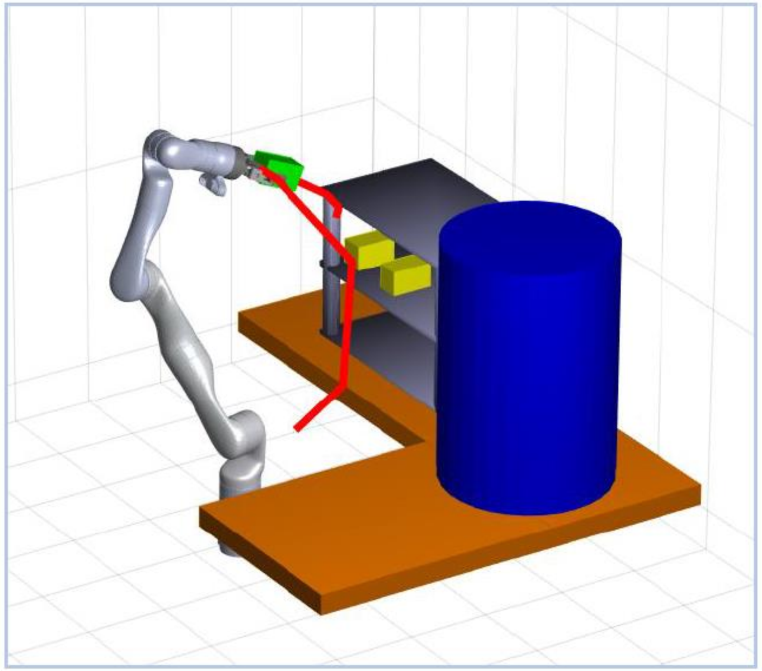
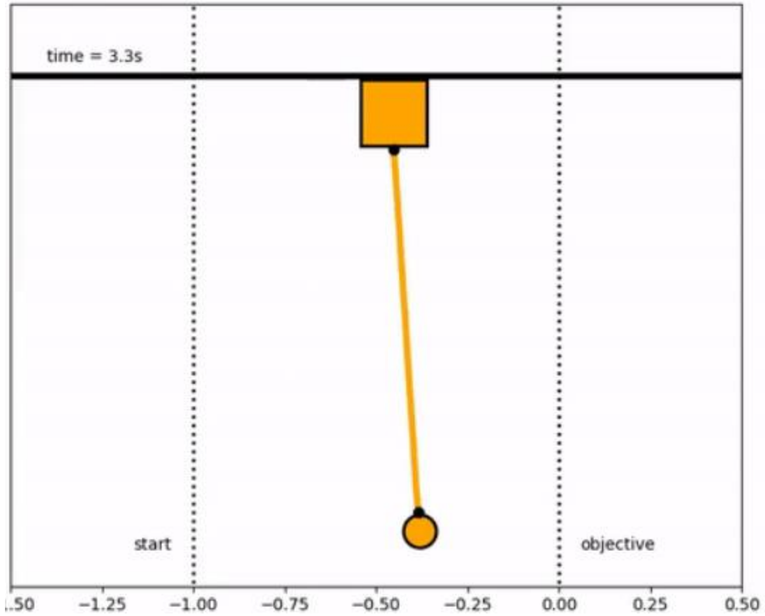
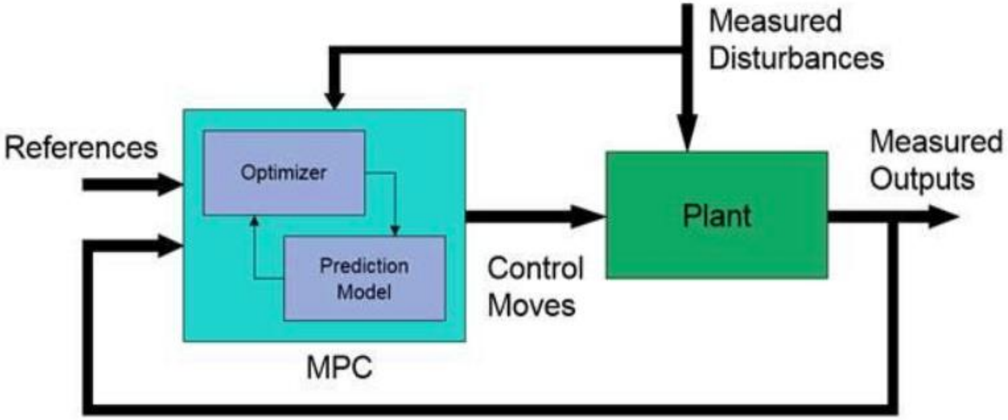
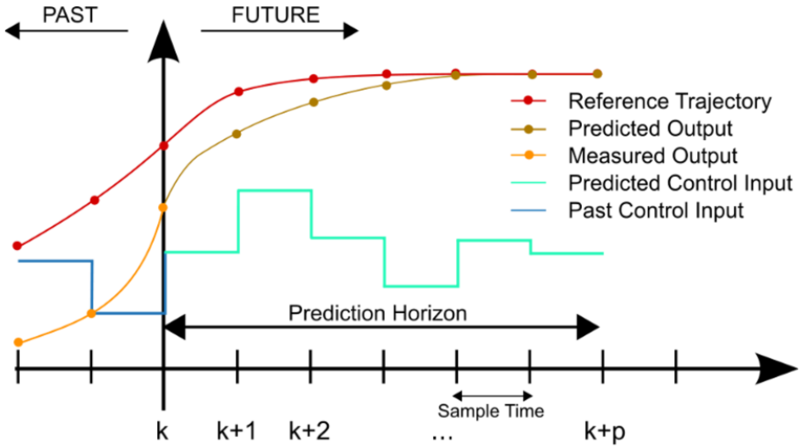




Model Predictive Control

MPC





	Intelligent Control System KOM 5101
1	Introduction to Intelligent Control Systems (knowledge-based vs data-driven systems)
2	Computational Thinking Tools
3	Dynamical Systems Modelling (Control System Toolbox could be used to transfer functions, state space models)
4	Model Predictive Control MPC (MPC Toolbox can be used)
5	Intro to Machine Learning (Stats & Machine Learning Toolbox could be used)
6	Data-driven Modeling -with machine learning (Stats & Machine Learning Toolbox could be used)
7	Data-driven Modeling -with system Identification (SysID toolbox could be used)
8	Midterm Exam
9	Data-driven Control Techniques -Extremum seeking (Simulink Control Design could be used)
10	Data-driven Control Techniques -Model reference adaptive control (Simulink Control Design could be used)
11	Intro to Deep Learning (Deep Learning Toolbox could be used)
12	Reinforcement Learning (RL Toolbox could be used)
13	Student's Projects
14	Student's Projects
15	Final Exam

MathWorks®

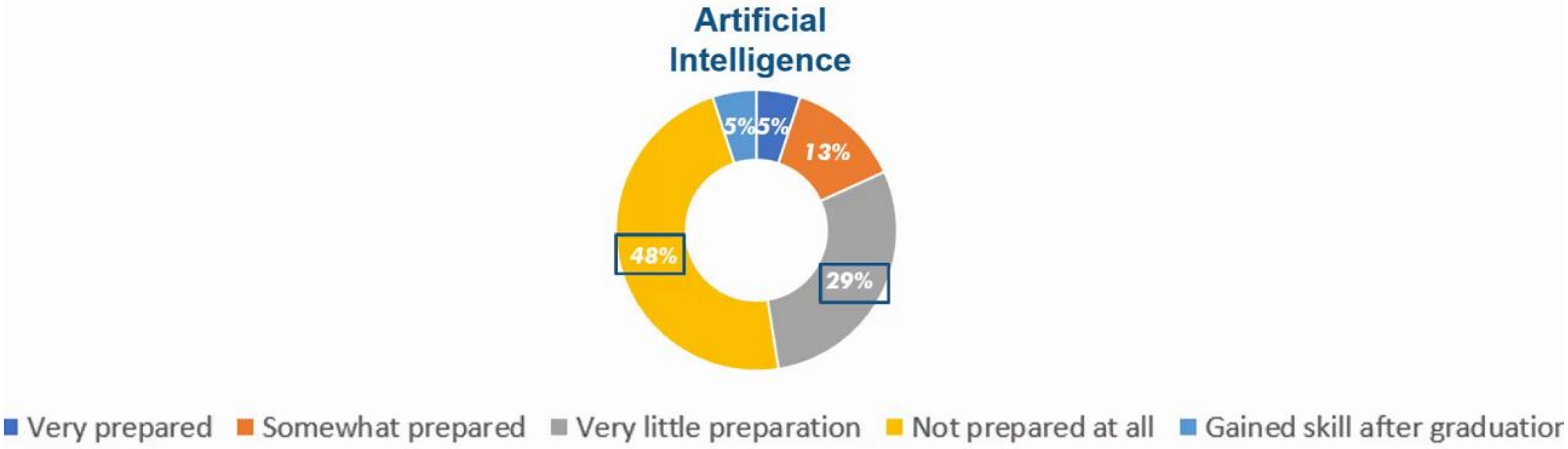
Dr. Julia Hoerner
Dr. Marco Rossi
Dr. Melda Ulusoy

MATLAB Drive

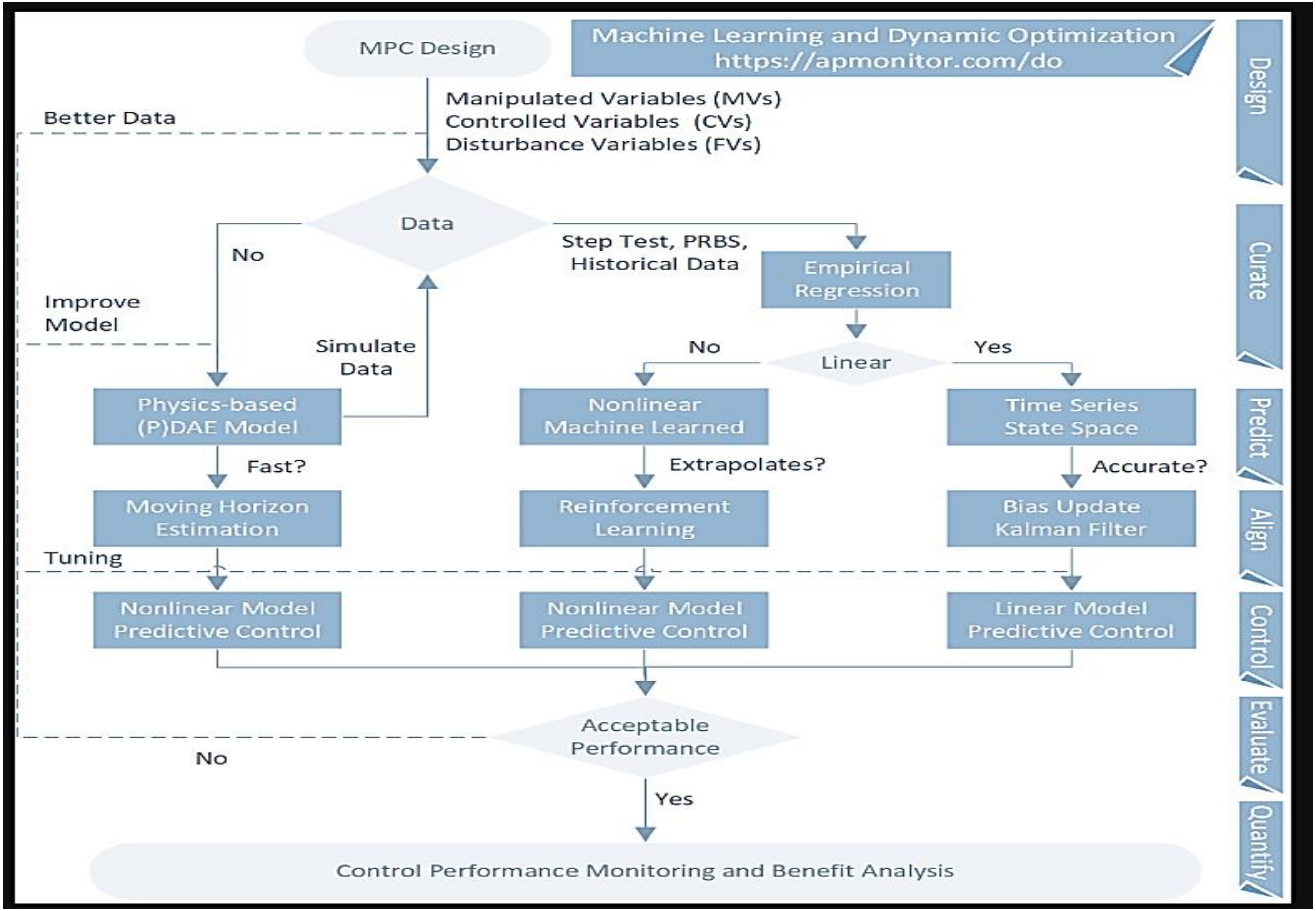
	Intelligent Control Systems KOM5101	Preparation + Homework	Matlab Drive
1	Introduction to intelligent control systems (knowledge-based vs data-driven systems)	Select the project from https://github.com/mathworks/MathWorks-Excellence-in-Innovation#mathworks-excellence-in-innovation-projects	https://drive.matlab.com/sharing/c1f9073b-a0b0-4966-95b0-c107691878da
2	Computational thinking tools	Work with the Virtual Hardware and Labs for Control. Solve the following Labs Lab4_PositionAnalysis.mlx Lab3_PositionControl.mlx Lab2_VehicleModel.mlx Lab1_CruiseControl.mlx	https://drive.matlab.com/sharing/77e65af2-6ffd-4709-a0bd-c36e0fbe50df
3	Dynamical systems modelling	1. Study and Obtain the state space model of a crane system. 2. Study and Obtain the state space model of the Lateral Vehicle Dynamics bicycle model with two degrees of freedom, lateral position and yaw angle.	https://drive.matlab.com/sharing/31e0ba39-b3f8-402c-a428-6a5b9d620081
4	Model Predictive Control MPC	Study and work with the MPC models explained. Use the MPC Toolbox of Matlab and the apmonitor server. Learn how to work with the drivingScenarioDesigner. Program the MPC algorithms using Simulink and Live scripts. Modify Models and MPC parameter and settings.	https://drive.matlab.com/sharing/398fa9fa-4650-4316-ab2b-0d228b24f48c

Technical Skills – Existing gaps

According to the Survey for Skills Gaps in Recent Engineering Graduates (ASEE, 2020):



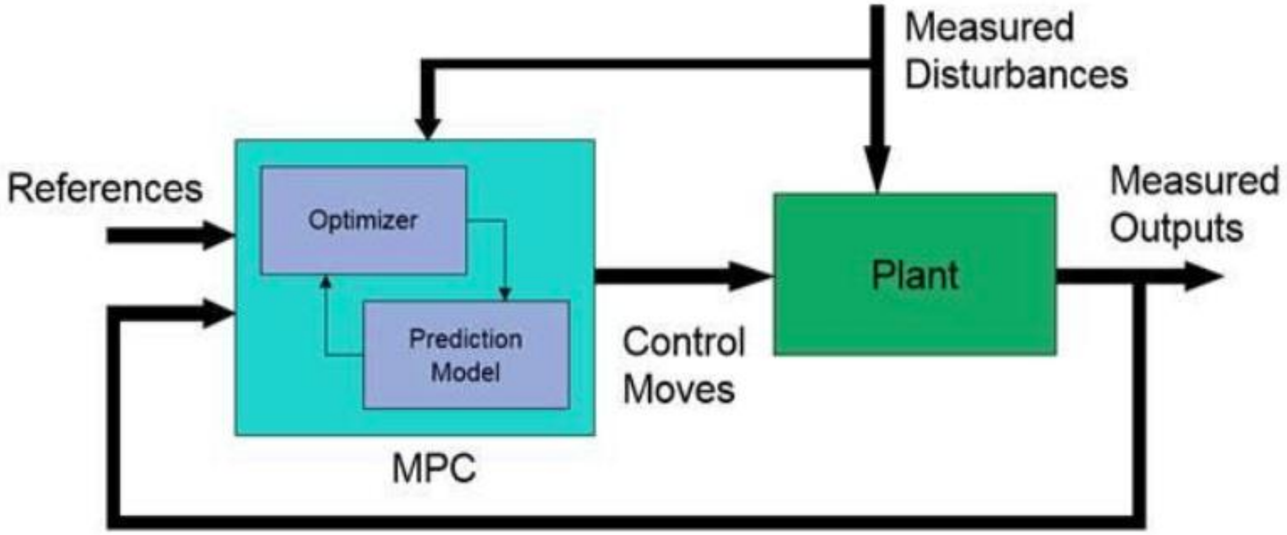

Before class
During class
After class

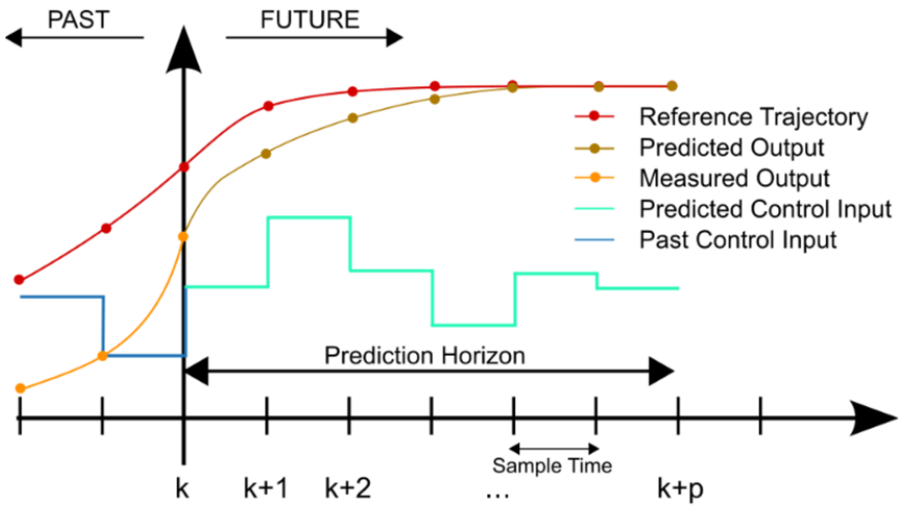


MPC Design
Machine Learning and Dynamic Optimization
https://apmonitor.com/do
Manipulated Variables (MVs)
Controlled Variables (CVs)
Disturbance Variables (FVs)
Better Data
Data
No
Step Test, PRBS,
Historical Data
Empirical Regression
Improve Model
Simulate Data
No
Linear
Yes
Physics-based (P)DAE Model
Nonlinear Machine Learned
Time Series State Space
Fast?
Extrapolates?
Accurate?
Moving Horizon Estimation
Reinforcement Learning
Bias Update Kalman Filter
Tuning
Nonlinear Model Predictive Control
Nonlinear Model Predictive Control
Linear Model Predictive Control
Acceptable Performance
No
Yes
Control Performance Monitoring and Benefit Analysis
Design
Curate
Predict
Align
Control
Evaluate
Quantify

What is Model Predictive Control?

MPC is an optimal control technique in which the calculated control actions minimize a cost function for a constrained dynamical system over a finite, receding, horizon. At each time step, an MPC controller receives or estimates the current state of the plant. It then calculates the sequence of control actions that minimizes the cost over the horizon by solving a constrained optimization problem that relies on an internal plant model and depends on the current system state. The controller then applies to the plant only the first computed control action, disregarding the following ones. In the following time step the process repeats.

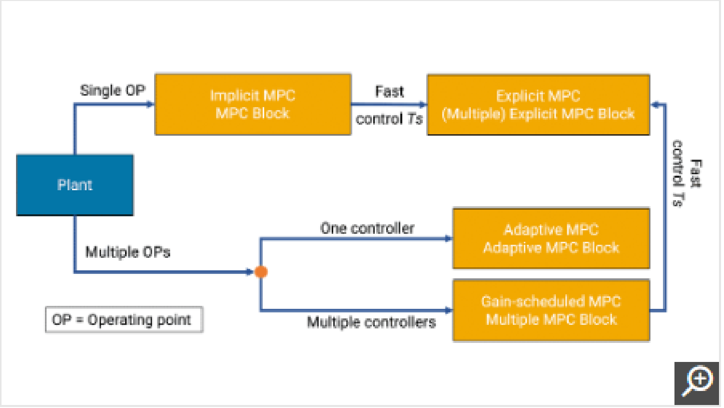




https://www.mathworks.com/videos/understanding-model-predictive-control-part-6-how-to-design-an-mpc-controller-with-simulink-and-model-predictive-control-toolbox-1535693503432.html

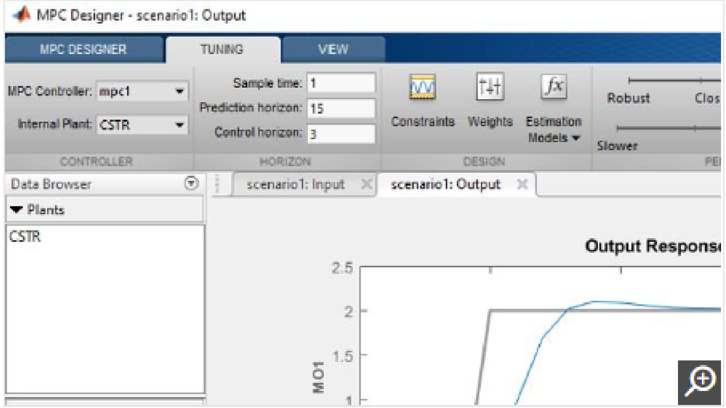
Model Predictive Control Toolbox: Design and simulate model predictive controllers

Model Predictive Control Toolbox™ provides functions, an app, Simulink® blocks, and reference examples for developing model predictive control (MPC). For linear problems, the toolbox supports the design of implicit, explicit, adaptive, and gain-scheduled MPC. For nonlinear problems, you can implement single- and multi-stage nonlinear MPC. The toolbox provides deployable optimization solvers and also enables you to use a custom solver.



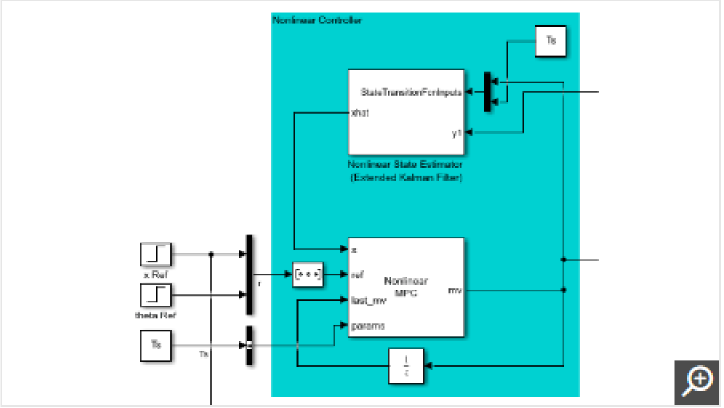
Linear MPC Design

Design implicit, gain-scheduled, and adaptive MPC controllers that solve a quadratic programming (QP) problem. Generate an explicit MPC controller from an implicit design. Use discrete control set MPC for mixed-integer QP problems.



MPC Designer App

Use the MPC Designer app to interactively design implicit MPC controllers, linearize your Simulink model with Simulink Control Design™, validate controller performance using simulation scenarios, and compare responses for multiple designs.



Nonlinear MPC Design

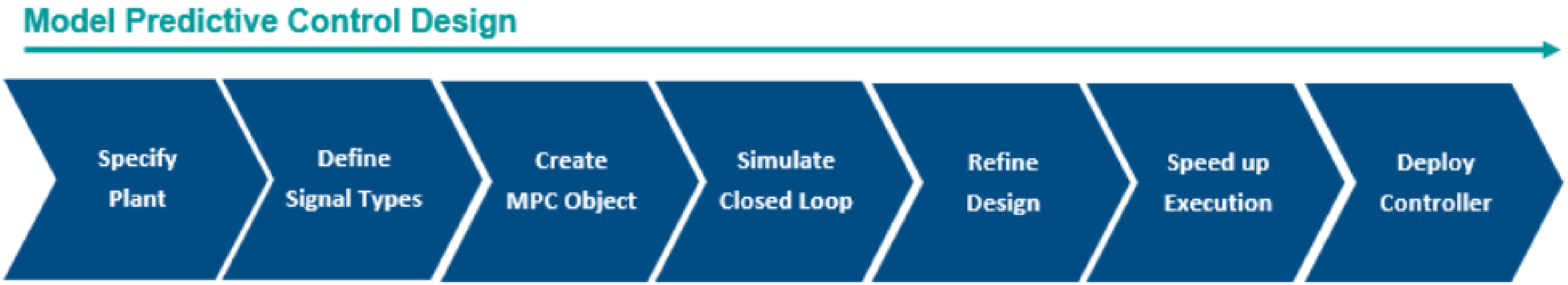
Design nonlinear and economic MPC controllers that use Optimization Toolbox™ to solve a nonlinear programming (NLP) problem. Use single- or multi-stage formulation for optimal planning and feedback control.

Model Predictive Control (MPC Toolbox will be used)

•Linear MPC design example using MPC Toolbox and Simulink: Will be used to teach students how to design linear MPCs using MPC Designer App.
Un example that uses an automated driving application will be used.

Linear MPC Overhead-crane with a State Space Model: We will design and program a model predictive controller for an overhead crane with a pendulum mass. The system has to meet specific control objectives by tuning the controller and using the state space model of the crane system.
Un example of a simulation and optimization will be accomplished for the pendulum system.

MPC Design Workflow



Plant: Construct Linear Time Invariant Models, Specify Multi-Input Multi-Output Plants, Linearize Simulink Models, Linearize Simulink Models Using MPC Designer, and Identify Plant from Data.

Signal Types: Identify the system variables and whether each plant output is measured or unmeasured, and whether each plant input is a manipulated variable (that is, a control input) or a measured or unmeasured disturbance.

Create and object: specify, in the object, controller parameters such as the sample time, prediction and control horizons, cost function weights, constraints, and disturbance models

Simulation: evaluate the performance of your controller by simulating it in closed loop with your plant

Refine design, speed up execution and deploy

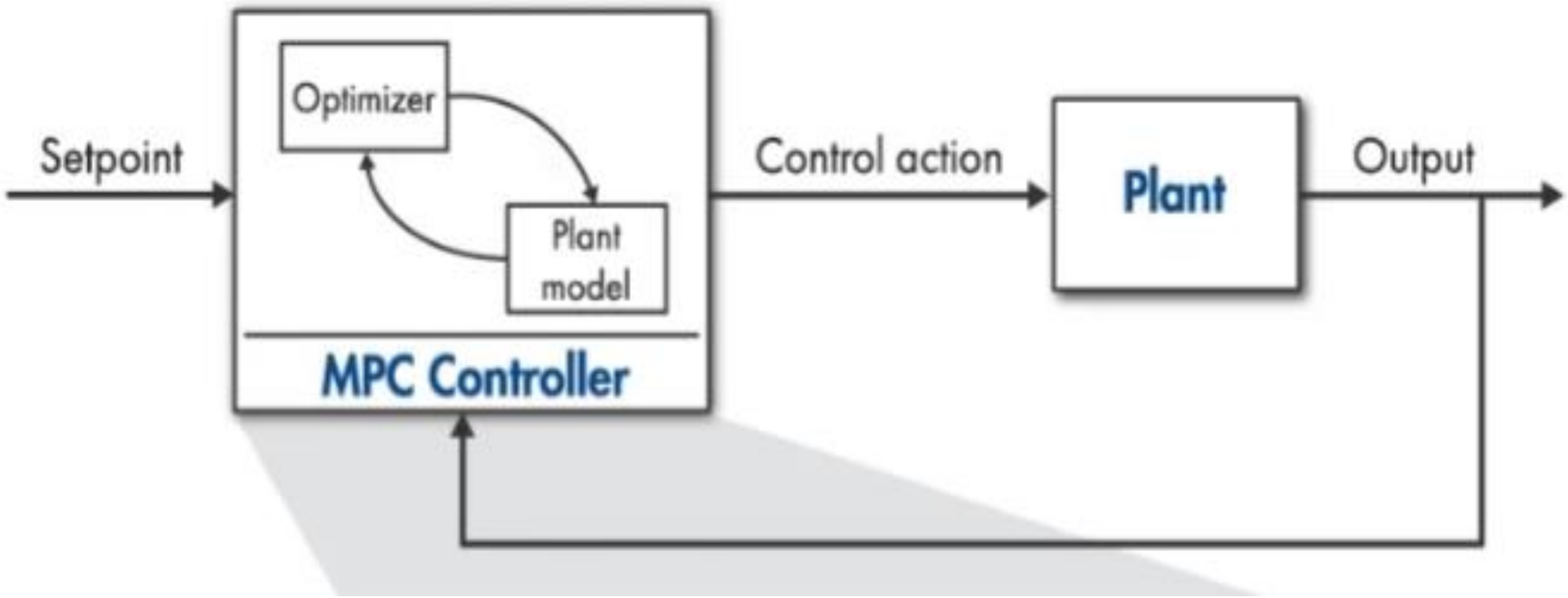
Model predictive control (MPC) is an advanced control technique that has been used for process control since the 1980s. With the increasing computing power of microprocessors, the use of MPC has spread to real-time embedded applications, often used in the automotive and aerospace industries.

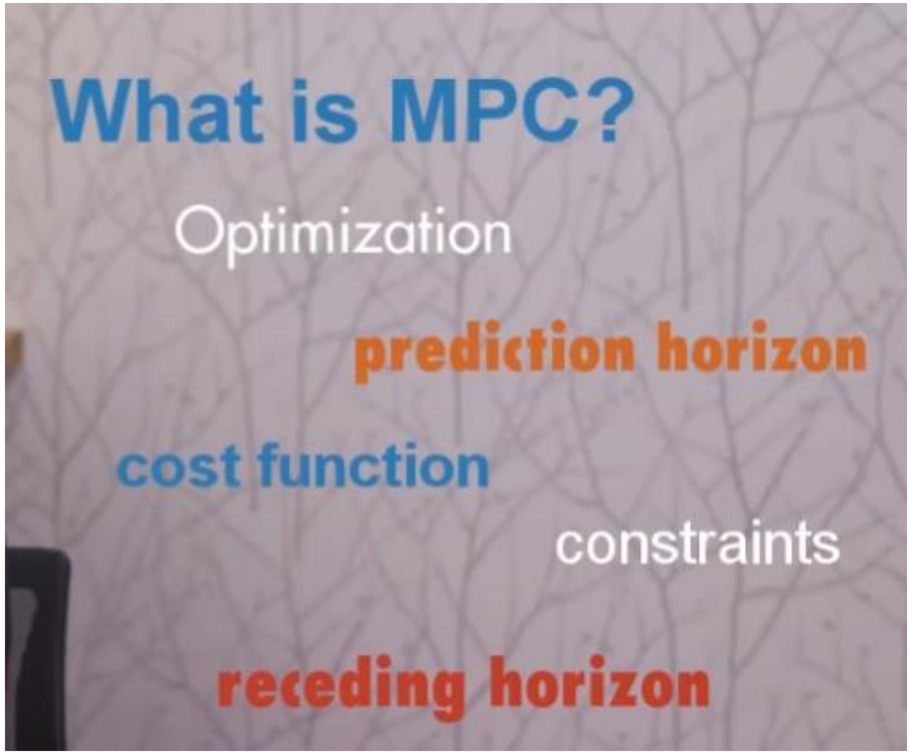
MPC can optimize multiple objectives, including economics, controls, and safety.

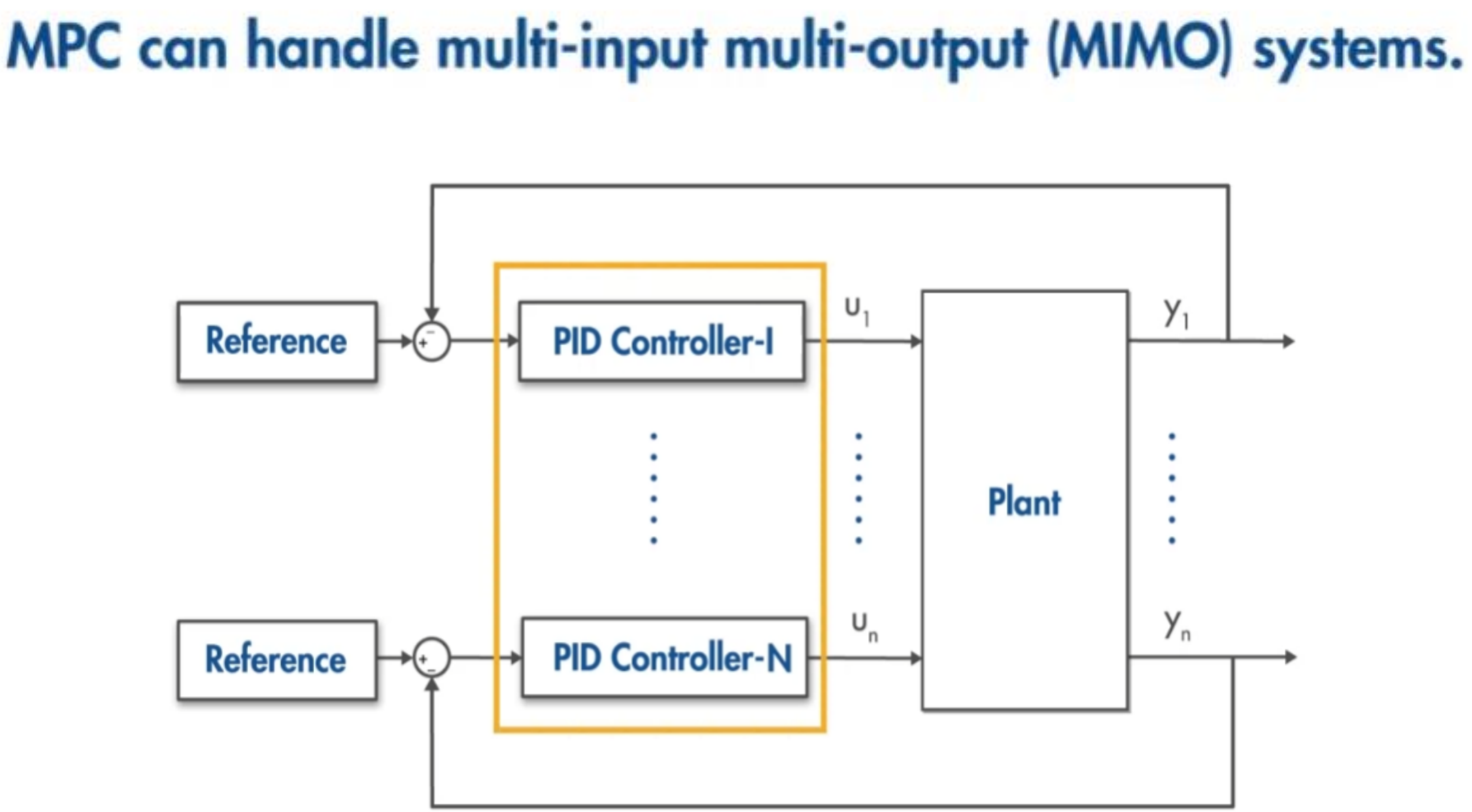
Automative
Aerospace
Energy
Food Processing
Industrial Manufacturing
Metallurgy and Mining
Robotics

Model Predictive Control MPC

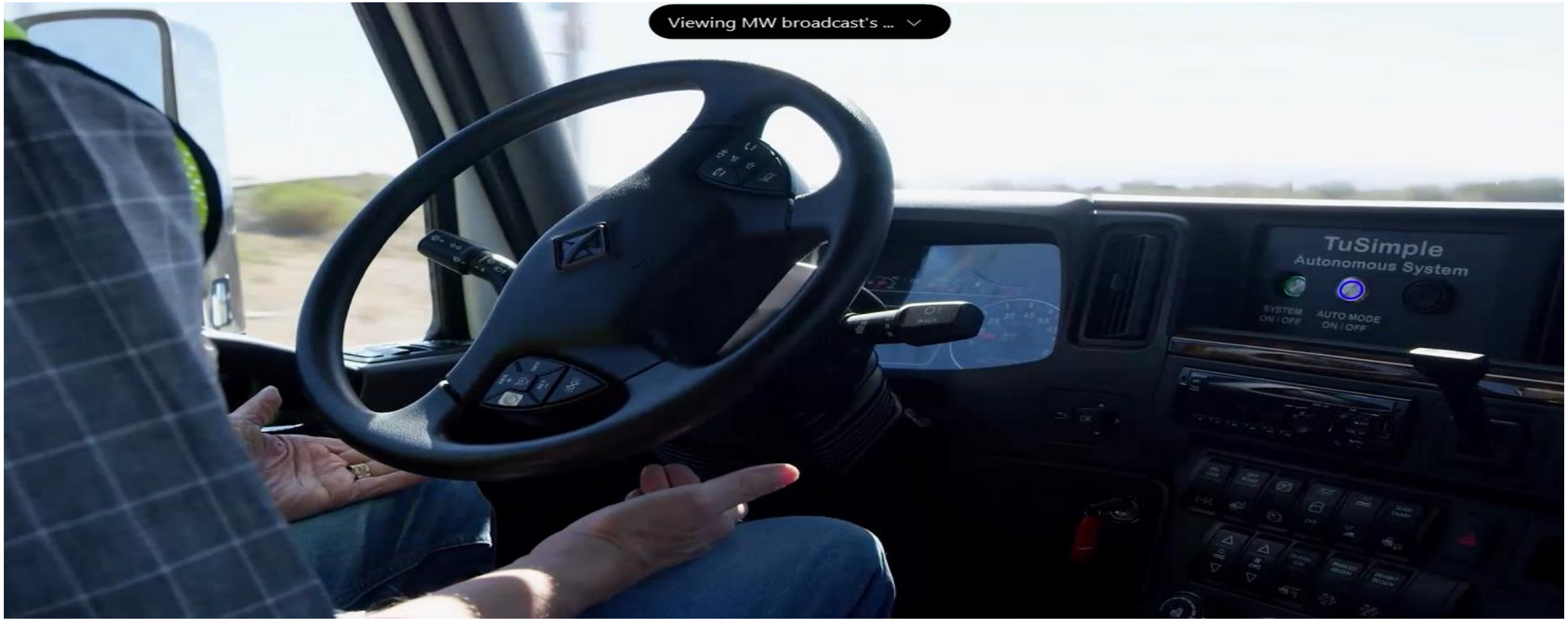
MPC uses the model of a system to predict its future behavior, and it solves an optimization problem to select the best control action.





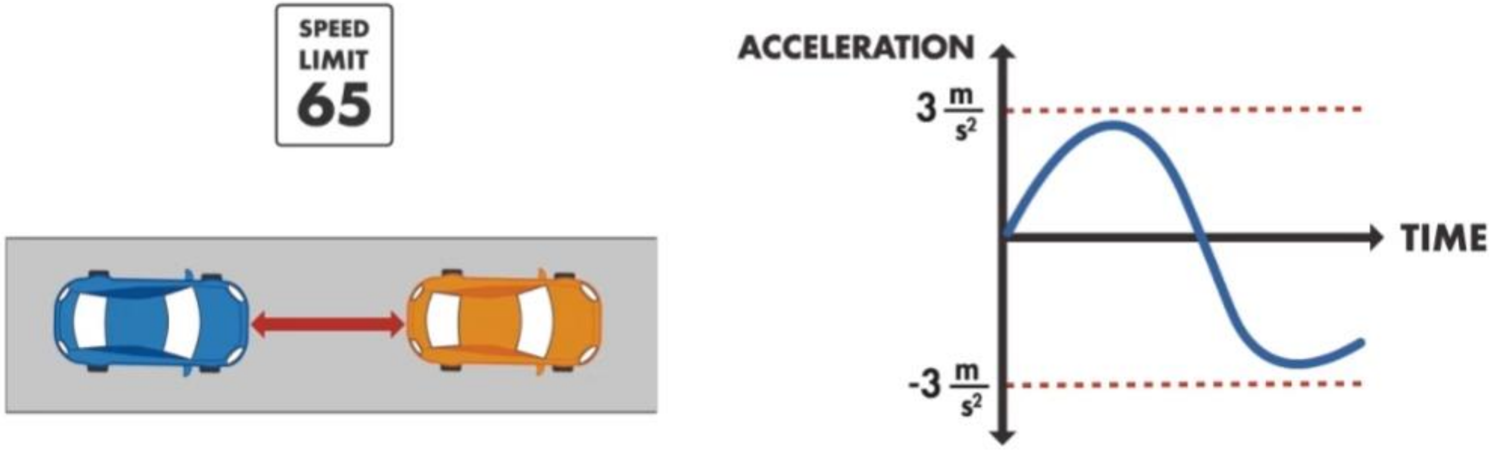


MPC can handle multi-input multi-output (MIMO) systems with coupled input-output channels, which simplifies the architecture of the control loop. Designing a controller for such a system with PID controllers, for example, would be challenging since the control loops and associated responses would be intertwined.



Automated driving application - Lateral Vehicle Dynamics

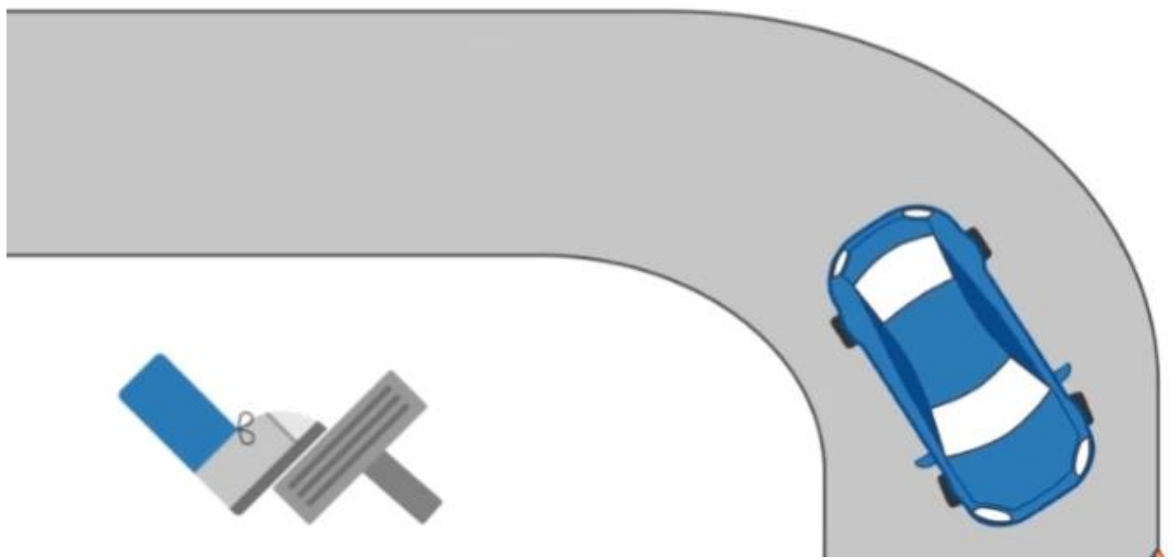
MPC can handle constraints.



MPC can handle constraints on the inputs, outputs, and states, which is important since real-world systems have physical limits that need to be respected. Other MIMO techniques such as linear quadratic regulators (LQRs) cannot handle constraints explicitly.

MPC has preview capability.

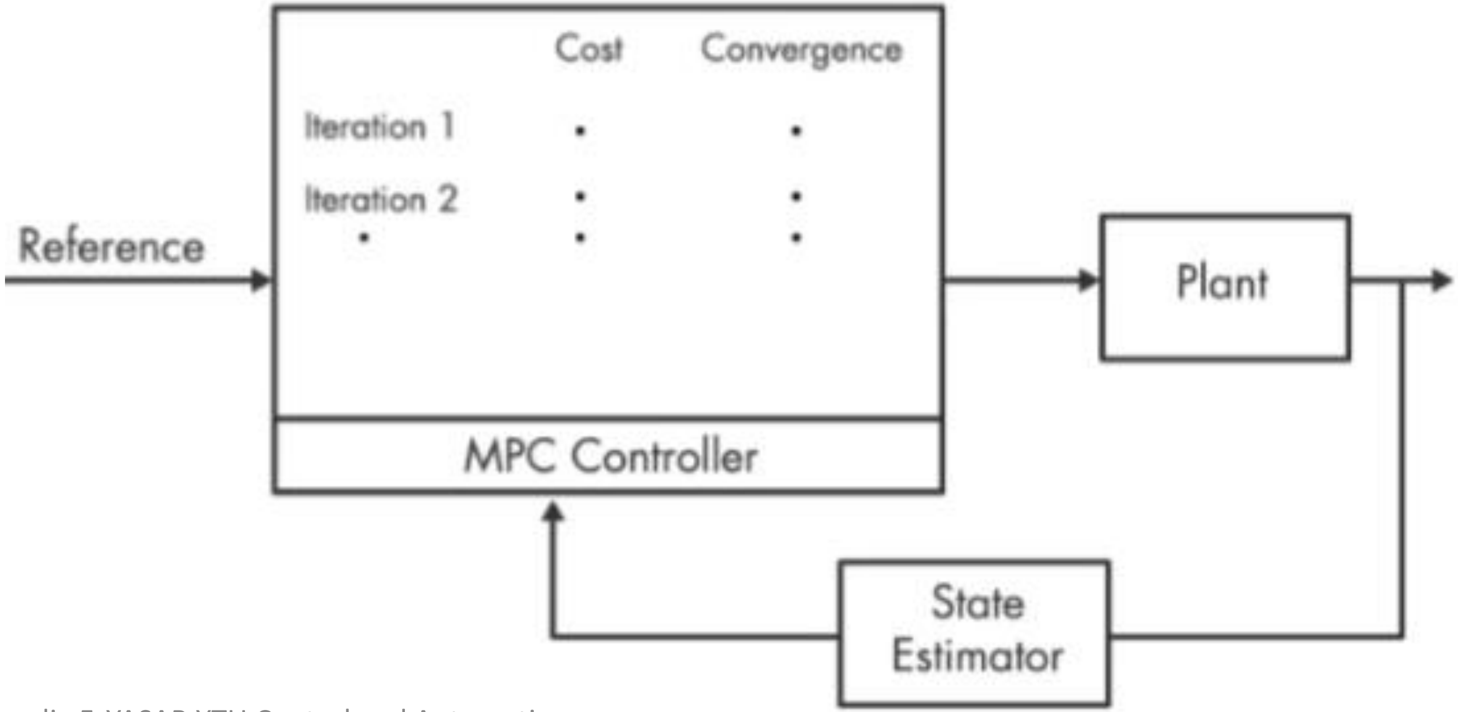
Includes reference information about the future trajectory.



The control action calculated with MPC take into account what will (likely) happen several steps into the future to improve performance. This is possible because the MPC uses an internal prediction model of the system that is controlled.

If MPC has all these advantages, do we still need traditional methods like PIDs?

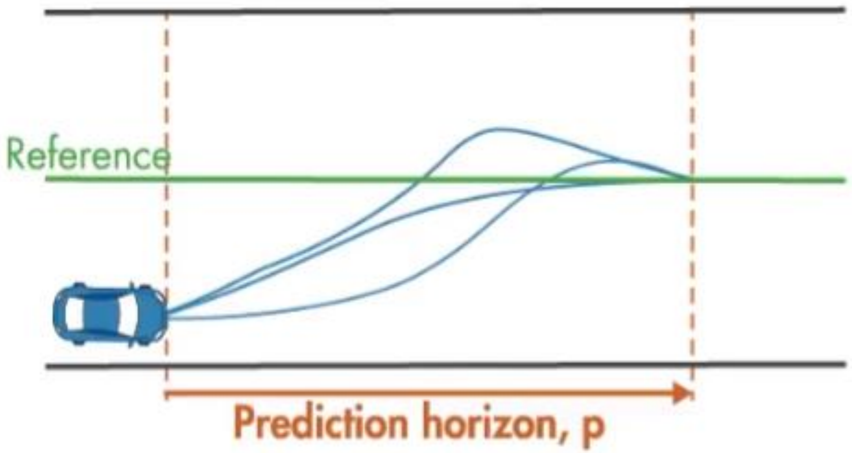
Unlike PIDs, MPC is more challenging to apply in high-bandwidth feedback control applications— that is, applications where the response time of the control loop must be short. This is because, for feedback control, MPC solves an optimization problem online, and that requires significant computing power and memory.

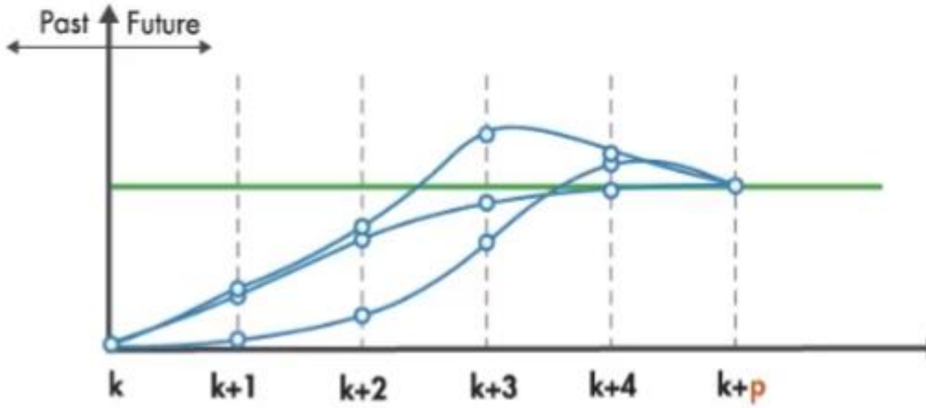


System's model

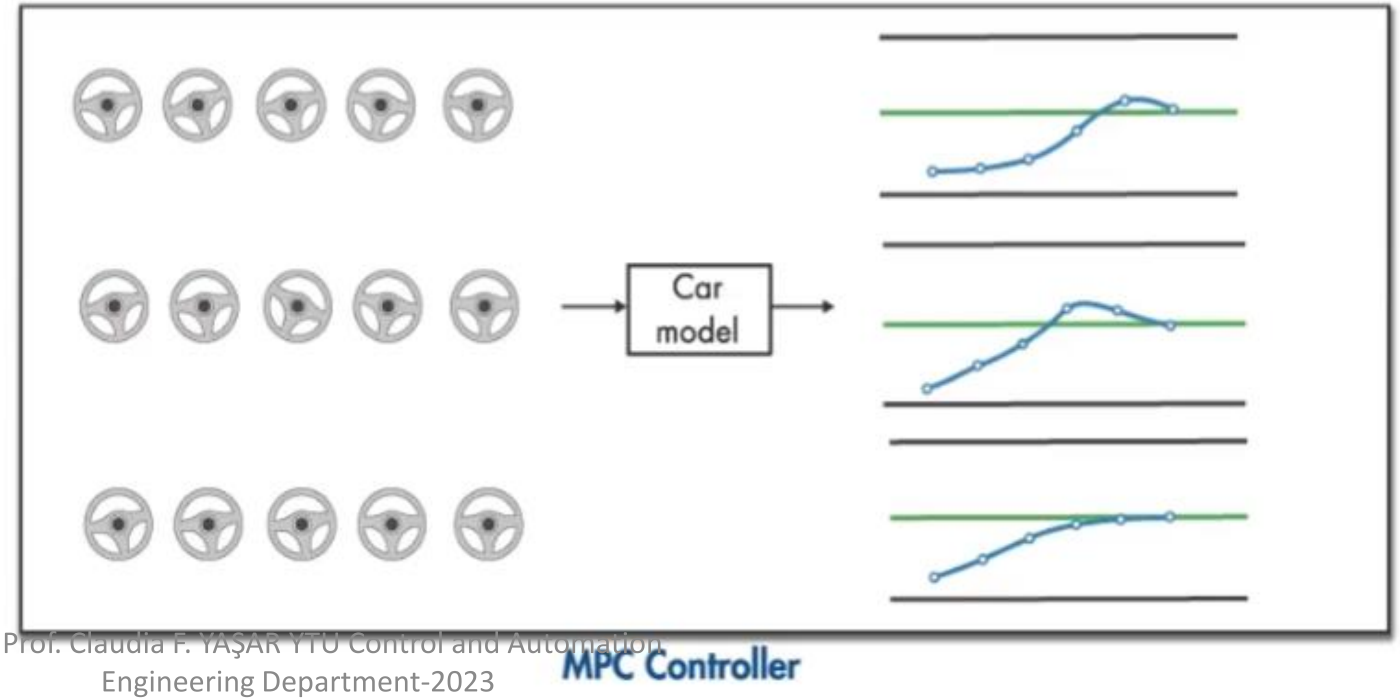
Simulate the K steps

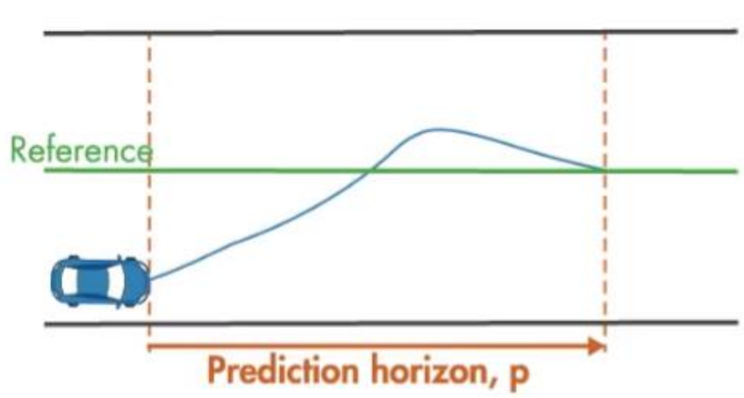
How far PMC looks to the future (Future time steps)

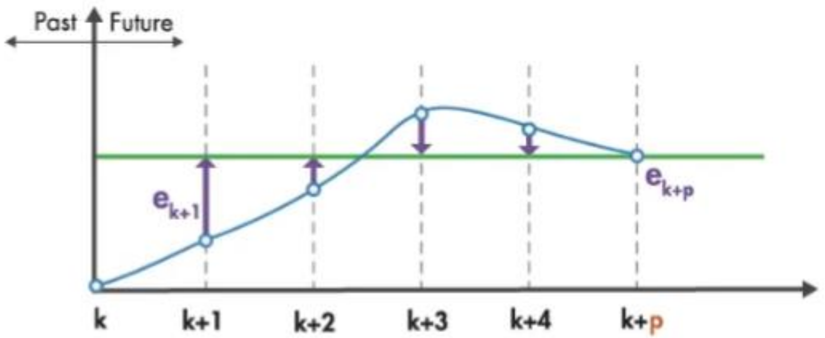




The best predicted path closest to the reference.

