MPC for Agile Robots

Final Project: Autonomous Cars

Overview:

In this project, we aim to design an MPC law that asymptotically stabilizes challenging trajectories for autonomous vehicles. The kinematic model is used for modeling, and the sampling time, control horizon, and P, Q, R matrices are chosen. The maximum velocity which the car can run is obtained and robustness analysis is performed.

Modeling:

The kinematic model is used:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\varphi} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{r} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} v_x \cos\varphi - v_y \sin\varphi \\ v_x \sin\varphi + v_y \cos\varphi \\ r \\ F_x/m \\ \left(\dot{\delta} v_x + \delta \dot{v}_x\right) * l_r/(l_r + l_f) \\ \left(\dot{\delta} v_x + \delta \dot{v}_x\right) * 1/(l_r + l_f) \\ \Delta\delta \end{bmatrix}$$

The state vector is given as $q := \mathrm{col}(X, Y, \varphi, v_x, v_y, r, \delta)$, and the inputs are $u := \mathrm{col}(F_x, \Delta\delta)$. X, Y, φ are the body position (X, Y) and yaw angle in global coordinates respectively. v_x and v_y are the longitudinal and lateral velocities, r is the yaw rate and δ is the steering angle. The inputs F_x and $\Delta\delta$ are the force applied by the wheels in the body direction, and the velocity of the steering angle respectively.

This model is of the form $\dot{x} = f(x, u)$.

This can be linearized and discretized at the current operating point (x_r, u_r) using Taylor Expansion and Euler's method:

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

$$\Rightarrow \dot{x} = f(x_r, u_r) + A * (x - x_r) + B * (u - u_r)$$

Where $A = \frac{\partial f}{\partial x}$ and $B = \frac{\partial f}{\partial u}$ operated at (x_r, u_r).

At time t, the operating point $(x_r, u_r) = (x(t), u(t-1))$.

Discretizing at time t with a sample time T_s, we get:

$$\frac{x(t+1) - x}{T_s} = f\big(x(t), u(t-1)\big) + A * x(t) + B * u(t) - A * x(t) - B * u(t-1)$$

$$\Longrightarrow x(t+1) = (I + A * T_s)x(t) + (B * T_s)u(t) + Ts * (f\big(x(t), u(t-1)\big) - A * x(t) - B * u(t-1))$$

Which is of the form $x(t+1) = A_d x(t) + B_d u(t) + d(t)$

Trajectory Tracking:

Taking the prediction horizon as N, the cost function is defined as:

$$\mathrm{J}_{0\to N}(x_{0,}U_{0\to N}) = \left\|x_N - x_N^{des}\right\|_P^2 + \sum_{k=0}^{N-1}\left\|x_k - x_k^{des}\right\|_Q^2 + \|u_k\|_R^2$$

subject to the constraints:

$x_{k+1} = A_d x_k + B_d u_k + d$ for k = 0, 1,…N-1

$x_{lb} \le x_k \le x_{ub}$ for k = 0, 1,…N-1

$(x_f)_{lb} \le x_N \le (x_f)_{ub}$

$u_{lb} \le u_k \le u_{ub}$ for k = 0, 1,…N-1

This is simplified as:

$$\min_{Z_{0\to N}} \left({Z_{0\to N}}^T \begin{bmatrix} \bar{Q} & 0 \\ 0 & \bar{R} \end{bmatrix} Z_{0\to N} + \mathrm{f} * Z_{0\to N} \right)$$

where $\bar{Q} = diag(Q, Q, \dots Q, P),\ \bar{R} = diag(R, \dots, R)$

$Z_{0\to N} = col(x_1, x_2 \dots x_{N,} u_0, u_1, \dots u_{N-1})$ and $\mathrm{f} = [-2(x_1^{des})^T Q, -2(x_2^{des})^T Q, \dots . -2(x_N^{des})^T P, 0,0 \dots .0]$

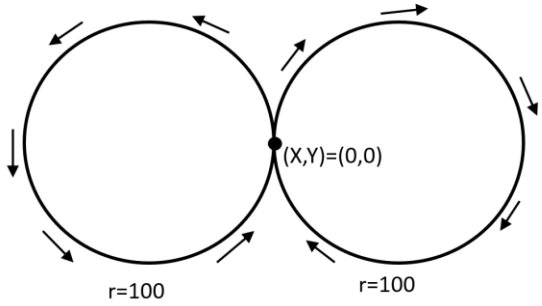
The state equations and the constraint equations can be written in matrix form as:

$$\begin{bmatrix} \mathbf{I}_{n_x} & 0 & \dots & 0 & 0 & -\mathbf{B} & 0 & \dots & 0 \\ -\mathbf{A} & \mathbf{I}_{n_x} & \dots & 0 & 0 & 0 & -\mathbf{B} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -\mathbf{A} & \mathbf{I}_{n_x} & 0 & 0 & \dots & -\mathbf{B} \end{bmatrix} [\mathbf{Z_{0\to N}}] = \begin{bmatrix} \mathbf{A}x_0 + \mathbf{K} \\ \mathbf{K} \\ \vdots \\ \mathbf{K} \end{bmatrix}$$

$$\begin{bmatrix} I_{2N*(n_x+n_u)} \\ -I_{2N*(n_x+n_u)} \end{bmatrix} [Z_{0\to N}] = \begin{bmatrix} (x_{ub})_{N-1*1} \\ (x_f)_{ub} \\ (u_{ub})_{N*1} \\ -(x_{lb})_{N-1*1} \\ -(x_f)_{lb} \\ -(u_{lb})_{N*1} \end{bmatrix}$$

Where K is the $d(t)$ term, n_x is the number of states (7 here), n_u is the number of inputs (2 here), and N is the control horizon.

