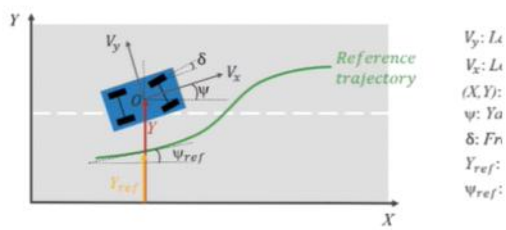
Project

Wednesday, July 09, 2025 6:12 PM

Model and simulate lateral vehicle dynamics

https://www.mathworks.com/matlabcentral/fileexchange/158356-model-predictive-control-mpc-virtual-lab?s_eid=PSM_15028&s_tid=FX_rc3_behav&status=SUCCESS

The lateral vehicle dynamics are depicted in the below picture. It's described using a bicycle model with two degrees of freedom, lateral position and yaw angle.



Here,

- X and Y denote the position of the vehicle in global coordinates
- V_x and V_y denote the longitudinal and lateral velocities of the vehicle in body-fixed coordinates with respect to the center of gravity
- ψ is the yaw angle of the vehicle
- δ is the front steering angle
- Y_{ref} is the reference lateral position
- ψ_{ref} is the reference yaw angle

Next, you will model the lateral vehicle dynamics which are separated from the longitudinal vehicle dynamics. Therefore, you will assume a constant longitudinal velocity, which you will define below.

Create a State Space Model :

📘 2. State Variables

Let the states be:

$$\mathbf{x} = \begin{bmatrix} e_y \\ e_\psi \\ v_y \\ r \end{bmatrix}$$

Where:

- $e_y = Y - Y_{ref}$: lateral position error
- $e_\psi = \psi - \psi_{ref}$: yaw angle error
- v_y: lateral velocity
- $r = \dot{\psi}$: yaw rate

Input:

$$\mathbf{u} = \delta \quad \text{(steering angle)}$$

📐 3. Vehicle Dynamics Equations (Lateral-Yaw)

Linearized lateral and yaw dynamics:

$$m(\dot{v}_y + V_x r) = F_{yf} + F_{yr}$$

$$I_z \dot{r} = aF_{yf} - bF_{yr}$$

Where:

- m: mass of the vehicle
- I_z: moment of inertia about z-axis
- a, b: distances from CG to front/rear axle
- F_{yf}, F_{yr}: lateral tire forces

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$$

$$\mathbf{y} = C\mathbf{x} + D\mathbf{u}$$

Symbol	Meaning
x\mathbf{x}	State vector (e.g., [ey, eψ, vy, r]T[e_y, e_\psi, v_y, r]^T)
u\mathbf{u}	Input vector (e.g., δ\delta, the steering angle)
x˙\dot{\mathbf{	Derivative of the state (rate of change)
y\mathbf{y}	Output vector (what you measure or want to control)
A	System matrix: how the current state affects the future state
B	Input matrix: how the input affects the state

- F_{yf}, F_{yr}: lateral tire forces

4. State Equations

After substituting and simplifying:

$$\dot{v}_y = \frac{-C_f - C_r}{mV_x} v_y + \left(\frac{-aC_f + bC_r}{mV_x} - V_x\right) r + \frac{C_f}{m}\delta$$

$$\dot{r} = \frac{-aC_f + bC_r}{I_z V_x} v_y + \frac{-a^2 C_f - b^2 C_r}{I_z V_x} r + \frac{aC_f}{I_z}\delta$$

And for tracking:

$$\dot{e}_y = v_y + V_x e_\psi$$

$$\dot{e}_\psi = r$$

x \dot{ \mathbf{	Derivative of the state (rate of change)
y\mathbf{y}	Output vector (what you measure or want to control)
A	System matrix: how the current state affects the future state
B	Input matrix: how the input affects the state
C	Output matrix: maps the state to the output
D	Feedthrough matrix: maps input directly to output (usually 0)

Define ABCD for state space model :

