Final Project

Anshul Nayak

May 2023

1 Introduction

In the current project, we design a control law for full state kinematic bicycle model. The objective of the project is to *track a reference trajectory provided the non-linear dynamics of the model.* The non-linear dynamics for the kinematic bicycle model especially for slow velocities can be represented as:

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{\psi} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{r} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} v_x\, cos(\psi) - v_y\, sin(\psi) \\ v_x\, sin(\psi) - v_y\, cos(\psi) \\ r \\ \frac{1}{m} F_x \\ (\dot{\delta} v_x + \delta \dot{v}_x) \frac{l_R}{l_R + l_F} \\ \dot{\delta} v_x + \delta \dot{v}_x) \frac{1}{l_R + l_F} \\ \triangle \delta \end{bmatrix} \tag{1}$$

The kinematic model has 7 controllable states, $\mathcal{X} = \begin{bmatrix} X; Y; \psi; v_x; v_y; r; \delta \end{bmatrix}$ such that $\mathcal{X} \in \mathbb{R}^{7\times 1}$. Here, X, Y represent the position, ψ represents the yaw angle, v_x, v_y represent velocity, r is yaw rate while $\triangle\delta$ is rate of change of steering angle respectively. The control input for the system are $\mathcal{U} = \begin{bmatrix} F_x; \triangle\delta \end{bmatrix}$ $\{\mathcal{U} : \mathcal{U} \in \mathbb{R}^{2\times 1}\}$.

Parameters	Description	values
m	mass of the car	4.78 kg
l_r	C.G to rear axle	0.18 m
l_f	C.G to front axle	0.18 m
I_z	Rotational moment of inertia for car	0.0665 $kg.m^2$
L	Distance between center of mass of body and wheel axis	1 m

Table 1: List of constants to describe the kinematic bicycle model

2 Methods

2.1 Trajectory Generation

Initially, we generate the desired trajectory that the kinematic bicycle model needs to follow. For the current research, we consider three trajectories namely a figure of 8 with length 48 m and height 24 m, an astroid with 20 m arc length and a rhodonea curve with 5 petals with each petal having an arm extension, a = 20m respectively (Figure 1).

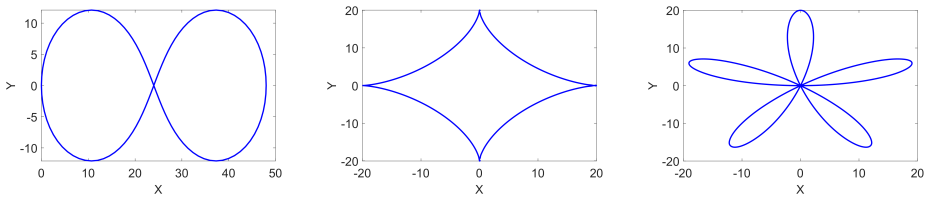

Figure 1: Three trajectories (a) 8 Figure (b) Astroid (c) Rhodonea curve with 5 petals are considered for tracking.

For the generation of each desired trajectory, we use the ***'waypoint trajectory generator'*** *MATLAB* function. The function syntax takes sampled coordinates (c) along the curve, time of arrival(toa) at each coordinate and sampling frequency(F_s) as inputs to generate the desired trajectory.

$$traj = \textbf{\textit{waypointTrajectory}}(c, toa, F_s)$$

Further, we can use the ***lookuppPose*** function at each sampling time to obtain the desired states,

$$\mathcal{X}_{des} = \left[X; Y; \psi; v_x; v_y; r; \delta\right]_{des} = \textbf{\textit{lookuppPose}}(traj, Ts)$$

The $\mathcal{X}_{des}$ will be our reference trajectory at each time while solving the MPC.

2.2 MPC Formulation

We use Euler equation to discretize the continuous non-linear dynamics into discrete state using sampling frequency, $F_S = \frac{1}{T_s}$. Our discrete set of state equations have been shown below.

$$f_d(x,u) = \begin{bmatrix} X \\ Y \\ \psi \\ v_x \\ v_y \\ r \\ \delta \end{bmatrix} + \begin{bmatrix} v_x\, cos(\psi) - v_y\, sin(\psi) \\ v_x\, sin(\psi) - v_y\, cos(\psi) \\ r \\ \frac{1}{m} F_x \\ (\dot{\delta} v_x + \delta \dot{v}_x) \frac{l_R}{l_R + l_F} \\ \dot{\delta} v_x + \delta \dot{v}_x) \frac{1}{l_R + l_F} \\ \triangle \delta \end{bmatrix} \tag{2}$$

