Path Planning Project

Introduction

This document serves to outline the logic I pursued to successfully meet the objectives of this project. Essentially, we are expected to calculate waypoints along the road for a car to follow such that we drive 4.32 miles without any incidents. Specifically, we must adhere to the following rules. Any events that do not follow the below requirements are deemed incidents.

- The car drives at the speed limit of 50 MPH unless obstructed by traffic
- A maximum acceleration of 10 m/s^2 and jerk of 10 m/s^3 is allowed.
- The car does not collide with any vehicles on the road
- The car stays in its lane unless required to change lanes
- The car is not allowed to spend more than 3 seconds outside of the lane lines (i.e. while changing lanes)
- The car is able to smoothly change lanes when it makes sense to do so, such as when behind a slowly moving car and an adjacent lane is clear of traffic

Implementation

The logic to perform the path planning was heavily borrowed from Aaron Brown's Q & A session for this project with some slight tweaks. The code has more verbose comments regarding the implementation, but a summary will be given here.

1. We first determine if there are any waypoints from the previous path that still need to be fulfilled at the current iteration. If so, we set the car's current longitudinal distance (i.e. the s-value) to be that of the ending one from the previous set of waypoints. This prevents the cold start problem where the car momentarily exceeds the limits for acceleration and jerk at the beginning of the simulation. In addition, we keep track of the relative velocity of the ego vehicle to finally minimise this from happening.

2. We iterate through the sensor fusion information to extract out the velocity components, the distance of the car along the path of the road from the origin, as well as the distance away from the double yellow lane reference (i.e. the d value). We thus determine whether we need to change lanes or not. For each car, we:

   (a) Project the position of the car in question forward based on its current velocity.

   (b) Check to see if the car is in the ego vehicle's lane

   (c) If the car is in our lane, and we see that the projected position of the car in question is closer than where we are going to end up *and* if the distance between the cars is less than 30 metres, we define this as a *too close* event. Note that this applies both *forwards* and *backwards*. The backwards checking allows us to safely merge back into the centre lane (more on this later).

   (d) If we detect that a car is on the left lane based on its d value and if we see that the distance between the ego vehicle and that car is less than 30 metres (longitudinally), then we detect that there is a car to the left of us.

   (e) Repeat (d) but look at the right lane to see if there's a car on the right lane.

(f) If we are too close and if we see that we aren't in the left lane, and if we see that there is no car in the left lane, mark that we need to move to the left lane.

(g) If we are too close and if we see that we aren't in the right lane, and if we see that there is no car in the right lane, mark that we need to move to the right lane.

(h) If we aren't too close, but if we see that we are in the left or right lane, we should move back to the middle lane (see (c)).

3. We first create a spline for the ego vehicle to follow that is uses set of 5 keypoints that the ego vehicle follows. The first two are based off of the previous path. We generate two points that are tangential to the ego vehicle's current trajectory. Next we define three waypoints such that their s values are 30, 60 and 90 metres away *in the target lane* from our current location longitudinally from where the ego vehicle is positioned. For our task, we always plan to make adjustments 30 metres out, but the addition of the 60 and 90 metre keypoint allows for an even smoother trajectory. Also note that the waypoints are defined with respect to the map, but then need to be transformed to be with respect to the ego vehicle as we have to determine what is close and what is far from the ego vehicle's perspective. Map coordinates won't help us here.

4. We then use the spline where the ending point is 30 metres longitudinally, and by using the spline to interpolate the lateral position (y coordinate), we can thus determine what the target distance is from the ego vehicle to where we need to go.

5. Using this target distance, we create waypoints such that we always have 50 waypoints to send to the output such that if we had any previous waypoints from the previous path, we simply append new waypoints that are further out from the end of the previous path until we reach the 50 quota. The reason why we choose 50 is because the waypoints are sampled at 0.02 seconds, thus making the travelled speed at 25 m/s, which is roughly 50 MPH. For each waypoint that we need to create:

   (a) If we were deemed too close, decrease the relative velocity by slowing down at 0.224 MPH, corresponding to a deceleration of 5 m/s^2

   (b) If we aren't too close and if we see that our relative velocity is less than 50 MPH, increase the relative velocity by 0.224 MPH.

