

Respected Company with BIC Technology

모델기반설계를 이용한 자율주행 소프트웨어 개발 적용 사례

2020. 7. 16
MANDO ADAS BU
이동한 책임



I. Challenges

■ Development Platform

- Linux / ROS (Robotic Operating System)
- Easy to implement application software components with ROS

■ Simulation

- Unit test for each software component
- Test scenario generation
- Easy to utilize external resources

■ Model Predictive Control

- Optimization solver
- Code generation

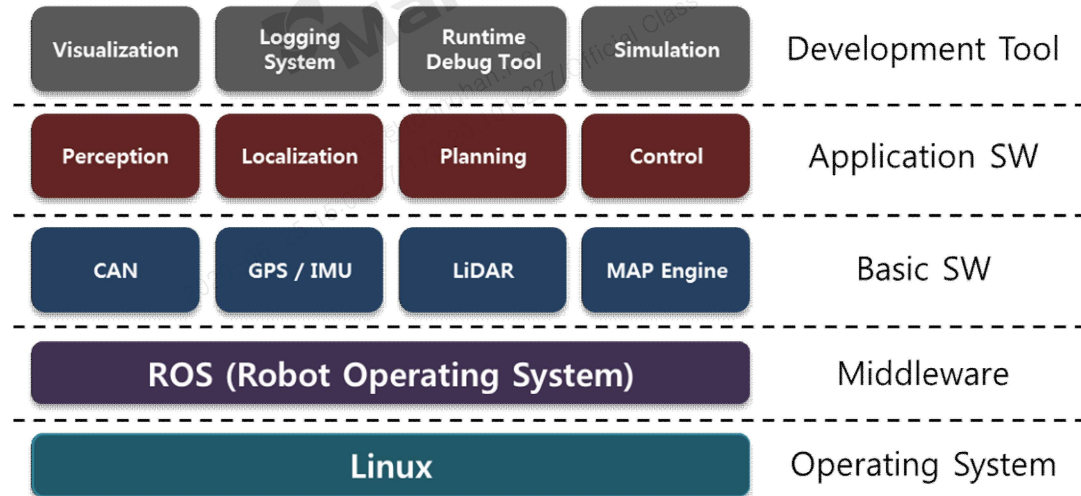


이동한(donghan.lee)
2020-06-25 15:03:27/172.20.101.227/Official Class

II. Simulink models with ROS interface

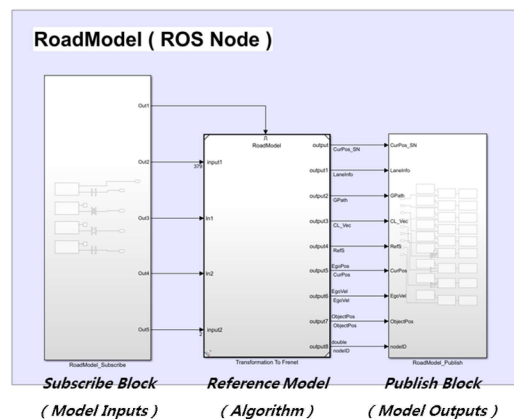
■ Development Environment

- SW Architecture based on Linux & ROS



- Two ways to develop Application SW

- Planning / Control : with MATLAB



- Perception / Localization : with C++

```

void CAN_2_CAN_msg_callback(const can_parser::FrameConstPtr& msg)
{
  canData canData;
  canData.n_channel = 4;
  canData.n_id = msg->id;
  for (int i = 0; i < 4; i++) ( canData.n_data[i] = (uint8_t)msg->data[i]);
  canData.n_timestamp = msg->timestamp;
  canData.n_timestamp = msg->timestamp;
  if (msg->is_error) { ROS_INFO("can Error Frame Received : CAN_C_CAN_msg"); }

  m_defencodinglogging_setCardata(canData);
  m_defencodingboschmccan_setCardata(canData);

  can_parser::CANDB_SCCLogging_2k_V100_pub_soc_msg;
  pub_soc_msg.header = msg->header;
  CAN2Topic_SCC(m_defencodinglogging_4pub_soc_msg);

  can_parser::CANDB_AD_CAN_msg_pub_boscherr_msg;
  pub_boscherr_msg.header = msg->header;
  CAN2Topic_BoschM8(m_defencodingboschmccan_4pub_boscherr_msg);

  soc_pub.publish(pub_soc_msg);
  boscherr_pub.publish(pub_boscherr_msg);
}

```

II. Simulink models with ROS interface

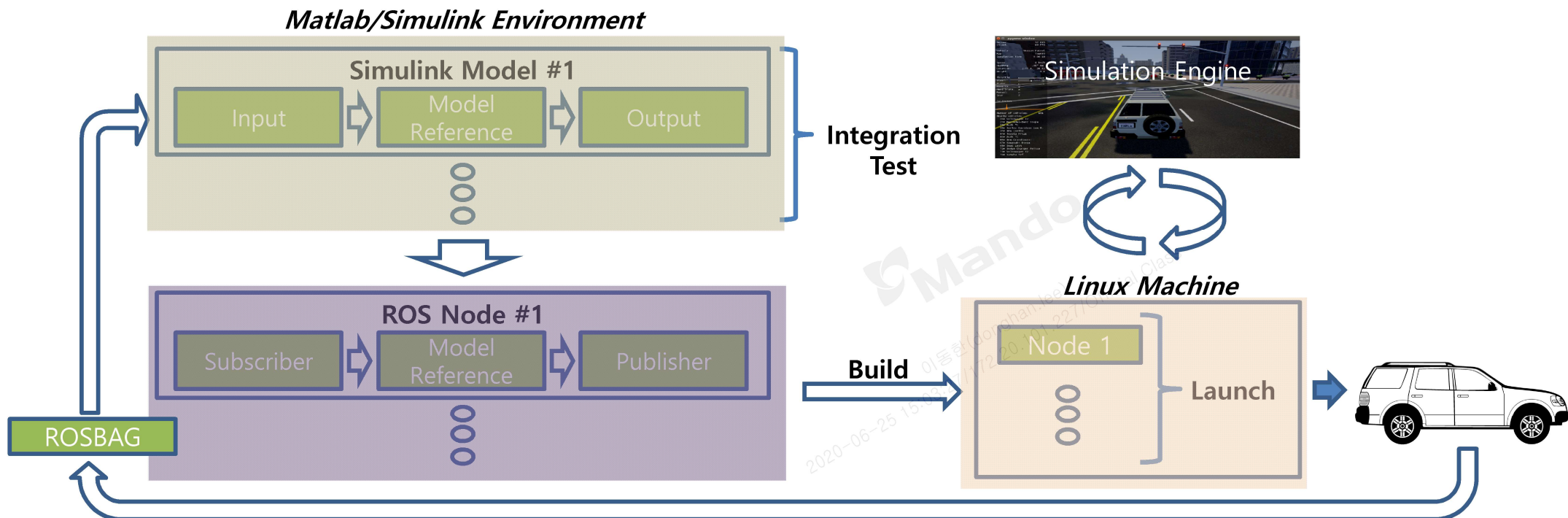
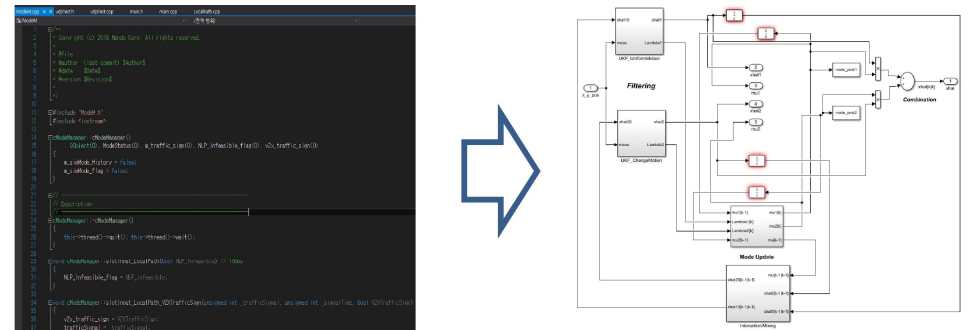
■ Pros

- Reduce the development cycle
- Improve reusability
- Easy to conduct unit / integration tests

■ Cons

- Difference from actual environment

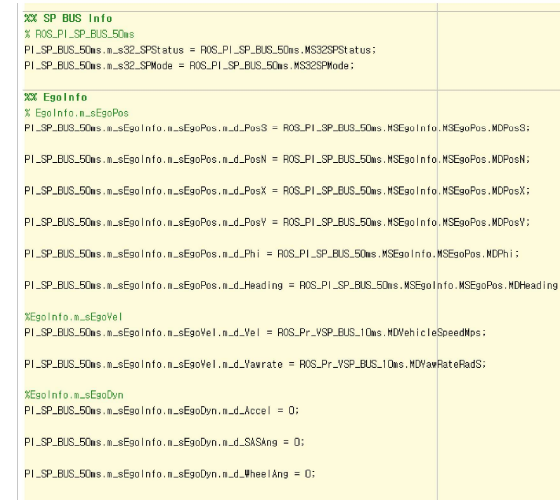
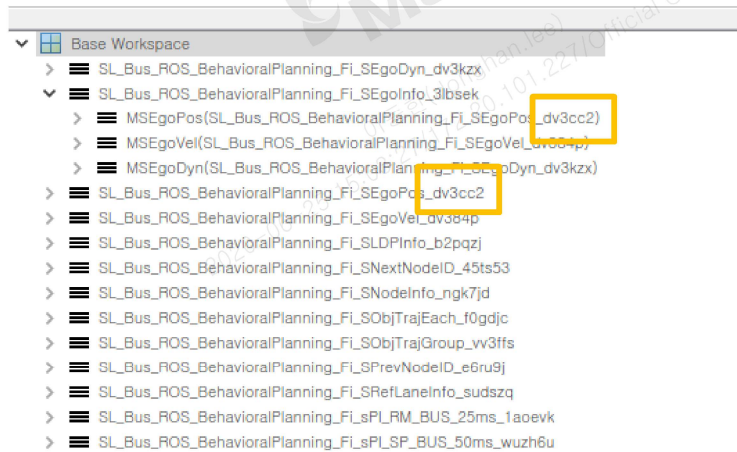
■ Model Configuration



II. Simulink models with ROS interface

■ Issues

- NOT correlate automatically generated ROS message bus with pre-defined ROS message