A periodic eight trajectory is chosen as the trajectory to be followed as shown in the figure below.



A function was created to generate the set tracking points, when given the radius and velocity. This gives us the X_{ref} and Y_{ref}. The references of the other states were considered to be 0, but their corresponding weights in the Q and P matrices of our cost function were taken to be very low, so that their errors largely didn't affect the cost function

The state constraints were taken as:

$$\begin{bmatrix} -300 \\ -200 \\ -50 \\ -20 \\ -20 \\ -20 \\ -20 \end{bmatrix} \leq q \leq \begin{bmatrix} 300 \\ 200 \\ 50 \\ 20 \\ 20 \\ 20 \\ 20 \end{bmatrix}$$ and for terminal region: $$\begin{bmatrix} -300 \\ -200 \\ -50 \\ -20 \\ -20 \\ -20 \\ -20 \end{bmatrix} \leq q \leq \begin{bmatrix} 300 \\ 200 \\ 50 \\ 20 \\ 20 \\ 20 \\ 20 \end{bmatrix}$$

And the input constraints as: $\begin{bmatrix} -100 \\ -10 \end{bmatrix} \leq u \leq \begin{bmatrix} 100 \\ 10 \end{bmatrix}$.

We have considered the following parameters:

Sampling Time (Ts) = 0.01 s

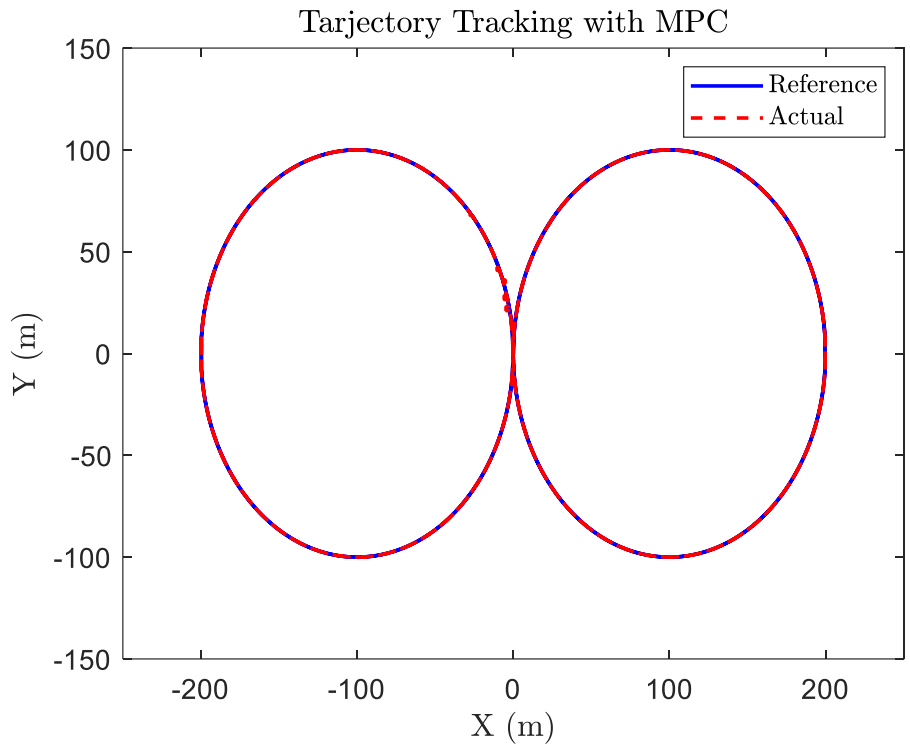
N = 20

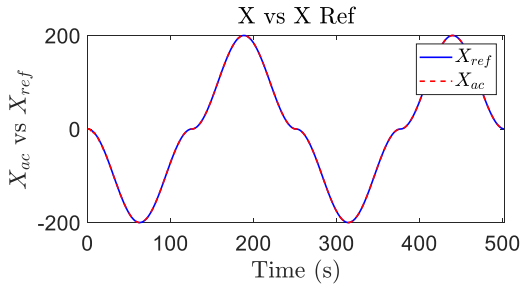
P = diag ([400 400 0.001 0.001 0.001 0.001 0.001])

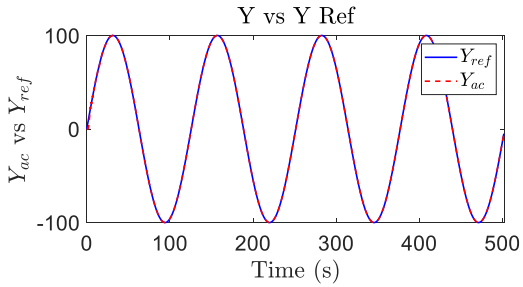
Q = diag ([1000 1000 0.001 0.001 0.001 0.001 0.001])