We compute the Jacobian of the discrete state, $f_d(x(t), u(t))$ at the equilibrium point, (x,u) = (x_0, u_0) to compute the state matrices

$$A_d = \frac{\partial f_d}{\partial x}(0,0) \quad B_d = \frac{\partial f_d}{\partial u}(0,0) \tag{3}$$

x^1_{des}, u_0 are randomly initialized at the beginning. Since, we linearize the non-linear dynamics to compute A_d and B_d, we usually ignore the higher order non-linear terms. But, in the current formulation, we consider the non-linear terms as

$$d = x_{new} - A_d\, x_0 - B_d\, u_0 \tag{4}$$

All the above parameters, A, B_d, d will remain constant within the control loop of MPC formulation. Meanwhile, for each time sample, the above parameters will be modified based on the updated state.

We define our cost function as a 2-Norm receding horizon control problem (5). The state evolves according to full-state non-linear dynamics(6) and is subject to the equality constraints (8) for each state and control input.

$$J_{0\to N}(x_0, U_{0\to N}) = (x_N - x_N^{des})^T \mathrm{P} (x_N - x_N^{des}) + \sum_{k=0}^{N-1} (x_k - x_k^{des})^T \mathrm{Q} (x_k - x_k^{des}) + \sum_{k=0}^{N-1} u_k^T \mathrm{R} u_k \tag{5}$$

S.t.

$$\mathcal{X}_{t+1} = \mathcal{X}_t + \int_t^{t+T_s} f_c(x(t), u(t)) \tag{6}$$

We use the *MATLAB ode45* solver to integrate the non-linear dynamics (6) and then sample at the next sample time to obtain the full-state non-linear dynamics. As, we have a desired trajectory for tracking, the cost function in the quadratic program formulation will be a combination of quadratic and linear terms.

$$f(z) = min \frac{1}{2} z^T \mathrm{H} z + f^T z$$

$$H = \begin{bmatrix} \bar{Q} & 0 \\ 0 & \bar{R} \end{bmatrix} \quad f = \begin{bmatrix} -2\,\bar{Q}\,\mathcal{X}^{des} \end{bmatrix} \tag{7}$$

We formulate the MPC solver as a quadratic program where we solve by minimising the cost function at each time step subject to equality constraints in

discrete form as below.

$$\begin{bmatrix} I & 0 & \dots & 0 & 0 & -B & 0 & \dots & 0 \\ -A & I & \dots & 0 & 0 & 0 & -B & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -A & I & 0 & 0 & \dots & -B \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ u_0 \\ \vdots \\ u_{N-1} \end{bmatrix} = \begin{bmatrix} Ax_0 + d \\ d \\ \vdots \\ d \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} I & 0 & \dots & 0 & 0 & -B & 0 & \dots & 0 \\ -A & I & \dots & 0 & 0 & 0 & -B & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -A & I & 0 & 0 & \dots & -B \end{bmatrix} \begin{bmatrix} X_{1\to N} \\ U_{0\to N-1} \end{bmatrix} = \begin{bmatrix} Ax_0 + d \\ d \\ \vdots \\ d \end{bmatrix}$$

$$\Rightarrow A_{eq} \begin{bmatrix} X_{1\to N} \\ U_{0\to N-1} \end{bmatrix} = B_{eq} \tag{8}$$

We solve for the optimal control, $U_{MPC},\ t = 0, 1, ...$ at each state and then, compute the next state based on non-linear state dynamics for a closed loop-system. For the MPC problem, we consider the control parameters, P $\in \mathbb{R}^{7\times7}$, Q $\in \mathbb{R}^{7\times7}$ and R $\in \mathbb{R}^{2\times2}$ as below.

Parameters	value
T_s	0.05 s
F_s	20 Hz
$T_{samples}$	100* F_s
N	10
$\mathcal{X}_0$	$\mathcal{X}_1^{des}$
P	col(1000,1000,1,200,200,1,1)
Q	col(1000,1000,1,200,200,1,1)
R	col(10,10)

Table 2: Parameters for MPC

3 Results

3.1 Figure of 8 Trajectory

We show the results for tracking on the figure of 8 curve using MPC. We show the evolution of seven states $\mathcal{X}_{MPC} = \left[X; Y; \psi; v_x; v_y; r; \delta\right]$ as well as the control input, $\mathcal{U} = \left[F_x; \triangle\delta\right]$ with time.