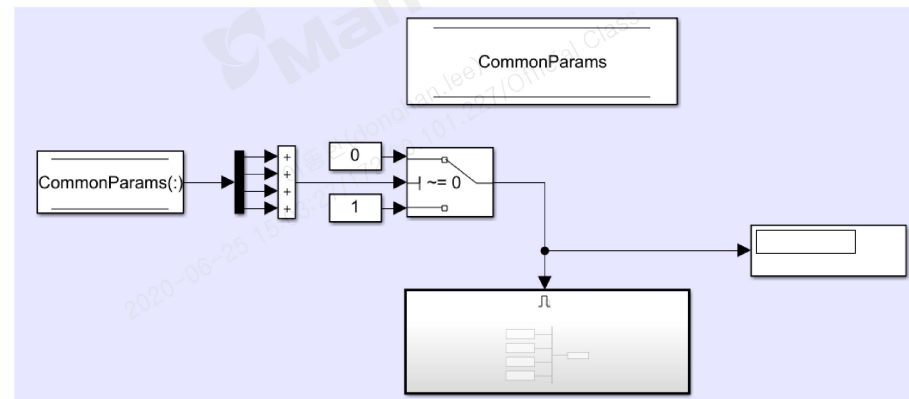
- 'ROS Get Parameter' block makes the computational problem
→ Resolved by the initialization with 'Enable block'

```

169 // Get the value for a named parameter from the parameter server.
170 * @param[out] dataPtr Pointer to initialized data variable. The retrieved parameter value will be
171 * written to this location
172 * @return Error code (=0 if value was read successfully, >0 if an error occurred)
173 */
174 template <class CppParamType, class ROSCppType>
175 uint8_t SimulinkParameterGetter<CppParamType, ROSCppType>::get_parameter(
176     CppParamType* dataPtr) {
177     XmlRpc::XmlRpcValue xmlValue;
178     ROSCppType paramValue;
179     bool paramRetrieved = false;
180
181     // Get parameter as XmlRpcValue and then parse it through our own function
182     if (nodePtr->getParam(paramName, xmlValue)) {
183         paramRetrieved = param_parser::getScalar(xmlValue, paramValue);
184     }
185
186     // Cast the returned value into the data type that Simulink is expecting
187     *dataPtr = static_cast<CppType>(paramValue);
188
189     // const uint8_t errorCode = process_received_data(dataPtr, paramRetrieved);
190     const uint8_t errorCode = 0;
191     return errorCode;
192 }

```

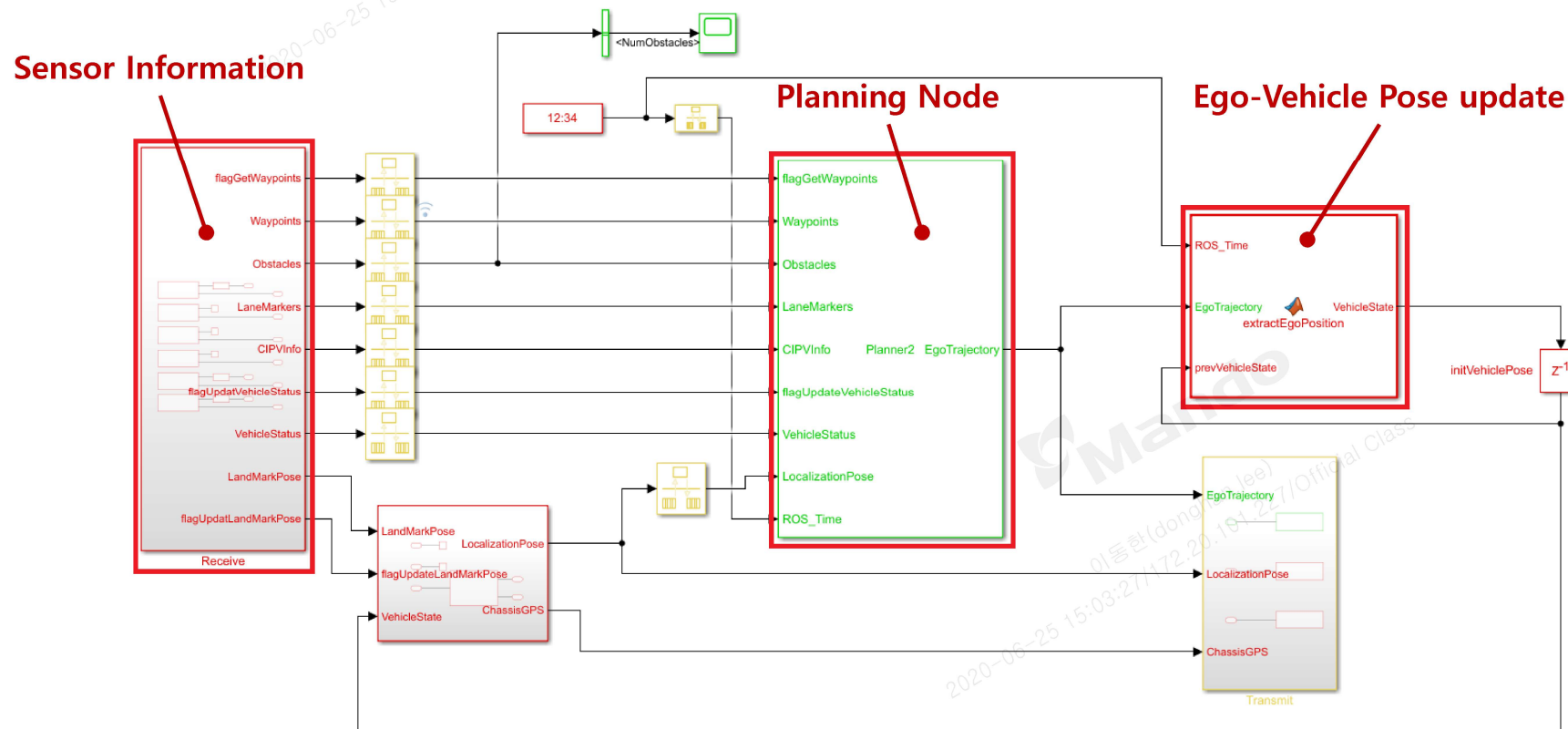
Line 189를 주석처리 하면, errorCode가 정의되지 않았다고 컴파일 에러가 발생하여
의미로 0값을 지정해주었습니다.



