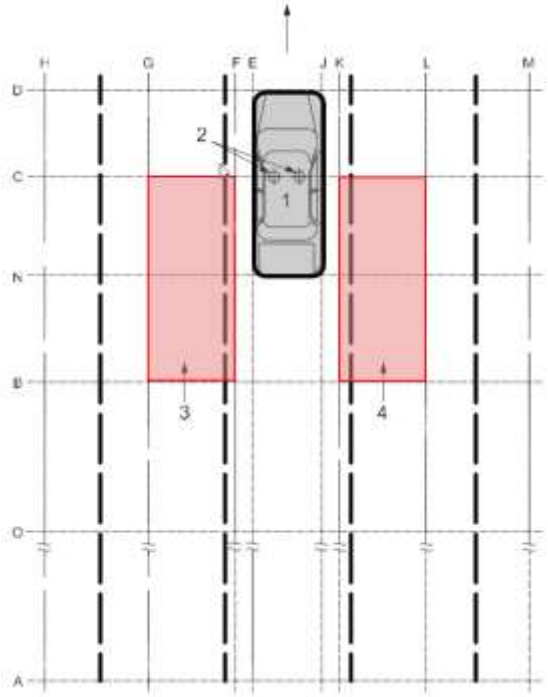
+ Lane Change

System requirements for lane change

Intelligent transport systems - Lane change decision aid systems (LCDAS)

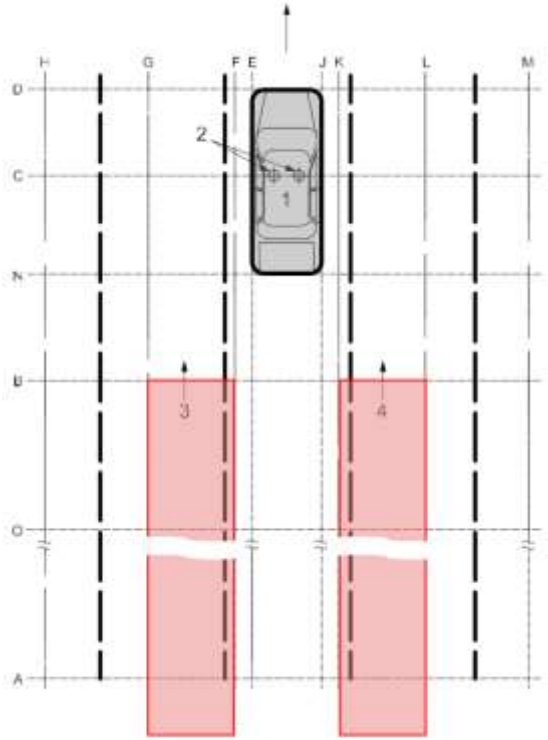
ISO 17387

Adjacent zones
for Blind Spot Detection



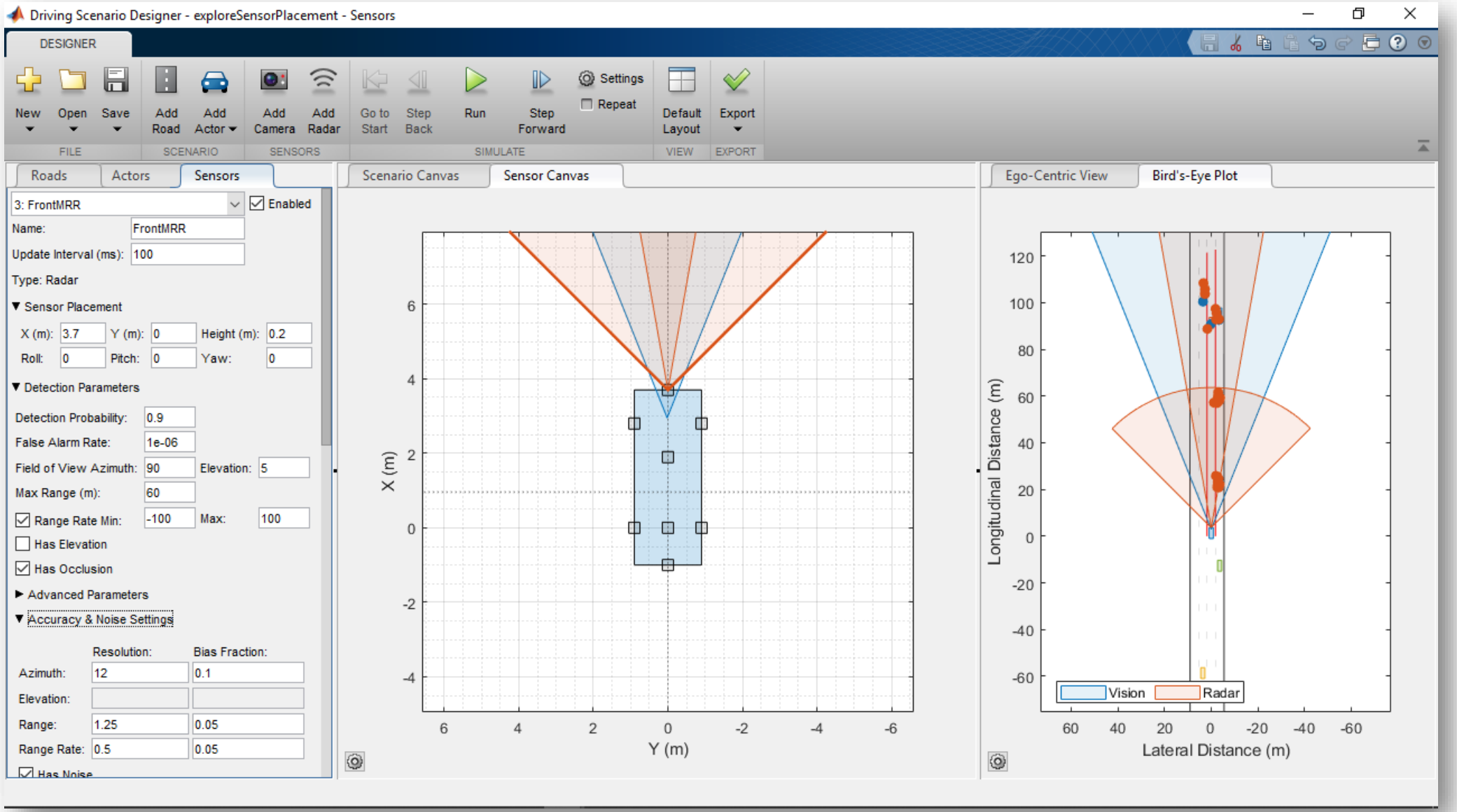
Typically implemented with
Short Range Radar

Rear zones
for closing vehicle warning



Typically implemented with
Mid Range Radar

Explore sensor placement with Driving Scenario Designer

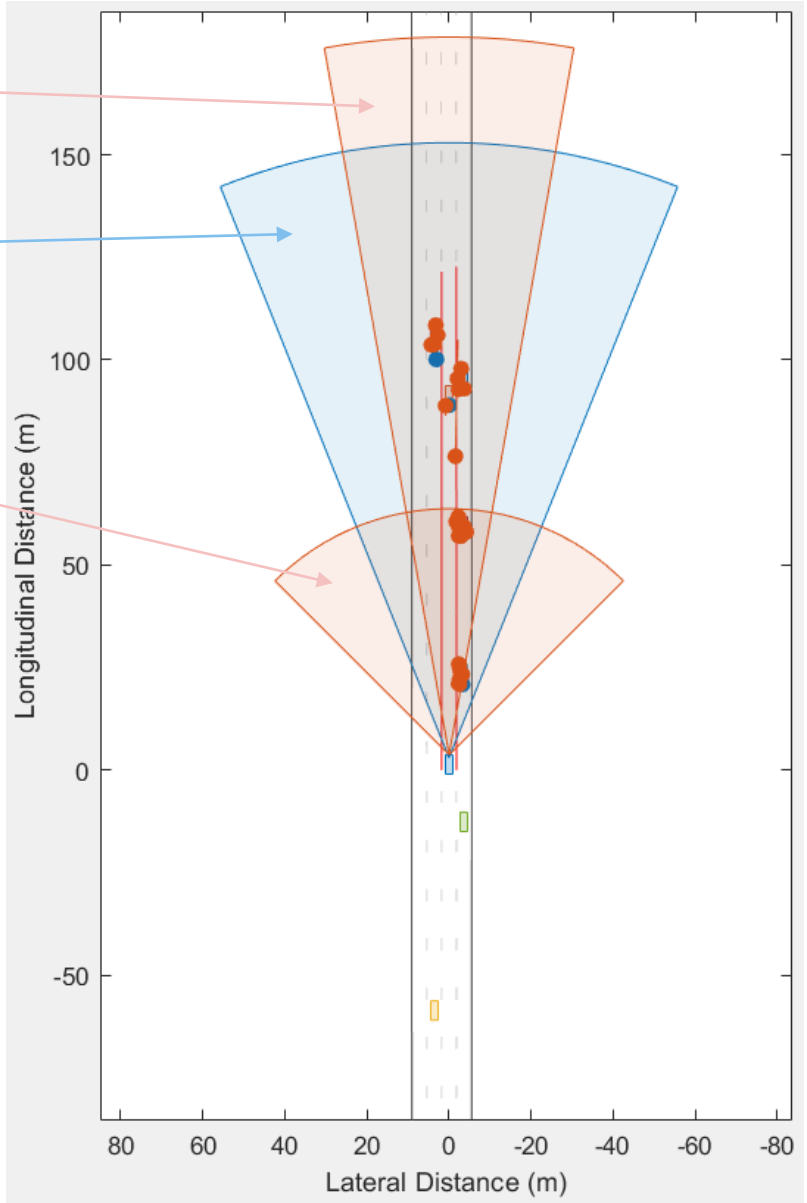


Review sensor configuration for **lane following** example

FrontLRR
FoV = 20° x 175m

Camera
FoV = 43.6° x 150m

FrontMRR
FoV = 90° x 60m



- SRR: Short-Range Radar
- MRR: Mid-Range Radar
- LRR: Long-Range Radar

Add rear looking sensors to support **left lane change**

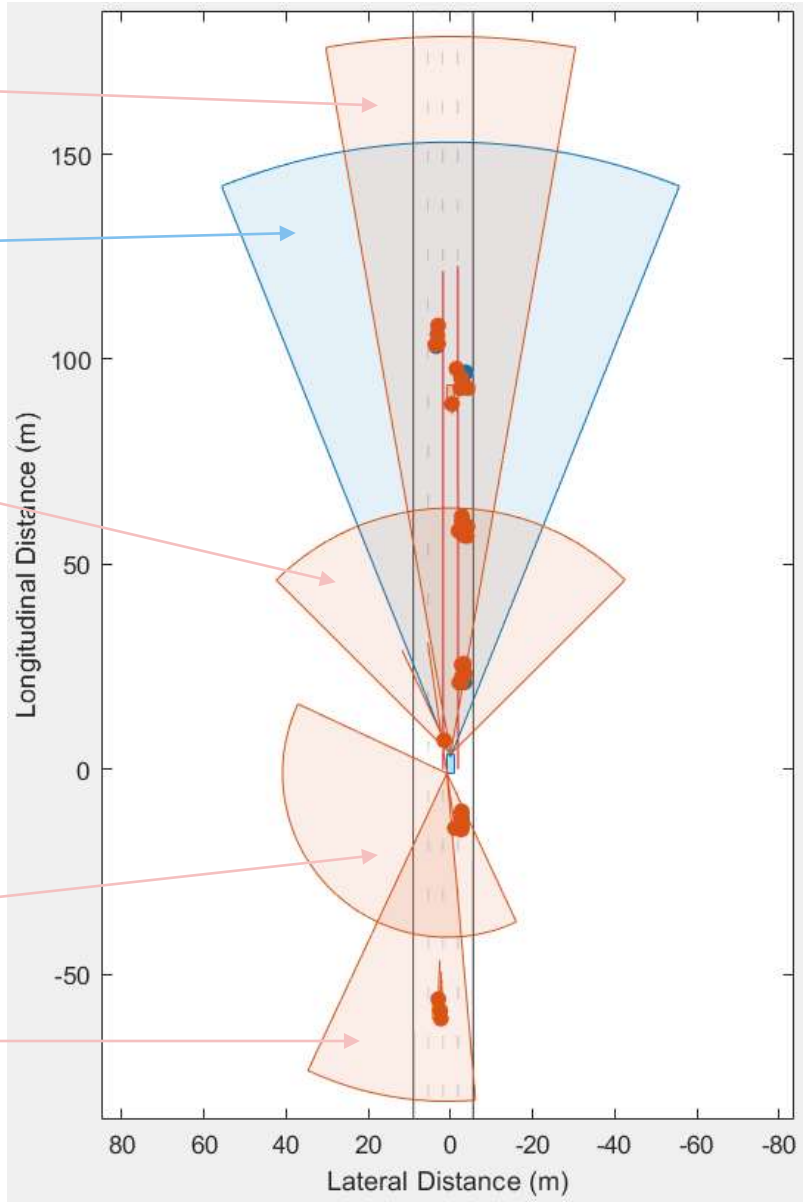
FrontLRR
FoV = 20° x 175m

Camera
FoV = 43.6° x 150m

FrontMRR
FoV = 90° x 60m

LeftRearSRR
FoV = 140° x 40m

LeftRearMRR
FoV = 30° x 80m



- SRR: Short-Range Radar
- MRR: Mid-Range Radar
- LRR: Long-Range Radar

Overall sensor configuration for lane following plus lane change

FrontLRR
FoV = 20° x 175m

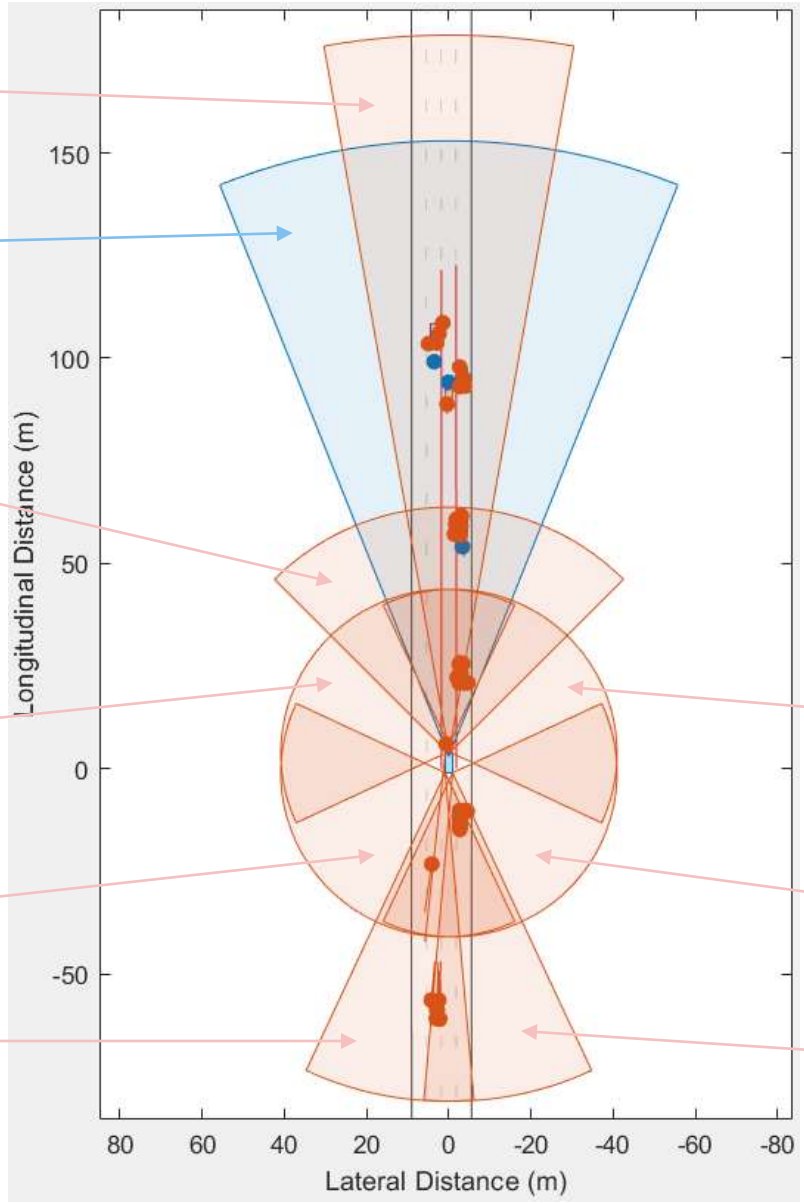
Camera
FoV = 43.6° x 150m

FrontMRR
FoV = 90° x 60m

LeftFrontSRR
FoV = 140° x 40m

LeftRearSRR
FoV = 140° x 40m

LeftRearMRR
FoV = 30° x 80m



×1 front camera
×8 radars

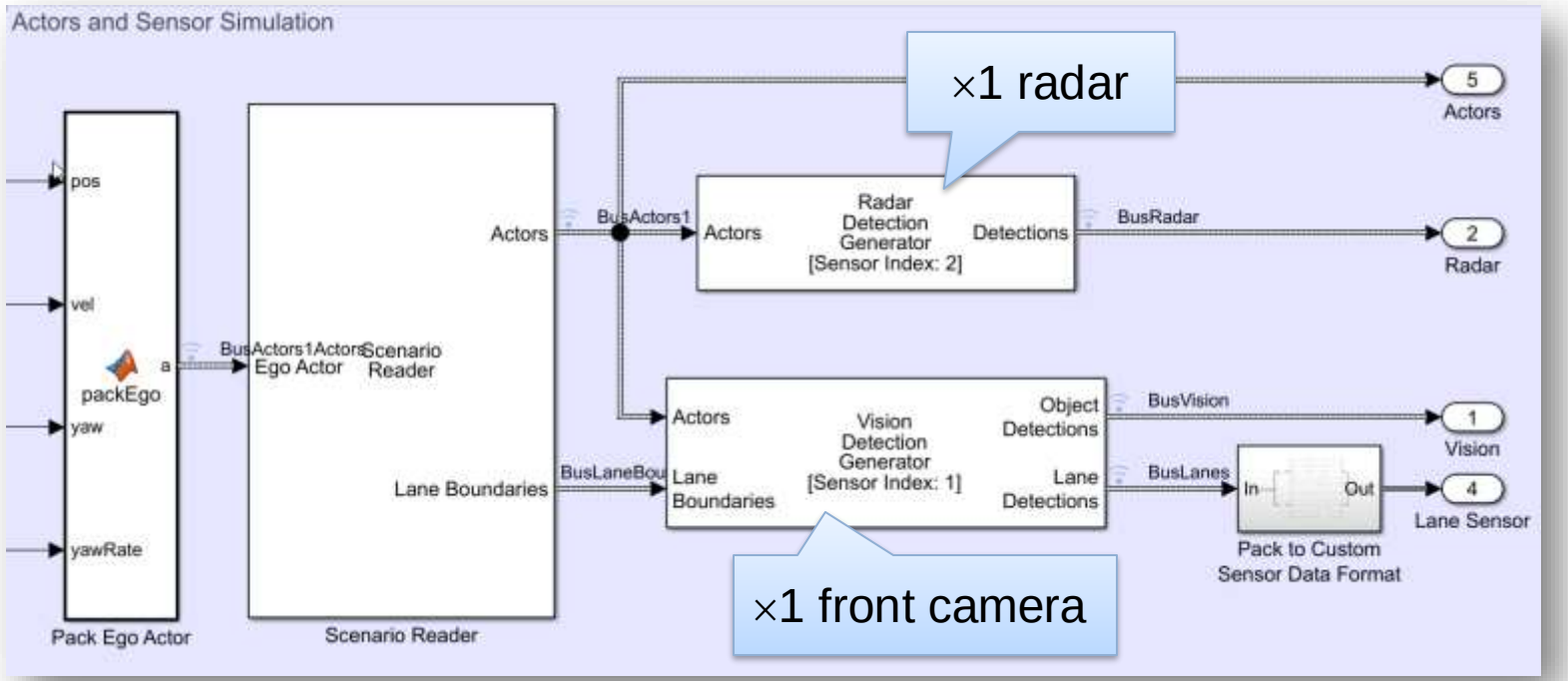
- SRR: Short-Range Radar
- MRR: Mid-Range Radar
- LRR: Long-Range Radar