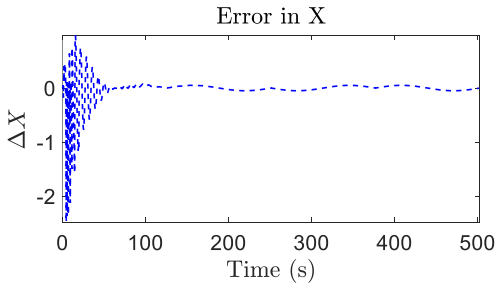
R = $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

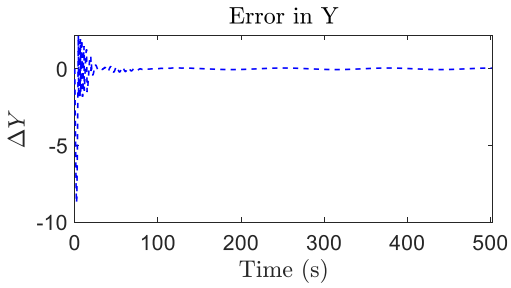
This path was aimed to follow with a speed of 5 m/s.

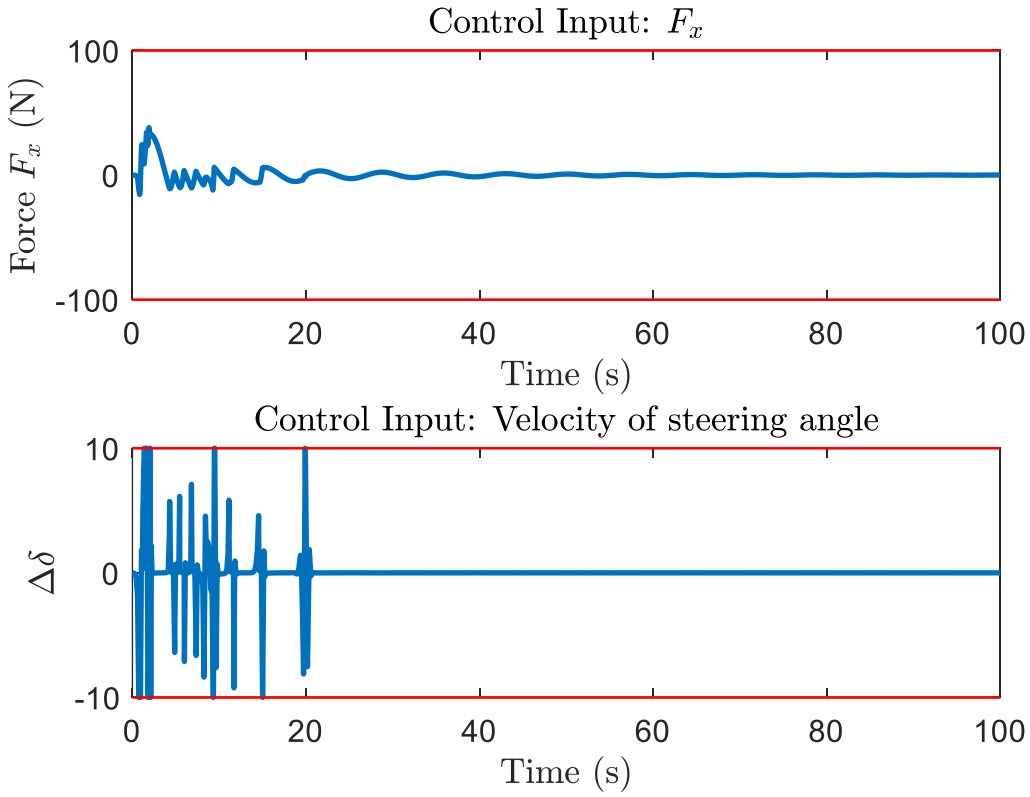

Tarjectory Tracking with MPC
Reference
Actual
Y (m)
X (m)
150
100
50
0
-50
-100
-150
-200
-100
0
100
200


X vs X Ref
X_{ref}
X_{ac}
X_{ac} vs X_{ref}
Time (s)
200
0
-200
0
100
200
300
400
500


Y vs Y Ref
Y_{ref}
Y_{ac}
Y_{ac} vs Y_{ref}
Time (s)
100
0
-100
0
100
200
300
400
500


Error in X
ΔX
Time (s)
0
-1
-2
0
100
200
300
400
500


Error in Y
ΔY
Time (s)
0
-5
-10
0
100
200
300
400
500



Maximum Velocity:

We keep the condition as: if the error in either X or Y is > 10m, then it is not tracking properly. Basis this condition, we can achieve a max velocity of 6 m/s.