III. Simulation to evaluate a trajectory planning

■ Simulation Model

- Goal : Planning node evaluation in diverse driving conditions
- Requirements
 - Easy to generate driving scenarios (Agents, Road attributes, etc.)
 - Work with ROS nodes
 - Take into account interaction between agents in the driving scene



III. Simulation to evaluate a trajectory planning

■ Graphic User Interface

Control | Road Network Graph | **Scenario File/ROS Model Selection**

Driving Scenario File: C:\Users\Mando\Desktop\...
 External ROS Model: AutonomousDrivingWith\...
 ROS Master URL: localhost **ROS Connected**

ROS Time = 162 Simulink Model Time = 6.5 Time Difference = 0
 Duration of Simulation = 2.2951 Execution Time = 0.00877

Online Simulation Mode
 ROS Target
 Simulink

Simulation Speed: 0.5
 Simulation Time: 6.50
 SampleTime: 0.1

Run Offline Simulation
Run Online Simulation

ROS Initialization

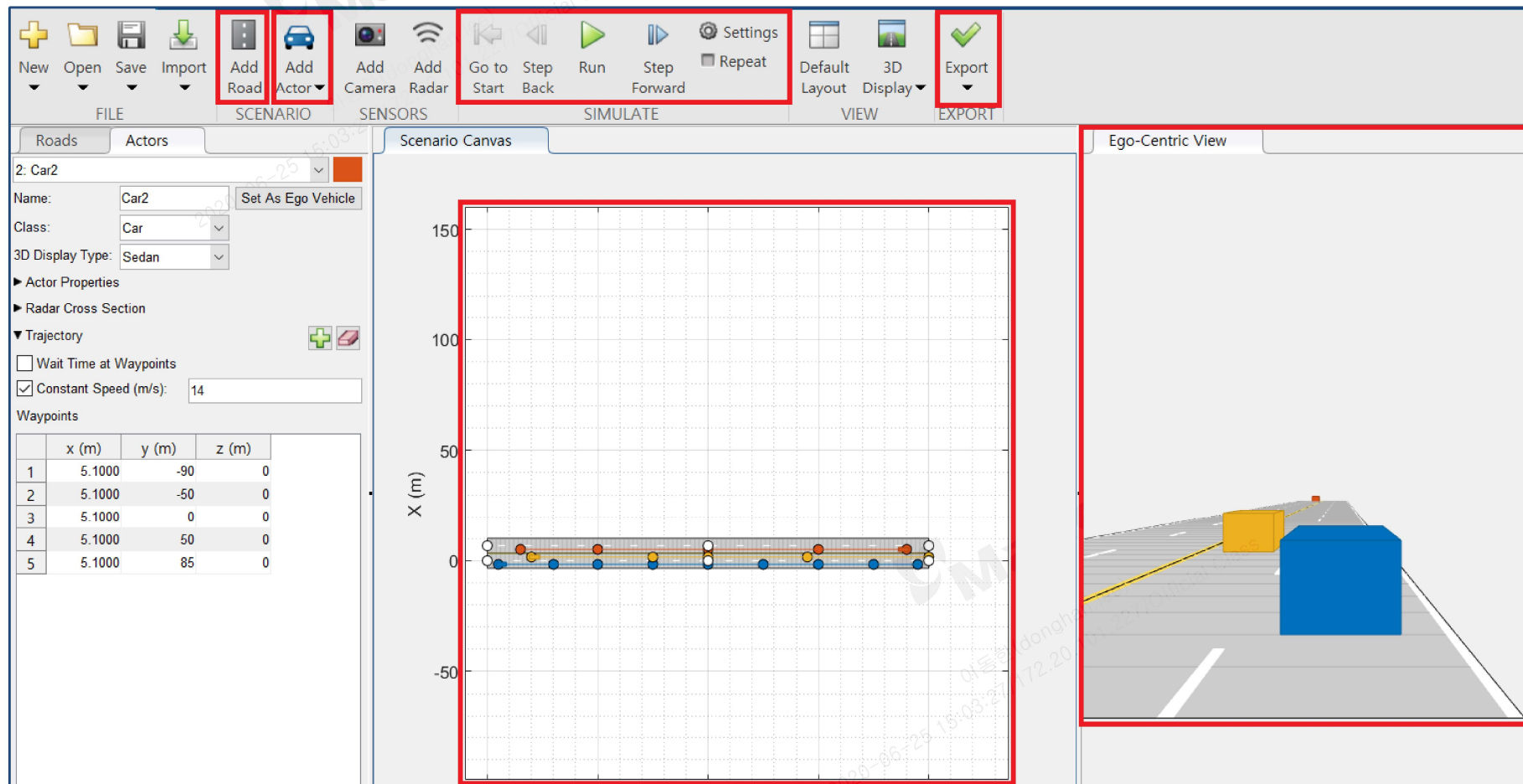
