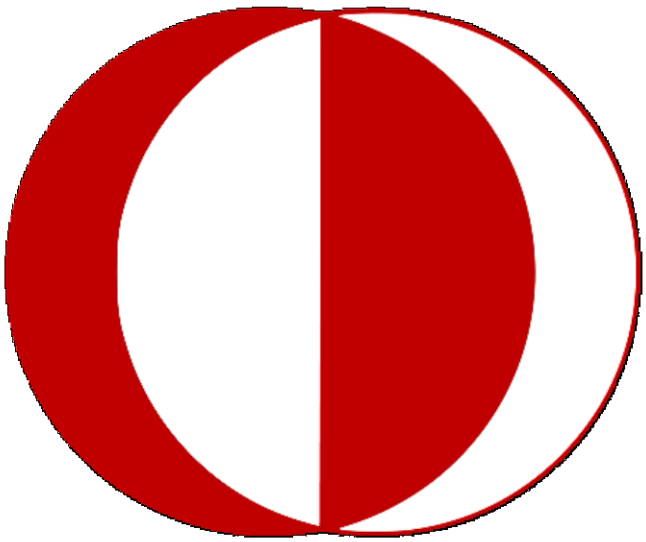


MIDDLE EAST TECHNICAL UNIVERSITY

ELECTRICAL AND ELECTRONICS ENGINEERING

EE498

CONTROL SYSTEM DESIGN AND SIMULATION

-TERM PROJECT-

Name: **İbrahim Güngen**

No: **1936939**

Instructor: **Assoc. Prof. Dr. Klaus Werner Schmidt**

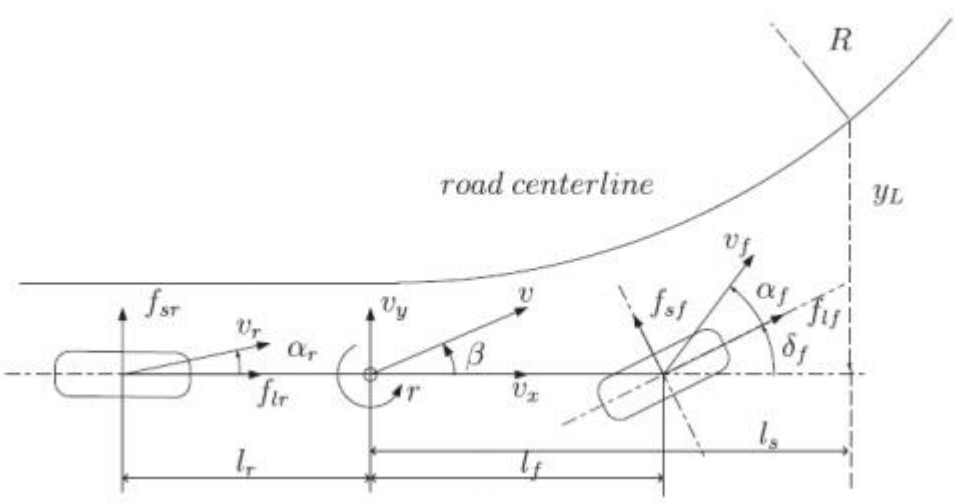
Index

1. Introduction

In this project, Model Predictive Control method will be implemented to steering system of the vehicle. Aim of this system is keep the vehicle at the center of the road autonomously. In real line number of horizon is limited at the MPC. Also, systems could be astable according to this value so that dual mode prediction method should be implemented. In order to design MPC for this system, system will be modeled first, design MPC then realize the MPC on MATLAB, and finally system will simulate in SIMULINK.

2. Modeling

At the following figure, details of the system are shown. Also model is supported.



- β: Slip angle (describes direction of the velocity vector of the vehicle with respect to the vehicle heading direction)
- v: Vehicle speed
- y_L: Distance of vehicle to the road centerline at a pre-view distance l_s
- ψ: heading angle
- $r = \dot{\psi}$: yaw rate (rotational velocity of the vehicle around its center of gravity)

$$\begin{bmatrix} \dot{\beta} \\ \dot{r} \\ \dot{\psi} \\ \dot{y}_L \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ v & l_s & v & 0 \end{bmatrix} \begin{bmatrix} \beta \\ r \\ \psi \\ y_L \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ 0 \\ 0 \end{bmatrix} \delta_f + \begin{bmatrix} 0 \\ 0 \\ -v \\ 0 \end{bmatrix} \rho$$

Figure 1. Modeling and illustration of the system

$$a_{11} = -\frac{(c_f + c_r)}{mv}, \quad a_{12} = -1 - \frac{(c_f l_f - c_r l_r)}{mv^2}$$

$$a_{21} = -\frac{(c_f l_f - c_r l_r)}{J}, \quad a_{22} = -\frac{(c_f l_f^2 + c_r l_r^2)}{Jv}$$

$$b_1 = \frac{c_f}{mv}, \quad b_2 = \frac{c_f l_f}{J}$$

Table 2
Vehicle parameters for the linear model (5).

m	2023 kg
l_f	1.26 m
c_f	2.864e+5 N/rad
J	6286 kg m^2
l_r	1.90 m
c_r	1.948e+5 N/rad

Figure 2. Modeling of the system 2

Specification of the model is supported with model document as previous. Also, steering angle is input of the system. This is limited +,-20 degree (in modeling, it should be used as radian). Output of the system is distance of the vehicle to the road centerline at the preview distance; controller should be made zero this value at the steady state.

3. MPC Designing and Matlab Realizing

At MPC, current state is measured at any time k first. Secondly, solving a finite horizon at future steps N, taking account future and current constraints and obtaining input value. Thirdly, first computed input is applied to the system, and then measured next state at k+1. This process is repeated all next steps. Following optimal control problem is solved each step.

Optimization Problem at Time Step k

$$\min_{u_{0|k},\ldots,u_{N-1|k}} \{J = \sum_{j=0}^{N-1} (x_{j|k}^T\, Q\, x_{j|k} + u_{j|k}^T\, R\, u_{j|k}) + x_{N|k}^T\, Q_{\mathsf{f}}\, x_{N|k}\} \qquad (1)$$

subject to

$$x_{j+1|k} = A\, x_{j|k} + B\, u_{j|k}, \quad j = 0, 1, \ldots, N-1; \quad x_{0|k} \text{ given} \qquad (2)$$

$$x_{\mathsf{min}} \leq x_{j|k} \leq x_{\mathsf{max}}, \quad j = 0, 1, \ldots, N \qquad (3)$$

$$u_{\mathsf{min}} \leq u_{j|k} \leq u_{\mathsf{max}}, \quad j = 0, 1, \ldots, N-1 \qquad (4)$$

$$f_x^T\, x_{j|k} \leq v_x, \quad j = 0, 1, \ldots, N \qquad (5)$$

$$f_u^T\, u_{j|k} \leq v_u, \quad j = 0, 1, \ldots, N-1 \qquad (6)$$

Figure 3. Formulas of optimization problem

In order to implement MPC we have to obtain A, B, C, D matrices, and then by using these we have to obtain state cost matrices (Q,Qf). R is the input cost matrices and depends on input weight value directly. In dual mode prediction Qf is equal to the P matrices which is obtained with Lyapunov equation. P is derived from the cost of the infinite horizon LQR problem.

$$P - (A - BK_\infty)^T\, P\, (A - B\, K_\infty) = Q + K_\infty^T\, R\, K_\infty$$

Figure 4. Lyapunov equation

Conversion to Quadratic Optimization Problem

$$\min_{u_{0|k},\ldots,u_{N-1|k}} \{J = U^T\, M\, U + 2\, \alpha^T\, U + \beta\}$$

subject to

$$F\, U \leq V$$

Figure 5. Quadratic optimization problem

I have chosen number of horizon 4. With dual mode prediction, there are no significant differences between values near to the 4 because after this horizon, controller assumes infinite horizon.

Main modeling parameters are given previous part and also A, B and C matrices should be calculated. Therefore, obtained matrices should be made discrete. Given parameters and values following matrices are obtained.

```
Ac = [a11 a12 0 0;
      a21 a22 0 0;
```

```
      0 1 0 0;
      v ls v 0];
Bc = [b1; b2; 0; 0];
C =  [0 0 0 1];
D =  [0];


A_dis =  exp(Ac*T)
B_dis =  integral(exp(Ac*x),x,[0,T])*Bc)
```

Other calculation parameters are calculated with following formulas for N=4;

```
G = [zeros(4,1) zeros(4,1)   zeros(4,1)   zeros(4,1);
       B         zeros(4,1)  zeros(4,1)   zeros(4,1);
      A*B          B         zeros(4,1)   zeros(4,1);
     A^2*B        A*B           B         zeros(4,1);
     A^3*B       A^2*B         A*B           B      ];
H = [eye(4); A; A^2; A^3; A^4];
Q = C'*C;
R = p* eye(4);
```

p is the input weight, which is effect performance of the controller.

Speed variable of the vehicle is not given at the model so that I will show that effect variable change of speed, initial distance to the centerline at preview distance, preview distance and road curvature.

In this model predictive controller, dual mode prediction is used. With this prediction, we prevent the instability. This can be occurred when finite horizon is used like this project.

a. Cost Matrices and Final Cost Matrices

Cost matrices(Q) could be chosen either identity matrix or C^TC according to the design. Also Qf could be P or Q according to using dual mode prediction or not. At this part, I would like to show these differences.