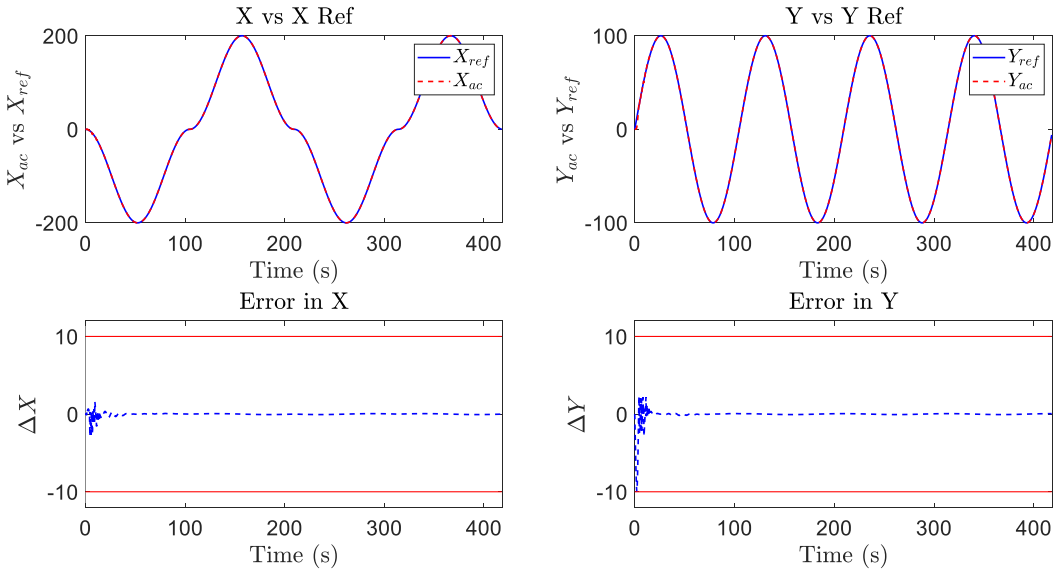


Fig: Trajectory tracking and error plots for 6 m/s

But if we consider the condition that the error converges to 0, the maximum velocity achieved is 14.9 m/s

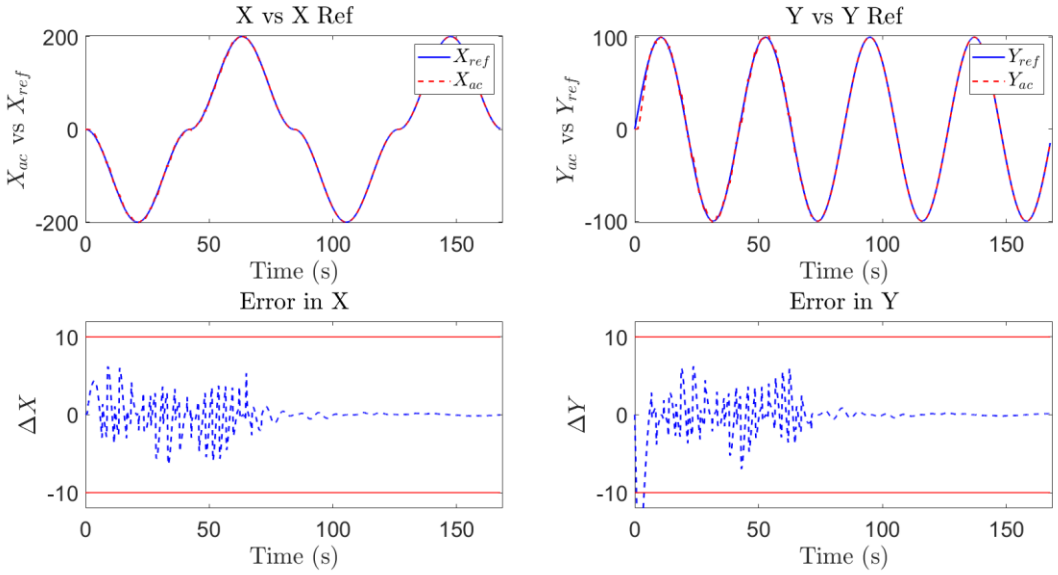


Fig: Trajectory tracking and error plots for 14.9 m/s

In the plot below, we can observe that at 15 m/s, the error doesn't converge to 0 and the car drives off the track.

