

MATLAB EXPO 2019

Develop and Test Vehicle Controllers for
ADAS/Automated Driving Applications through
System Simulation

Abhisek Roy



Highway Traffic Jam Assist

- It helps drivers to follow the preceding vehicle automatically with a predefined time interval in a dense traffic condition

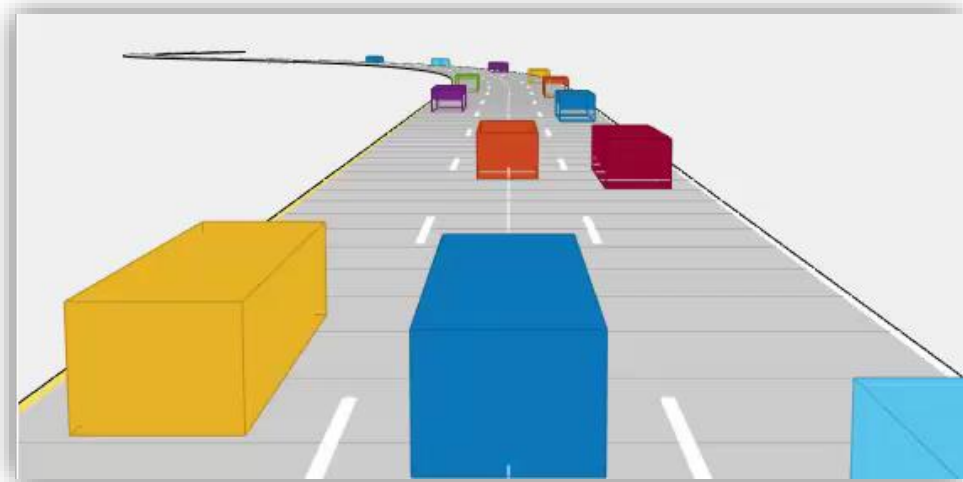


Longitudinal control with ACC with stop & go

... while controlling steering for keeping current lane.



Lateral control with lane following



- Partial/conditional automation at level 2/3
 - Speed limit < 60~65 km/h
 - Dense traffic condition in highway

Challenges

- Wide variety of scenarios and difficult to gather real data
- Complex interplay between multiple sensors
- Incorporate models of right fidelity for various system components
- Costly and hazardous in-vehicle testing

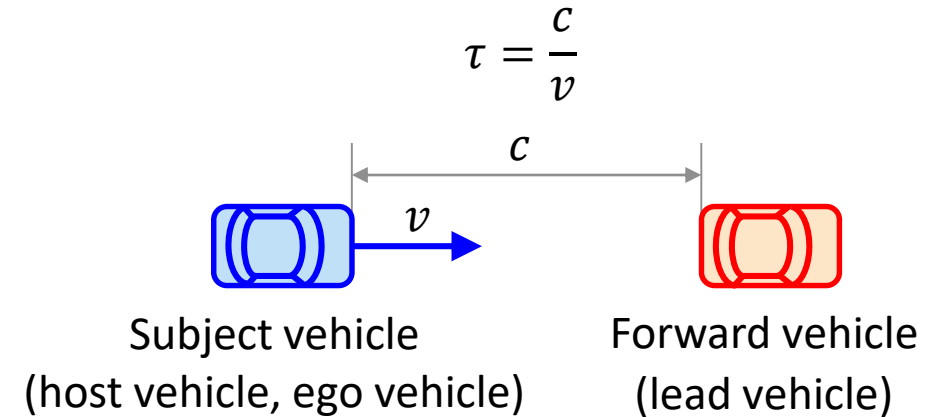
Agenda

- Design model-predictive control-based vehicle controllers
- Run close-loop simulation with synthetic scenarios and test sensor fusion and control algorithms at a model level
- Improve simulation fidelity by incorporating detailed vehicle models and integrating with Unreal gaming engine

Performance Requirements: Longitudinal Control

- **Ego velocity control :** $v \leq v_{set}$
where, v : ego velocity, v_{set} : set velocity

- **Time gap control:** $\tau \geq \tau_{min}$
where, $\tau = \frac{c}{v}$: time gap = 1.5 .. 2.2 sec
 τ_{min} : min time gap = 0.8 sec



- **Operation limits**
 - Minimum operational speed, $v_{min} = 5\text{m/s}$
 - Average automatic deceleration $\leq 3.5 \text{ m/s}^2$ (average over 2s)
 - Average automatic acceleration $\leq 2.0 \text{ m/s}^2$

Performance Requirements: Lateral Control

- Vehicle should follow the lane center with allowable lateral deviation.

$$|(d_{left} + d_{right})/2| \leq e_{max}$$

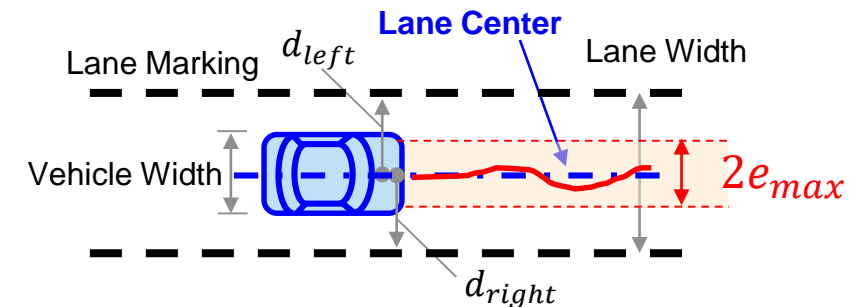
where,

d_{left} : lateral offset of left lane w.r.t. ego car

d_{right} : lateral offset of right lane w.r.t. ego car

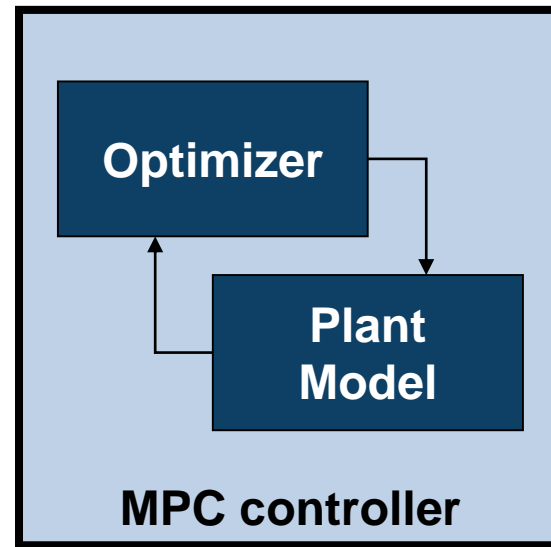
e_{max} : allowable lateral deviation

For example, $e_{max} = (LaneWidth - VehicleWidth)/2 = (3.6-1.8)/2 = 0.9$ m



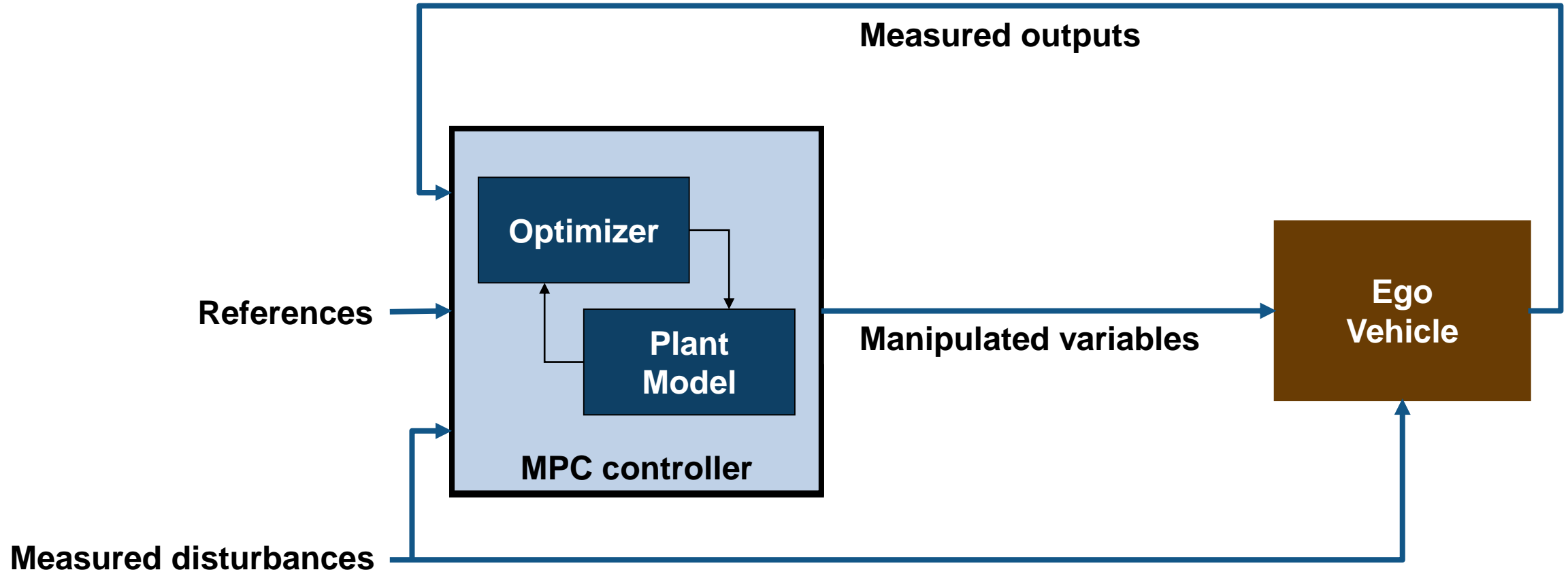
What is model predictive control (MPC)?

- **Multi-variable control** strategy leveraging an internal model to predict plant behavior in the near future
- **Optimizes** for the current timeslot while keeping future timeslots in account



- **Suitable** for our problem statement
 - Handles MIMO systems with coupling
 - Handles constraints
 - Has preview capabilities

How can MPC be applied to Highway Traffic Jam Assist?



How can MPC be applied to Highway Traffic Jam Assist?

minimize:

$$w_1 |V_{ego} - V_{set}|^2 + w_2 |E_{lateral}|^2$$

References

- Ego velocity set point (V_{set})
- Target lateral deviation (=0)

Measured disturbances

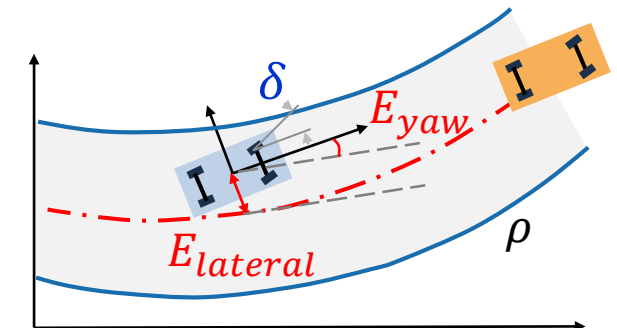
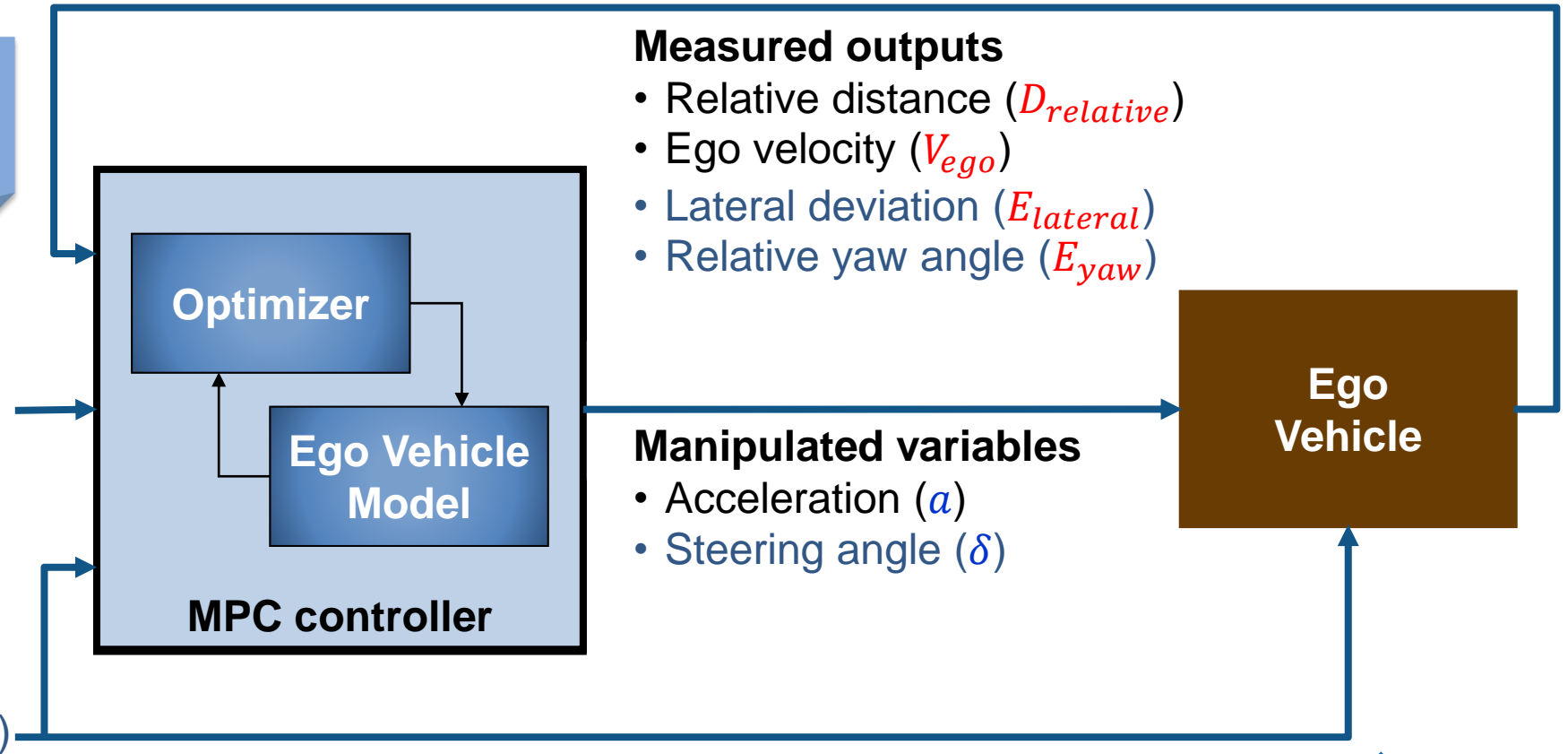
- MIO velocity (V_{mio})
- Previewed road curvature (ρ)

subject to:

$$D_{relative} \geq D_{safe}$$

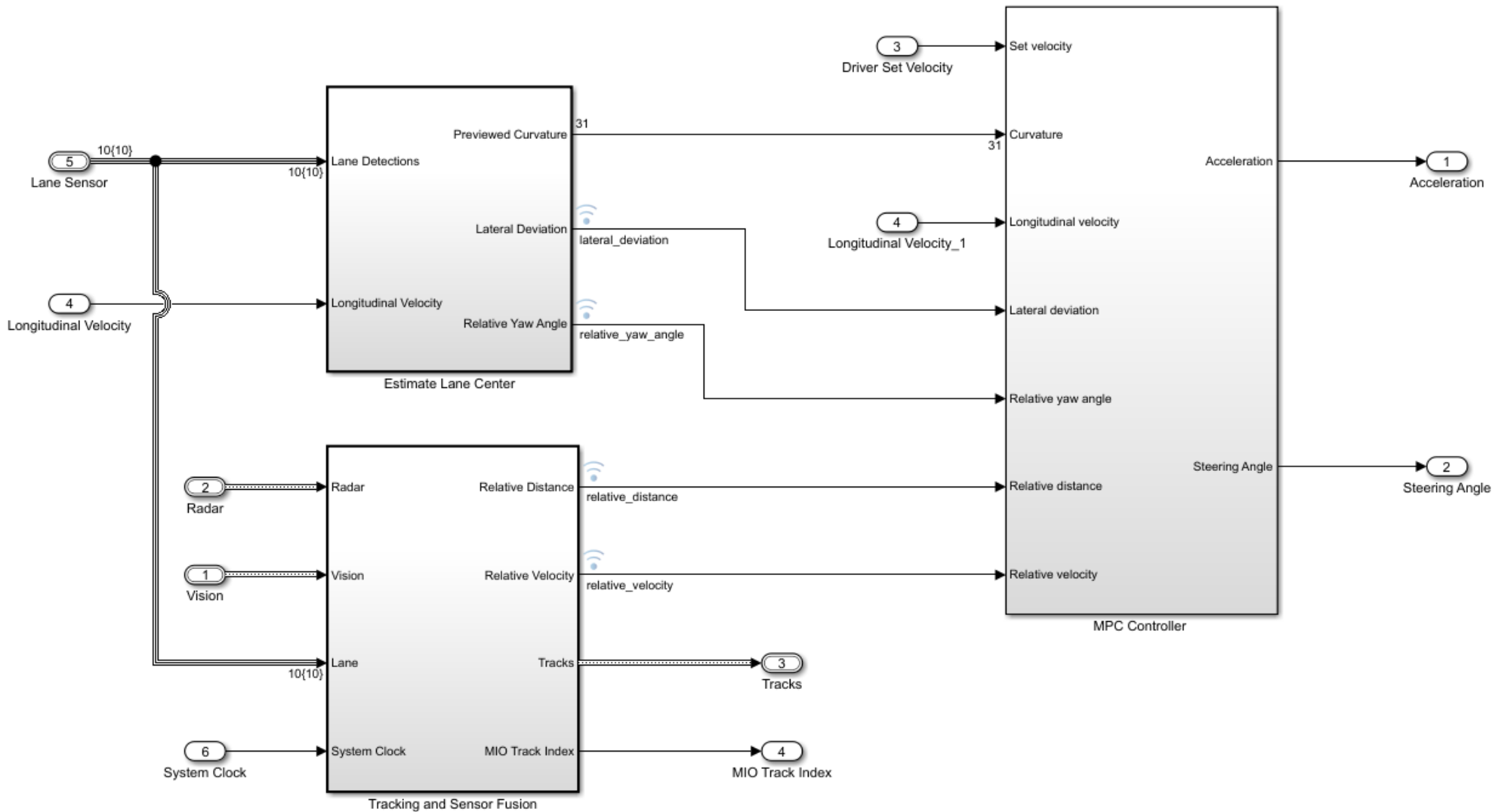
$$a_{min} \leq a \leq a_{max}$$

$$\delta_{min} \leq \delta \leq \delta_{max}$$



Control Algorithm

Lane Following with Spacing Control



Control Algorithm

Block Parameters: Path Following Control System

Path following control (PFC) system (mask) (link)

Keep the ego vehicle traveling along the center of a straight or curved road, track a set velocity and maintain a safe distance from a lead vehicle by adjusting the longitudinal acceleration and the front steering angle of the ego vehicle.

