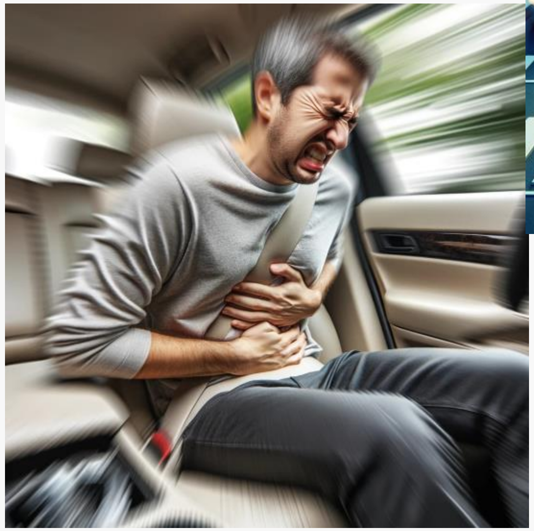
Adaptive control algorithms that provide fast and accurate vehicle control for vehicles with various characteristics

Masahiro Kubota

Team : norikenpi

1. Importance of high-precision vehicle control in autonomous driving
2. MPC as a high-precision control method
3. Modeling vehicles using driving data
4. Simulation and Simulation Results
5. Conclusion

Importance of high-precision vehicle control in autonomous driving

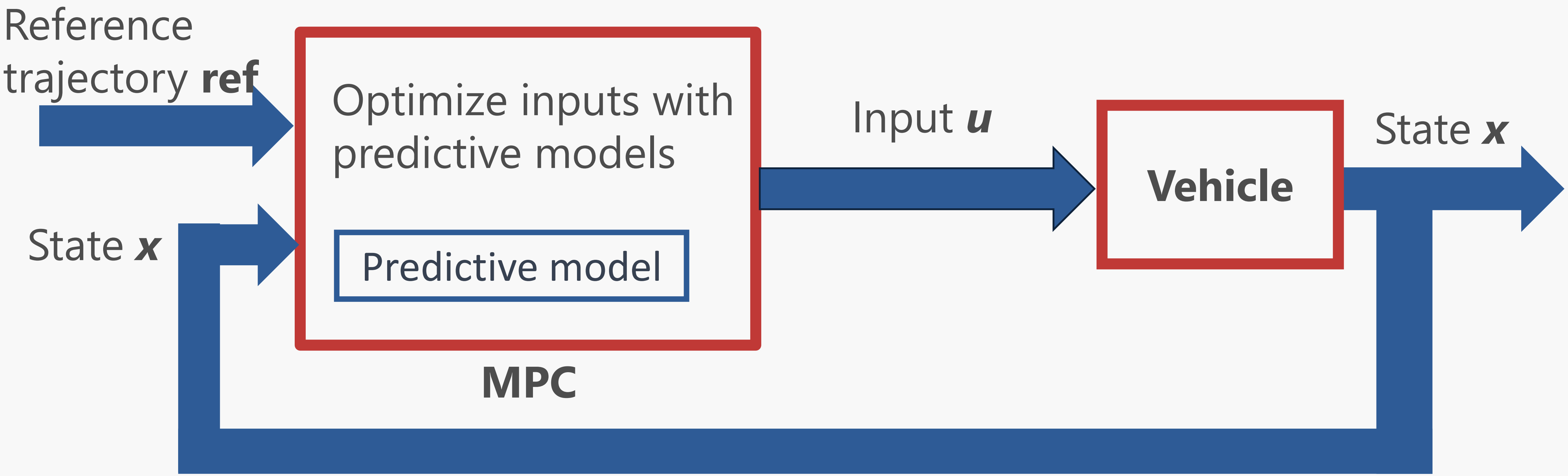


Safety :
Need to coexist with other vehicles, pedestrians and bicycles without contact.

Traffic rules :
Need to obey stop lines, speed limits, lanes, etc.

Comfort :
Sudden acceleration and steering operation are detrimental to comfort.

To make it easy for anyone to develop autonomous vehicles, algorithms are needed to achieve high-precision vehicle control for vehicles with various characteristics.



- **To solve the optimization problem, the control target must be represented by a discrete-time state-space model such as Equation 1.**

$$\boldsymbol{x}(k+1) = \boldsymbol{F}\boldsymbol{x}(k) + \boldsymbol{G}\boldsymbol{u}(k) + \boldsymbol{w}(k) \qquad (1)$$

$\boldsymbol{x}(k)$: State vector of the system at time k
$\boldsymbol{u}(k)$: Input vector to the system at time k
$\boldsymbol{w}(k)$: Disturbance vector to the system at time k

Need to find the optimum control input to minimize tracking error.

Need to be able to accurately predict the future state of the vehicle

$$\boldsymbol{x}(k+1) = \boldsymbol{F}\boldsymbol{x}(k) + \boldsymbol{G}\boldsymbol{u}(k) + \boldsymbol{w}(k) \qquad (1)$$

$\boldsymbol{x}(k)$: State vector of the system at time k
$\boldsymbol{u}(k)$: Input vector to the system at time k
$\boldsymbol{w}(k)$: Disturbance vector to the system at time k

Requires accurate discrete-time state-space model

How do we obtain a discrete-time state-space model of a vehicle that contains many complex elements?

MPC as a high-precision control method 3/3

- **MPC is used in Autoware for lateral control of vehicles, and three different models are implemented.**
  - Bicycle kinematics model with steering 1st-order delay (default)
  - **Bicycle kinematics model without steering delay**
  - Bicycle dynamics model considering slip angle

- **About bicycle kinematics model without steering delay (lateral motions of the vehicle)**

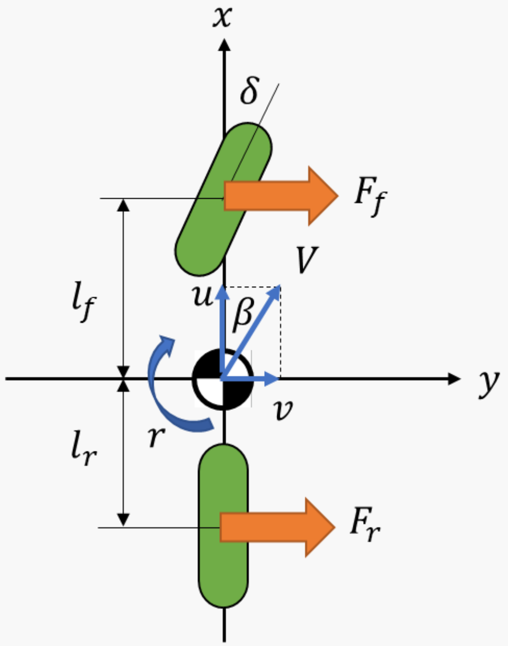


Fig. 1: Bicycle kinematics model [3]

$$\begin{bmatrix} y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & V\Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{V}{L}\Delta t & -\Delta t \end{bmatrix} \begin{bmatrix} \delta_k \\ r_{ref} \end{bmatrix} \tag{2}$$

y_k : Vehicle lateral deviation from the target position.
θ_k : Vehicle heading deviation from the target heading.
δ_k : Steering input
r_{ref} : Target yaw rate
V: Direction of travel velocity
L: Length from front wheel axle to rear wheel axle
Δt: simulation time step

Three or more wheels

Time constant

Black box software

Insufficient control cycles

Pedal lag

Dead zone

Tire stiffness

Air resistance

How to take into account?

Investigate the stiffness and other properties of the components

Contact the vehicle manufacturer

Describe complex dynamics mathematically

These take a lot of time and money.

Driving data

Vehicle states(position, posture, etc.) and inputs (steer angle, acceleration) at each time.

Least-squares method

ARX model

ARX model: a model that assumes that output at a given point in time depends on past output values and past inputs

$$y(k) + a_1 y(k-1) + \cdots + a_{na} y(k-na) = b_1 u(k-1) + \cdots + b_{nb} u(k-nb) \quad (3)$$

Rearranging an equation

Discrete-time state space model

In this case, $na = 1, nb = 1$

$$y(k+1) = -a_1 y(k) + b_1 u(k) \quad (4)$$

- **AWSIM**
  - Easy to integrate with Autoware
  - Bicycle model

https://github.com/tier4/AWSIM

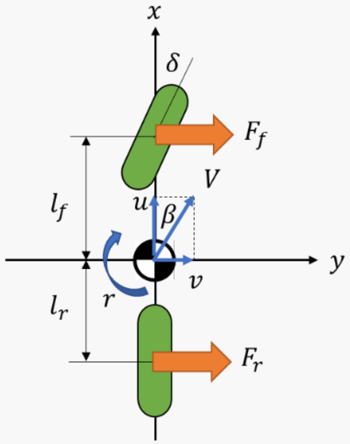


Fig. 1: Bicycle kinematics model [3]

- **Vehicle Body 3DOF Dual Track (Simulink)**
  - Four-wheel model
  - Lateral Corner Stiffness and Relaxation Dynamics
  - Air resistance etc.

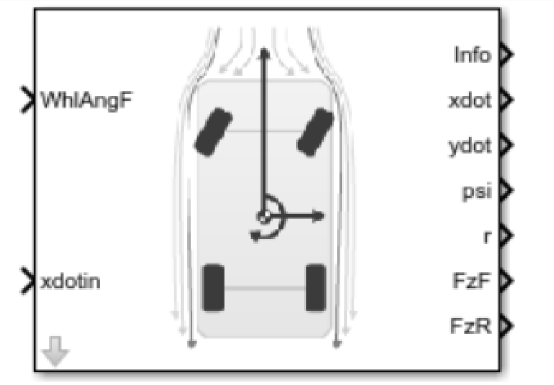


Fig. 2: Vehicle Body 3DOF Dual Track [1]

In this case, simulink's Vehicle Body 3DOF Dual Track was used as the vehicle motion simulator.

