Path Planning Project

The goal of the project is :

Design a path planner that is able to create smooth, safe paths for the car to follow along a 3 lane highway with traffic. A successful path planner will be able to keep inside its lane, avoid hitting other cars, and pass slower moving traffic all by using localization, sensor fusion, and map data.

Rubric Points

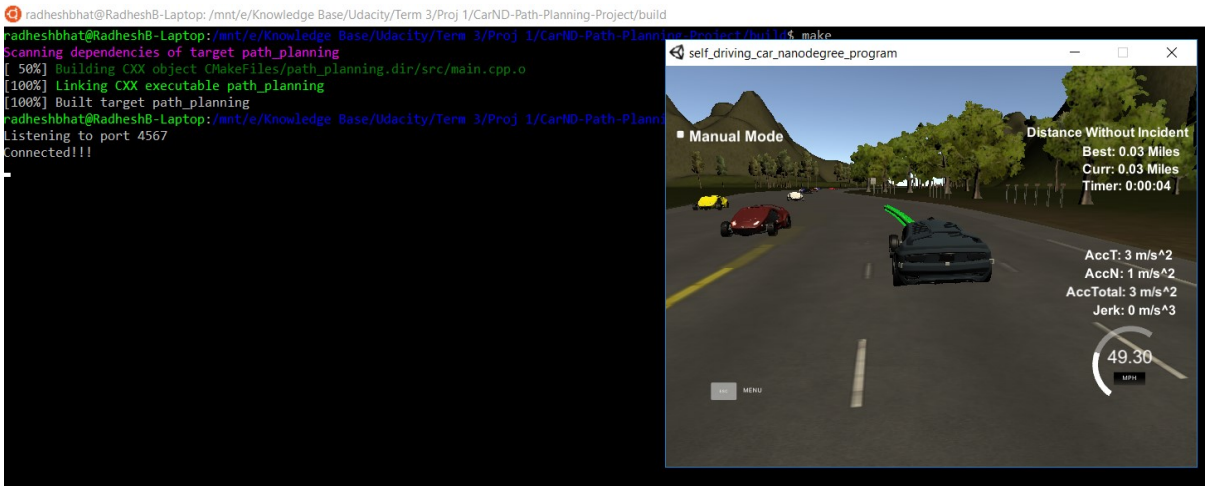
The code compiles correctly.

Yes. There was no change made in cmake configuration. Only a new file was added src/spline.h.

Valid Trajectories

The car is able to drive at least 4.32 miles without incident..

Yes. Able to run the simulator without any red flags or incidents.



The car drives according to the speed limit.

Yes, no speed limit red flag was seen.

Max Acceleration and Jerk are not Exceeded.

No red message for exceeding max jerk observed

Car does not have collisions.

No collisions observed.

The car stays in its lane, except for the time between changing lanes.

The car stays in its lane most of the time but when it changes lane because of traffic or to return to the centre lane.

The car is able to change lanes

The car change lanes when the there is a slow car in front of it, and it is safe to change lanes

Reflection

This was great and fun project to work on. The Q&A session really helped to bridge the gap in understanding the concept and real implementation.

The implementation code consists of 3 parts:

Prediction (main.cpp - line 259 to line 294)

The section of the code deals with the telemetry and sensor fusion data.We want to know three aspects of it:

- Is there a car in front of us blocking the traffic?
- Is there a car to the right of us making a lane change not safe?
- Is there a car to the left of us making a lane change not safe?

Above questions are answered by calculating the lane each other car is and the position it will be at the end of the last plan trajectory. A car is considered "dangerous" when its distance to our car is less than 30 meters in front or behind us.

Behaviour (main.cpp - line 297 to line 319)

This section of the code is trying to address:

- If we have a car in front of us, do we change lanes?
- Do we speed up or slow down?

Based on the prediction, this code logic increases the speed, decrease speed, or make a lane change when it is safe. Instead of increasing the speed at this part of the code, a speed_diff is created to be used for speed changes when generating the trajectory in the last part of the code. This approach makes the car more responsive acting faster to changing situations like a car in front of it trying to apply breaks to cause a collision.

Trajectory (main.cpp - line 321 to line 420)

The section of the code implements the logic of the calculation of the trajectory based on the speed and lane output from the behaviour, car coordinates and past path points.

Initially, the last two points of the previous trajectory are used in conjunction three points at a far distance to initialize the spline calculation. To make the work less complicated to the spline calculation based on those points, the coordinates are transformed (shift and rotation) to local car coordinates. In order to ensure more continuity on the trajectory (in addition to adding the last two point of the pass trajectory to the spline adjustment), the pass trajectory points are copied to the new trajectory. The rest of the points are calculated by evaluating the spline and transforming the output coordinates to not local coordinates . It is important to notice the change in the velocity of the car from line 397 to 402. The speed change is decided on the behaviour part of the code, but it is used in that part to increase/decrease speed on every trajectory points instead of doing it for the complete trajectory.