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0.001; %1/R
v = 30;          %m/sec
ls = 10; % m
Qf = P;
%%
```

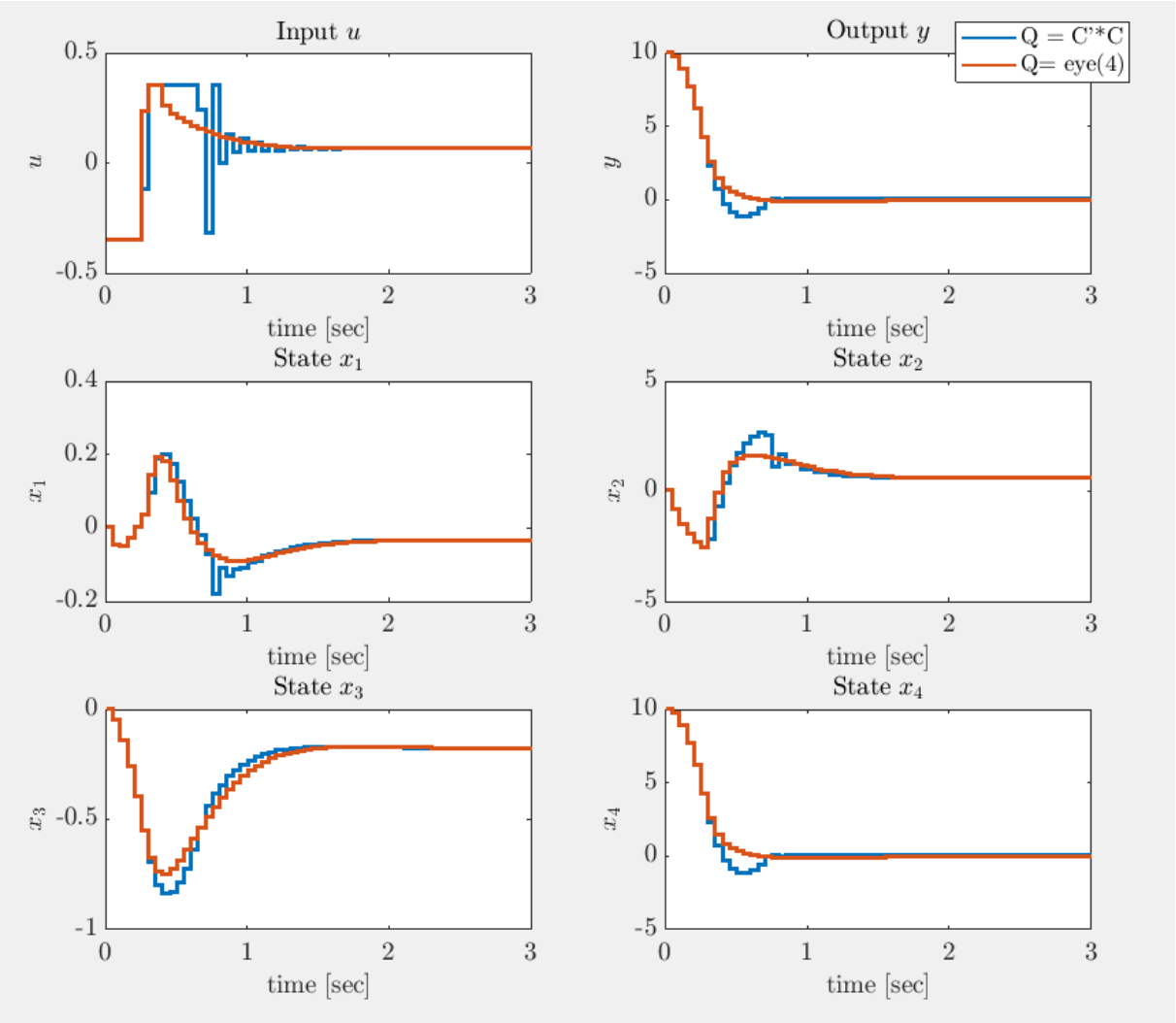


Figure 6. Identity cost matrices differences

For this system identity cost matrix response is better.

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0.001; %1/R
v = 50;          %m/sec
ls = 10; % m
Q = C'*C;
%%
```

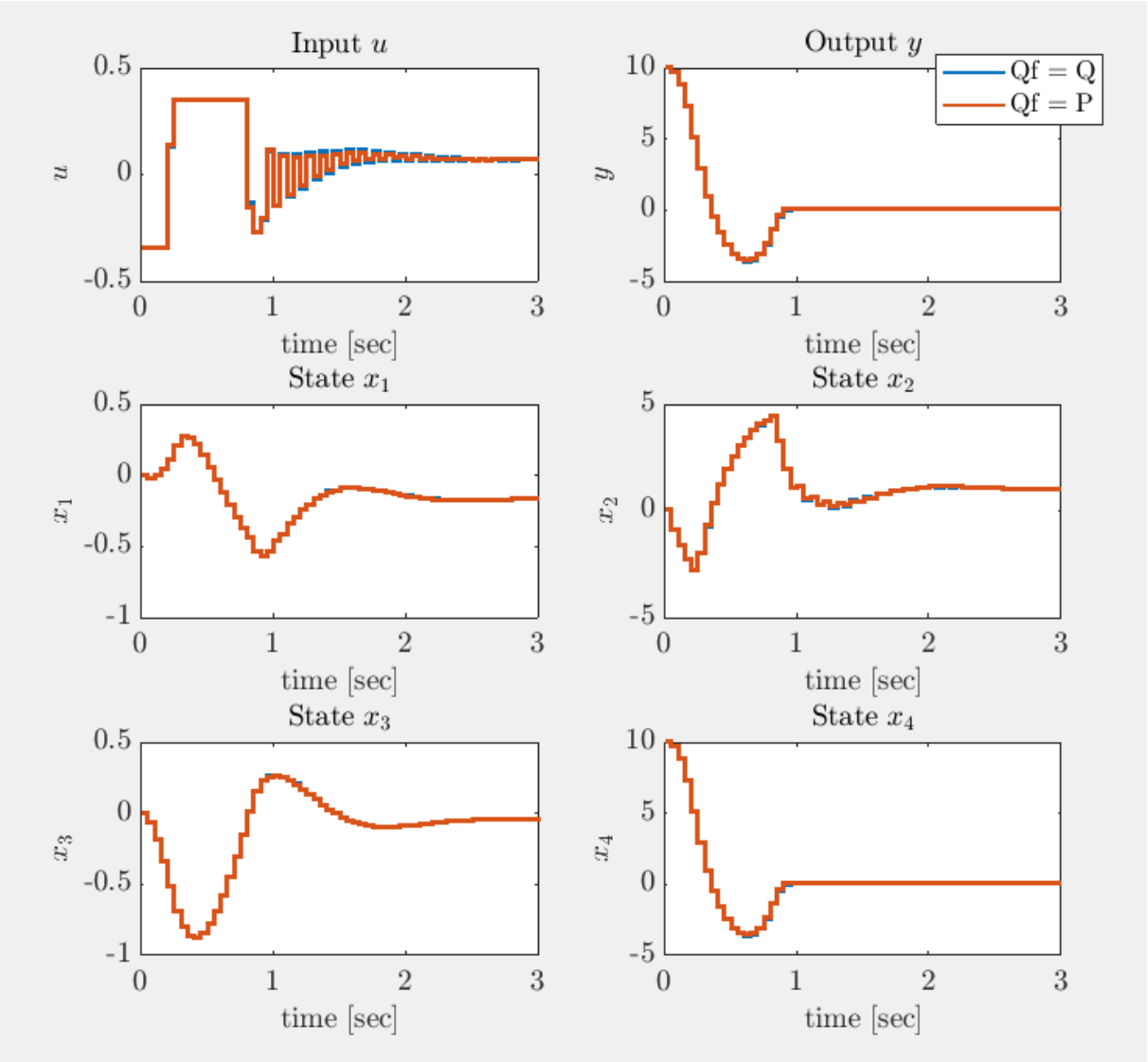


Figure 7. Final identity cost matrices according to dual mode prediction

Responses are almost same.

b. Input Weight

At this system, effect of input weight cannot be observed obviously because input signal is limited, input weight effect input signal. If it is not bounded, input signal became higher when input weight slower.