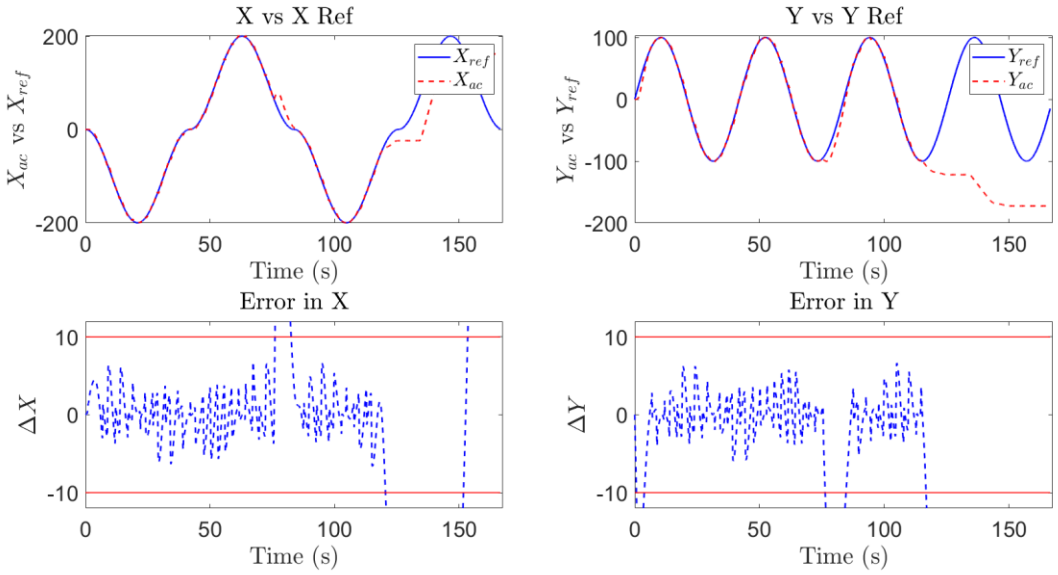


Fig: Trajectory tracking and error plots for 15 m/s

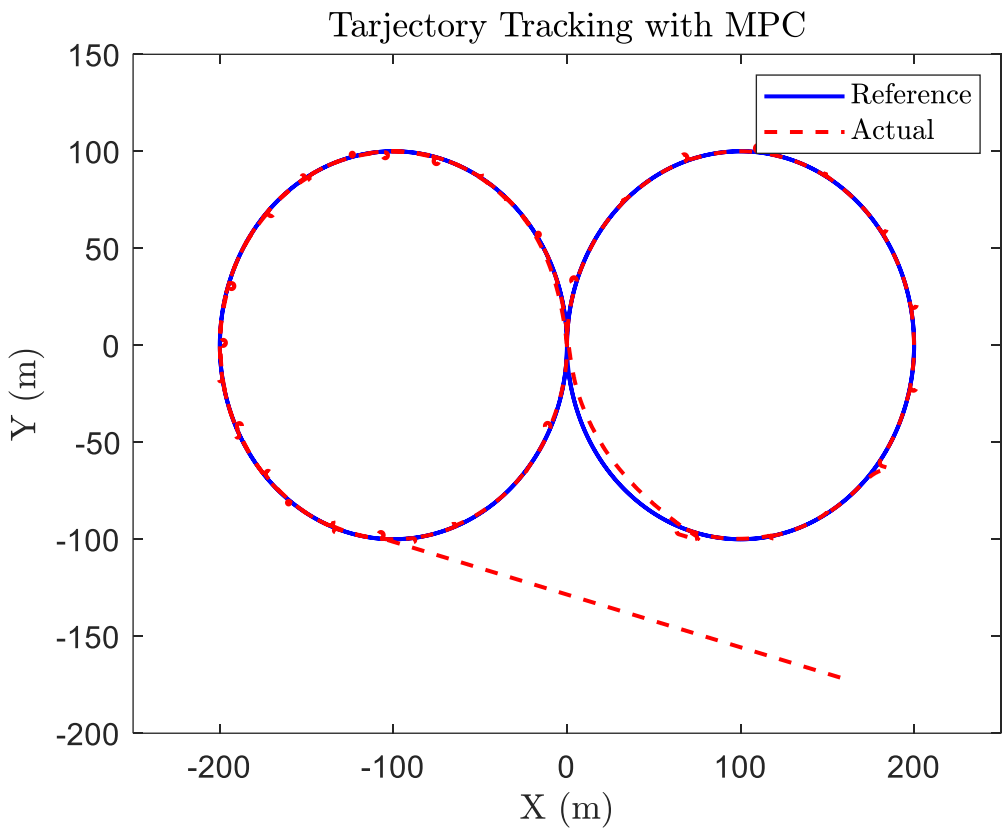


Fig: Trajectory tracking 15 m/s. The car goes off path

Robustness Analysis

The robustness analysis is done at the velocity of 5 m/s. The mass was changed to observe if the car was able to track with the original model implemented to MPC

Upper Bound

Considering the maximum error of 10m, the tracking is good till 46% change in mass.

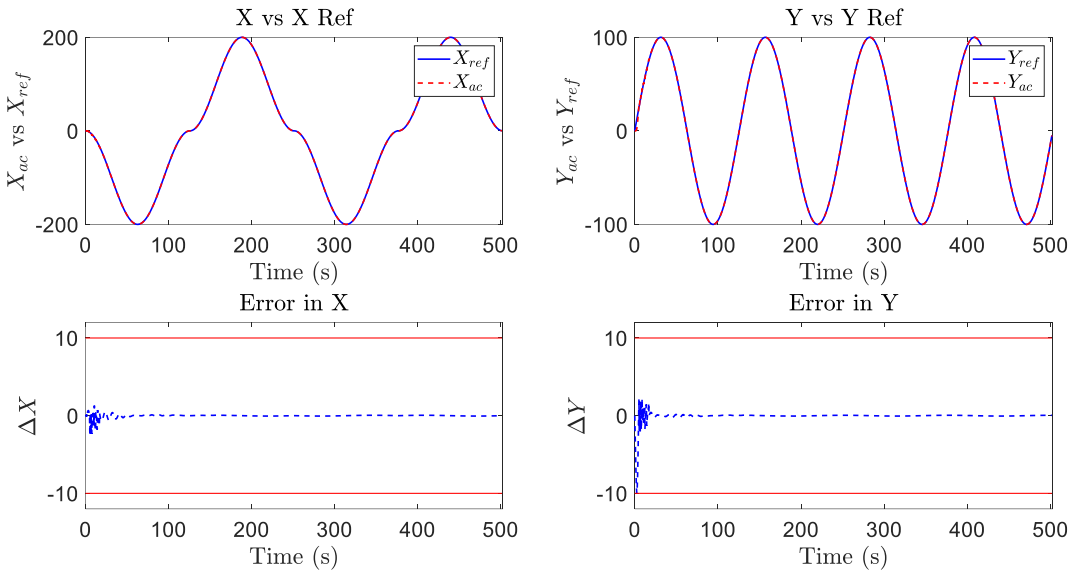


Fig: Trajectory tracking 5 m/s for 46% change in mass

Considering error convergence and input signal convergence, the maximum change in mass ~250%

