Project 5 – Model Predictive Control

1. The Model

The model predictive control (MPC) has been increasingly adopted in the industry due to its simplicity and robustness against external disturbances and changes in the environment. Actually, MPC optimizes a pre-defined cost function each sampling time, taking into account the next state prediction to follow a desired trajectory in a future horizon composed of N samples. In this context, a comprehensive system model is required in order to predict the next state and implement the MPC accordingly.

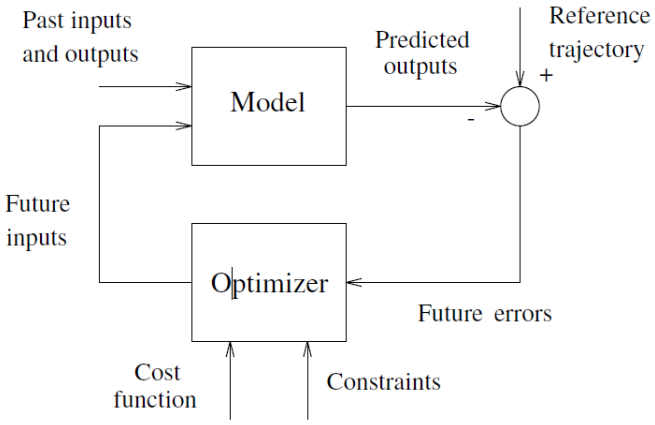

Figure 1: Basic structure of MPC

As depicted in the Figure 1, the MPC model take as input the past model's inputs and outputs and future actuator values. In this project, it was used a global kinematic model, that take into account the car's velocity (v), the car's position in the global reference (x, y) as well as car's angle in relation to the global reference (ψ) for vehicle localization in the map – see Figure 2 for more detail.

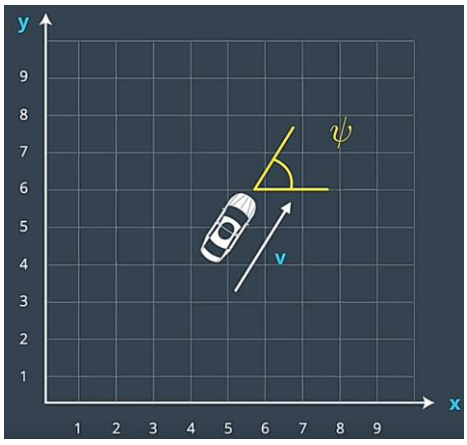

Figure 2: Car's position in the global mapping reference

Besides the vehicle position variables, the states of the model predictive control consider the cross tracking error (cte) and the vehicle's angle error ($e\psi$). Therefore, the state of the model predictive control is composed by these six variables:

$$[x \quad y \quad v \quad \psi \quad cte \quad e\psi]$$

Knowing these six variables, it is possible to predict the next state using the kinematic model, as detailed in the following equations:

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

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

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

$$\psi_{t+1} = \psi_t + \frac{v_t}{L_f} \delta dt$$

where L_f is the distance between the front of the vehicle and its centre of gravity and δ is the vehicle's steering angle. Once, the states are updates, the MPC optimize an object function that aim to minimize both cte and $e\psi$, in a safe and smooth way.

In this context, in order to minimize the errors, the MPC use as actuators the throttle and brake pedals (a) as well as the steering wheel angle (δ). It is noteworthy that in the Udacity's simulator, the throttle pedals must be within [1,1] and the steering wheel angle within $[-25°, 25°]$. That is why the MPC concept consists of solving in real time a constraint object function to determine the optimal values of the actuators:

$$[a \quad \delta]$$

2. Timestep Length and Elapsed Duration (N & dt)

The MPC algorithm will solve the object function in a future horizon composed of N samples, but only the first value is sent to the system. The command sent to the system must consider the delay between each timestep (dt), because during this period the car can diverge from the trajectory but the controller will only sent another command to correct it in the next dt. As a result, without a proper consideration of the delay, the car can easily get out from the road.In order to reduce the effect of the delay, the timestep was fixed to 100 ms to be consistent with the latency in the actuators. In this way, the actuator values will be in accordance with the next model state prediction.

Another fundamental parameter in MPC is the horizon length determined by the number of samples N. After trying different values of N during the project resolution, it was remarked that there is a tradeoff between trajectory oscillations and priori command knowledge. In fact, with large N, the car knows a priori the future trajectory and makes better decision concerning the actuator values. However, when the error (in cte or δ) is big, the future trajectory becomes oscillatory, resulting in system instability. As a result, the value of N was fixed as 10, a reasonable value that can ensure enough priori trajectory estimation and reduce the oscillatory behaviour in the trajectory.

3. Polynomial Fitting and MPC Preprocessing

In order to determine the trajectory reference, the future waypoints were fitted in a third order polynomial using the function *polyfit*. In this way, the trajectory can be written as the form of:

$$traj(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3$$

Since, the values of car's position coming from the Udacity's simulator are on the global mapping reference, the state calculation can be troublesome. Therefore, for the sake of simplicity, all the measurements were changing to the cars reference. In this way, (x_0, y_0) coordinate becomes (0,0) and the car's angle $\psi_0 = 0$. As a result, the current car state is simplified as stated bellow.

$$[0 \quad 0 \quad 0 \quad v \quad cte \quad e\psi]$$

With the definition og the future trajectory and the current state in the car's reference, it was possible to calculate the cross tracking error and the steering angle error as the following equations:

$$cte = y_0 - traj(x_0) = c_0$$

$$e\psi = \psi_0 - \arctan\big(traj'(x_0)\big) = -c_1$$

4. Model Predictive Control with Latency

In order to reduce the side effects of latency, before calling the function *Solve* of the *MPC* class, the next state was estimated based on the current state ($[0 \quad 0 \quad 0 \quad v \quad cte \quad e\psi]$). This technique is as feed forward controller to improve the transient of the vehicle trajectory and anticipate future changes.

Moreover, the object function was changed in order to reduce the changes and use of actuators. It was noted that mainly the change in the steering angle affected directly the future trajectory estimation. As a result, a constant changing in the steering angle lead changes in the trajectory, which in its turning resulted in instability. As a solution, it was added a cost function to penalize the changes in the steering angle, by forcing the current and past steering angle values being as close as possible:

$$f_t^{\,1} = 300000(\delta_t - \delta_{t-1})^2$$

The same strategy was adopted with the throttle paddle:

$$f_t^{\,2} = 1000(a_t - a_{t-1})^2$$

Additionally, it was noted that the vehicle had difficulty in making curves because of the high velocity and latency. To overcome this issue, it was added another cost function with lower weight that try to reduce the velocity while doing curves:

$$f_t^{\,3} = 2(a_t \delta_t)^2$$

Also to reduce the use of actuators, other two cost functions were incorporated in the final objective function:

$$f_t^4 = 100(a_t)^2$$

$$f_t^5 = 100(\delta_t)^2$$

It was also remarked that the angle error was much more important to keep as low as possible than the cross tracking error and velocity error that is why the weights were chosen as follow:

$$f_t^5 = 100(cte_t)^2$$

$$f_t^6 = 30000(e\psi_t)^2$$

$$f_t^7 = \left(v_t - v_{ref}\right)^2$$

As a result, the final objective function of the MPC was defined as the sum of all these individual interesting aspects, forming in this way a multi-objective function: