Term 3 Project 1: Path Planning

In this project we were tasked with creating and modifying the path of our autonomous vehicle in a highway situation in order to get through the course without colliding with other vehicles, by switching lanes, and by staying as close to the 50 mph speed limit as possible for the majority of the time. For the project I followed closely along with the walkthrough as it made a lot of sense to me. First, I made the vehicle go straight by just giving the simulator a constant velocity set of points. This worked but obviously the car ran off the track. Next, I got the vehicle to stay in its lane, by defining the starting lane as the center lane and then using Frenet coordinates to fix the vehicle 'd' value to the middle lane. This resulted in choppy steering. I followed the suggestion in the walkthrough and implemented the spline library to get a smoothed curvature for the curves of the road. This was done as so:

```
tk::spline s; //create a spline from the header
s.set_points(ptsx,ptsy); //set (x,y) pts to the spline

double x_point = x_add_on + target_x/N;
double y_point = s(x_point);
```

The spline library is very simple to use. The result was a smooth curve, however I kept running into cars ahead of me.

Next, I got the vehicle to drive at the desired speed limit by setting a reference velocity of 49.5 mph. To get rid of the acceleration and jerk errors, I used an incremental acceleration to get to that point, as seen in line 324, where the relative velocity increases by 0.224 mph at a time. The next step was to deal with vehicle interactions.

To deal with situations where a vehicle was going slow in my lane, the process I thought of was to first slow down, then change lane when safe, if safe. To slow down I used the sensor fusion data received from the simulator. If there was a vehicle in front of me in my lane, then I would instruct the vehicle to slow down the speed. This was done by making a Boolean parameter 'too_close' and marking it true when I was approaching another car, and decreasing the reference velocity (253-272). Then to implement lane change, I made two more Boolean parameters: LCL (lane change left) and LCR (lane change right). If indeed I was approaching a car in my lane, I would look to change lanes to the left. If there is no cars in a range of 20m behind me and 15m in front of me in the desired lane, then I would perform the lane change. If either there was a car in that area, or I was in the left most lane, then I would prohibit the left lane change by setting LCL to false. If that was the case, I would look at changing lanes to the right. The same determination was done with this case. The code for checking cars on the right looks like below:

```
else {  //prepare for a lane change right
     LCR = true;
     if (lane<2) {
        for(int i=0;i<sensor_fusion.size();i++) {   //loop through objects
           double left_d = sensor_fusion[i][6]; //get distance d to check the lane
           int lane_des = lane+1;
```

```
            if(left_d<(2+4*lane_des+2) && left_d>(2+4*lane_des-2)) {    //if there is a car
in the left lane
               double switch_car_s = sensor_fusion[i][5]; //get the car's distance
               if(switch_car_s>(car_s-20) && switch_car_s<(car_s+15)) {    //there is a car
in the way.
                  LCR=false; //don't change lane
               }
             }
          }
       }
```

Once again, if the car was in the right most lane, then a lane change to the right was prohibited. For either case, if the lane change was decided, then the lane value was adjusted in the correct direction: lane -= 1 for a lane change left, and lane +=1 for a lane change right. This was then used in the spline function to find a smooth lane change.

This gave me a working solution. I played with the distance values and the speed values for changing lanes to get the best solution I could. Overall, I ran the simulation multiple times, working my way up step by step. This really simplified the project. The walk through was very helpful in getting started in the right direction.