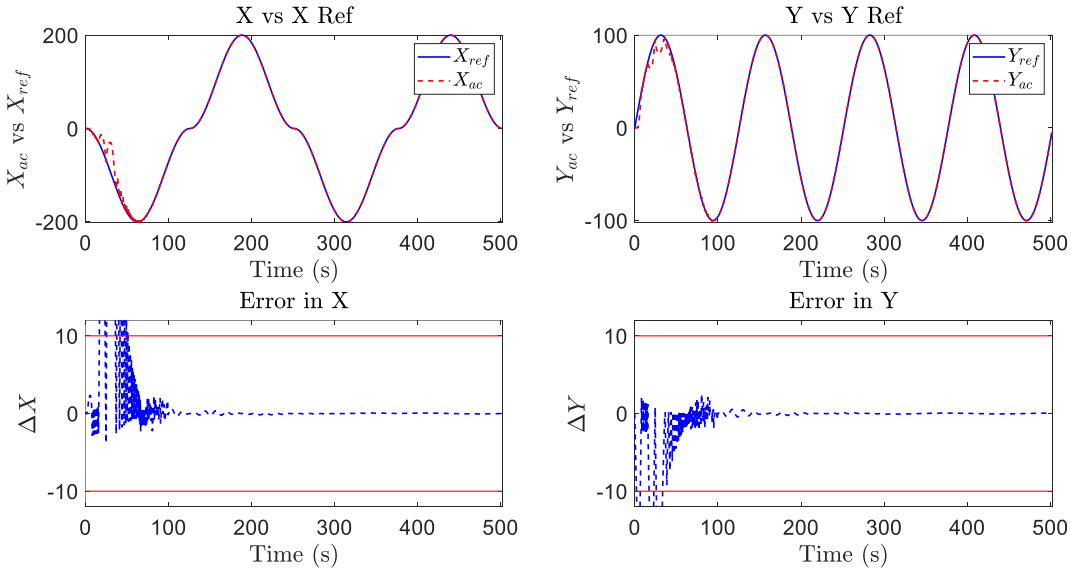


Fig: Trajectory tracking 5 m/s for 250% change in mass

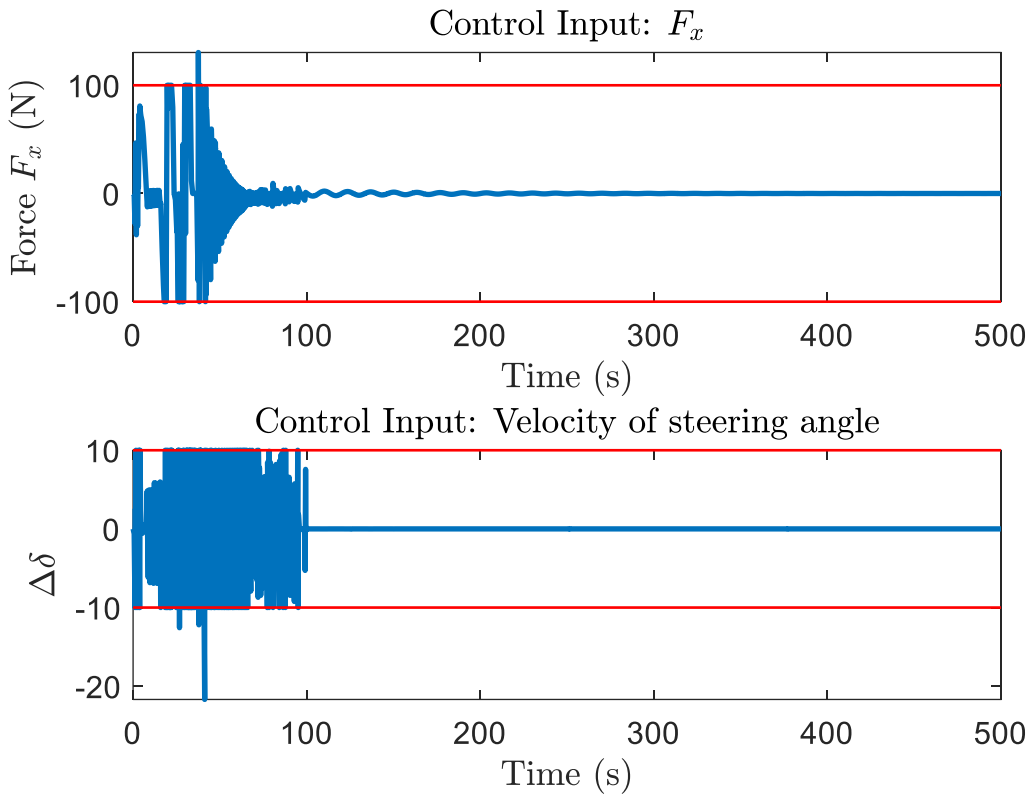


Fig: Control input for 5 m/s for 250% change in mass

Below, we can see that the car goes off path for 260% increase in mass.