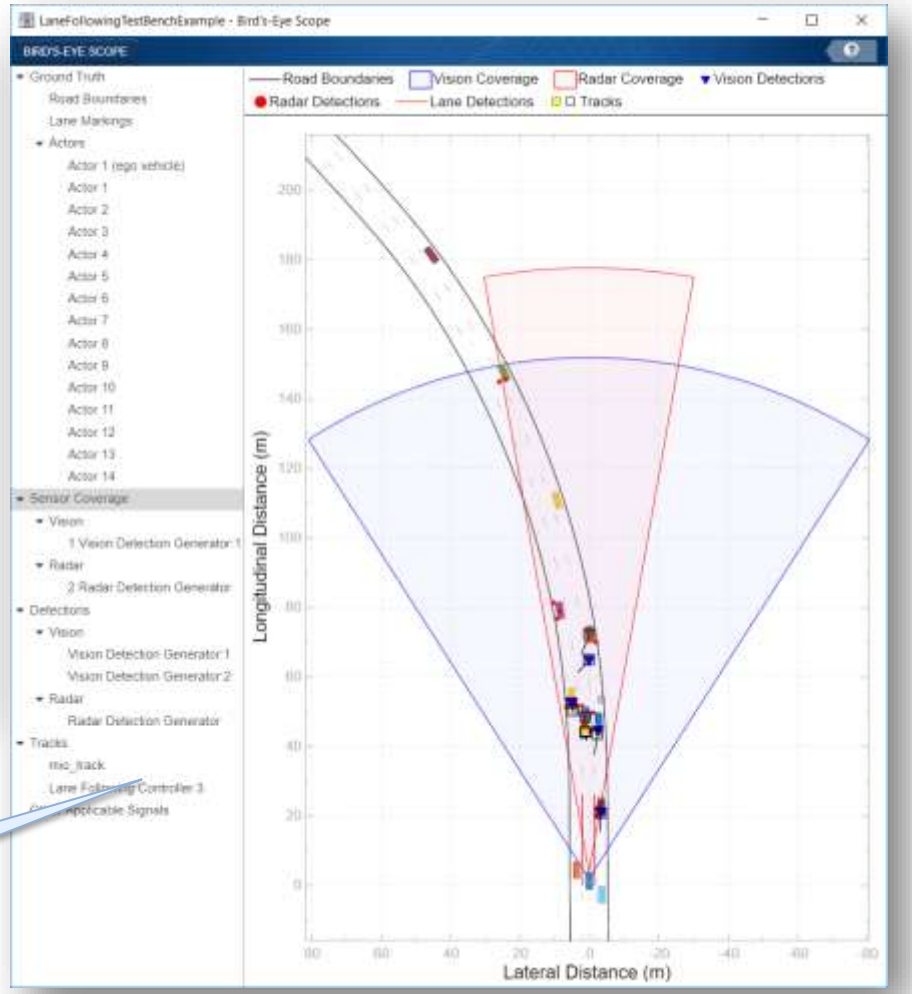
RightFrontSRR
FoV = 140° x 40m

RightRearSRR
FoV = 140° x 40m

RightRearMRR
FoV = 30° x 80m

Review sensor models for lane following controller

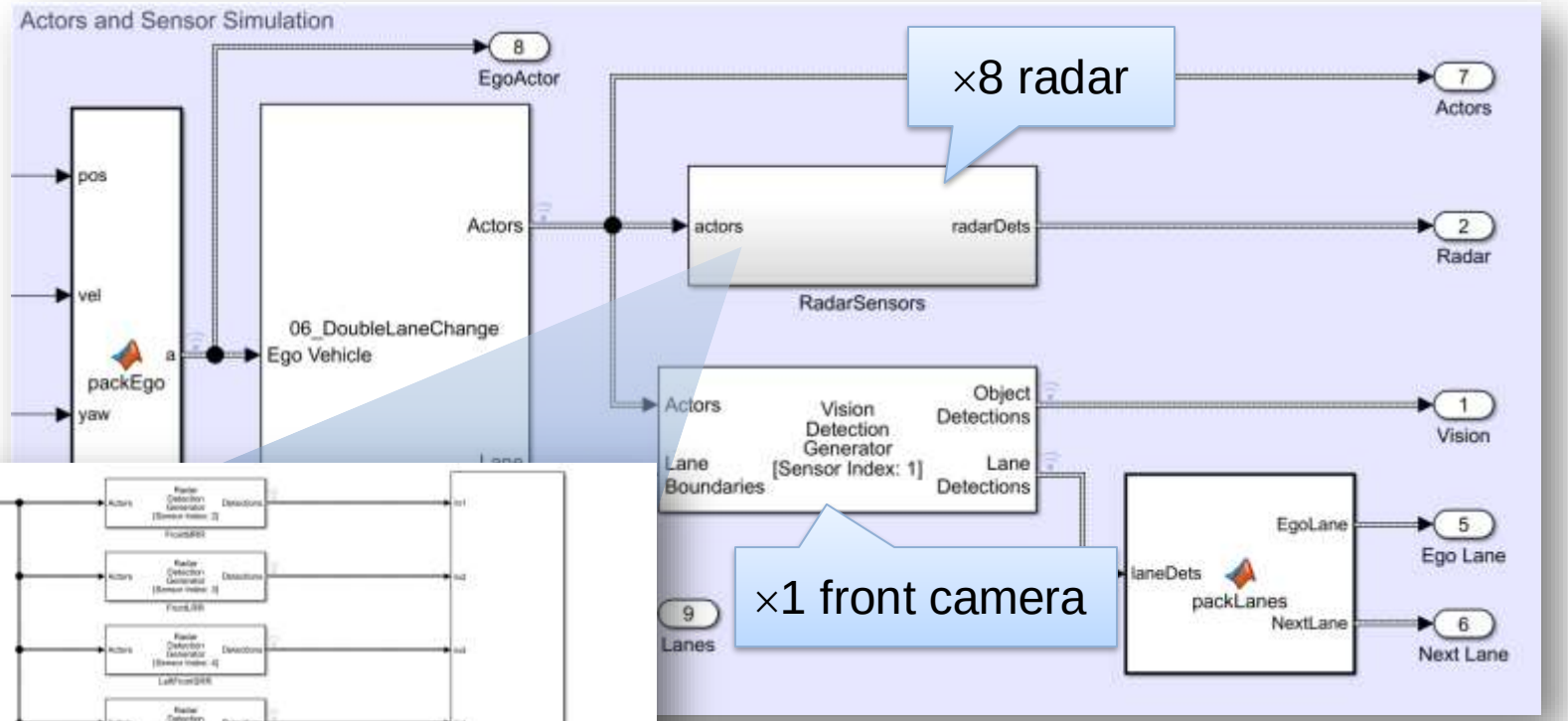


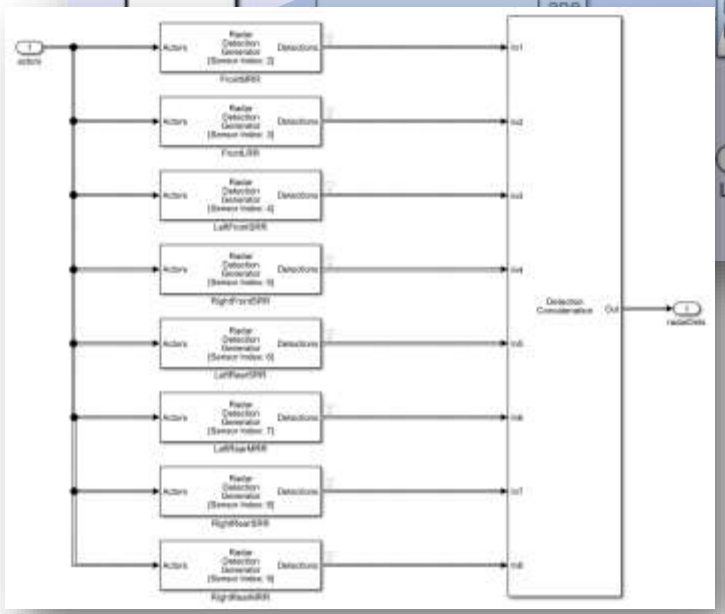


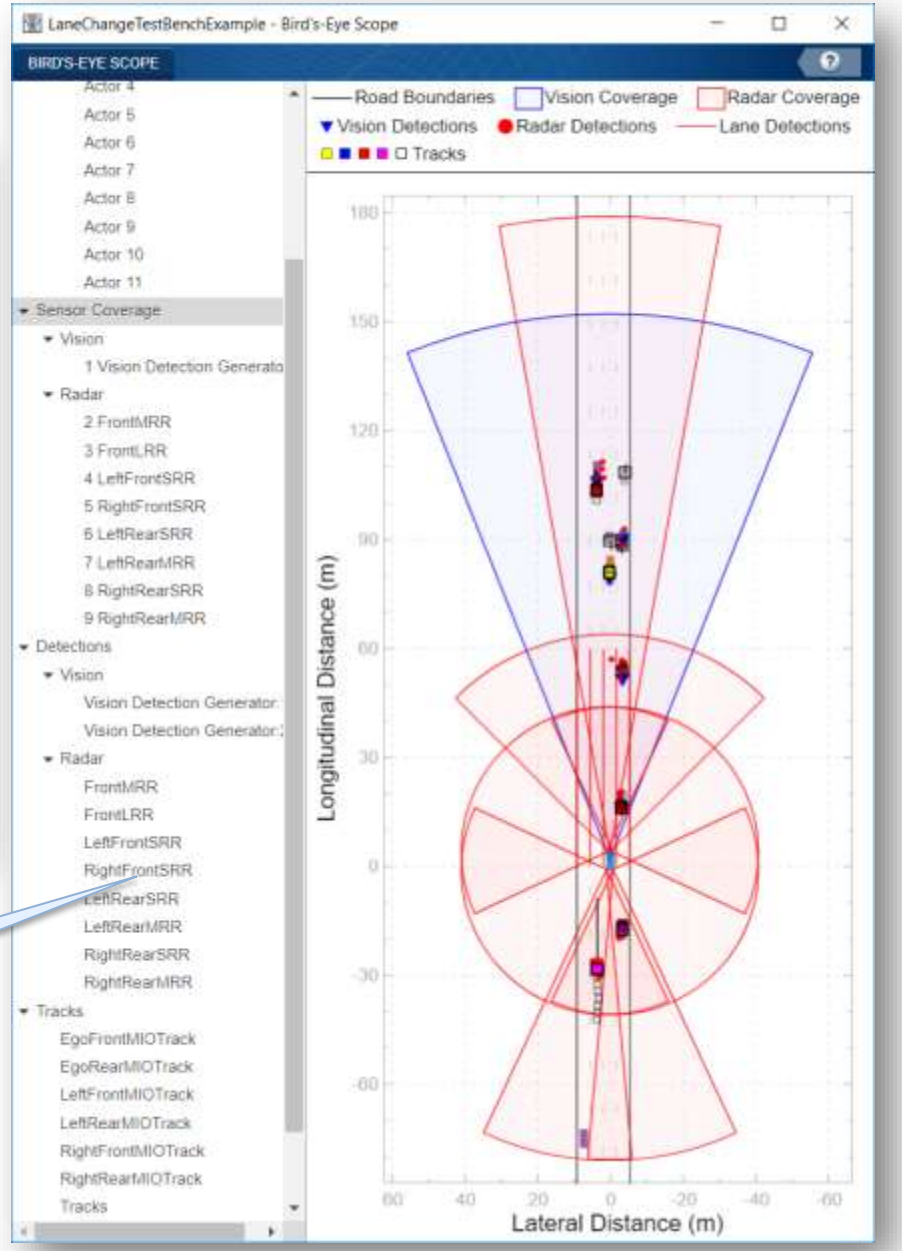
Visualize with
Birds Eye Scope

Baseline example

Add sensor models for lane change



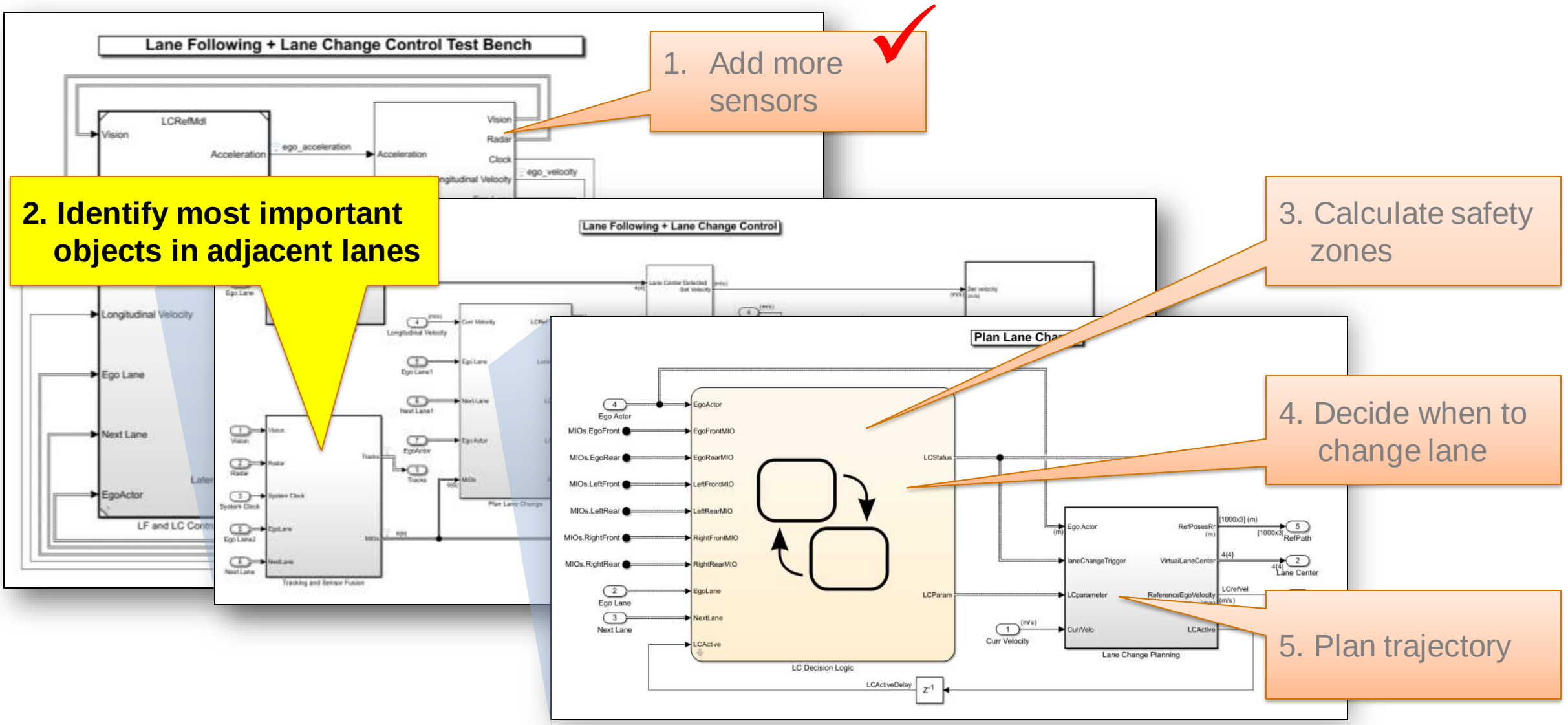




Visualize with Birds Eye Scope

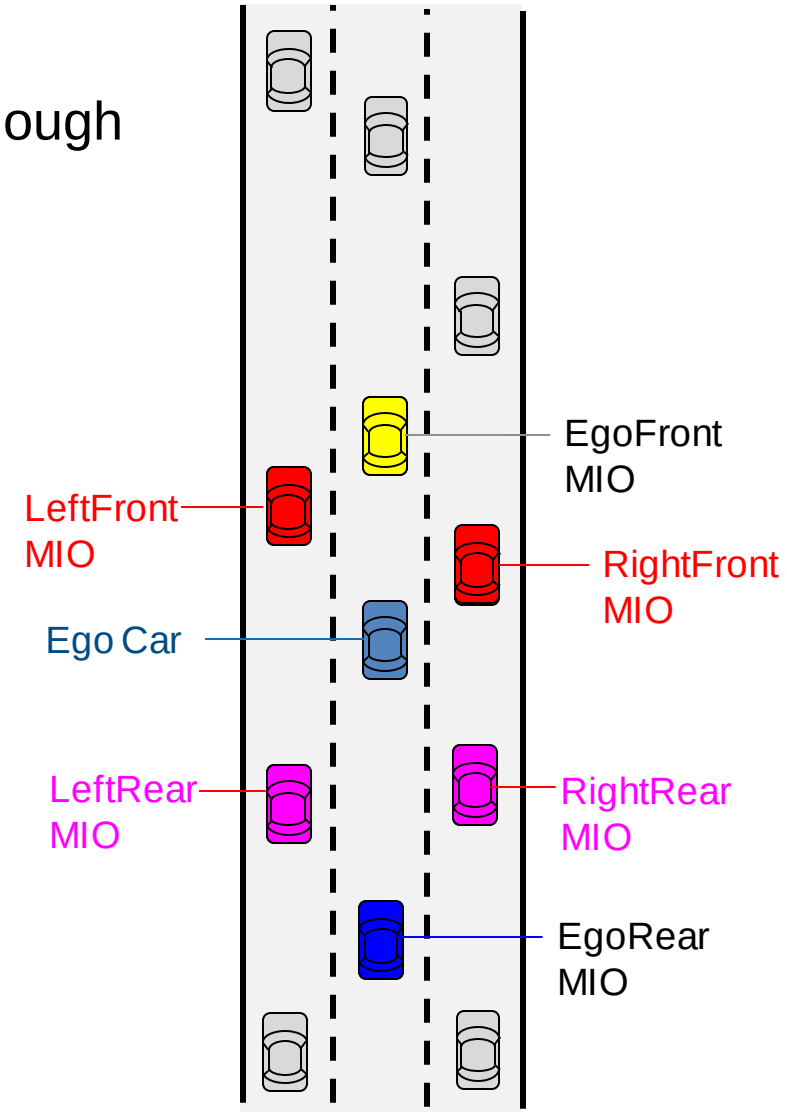
+ Lane Change

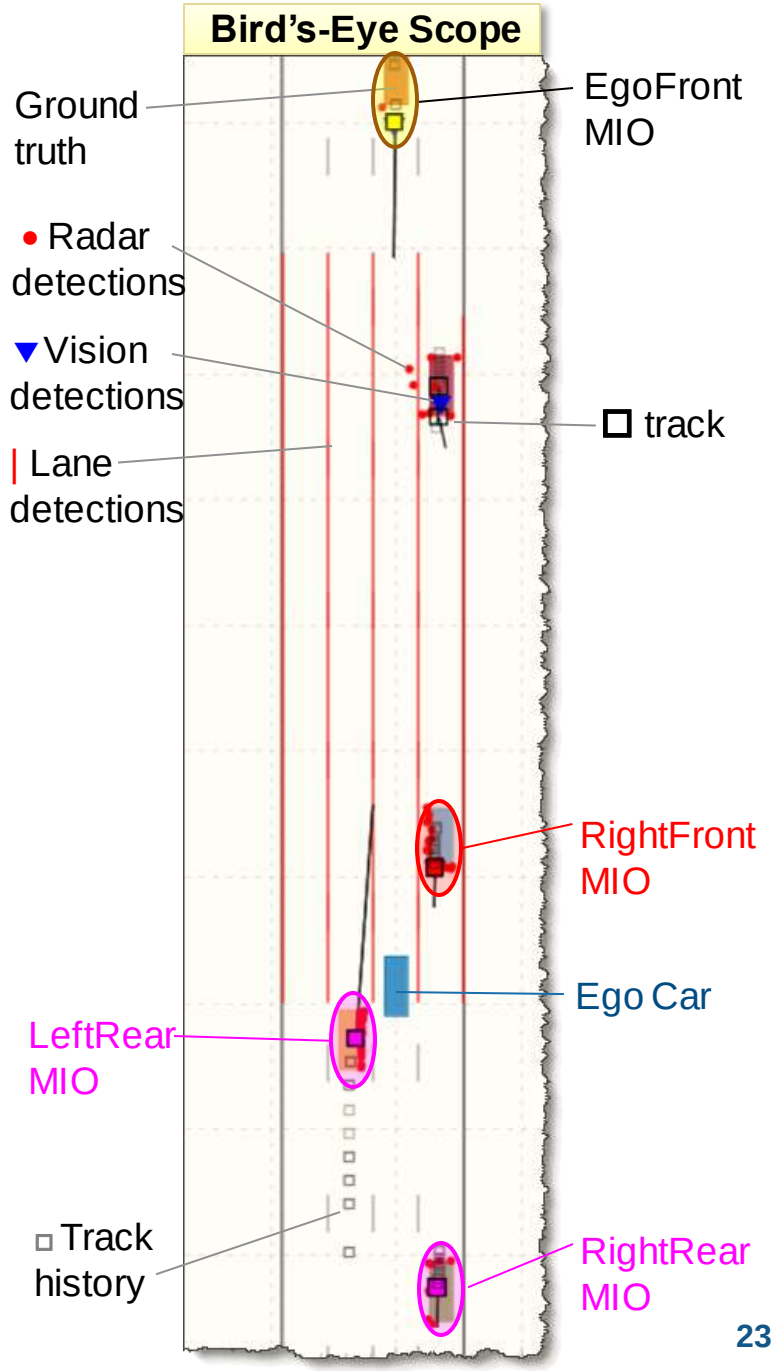
Add lane change functionality to lane following controller



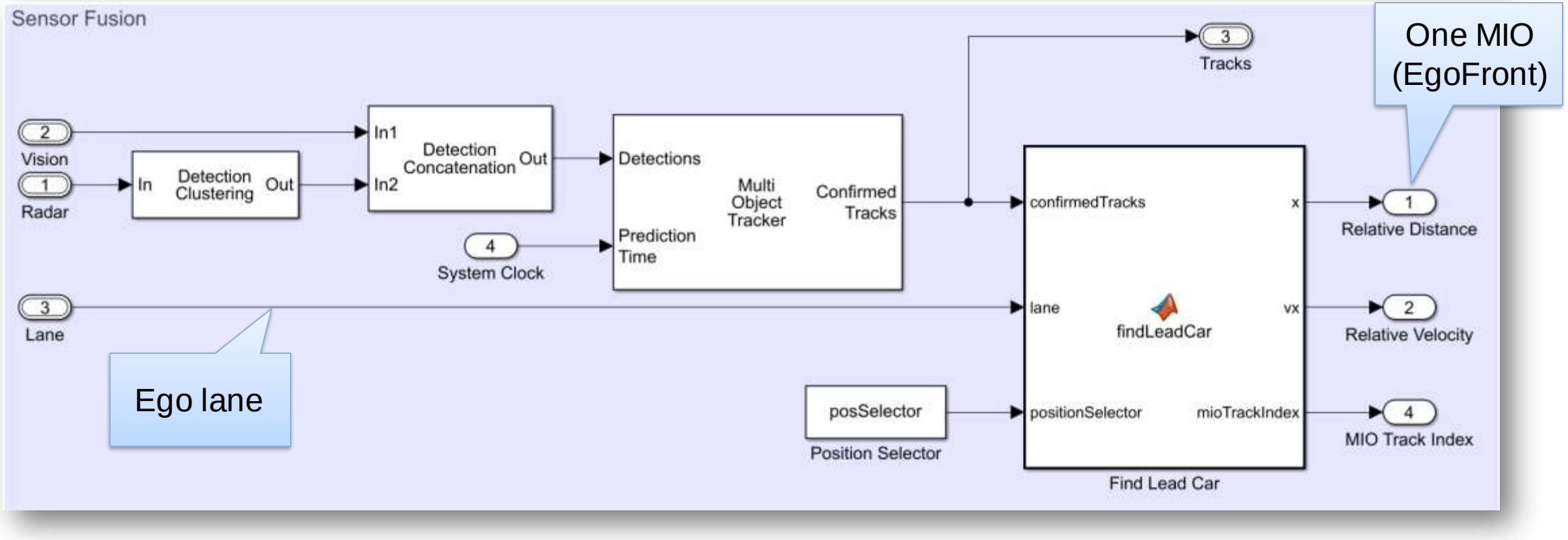
Identify Most Important Objects (MIO) to detect

- **Lane following**
  - one EgoFront MIO is enough
- **Lane change**
  - needs more MIOs surrounding ego car



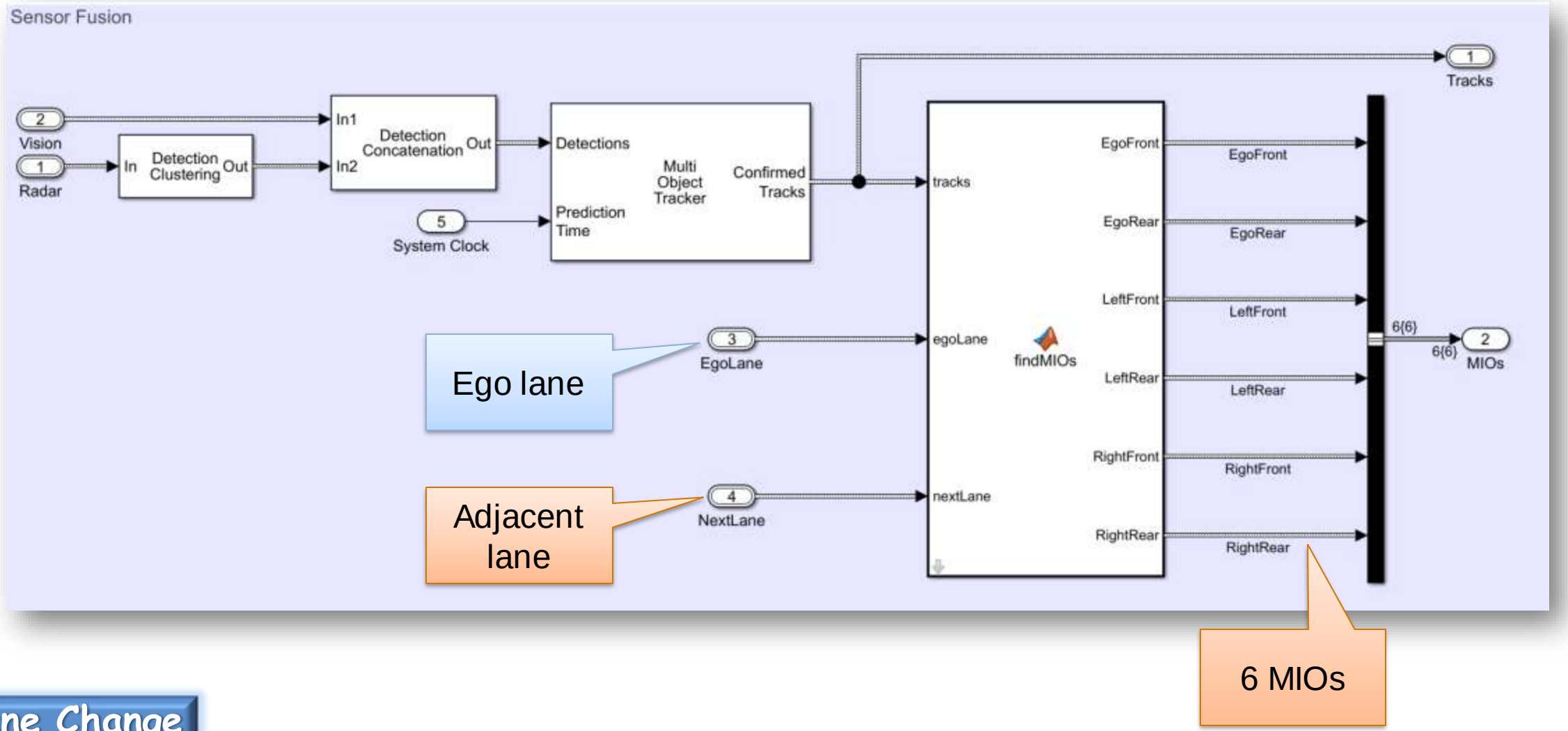


Review baseline MIO detector architecture for lane following controller



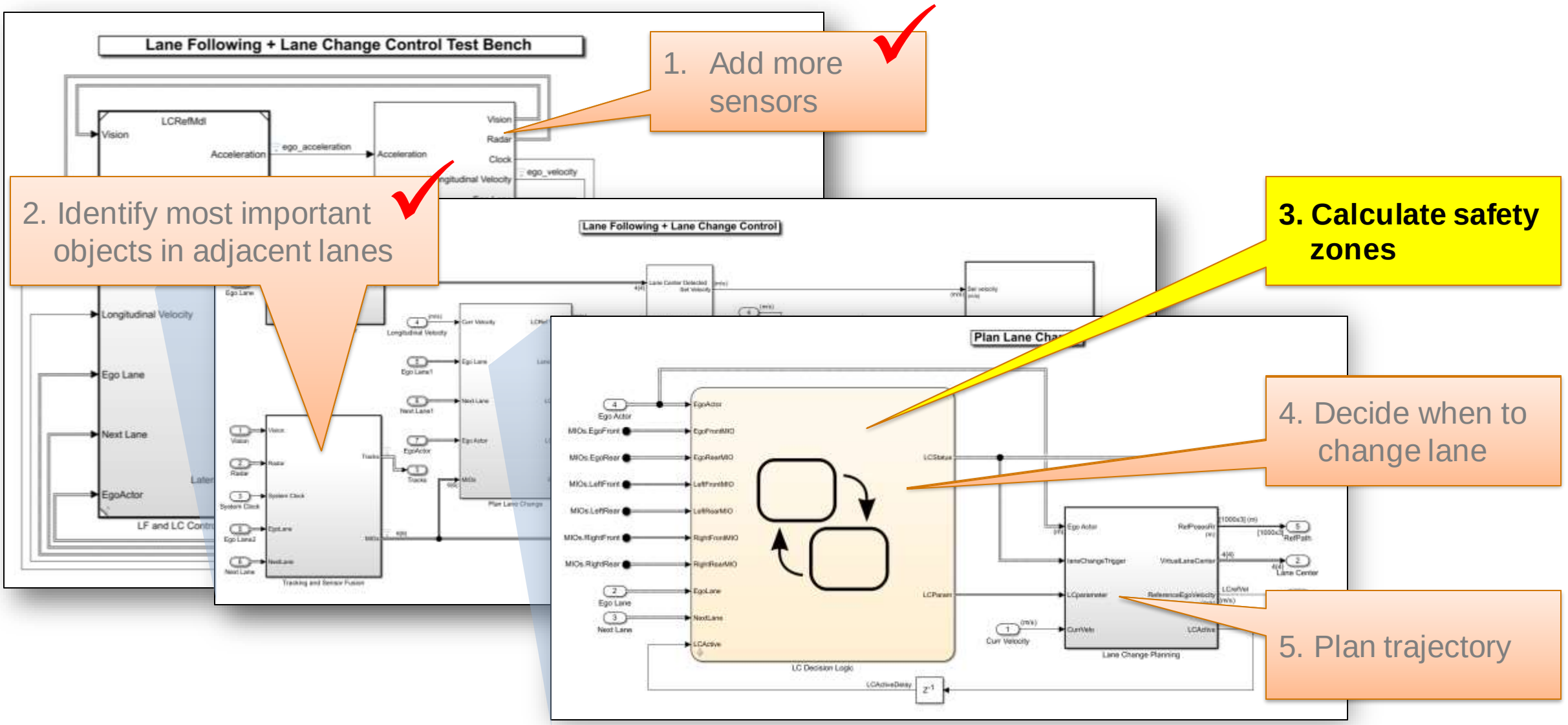
Baseline example

Add MIO detectors for lane change



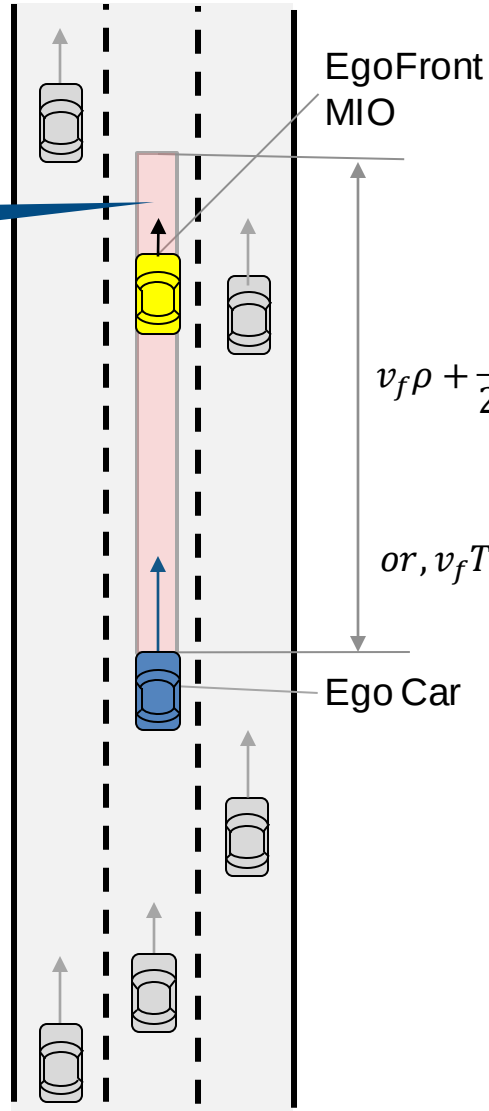
+ Lane Change

Add lane change functionality to lane following controller



Calculate safety zones

Safety Zone
for lane following



$v_f\rho + \frac{v_f^2}{2a_{brake}}$

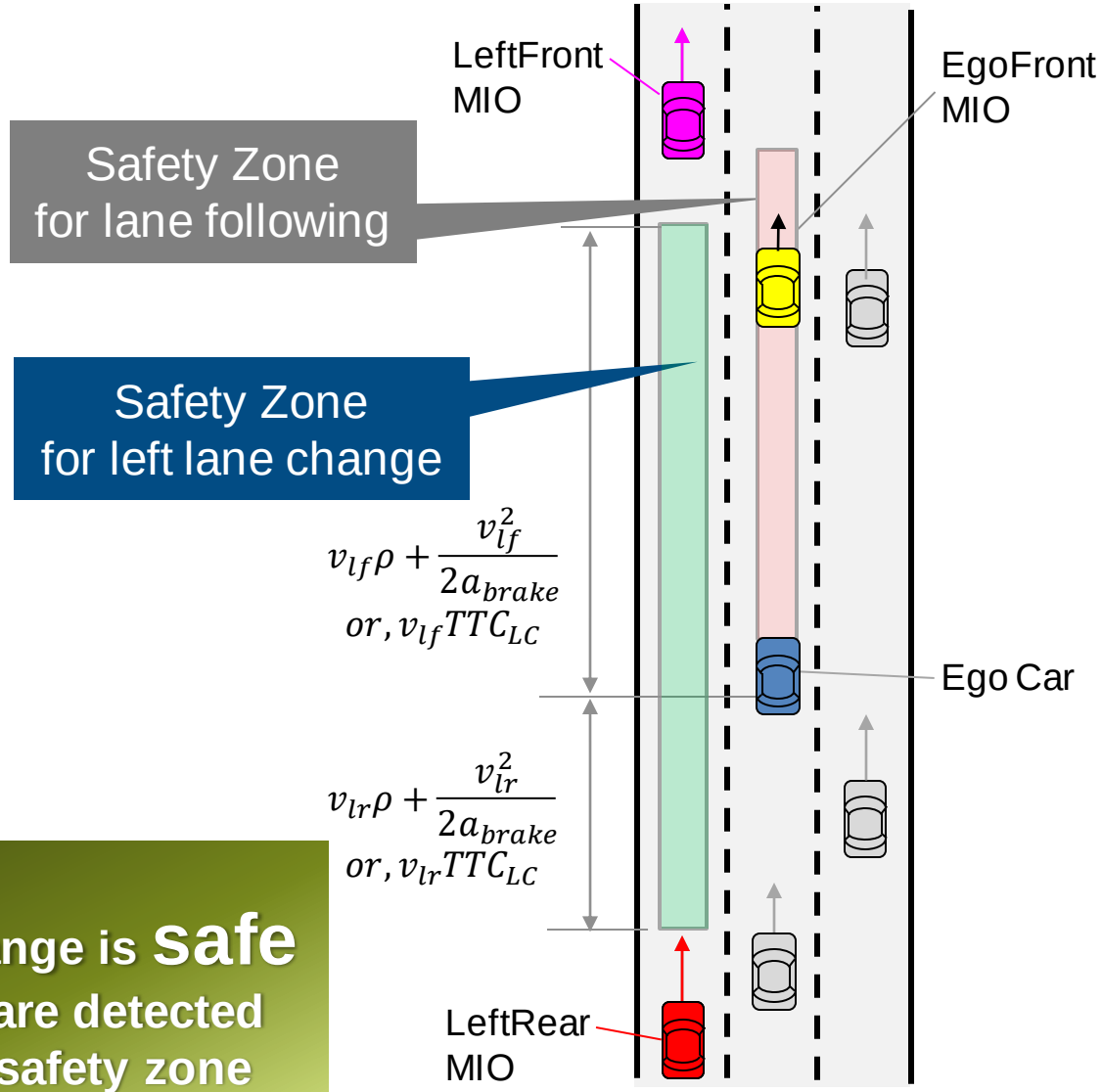
$or, v_f TTC_{FCW}$

Lane following is unsafe if ego front MIO is detected within the safety zone

Longitudinal safe distance
= travel distance during response time(ρ)
+ braking distance with a_{brake}

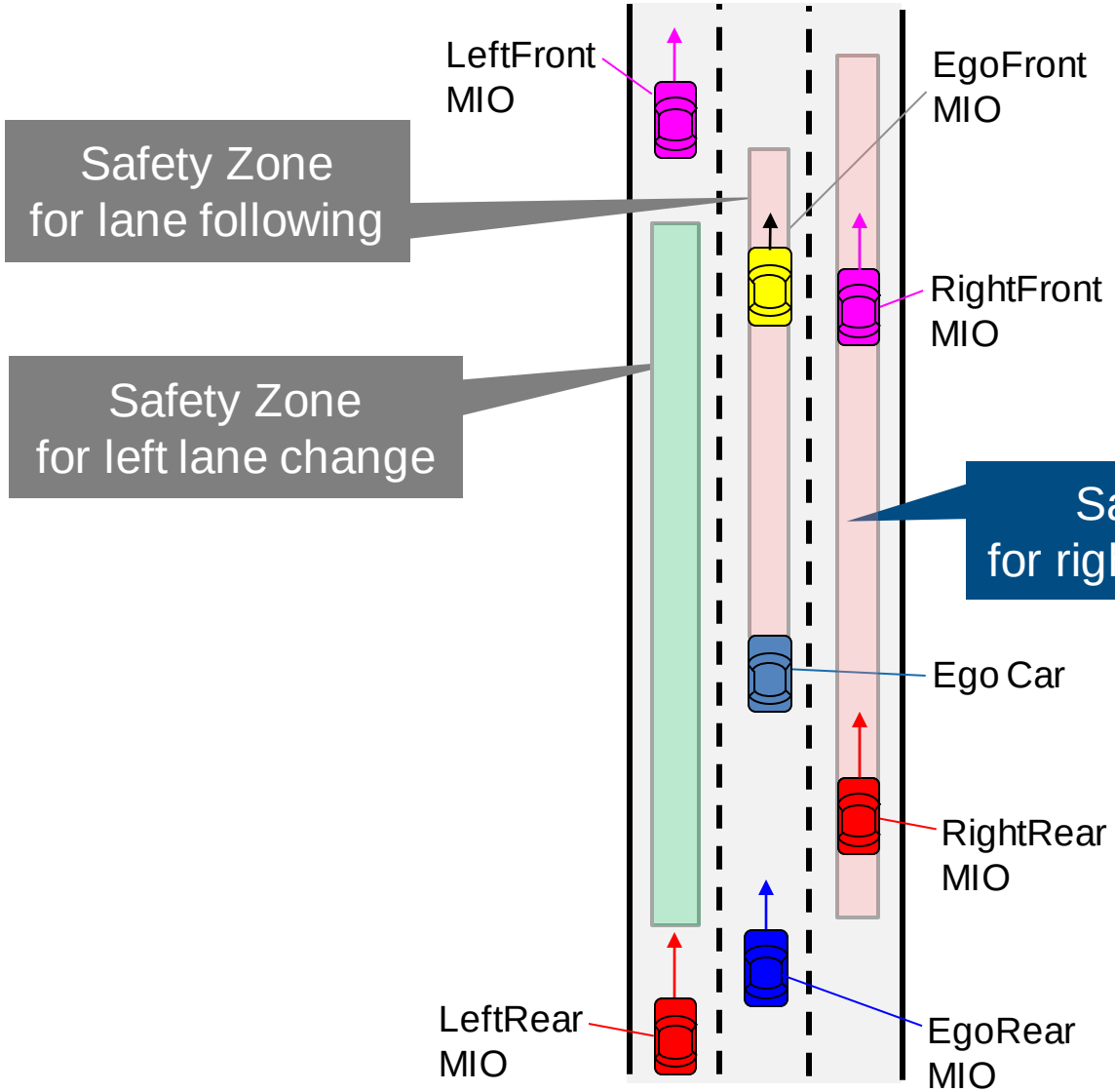
TTC_{FCW} : Time-to-Contact

Calculate safety zones



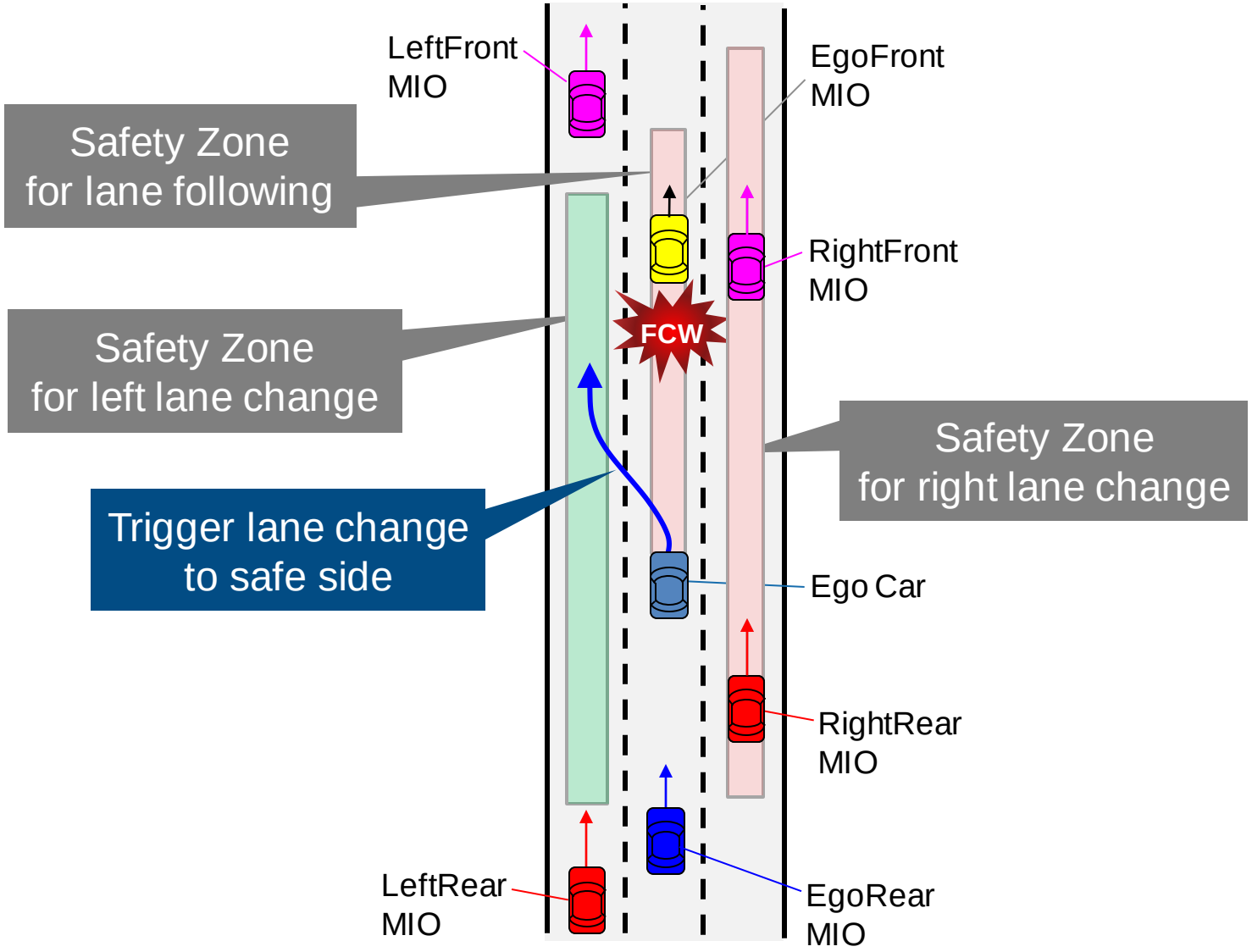
Left lane change is safe if no MIOs are detected within the safety zone