Parameters Controller **Block**

Optimization

Use suboptimal solution Maximum iteration number: 10

Data Type

double single

Optional Inports

Use external signal to enable or disable optimization

Use external control signal for bumpless transfer between PFC and other controllers

Customization

To customize your controller, generate an PFC subsystem from this block and modify it. The controller configuration data is exported as a structure in the MATLAB workspace.

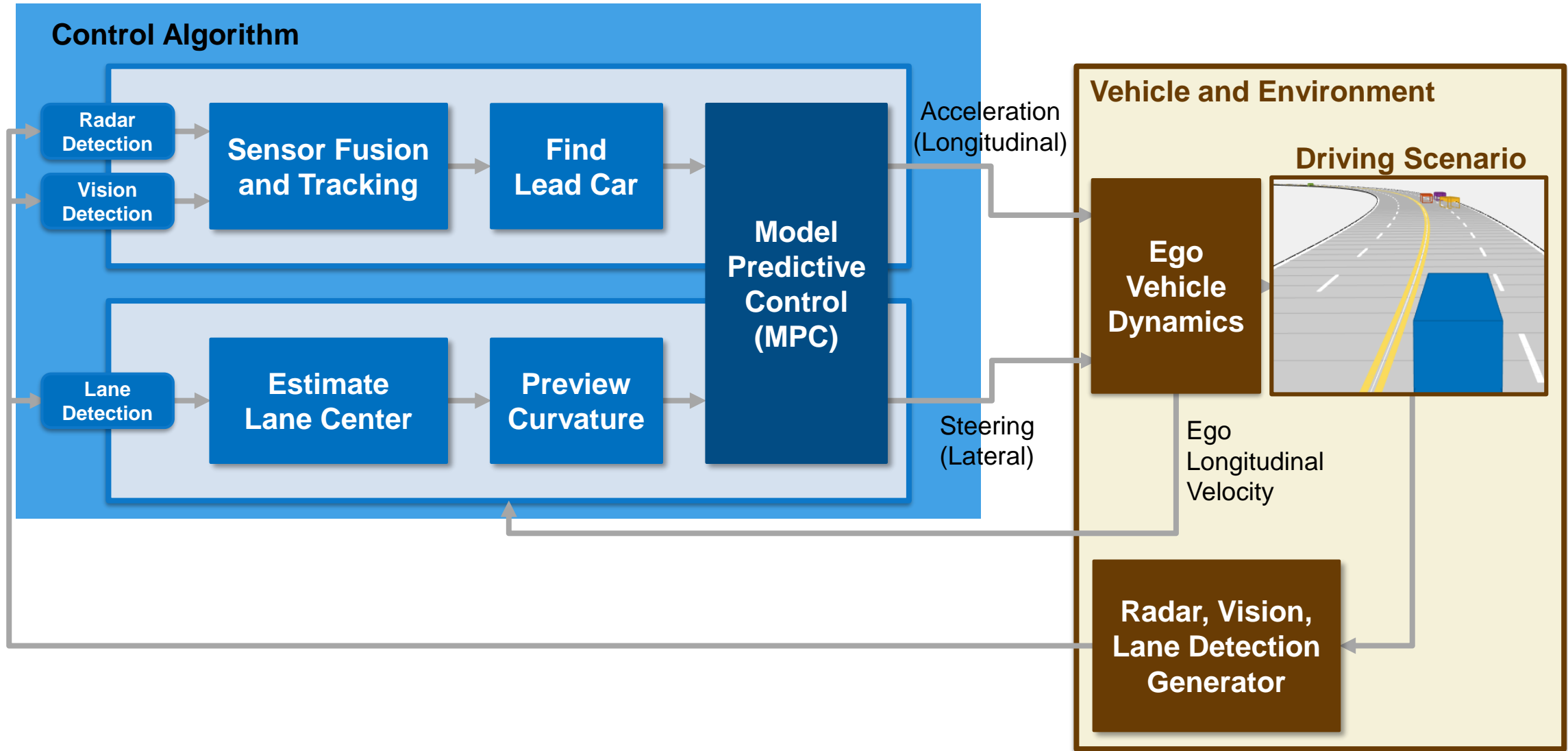
Create PFC subsystem

OK Cancel Help Apply

Agenda

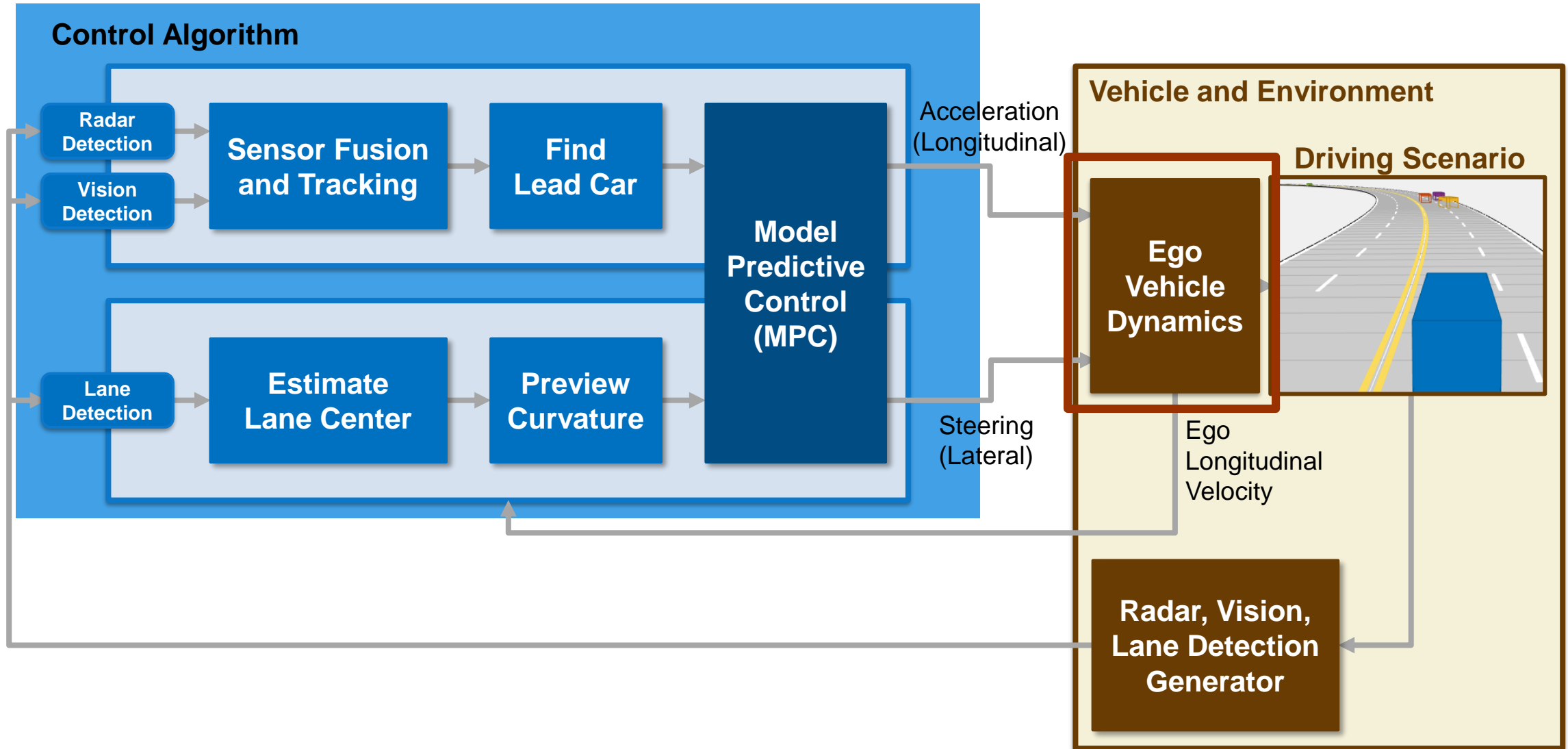
- Design model-predictive control-based vehicle controllers
- Run close-loop simulation with synthetic scenarios and test sensor fusion and control algorithms at a model level
- Improve simulation fidelity with gaming engine integration, vehicle dynamics modelling, and automated scenario creation from recorded data

Architecture for Traffic Jam Assist Controller



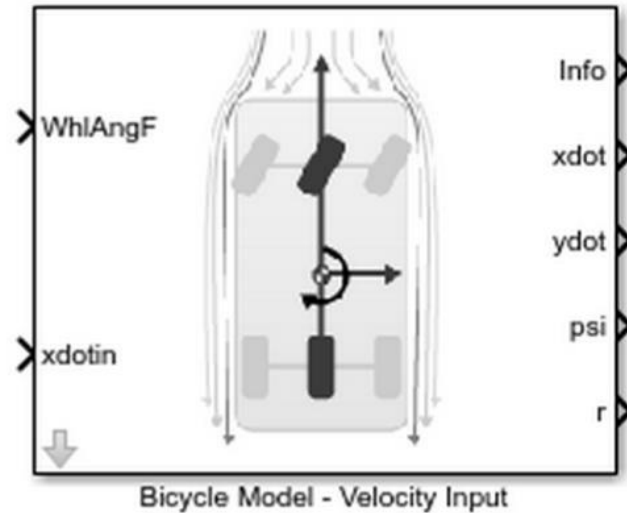
Develop and Test Vehicle Controller

Traffic Jam Assist

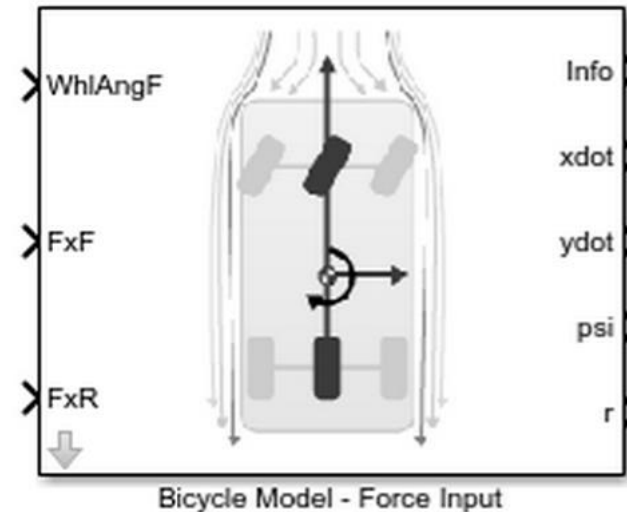


Incorporate Ego Vehicle Dynamics

Bicycle Model - Velocity Input

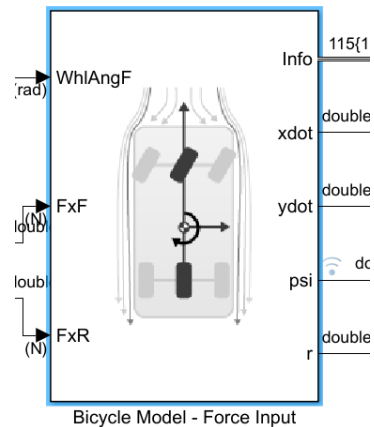


Bicycle Model - Force Input

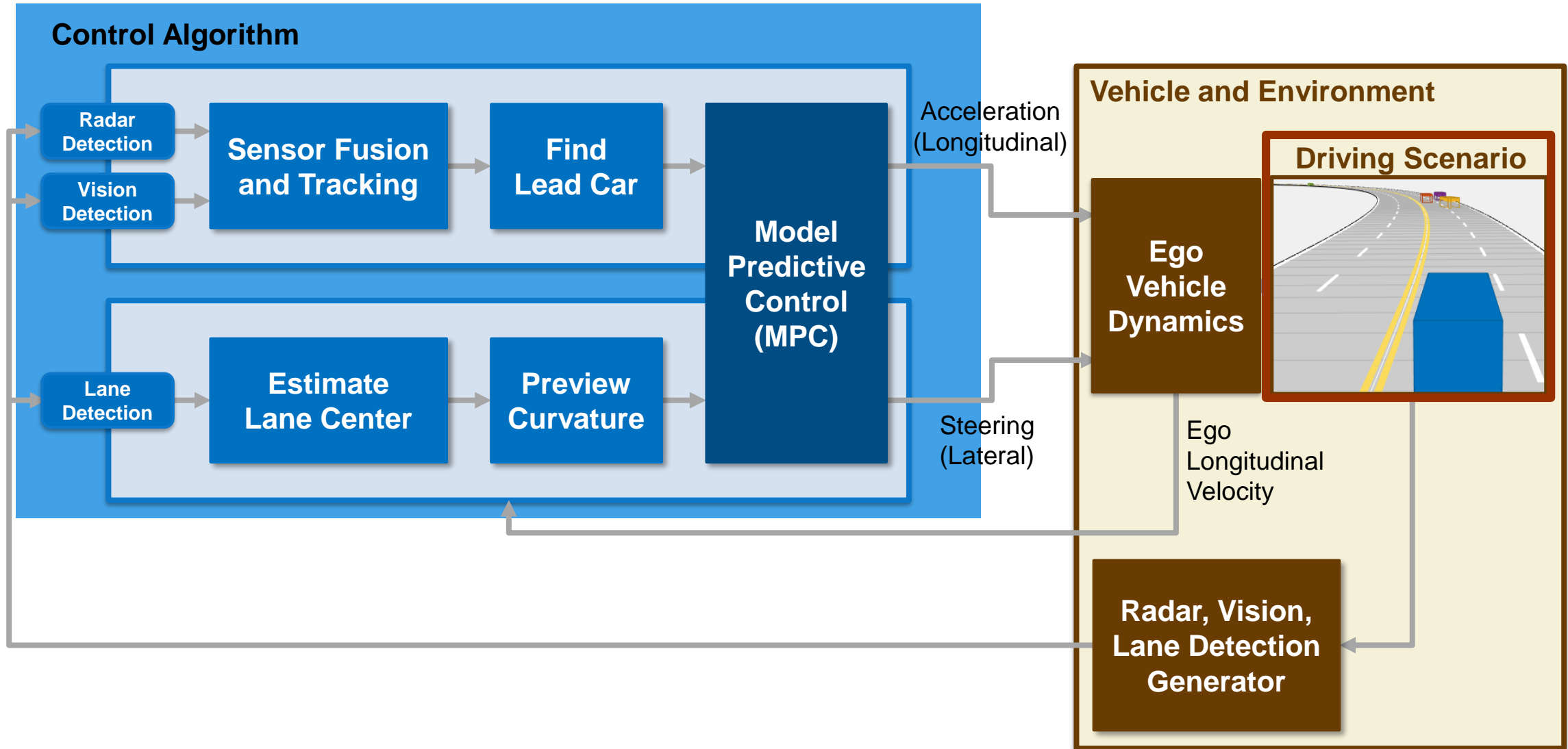


- Implement a single track 3DOF rigid vehicle body to calculate longitudinal, lateral, and yaw motion
- Block calculates only lateral forces using the tire slip angles and linear cornering stiffness.

Vehicle Dynamics



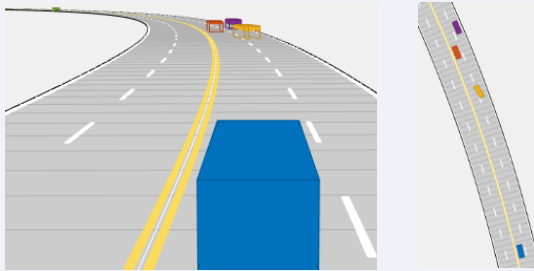
Develop and Test Vehicle Controller Traffic Jam Assist



Create Test Scenario using Driving Scenario Designer

Test Description

Lead car cut in and out in curved highway
(curvature of road = 1/500 m)