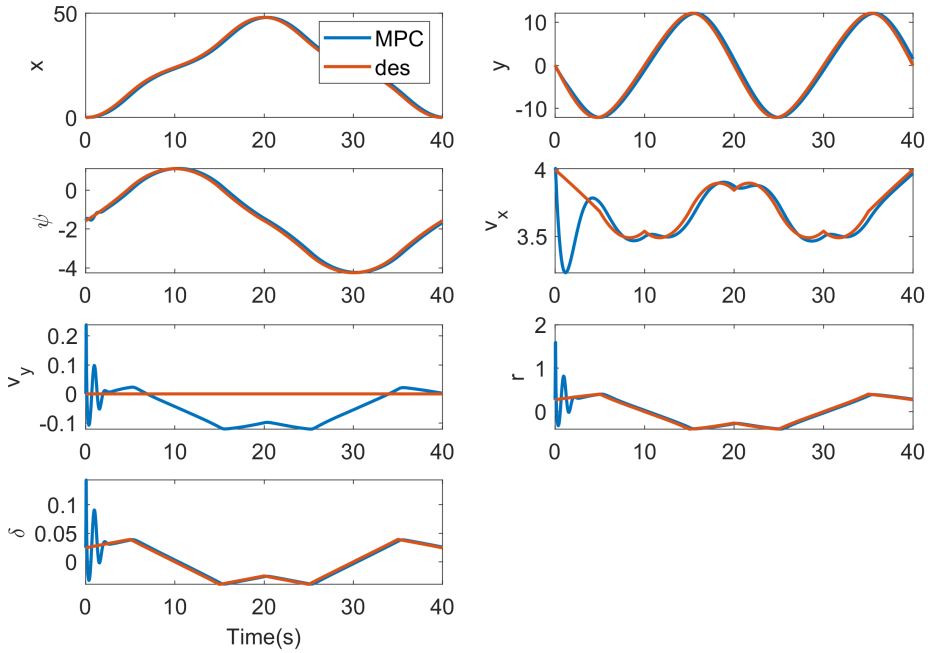

Figure 2: Evolution of states with time for 2-Norm MPC for the figure of 8 trajectory.

Figure 2 shows the evolution of all 7 states and with time, all states match the reference trajectory. For states like, v_x, r, δ, the states oscillate initially, however, with time, follow the desired states. The control input, $u(t)$ increases to -3N and later stabilizes to almost 0N. Meanwhile, the rate of steering angle, $\triangle\delta$ is initially high up to $\pm 2\pi$ but then decreases to zero (Figure 3a). The tracking video for figure of 8 trajectory can be found here which the car tries to cover in 40 seconds.

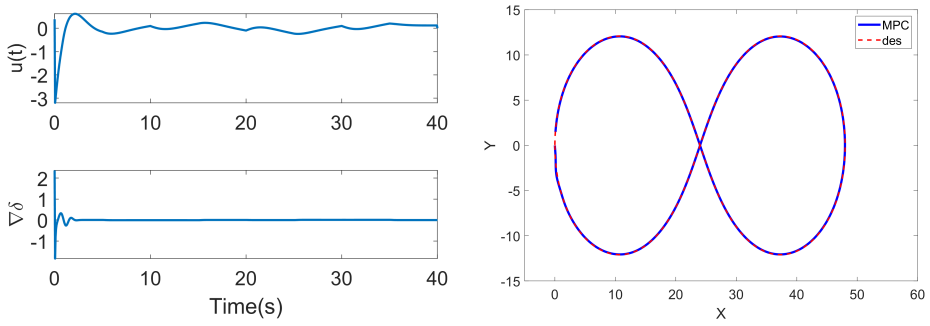

Figure 3: (a) Evolution of control input with time (b) comparison of MPC generated optimal trajectory and desired path

3.2 Astroid Trajectory

Below, we show the trajectory tracking using MPC for the Astroid trajectory. As discussed earlier, the trajectory has an arc length of 20 m. *Unlike, the previous smooth 8-figure trajectory, the current Astroid trajectory has sharp corners and it would be interesting to observe how the MPC behaves?*

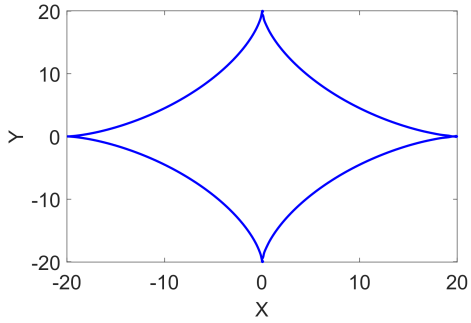

Figure 4: Atroid Trajectory with arc length of 20m.

