Path Planning project (Highway Driving)

Goal

In this project the goal is to safely navigate around a virtual highway with other traffic that is driving +-10 MPH of the 50 MPH speed limit. The ego vehicle goes as close as possible to the 50 MPH speed limit, which means passing slower traffic when possible and also the car isn't driving much slower than speed limit unless obstructed by traffic.The car is able to change lanes and does not have collisions as well as driving inside of the marked road lanes at all times, unless going from one lane to another. Also the car does not experience total acceleration over 10 m/s^2 and jerk that is greater than 10 m/s^3. Finally, it is able to make one complete loop around the 6946m highway without incident.

The project consists of 3 main parts:

- Prediction
- Behavior Planning
- Trajectory Planning

The picture from the class material helps us to create the path planner. For behavior planning, we get the inputs from sensor fusion and use localization and prediction models to create behavoir model. The output from behavior planner goes to trajectory planner, so it sends trajectory to the mothion control.

All codes for the 3 main parts has been written in main.cpp (lines 100 to 300)

Prediction

The prediction codes consist of lines 108 to 152. For prediction I use sensor fusion data[id, x, y, vx, vy, s, d] which shows other vehicles in highway. using sensor fusion data gives us the information about position of the cars, the lanes they are driving, velocity of the cars and the positionof the cars in global map. So using this data I can predict whether the cars are placed ahead of the ego vehicle and there are cars in the left or right lane of the ego vehicle. When there is a car with distance of 30 m(behind or in front) of ego vehicle , I consider it dangerous.

Behavior Planning

The behavior model codes consist of lines 155 to 183. In this part, I am trying to change lane whenever there is a car ahead and it is safe. So if the ego car is in the right or middle lane and there is no car in left lane, it changes lane to the left. I did the same pattern when the ego car is in the left or middle lane and there is no car in right lane, it changes the lane to the right. The speed_d variable is used for speed changes and helps the car reacts faster to changing situations.

Trajectory Planning

The trajectory codes consist of lines 185 to 300. the trajectory determines which trajectory is best for executing the behavior. First I cheked if there is any prevoius points to use, so we can use the current car position for finding the pervoius point (line 193 to 203). Otherwise we use pervious two points as starting reference that make the path tangent to the ego vehicle (line 205 to 220). Also we add 3 more future points to ptsx, ptsy vecotrs; in frenet add evenly 30 m spaced points ahead of the starting refrence (line 223 to 233). Then we use spline for trajectory generation and initialize it with ptsx, ptsy and add all previous points to the new trajectory (next_x_vals and next_y_vals). Then we calculate how to breakup spline points so the ego vehicle can travel at our desired velocity (lines 262 to 264). Finally next_x_vals and next_y_vals have all 50 points (lines 269 to 300).

.

In [ ]: