UDACITY

PROJECT

Model Predictive Control (MPC)

A part of the Self-Driving Car Engineer Program

PROJECT REVIEW

CODE REVIEW 6

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

You did a great job on implementing the Model Predictive Control! Your controller is able to effortlessly steer the car around the track, while also staying safe in the sharp turns. Your code is structured very well and builds without any problems. Your report is also clear and to the point! I have left some remarks in the code review on how you could further improve your code in some areas.

This marks the end of the second term, I wish you all the best for the last term!

Compilation

✓

Code must compile without errors with `cmake` and `make`.

Given that we've made CMakeLists.txt as general as possible, it's recommend that you do not change it unless you can guarantee that your changes will still compile on any platform.

Your project builds without problems!

Implementation

✓

Student describes their model in detail. This includes the state, actuators and update equations.

You clearly describe the motion model in the included report!

✓

Student discusses the reasoning behind the chosen *N* (timestep length) and *dt* (elapsed duration between timesteps) values. Additionally the student details the previous values tried.

You clearly describe how you arrived at your choice of *N* and *dt*.

✓

A polynomial is fitted to waypoints.

If the student preprocesses waypoints, the vehicle state, and/or actuators prior to the MPC procedure it is described.

You transform the waypoint coordinates from global to vehicle coordinates before you fit the 3rd order polynomial.

✓

The student implements Model Predictive Control that handles a 100 millisecond latency. Student provides details on how they deal with latency.

Great work on projecting the current state into the future using the state equations!

Some tips on further improving the accuracy of the state prediction:

- Convert the velocity to m/s.
- Measure the actual latency by using `std::chrono::system_clock::now()`.
- Divide the prediction into smaller time steps. That way the current steering angle would iteratively change the yaw angle.

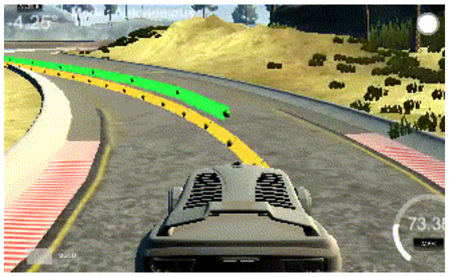
Simulation

✓

No tire may leave the drivable portion of the track surface. The car may not pop up onto ledges or roll over any surfaces that would otherwise be considered unsafe (if humans were in the vehicle).

The car can't go over the curb, but, driving on the lines before the curb is ok.

Your controller manages to steer the car safely around the track at considerable speed! Great work!



DOWNLOAD PROJECT

6 CODE REVIEW COMMENTS ›

RETURN TO PATH

Rate this review

Student FAQ