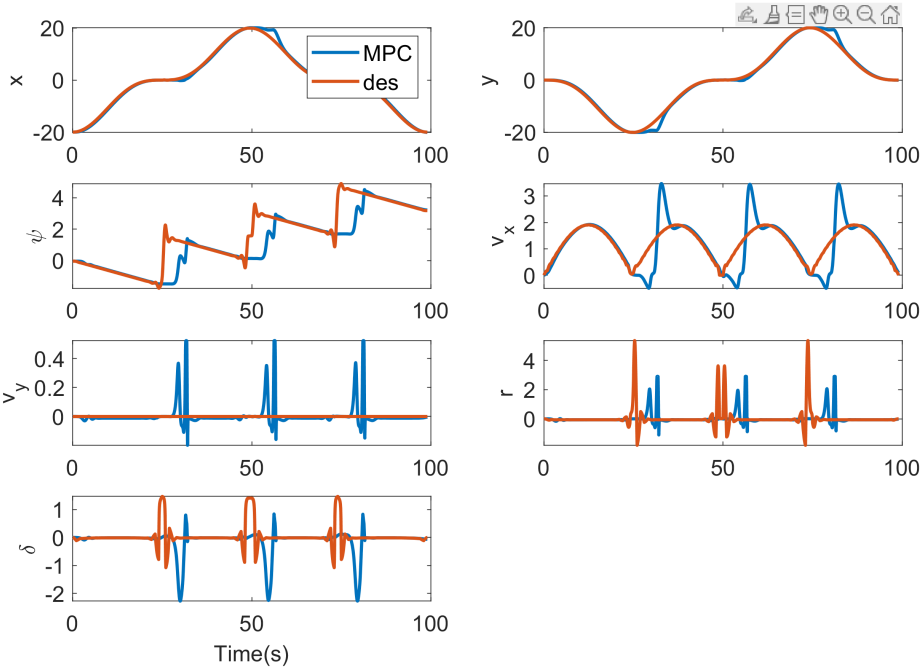

Figure 5: Evolution of states for the Astroid curve using 2-Norm MPC

In Figure 5, the states x and y are tracked effectively using the MPC controller and the states match the desired trajectory. However, it was interesting to observe, the MPC was unable to capture v_x properly even though the desired

state was smooth. Further, a lag was observed during tracking for yaw angle, ψ and angular velocity, r.

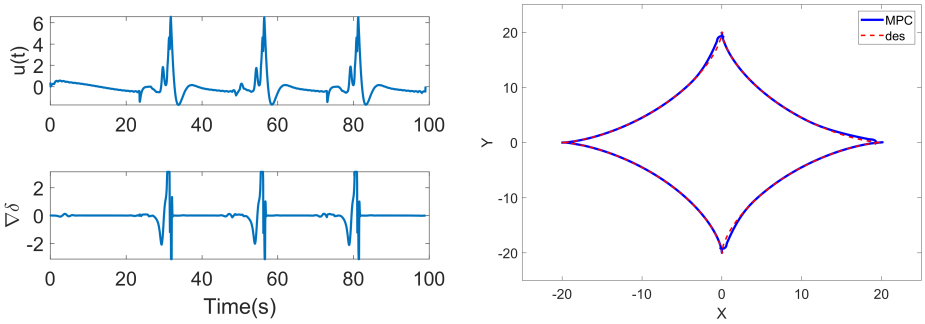

Figure 6: (a) Evolution of control input with time (b) comparison of MPC generated optimal trajectory and desired path

In Figure 6a, we show the evolution of control inputs with time. *The control input to the system was very high near the vertex of the curve* where the car has to change directions. We observe $u(t)$ as high as 6N and similarly the value of $\triangle\delta$ was also high showing the car has to make rapid changes to the steering angle to make such sharp turns. Figure 6b shows the comparison between actual and desired path for the Astroid curve. A video highlighting the desired tracking for the Astroid curve has been provided.

3.2.1 Minimum Time Period

Q. Show when the MPC fails to follow the desired path, $\mathcal{X}^{des}$? And, what is the minimum possible time period attained for traversing the above trajectory successfully?

To address this question, we consider three time periods, T = 10, 20 and 80 seconds for the car to cover the length of the trajectory using MPC. We kept the upper and lower bound of the constraints the same for all three cases. The upper and lower bound were decided based on the physical limit of the specified UGV having mass m = 4.8 kg. The maximum velocity along x-direction was restricted to 5 m/s owing to the small size of the car. Similarly, the control input was restricted to $\pm$30 N. Further, the angular states like yaw angle ψ, angular velocity r and steering angle δ were limited to $\pm 2\pi$.

Having formulated the above parameters, we run MPC in a loop for the three considered times, T = 10, 20, and 80 seconds. We compare the trajectories generated using MPC for the three time period with the desired trajectory (Figure 7). The plot reveals at about T = 80 seconds, the car is able to track the desired trajectory effectively using MPC and with a time period less than that the MPC fails to track. As can be seen for T = 10 and 20 seconds, the

generated trajectories using MPC are incomplete and not accurate. A video showing the failure during tracking has been shown for T = 20 seconds.

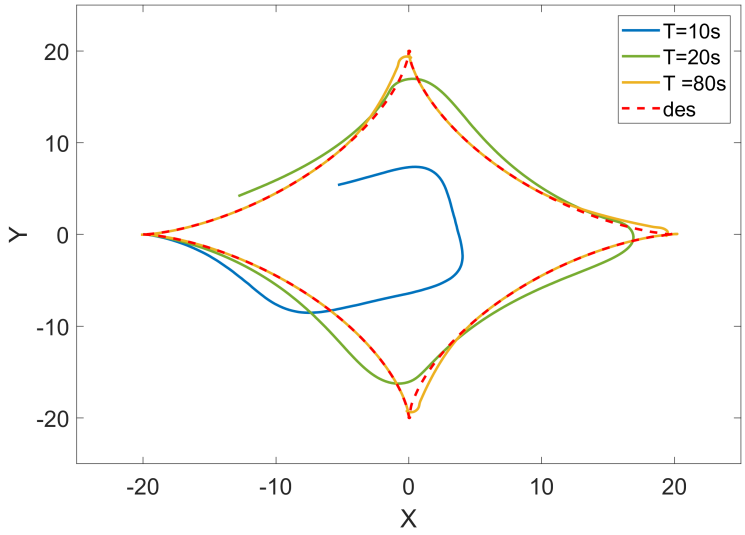

Figure 7: Trajectory tracking failure for the Astroid curve with specified time period.

3.3 Robustness Analysis

In the robustness analysis, we change the parameters like mass m and length of the axle $\{l_r, l_f\}$ and check how robust the MPC is to such changes.

In order to achieve this, we change the mass of the system. We considered three variations for mass, M = [0.5, 2, 7.5]m_0, where m_0 = 4.78 Kg was the original mass. Here, lower mass signifies stripping down extra weight while higher values of mass, M signifies adding extra payload to the system. The maximum allowable controllable input, $u(t) = \pm 10N$ while the allowable speed was 5 m/s. Rest all parameters were kept similar for tracking the Astroid curve.

Interestingly, the MPC optimal controller was robust for all the cases (Figure 8a) even upto carrying as high as 6.5 times m_0 as payload. However, for the highest mass, the car does some weird spiral behavior. And, beyond the peak value of M = 7.5 m_0, the QP solver failed to obtain optimal solution at each time under the provided state and control constraints. Meanwhile, varying the distance between C.G and axle, $\{l_f, l_r\}$ shows very interesting results, especially with increase in the values of $l_f and l_r$. We considered three variations in length, $l_f = l_r = [0.09, 0.36, 0.54]$ metres. The original length of the axle was 0.18m. Lowering each axle length to 0.09 m had negligible effect on the tracking the model was fairly robust. However, increasing the distance of l_F and l_r had

significant effects. Especially, midway, the car is unable to track the exact trajectory and veers of the course for some time again retracking the path.

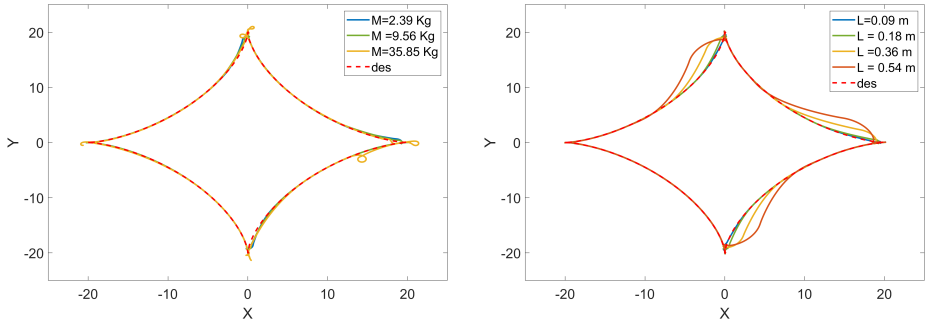

Figure 8: Robustness of MPC controller with respect to (a) mass (b) length of the axle.