Host car

initial velocity = 20.6m/s

HWT(Headway Time) to lead car = 4sec

HW(Headway) to lead car = ~80m

v_set (set velocity for ego car) = 21.5m/s

Lead Car

Initially, fast moving car (orange) at 19.4m/s

Passing car (yellow) at 19.6m/s cut in the ego path with HWT=2.3s, then cut out

Third Car

Slow moving car (purple) at 11.1m/s in the 2nd lane

Driving Scenario Designer - PFACC_05_Curve_CutInOut.mat - Scenario Canvas

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): 14.7
Bank Angle (deg): 0

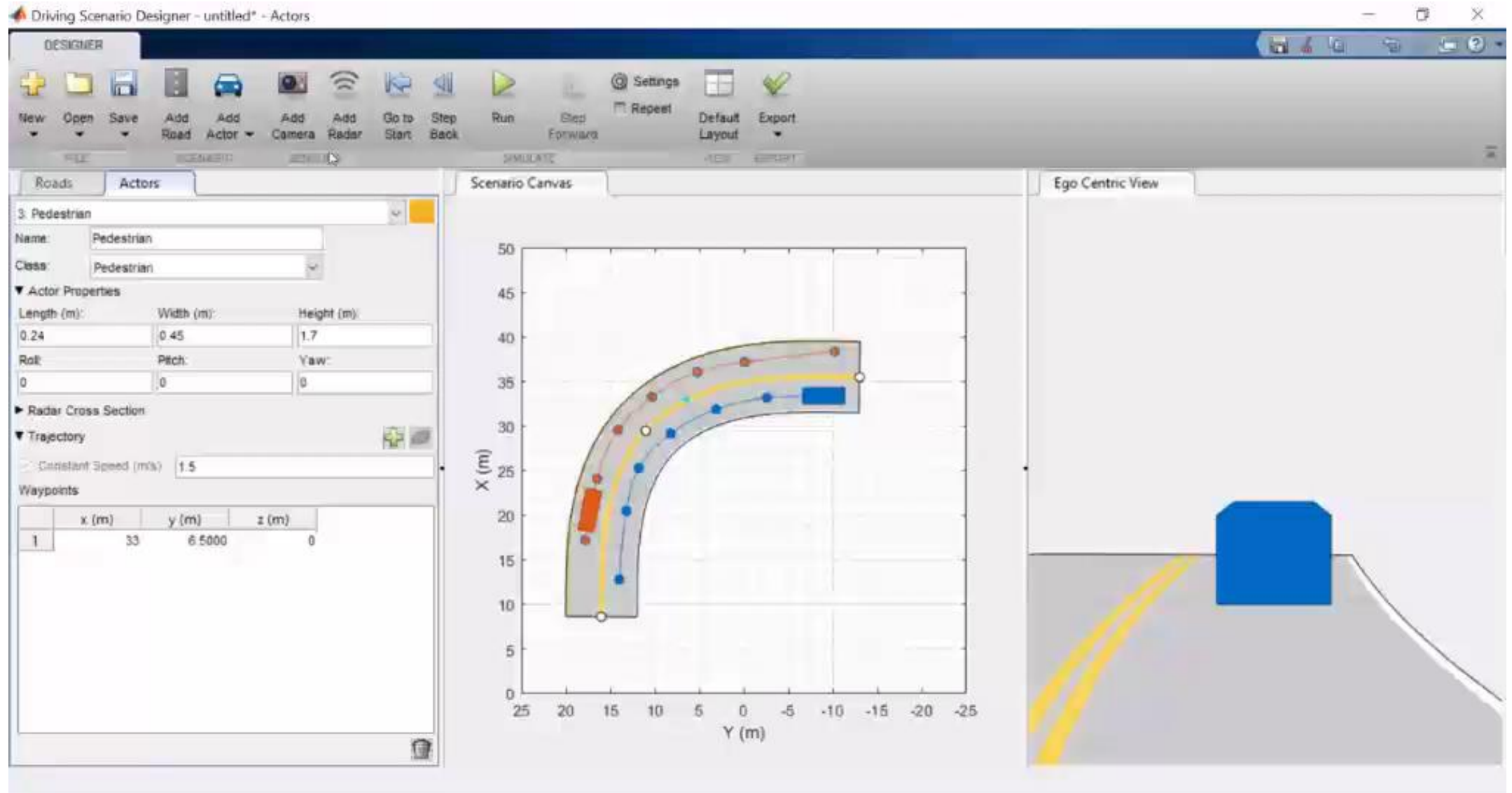
Lanes
Number of lanes: [2 2]
Lane Width (m): 3.6
Marking: 1.Solid

Road Centers

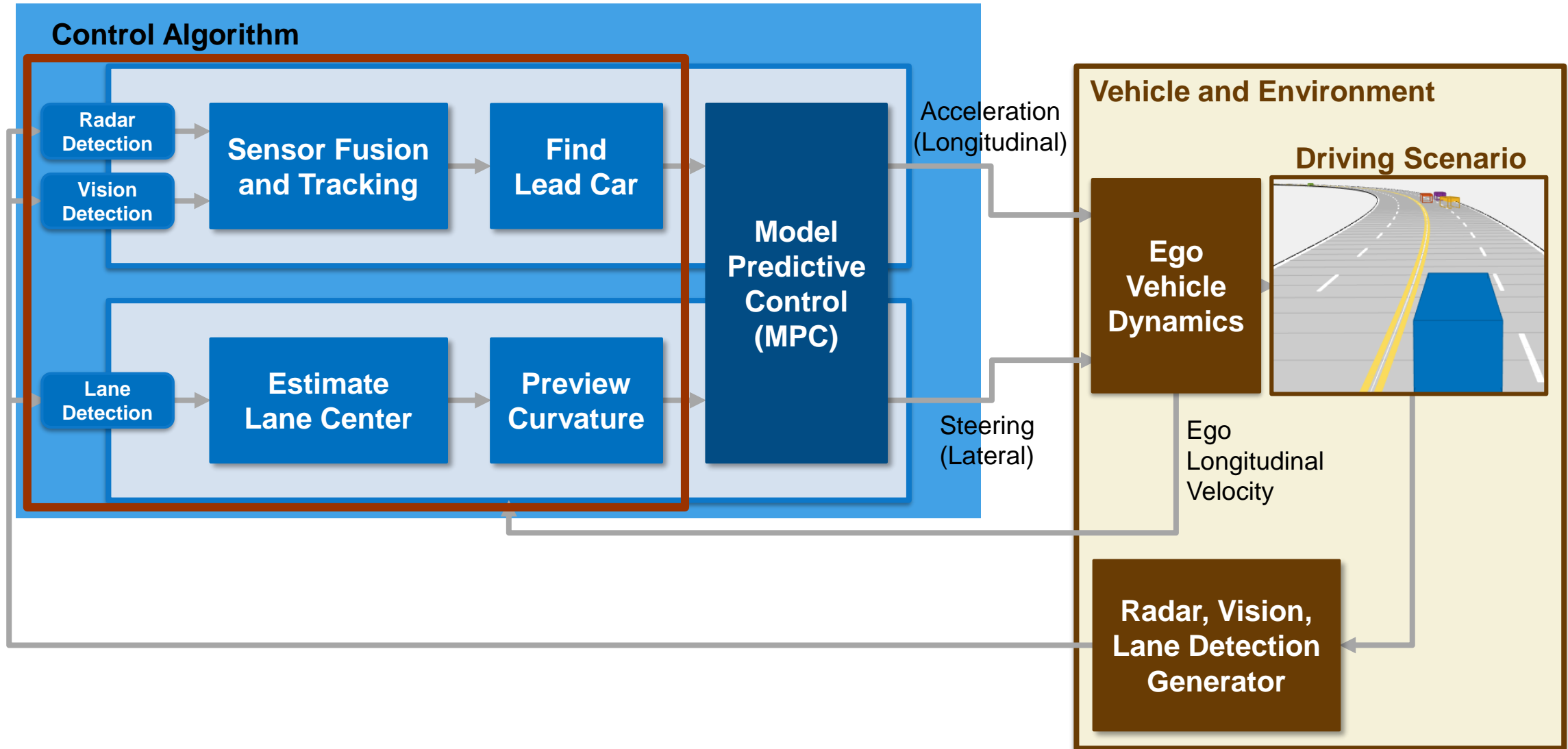
	x (m)	y (m)	z (m)
1	0	-500	
2	34.8782	-498.7820	
3	69.5866	-495.1340	
4	103.9558	-489.0738	
5	137.8187	-480.6308	
6	171.0101	-469.8463	
7	203.3683	-456.7727	
8	234.7358	-441.4738	
9	264.9596	-424.0240	
10	293.8926	-404.5085	

X (m) Y (m)

Add sensors to test scenario



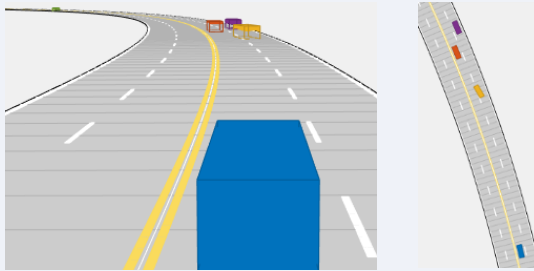
Develop and Test Vehicle Controller Traffic Jam Assist



Simulation with Simulink Model for Traffic Jam Assist

Test Description

Lead car cut in and out in curved highway
(curvature of road = 1/500 m)



Host car

initial velocity = 20.6m/s
HWT(Headway Time) to lead car = 4sec
HW(Headway) to lead car = ~80m
 v_set (set velocity for ego car) = 21.5m/s

