LAB2
EL2700 - Model Predictive Control

Lukas Bjarre
lbjarre@kth.se
921008-0256

---

1 Vehicle model

The first step of designing a good MPC is to obtain a good model of the system which is to be controlled.

1.1 Continous time system

The continous time model used will be one derived by the bicycle model of the car. Given the internal states $z(t) = [x(t),\, y(t),\, v(t),\, \psi(t)]^T$ and the inputs $u(t) = [a(t),\, \beta(t)]^T$, the system dynamics can be expressed on the form $\dot{z}(t) = f(z(t),\, u(t))$ as:

$$\underbrace{\begin{bmatrix} \dot{x(t)} \\ y(t) \\ v(t) \\ \psi(t) \end{bmatrix}}_{\dot{z}(t)} = \underbrace{\begin{bmatrix} v(t)\cos\left(\psi(t)+\beta(t)\right) \\ v(t)\sin\left(\psi(t)+\beta(t)\right) \\ a(t) \\ \frac{v(t)}{l_r}\sin\left(\beta(t)\right) \end{bmatrix}}_{f(z(t),\,u(t))} \tag{1}$$

1.2 Discrete time system

In order to implement the MPC the model needs to be discretizised. This is here done by the explicit Euler method. Introducing the sampling time T_s and the discrete time variable k such that $kT_s = t$ the discretization yield the system:

$$z_{k+1} = z_k + T_s f\left(z_k,\, u_k\right) \tag{2}$$

Equation (2) is the model which is both used in the controller and the simulation of the system.

1.3 Linearized model

Nonlinear models can be simplified by linearization around an operational point, which in this case will be $\psi_0 = 0$, $v_0 = v_0$ and $\beta_0 = 0$ (x_0 and y_0 does not end up in the final equations). Further simplifying the system to be constant speed (i.e. removing the acceleration input from $u(t)$) causes v_0 to be an equilibrium point, and the model can be approximated as:

$$\begin{bmatrix} \Delta\dot{x(t)} \\ \Delta y(t) \\ \Delta v(t) \\ \Delta\psi(t) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & v_0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{A} \begin{bmatrix} \Delta x(t) \\ \Delta y(t) \\ \Delta v(t) \\ \Delta\psi(t) \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ v_0 \\ 0 \\ \frac{v_0}{l_r} \end{bmatrix}}_{B} \Delta\beta(t) \tag{3}$$

The discretized system $\Delta z_{k+1} = \Phi\Delta z_k + \Gamma\Delta\beta_k$ is calculated via $\Phi = e^{AT_s}$ and $\Gamma = \int_0^{T_s} e^{As}B\mathrm{d}s$, which yields:

$$\Delta z_{k+1} = \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & v_0T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Delta z_k + \begin{bmatrix} 0 \\ v_0T_s\left(1+\frac{v_0T_s}{2l_r}\right) \\ 0 \\ \frac{v_0T_s}{l_r} \end{bmatrix} \Delta\beta_k \tag{4}$$

2 Track model

2.1 Obstacle model

The track has obstacles in it which the controller need to avoid. Each obstacle is defined by its center position and its size in the x and y dimensions. We define the set $\mathcal{O}$ as all the obstacles in the track and assume perfect knowledge of all obstacles.

The controller should avoid collision with these obstacles. This will be implemented through a function $C(x)$ taking the optimization variable x, which the solver will constrain to $C(x) \geq 0$. The function which will be used can intuitively be interpreted as the minimum distance to an edge point of the obstacle, where distances inside the box are negative. Defining the sets $\mathcal{B}$ as all the points inside the obstacle and $\mathcal{B}_\mathcal{E}$ as all the points on the edge of the obstacle the function can be defined as:

$$d_{\text{obs}}(x,\, y) = \begin{cases} \min\limits_{(x_e,\, y_e) \in \mathcal{B}_\mathcal{E}} \|(x,\, y) - (x_e,\, y_e)\|_2 & \text{if } (x,\, y) \notin \mathcal{B} \\ -\min\limits_{(x_e,\, y_e) \in \mathcal{B}_\mathcal{E}} \|(x,\, y) - (x_e,\, y_e)\|_2 & \text{if } (x,\, y) \in \mathcal{B} \end{cases} \tag{5}$$

This function can be seen in fig. 1 for a box with the sizes used in the lab. This function works well since it is continous for all points which makes methods such as grandient descent possible for the solvers when trying to fulfill the constraint.