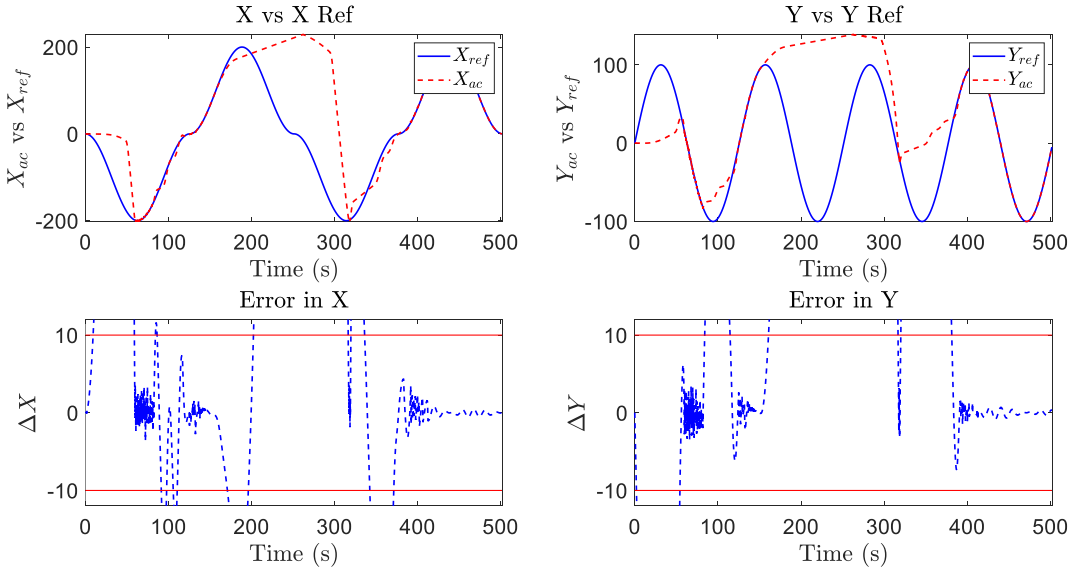


Fig: Trajectory tracking 5 m/s for 260% change in mass. The car goes off path

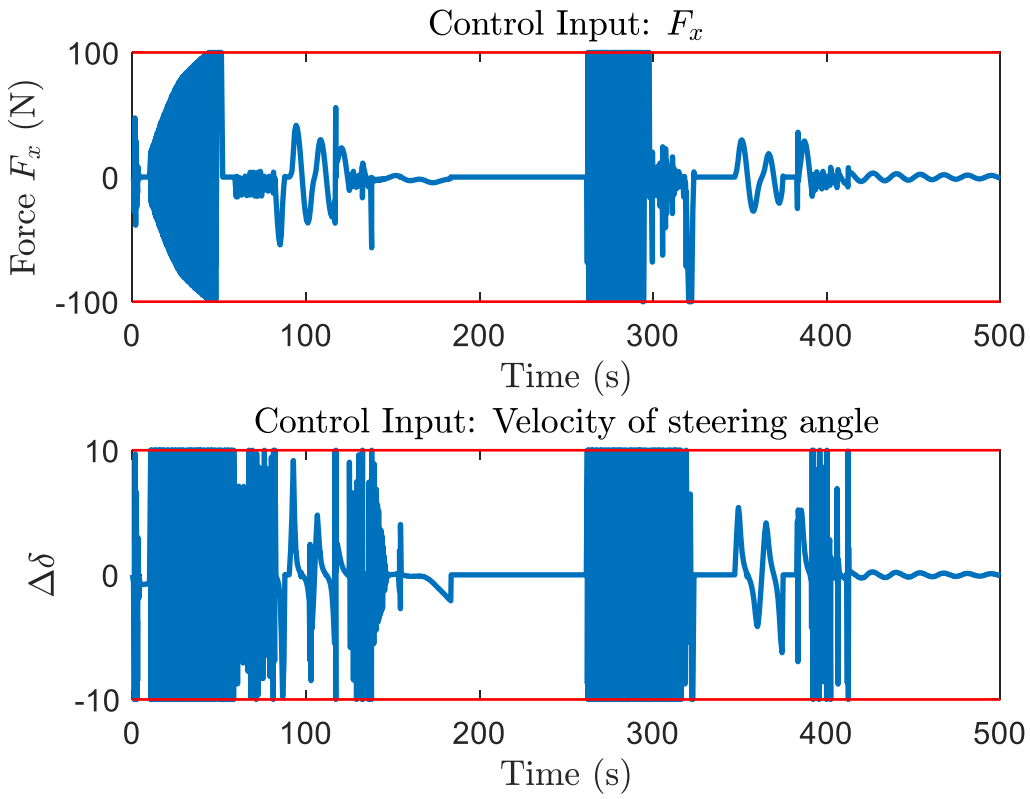


Fig: Control input for 5 m/s for 260% change in mass

Lower Bound

As we decrease the mass, the model initially starts performing better, but at 97% decrease in mass, the input goes out of the feasible set.