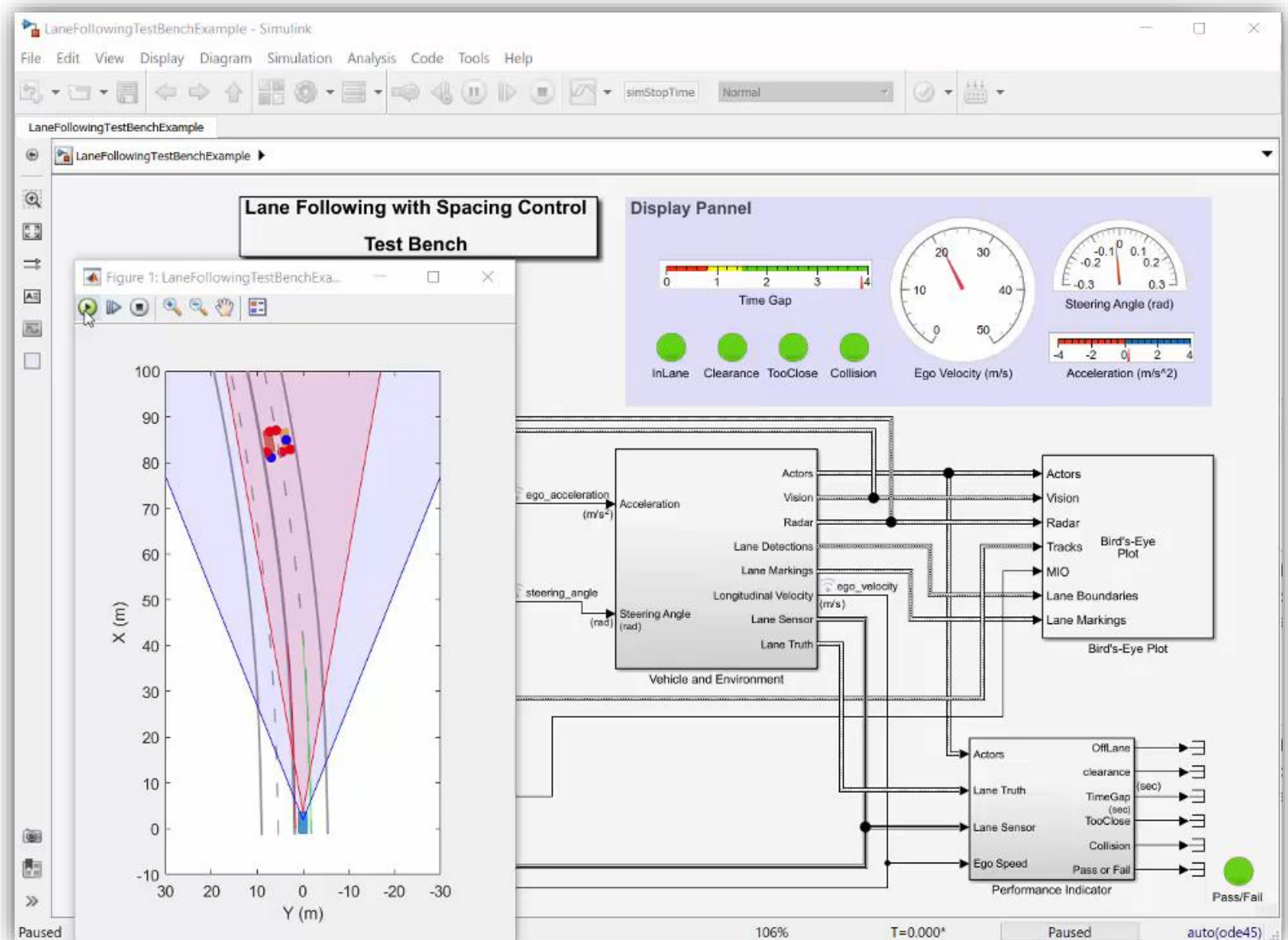
Lead Car

Initially, fast moving car (orange) at 19.4m/s

Passing car (yellow) at 19.6m/s cut in the ego path with HWT=2.3s, then cut out

Third Car

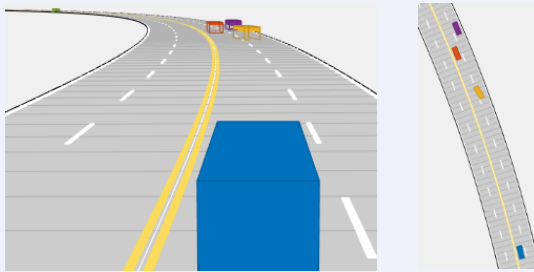
Slow moving car (purple) at 11.1m/s
in the 2nd lane



Simulation with Simulink Model for Traffic Jam Assist

Test Description

Lead car cut in and out in curved highway (curvature of road = 1/500 m)



Host car

initial velocity = 20.6m/s
 HWT(Headway Time) to lead car = 4sec
 HW(Headway) to lead car = ~80m
 v_set (set velocity for ego car) = 21.5m/s

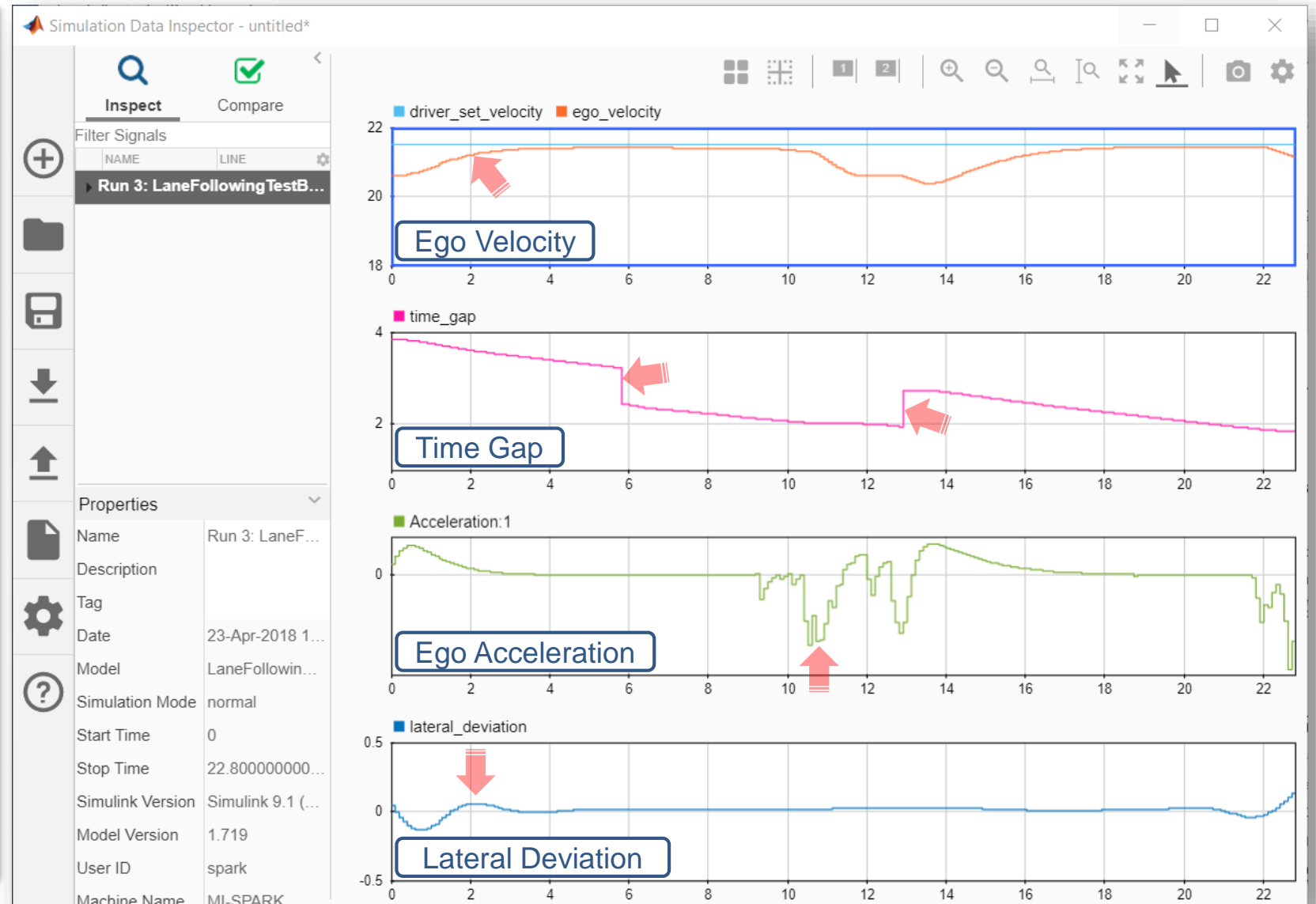
Lead Car

Initially, fast moving car (orange) at 19.4m/s

Passing car (yellow) at 19.6m/s cut in the ego path with HWT=2.3s, then cut out

Third Car

Slow moving car (purple) at 11.1m/s in the 2nd lane

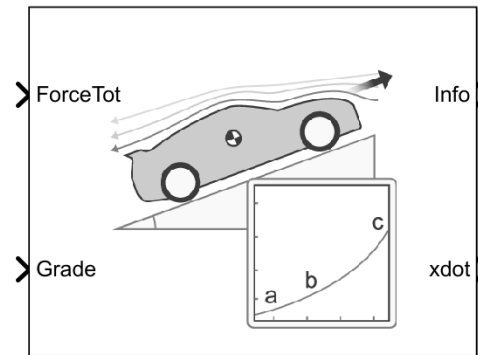


Agenda

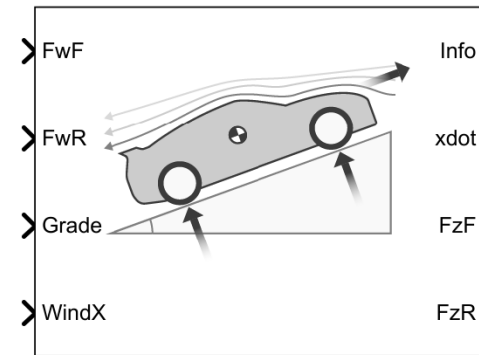
- Design model-predictive control-based vehicle controllers
- Run close-loop simulation with synthetic scenarios and test sensor fusion and control algorithms at a model level
- Improve simulation fidelity with gaming engine integration, vehicle dynamics modelling, and automated scenario creation from recorded data