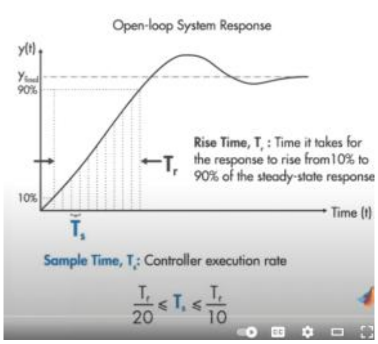
```
%x1= Vy (Lateral Velocity)
%x2= psi (Yaw rate)
%x3= e_y Lateral Position Error i.e. Deviation from the reference trajectory in the lateral (Y) direction
%x4= e_psi Yaw angle error i.e Difference between the vehicle's heading ψ and the desired heading

% A Matrix
A = [-(2*Cf + 2*Cr)/(m*Vx), 0, -Vx, -(2*Cf*lf - 2*Cr*lr)/(m*Vx);
0, 0, 1, 0;
-(2*Cf*lf - 2*Cr*lr)/(Iz*Vx), 0, 0, -(2*Cf*lf^2 + 2*Cr*lr^2)/(Iz*Vx);
1, Vx, 0, 0];
% B Matrix
B = [2*Cf/m;0;2*Cf*lf/Iz;0];
% C Matrix
C = [0 0 0 1; 0 1 0 0];
% D Matrix
D = [0; 0];
```

MATLAB based MPC Project:

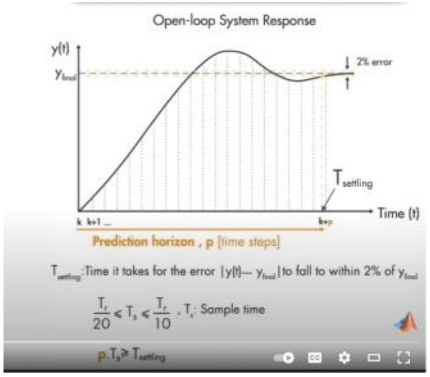
Design Parameters of MPC

<u>How to sample time(time frame in which the control runs the algorithm).</u>
Recommended Sample time:



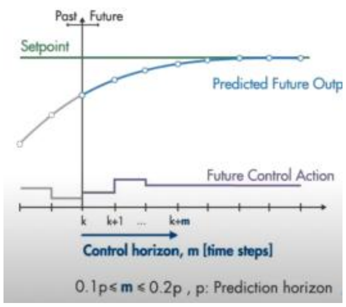
<u>Prediction Horizon:</u>

It is how far ahead you are looking into the environment.



<u>Control Horizon:</u>

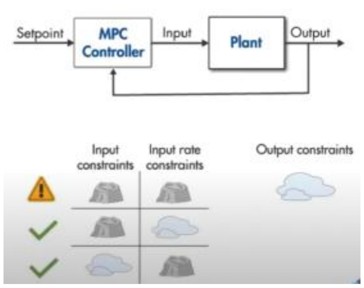
How far in future applying the logic to.



<u>Type of Constraints:</u>

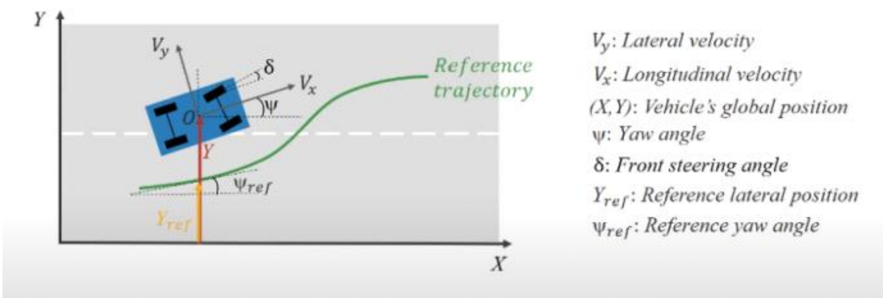
Hard: Those which sats always throw-out in search environment.
e.g.: Upper limit of the accelerating peddle

Soft: Those which are likely to appear in environment but not always present.
e.g. Weather conditions



Follow a trajectory with basic vehicle model

Monday, August 18, 2025 4:12 PM



Scenario: **you have a linearized vehicle model which needs to be used with MPC Toolbox.**

Step 1 :
Design the Model and Linearize it:

Youtube: [Mod-01 Lec-01 Introduction to Vehicle Dynamics]

Step 2 :
Create State Space Model and create Matlab file storing all the variable.

Here: Input_Variables.mlx

$$A = \begin{bmatrix} -\frac{(2C_f + 2C_r)}{mV_x} & 0 & -V_x - \frac{(2C_f l_f - 2C_r l_r)}{mV_x} & 0 \\ 0 & 0 & 1 & 0 \\ -\frac{(2C_f l_f - 2C_r l_r)}{I_z V_x} & 0 & -\frac{(2C_f l_f^2 + 2C_r l_r^2)}{I_z V_x} & 0 \\ 1 & Vx & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{2C_f}{m} \\ 0 \\ \frac{2C_f l_f}{I_z} \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad D = 0$$

Step 3:
Create a Open loop Model to see plants Behavior

Input: Steering Angle (rad)

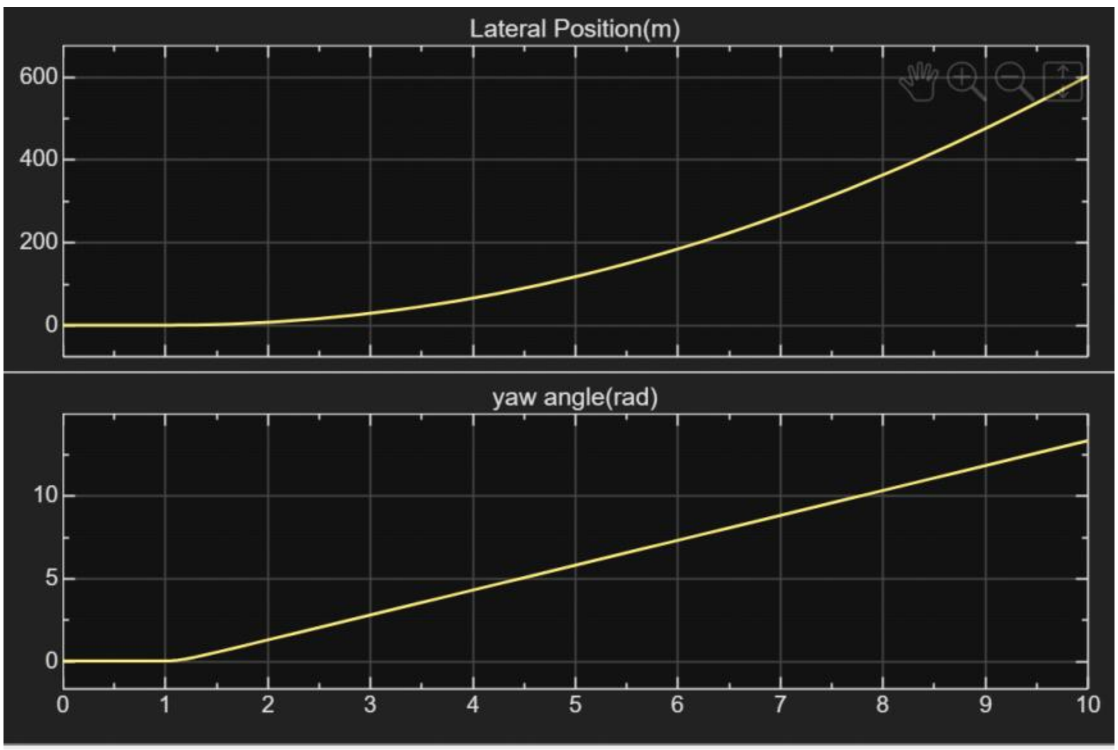
Plant(State Space of Vehicle)

Output : Lateral distance travelled ; Yaw Angle(rad)

Step 4:
Implementation:

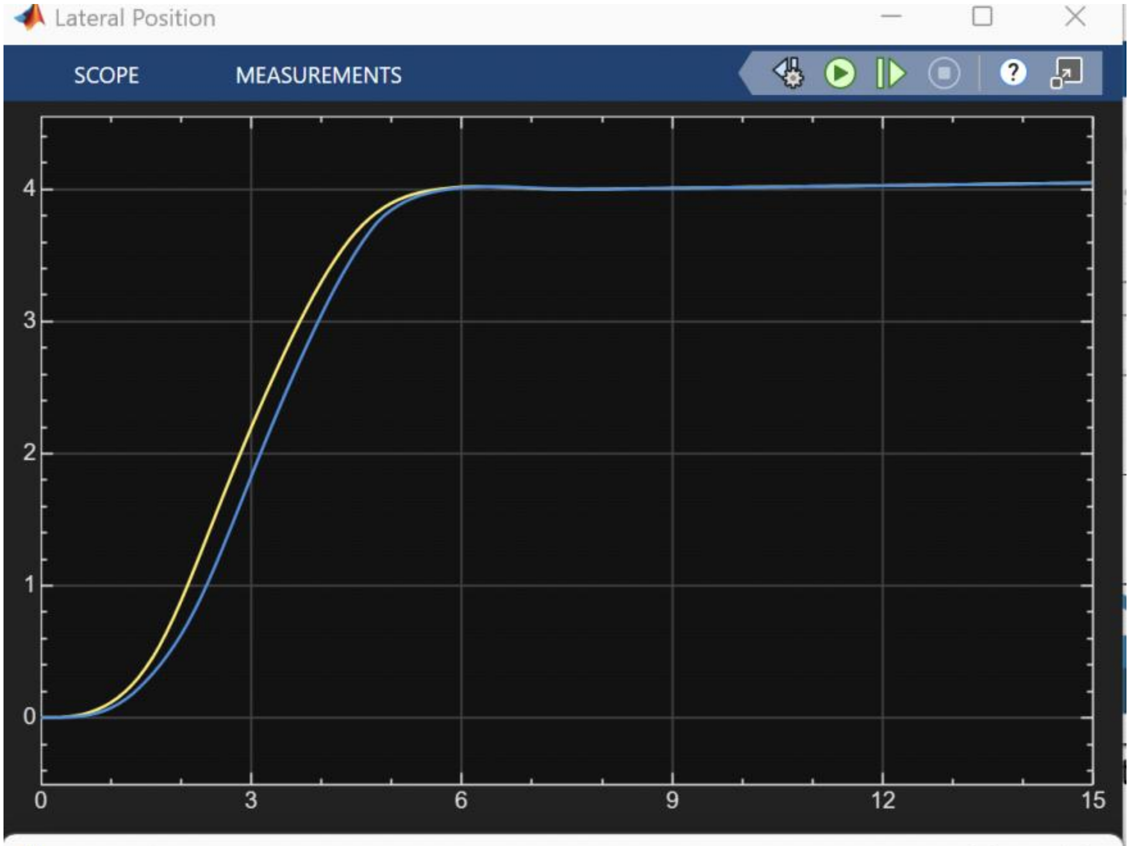
Download MPC_Basic Repo.

