**Matlab Plant Model script:**

The paper we took for our project does not provide any information on the configuration of the vehicle which was used to simulate the results.

We have assumed the vehicle as the Car with four wheels and simulated the results. The Plant model is designed as below with the help of MATLAB coding.

function [A,B,C,D,U,Y,X,DX] = fcn(Vx,u,x)

% Sample time

Ts = 0.1;

% Model parameters

m = 1575;

Iz = 2875;

lf = 1.2;

lr = 1.6;

Cf = 19000;

Cr = 33000;

% Continuous-time model

Ac = [-(2\*Cf+2\*Cr)/m/Vx, 0, -Vx-(2\*Cf\*lf-2\*Cr\*lr)/m/Vx, 0;

0, 0, 1, 0;

-(2\*Cf\*lf-2\*Cr\*lr)/Iz/Vx, 0, -(2\*Cf\*lf^2+2\*Cr\*lr^2)/Iz/Vx, 0;

1, Vx, 0, 0];

Bc = [2\*Cf/m 0 2\*Cf\*lf/Iz 0]';

Cc = [0 0 0 1; 0 1 0 0];

Dc = zeros(2,1);

% Discretize the continuous-time model using zero-order hold

% on the inputs and a sample time of Ts seconds

nx = size(Ac,1);

nu = size(Bc,2);

M = expm([[Ac Bc]\*Ts; zeros(nu,nx+nu)]);

A = M(1:nx,1:nx);

B = M(1:nx,nx+1:nx+nu);

C = Cc;

D = Dc;

% Nominal conditions for discrete-time plant

X = x;

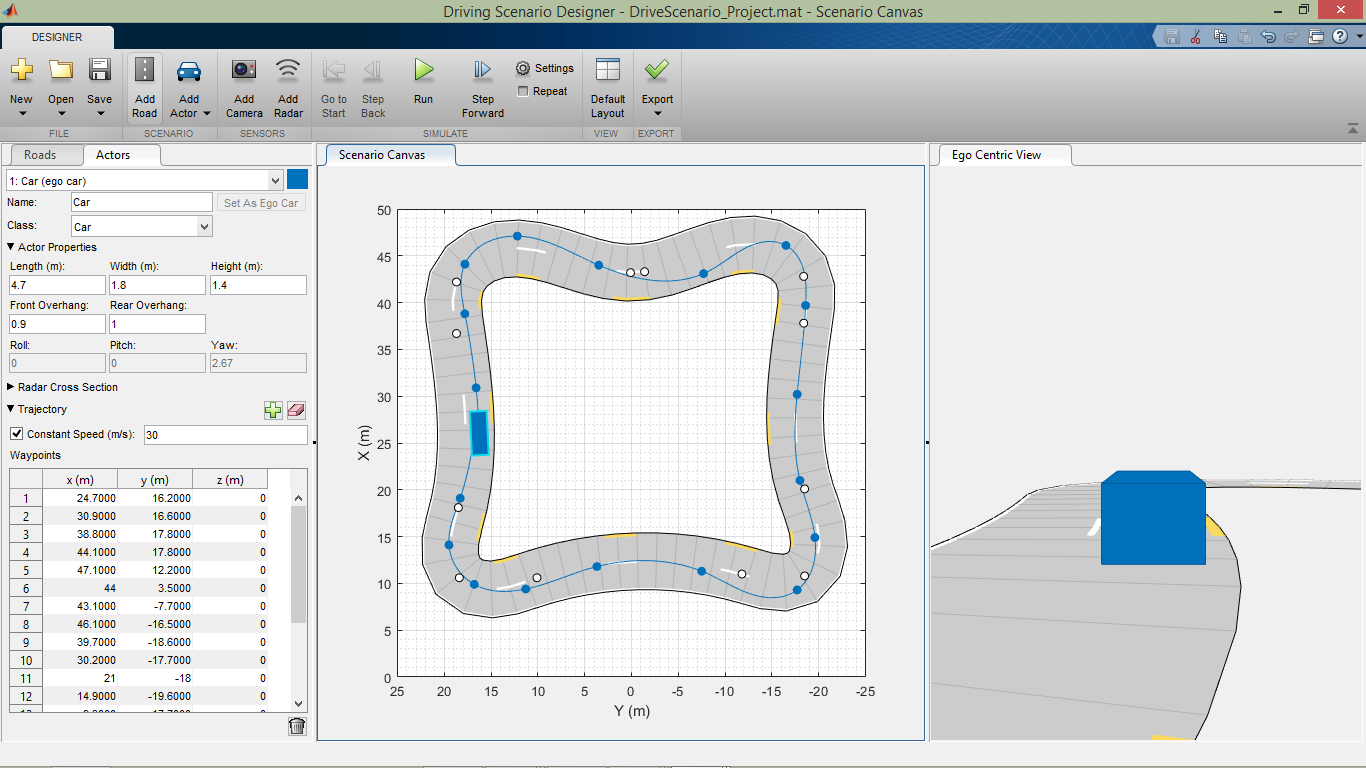
U = u;

Y = C\*x + D\*u;

DX = A\*x+B\*u-x;

Driving scenario (Tracking Path)

Also the paper did not provide the details of the type of the road on which they simulated the vehicle performance. The following Driving environment is assumed to analyse the controller and vehicle dynamic response.



function [scenario, egoCar] = DriveScenario\_Project()

% createDrivingScenario Returns the drivingScenario defined in the Designer

% Generated by MATLAB(R) 9.4 and Automated Driving System Toolbox 1.2.

% Generated on: 19-Nov-2018 20:52:41

% Construct a drivingScenario object.

scenario = drivingScenario;

% Add all road segments

roadCenters = [42.2 18.7 0;

36.7 18.7 0;

18.1 18.5 0;

10.6 18.4 0;

10.6 10.1 0;

11 -11.8 0;

10.8 -18.5 0;

20.1 -18.5 0;

37.8 -18.4 0;

42.8 -18.4 0;

43.3 -1.4 0;

43.2 0.1 0;

42.2 18.7 0];

marking = [laneMarking('Dashed', 'Color', [0.98 0.86 0.36], 'Width', 0.3)

laneMarking('Dashed')

laneMarking('Solid')];

laneSpecification = lanespec(2, 'Width', 2.925, 'Marking', marking);

road(scenario, roadCenters, 'Lanes', laneSpecification);

% Add the ego car

egoCar = vehicle(scenario, ...

'ClassID', 1, ...

'Position', [24.7 16.2 0]);

waypoints = [24.7 16.2 0;

30.9 16.6 0;

38.8 17.8 0;

44.1 17.8 0;

47.1 12.2 0;

44 3.5 0;

43.1 -7.7 0;

46.1 -16.5 0;

39.7 -18.6 0;

30.2 -17.7 0;

21 -18 0;

14.9 -19.6 0;

9.3 -17.7 0;

11.3 -7.5 0;

11.8 3.7 0;

9.4 11.3 0;

9.9 16.8 0;

14.1 19.5 0;

19.1 18.3 0;

27.9 16.4 0];

speed = 30;

trajectory(egoCar, waypoints, speed);

**Simulation Results:**

Model Predictive Controller - Performance script:

The Model Predictive controller here is used to control the vehicle response to track the reference path.

%% create MPC controller object with sample time

mpc1 = mpc(plant\_C\_2, 0.1);

%% specify prediction horizon

mpc1.PredictionHorizon = 10;

%% specify control horizon

mpc1.ControlHorizon = 3;

%% specify nominal values for inputs and outputs

mpc1.Model.Nominal.U = 0;

mpc1.Model.Nominal.Y = [0;0];

%% specify constraints for MV and MV Rate

mpc1.MV(1).Min = -0.523598775598299;

mpc1.MV(1).Max = 0.523598775598299;

mpc1.MV(1).RateMin = -0.261799387799149;

mpc1.MV(1).RateMax = 0.261799387799149;

%% specify constraints for OV

mpc1.OV(1).Min = -2;

mpc1.OV(1).Max = 6;

mpc1.OV(2).Min = -0.2;

mpc1.OV(2).Max = 0.2;

%% specify overall adjustment factor applied to weights

beta = 2.8292;

%% specify weights

mpc1.Weights.MV = 0\*beta;

mpc1.Weights.MVRate = 0.1/beta;

mpc1.Weights.OV = [1 0.1]\*beta;

mpc1.Weights.ECR = 100000;

%% specify simulation options

options = mpcsimopt();

options.RefLookAhead = 'off';

options.MDLookAhead = 'off';

options.Constraints = 'on';

options.OpenLoop = 'off';

%% run simulation

sim(mpc1, 101, mpc1\_RefSignal\_2, mpc1\_MDSignal\_2, options);

