CarND-Path-Planning-Project

Self-Driving Car Engineer Nanodegree Program

Output

The final Video can be observed Below:

Rubric Points

- The Code compiles correctly. - The code compiled successfully.
- The car is able to drive at least 4.32 miles without incident. - The car is able to drive across the highway easily.
- The car drives according to the speed limit. - There was no incident while testing. The car drove within the speed limit at a speed of around 49.5 MPH if there at not any traffic conditions. If there is some obstacle in front of the car, then it tries to change the lane if possible and the drive at full speed else it continues to drive at the same speed as of the car in front.

- Max Acceleration and Jerk are not Exceeded. - There was not an incident.
- Car does not have collisions. - There was not an incident.
- The car stays in its lane, except for the time between changing lanes. - The car stays inside the lane, we are just responsible for providing the lane number, the control section is handled by the simulator.
- The car is able to change lanes- Whenever there is slow moving traffic the car tries to change the lane and it does it successfully if the adjacent lane is free.

Model Documentation

The simulator provides Car's location, velocity, yaw rate, speed, frenet coordinates and sensor fusion data. We used spline to generate the trajectory. Jerk minimization techniques has also been used.

Prediction

We receive data like Car's location, yaw rate, frenet coordinates, sensor fusion, velocity and speed from the simulator. By using this data we predict the behavior of other objects in our case vehicles in future and we plan behaviour of our car based on the behaviour of the nearby objects.

Behavior Planning

I have implemented behavior planning in following steps:-

- check 30 m ahead if there is any car present or not.
- If a car is present 30 m ahead and it is slow moving, try to change the lane.
- For changing lane, I have checked whether any car is present in that lane. IF no, it is safe to change lane and If yes, it should have distance of 30 m ahead of car or 15m back of car to change lane otherwise we remain on same lane
- If no option for lane change then try to maintain a velocity of the Car moving in front.

Trajectory Generation

To compute trajectory, we use car speed, the speed of surrounding cars, current lane, intended lane and previous points. For making trajectory smoother we add immediate two points from previous trajectory. If there are no previous points, then we use yaw rate and current car coordinates to compute previous points. Then we add 3 points at 30, 60, 90 meters to the trajectory. To make mathematics bit easier we shift all the points car coordinate system and after processing, we convert them back to map coordinate system before passing those to the car.

Logic

Detecting if cars are present in the lanes

```
int lane_id = 1;

for(int i = 0;i< sensor_fusion.size();i++){
    float d = sensor_fusion[i][6];
    if (d > 0 && d < 4){
            lane_id = 0;
        }
    else if (d > 4 && d < 8)
        {
            lane_id = 1;
        }
    else if(d > 8 && d < 12){
            lane_id = 2;
        }

    double vx = sensor_fusion[i][3];
    double vy = sensor_fusion[i][4];

    double check_speed = sqrt(vx*vx+vy*vy);
    double check_car_s = sensor_fusion[i][5];

    check_car_s+=((double)prev_size*.02*check_speed);

    if(lane_id-lane == 0 && check_car_s > car_s && check_car_s < car_s + 30  ){
        ahead_car = true;
    }
    if(lane_id-lane == -1 && check_car_s > car_s - 15 && check_car_s < car_s + 30  ){
        left_car = true;
    }
    if(lane_id-lane == 1 && check_car_s > car_s - 15 && check_car_s < car_s + 30  ){
        right_car = true;
    }
}
```

Changing Lanes

```
            if ( ahead_car == false && right_car == false && left_car == false && ref_val<49.5){
ref_val += 0.224;
lane = 1;
      }
 else{
                    if(ahead_car){
                        if(!left_car && lane != 0){
                            lane = lane - 1;
                            ref_val -= 0.5*0.224;
                        }
                        else if (!right_car && lane != 2){
                            lane = lane + 1;
                            ref_val -= 0.5*0.224;

                        }
                        else{
                                ref_val -= 1.5*0.224;
                            }
                    }

                    if(!ahead_car && ref_val < 49.5){
                            ref_val += 0.224;

                    }
```