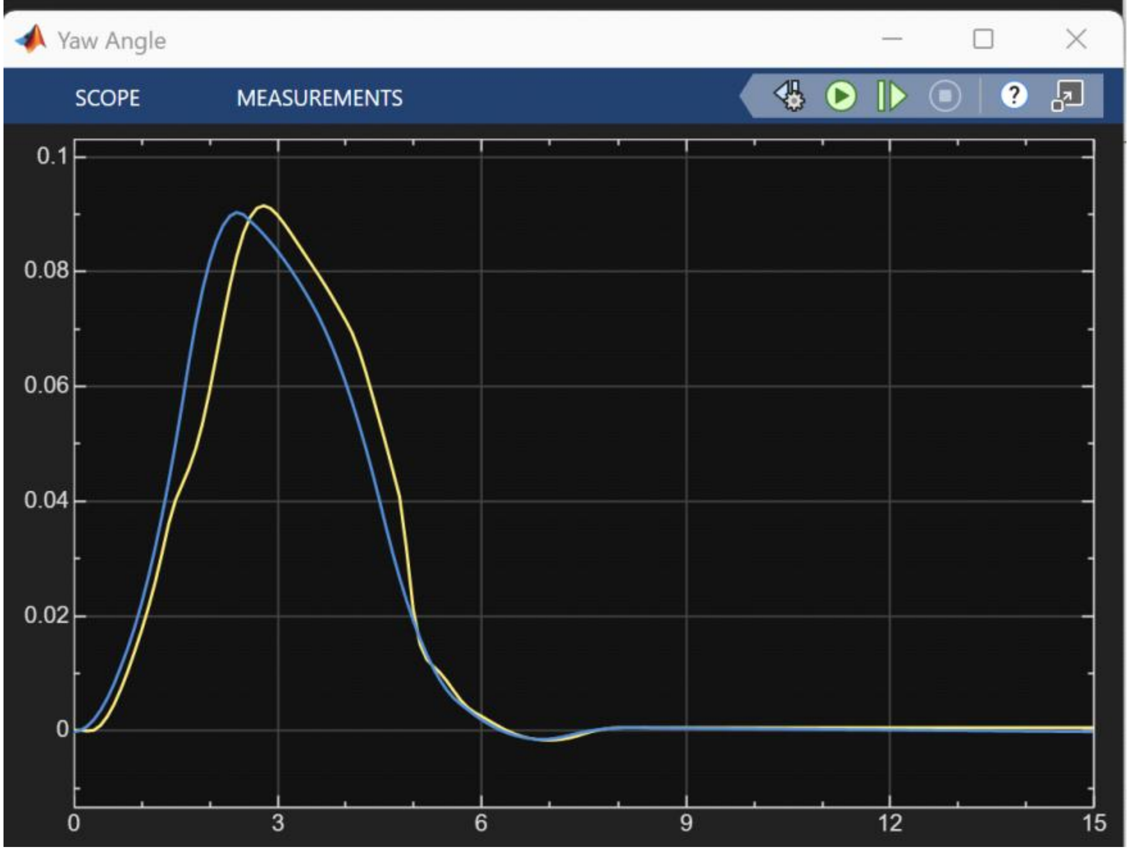
1) Open Main_script.m
Run **Main_script.m**
2)Follow scope in Plant characterization.m



Output 1)Correlation between Steering angle , yaw angle and Lateral Position.

3)Open and run AutonomousSteeringSystem.slx.


Lateral Position
SCOPE
MEASUREMENTS
4
3
2
1
0
0
3
6
9
12
15


Yaw Angle
SCOPE
MEASUREMENTS
0.1
0.08
0.06
0.04
0.02
0
0
3
6
9
12
15

MPC

Wednesday, July 09, 2025 6:14 PM

Key Elements of MPC

- **Formulation of the control problem as an (deterministic) optimization problem**

$$\min_{u_i} \sum_{i=0}^{p} \phi_i(x_i, u_i)$$
$$g_i(x_i, u_i) \geq 0 \quad \xrightarrow{?} \quad u_0 = \mu(x_0)$$
$$x_{i+1} = F(x_i, u_i) \qquad \text{HJB Eqn.}$$

- **On-line optimization**
- **Receding horizon implementation (with feedback update)**

At $t = k$, Set $x_0 = \hat{x}_k$ (Estimated Current State)
Solve the optimization problem numerically
Implement solution u_0 as the current move.
Repeat!