Simulation-Mode Selection

Simulation Execution

III. Simulation to evaluate a trajectory planning

■ Driving Scenario Designer

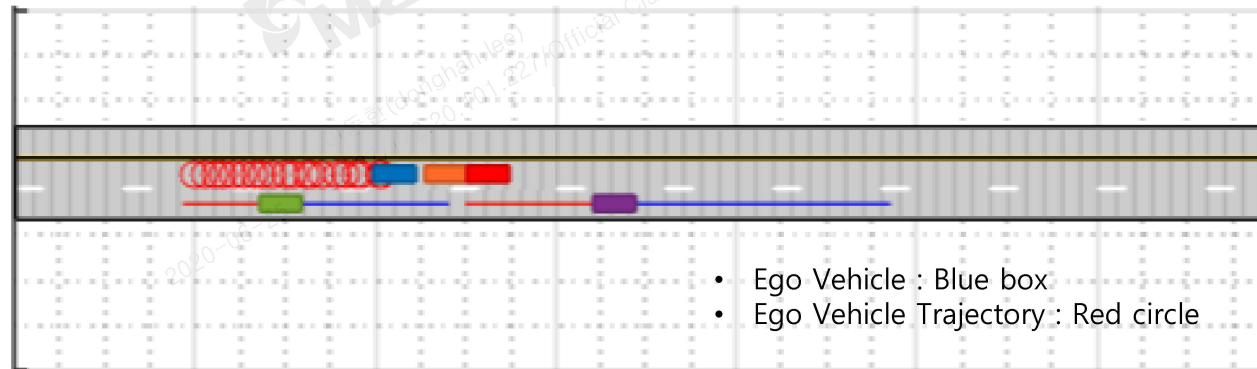
- Easy to build driving scenarios
- Export information regarding road-networks and actors



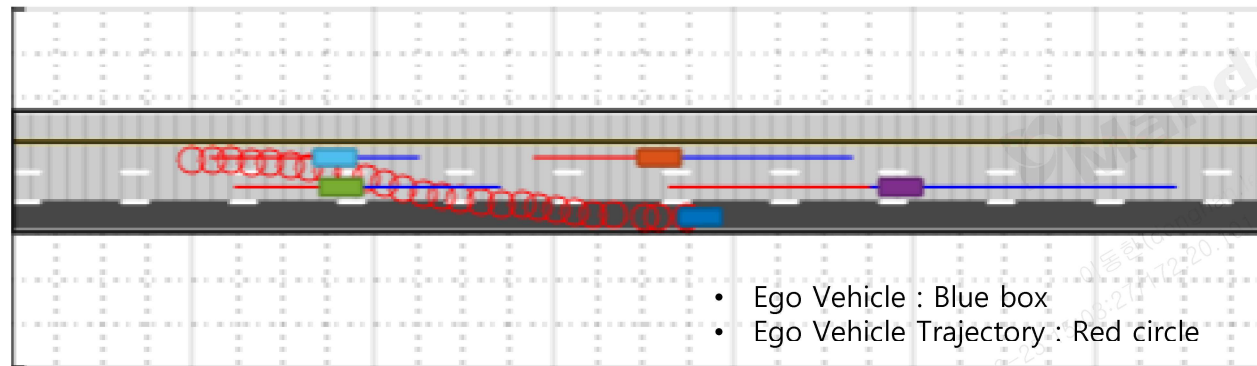
III. Simulation to evaluate a trajectory planning

■ Test case

- MRM(Minimum Risk Maneuver) / Unplanned Event
- Case 1. within the critical situation : Slow down inside the lane



- Case 2. without the critical situation : Lane change maneuvers



III. Simulation to evaluate a trajectory planning

■ Issues

- Repetitively simulation features on different driving conditions in a given scenario (Automated Testing)

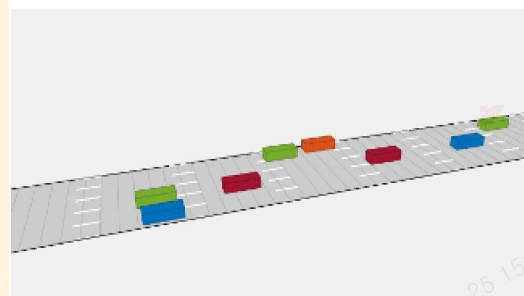
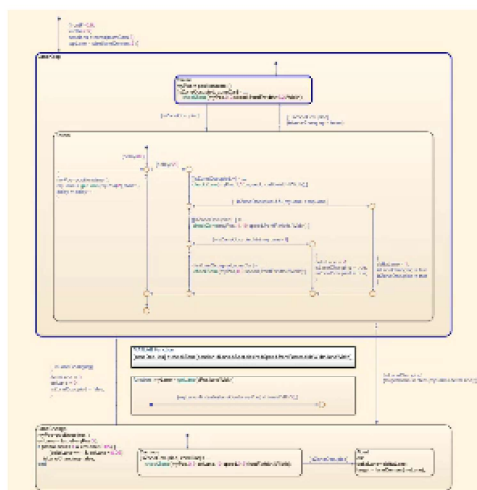
Time data	
Start	Thu Jul 30 2015 13:58:48 GMT+0200 (CEST)
End	Thu Jul 30 2015 13:59:17 GMT+0200 (CEST)
Duration	29 sec.