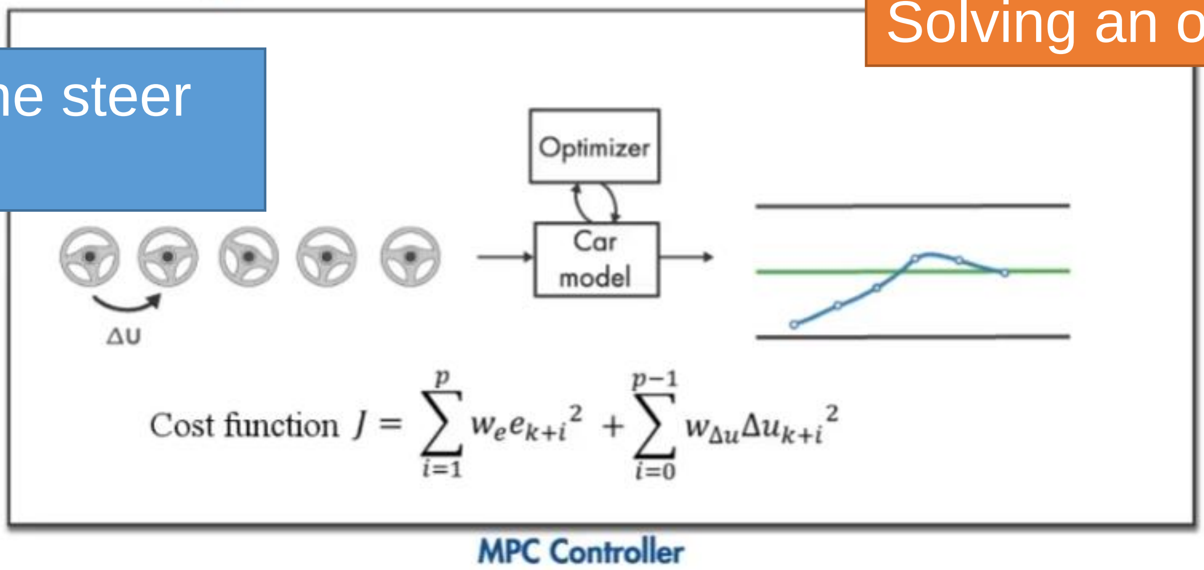






Solving an online optimization problem

Tries to minimize the steer wheel changes

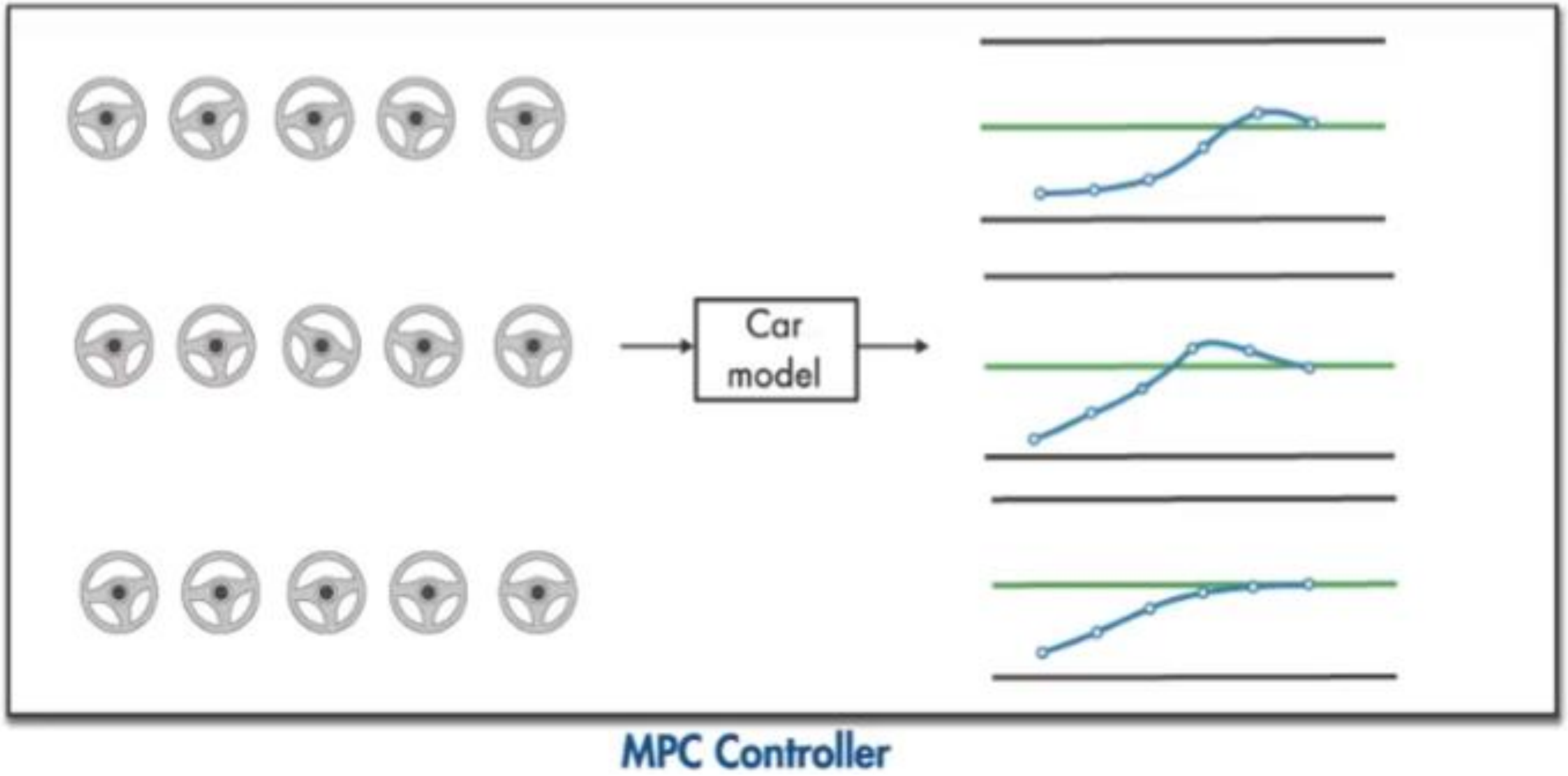


Minimizing a cost function MPC makes sure steer wheel angle and car's position under the principles and limits

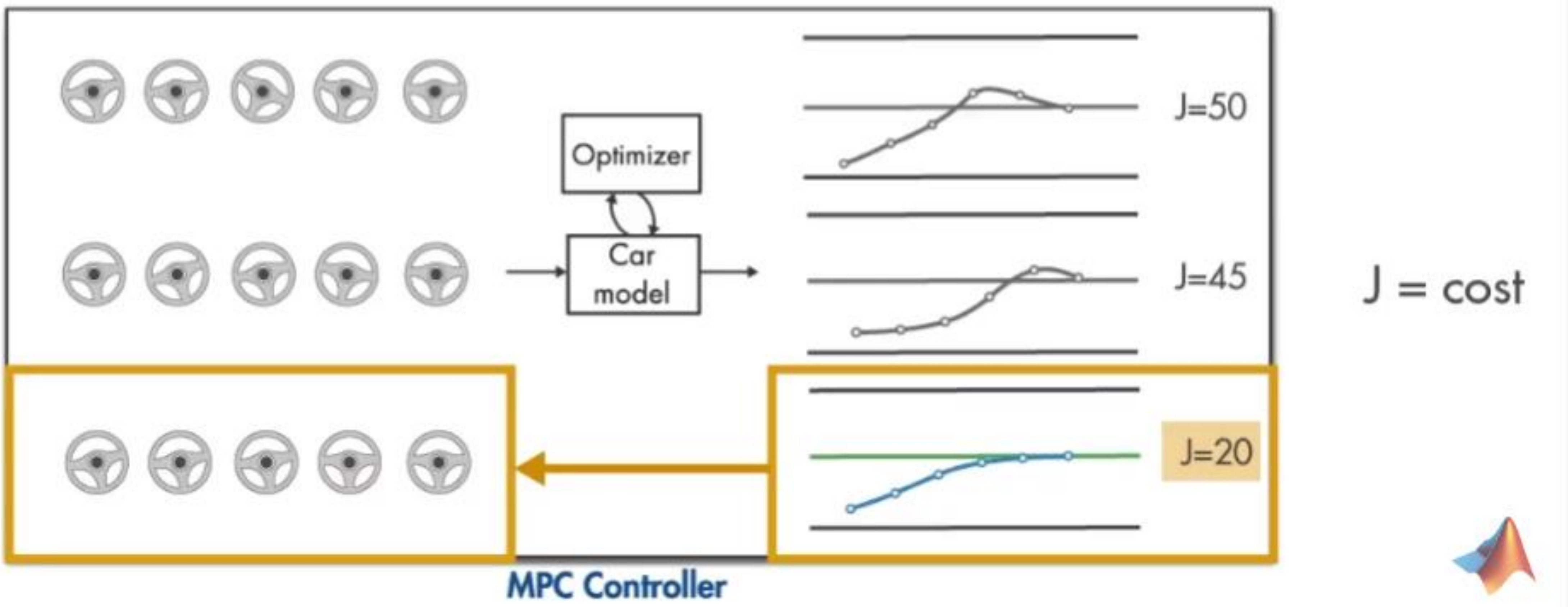
Weighted squared of some of the predicted errors

Steer wheel angle increments

Constrains: car position and how much the steer wheel can change

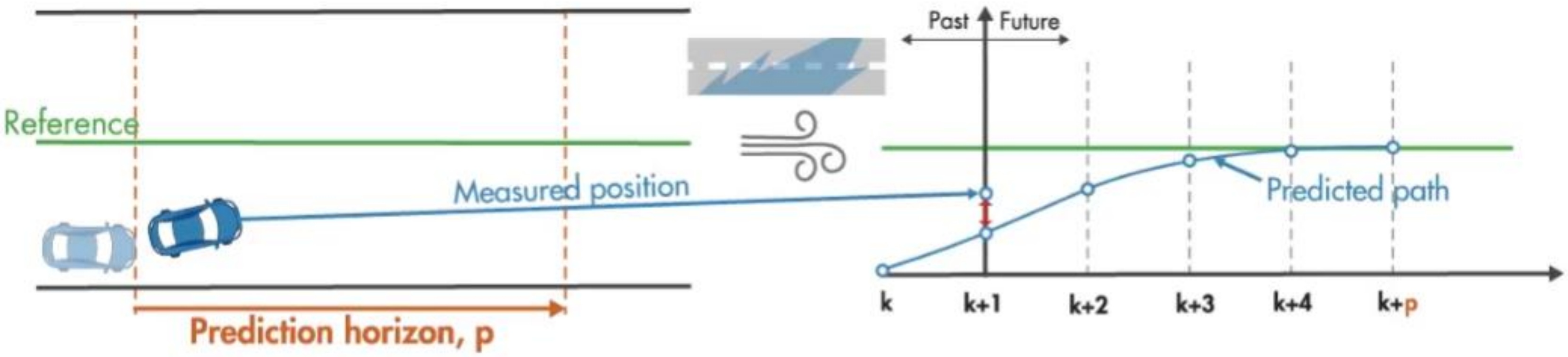


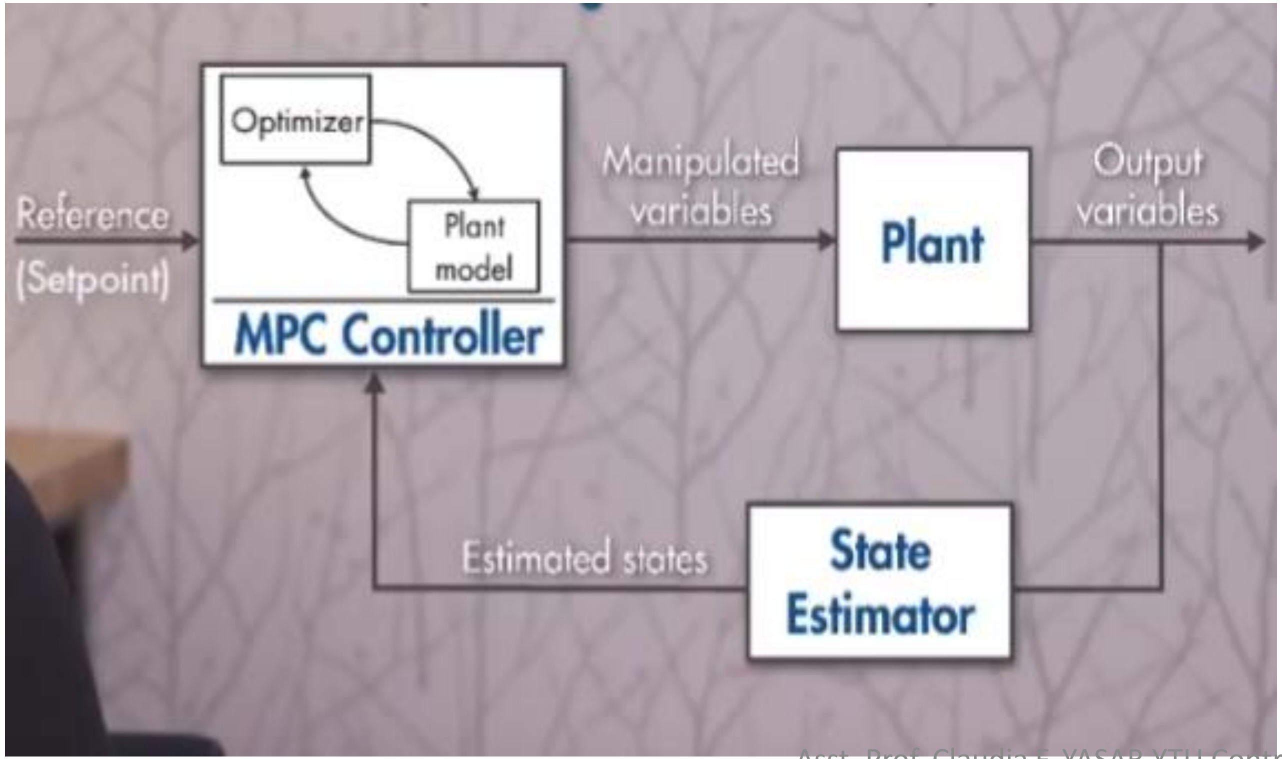
The predicted path with the smallest J is the optimal solution. Optimal steer wheel angle sequence that keeps the car close to the reference



The predicted path with the smallest J is the optimal solution. Optimal steer wheel angle sequence that keeps the car close to the reference

Measurement of the car's lateral position that can be different to what the MPC has predicted due to measurement disturbance





The measurement may be measured or can also be estimated

Kalman Filter

https://www.mathworks.com/matlabcentral/fileexchange/105525-kalman-filter-virtual-lab?s_tid=ta_fx_results

Recommendations for choosing the controller sample time, prediction and control horizons, constraints and weights.

