MPC cheatsheet

December 16, 2018

1 Model Predictive Control

1.1 MPC setup

We deal with a **discrete** or discretized problem considered over a **N steps horizon**. Every step has a dt duration.

$$\min_{u_t, u_{t+1}, \ldots, u_{t+N-1}} \sum_{k=t}^{t+N-1} l(x_k, u_k)$$

$$\text{subj. to} \begin{cases} x_{k+1} = f(x_k, u_k) & k = t, \ldots, t+N-1 \\ u_k \in \mathcal{U} & k = t, \ldots, t+N-1 \\ x_k \in \mathcal{X} & k = t, \ldots, t+N-1 \\ x_t = x(t) & k = t, \ldots, t+N-1 \end{cases}$$

With:

- u a control command e.g. $u_k = [a_x, a_y]^T$ at time step k or $u_k = [a_k, \delta_k]^T$
- x_k a state vector e.g. $\mathbf{x}_k = [x_k, y_k, x'_k, y'_k]^T$ or $x_k = [x, y, \theta, v]^T$
- $x_{k+1} = f(x_k, u_k)$ usually corresponds to our vehicle dynamics model e.g. $x_{k+1} = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_k + \begin{bmatrix} \frac{dt^2}{2} & 0 \\ 0 & \frac{dt^2}{2} \end{bmatrix} u_k = Ax_k + Bu_k = f(x_k, u_k)$
- l is a cost function we want to optimize: examples include jerk (to optimize comfort), difference with a planned trajectory, time to goal or a combination of objectives

We are solving $\min_{u_{t:t+N-1}} \sum_{k=t}^{t+N-1} l(x_k, u_k)$ and not $\min_{u_{[t:t+N-1]}} \int_t^{t+N} l(x(\tau), u(\tau)d\tau$!!! Soon or later you need to discretize anyways

The algorithm will run at every time steps:

- At time t:

- Measure (or estimate) the current state
  - Find the optimal input sequence $U^* = \{u_t^*, u_{t+1}^*, u_{t+2}^*, \ldots, u_{t+N-1}^*\}$
  - Apply only $u(t) = u_t^*$ and discard $u_{t+1}^*, u_{t+2}^*, \ldots, u_{t+N-1}^*$

- Repeat the same procedure at time $t + 1$

  - Even assuming perfect model and no disturbances: the predicted open-loop trajectories are DIFFERENT than the closed-loop trajectories

This algorithm has the following characteristics: multivariate, model based, nonlinear, constraints based, predictive. At each sampling time, starting at the current state, an open-loop optimal control problem is solved over a finite horizon

1.2 MPC requirements

In order to solve a MPC problem we need:

- A discrete-time model of the system
- A state observer
- To setup an Optimization Problem
- To solve an Optimization Problem (Matlab Optimization toolbox, NPSOL, cvxgen, C++ Ipopt/CppAd tools)
- Verifiy that the closed-loop system performs as desired (avoid infeasibility, unstability)
- Make sure it runs in real-time

1.3 MPC with Continuous Time models

Use e.g. Euler Discretization of Nonlinear Models. Given a CT model

$$\begin{cases} \frac{d}{dt}x(t) & = f(x(t), u(t), t) \\ y(t) & = h(x(t), u(t), t) \end{cases}$$

We approximate with finite differences:

- $\frac{d}{dt}x(t) \approx \frac{x(t+\Delta t)-x(t)}{\Delta t}$
- Δt is the sampling time

Use Euler discretization or better discretization approaches (cf matlab: help c2d) and check the performance of your model data vs simulated.

2 Optimization

2.1 Convex optimization problems

- A set $\mathcal{S}$ is convex if $\lambda z_1 + (1-\lambda) z_2 \in \mathcal{S}$ for all $z_1, z_2 \in \mathcal{S}, \lambda \in [0,1]$
- A function $f : \mathcal{S} \to \mathbb{R}$ is convex if
  - $\mathcal{S}$ is convex
  - $f(\lambda z_1 + (1-\lambda) z_2) \leq \lambda f(z_1) + (1-\lambda) f(z_2)$ for all $z_1, z_2 \in \mathcal{S}, \lambda \in [0,1]$
- A function $f : \mathcal{S} \to \mathbb{R}$ is concave if $\mathcal{S}$ is convex and $-f$ is convex
- Some operations preserve convexity: intersection of convex sets, $f(Ax+b)$ ...

An optimization problem is said to be convex if the cost function f is convex over a convex set. A fundamental property of convex optimization is that local optimizers are also global optimizers. It suffices to compute a local minimum to determine its global minimum. Non-convex problems can be transformed into convex problems through a change of variables and manipulations on cost and constraints.

2.2 Optimality conditions

2.3 Optimization algorithm: overall intuitive presentation

2.4 Optimization tools