Calculate safety zones

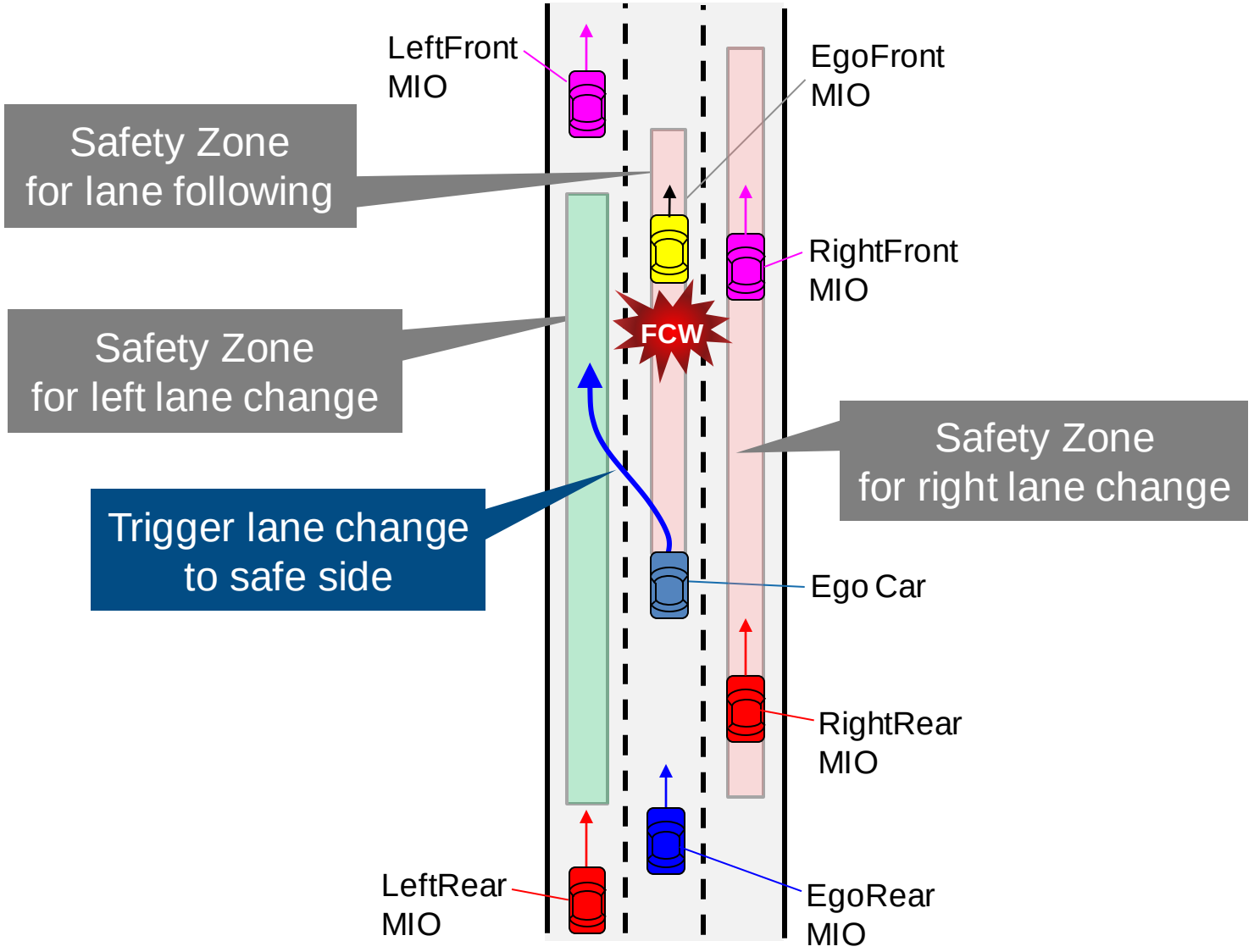


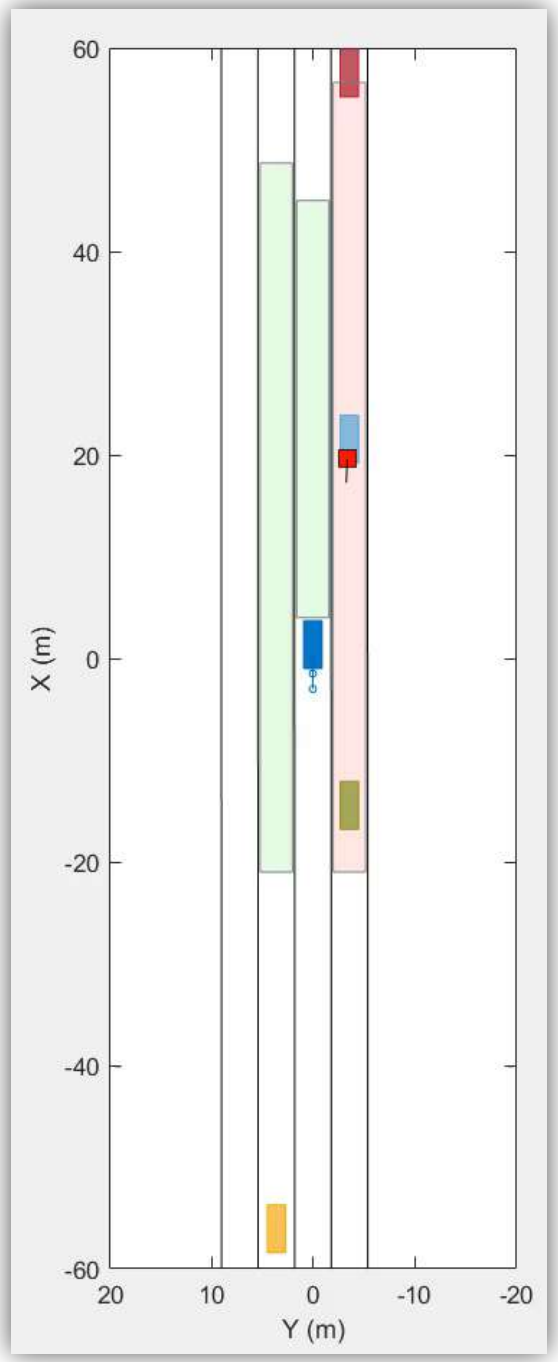
Right lane change is unsafe if MIOs are detected within the safety zone

Calculate safety zones

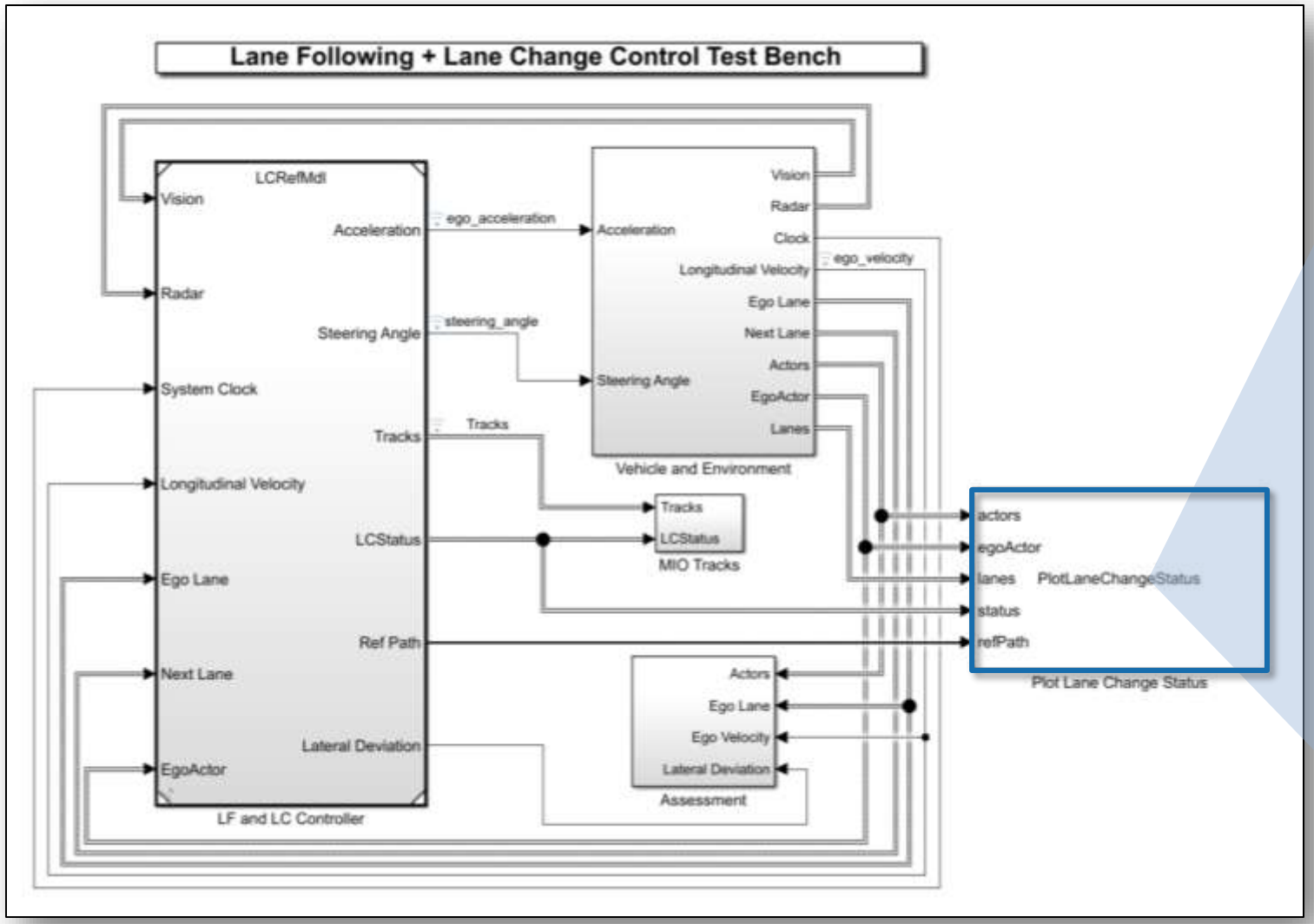


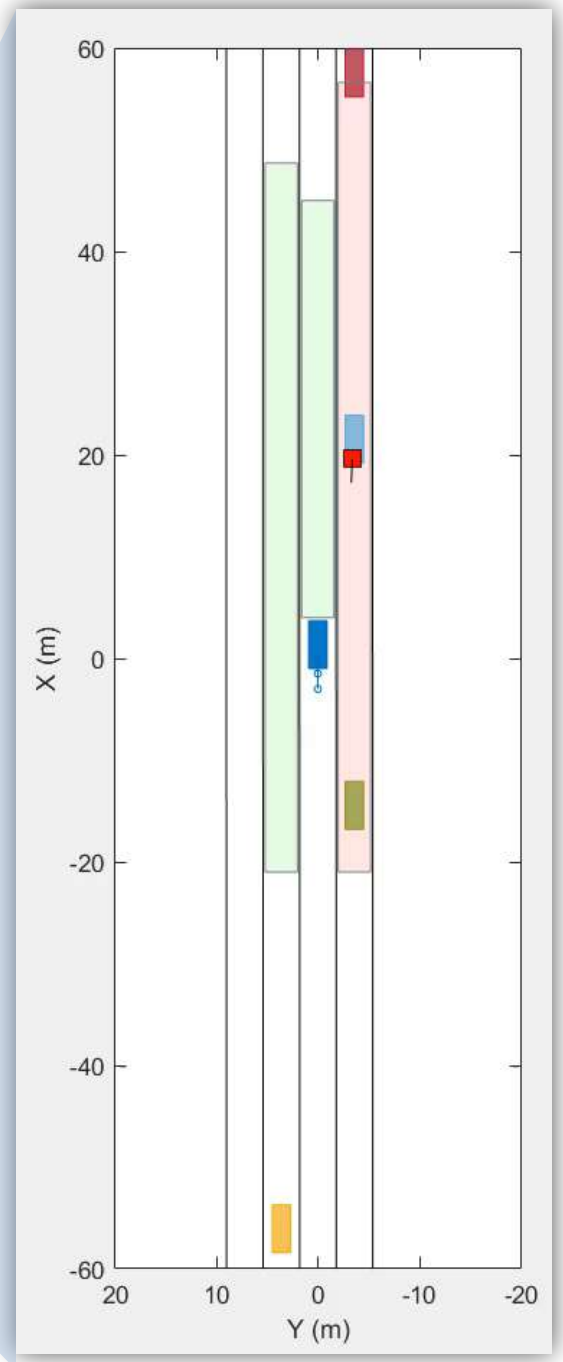
Visualize safety zones



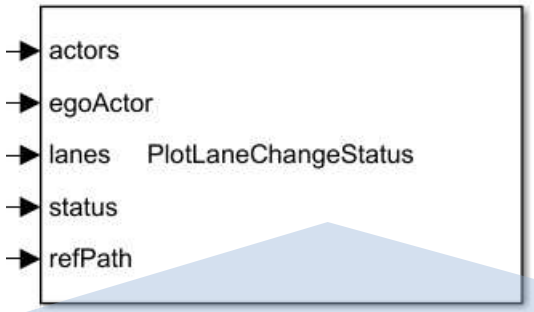


Visualize safety zones and trajectory



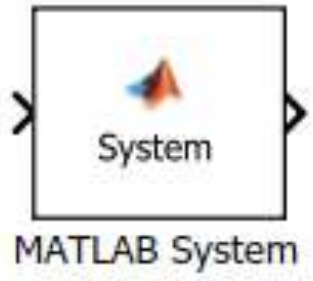


Create custom visualization for safety zones and trajectory



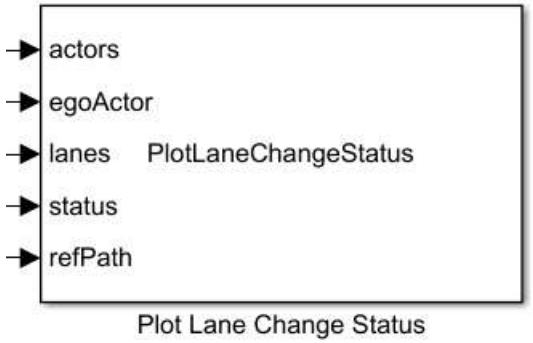
PlotLaneChangeStatus.m

```
1   classdef PlotLaneChangeStatus < matlab.System
2       % Custom helper visualization to show status of MIOs, safety zones, and
3       % ego trajectory during lane change
4
5       properties(Access = private)
6           Figure
7           BEP
8           OutlinePlotter
9           LaneBoundaryPlotter
10          SafeMIOPlotter
11          UnSafeMIOPlotter
12          ActorPatches
13          ZoneFront
14          ZoneLeft
15          ZoneRight
16          EgoTrace
17          EgoPath
18          LCPath
```



- The MATLAB System block brings existing System objects (based on matlab.System) into Simulink®

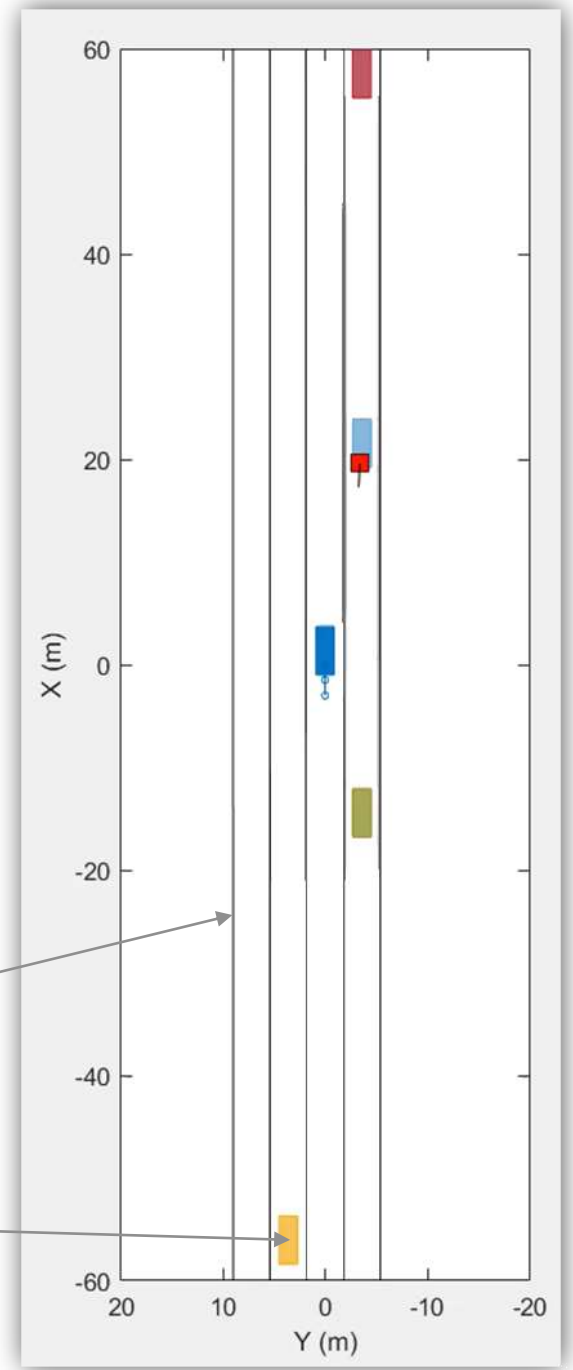
Create birds eye plot with utilities from Automated Driving Toolbox



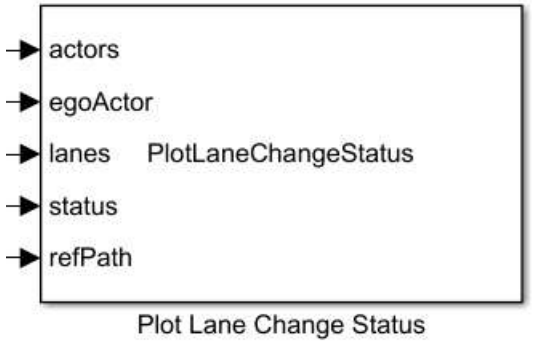

```
% create birds eye plot
obj.BEP = birdsEyePlot('Parent', hax,...
    'XLimits', [-60, 60],...
    'YLimits', [-20, 20]);
```

```
% create lane plotter
obj.LaneBoundaryPlotter = laneBoundaryPlotter(obj.BEP,...
    'DisplayName','Lane boundaries');
```

```
% create outline plotter for target actors
obj.OutlinePlotter = outlinePlotter(obj.BEP);
```

