**README**

**Self Driving Car – Path Planning Project**

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

The car was able to drive atleast 4.32 miles without any incident

**The car drives according to the speed limit**

The car drives most of the time at 49.5 miles/hour. But when it detects another vehicle in its lane it decelerates to a lower speed.

**Max Acceleration and Jerk are not Exceeded.**

Max acceleration and jerk were maintained using waypoints created by spline library.

**Car does not have collisions.**

If the car detected another vehicle in its lane it reduced its own speed to cater to the slow vehicle. Then the car looked for option of changing lane.

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

The car does stay in its lane most of the time. Only when the car detects another vehicle in its lane does it change its lane

**The car is able to change lanes**

Depending on the vehicular traffic in the adjoining two lanes the car is able to change its lane.

**Reflection**

**The planner receives primarily two types of data from the simulator**

1. Self driving car’s vitals viz. x, y, s, d, yaw, speed .
2. Other vehicle vitals on the same side of the road viz. x, y, s, d, vx and vy.

The simulator sends the ego vehicle’s current x, y, s, d, yaw and speed in a telemetry message, along with the previous remaining (x,y) points that the simulator could not render in the previous run, and the sensor fusion data. The number of these points can vary randomly.

In response to the above message the path planner plans the next 50 points that the simulator should render. These 50 points are planned using 5 anchor points. These 5 anchor points describe a complete path that the simulator should follow in 1 second. Each point is traversed by the ego vehicle in 20 ms.

These 5 anchor points are interpolated to 50 points using the spline library so that the transitions from one point to another are not sudden and don’t cause a jerk to the passenger.

The 5 anchor points are calculated using equations. These equations create points using the current s and d coordinates of the ego vehicle. We space these points at a difference of 30 units along s direction and specify the d based on a lane variable.

We use the sensor fusion data in the following manner:

1. We check each vehicle to find which vehicle is in the same lane as the ego vehicle.
2. We predict where this vehicle will be i.e. the s coordinate after a specific time and compare this to the ego vehicle’s s coordinate using the end\_path\_s variable.
3. If this vehicle is less than 30 mtrs ahead of the ego vehicle then we set the too\_close variable to true.
4. The moment we set the too\_close variable to true we check whether it is possible to change the lane.
   1. We first check whether we can change to the left lane. For this we check the predicted s coordinates of the all the vehicles that are in the left lane of the ego vehicle. If we find that the predicted s coordinate of the vehicle either ahead and behind the ego vehicle predicted s coordinate then we decide against left lane change.
   2. If we don’t find any other vehicle within 30 mtrs of the ego vehicle we set the safe\_left\_turn flag to true and update the lane variable to current lane – 1
   3. As already stated the d value of anchor points is calculated using the lane variable. So the spline library now creates a path to change lane gently to left.
5. If the left turn is not possible we similarly check for right turn.
   1. We first check whether we can change to the right lane. For this we check the predicted s coordinates of the all the vehicles that are in the right lane of the ego vehicle. If we find that the predicted s coordinate of the vehicle either ahead and behind the ego vehicle predicted s coordinate then we decide against right lane change.
   2. If we don’t find any other vehicle within 30 mtrs of the ego vehicle we set the safe\_right\_turn flag to true and update the lane variable to current lane + 1
   3. As already stated the d value of anchor points is calculated using the lane variable. So the spline library now creates a path to change lane gently to right.
6. If the too\_close variable is set we decrement the ref\_vel variable by 0.224 miles. The spline generates points based on the ref\_vel. If the ref\_vel is small the generated points are spaced closely. Hence the covered distance is small. But if the ref\_vel is large the generated points by spline are spaced farther. Hence the covered distance is large.

**A successful run of the simulator for 4.49 miles using this path planner.**