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0.001; %1/R
v = 30;          %m/sec
ls = 10; % m
%%
```

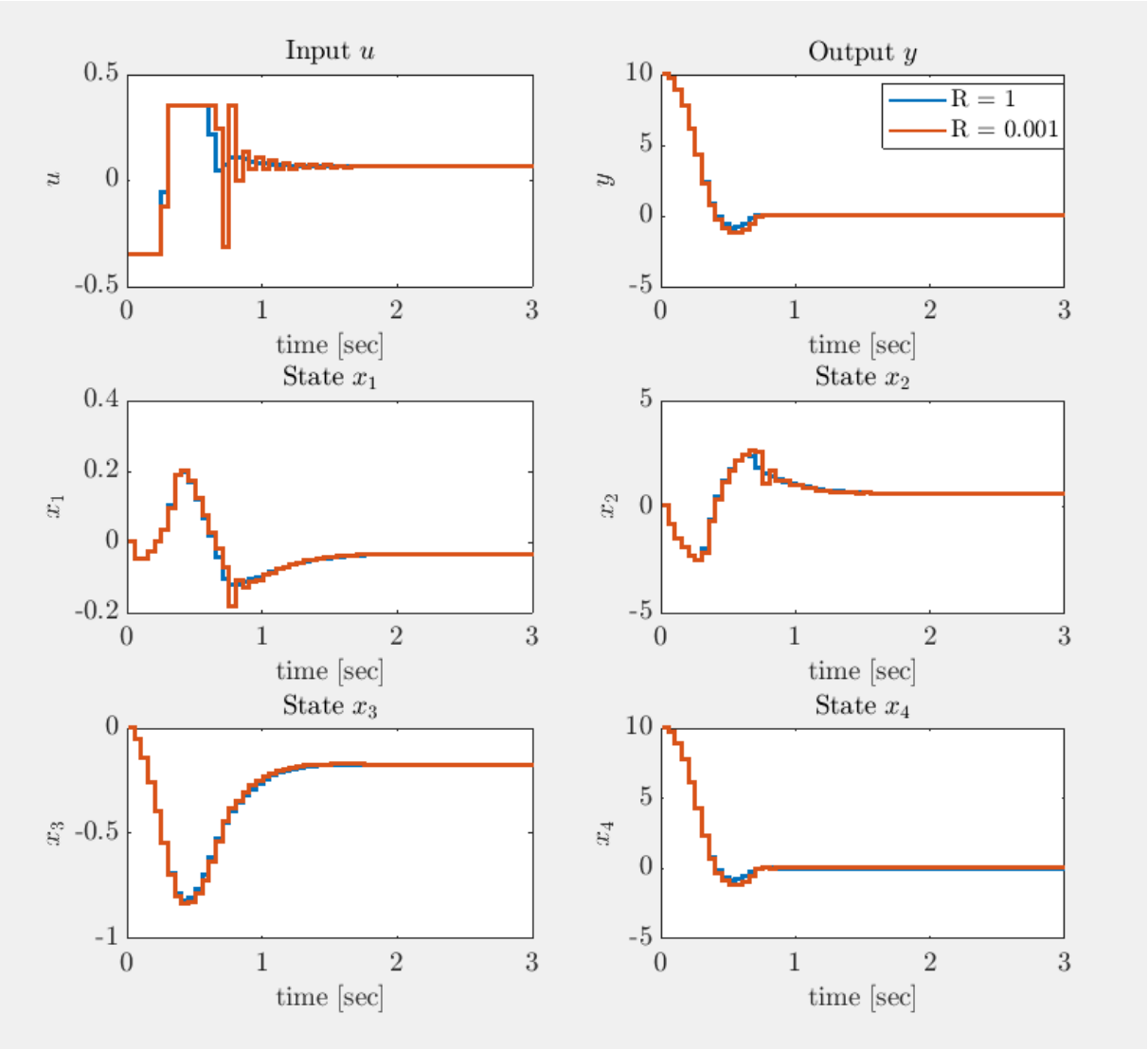


Figure 8. input weight differences

c. Road Curvature

According to road curvature, steering angle of the vehicle which means input signal has an offset. As following figure shows differences of disturbance signal with constant other parameters. Speed is 30 m/s and R is infinity (straight line) and 1000m, initial distance 10m and preview distance is 5m taken.

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0, 0.001; %1/R
v = 30;          %m/sec
ls = 5; % m
```

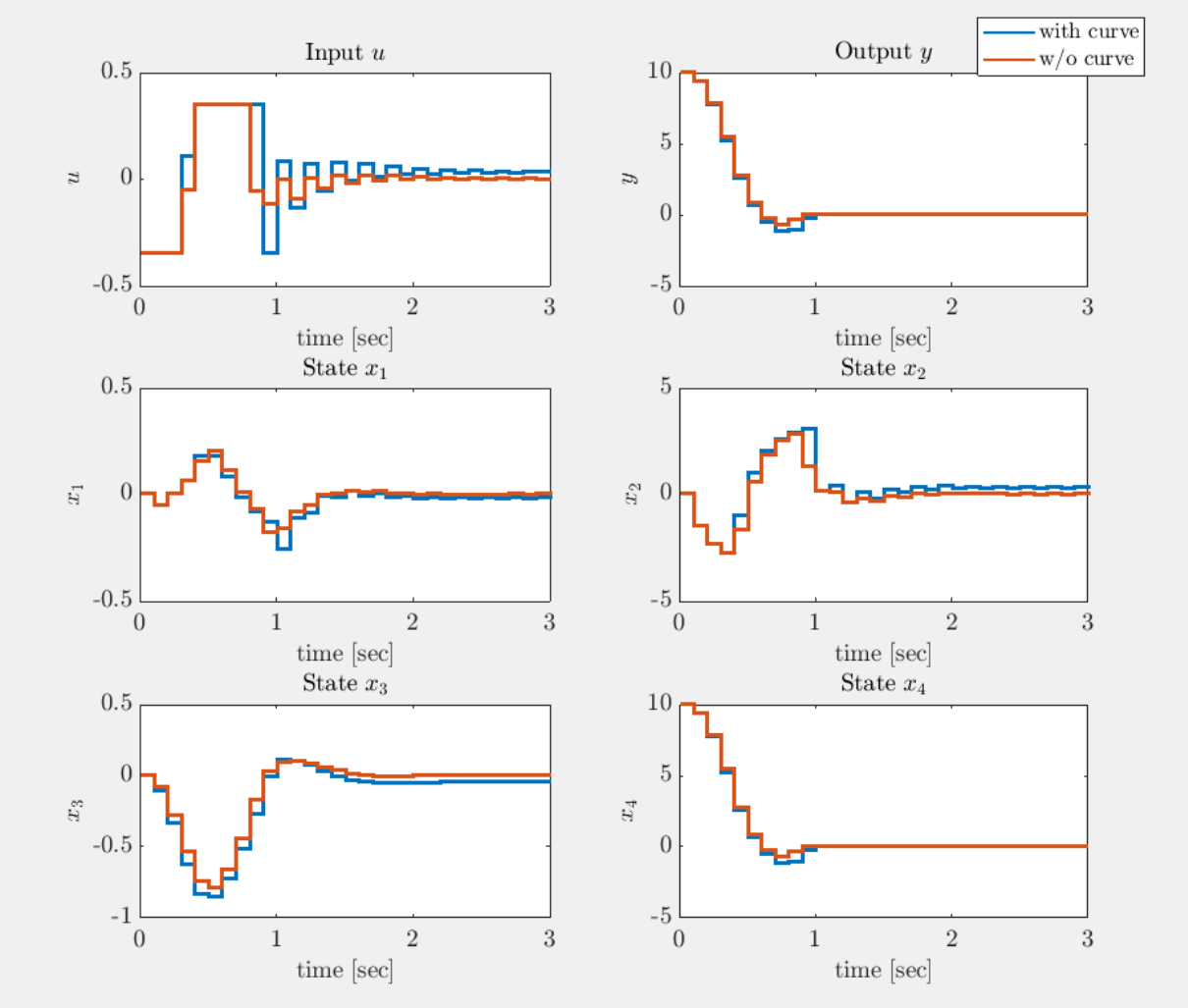


Figure 9. road curvature differences

Note: Input values of the system limited -20,20 degree but in these calculation this value should be taken radian so that all figure shows that -0.3491, 0.3491radian values.

d. Velocity

When velocity increases, undershoot occurs at the output response and also input response rising time decreases, after some high speed controller cannot handle with this parameters. But this speeds are not realized with vehicle today (80m/s = 288km/h).

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0.001; %1/R
v = 20, 60, 80;          %m/sec
ls = 5; % m
```

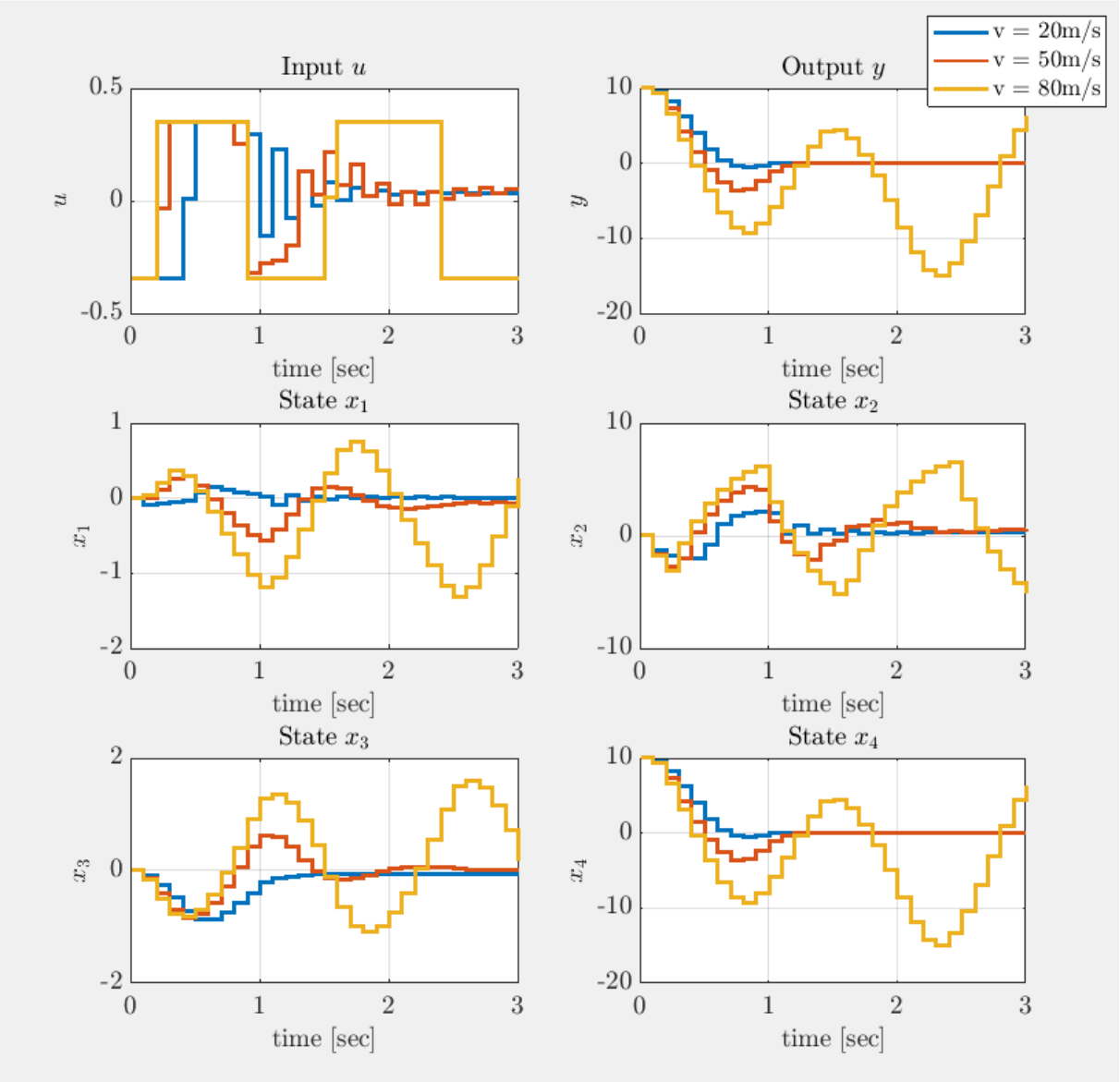


Figure 10. speed differences

Note: Input values of the system limited -20,20 degree but in these calculation this value should be taken radian so that all figure shows that -0.3491, 0.3491radian values.

e. Initial Distance from Center Line

Aim of this controller is keep the vehicle at the road line so that initial distance to the centerline at preview distance should become 0. So that initial value of this value is given to the controller. Because of the speed of the response undershoot is observed and converging time of the distance is increase and vehicle goes oscillating.