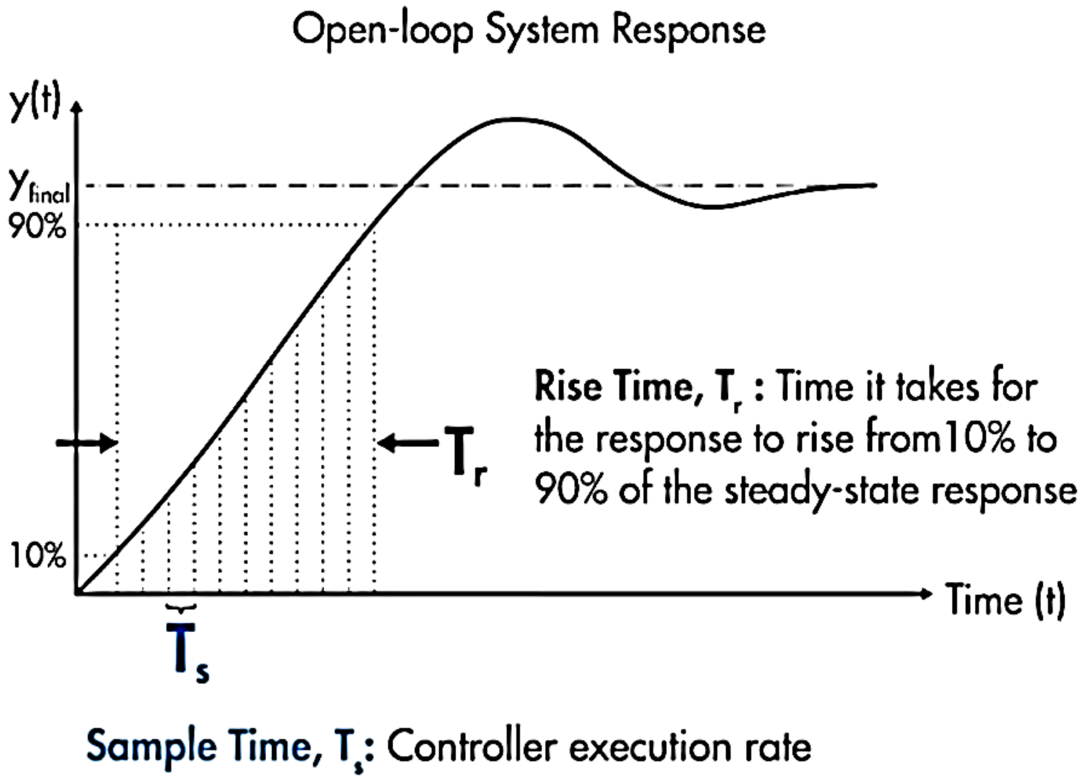
Sample Time

The sample time is a key concept in model predictive control and can be separated into two parts: the prediction sample time and the control sample time. When designing an MPC problem, the prediction sample time and control sample time are often set to be equal or even treated as one parameter, but it is important to distinguish between the two and how they affect performance.

Sample Time

The prediction *Ts* is the sample time of the internal prediction model. It defines how long each prediction step lasts, and the value of all MVs remains constant during each prediction step

Ts is determined by plant dynamics and control response time.



$$\frac{T_r}{20} \leqslant T_s \leqslant \frac{T_r}{10}$$

If prediction *Ts* is slow, you may not have enough control bandwidth to stabilize an open-loop unstable plant. On the other hand, faster prediction *Ts* will require a longer prediction horizon to keep the prediction time constant which makes the optimization problem larger (higher memory footprint) and more complex to solve.

Control Ts

The control *Ts* determines the sample time of the MPC controller. It defines how often the MPC optimization problem is solved at run time.
The control sample time is typically equal to the prediction sample time, but it can also be set to be faster (but not slower).

Faster control *Ts* generally improves performance (i.e., bandwidth) and robustness (i.e., gain and phase margins) to some extent. Also, as control *Ts* gets smaller, rejection of unknown disturbances, including discrepancies between internal MPC model and actual plant, usually improves and then plateaus. Qualitatively, this makes sense as the controller is able to respond faster to changes in the environment. The control sample time value at which performance typically depends on the plant dynamic characteristics. For example, processes with slow dynamics will not benefit much from small control sample times, unlike real-time control applications such as motor control.

Prediction Horizon

The *prediction horizon*, *p*, is the number of future control intervals the MPC controller must "plan" for (using the internal plant model for prediction) when optimizing its MVs. The duration of each control interval is determined by the prediction sample time. The choice of prediction horizon depends on the characteristics of the plant dynamics. The main guideline on how to select *p* is in fact to satisfy the prediction time (*p* * prediction *Ts*) requirements for the system of interest. Typically, systems with slower dynamics require longer prediction times such that the MPC controller can sufficiently predict how the manipulated variables may affect the cost/outputs of interest. Thus, the values of prediction horizon *p* and prediction *Ts* are, in a sense, intertwined.

Control Horizon

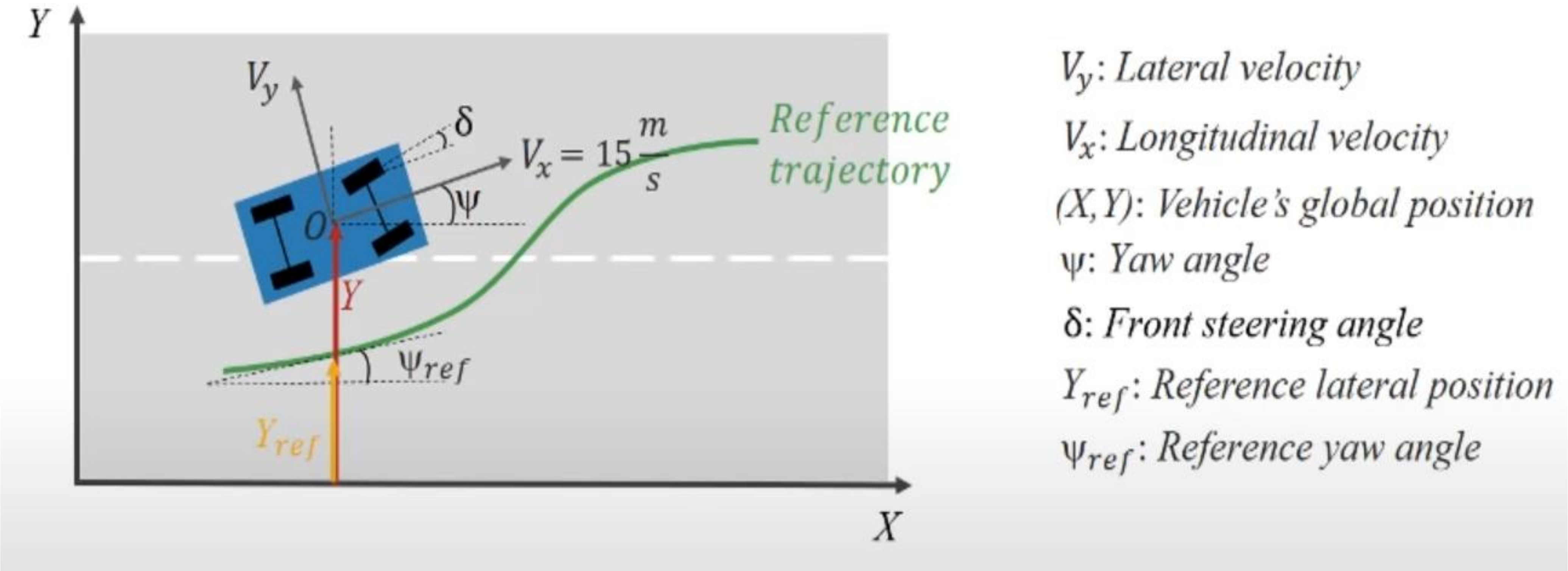
The control horizon, m, is the number of MV moves to be optimized at control interval k and takes values between 1 and the prediction horizon p (Model Predictive Control Toolbox uses a default control horizon value of $m = 2$). MV moves determine the values of the MPC solution (open-loop control) at each step of the control horizon, for each manipulated variable specified in the problem. As a result, the number of variables that need to be optimized by the solver grow with the number of control inputs and the control horizon value.

Constraints

MPC solves a constrained optimization problem at each time step. Typically, the larger the number of constraints, the longer it takes to solve the problem as complexity grows. Constraints limit the space of admissible controls, increasing the risk of running into infeasible problems—problems that are impossible to satisfy. Keeping the number of constraints to the absolutely necessary makes the optimization problem easier to solve.

Example: Lateral Vehicle Dynamics

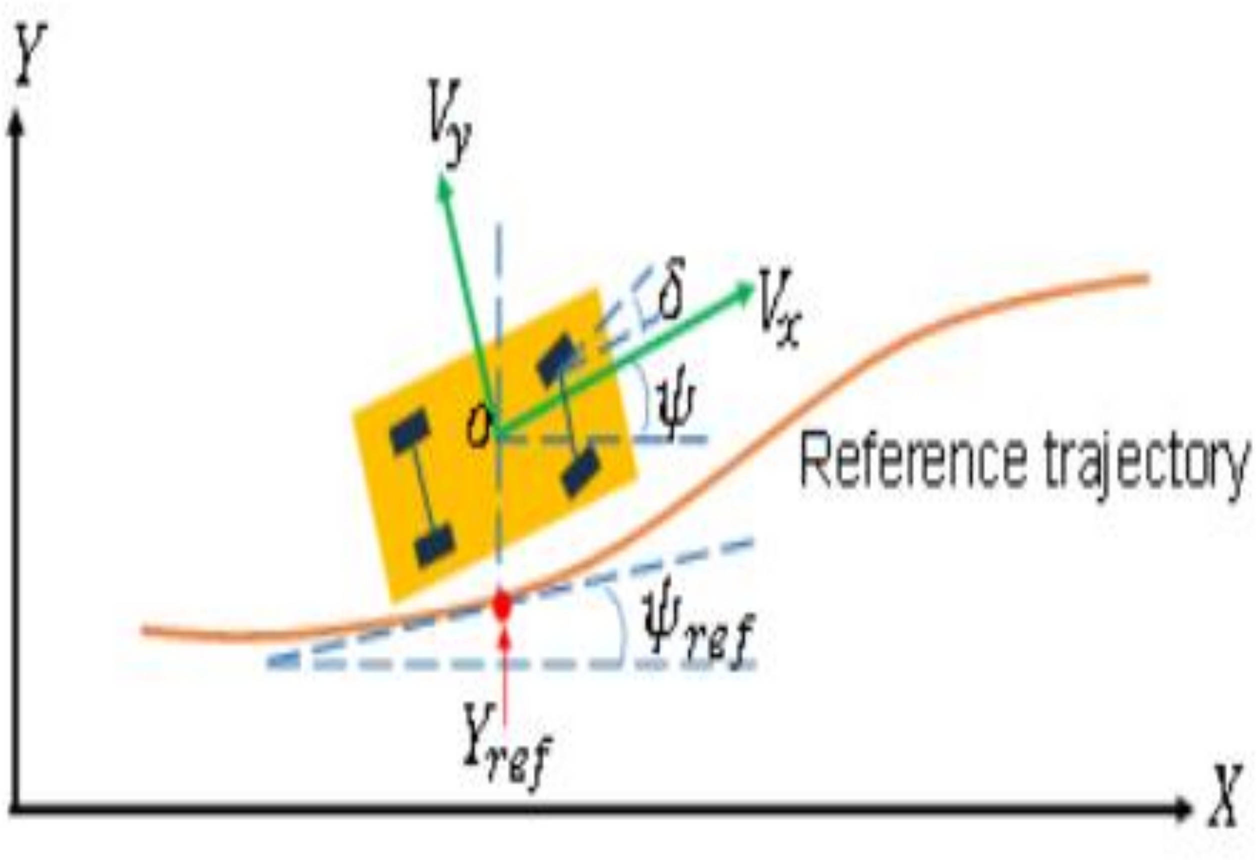
MPC controller's strategy for finding the **optimal steering wheel angle to control the car's longitudinal speed**. At each time step, the MPC controller makes predictions about the future lateral positions of the car.



https://www.mathworks.com/matlabcentral/fileexchange/68992-designing-an-mpc-controller-with-simulink
https://www.youtube.com/watch?v=8U0xiOkDcmw&list=PLn8PRpmsu08ozoeoXgxPSBKLyd4YEHww8

To describe the lateral vehicle dynamics, this example uses a ***bicycle model* with two degrees of freedom, lateral position and yaw angle.**

- m is the total vehicle mass (kg)
- Iz is the yaw moment of inertia of the vehicle (mNs^2).
- lf is the longitudinal distance from the center of gravity to the front tires (m).
- lr is the longitudinal distance from center of gravity to the rear tires (m).
- Cf is the cornering stiffness of the front tires (N/rad).
- Cr is the cornering stiffness of the rear tires (N/rad).