Driving data acquisition simulator

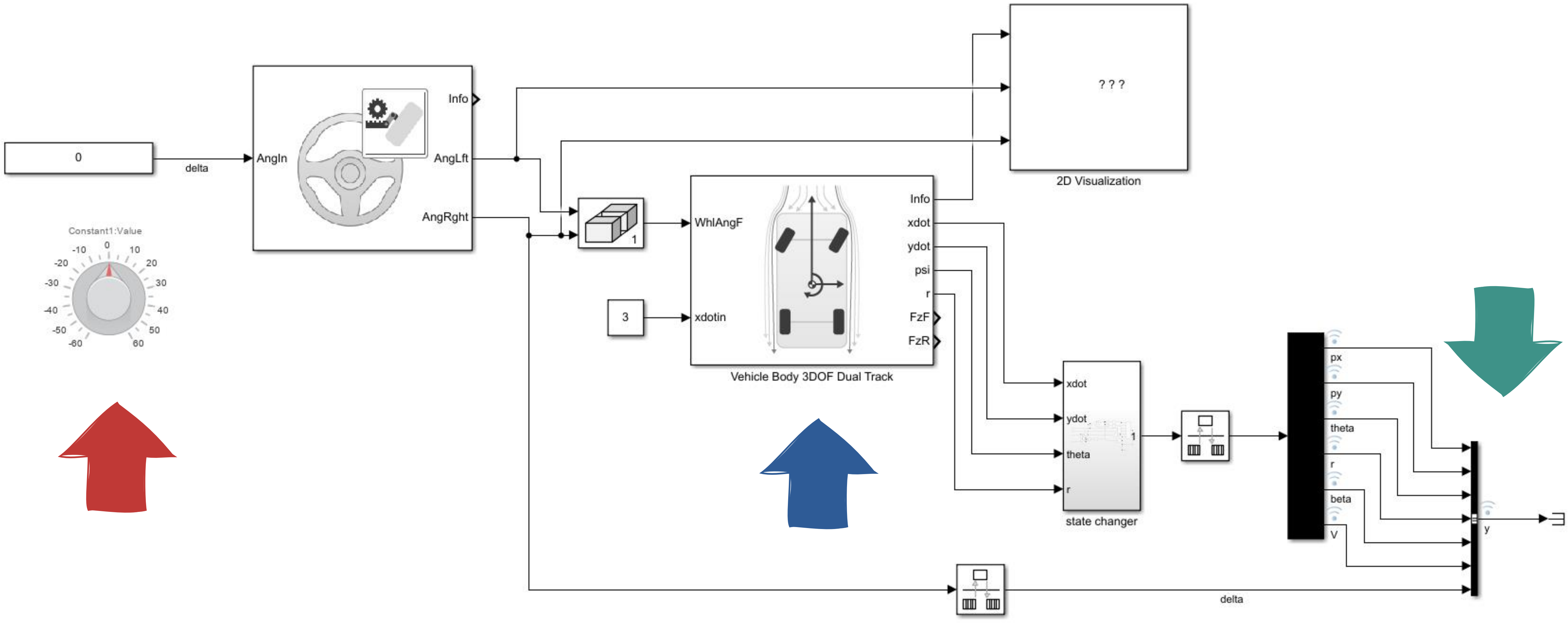
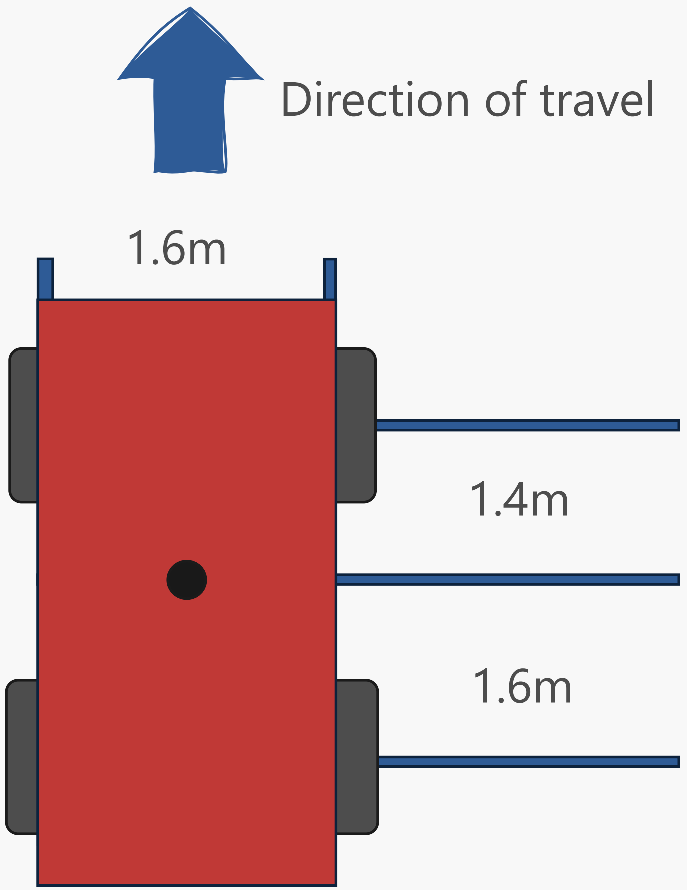