```
%% changable values
x0 = [ 0;0;0;10], [ 0;0;0;100] ;
p_dis = 0.001; %1/R
v = 30;          %m/sec
ls = 5; % m
```

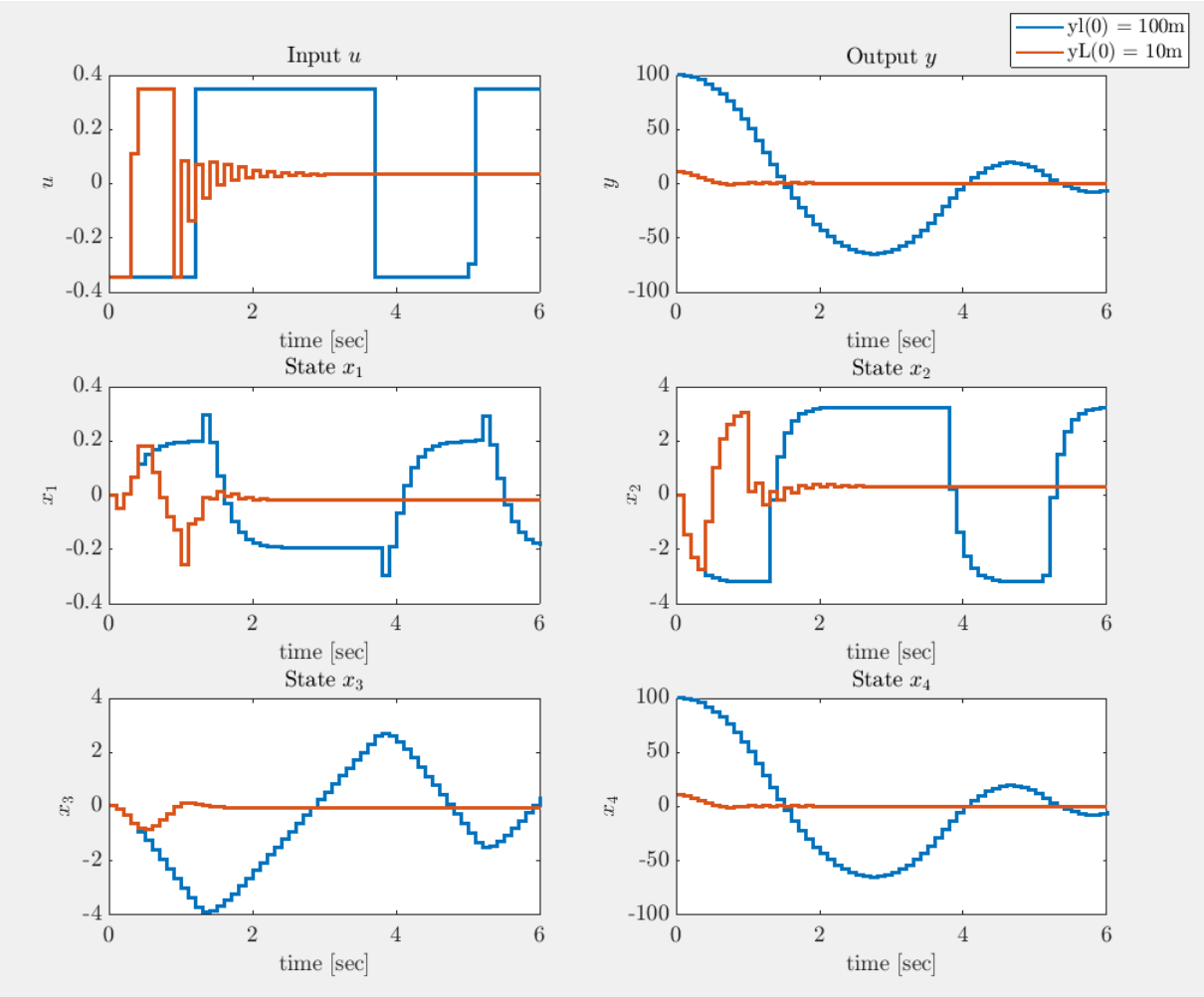


Figure 11. initial error differences

Note: Input values of the system limited -20,20 degree but in these calculation this value should be taken radian so that all figure shows that -0.3491, 0.3491 radian values.

f. Preview Distance

Increasing preview distance of the controller, converging time of the vehicle decreases because predicting to the signal is more realistic.

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0.001; %1/R
v = 30;          %m/sec
ls = 5,10,20; % m
```

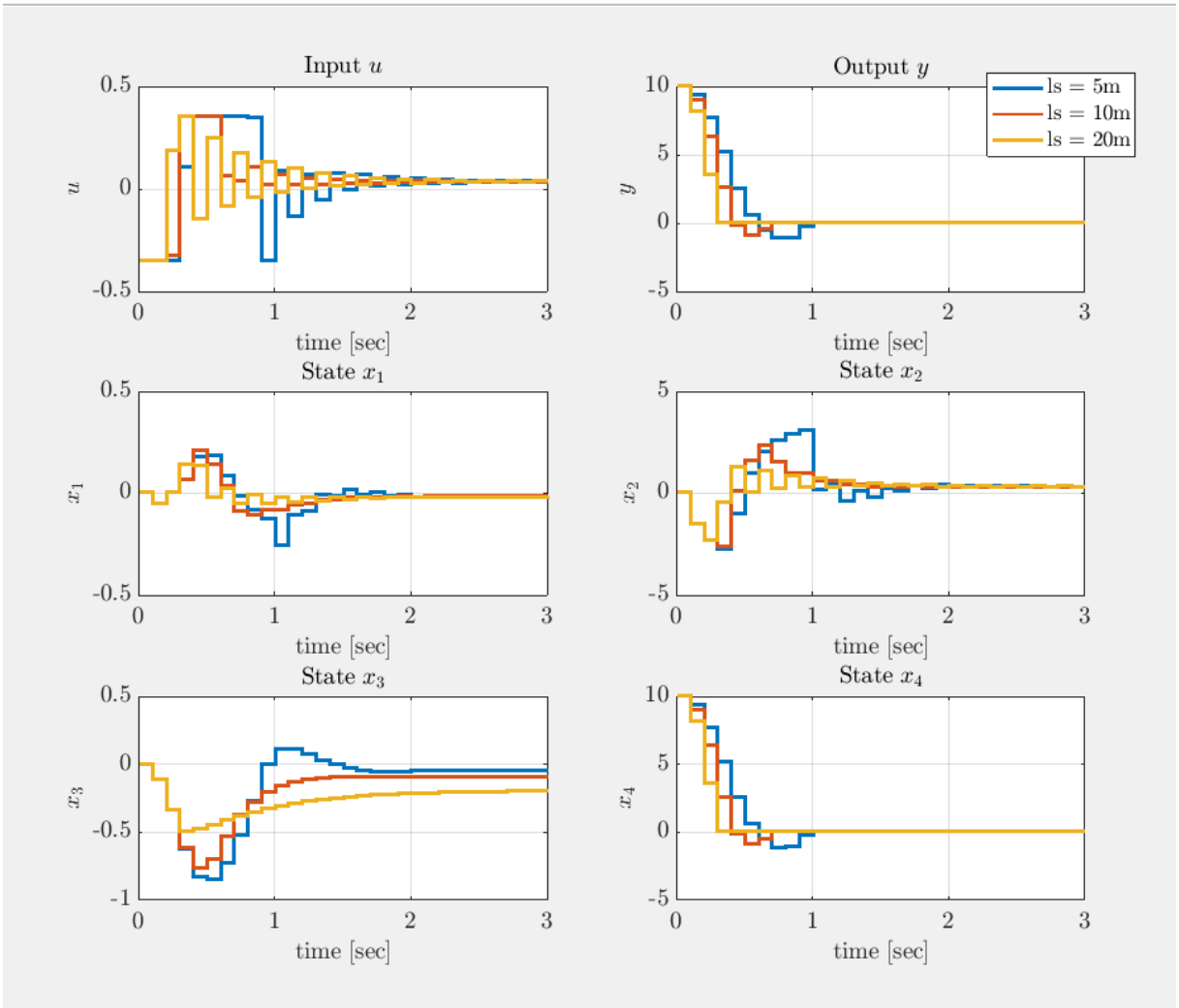


Figure 12. preview distance differences

Note: Input values of the system limited -20,20 degree but in these calculation this value should be taken radian so that all figure shows that -0.3491, 0.3491 radian values.

g. Sampling Time

Until now, all simulations are made with Ts = 0.1s. But this value should be arranged according to the system and system requirement. This part of the project differences of the sampling time values are shown.