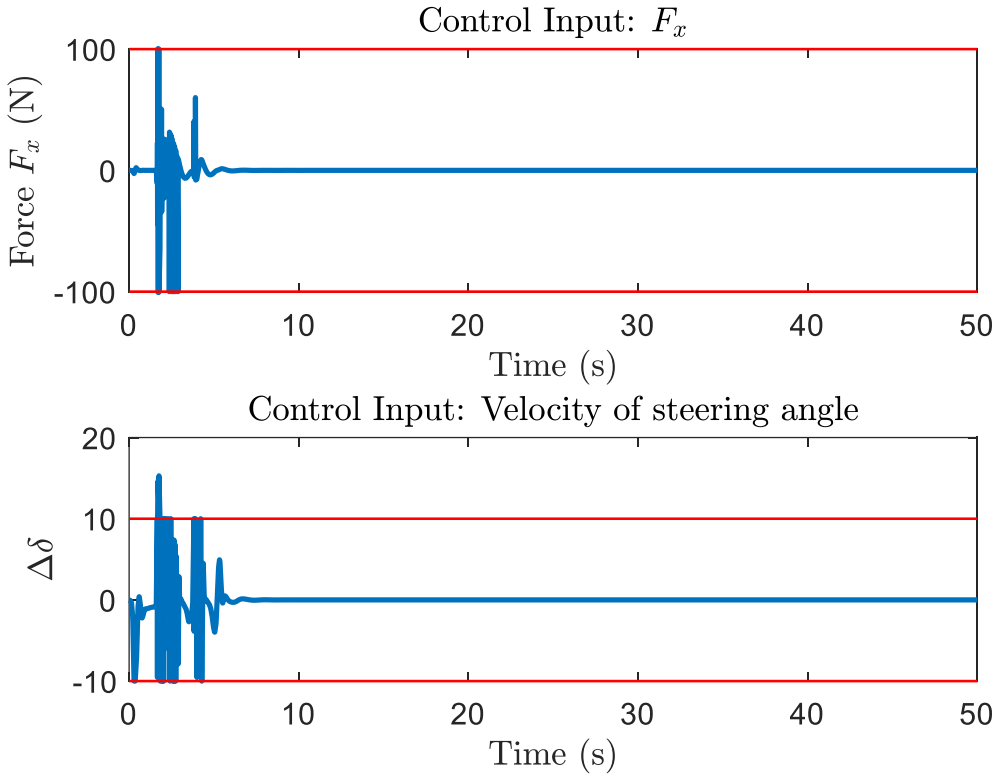
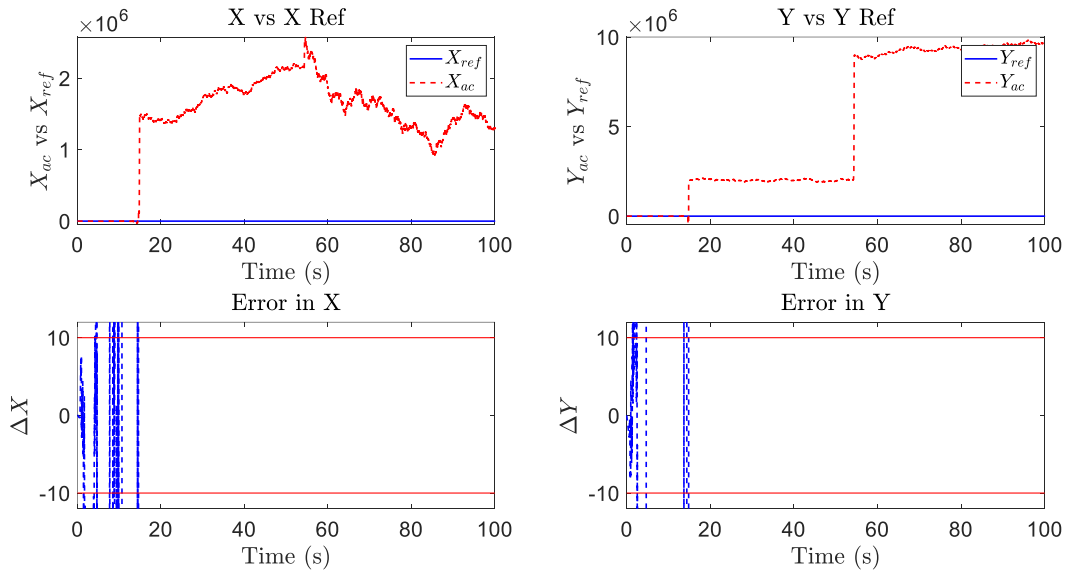


Fig: Control input for 5 m/s for -93% change in mass

At ~99.6%, the error becomes very large and doesn't converge.



Sin Trajectory

The sin trajectory is also tried out. The MPC parameters were kept same except the sample time, which was taken to be 1 s.

The velocity taken is 1 m/s