Table. 1: Main simulation parameters

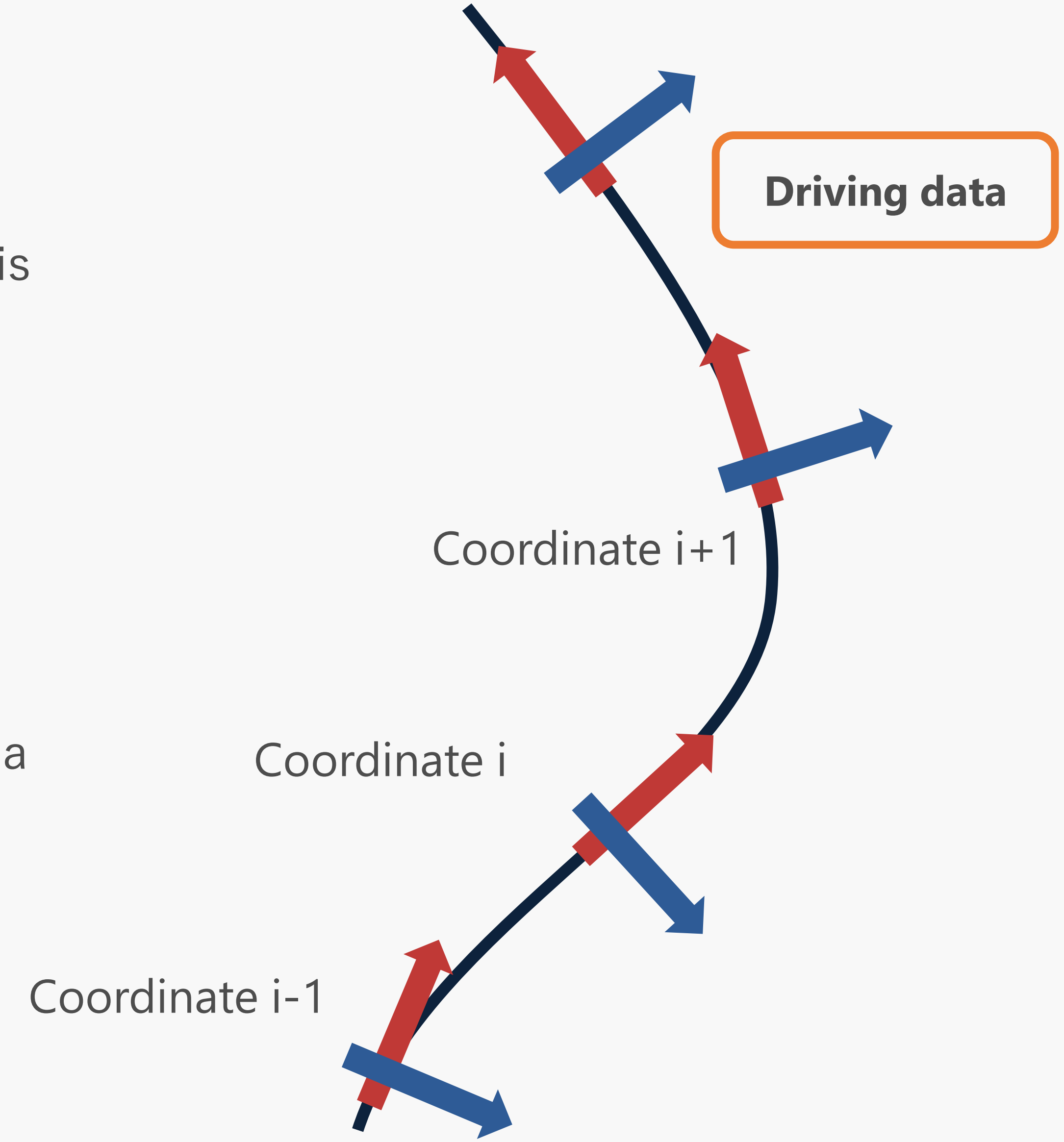
The speed in the direction of vehicle travel	3m/s (11km/h)
Length of driving data	110sec
Driving data interval	0.1sec
Distance from center of gravity to front axle	1.4m
Distance from center of gravity to rear axle	1.6m
Truck width	1.6m
Vehicle weight	2000kg



Simulation Overview 2/2

- **System identification**
  - Since the longitudinal and lateral motions of the vehicle are considered separately, the coordinate transformation must be sequential so that the x-axis is always in the direction of the vehicle's travel.
  - The 110-second driving data set was cut into one-second segments, and the coordinates were transformed with respect to the positional posture of the first driving data in each driving data set.

- **Evaluation of vehicle control performance**
  - Comparison of tracking errors between MPC using a bicycle kinematics model and MPC using a model identified by the system from driving data.