To describe the lateral vehicle dynamics, this example uses a ***bicycle model* with two degrees of freedom, lateral position and yaw angle.**

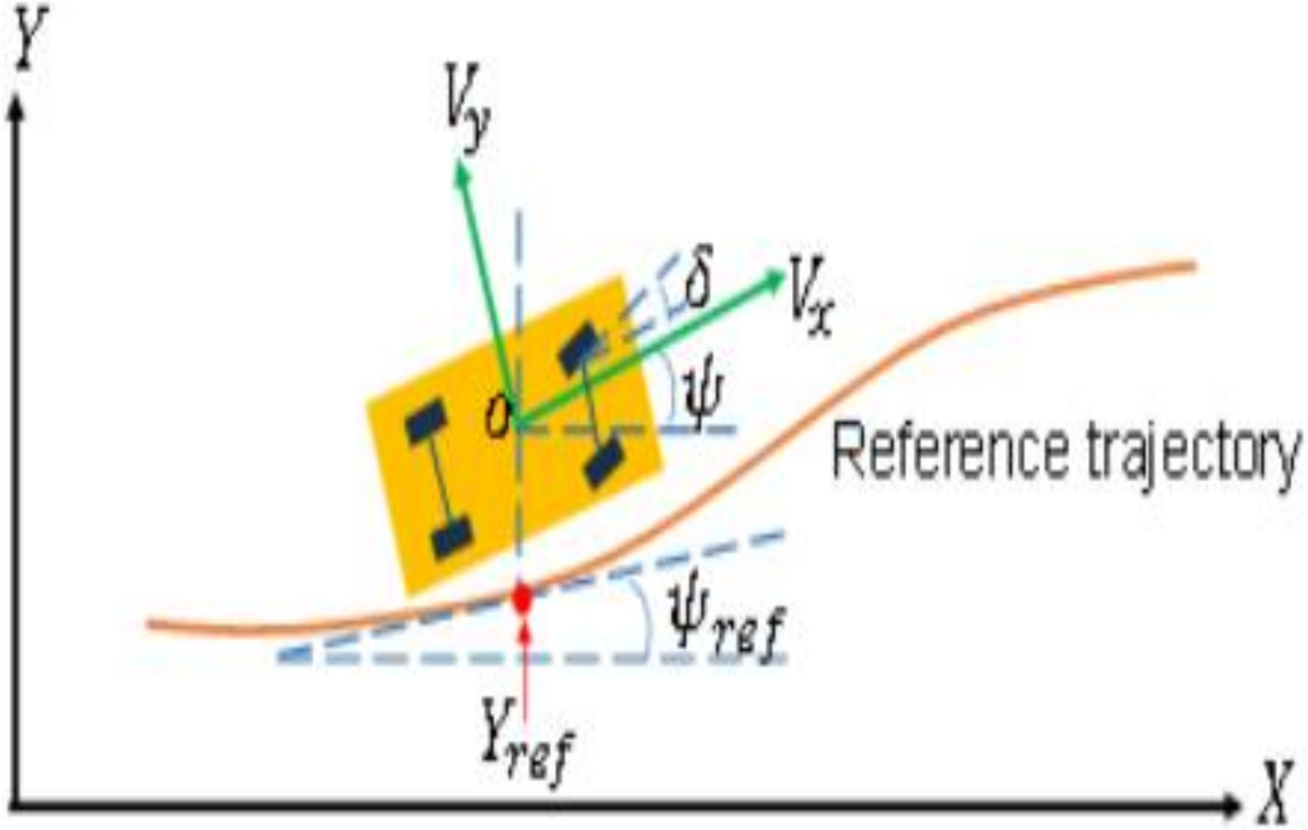
- **State variables:** Lateral velocity $V_y = \dot{Y}$, yaw angle ψ, yaw angle rate $\dot{\psi}$, Lateral position Y
- **Input variable:** Front steering angle δ
- **Output variable:** Lateral position Y and yaw angle ψ

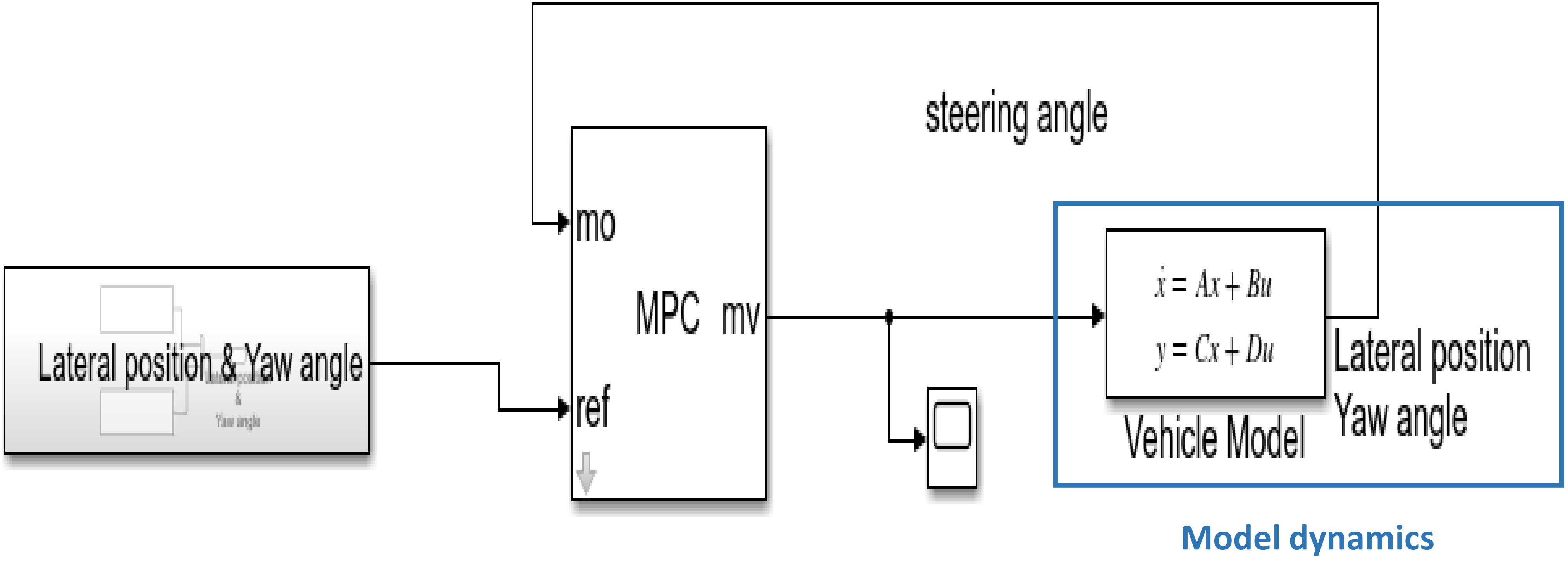
- Global Y position:

$$\dot{Y} = V_x \psi + V_y$$

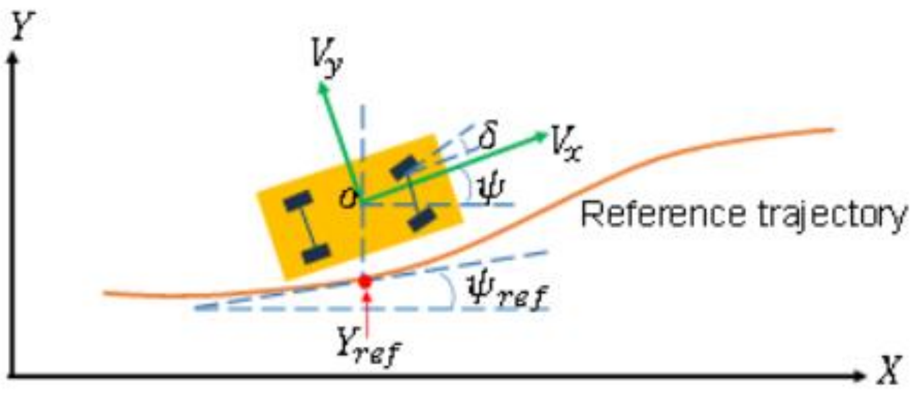
ψ small

$\sin\psi \approx \psi$




steering angle
mo
MPC mv
ref
Lateral position & Yaw angle
$\dot{x} = Ax + Bu$
$y = Cx + Du$
Vehicle Model
Lateral position
Yaw angle
Model dynamics

***Bicycle model* with two degrees of freedom, lateral position and yaw angle.** The vehicle model is depicted in the following figure.



$$\frac{d}{dt}\begin{bmatrix}\dot{y}\\ \psi\\ \dot{\psi}\\ Y\end{bmatrix} = \begin{bmatrix} -\frac{2C_f+2C_r}{mV_x} & 0 & -V_x-\frac{2C_fl_f-2C_rl_r}{mV_x} & 0\\ 0 & 0 & 1 & 0\\ -\frac{2C_fl_f-2C_rl_r}{I_zV_x} & 0 & -\frac{2C_fl_f^2+2C_rl_r^2}{I_zV_x} & 0\\ 1 & V_x & 0 & 0\end{bmatrix}\begin{bmatrix}\dot{y}\\ \psi\\ \dot{\psi}\\ Y\end{bmatrix} + \begin{bmatrix}\frac{2C_f}{m}\\ 0\\ \frac{2C_fl_f}{I_z}\\ 0\end{bmatrix}\delta$$

$\dot{x}$ A x B u

$$\dot{x} = Ax + Bu$$

$$y = Cx + Du$$

$$y = \begin{bmatrix}0 & 0 & 0 & 1\\ 0 & 1 & 0 & 0\end{bmatrix}\begin{bmatrix}\dot{y}\\ \psi\\ \dot{\psi}\\ Y\end{bmatrix} + \begin{bmatrix}0\\ 0\end{bmatrix}\delta$$

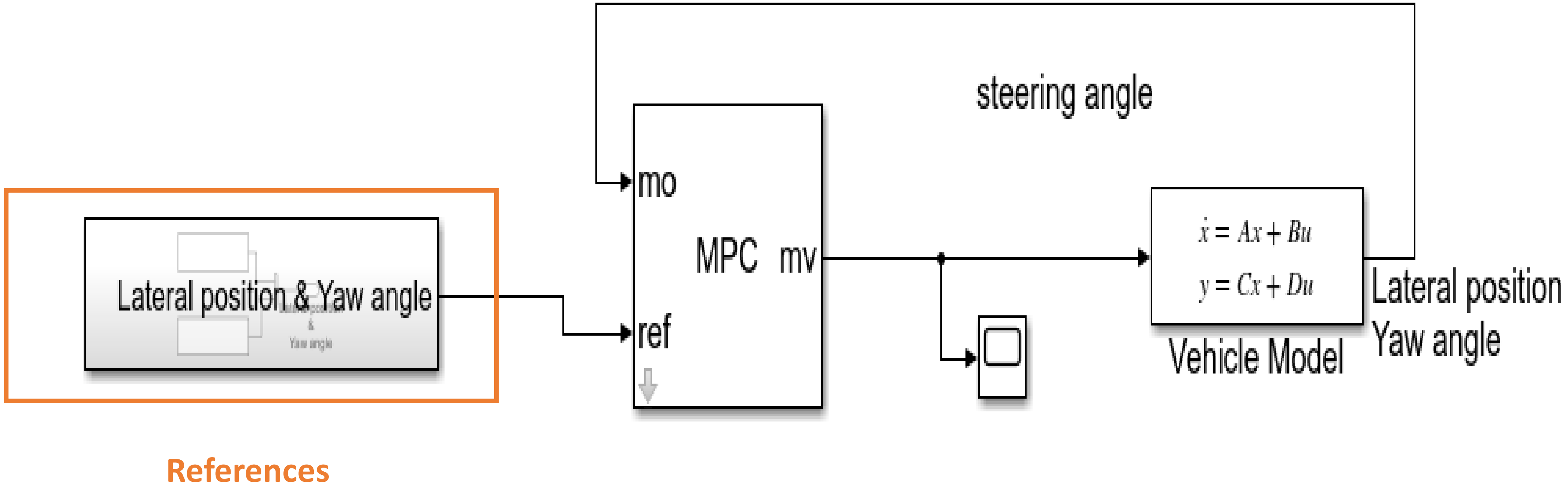
y C x D u

Input to the model: front steering angle δ

Output of the model: Lateral position Y
Yaw angle ψ

m = 1575; Iz = 2875; lf = 1.2;
lr = 1.6; Cf = 19000; Cr = 33000;
Vx = 15;

