Path Planning

Reflection

The goal of this project is to build a path planner that creates smooth, safe trajectories for the car to follow. The highway track has other vehicles, all going different speeds, but approximately obeying the 50 MPH speed limit.The car transmits its location, along with its sensor fusion data, which estimates the location of all the vehicles on the same side of the road.

The whole project can be divided into 3 parts :

Prediction (lines 254 to 298 in the code)

This part takes in sensor fusion data detecting other objects around the ego vehicle and predicts what trajectory they might take in future. Usually it can be done through model or data driven or hybrid approach for more complex scenarios. For this project, I used a simple prediction technique to just detect a car surrounding the ego vehicle maintaining the lane and if it would be safe to change lanes or drop speed based on the medium article by Dhanoop karunakaran (https://medium.com/intro-to-artificial-intelligence/path-planning-project-udacitys-self-driving-car-nanodegree-be1f531cc4f7)
The code determines which lanes other vehicles are in based on sensor data and how far off they are from the ego vehicle. Then the following steps are taken:

- Check if there is a vehicle in front of the ego vehicle within 30m - Set a flag
- Check if there is vehicle to the left of the vehicle within 30m - Set a flag
- Check if there is vehicle to the left of the vehicle within 30m - Set a flag

These flags are used to determine if the vehicle should change lane or slow down in next section.

Behaviour Planning (lines 300 to 322 in the code)

This determines what behaviour the ego vehicle should exhibit at any given time.

Based on the set flags the vehicle determines if it should slow down or with lanes. (based on medium article by Dhanoop karunakaran)

If there is vehicle in front obstructing traffic:

- If there is no car to you left and you are not in the left most lane- change left
- If there is no car to your right and you are not in the right most lane- change right
- Otherwise slow down

Speed change is done in incremental way so as to not exceed the deceleration or jerk levels. The algorithm also tries to be on the centre lane and maintain the speed limit in absence of any obstruction

Trajectory Generation (lines 325 to 420 in the code)

Based on the behaviour planning output, trajectory generation module outputs the desired trajectory of the vehicle to follow. The spline library is used instead of a polynomial trajectory generation because of the simplicity. This code is based on the project walk through provided by Udacity:

- Firstly a spline is created using 2 previous points and 3 points in future at 30,60,90 m respectively.
- All the previous points are copied to the final control values
- Finally spline points are created till the horizon (30m) spacing them out to get the desired speed using the spline calculated earlier.

50 points are calculated in total (including past points) and passed on as the final control values.