Note that the function can be impemented in code in such a way that no optimization problem needs to be solved.

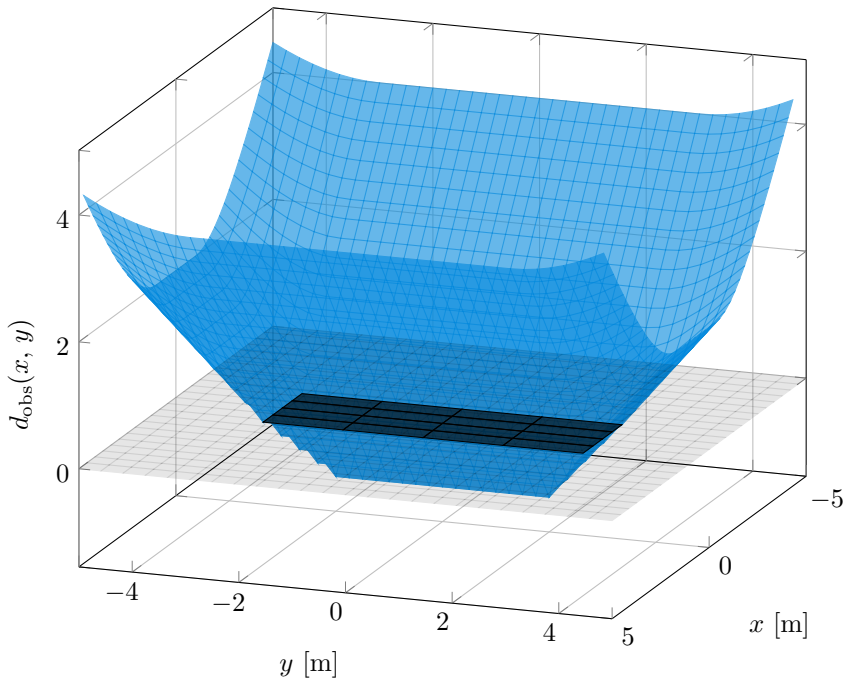

Figure 1: Plot of the distance function $d_{\text{obs}}(x,\, y)$ around an obstacle centered at $(0,\, 0)$ and dimensions $x_{\text{size}} = 2$, $y_{\text{size}} = 6$.

3 Controller design

The controller needs a cost function to minimize to calculate the optimal control. Two parts will be considered here for the cost function: distance to goal and control size.

3.1 Distance to goal

The main objective in this setting is to reach the end of the track as quickly as possible, so punishing the distance to the end will cause the controller to get there as quickly as possible.

3.2 MPC formulation

$$
\begin{aligned}
\text{minimize} \quad & \sum_{n=0}^{N-1} \left[u_n^T Q_u u_n + (x_{\text{goal}} - x_n) Q_x \right] + (x_{\text{goal}} - x_N) Q_x \\
\text{subject to} \quad & z_{k+1} = z_k + T_s f(z_k, u_k) \\
& d_{\text{obs}}(x_{k+1},\, y_{k+1}) \geq d_{\text{margin}} \\
& -8 < y_k < 8 \\
& -a_{\max} \leq a_k \leq a_{\max} \\
& -\beta_{\max} \leq \beta_k \leq \beta_{\max} \\
& -\dot{\beta}_{\max} \leq \frac{\beta_k - \beta_{k-1}}{T_s} \leq \dot{\beta}_{\max} \\
& \forall k = 0,\, 1,\, \ldots,\, N-1, \quad \forall \text{obs} \in \mathcal{O}
\end{aligned}
\tag{6}
$$

4 Simulations