John McConnell
Udacity Path Planning Project

In this project the goal was to create a “behavioral planner” that would generate a vector of points to control the car. These points should not only navigate the car around safely but efficiently. This was accomplished in 3 main blocks of code, first the parsing of the sensor fusion data, followed by the cost function, next the safety checks, and lastly generating the output points.

In order to parse out the sensor fusion data I used a for loop. Inside this for loop I did a variety of things, check to see if there was a car to close to us in our lane, if it’s safe to change lanes left and right, etc. This block of code as a generalization accomplishes two things, changes the pertinent boolean flags to true or false and compiles vectors of cars ahead of us for the cost function.

The cost function is the core of this code and can be seen in “cost.h.” This code takes the previously prepared vectors of cars forward of our car and decides which lane is best to be in. The cost function is very simple, it is trying to find the lane with the most free road between the car and the car from the sensor fusion vector. It then returns this to main.cpp and it later taken into account.

The next block is safety checks, the boolean flags that have been set in the previous block are looked at here. First I am checking to see if there is any need to conduct an emergency stop or just a regular slow down. Next the lane we are in is considered, if we are not in the optimum lane then we need to change. In order to execute this change the safety checks must be fulfilled, these are if the lane we are changing to is safe (no cars to hit), the car is going at highway speed (over 40 mph) and we are not in the middle of lane change. These safety checks help complete some key points of the rubric, not hitting other cars and not driving outside of a lane.

The last block is mostly from the project video, this is where I generate the list of points for the controller. This is pretty straightforward, first I am generating some anchor points and fitting a spline to them. After that I am just generating some points using the spline library. There are some subtleties here like shifting the perspective to the car to avoid some issues with the spline.h library. The great thing about this code is it allows me to change speed and lane on the fly with the addition of the code above it.