	Test Suites	Test Cases
✓ Passes	1	1
! Warnings	0	0
❌ Failures	0	0
ⓘ Skipped	0	0
Total	1	1

	Features	Scenarios	Steps
✓ Passes	1	4	27
! Warnings	0	0	0
❌ Failures	0	0	0
ⓘ Skipped	0	0	0
Total	1	4	27

[\[https://doc.froglogic.com/squish/6.0/ug-batchtesting.html\]](https://doc.froglogic.com/squish/6.0/ug-batchtesting.html)

- Interactions with other agents in a given scenario (e.g. Intersection / Merge-in / Merge-out)
→ Multi-agent simulation will be implemented based on the shipping demo. In Stateflow



IV. Model Predictive Control Toolbox

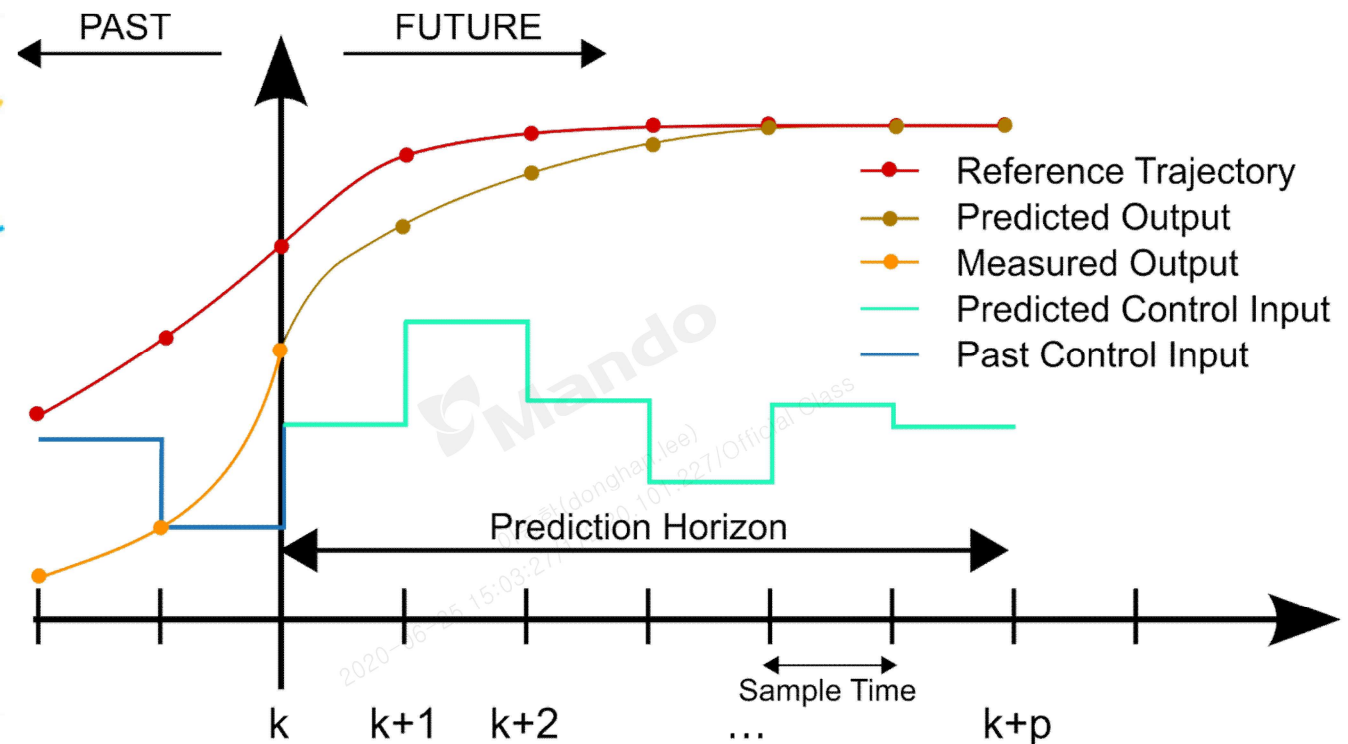
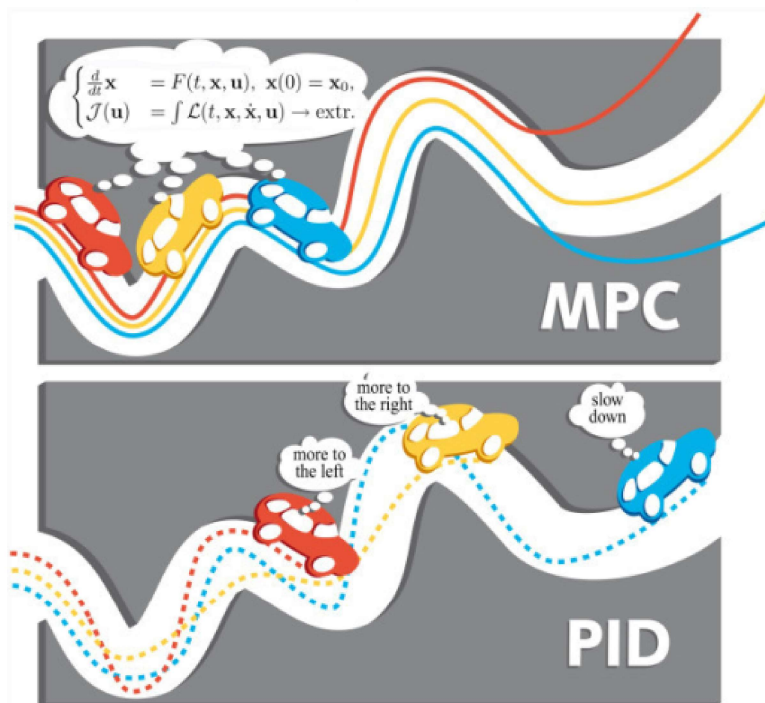
■ Model Predictive Control