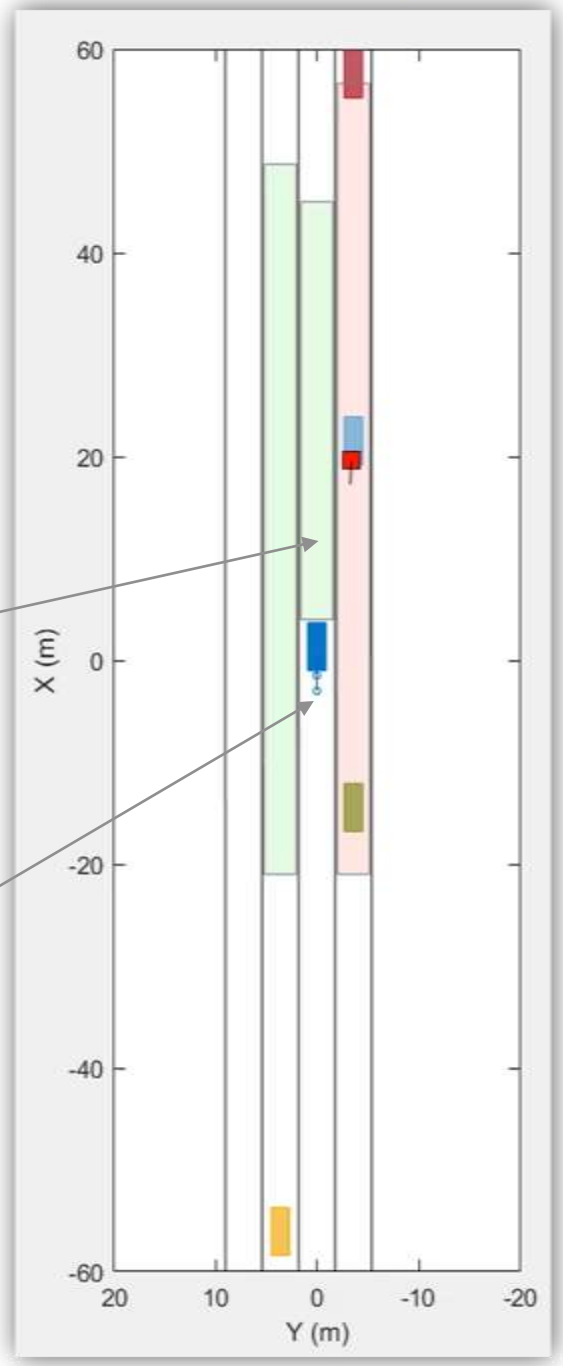


Plot safety zones and trajectory with MATLAB

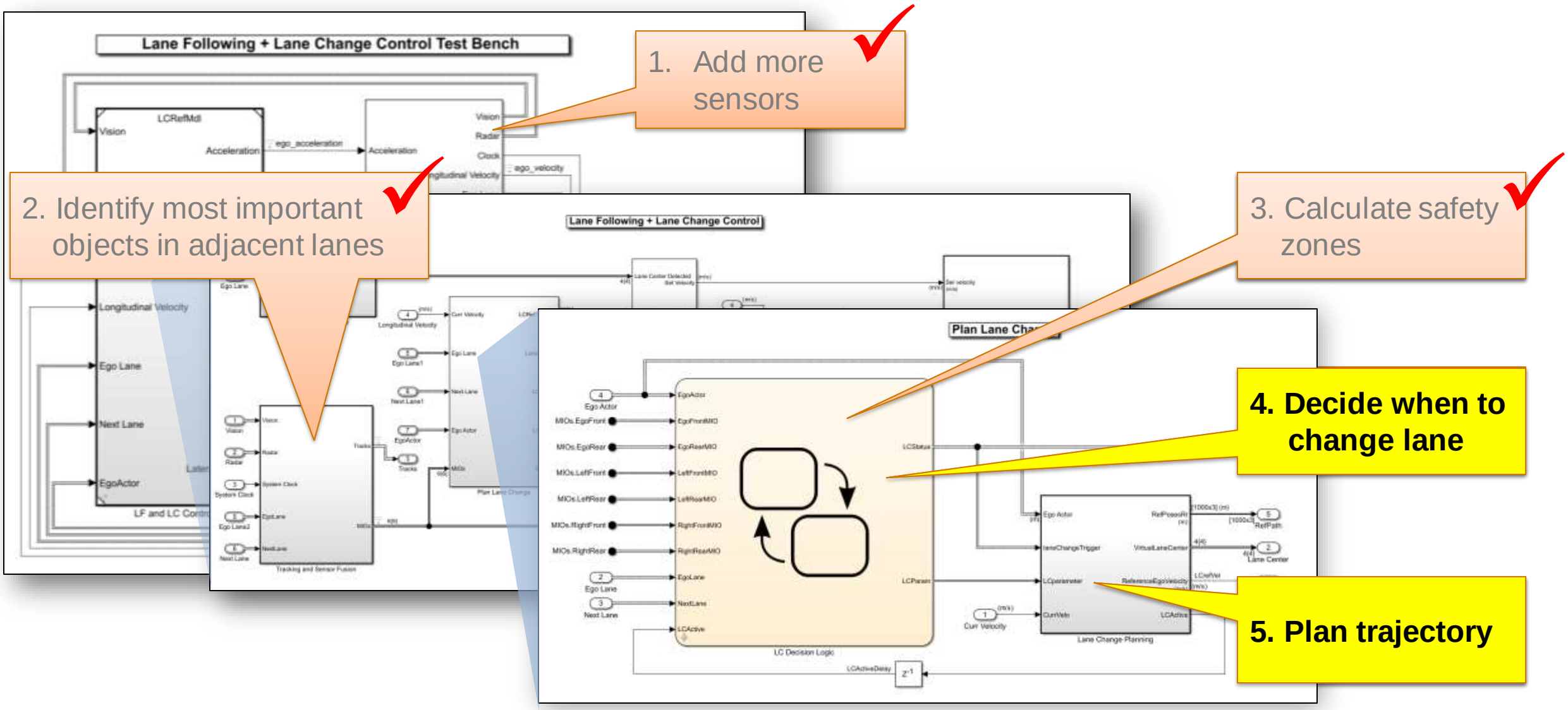



```
% create patches for safety zones
obj.ZoneFront = patch(hax,0,0,[0 0 0]);
set(obj.ZoneFront,'XData',[],'YData',[],...
    'FaceColor','green','FaceAlpha',0.1);
```

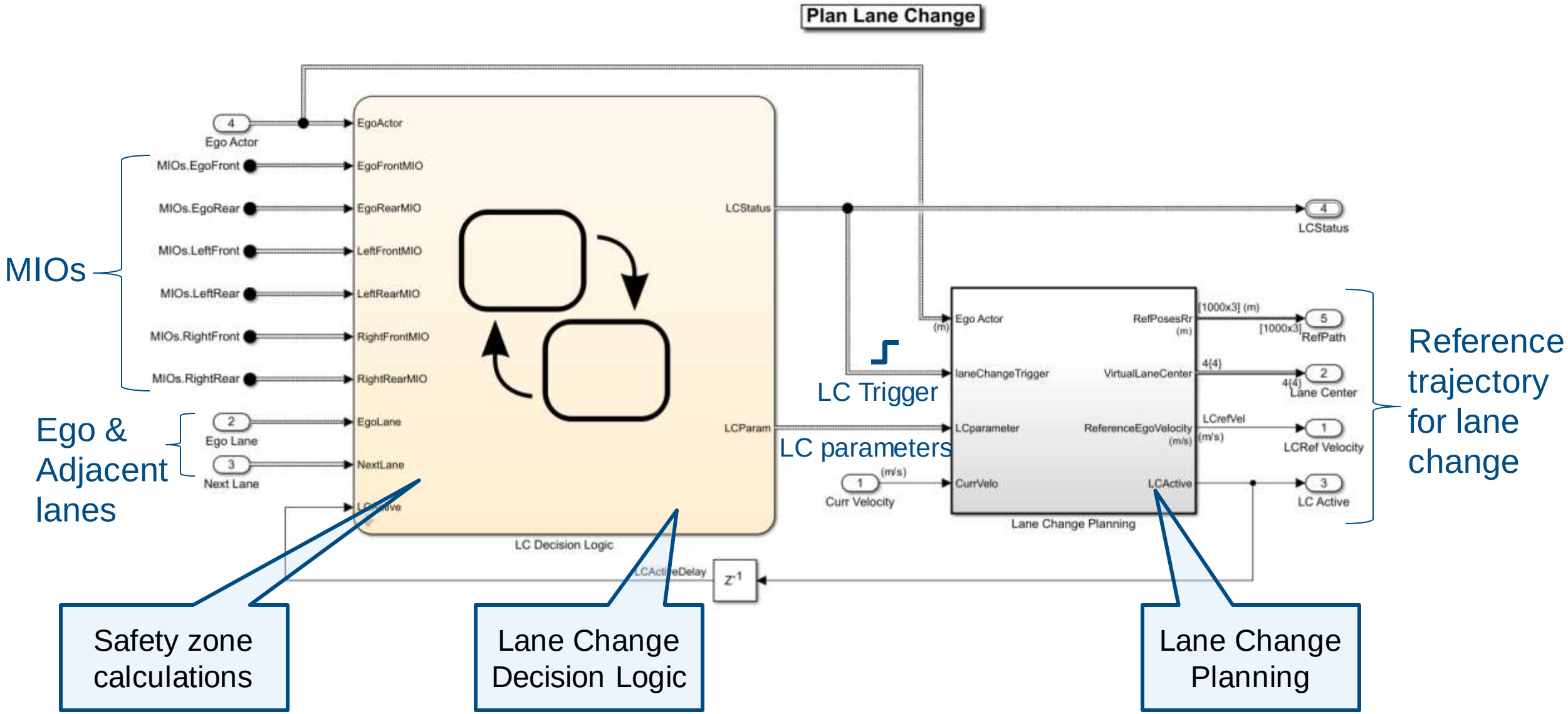
```
% create line for trajectory path
obj.LCPath = line(hax, 0, 0,...
    'Color','blue',...
    'LineWidth',2,...
    'LineStyle','-');
```



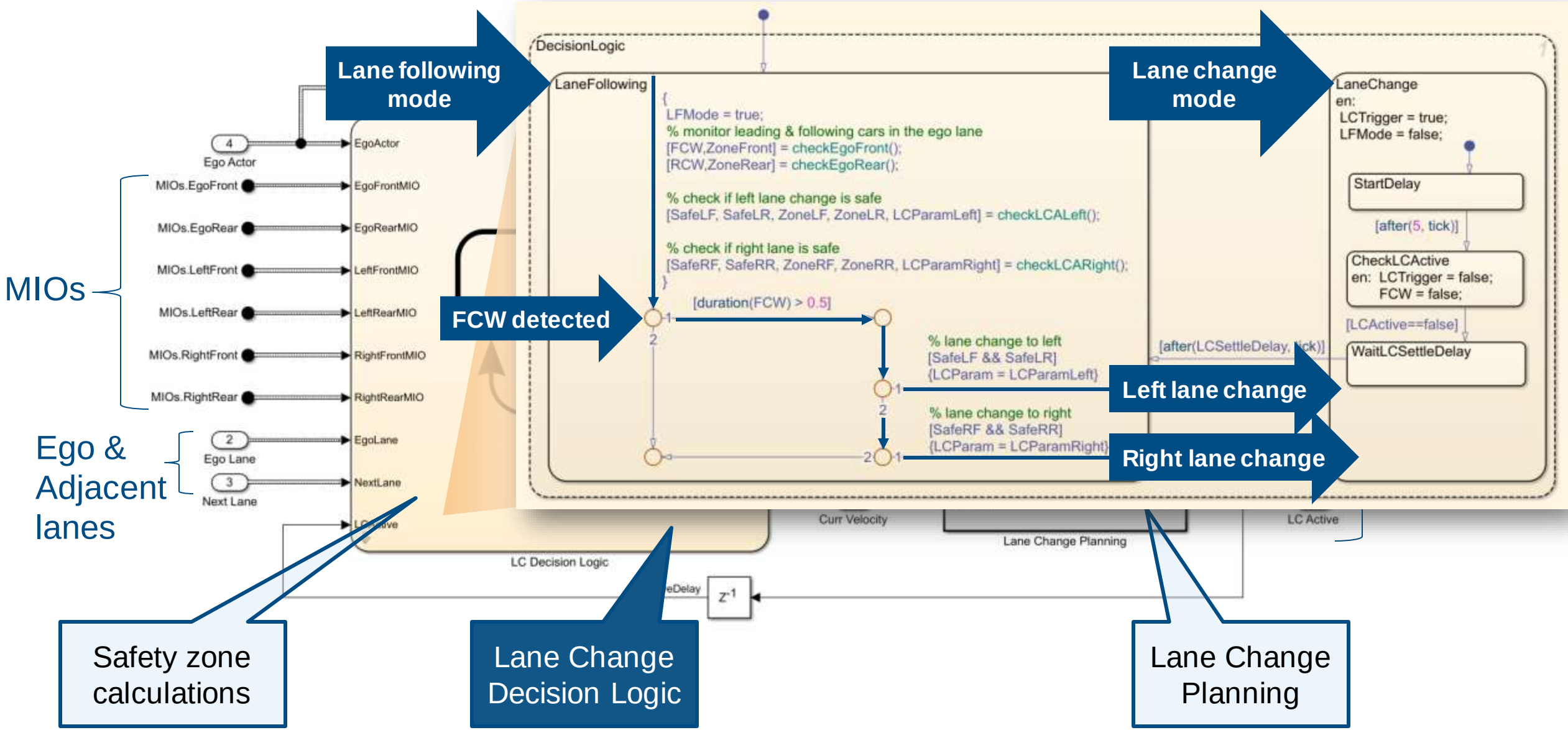
Add lane change functionality to lane following controller



Lane change decision logic and planning



Design lane change decision logic using Stateflow™

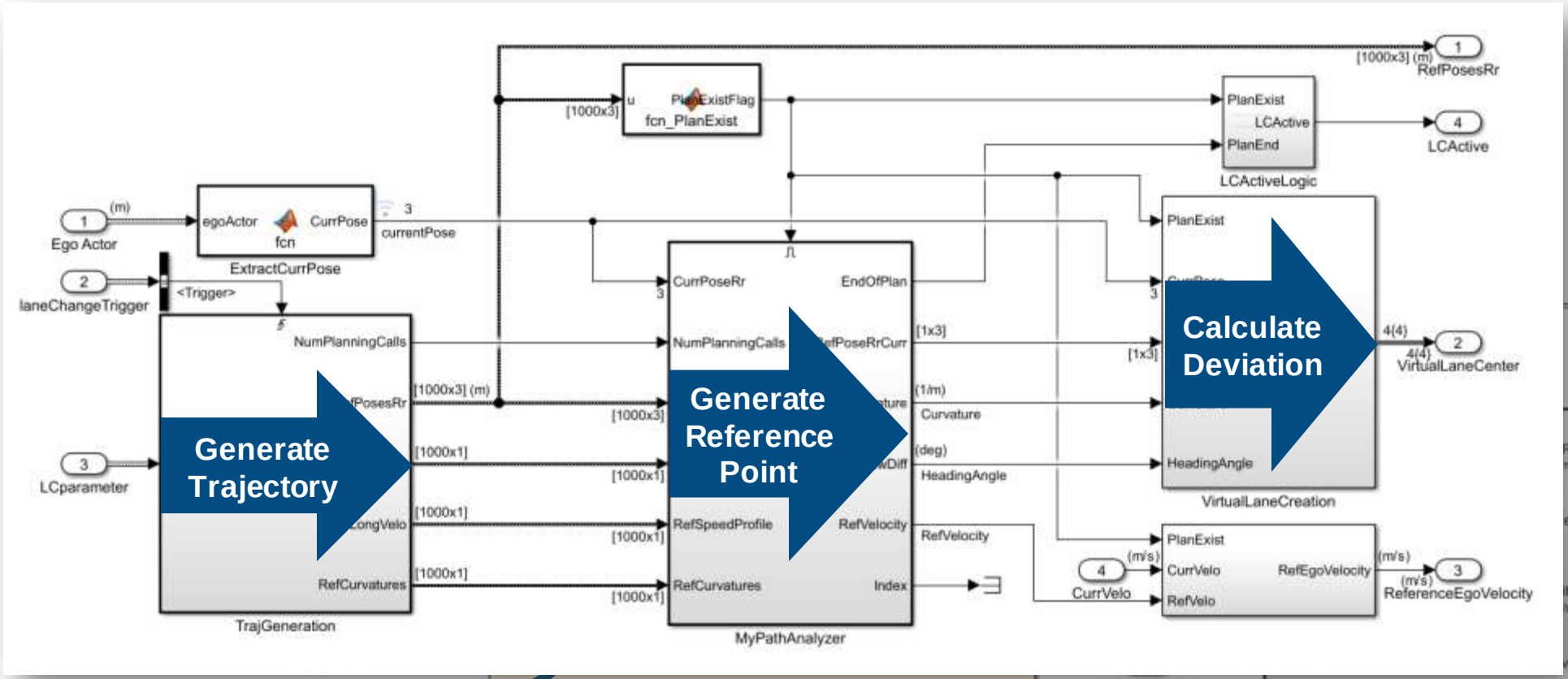


Safety zone calculations

Lane Change Decision Logic

Lane Change Planning

Design lane change planning



lanes

Safety zone calculations

Lane Change Decision Logic

Lane Change Planning

Reference trajectory for lane change

Generate trajectory

- Quintic polynomial

$$s(t) = a_5t^5 + a_4t^4 + a_3t^3 + a_2t^2 + a_1t + a_0$$
$$\dot{s}(t) = 5a_5t^4 + 4a_4t^3 + 3a_3t^2 + 2a_2t + a_1$$
$$\ddot{s}(t) = 20a_5t^3 + 12a_4t^2 + 6a_3t + 2a_2$$

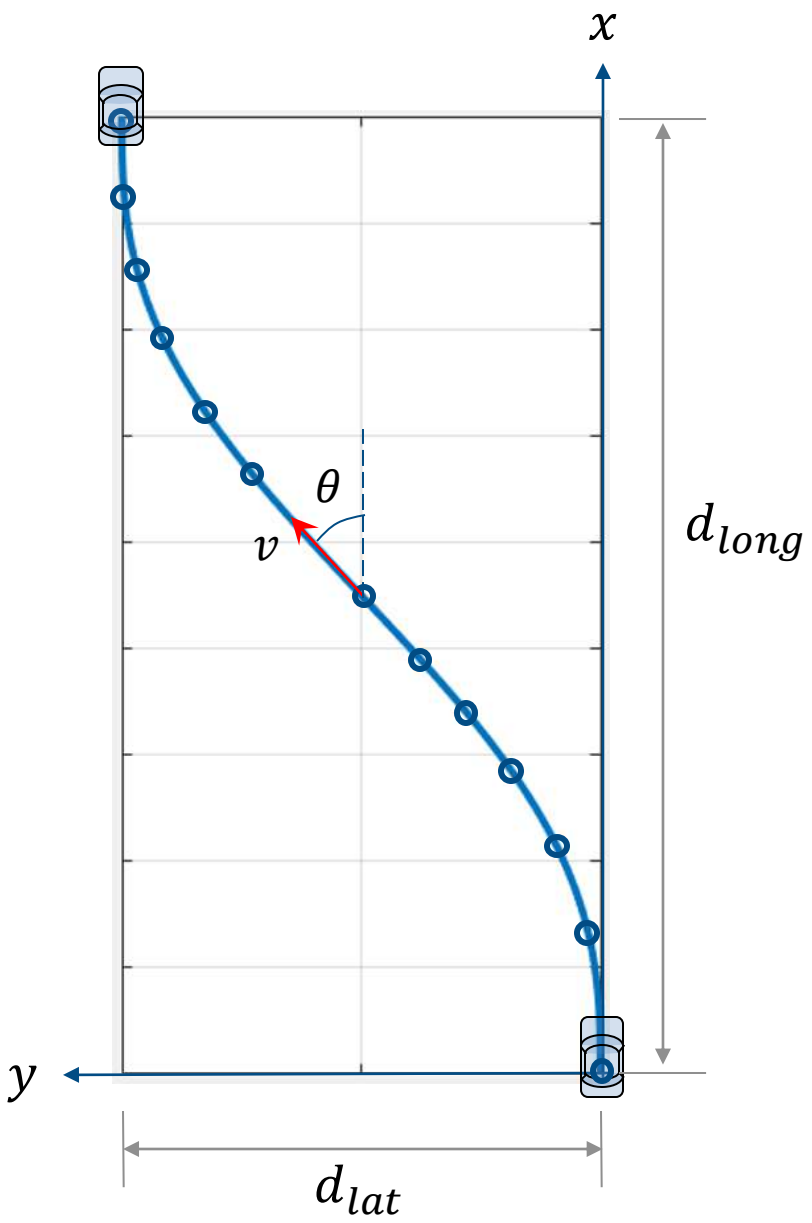
$$where\ s = longitudinal\ or\ lateral\ distance$$

- Start boundary conditions

$$a_0 = s_{start}$$
$$a_1 = \dot{s}_{start}$$
$$2a_2 = \ddot{s}_{start}$$

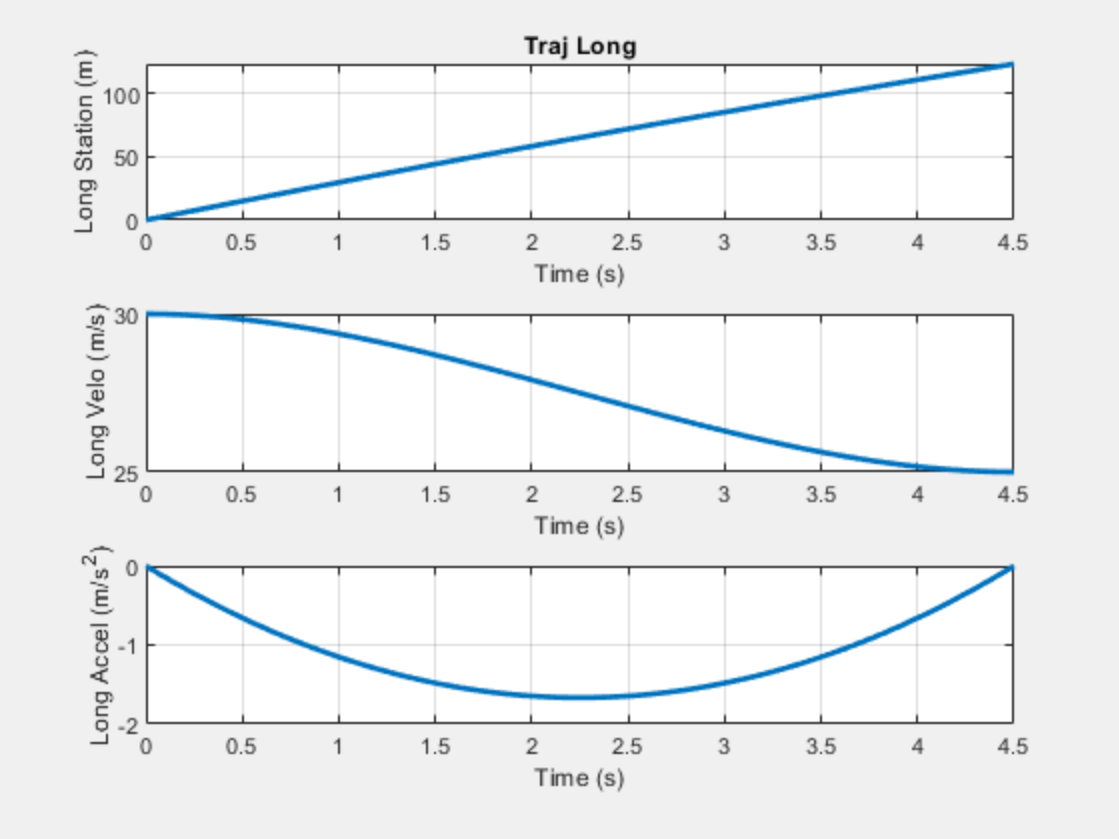
- End boundary conditions

$$a_5t_f^5 + a_4t_f^4 + a_3t_f^3 + a_2t_f^2 + a_1t_f + a_0 = s_{end}$$
$$5a_5t_f^4 + 4a_4t_f^3 + 3a_3t_f^2 + 2a_2t_f + a_1 = \dot{s}_{end}$$
$$20a_5t_f^3 + 12a_4t_f^2 + 6a_3t_f + 2a_2 = \ddot{s}_{end}$$