4 Appendix

In the appendix, we show the trajectory tracking with the evolved states and control input for tracking a Rhodonea curve with 5 petals and each petal having a length of 20m. The specified time period is 100 seconds. A video showing the tracking for the curve can be found here.

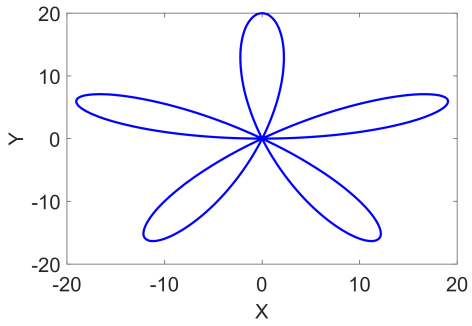

Figure 9: (a) Rhodonea (Rose) curve with 5 petals and each petal having a radius of 20m.

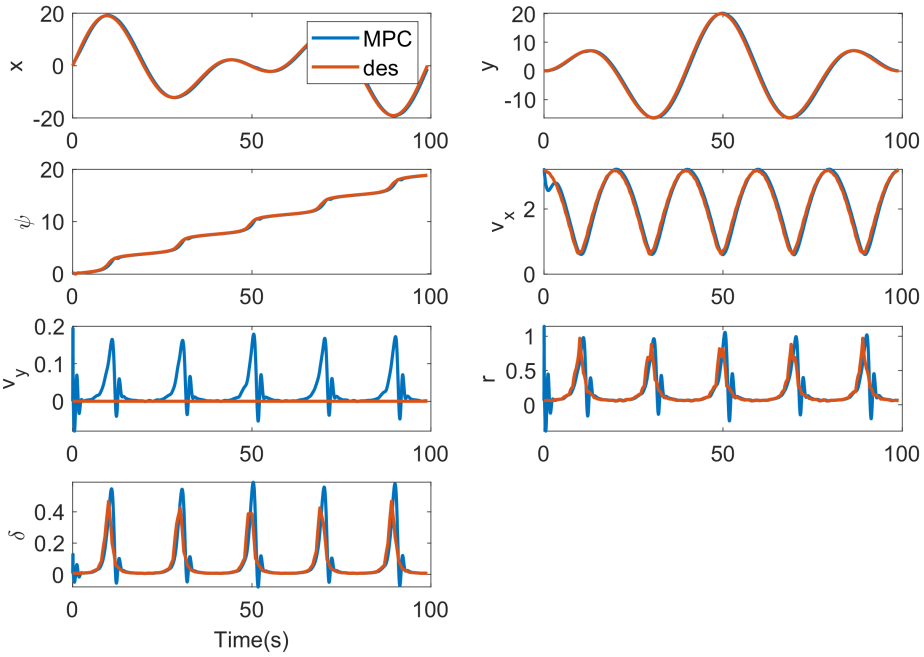

Figure 10: Evolution of states with time for 2-Norm MPC for the Rhodonea curve with 5 petals.

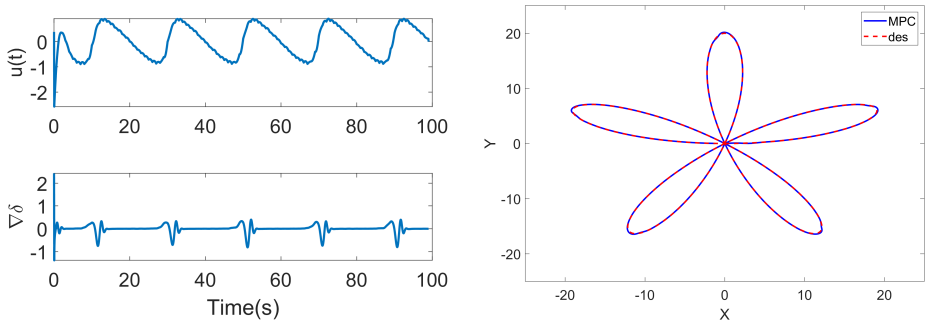

Figure 11: (a) Evolution of control input with time (b) comparison of MPC generated optimal trajectory and desired path