   (c) By using the relative velocity, we can accurately define the x value of the waypoint relative to the ego vehicle. Therefore, when the car speeds up, the position will be farther out versus when the velocity is slower, the waypoint is closer to the ego vehicle.

   (d) Because this waypoint is with respect to the ego vehicle, we must transform this back with respect to the map.

6. After defining the waypoints we need from (5), send this to the output

7. Repeat 1-6 ad infinitum.

Sample Snapshots

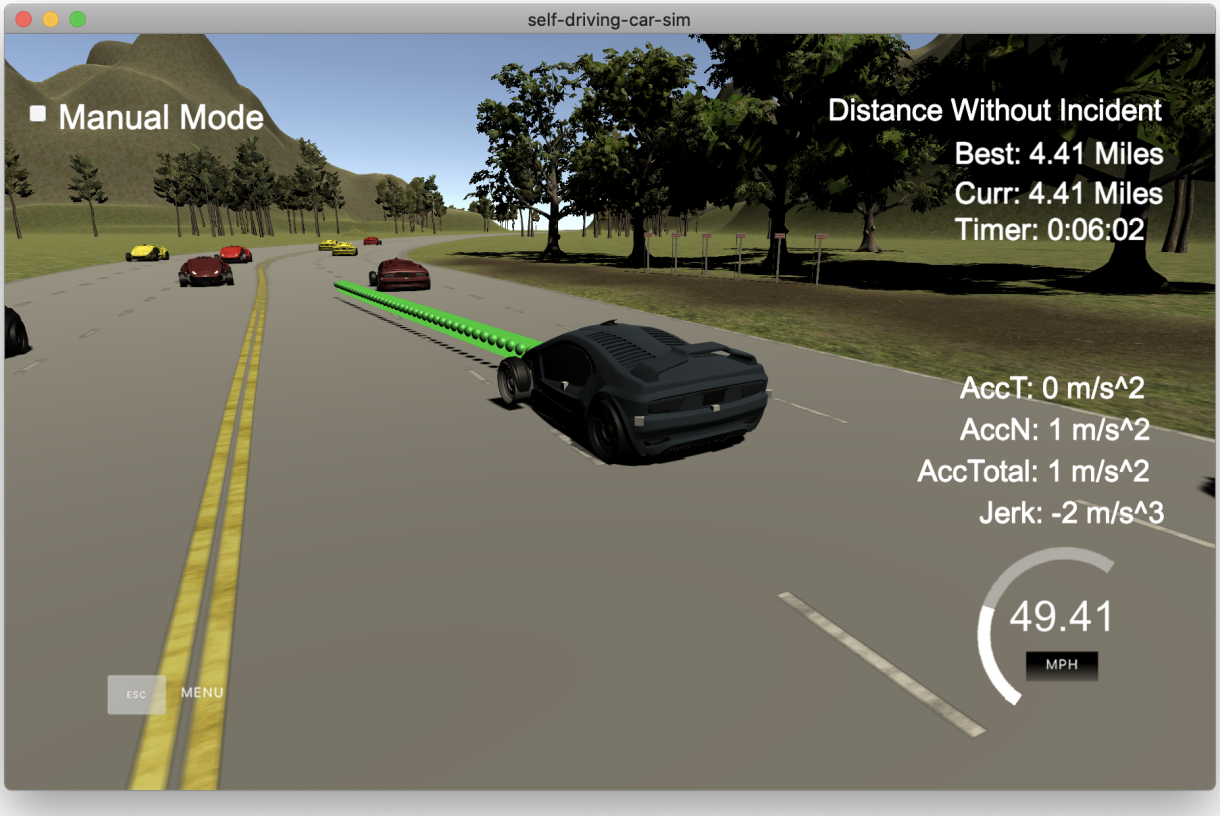
Here are some sample snapshots of the path planning logic in action.

Keeping in the lane

Left Lane Change and Moving Back to Centre Lane

Right Lane Change and Moving Back to Centre Lane

Meeting the 4.32 mile requirement with no incidents


self-driving-car-sim
Manual Mode
Distance Without Incident
Best: 4.41 Miles
Curr: 4.41 Miles
Timer: 0:06:02
AccT: 0 m/s^2
AccN: 1 m/s^2
AccTotal: 1 m/s^2
Jerk: -2 m/s^3
49.41
MPH
ESC
MENU