- **The parameters of the ARX model were estimated and transformed into a discrete-time state-space model based on the driving data with trajectories as shown in Fig. 3, resulting in the following discrete-time state-space model.**

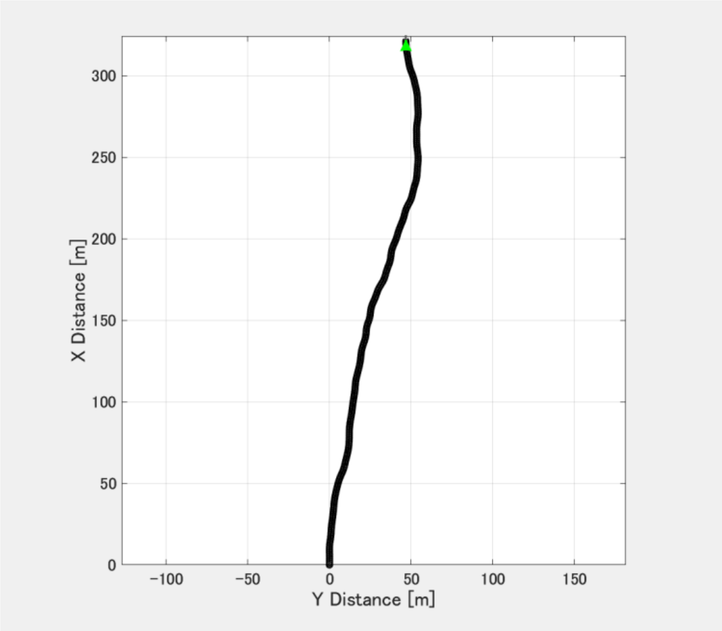


Fig. 3: Trajectory of driving data

$$\begin{bmatrix} y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0.3 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0.1 & -0.1 \end{bmatrix} \begin{bmatrix} \delta_k \\ r_{ref} \end{bmatrix} \quad (5)$$

Discrete-time state-space model of the bicycle kinematics model.

$$\begin{bmatrix} y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} 1.027 & 0.8819 \\ 0.0006119 & 0.9975 \end{bmatrix} \begin{bmatrix} y_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} 0.6952 & -0.3238 \\ 0.003328 & 0.8923 \end{bmatrix} \begin{bmatrix} \delta_k \\ r_{ref} \end{bmatrix} \quad (6)$$

Discrete-time state-space model of the model with system identification from driving data.