Popularity of Quadratic Objective in Control

- **Quadratic objective**

$$\sum_{i=0}^{p} x_i^T Q x_i + \sum_{i=0}^{m-1} u_i^T R u_i$$

Linear State Space System Model

$$x_{k+1} = Ax_k + Bu_k$$
$$y_k = Cx_k$$

- **Fairly general**
  - State regulation
  - Output regulation
  - Setpoint tracking
- **Unconstrained linear least squares problem has an analytical solution. (Kalman's LQR)**
- **Solution is smooth with respect to the parameters**
- **Presence of inequality constraints → no analytical solution**

Classical Process Control

$$p = k_c\left(1 + \frac{1}{\tau_I}\int_0^t e(t')dt' + \tau_D \frac{de}{dt}\right)$$

Ad Hoc Strategies, Heuristics →

- Regulation
- Constraint handling
- Local optimization

Lead / Lag Filters

- **Model is not explicitly used**

Classical Process Control

$$p = k_c\left(1+\frac{1}{\tau_I}\int_0^t e(t')dt' + \tau_D\frac{de}{dt}\right)$$

Lead / Lag Filters
Switches
Min, Max Selectors
If / Then Logics
Sequence Logics

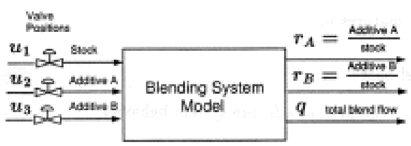
Ad Hoc Strategies, Heuristics

- Regulation
- Constraint handling
- Local optimization

- **Model is not explicitly used inside the control algorithm**
- **No clearly stated objective and constraints**

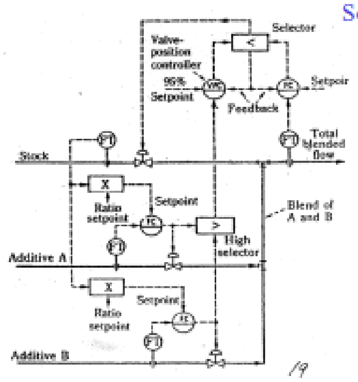
- Inconsistent performance
- Complex control structure
- Not robust to changes and failures
- Focus on the performance of a local unit

Example 1: Blending System

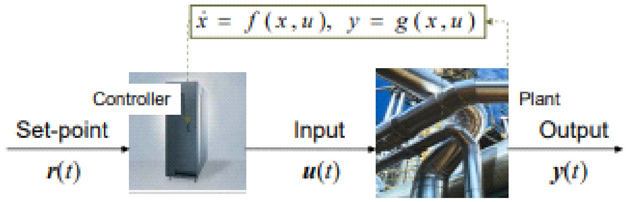


- Control r_A and r_B
- Control q if possible
- Flowrates of additives are limited

Classical Solution



Model-Based Optimal Control

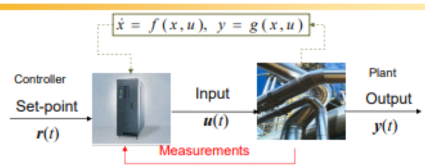


Open-Loop Optimal Control Problem

$$\min_{u_0,\dots,u_{p-1}} \left\{ \underbrace{\sum_{i=0}^{p-1} \phi(x_i, u_i)}_{\text{stage-wise cost}} + \underbrace{\phi_p(x_p)}_{\text{terminal cost}} \right\}$$

$g_i(x_i, u_i) \ge 0$	Path constraints
$g_p(x_p) \ge 0$	Terminal constraints
$\dot{x} = f(x,u)$	Model constraints

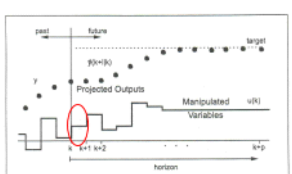
Model-Based Optimal Control



- Open-loop optimal solution is not robust
- Must be coupled with on-line state / model parameter update
- Requires on-line solution for each updated problem
- Analytical solution possible only in a few cases (LQ control)
- Computational limitation for numerical solution, esp. back in the '50s and '60s

$\dot{x} = f(x,u)$ Model constraints

Model Predictive Control (Receding Horizon Control)



- At time k, solve the open-loop optimal control problem **on-line with $x_0 = x(k)$**
- Apply the optimal input moves **$u(k) = u_0$**
- Obtain new measurements, update the state and solve the OLOCP at time $k+1$ with **$x_0 = x(k+1)$**
- Continue this at each sample time

Implicitly defines the feedback law $u(k) = h(x(k))$

Lecturenot eonMPC-J...