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0.001; %1/R
v = 30;          %m/sec
ls = 10; % m
```

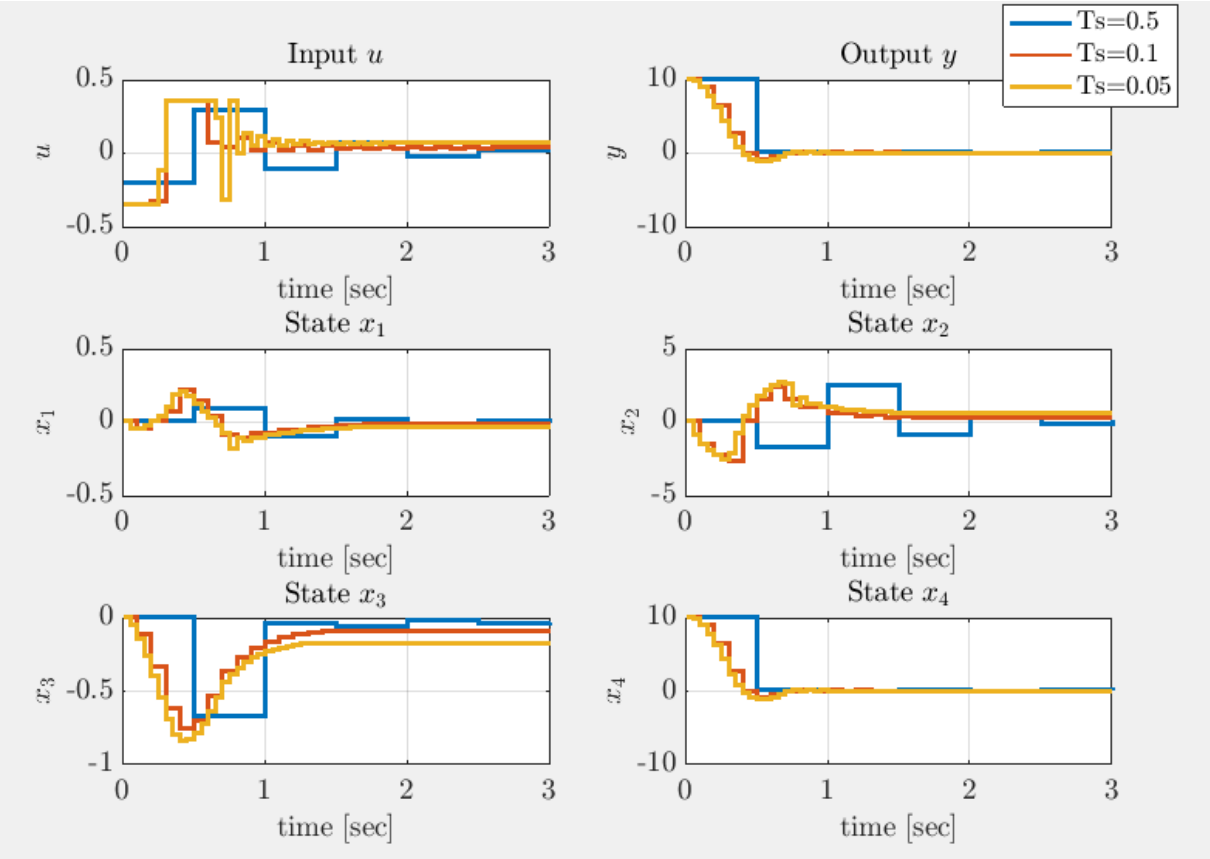


Figure 13. sampling time differences

4. Simulink Simulation

In order to make a simulation with SIMULINK, I have used MPC toolbox. This toolbox enables to design MPC easily by connecting input, output and reference signals. Also, like boundaries, measured disturbance can be arranged. When we double click to the MPC block we can reach MPC designer tool of the Simulink. Inside the designer, we can arrange number of horizon, step size, constraints, performance, weights and all related parameter is set. After arranging these, designer gives us input and output response of the MPC.

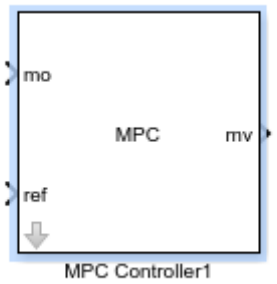


Figure 14. block of the MPC

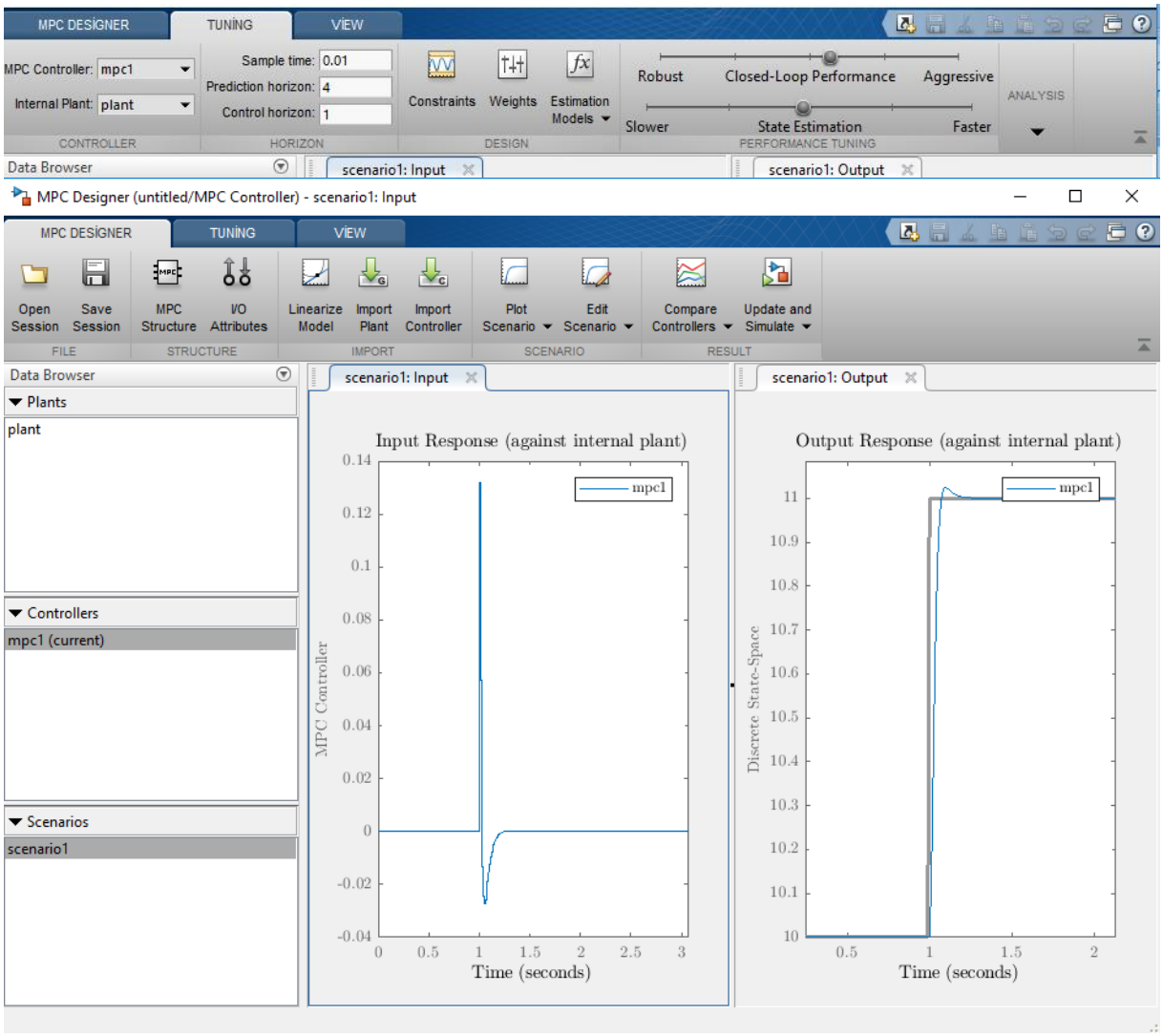


Figure 15. MPC designed of Simulink

To make a simulation, I have designed discrete time state-space model and connected to the MPC. Moreover, with this model, disturbance input could not model.

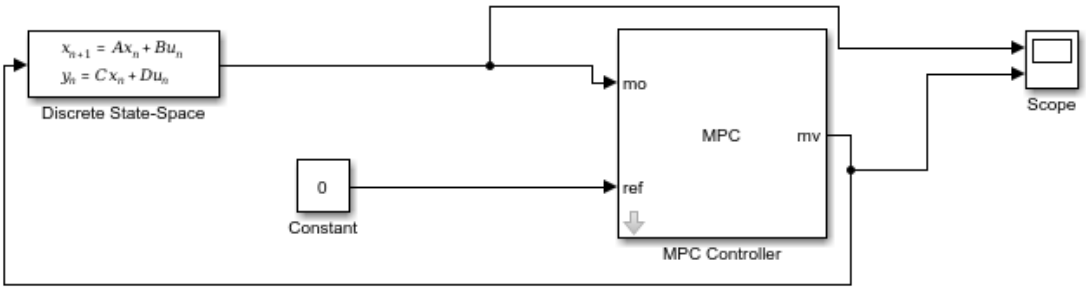


Figure 16. Simulink model 1

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0; %1/R
v = 30;          %m/sec
ls = 10; % m
%%
```

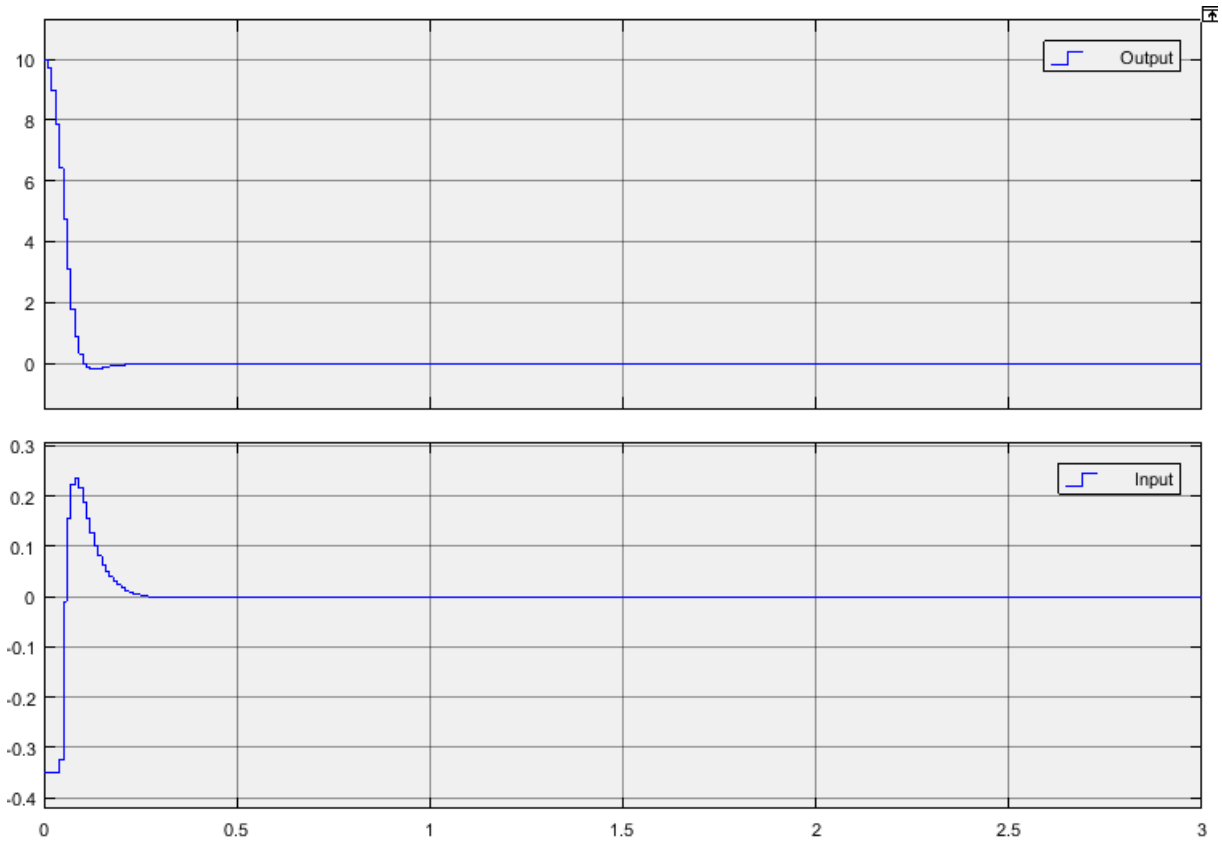


Figure 17. results of values are shown up

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0; %1/R
v = 50;          %m/sec
ls = 20; % m
%%
```

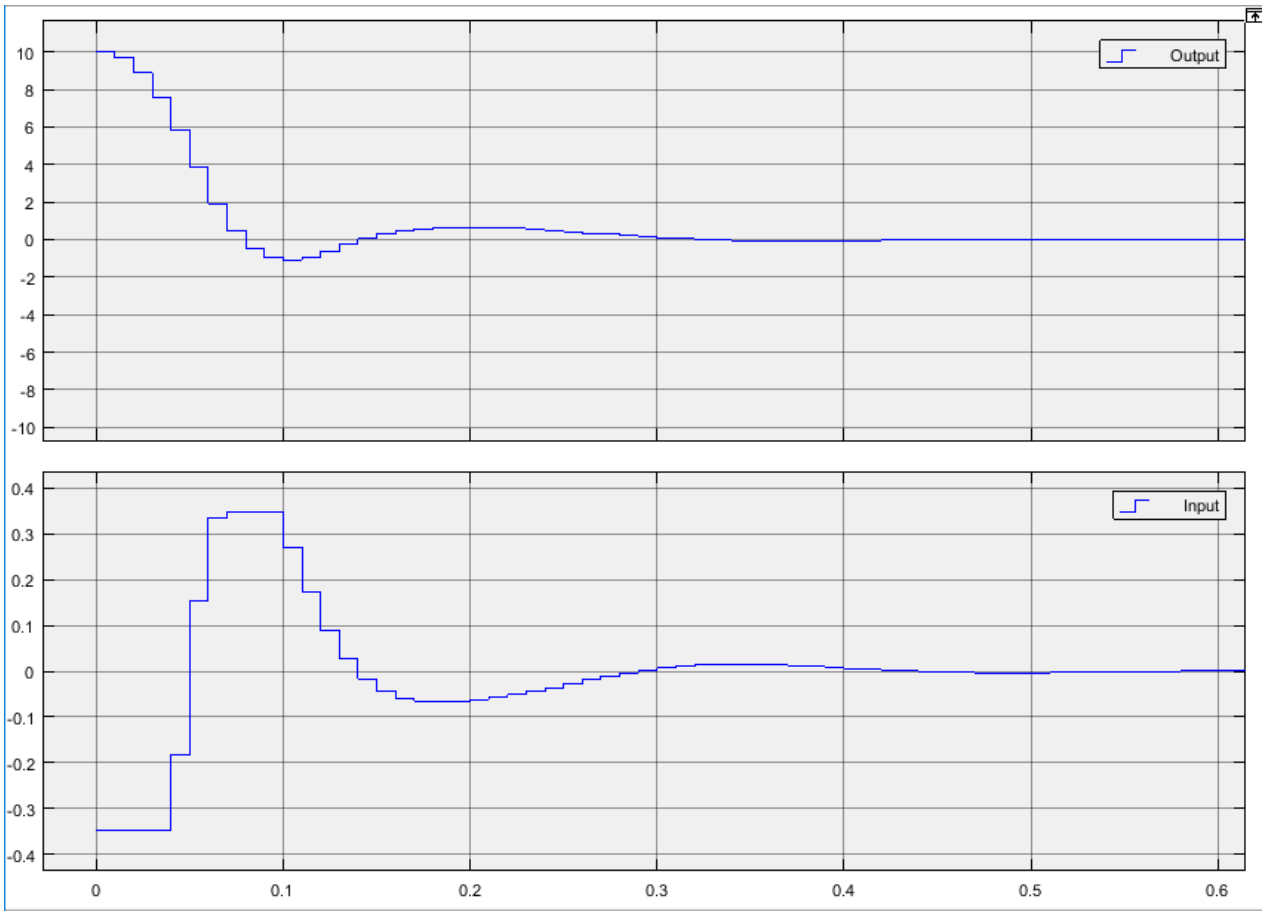


Figure 18. results of values are shown up

```
%% changable values
x0 = [ 0;0;0;50];
p_dis = 0; %1/R
v = 30;          %m/sec
ls = 5; % m
%%
```

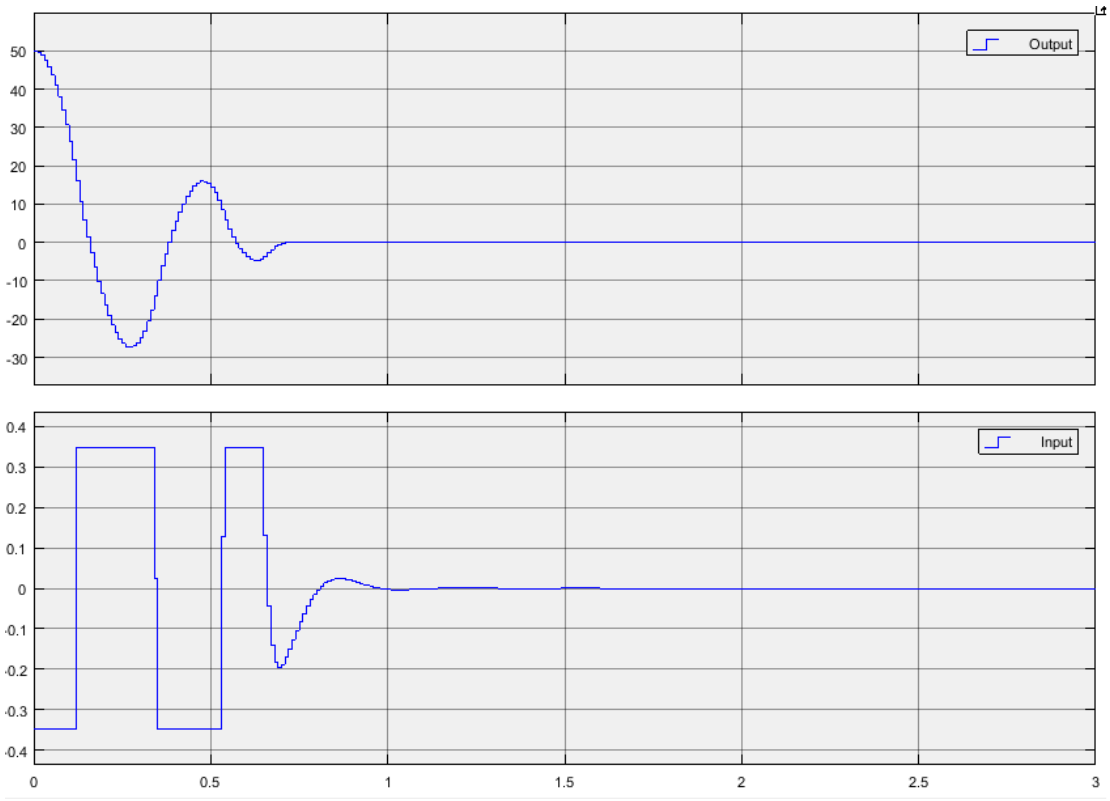


Figure 19. results of values are shown up

```
%% changable values
x0 = [ 0;0;0;50];
p_dis = 0; %1/R
v = 30;          %m/sec
ls = 10; % m
%%
```

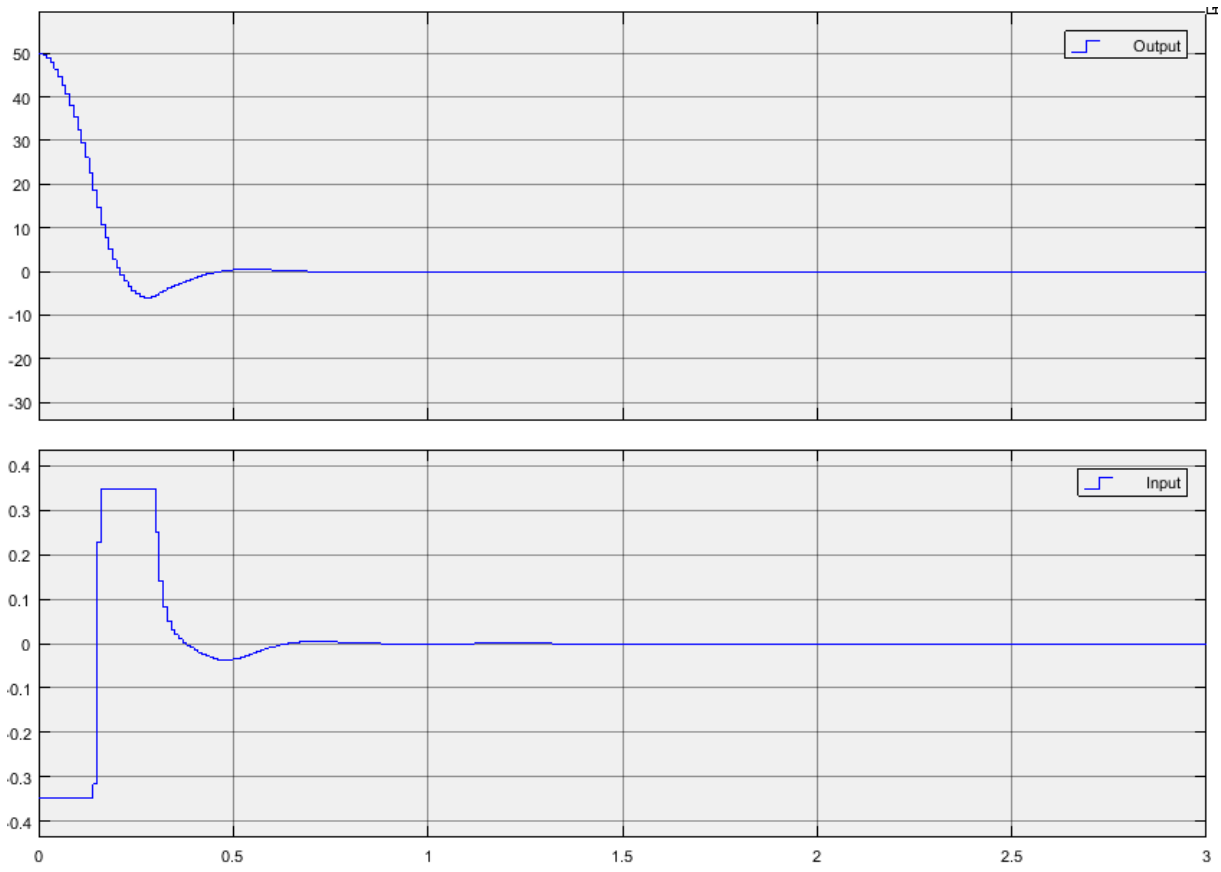


Figure 20. results of values are shown up

In order to modeling disturbance signal, state space modeling is obtained with Matlab block of the Simulink. I only cover this result a few results simulated up.

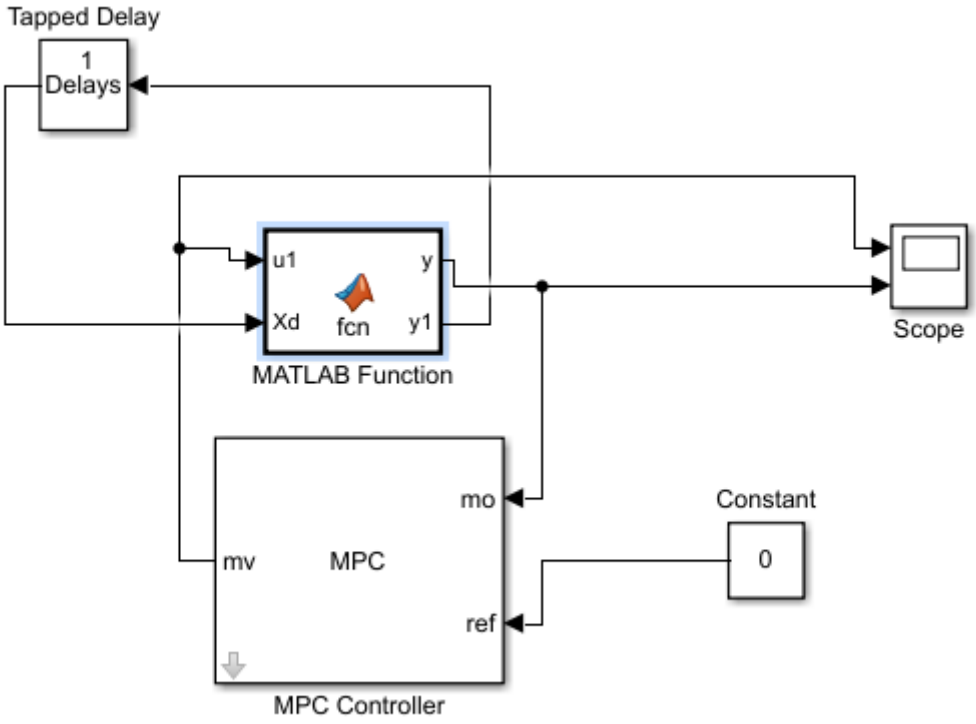


Figure 21: Simulink model 2

```
function [y,yl] = fcn(ul,Xd)
%#codegen
Xl = Xd;
A = [0.671440949146974 -0.0349851588312698 0 0;
    0.0517781225234599 0.734336221121412 0 0;
    0.00146077048938851 0.0430296983691291 1 0;
    1.26764831097370 0.864914162037313 1.50000000000000 1];

B = [[0.138024770584345;2.47712399331690;0.0650504336464155;1.45998769806594];

C = [0 0 0 1];

D =[0];

p_diss = [0; 0; -50; 0];
R = 50; %m

X2 = A*Xl+ B* ul + p_diss*(1/R);
Y = C*Xl + D*ul;

y = Y;
yl = X2;
```

Figure 22. Matlab code of Simulink block

As figure shows that, input signal have an ofset as Matlab results shows.

```
%% changable values
x0 = [ 0;0;0;10];
p_dis = 0.001; %1/R
v = 30;          %m/sec
ls = 20; % m
%%
```

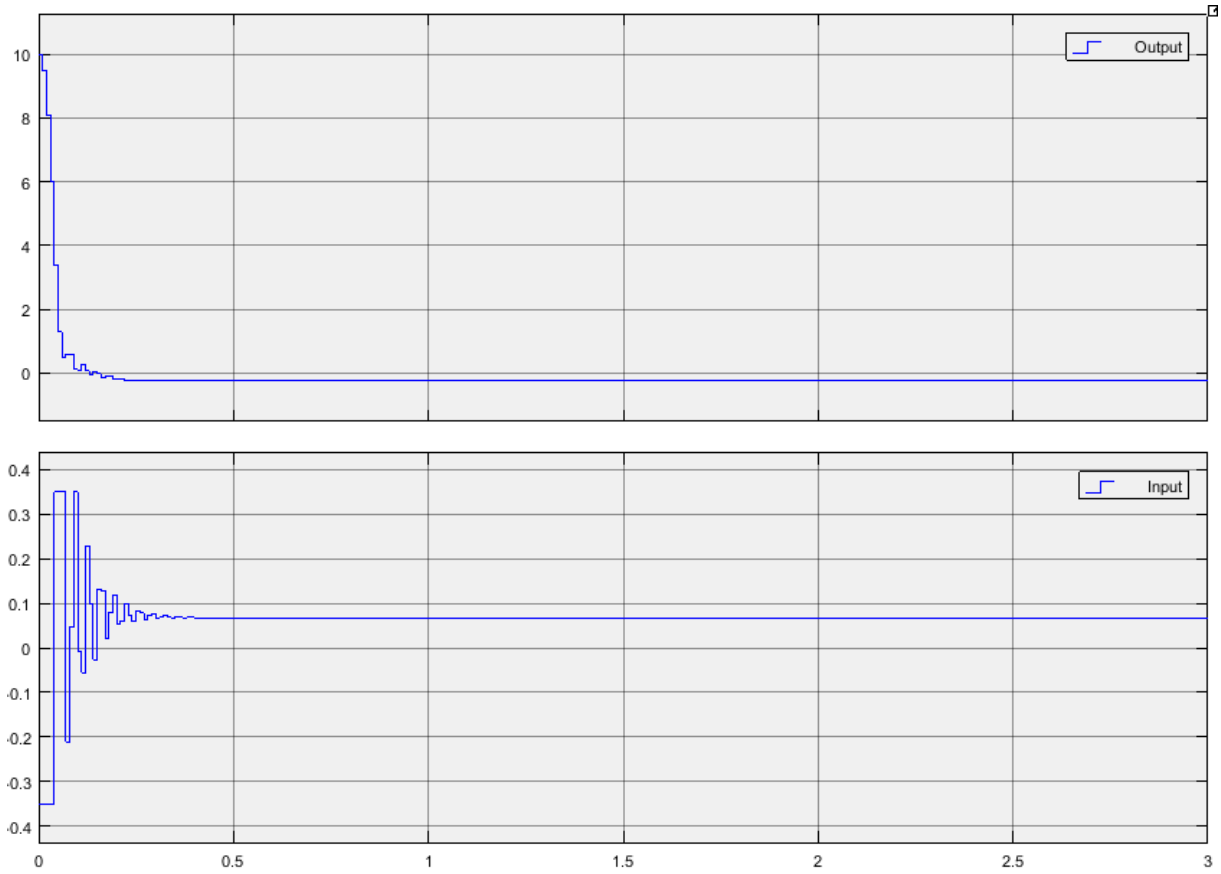


Figure 23. results of values are shown up