Improve simulation fidelity: Include detailed vehicle dynamics

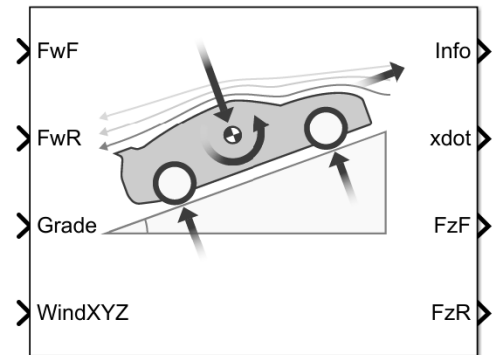
Vehicle Dynamics Blockset



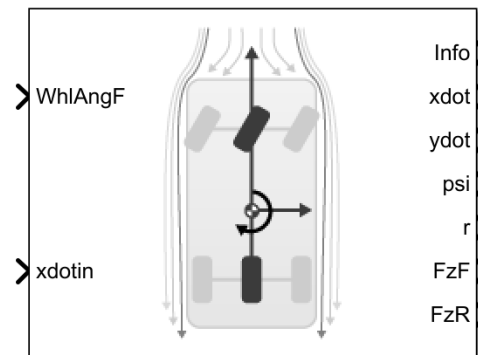
Vehicle Body Total Road Load



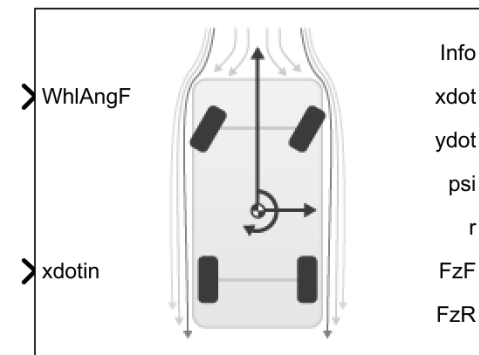
Vehicle Body 1DOF Longitudinal



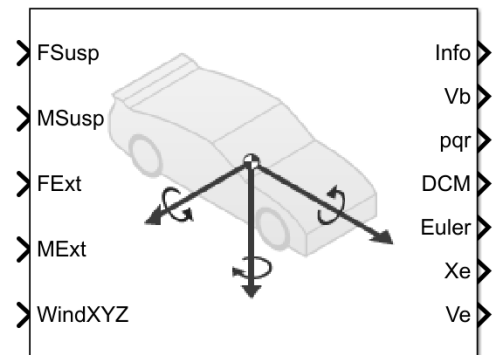
Vehicle Body 3DOF Longitudinal



Vehicle Body 3DOF Single Track



Vehicle Body 3DOF Dual Track

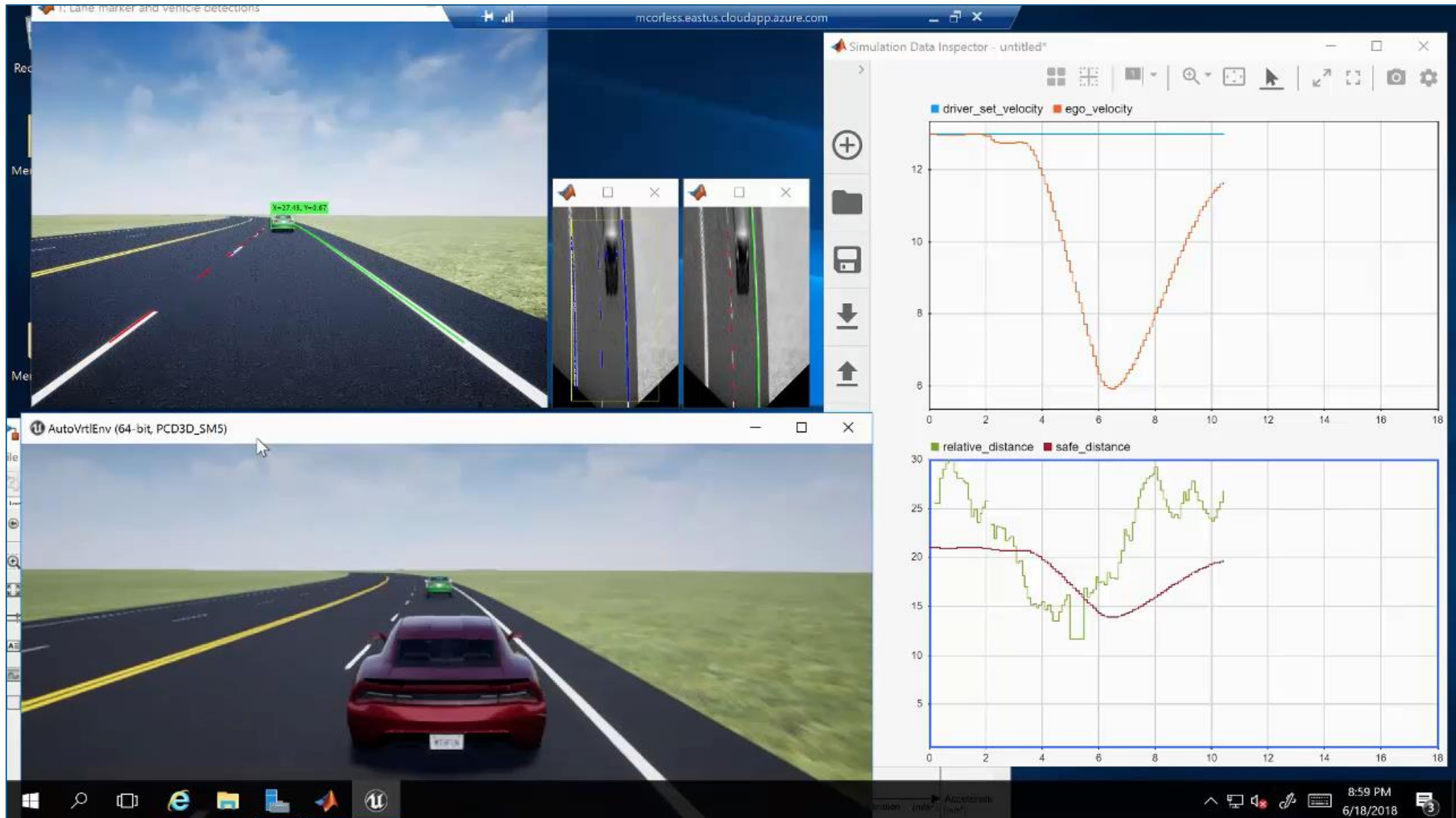


Vehicle Body 6DOF

Improve simulation fidelity: Include detailed vehicle dynamics

Vehicle Model	Description	Vehicle Body Degrees-of-Freedom (DOFs)	Wheel DOFs																								
Passenger 14DOF Vehicle	<ul style="list-style-type: none"> Vehicle with four wheels Available as model variant in the maneuver reference applications 	Six <table border="1"> <thead> <tr> <th colspan="2">Translational</th> <th colspan="2">Rotational</th> </tr> </thead> <tbody> <tr> <td>Longitudinal</td> <td>✓</td> <td>Pitch</td> <td>✓</td> </tr> <tr> <td>Lateral</td> <td>✓</td> <td>Yaw</td> <td>✓</td> </tr> <tr> <td>Vertical</td> <td>✓</td> <td>Roll</td> <td>✓</td> </tr> </tbody> </table>	Translational		Rotational		Longitudinal	✓	Pitch	✓	Lateral	✓	Yaw	✓	Vertical	✓	Roll	✓	Two per wheel - eight total <table border="1"> <thead> <tr> <th colspan="2">Translational</th> <th colspan="2">Rotational</th> </tr> </thead> <tbody> <tr> <td>Vertical</td> <td>✓</td> <td>Rolling</td> <td>✓</td> </tr> </tbody> </table>	Translational		Rotational		Vertical	✓	Rolling	✓
Translational		Rotational																									
Longitudinal	✓	Pitch	✓																								
Lateral	✓	Yaw	✓																								
Vertical	✓	Roll	✓																								
Translational		Rotational																									
Vertical	✓	Rolling	✓																								
Passenger 7DOF Vehicle	<ul style="list-style-type: none"> Vehicle with four wheels Available as model variant in the maneuver reference applications 	Three <table border="1"> <thead> <tr> <th colspan="2">Translational</th> <th colspan="2">Rotational</th> </tr> </thead> <tbody> <tr> <td>Longitudinal</td> <td>✓</td> <td>Pitch</td> <td></td> </tr> <tr> <td>Lateral</td> <td>✓</td> <td>Yaw</td> <td>✓</td> </tr> <tr> <td>Vertical</td> <td></td> <td>Roll</td> <td></td> </tr> </tbody> </table>	Translational		Rotational		Longitudinal	✓	Pitch		Lateral	✓	Yaw	✓	Vertical		Roll		One per wheel - four total <table border="1"> <thead> <tr> <th colspan="2">Rotational</th> </tr> </thead> <tbody> <tr> <td>Rolling</td> <td>✓</td> </tr> </tbody> </table>	Rotational		Rolling	✓				
Translational		Rotational																									
Longitudinal	✓	Pitch																									
Lateral	✓	Yaw	✓																								
Vertical		Roll																									
Rotational																											
Rolling	✓																										
Passenger 3DOF Vehicle	<ul style="list-style-type: none"> Vehicle with ideal tire 	Three <table border="1"> <thead> <tr> <th colspan="2">Translational</th> <th colspan="2">Rotational</th> </tr> </thead> <tbody> <tr> <td>Longitudinal</td> <td>✓</td> <td>Pitch</td> <td></td> </tr> <tr> <td>Lateral</td> <td>✓</td> <td>Yaw</td> <td>✓</td> </tr> <tr> <td>Vertical</td> <td></td> <td>Roll</td> <td></td> </tr> </tbody> </table>	Translational		Rotational		Longitudinal	✓	Pitch		Lateral	✓	Yaw	✓	Vertical		Roll		None								
Translational		Rotational																									
Longitudinal	✓	Pitch																									
Lateral	✓	Yaw	✓																								
Vertical		Roll																									

