Model explanation

The applied model used is called "kinematic bicycle model", which considers (non-linear) changes in the heading direction, but does not consider other dynamical effects, i.e., friction or torque. The model comprises several equations:

$$x_{t+1} = x_t + v_t * \cos(\psi_t) * dt \tag{1}$$

$$y_{t+1} = y_t + v_t * \sin(\psi_t) * dt \tag{2}$$

$$\psi_{t+1} = \psi_t + \frac{v_t}{Lf} * \delta_t * dt \tag{3}$$

$$v_{t+1} = v_t + a_t * dt \tag{4}$$

$$cte_{t+1} = f(x_t) - y_t + v_t * \sin(e\psi_t) * dt \tag{5}$$

$$e\psi_{t+1} = \psi_t - \psi_{dest(t)} + \frac{v_t}{Lf} * \delta_t * dt \tag{6}$$

where x and y represent the vehicle position, v the velocity, ψ the heading direction, $e\psi$ the error in orientation, and cte the cross-track error.

In the model, a reference trajectory is predicted ahead of the vehicle's actual position, which is calculated by

$$T = N * dt \tag{7}$$

where N is the number of timesteps we want to go into the future, and dt gives the time delta in seconds. I have no specific explanation for why I chose the specific model parameters currently governing the model. The final model is mostly based on experimental results, and my final parameters are $N = 15$ and $dt = 0.05$. I initially started with a value of N=30, as my assumption was that the higher the horizon is, the better the performance will be. This turned out to be problematic, especially at higher speed (> 40 mph), as the car would almost immediately leave the track. I then decreased to 20, which gave better accuracy and performance, but still, the car would leave the track in certain occasions, i.e., when the car approaches a curve to the left from the outer right side of the track. Changing it to a value < 10 would result in oscillation between the left and right side of the lane, especially at high speeds, which after seconds becomes so bad that the car is not able to stay on track as soon as it is in the curve. It seems that setting the horizon to anything between 12 and 17 is most stable, so I decided to go for a value of 15. Even with high speeds, this works fine. I have no scientific argument behind my decision, it is just based on trial and error.

The reference trajectory is used to calculate the error by fitting a polynomial to the reference trajectory for each waypoint: the more off the vehicle is from the reference trajectory (the higher the cross-track error and $e\psi$), the higher the error. Furthermore, the rate of change of $e\psi$ and the cross-track error is considered: the higher the change rate, the higher the error. Finally, the difference between actuations are penalized. Scaling factors for each of these components have been used to allow for individual manipulation of each of those.

Latency

The time interval given by the execution of an actuation-command to its effect in the physical world is called latency. This may be only in the lower 3 digit ms-range, however if not handled properly, the vehicle cannot be controlled adequately with respect to its dynamically changing environment. In the worst case, this results in a crash. The latency can be addressed by calculating actuations n time steps ahead, depending on the time it takes from execution to effect. In the given example, 2 time steps ahead seem to be appropriate, and it can be found in Prediction MPC::Pred. The used constant latency_dt, which gives the latency in dt-units, is defined in MPC.h.

References

http://www.me.berkeley.edu/~frborrel/pdfpub/IV_KinematicMPC_jason.pdf

https://inst.eecs.berkeley.edu/~ee192/sp13/pdf/steer-control.pdf

https://nabinsharma.wordpress.com/2014/01/02/kinematics-of-a-robot-bicycle-model/

https://github.com/laventura/CarND-MPC