```
%% changable values
x0 = [ 0;0;0;50];
p_dis = 0.02; %1/R
v = 30;          %m/sec
ls = 10; % m
%%
```

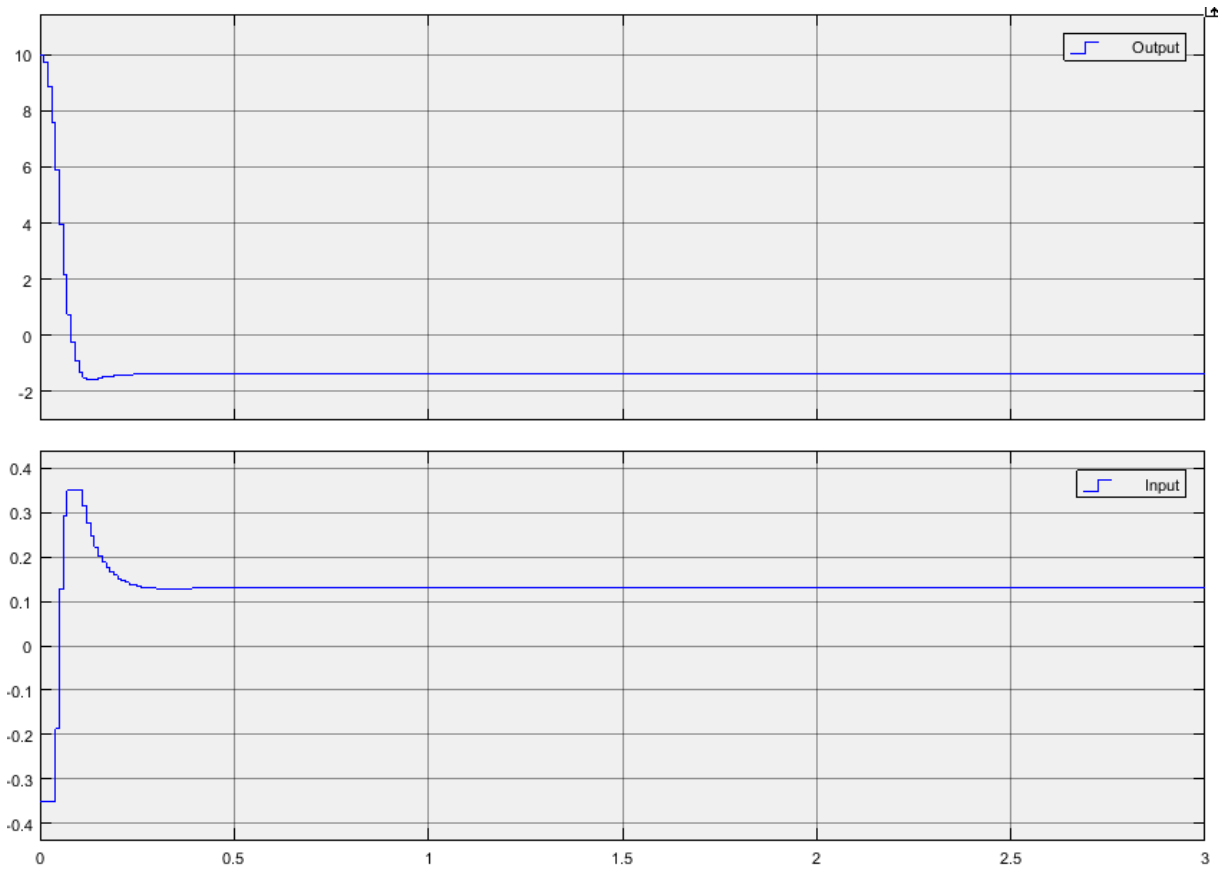


Figure 24. results of values are shown up

5. Conclusion

In this project, I have experienced MPC design with and without disturbance with Matlab and Simulink. I have observed effects of states on the motion. MPC is appropriate controller for the lane keeping system. But, according to speed and curvature MPC could be failed so that customers should not trust controller at high speed (appx. up to 50 m/s which is very high to autonomous driving). This project was very beneficial to apply basics of the MPC controller.

6. Appendix

Matlab Code

```
%% EE498 Term Project
%%
% İbrahim Güngen
```

```
% 1936939

%%
% Some figure formatting
set(groot,'defaulttextinterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex');
clc
clear all

%% Introduction
%%
%
%
%%



% System parameters
m = 2023;        %kg
lf = 1.26;       %m
cf = 2.864e5;    %N/rad
J = 6286;        %kg.m2
lr = 1.90;       %m
cr = 1.948e5;    %N/rad

%% changable values
x0 = [ 0;0;0;10];
p_dis = 0.001; %1/R
v = 30;          %m/sec
ls = 10; % m
%%


a11 = -(cf+cr)/(m*v);
a12 = -1-(cf*lf-cr*lr)/(m*v^2);
a21 = -(cf*lf-cr*lr)/(J);
a22 = -(cf*lf^2 + cr*lr^2)/(J*v);
b1 = cf / (m*v);
b2 = cf*lf / J;



% Finite Horizon MPC as quadratic program
% State-Space Model of the System
Ac = [a11 a12 0 0;
      a21 a22 0 0;
      0 1 0 0;
      v ls v 0];
Bc = [b1; b2; 0; 0];
C =  [0 0 0 1];
D =  [0];

Ts = 0.05;

%%
T = sym('T');
x = sym('x');

A_dis = expm(Ac*T);
```

```
B_dis = int(expm(Ac*x),x,[0 T])*Bc;
%%

A_dis = subs(A_dis,T,Ts);
B_dis = subs(B_dis,T,Ts);
A = double(A_dis);
B = real(double(B_dis));



% Optimal control solution for $N = 4$
G = [zeros(4,1) zeros(4,1) zeros(4,1) zeros(4,1); B zeros(4,1) zeros(4,1)
zeros(4,1); ...
    A*B B zeros(4,1) zeros(4,1); A^2*B A*B B zeros(4,1); A^3*B A^2*B A*B
B];
H = [eye(4); A; A^2; A^3; A^4];
Q = C'*C;
R = 0.001;

% Q = eye(2);
Pinf = dare(A,B,Q,R,zeros(4,1),eye(4));
Kinf = inv(R+B'*Pinf*B)*B'*Pinf*A;
% A*X*A' - X + Q = 0;  X = dlyap(A,Q)
P = dlyap( (A-B*Kinf)',Q+Kinf'*R*Kinf );
Qf = P;
Qbar = blkdiag(Q,Q,Q,Q,Qf);
Rbar = blkdiag(R,R,R,R);
M = G'*Qbar*G + Rbar;
% input bound: umin <= u <= umax
umin = -0.3491; % -20 degrees
umax = 0.3491; % 20 degrees
lb = [umin;umin;umin;umin];
ub = [umax;umax;umax;umax];
% Apply MPC steps
xVec(:,1) = x0;
yVec(1) = C*x0;
uVec = [];
for kk = 1:60
    alpha = G'*Qbar*H*xVec(:,kk);
    Usol = quadprog(M,alpha',[],[],[],[],lb,ub);
    uVec(kk) = Usol(1);
    xVec(:,kk+1) = A*xVec(:,kk) + B*uVec(kk) + [0; 0; -v*p_dis; 0];
    yVec(kk+1) = C*xVec(:,kk+1);
    Xsol(:,1) = xVec(:,kk);
    Xsol(:,2) = A*Xsol(:,1) + B*Usol(1);
    Xsol(:,3) = A*Xsol(:,2) + B*Usol(2);
    Xsol(:,4) = A*Xsol(:,3) + B*Usol(3);
    Ysol(1) = C*Xsol(:,1);
    Ysol(2) = C*Xsol(:,2);
    Ysol(3) = C*Xsol(:,3);
    Ysol(4) = C*Xsol(:,4);
end

uVec = [uVec uVec(end)];
tVec = [0:Ts:3];
%figure;
subplot(3,2,1)
stairs(tVec,uVec,'LineWidth',2);
hold all;
xlabel('time [sec]')
```

```
grid
ylabel('$u$')
title('Input $u$')

subplot(3,2,2)
stairs(tVec,C*xVec,'LineWidth',2)
hold all;
grid
xlabel('time [sec]')
ylabel('$y$')
title('Output $y$')

subplot(3,2,3)
stairs(tVec,[1 0 0 0]*xVec,'LineWidth',2)
hold all;
grid
xlabel('time [sec]')
ylabel('$x_1$')
title('State $x_1$')

subplot(3,2,4)
stairs(tVec,[0 1 0 0]*xVec,'LineWidth',2)
hold all;
grid
xlabel('time [sec]')
ylabel('$x_2$')
title('State $x_2$')

subplot(3,2,5)
stairs(tVec,[0 0 1 0]*xVec,'LineWidth',2)
hold all;
grid
xlabel('time [sec]')
ylabel('$x_3$')
title('State $x_3$')

subplot(3,2,6)
stairs(tVec,[0 0 0 1]*xVec,'LineWidth',2)
hold all;
grid
xlabel('time [sec]')
ylabel('$x_4$')
title('State $x_4$')

set(findall(gcf,'Type','line'),'LineWidth',2)
set(findall(gcf,'-property','FontSize'),'FontSize',14);
% legend('$u_{max} = 1.5$','$u_{max} = 2.5$','$u_{max} = 4$')
```