Improve simulation fidelity: Co-simulate with Unreal Engine



Game Engine Co-Simulation

Simulink

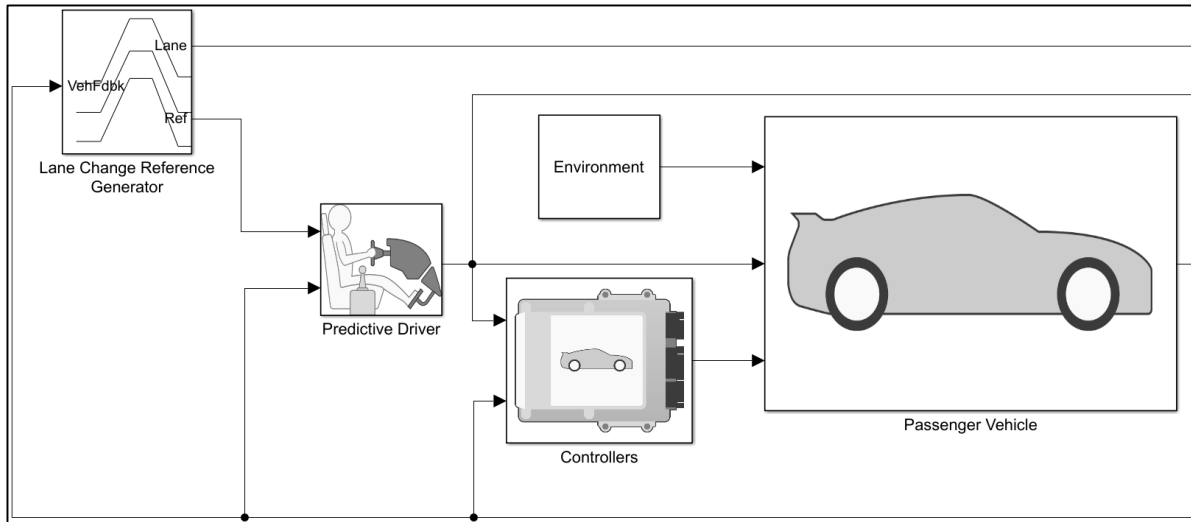
- Physics of vehicle
- Initialization of game engine camera

vehicle / camera location

Unreal Engine

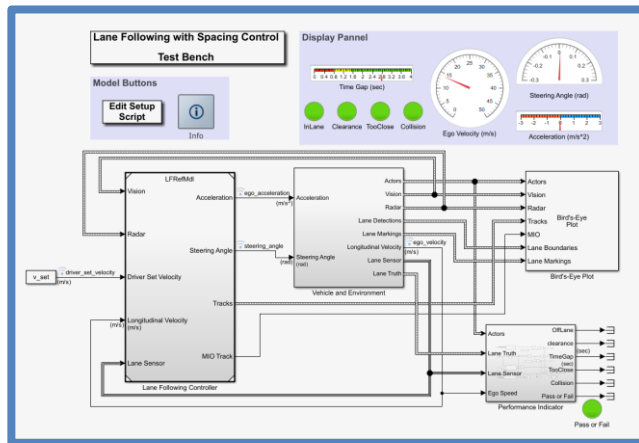
- Rendering / lighting
- Physics of non-Simulink objects
- Collision detection

camera image, ground height, ...



Develop and Test Vehicle Controller

Traffic Jam Assist: Key takeaways

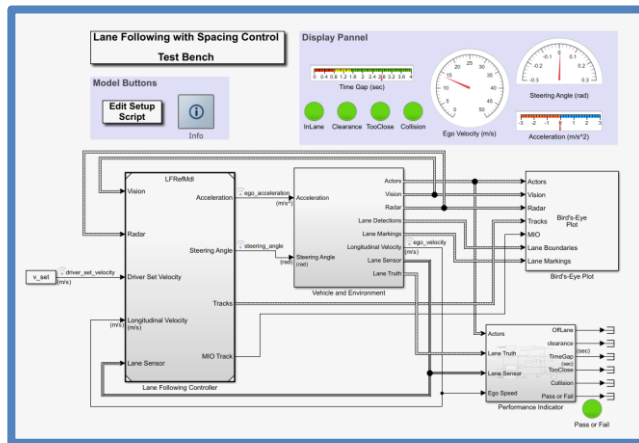


Design Traffic Jam Assist Controller

- Create driving scenario
- Synthesize sensor detection
- Include Vehicle Dynamics
- Design sensor fusion algorithm
- Design controller using MPC

Develop and Test Vehicle Controller

Traffic Jam Assist: Next Steps



Design Traffic Jam Assist Controller

- Create driving scenario
- Synthesize sensor detection
- Include Vehicle Dynamics
- Design sensor fusion algorithm
- Design controller using MPC

Reference examples to get started:

1. [Lane Following Using Nonlinear Model Predictive Control](#)
2. [Lane Following Control with Sensor Fusion and Lane Detection](#)
3. [Testing a Lane-Following Controller with Simulink Test](#)

Hitachi develops model-predictive controller for adaptive cruise control in traffic jam



Model Predictive Control Approach to Design Practical Adaptive Cruise Control for Traffic Jam

Taku TAKAHAMA ¹⁾ Daisuke AKASAKA ²⁾

¹⁾ Hitachi Automotive Systems, Ltd.

4-7-1 Oma, Atsugi, Kanagawa, 243-8510, Japan (E-mail: taku.takahama.tz@hitachi-automotive.co.jp)

²⁾ The MathWorks GK

4-15-1 Akasaka, Minato-ku, Tokyo, 107-0052, Japan

The MPC controller was implemented in an embedded microprocessor (Renesas SH-4A, 32-bit processor), we confirmed the processing time of the MPC. The measurement result is shown in Fig. 5, the average time of the ACC function was 1.1ms. The C-code is automatically generated from a Simulink model using Embedded Coder[®].

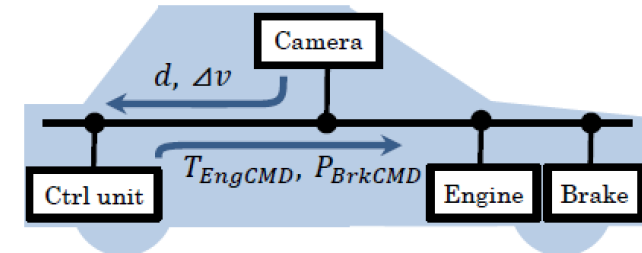


Fig. 4 System configuration of the experimental vehicle

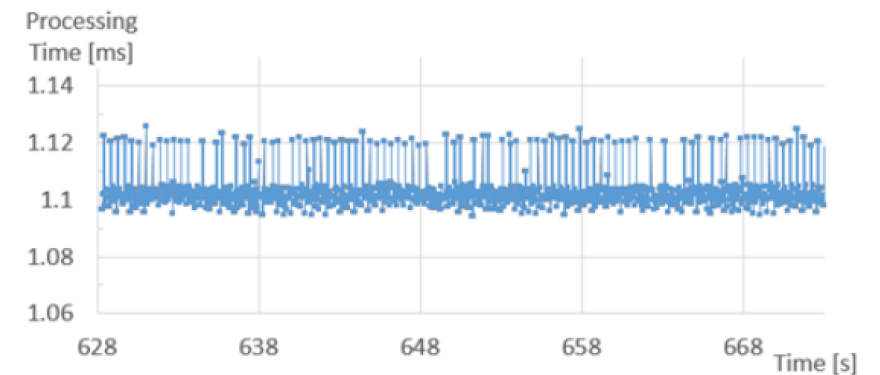


Fig. 5 The processing time of the proposed MPC

Hitachi paper published with SAE, Japan 2017

Hitachi also presented at 2017 MathWorks Expo, Japan

Call to action

- Visit the booth!

- Attend the session:
 - Simplifying Requirements-Based Verification with Model-Based Design

- MATLAB Tech Talk:
 - [Understanding Model Predictive Control](#)