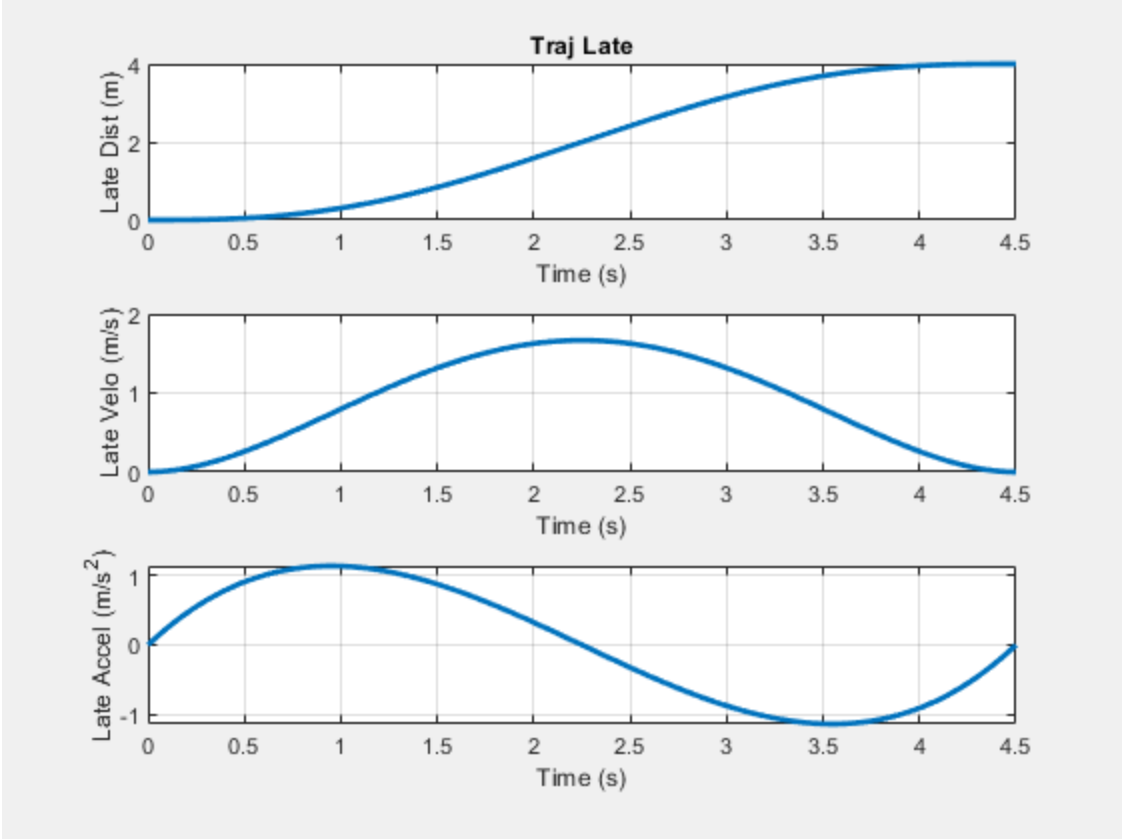


Example of trajectory generation for lane change

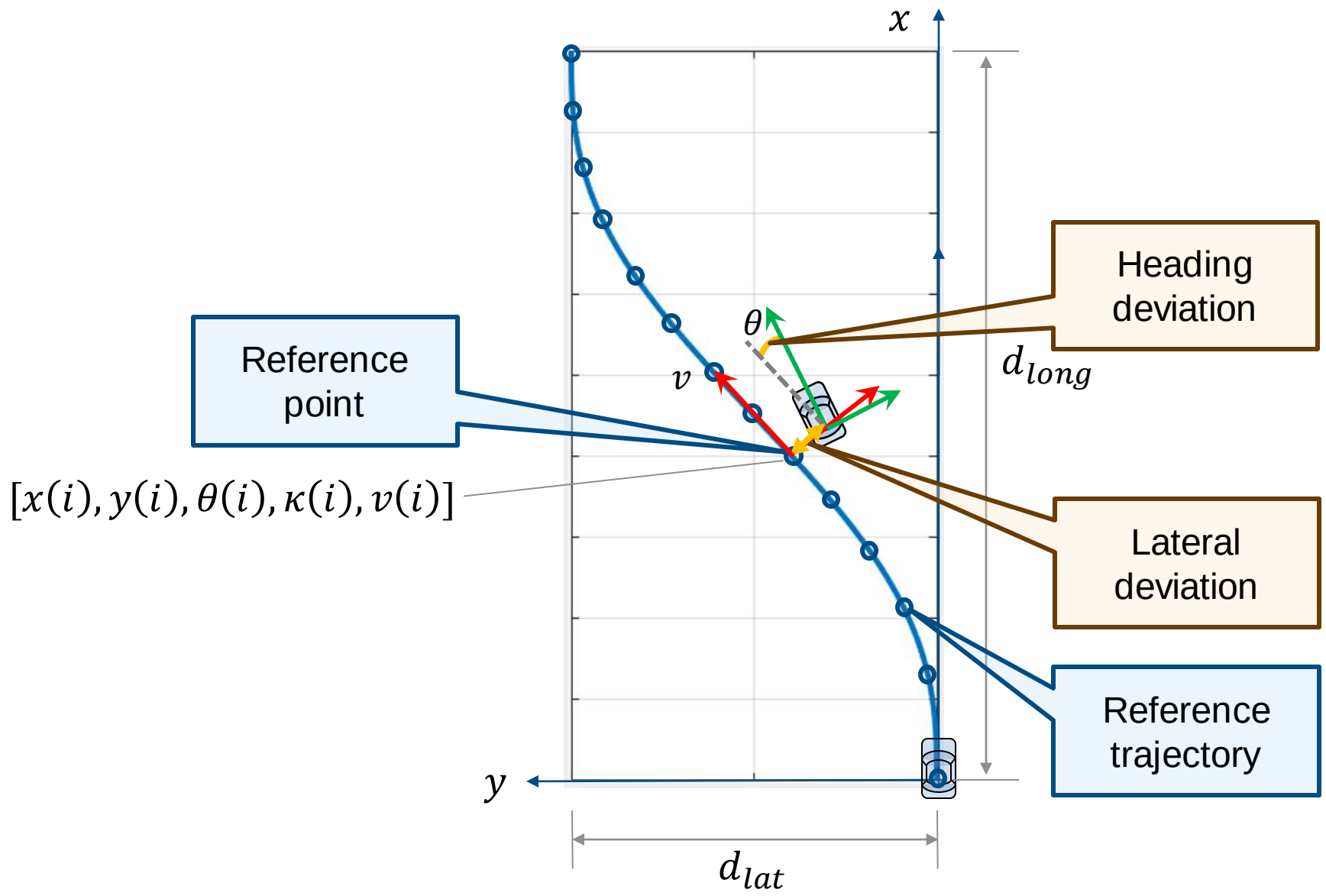
Longitudinal trajectory



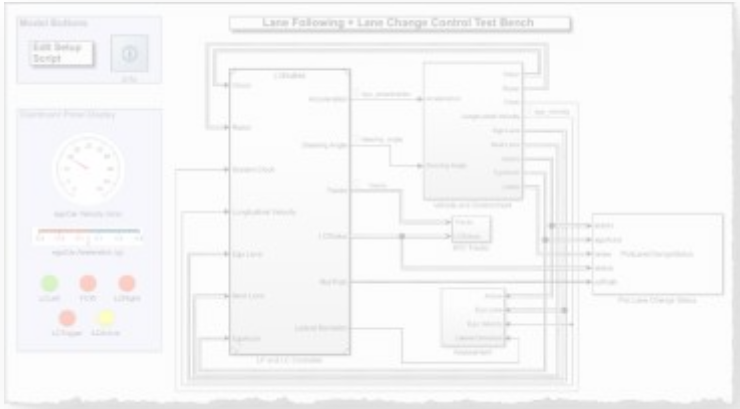
Lateral trajectory



Calculate deviations from reference point

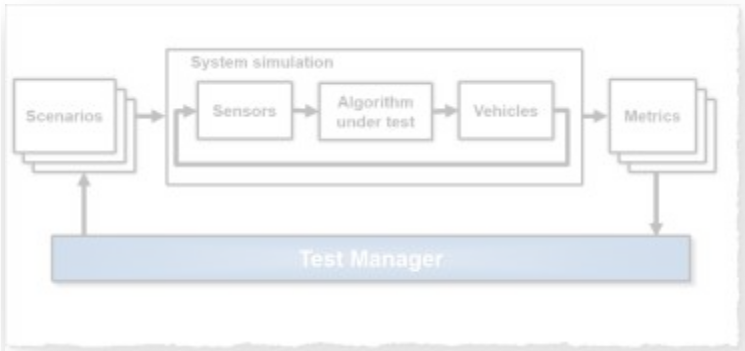


Case Study for Lane Following plus Lane Change



Design lane following + lane change controller

- Review baseline LF example
- Design sensor configuration
- Design additional MIO detectors
- Design safety zone calculation
- Design lane change logic
- Design trajectory planner



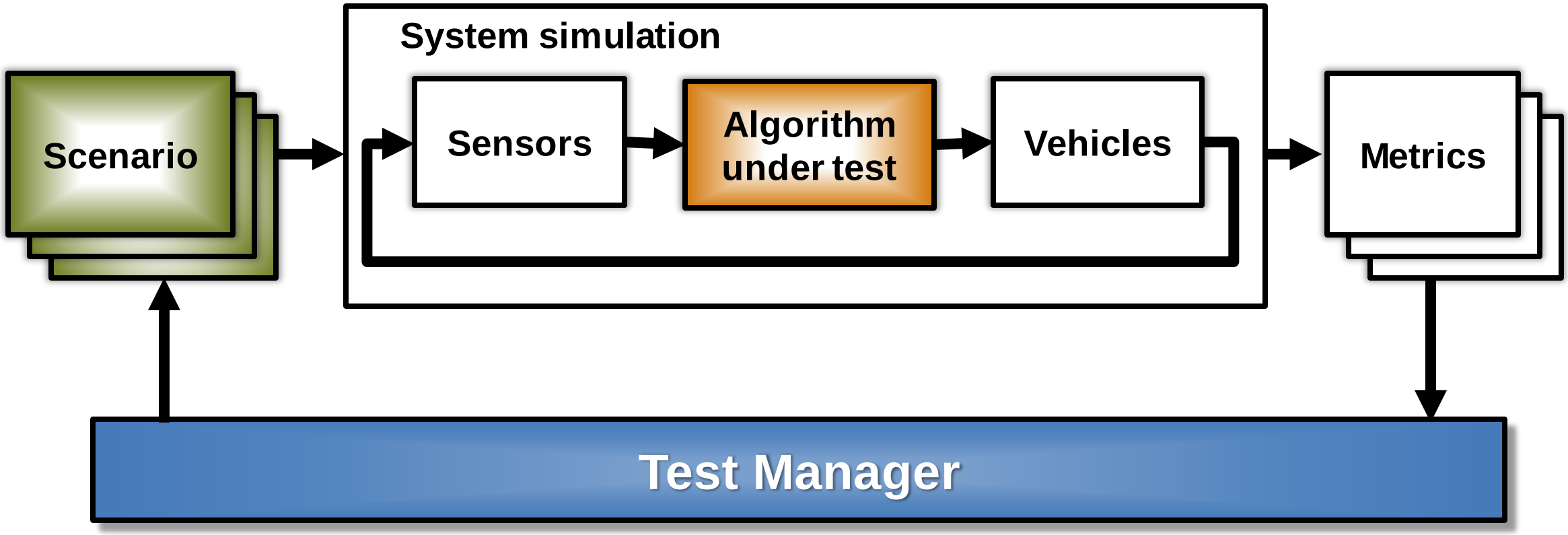
Automate regression testing

- Define assessment metrics
- Add predefined scenarios
- Run Simulink test

Test robustness with traffic agents

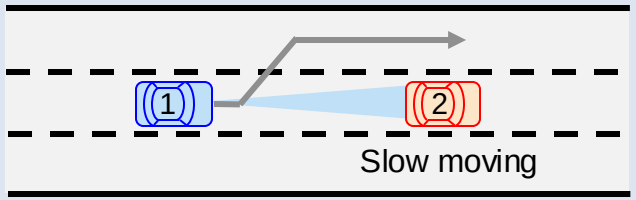
- Specify driver logic for traffic agents
- Randomize scenarios using traffic agents
- Identify and assess unexpected behavior

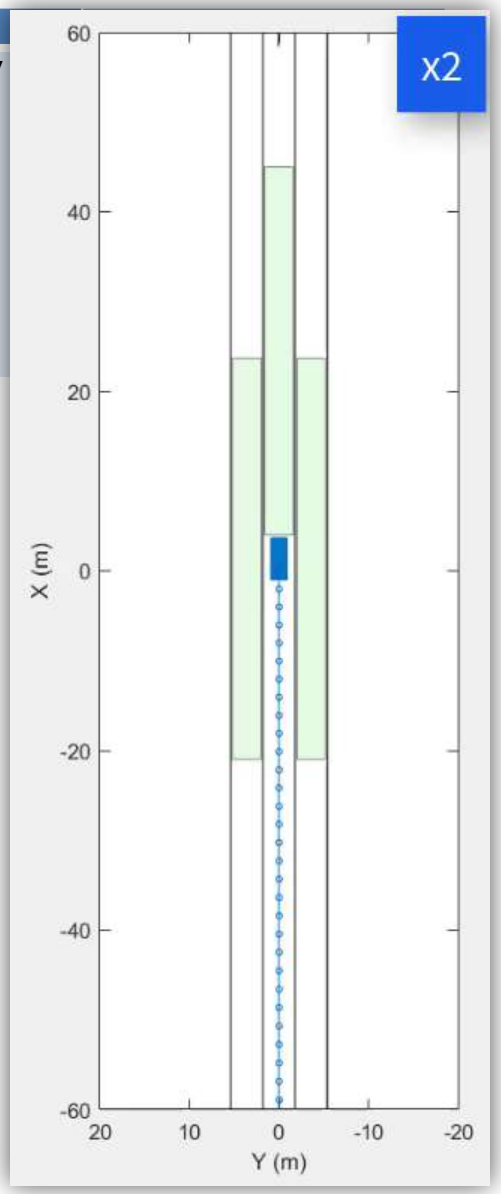
Manage testing against scenarios



Create test scenarios

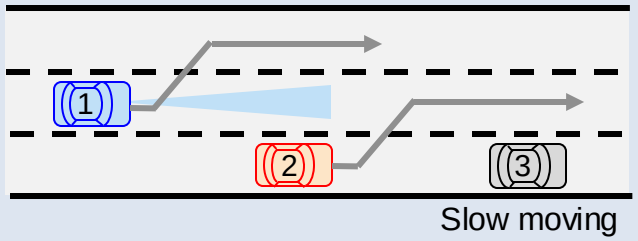
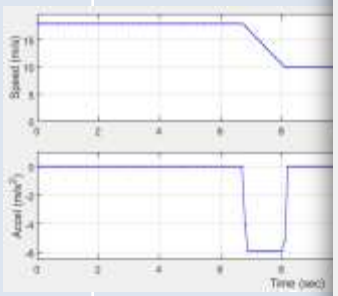
HW : Headway
HWT : Headway time
v_set : set velocity for ego car

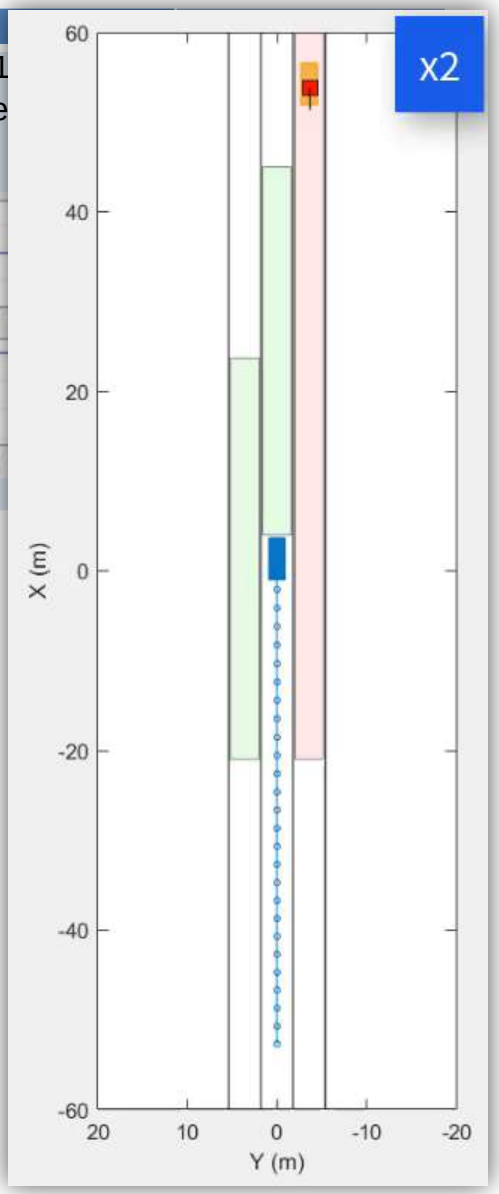
No	Test Name	Test Description	Host car	Lead car
1	01_SlowMoving	Passing for slow moving lead car 	initial velocity = 20m/s HWT = 6.5sec (HW = 130m) v_set = 20m/s	constant velocity 10m/s



Create test scenarios

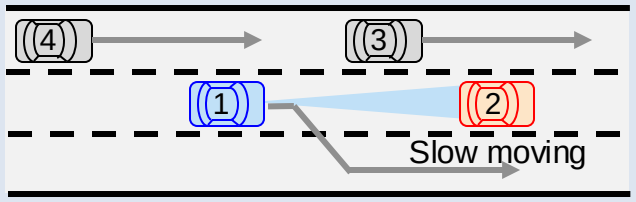
HW : Headway
HWT : Headway time
v_set : set velocity for ego car

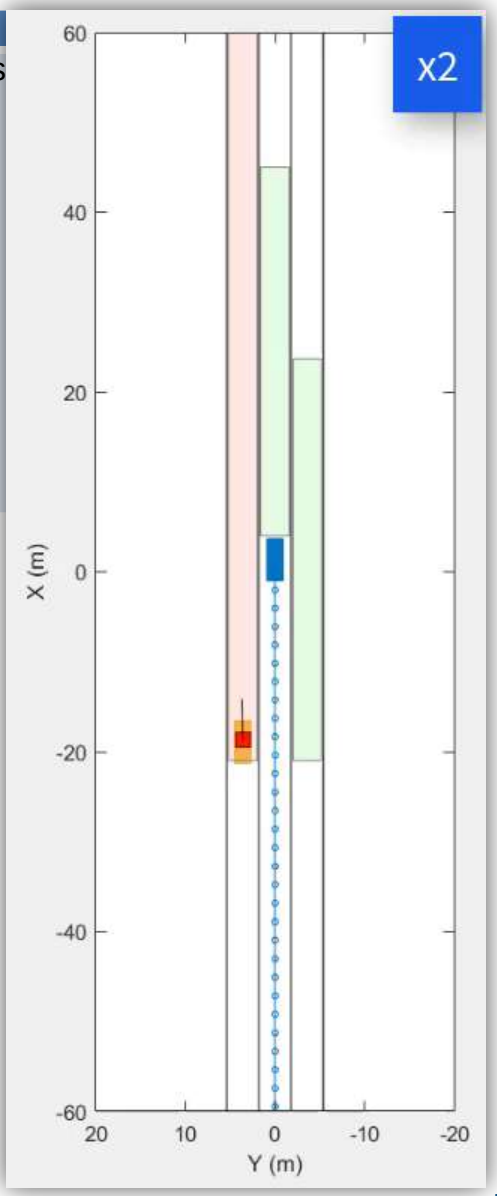
No	Test Name	Test Description	Host car	Lead car
4	04_CutInWithBrake	Passing for cut-in car with brake 	initial velocity = 20m/s v_set = 20m/s	initial velocity = 1 Cut-in with brake (18m/s→10m/s) 