Simulation Results 2/2

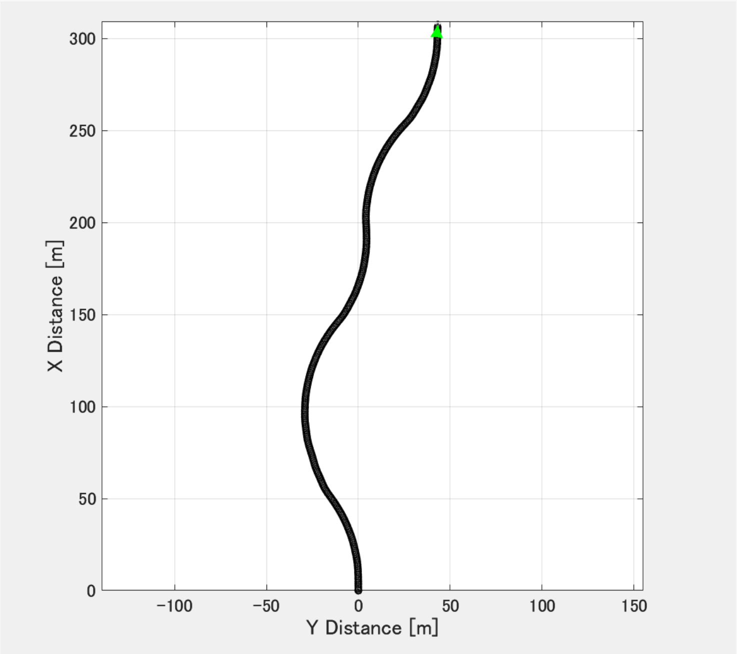


Fig. 4: Reference trajectory

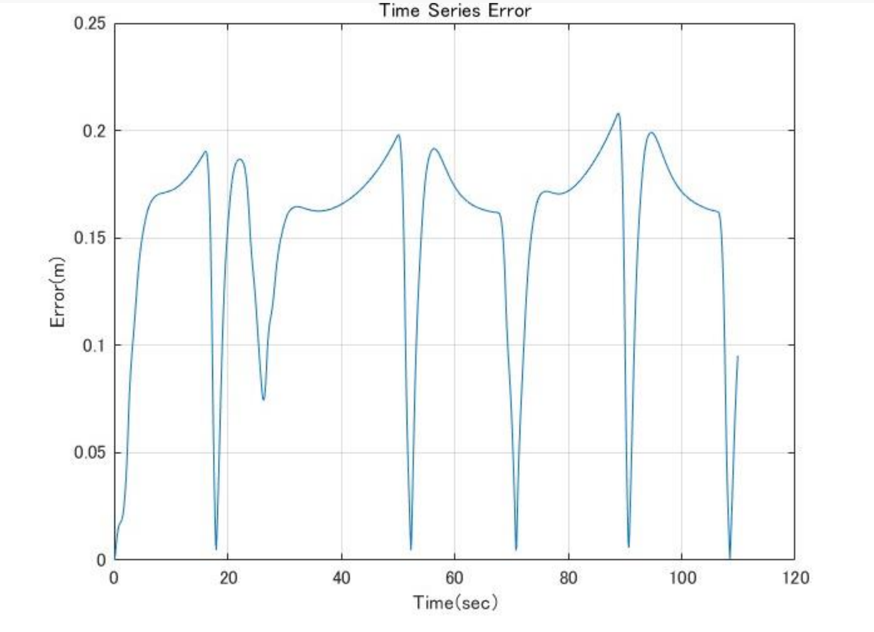


Fig. 5: MPC tracking error using a bicycle kinematics model.

RMSE=15.23cm

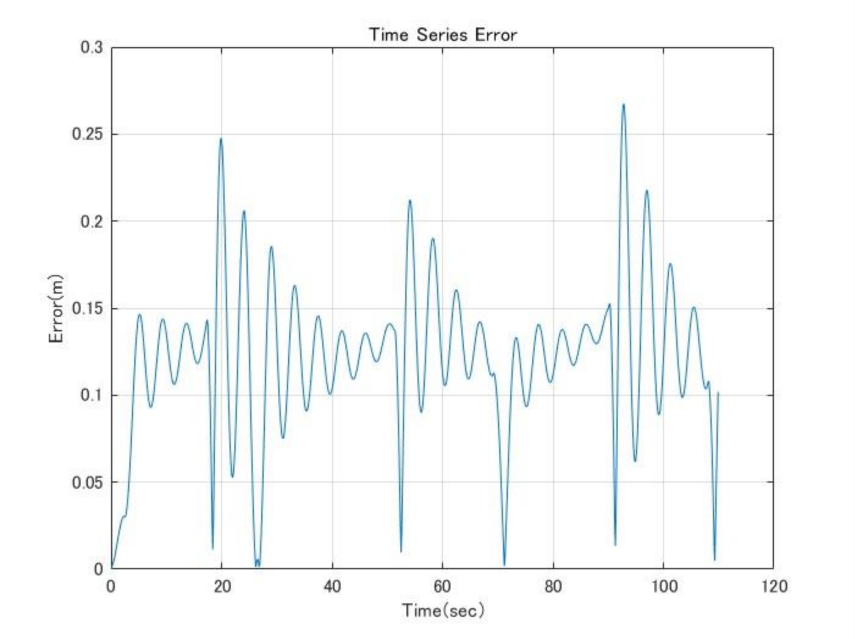


Fig. 6: MPC tracking error using a model with system identification from driving data.

RMSE=12.15cm

The vehicle control was more accurate using a model with system identification from driving data.

Conclusion

- MPC using a model that was system-identified from driving data allowed for more accurate vehicle control than MPC using a bicycle kinematics model.
- The simulator with the simulink model and matlab code constructed in this study can be used to evaluate other system identification and control methods.
- The simulink model and matlab code used in this study have been uploaded to the following github repository.
  https://github.com/norikenpi/adaptive_mpc

Possible methods for more precise vehicle control

- **Non-linear modeling of the vehicle with neural nets, etc., and sequential linearization to perform MPC.**
- **System identification is performed by increasing the order of states so that more complex states can be represented.**