■ Pros

- Any objective
- Any model
- Optimization with constraints

■ Cons

- Computationally demanding in the general case
- Stability & Feasibility are not guaranteed



IV. Model Predictive Control Toolbox

■ Constrained finite time optimal control problem

$$\min_{\kappa_{0:N-1|t}(\cdot)} \sum_{k=0}^{N-1} E \left[J(x_{k+1|t}^c, x_{k+1|t}^{ref}, u_{k|t}^c, u_{k-1|t}^c) \right]$$

$$\text{subject to: } x_{k+1|t}^c = f^c(x_{k|t}^c, u_{k|t}^c, w_{k|t}^c),$$

$$u_{k|t}^c = \kappa_{k|t}(x_{k|t}^c),$$

$$[u_{k|t}^c, u_{k-1|t}^c] \in \mathcal{U},$$

$$g(x_{k|t}^c, x_{k|t}^e) \leq 0,$$

$$(k = 0, \dots, N - 1)$$

$$x_{0|t}^c = x_t^c, \quad u_{-1|t}^c = u_{t-1}^c.$$

System Dynamics

Control policy

Input Constraint

State Constraint

■ Three different approaches depending on characteristics of uncertainty

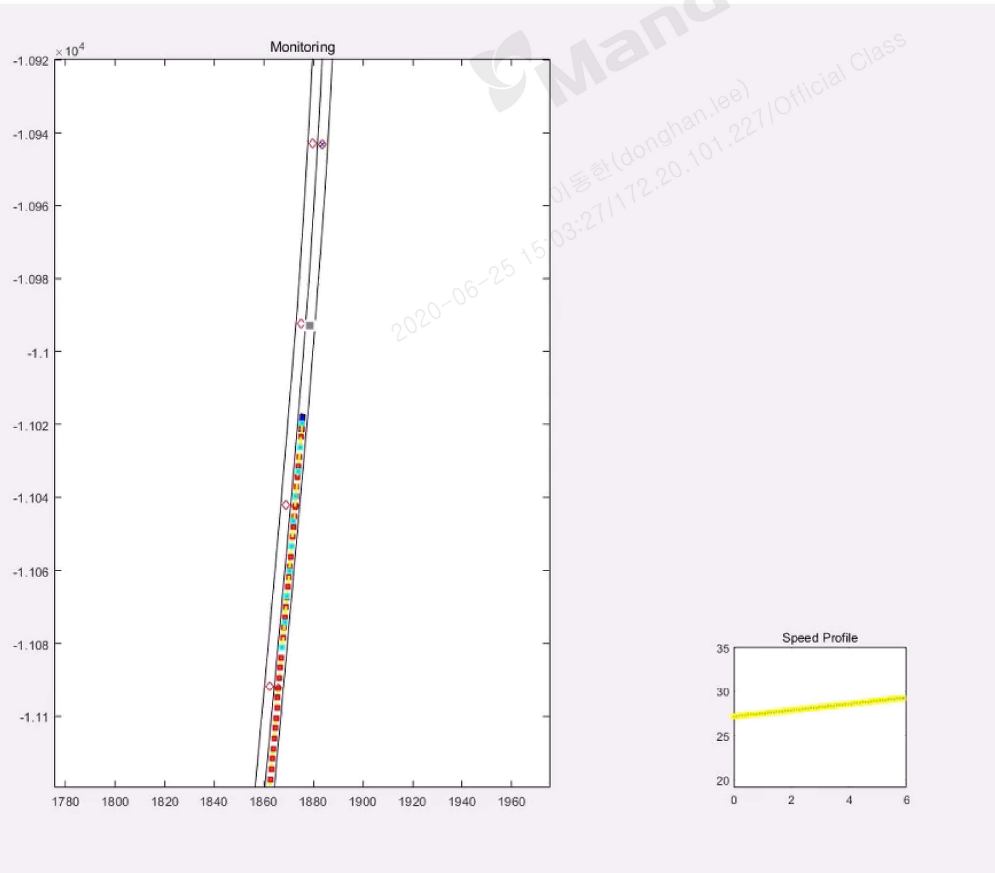
- Nominal MPC: $g(x_k^c, x_k^e) \leq 0$ with $w_i^* = \bar{w}_i^* \quad \forall i < k$