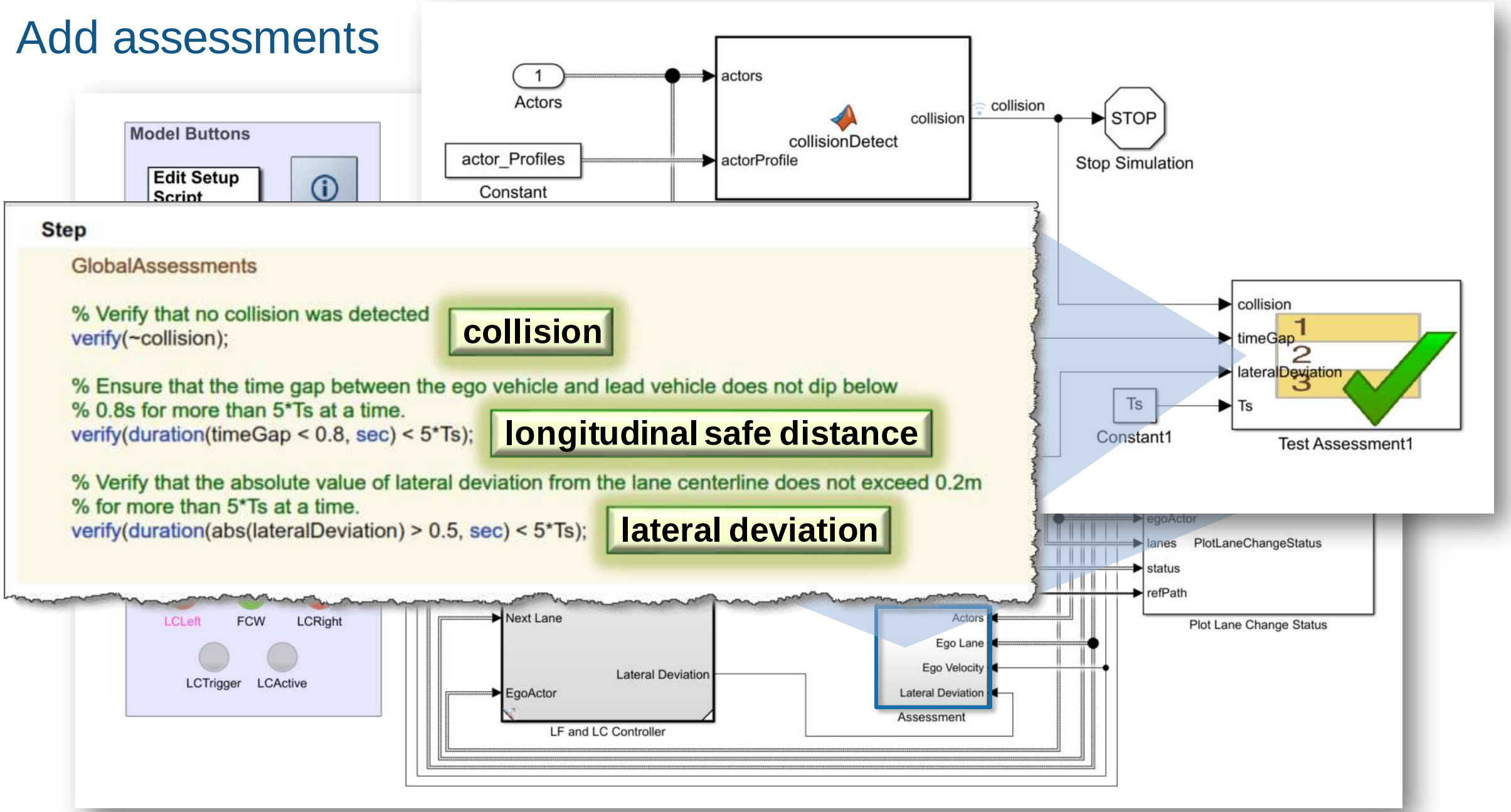
Create test scenarios

HW : Headway
HWT : Headway time
v_set : set velocity for ego car

No	Test Name	Test Description	Host car	Lead car
7	07_RightLaneChange	Passing for slow moving lead car to right lane 	initial velocity = 20m/s HWT = 6.5sec (HW = 130m) v_set = 20m/s	constant velocity = 10m/s



Add assessments



Review report generated by Test Manager test cases

Report Generated by Test Manager

Title: Lane Following + Lane Change Control Test
Author: Seo-Wook Park
Date: 04-Apr-2019 12:03:36

Test Environment

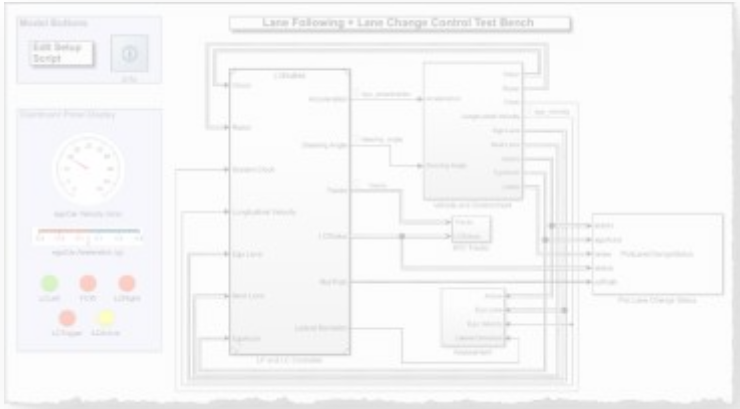
Platform: PCWIN64
MATLAB: (R2019a)

TestReport

Summary

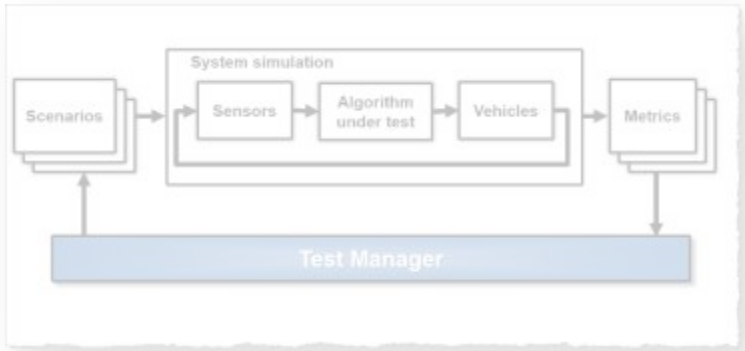
Name	Outcome	Duration (Seconds)
LCTestCases	7	2059
StraightPath	7	2059
01_SlowMoving	✓	304
02_SlowMovingWithPassingCar	✓	224
03_DisabledCar	✓	330
04_CutInWithBrake	✓	235
05_SingleLaneChange	✓	314
06_DoubleLaneChange	✓	420
07_RightLaneChange	✓	228

Case Study for Lane Following plus Lane Change



Design lane following + lane change controller

- Review baseline LF example
- Design sensor configuration
- Design additional MIO detectors
- Design safety zone calculation
- Design lane change logic
- Design trajectory planner



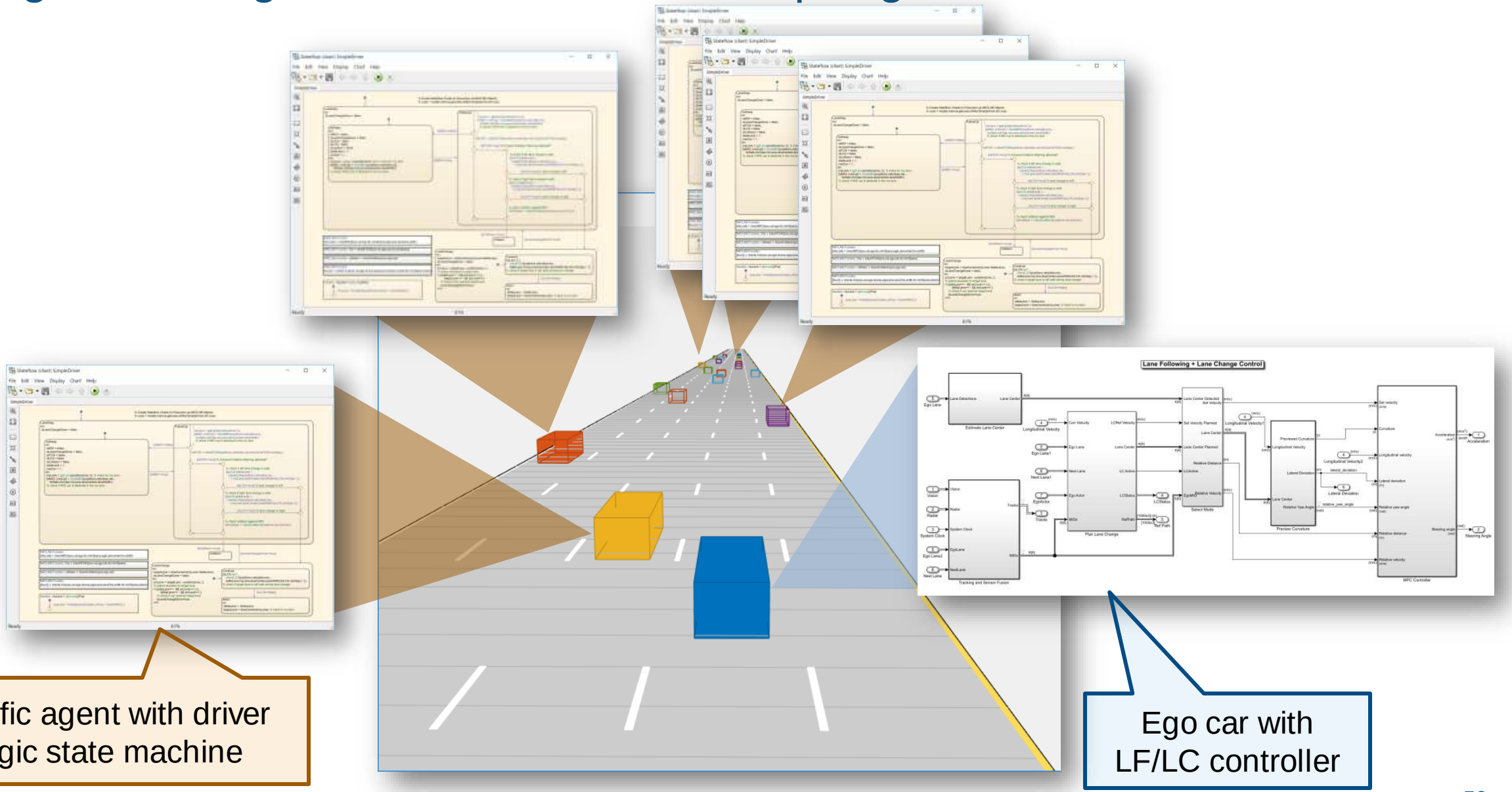
Automate regression testing

- Define assessment metrics
- Add predefined scenarios
- Run Simulink test

Test robustness with traffic agents

- Specify driver logic for traffic agents
- Randomize scenarios using traffic agents
- Identify and assess unexpected behavior

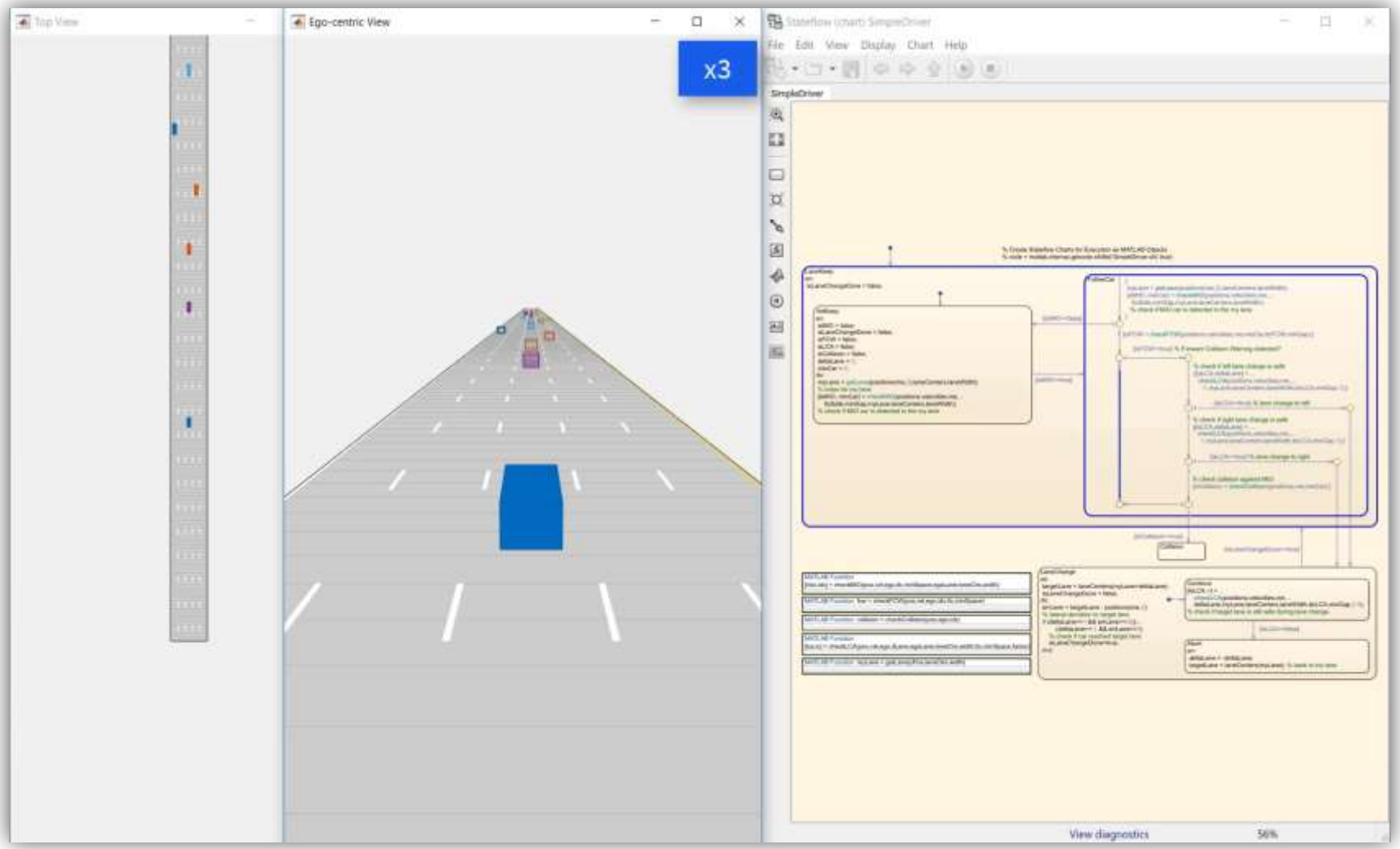
Assign traffic agents to all vehicles except ego car



Simulate interaction between traffic agents

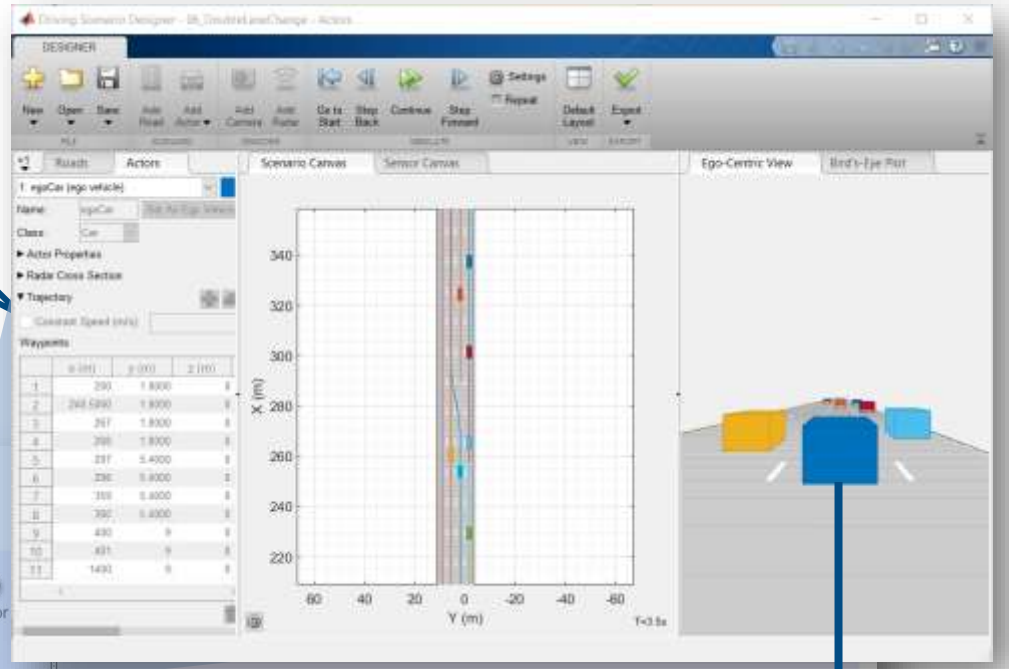
Proof of Concept

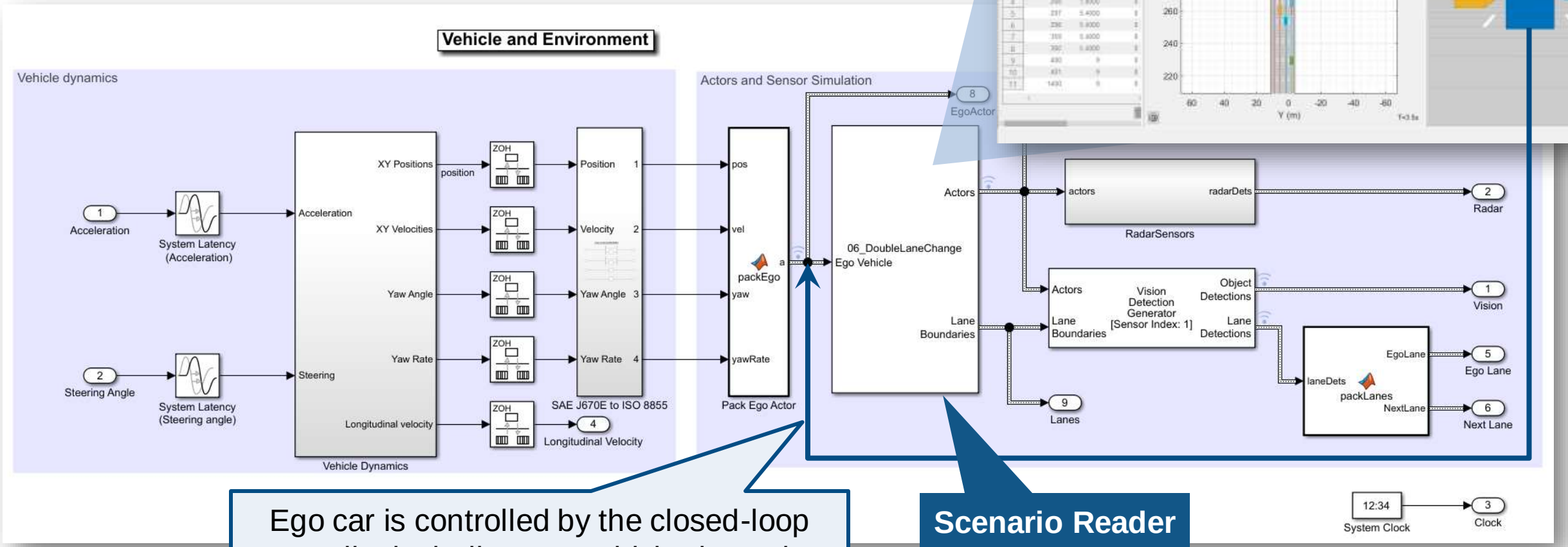
- Driver decision logic implemented by Stateflow™
- Rules are based on ground truths
- Integrate into cuboid driving scenario
- Visualize and debug



