MPC project – reflection

Author: Jelena Kocić
Date: 26.06.2017.

The Model

The model is "Mind The Line" from the lesson 19, class 8 and 9. Here we have 6 states and 2 actuators:
State: x, y, psi, v, cte, epsi,
Actuator: steering, speed.

The cost function is defined as a weighted sum of several factors we want to minimize:

- distance to the referenced central line (file MPC.cpp, lines 67-69)
- no drastic changes in steering/speed (file MPC.cpp, lines 75, 76)
- no significant values of steering/speed (file MPC.cpp, lines 82, 83)

In each measurement update, we use an optimizer to minimize the cost, with the constraint expressed as motion update models: file MPC.cpp, lines 124-129.

Timestep Length and Elapsed Duration (N & dt)

Final choice for timestep length and elapsed duration are: N=10, dt =0.07. This combination provides me the speed around 90km/h, and in one moment even over 102km/h (please look at the video https://www.youtube.com/watch?v=imLFqbLnfJ0&t=44s , time: 1.22 min)

I have started with tuning this parameters with values N=10, dt=0.1 and this was fine, car was driving safely, speed was about 60km/h. When I changed parameters on N=20, dt=0.05 car was very unstable. Even if it is general recommendation that prediction horizon (T = N*dt) has to be as large as possible, and time between actuations, dt, has to be as small as possible, this combination in this implementation did not give the good result.

Further tuning of hyperparameters implies that even if I have enough computational time to increase N (number of timestamps in the horizon) this did not helped with remaining stability and getting higher speed. I leave N=10, and start reduce dt in order to get speed. dt=0.7 gives me enough speed (102 km/h is max) while car remain on the road all the time.

Polynomial Fitting and MPC Preprocessing

Polynomial fitting is given in main.cpp file in the file main.cpp, line of 139, here we calculate the coefficients. In lines 141 and 142 we are evaluating this coefficients in order to calculate cte (the cross track error - the error between the center of the road and the vehicle's position) and epsi (orientation error).

All computations are performed in the vehicle coordinate system. The coordinates of waypoints in vehicle coordinates are obtained by first shifting the origin to the current poistion of the vehicle and a subsequet 2D rotation to align the x-axis with the heading direction. Main.cpp, lines 121-131. The initial position of the car and heading direction are always zero in this frame. Thus the state of the car in the vehicle cordinate system is `state << 0, 0, 0, v, cte, epsi;` main.cpp, line 148.

Model Predictive Control with Latency

Here we deal with latency after receiving the measurements. Latency is added to four states: coordinates x, y, angle psi and velocity. The update equations used are given in main.cpp, lines 116-119. These values are then passed to the Solver.

Dealing with latency is equivalent to looking ahead while you are driving, realizing you can't do that much about what is immediately in front of you at highway speeds. So decisions are now affecting location, heading and speed a few meters in front, not at the current state.