- Stochastic MPC: $\mathbf{P}(g(x_k^c, x_k^e) \leq 0) \geq 1 - \epsilon_k$ with $w_i^* \sim p_i^* \quad \forall i < k$

- Scenario-based MPC: $g(x_k^{c(s)}, x_k^{e(s)}) \leq 0$ with $w_i^* = w_i^{*(s)} \quad \forall s = 1, \dots, S, \quad \forall i < k$

IV. Model Predictive Control Toolbox

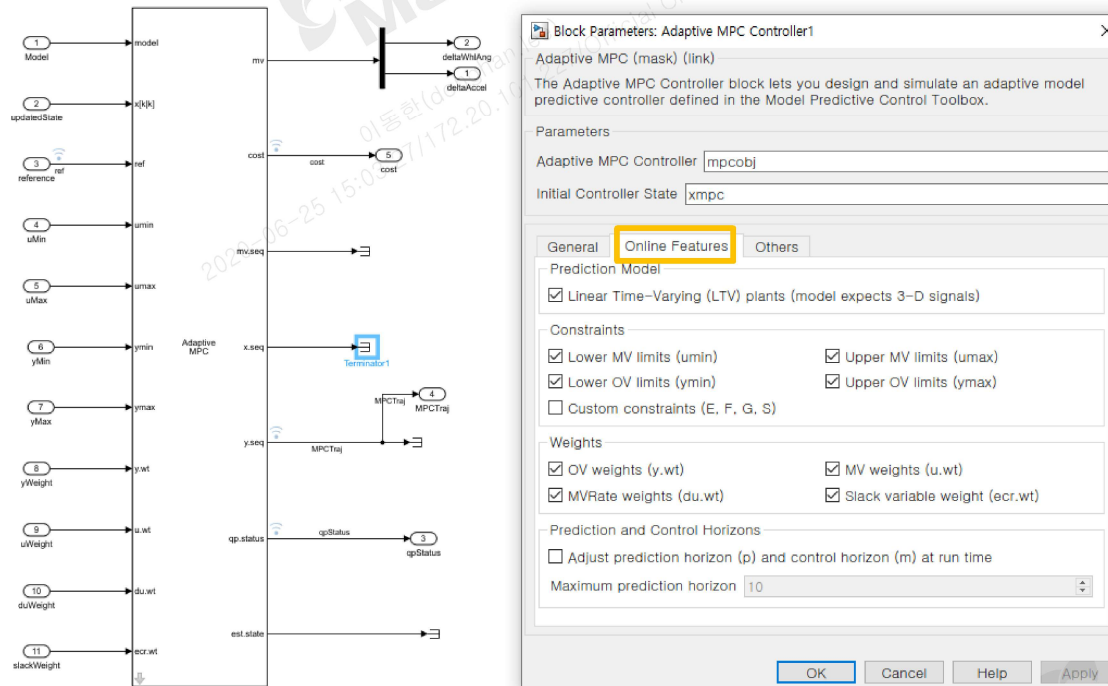
■ Simulation example



IV. Model Predictive Control Toolbox

■ Issues

- 'Adaptive MPC' has no MVRate-Constraints in Online Features



- Hard to define a problem formulation in the model
 - Script-based problem formulation option
 - Better for the design flexibility

Yalmip example code

```

Plant = ss(tf(1,[1 0 0]));
A = Plant.A;
B = Plant.B;
C = Plant.C;
D = Plant.D;
[nx,nu] = size(B);
Ts = 0.1;
Gd = c2d(Plant,Ts);
Ad = Gd.A;
Bd = Gd.B;

% Define data for MPC controller
N = 10;
Q = 10;
R = 0.1;

% Avoid explosion of internally defined variables in YALMIP
yalmip('clear')

% Setup the optimization problem
u = sdpvar(repmat(nu,1,N),repmat(1,1,N));
x = sdpvar(repmat(nx,1,N+1),repmat(1,1,N+1));

% Define simple standard MPC controller
% Current state is known so we replace this
x{1} = currentx;
constraints = [];
objective = 0;
for k = 1:N
    objective = objective + (r-C*x{k})*Q*(r-C*x{k})+u{k}'*R*u{k};
    constraints = [constraints, x{k+1} == Ad*x{k}+Bd*u{k}];
    constraints = [constraints, -5 <= u{k}<= 5];
end

% Solve!
sol = optimize(constraints,objective);

% ...and return the optimal input
uout = value(u{1});
  
```

V. Summary & Future work

■ Summary

- Model-based design with Mathworks solutions works well in L4 AD application
 - Development cycle time reduction
 - Support flexible platform
 - Easy to evaluate each SW-C
 - Provide fast and robust optimization solvers
- Case study in the model-based design to develop L4 AD

Challenges	Our Solution	Mathworks Support
Development Platform	Simulink models with ROS	Robotics System Toolbox
Simulation	Design a simulator for unit test	Automated Driving Toolbox Driving Scenario Designer
Model Predictive Control	Adaptive MPC model	Model Predictive Control Toolbox

■ Future work

- Keep improving model-based design environment
- Adopt data-driven approaches in order to improve the performance and the reliability of the L4 AD system