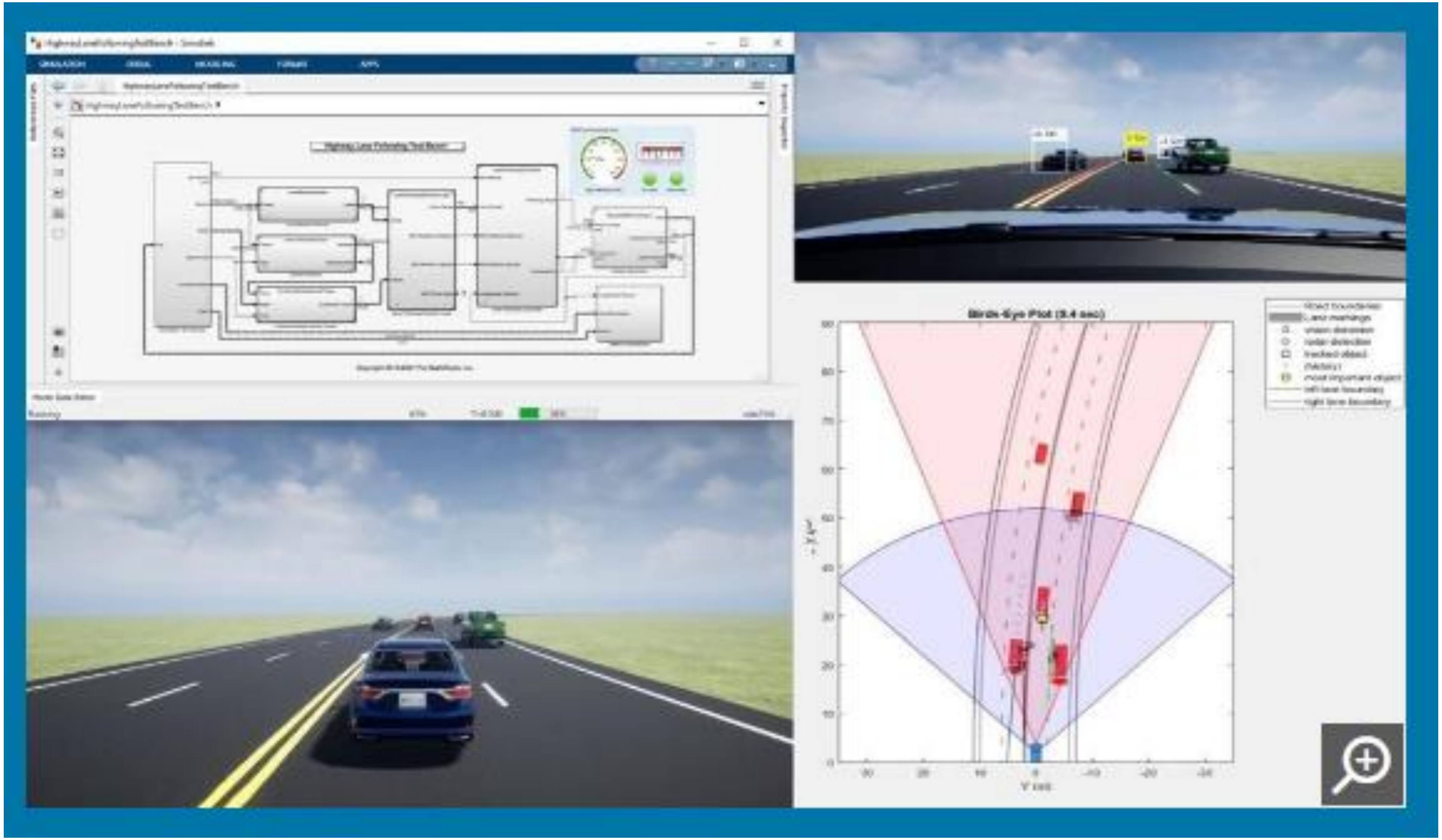
https://www.mathworks.com/help/releases/R2017b/mpc/examples/autonomous-vehicle-steering-using-model-predictive-control.html?s_eid=PSM_15028



Lateral position & Yaw angle
mo
ref
MPC
mv
steering angle
$\dot{x} = Ax + Bu$
$y = Cx + Du$
Vehicle Model
Lateral position
Yaw angle
References

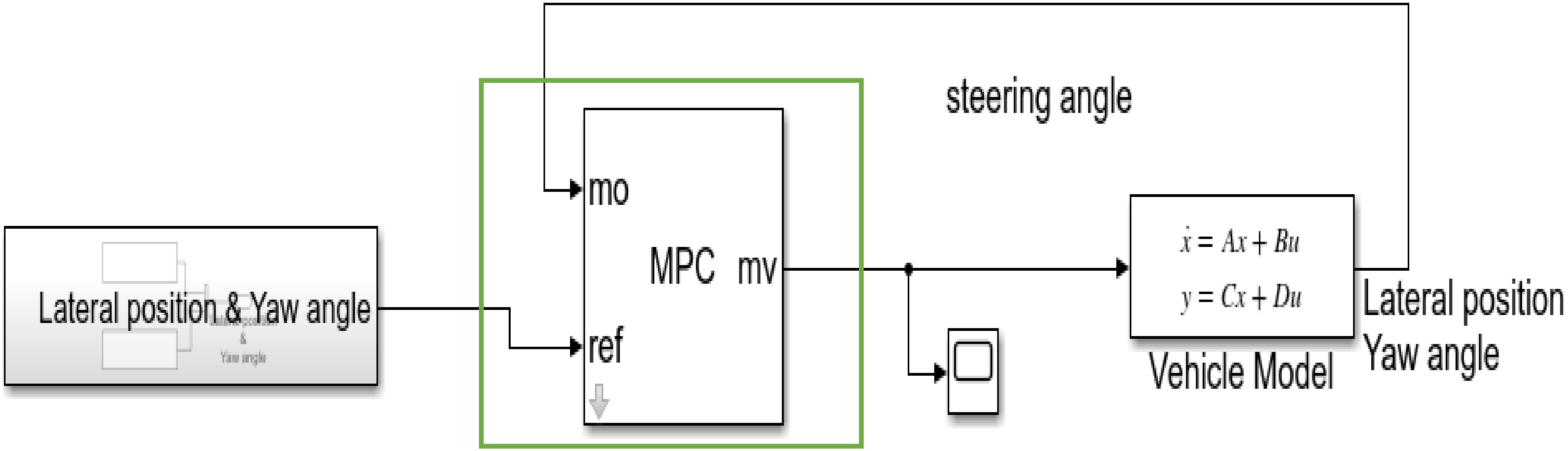
Command Window

```
>> drivingScenarioDesigner
```



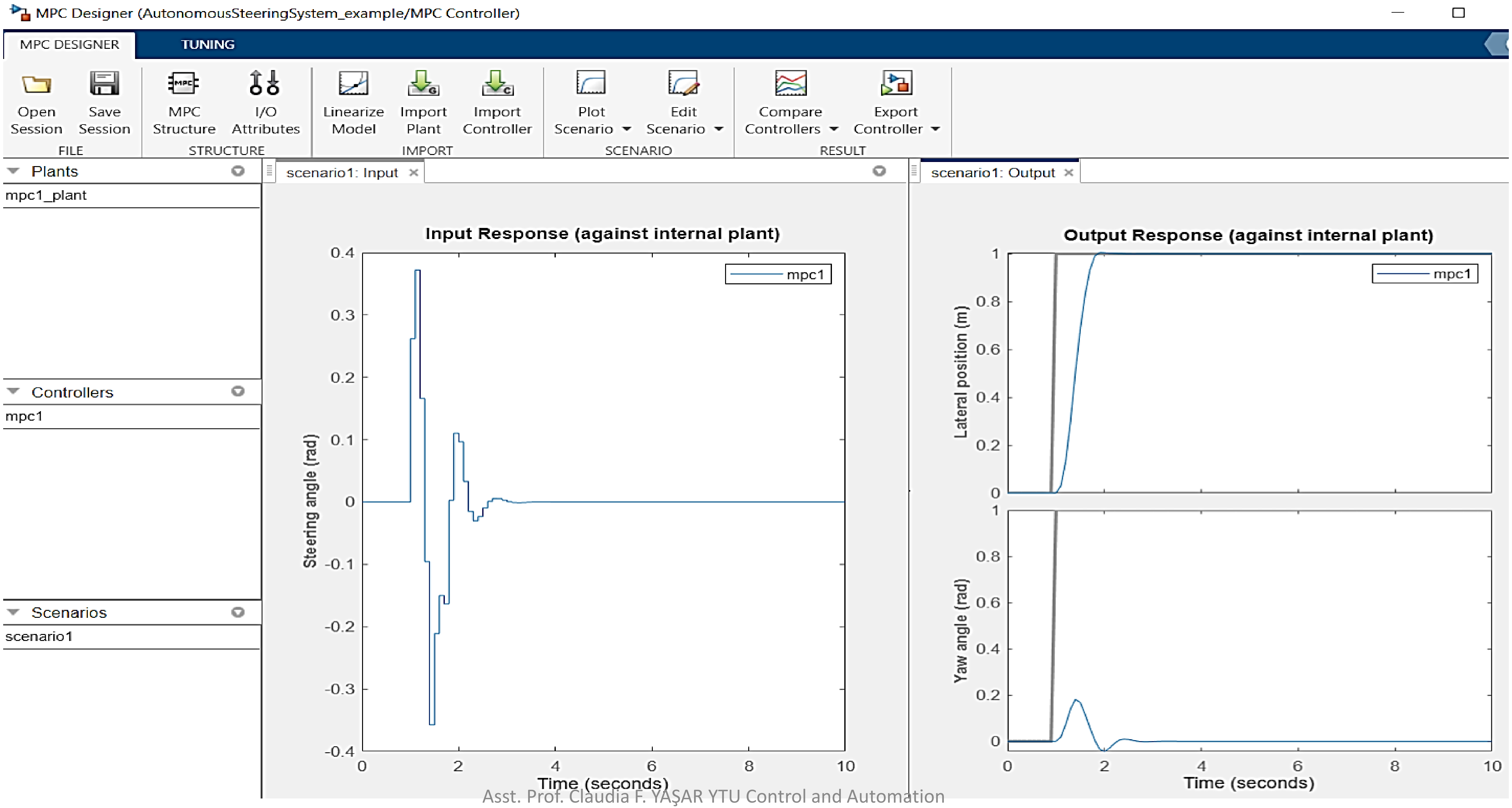
drivingScenarioDesigner

https://www.mathworks.com/products/automated-driving.html


steering angle
mo
MPC mv
ref
Lateral position & Yaw angle
$\dot{x} = Ax + Bu$
$y = Cx + Du$
Vehicle Model
Lateral position
Yaw angle

MPC

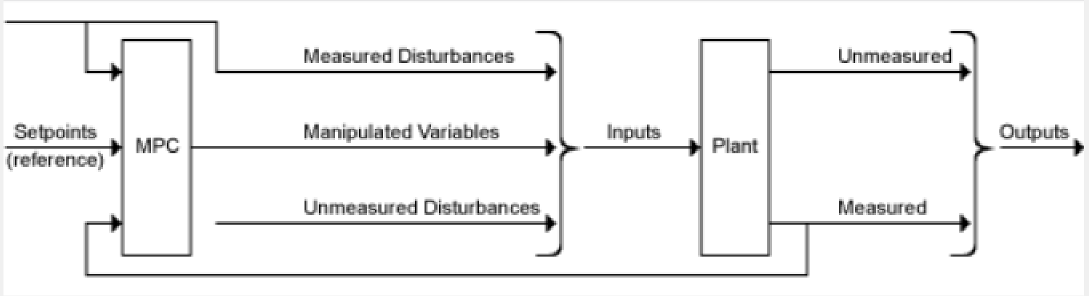
Model Predictive Control Toolbox- Design and simulate model predictive controllers



https://www.mathworks.com/products/model-predictive-control.html

View MPC Structure

MPC Structure



Plant Inputs

	Signal Type	Size	Channel Indices
1	Manipulated Variables (MV)	1	1
2	Measured Disturbances (MD)	0	
3	Unmeasured Disturbances (UD)	0	

Plant Outputs

	Signal Type	Size	Channel Indices
1	Measured Outputs (MO)	2	[1;2]
2	Unmeasured Outputs (UO)	0	

Input and Output Channel Specifications

Plant Inputs

	Channel	Type	Name	Unit	Nominal Value	Scale Factor
1	u(1)	MV	Steering angle	rad	0	1

Plant Outputs

	Channel	Type	Name	Unit	Nominal Value	Scale Factor
1	y(1)	MO	Lateral position	m	0	1
2	y(2)	MO	Yaw angle	rad	0	1

Simulation Scenario: scenario1

Simulation Settings

Plant used in simulation: Default (controller internal model)

Simulation duration (seconds) 10

☐ Run open-loop simulation ☐ Use unconstrained MPC

☐ Preview references (look ahead) ☐ Preview measured disturbances (look ahead)

Reference Signals (setpoints for all outputs)

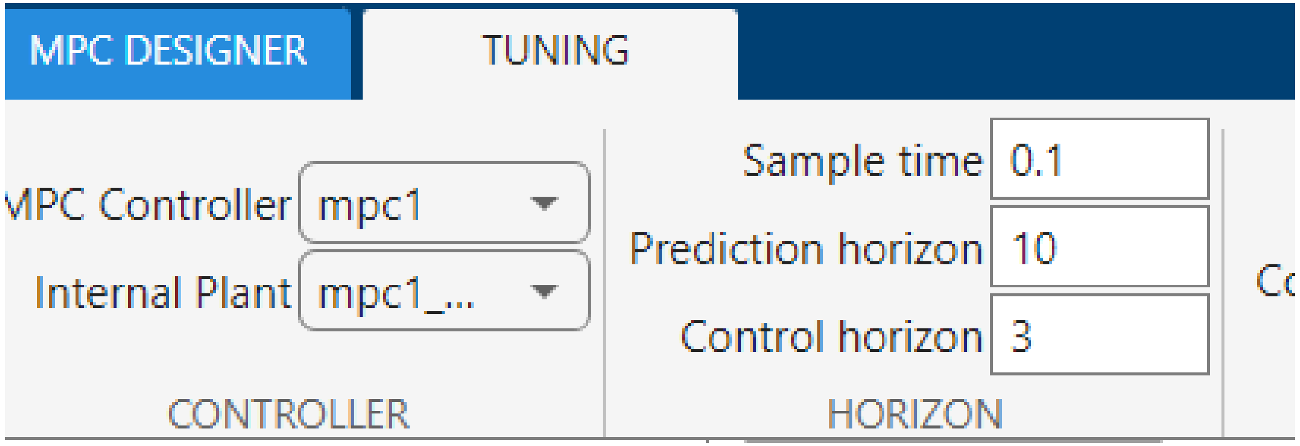
	Channel	Name	Nominal	Signal	Size	Time	Period
1	r(1)	Ref of Lateral position	0	Step	1	1	
2	r(2)	Ref of Yaw angle	0	Step	1	1	

Output Disturbances (added at MO channels)

	Channel	Name	Nominal	Signal	Size	Time	Period
1	y(1)	Lateral position	0	Constant			
2	y(2)	Yaw angle	0	Constant			

Load Disturbances (added at MV channels)

	Channel	Name	Nominal	Signal	Size	Time	Period
1	u(1)	Steering angle	0	Constant			



Input and Output Constraints

Channel	Type	Min	Max	RateMin	RateMax
▼ Inputs					
u(1)	*MV*	-0.52359877...	0.523598775...	-0.26179938...	0.261799387...
▼ Outputs					
y(1)	*MO*	-2	6		
y(2)	*MO*	-0.2	0.2		