Trajectory Tracking using Linear MPC

Aim:- The Robot Linear Model is used in MPC for Cubic Spline Trajectory Tracking.

The kinematic model of non-holonomic wheeled robot can be described as:

$\dot{x} = \vartheta * cos\,(\Theta)$

$\dot{y} = \vartheta * sin\,(\Theta)$

$\dot{\Theta} = w$

Or, in a more general way,

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

where $\boldsymbol{x} = [x\ y\ \Theta]^T$ describes the position and orientation of the center of the axis of the wheels, with respect to a global inertial frame. $\boldsymbol{u} = [\vartheta\ w]^T$ is the control input, where ϑ and w are the linear and the angular velocities of robot, respectively.

Considering a sampling period T, a sampling instant k and applying the Euler's approximation, we obtain the following discrete-time model for the robot motion:

$$x(k+1) = x(k) + \vartheta(k) * cos\,\Theta(k) * T$$

$$y(k+1) = y(k) + \vartheta(k) * sin\,\Theta(k) * T$$

$$\Theta(k+1) = \Theta(k) + w(k) * T$$

The problem of trajectory tracking can be stated as to find a control law such that

$$\mathbf{x} - \mathbf{x}_r = \mathbf{0}$$

where $\mathbf{x}_r$ is a known, pre-specified reference trajectory.

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

A linear model of the system dynamics can be obtained by computing an error model with respect to a reference car. By expanding the right side of $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$ in Taylor series around the point (x_r, y_r) and eliminating the high order terms it follows that:

$$\dot{\mathbf{x}} = f(\mathbf{x}_r, \mathbf{u}_r) + \frac{\partial f(\mathbf{x}, \mathbf{u})}{\partial x}\Big|_{(\mathbf{x}=\mathbf{x}_r, \mathbf{u}=\mathbf{u}_r)}(\mathbf{x} - \mathbf{x}_r) + \frac{\partial f(\mathbf{x}, \mathbf{u})}{\partial u}\Big|_{(\mathbf{x}=\mathbf{x}_r, \mathbf{u}=\mathbf{u}_r)}(\mathbf{u} - \mathbf{u}_r)$$

$$\dot{\mathbf{x}} = f(\mathbf{x}_r, \mathbf{u}_r) + f_{x,r}\,(\mathbf{x} - \mathbf{x}_r) + f_{u,r}\,(\mathbf{u} - \mathbf{u}_r)$$

where $f_{x,r}$ and $f_{u,r}$ are the jacobians of f with respect to $\mathbf{x}$ and $\mathbf{u}$, respectively. Now, error model can be,

$$\dot{\tilde{x}} = f_{x,r}\tilde{x} + f_{u,r}\tilde{u}$$

where, $\tilde{x} = (\mathbf{x} - \mathbf{x_r})$ represents the error with respect to the reference car and $\tilde{u} = (\mathbf{u} - \mathbf{u_r})$ is its associated error control input.

The approximation of $\dot{\mathbf{x}}$ by using forward differences gives the following discrete-time system model:

$$\tilde{x}(k+1) = A(k)\tilde{x}(k) + B(k)\,\tilde{u}(k)$$

Where $\tilde{x} = [\Delta x\ \Delta y\ \Delta\theta]^T$ and $\tilde{u} = [\Delta\vartheta\ \ \Delta w]^T$

$$A(k) = \begin{bmatrix} 1 & 0 & -\vartheta_r(k)\sin\theta_r(k)T \\ 0 & 1 & \vartheta_r(k)\cos\theta_r(k)T \\ 0 & 0 & 1 \end{bmatrix}$$

$$B(k) = \begin{bmatrix} \sin\theta_r(k)T & 0 \\ \cos\theta_r(k)T & 0 \\ 0 & T \end{bmatrix}$$

Cost function of MPC is,

$$\min(\varphi) = Q_f(\widetilde{\mathbf{x}_T})^2 + \sum_0^{T-1}[Q(\widetilde{\mathbf{x}_t})^2 + + R(\widetilde{\mathbf{u}_t})^2] + R_d((\widetilde{\mathbf{u}_{t+1}}) - (\widetilde{\mathbf{u}_t}))^2$$

where $Q_f = diag[10\ \ 10\ \ 0], Q = diag[10\ \ 10\ 0], R = [\ 1000\ 0.01], R_d = diag[100\ \ 100\ \ 0]$

Subject to,

- Robot linear model.

$$\tilde{x}(k+i+1) = A(k+i)\tilde{x}(k+i) + B(k+i)\,\tilde{u}(k+i)$$

- Kinematic model of unicycle robot

$$\begin{bmatrix} w_{\,l}(k) \\ w_{\,r}(k) \end{bmatrix} = \begin{bmatrix} \frac{1}{r} & \frac{b}{2*r} \\ \frac{1}{r} & \frac{-b}{2*r} \end{bmatrix} * \begin{bmatrix} \Delta\vartheta(k) + \vartheta_{ref} \\ \Delta w(k) + 0 \end{bmatrix}; \textit{ since } w_{\,ref} = 0$$

