UDACITY

PROJECT

Path Planning

A part of the Self-Driving Car Engineer Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Meets Specifications

Greetings Student,
Congratulations for completing this project.👏
The ideas implemented were very good and computationally efficient which improved the smoothness of the vehicle's driving. The documentation was also very explicit and outlined the various steps and techniques used in planning a safe path for the ego vehicle. It was delightful reviewing your work, and I exhort you to keep up the good work which will make you an outstanding Self Driving Car Engineer. Best of luck with the rest of your projects.

Extra material

- Udacity CS373: Programming a Robotic Car Unit 4: Motion
- Path Planning and Collision Avoidance
- Safe Motion Planning for Autonomous Driving
- Local and Global Path Generation for Autonomous Vehicles Using Splines
- Medium- Path Planning in Highways for an Autonomous Vehicle
- Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions

More comments

About the error (segmentation fault), I want to add this:
Segmentation fault is a specific kind of error caused by accessing memory that “does not belong to you.” It’s a helper mechanism that keeps you from corrupting the memory and introducing hard-to-debug memory bugs. Whenever you get a segfault you know you are doing something wrong with memory – accessing variable that has already been freed, writing to a read-only portion of the memory, etc. Segmentation fault is essentially the same in most languages that let you mess with the memory management, there is no principial difference between segfaults in C and C++.

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.

The code successfully compiles without errors with cmake and make to show that compilation errors were completely avoided in the code. Great work!

Suggestions and Comments

To know more about cmake and make, visit these links:

- Cmake FAQS.
- Using make and writing Makefiles.
- Youtube set of tutorials on using make and writing Makefile.
- MakeFiles

Valid Trajectories

✓

The top right screen of the simulator shows the current/best miles driven without incident. Incidents include exceeding acceleration/jerk/speed, collision, and driving outside of the lanes. Each incident case is also listed below in more detail.

✓

The car doesn't drive faster than the speed limit. Also the car isn't driving much slower than speed limit unless obstructed by traffic.

The vehicle coasted through a full lap of the highway without exceeding the speed limits set on the highway. This is very good work and shows how much time and commitment was put into completing the project. Keep it up!

✓

The car does not exceed a total acceleration of 10 m/s^2 and a jerk of 10 m/s^3.

The vehicle successfully drives through the highway without exceeding the total acceleration of 10m/s^2 and jerk of 10m/s^3. Excellent work!

✓

The car must not come into contact with any of the other cars on the road.

The vehicle slows down when close to the leading vehicle. It also changes its lane when a lane change is possible and safe. This helped in avoiding any collision with other vehicles on the highway. Good work!

✓

The car doesn't spend more than a 3 second length out side the lane lanes during changing lanes, and every other time the car stays inside one of the 3 lanes on the right hand side of the road.

The vehicle is very swift when changing lanes and does not exceed the 3s limit on the lane lines while performing lane change. The vehicle does not also spend 3s outside the lane when going through the track. Good work!

✓

The car is able to smoothly change lanes when it makes sense to do so, such as when behind a slower moving car and an adjacent lane is clear of other traffic.

Lane change is quite smooth and occurs without exceeding jerk and maximum acceleration limits. Very good work!

Reflection

✓

The code model for generating paths is described in detail. This can be part of the README or a separate doc labeled "Model Documentation".

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

☆☆☆☆☆

Student FAQ