CarND-Controls-MPC

Self-Driving Car Engineer Nanodegree Program

Compile

This project could be compile build and run well on Linux(ubuntu) by following instructions provided.

The Model

For the car with a state vector $S = (x, y, \psi, v)$. The kinematic model can be expressed as following equations:

$x_t = x_{t-1} + v_{t-1} * \cos(\psi_{t-1}) * dt$

$y_t = y_{t-1} + v_{t-1} * \sin(\psi_{t-1}) * dt$

$\psi_t = \psi_{t-1} + \frac{v_{t-1}}{L_f} * \delta * dt$

$v_t = v_{t-1} + a_{t-1} * dt$

where (x, y) is the coordinate of the car, ψ is the heading angle, and v is the velocity. a is the **throttle acceleration** and δ is the steering angle that need to be calculated to optimize.
The cross-trace error and orientation error update could be expressed below:

$cte_t = cte_{t-1} + v_t * sin(e\psi_{t-1}) * dt$

$e\psi_t = e\psi_{t-1} + \frac{v_{t-1}}{L_f} * \delta * dt$

Line 103 - 109 in MPC.cpp reflects the model update in the code.

Timestep Length and Elapsed Duration

Time horizon T = N * dt here N is 10 and dt is 0.1. Other combinations have been tried were proven that easy to cause the car behavior weird A length of 100ms time step is also consistent with latency.

Polynomial Fitting and MPC Preprocessing

From line 99 - 120 in main.cpp, I preprocessed waypoints to vehicle's cordinate system also with state vector, then used the provide utils to fitting the polynomial.

Model Predictive Control with Latency

Since the timestep and latency is consistent, which is 100ms, we are able to use previous step's actuations to apply to current step.

Simulation

Simulation looks well, car can run and run over laps within the track.