Scenario Reader

Driving scenario is pre-defined by DSD



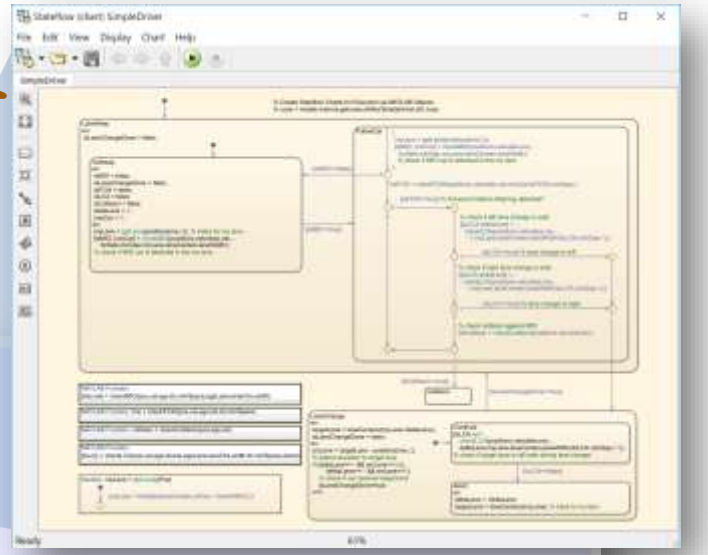


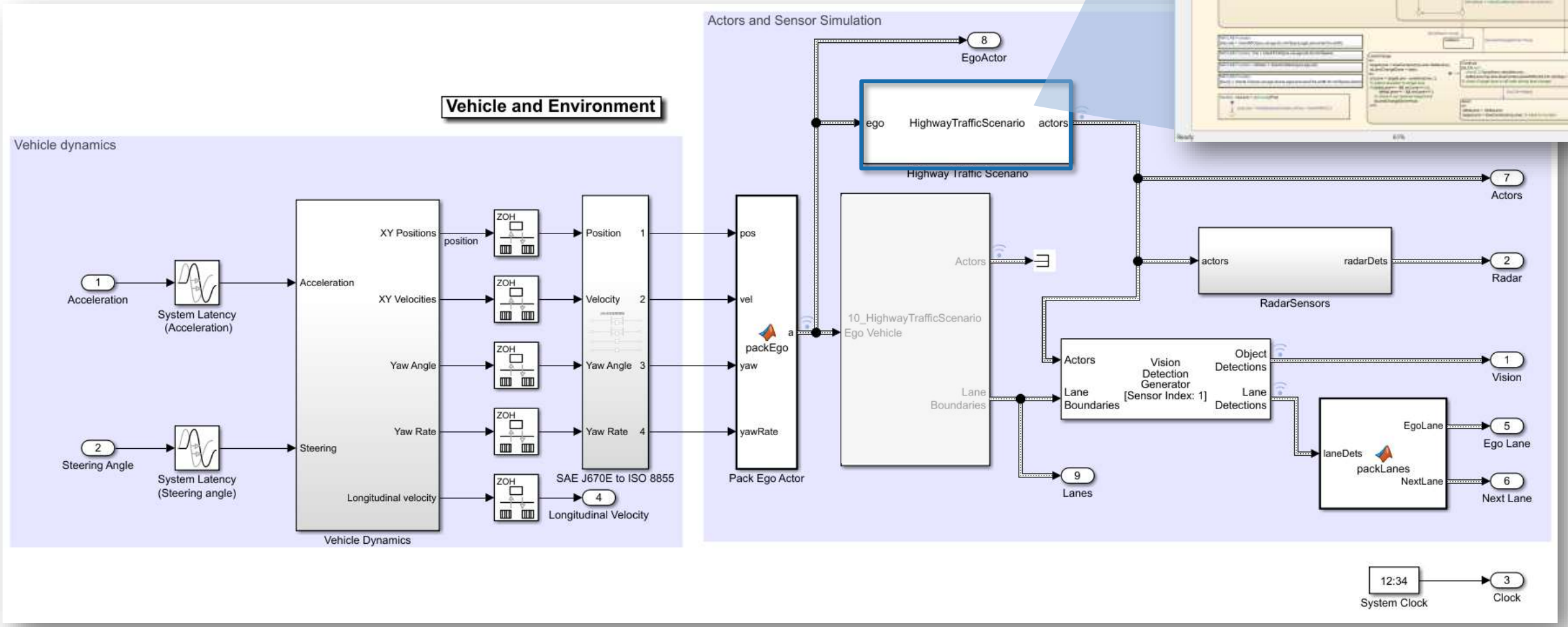
Ego car is controlled by the closed-loop controller including ego vehicle dynamics

Scenario Reader Block

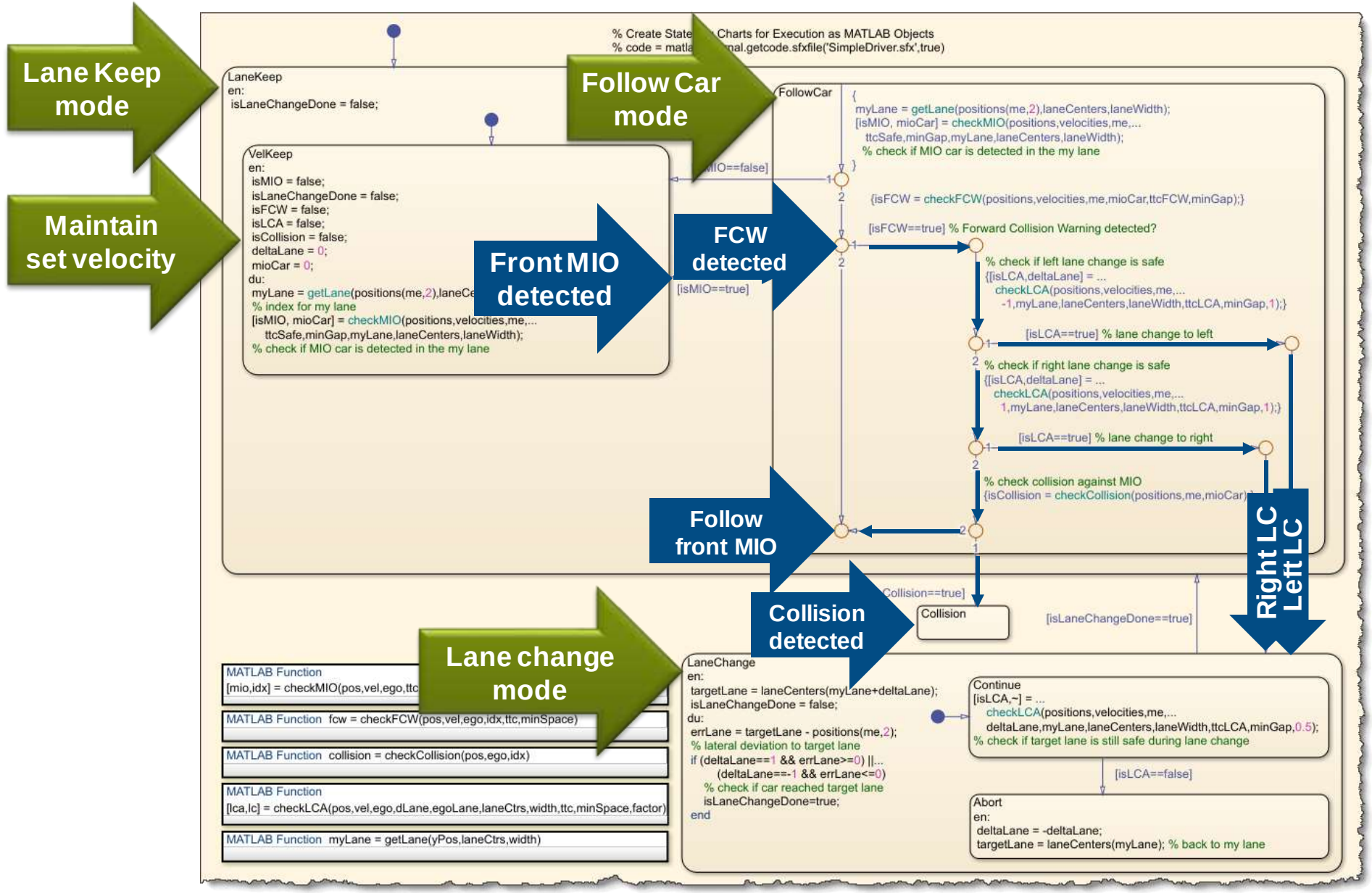
Traffic agent

State machine implementing driver logic

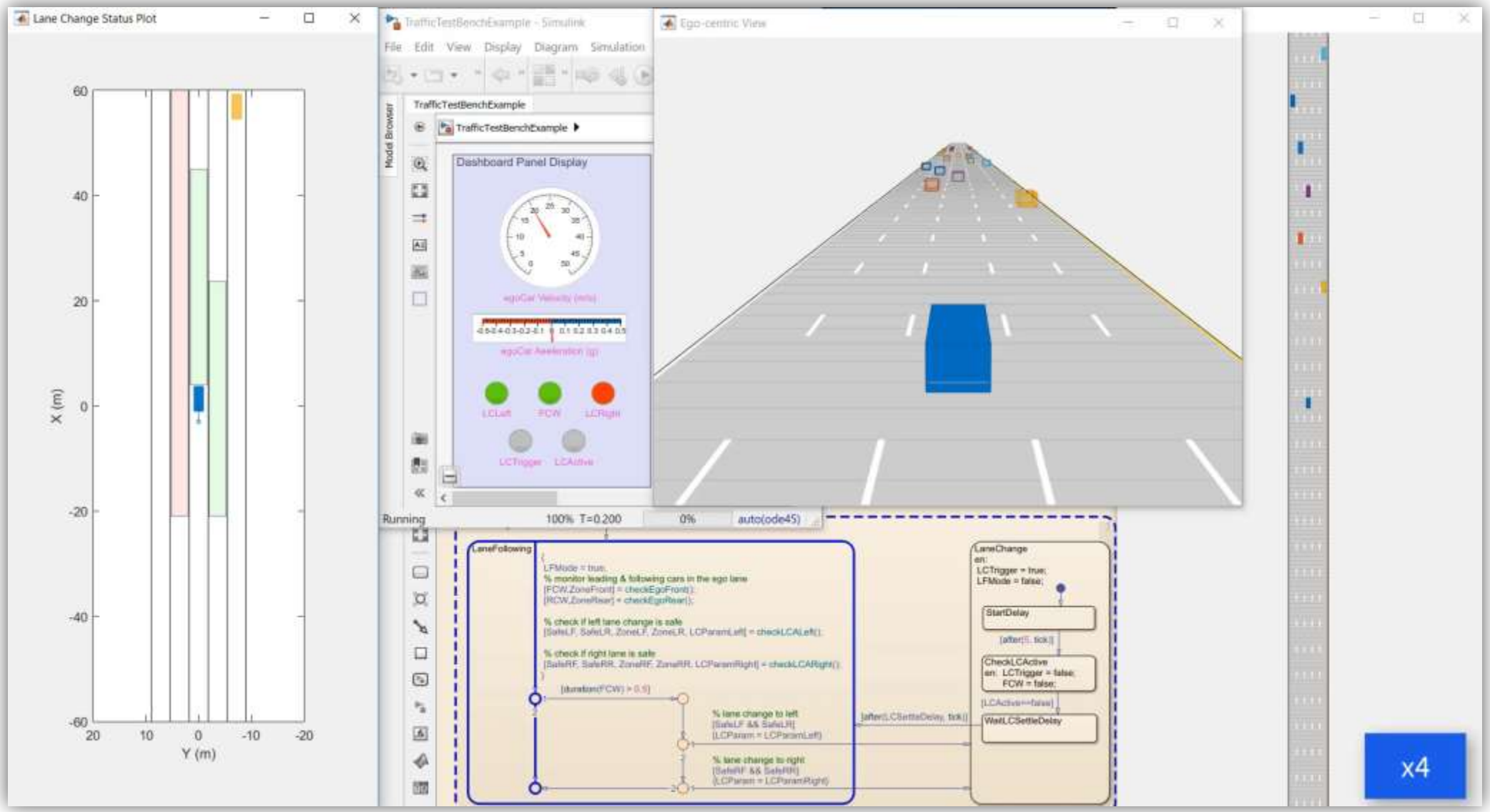




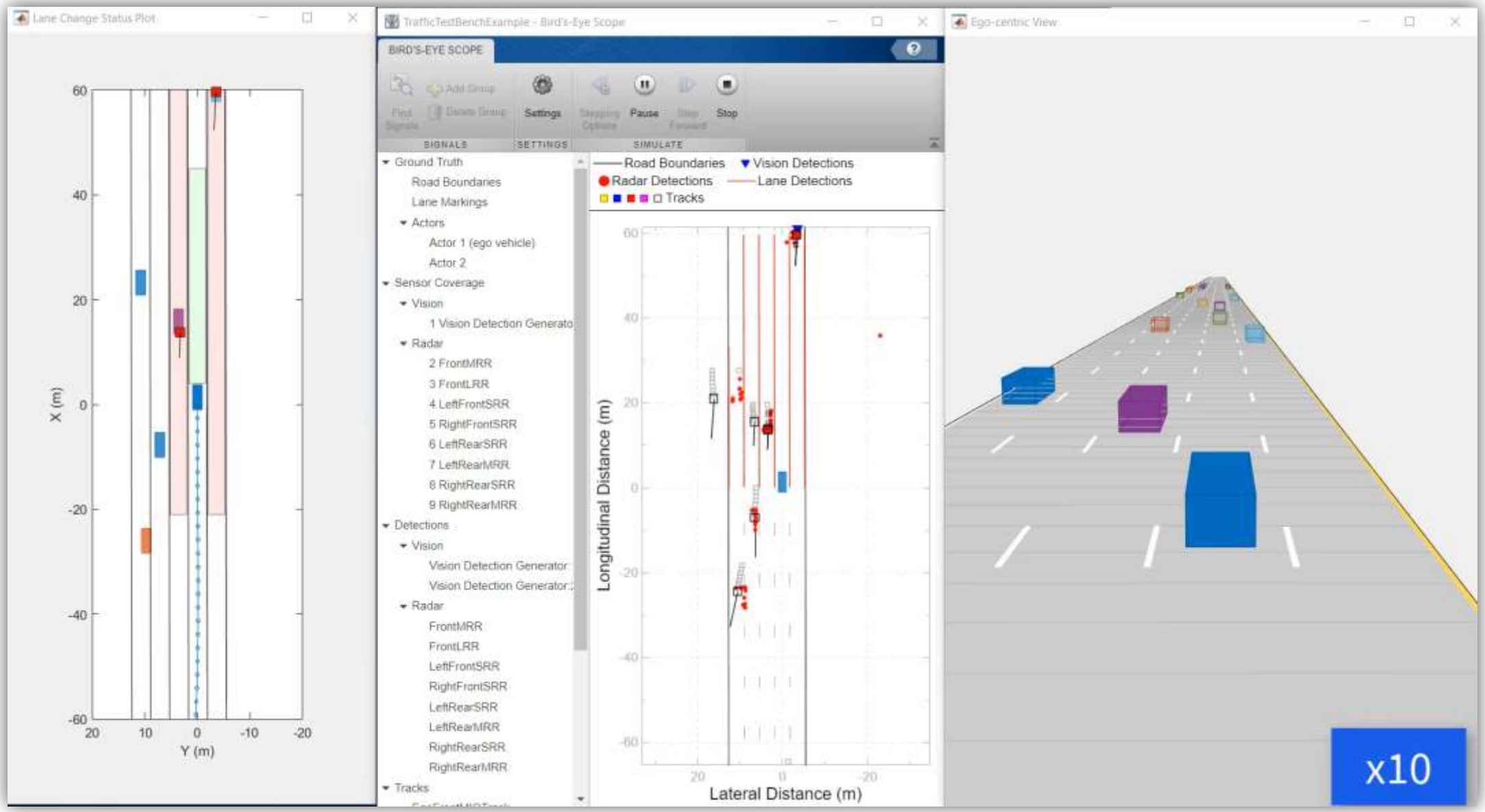
Implement driver logic for traffic agent using Stateflow™



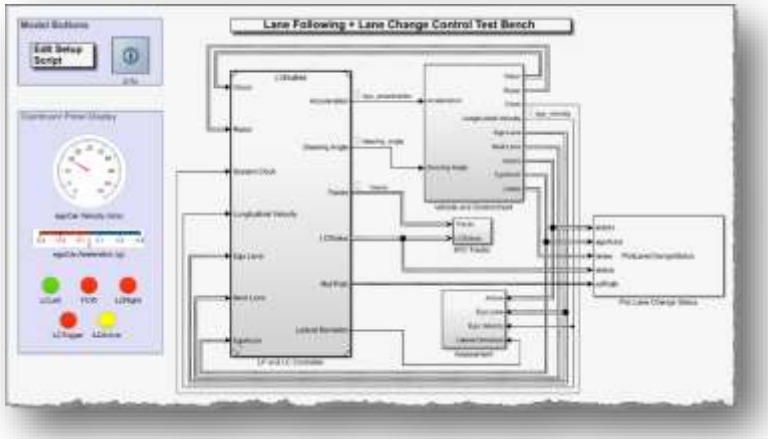
Simulate with traffic agents



Analyze results for near collision situation

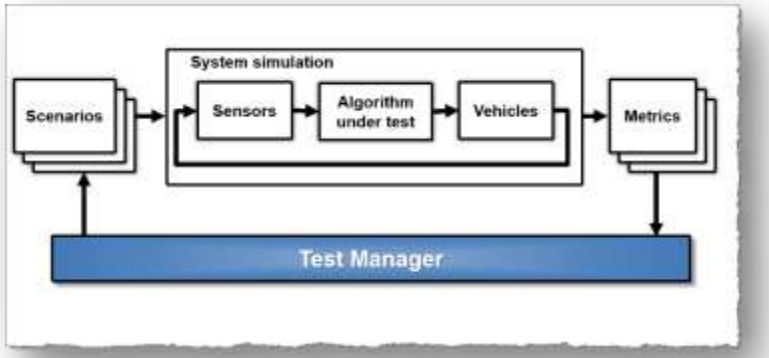


Recap: Case Study for Lane Following plus Lane Change



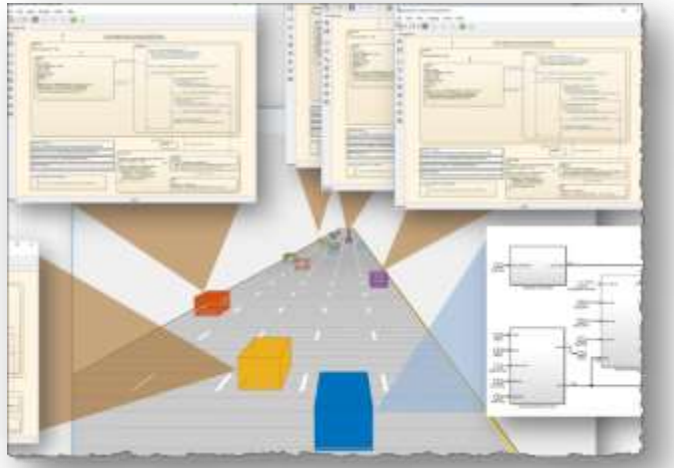
Design lane following + lane change controller

- Review baseline LF example
- Design sensor configuration
- Design additional MIO detectors
- Design safety zone calculation
- Design lane change logic
- Design trajectory planner



Automate regression testing

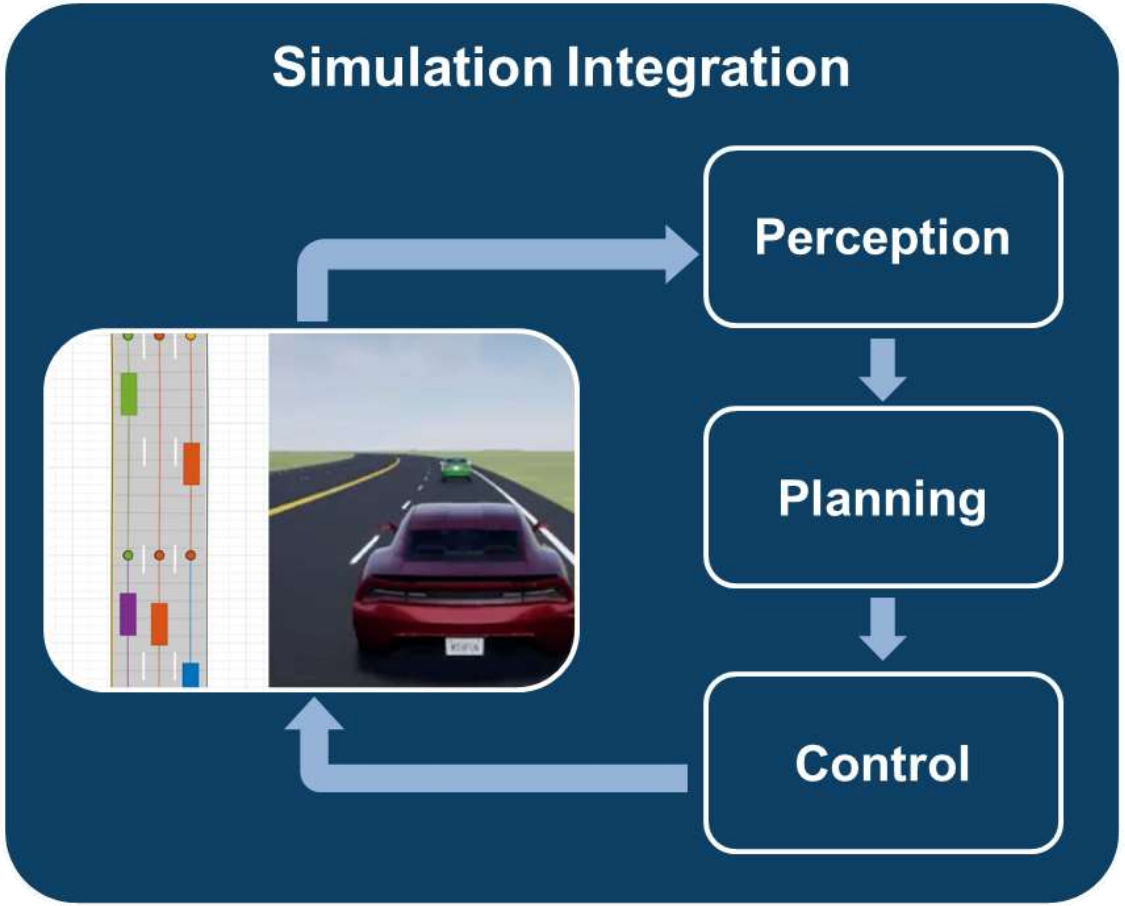
- Define assessment metrics
- Add predefined scenarios
- Run Simulink test



Test robustness with traffic agents

- Specify driver logic for traffic agents
- Randomize scenarios using traffic agents
- Identify and assess unexpected behavior

Contact us to learn more



Would you like to discuss any of these topics in more detail?

Contact your local team or reach out to me at spark@mathworks.com