$w_{\,l}$ = left wheel angular speed & $w_{\,r}$ = right wheel angular speed & b = width of robot & r = wheel's radius

- Lower and upper bound of motor speed

$$\begin{bmatrix} 0\ RPM \\ 0\ RPM \end{bmatrix} \le \begin{bmatrix} w_{\,l}(k) \\ w_{\,r}(k) \end{bmatrix} \le \begin{bmatrix} 150\ RPM \\ 150\ RPM \end{bmatrix}$$

- Initial State

$$\tilde{x}_0 = (\mathbf{x} - \mathbf{x_r})_{obs}$$

Note: Robot's linear and angular velocity is bounded by motors angular velocity.

Python MPC simulation Results: -

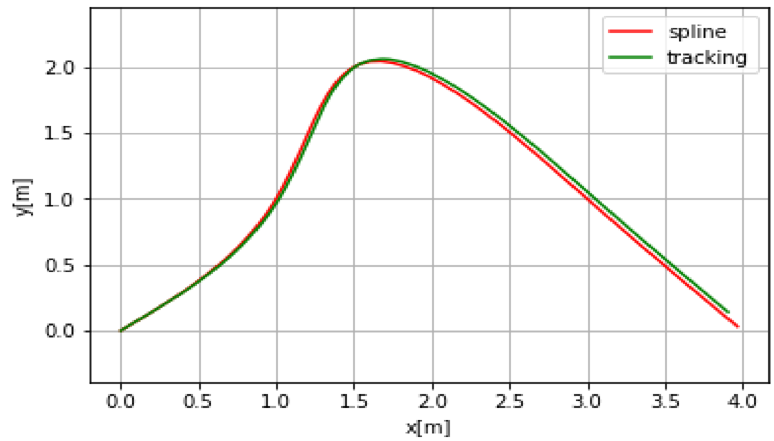

Fig1: Trajectory tracking result; red is desired trajectory and green is robot's tracked trajectory

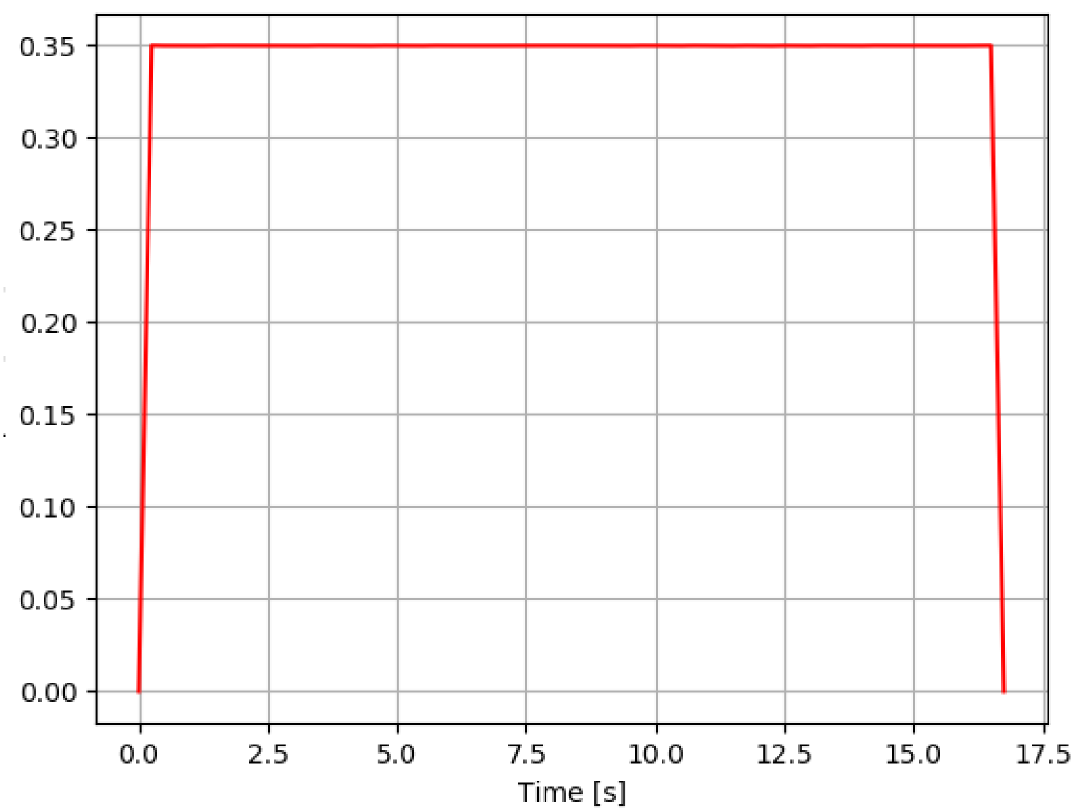

Fig 2: Speed vs time (v ref is 0.35 m/sec)