# Path Planning

## 1. Files Added, Modified and Submitted

- main.cpp – the modified main program
- spline.h – the added spline interpolation header file

## 2. Reflection: How to Generate Paths.

The given start code mainly shows the transform between Cartesian (x,y) coordinate and Frenet (s,d) coordinate, and how to identify the index of closest waypoints before it goes to main function.

Detailed comments are provided to the rest of the initial part to improve the code readability.

### 2.1 Prediction.

The loaded telemetry and sensor fusion data empower the calculation of other vehicle's position, and then feed to different lane cases and traffic scenarios in other lanes, including:

1) identify which lane we are according to different d value
2) whether there is a car ahead if the ego car is in middle lane
3) whether it is safe to turn to the left lane if no other car within 30 meters' range in that lane
4) whether it is safe to turn to the right lane if no other car within 30 meters' range in that lane

A car is considered "dangerous" when its distance to the ego car is less than 30 meters' range (ahead or behind).

### 2.2 Behavior Execution

The corresponding behavior to different cases include:

1) when there is a car ahead,
    a) change to the left lane if no car in the left lane and left lane is available
    b) change to the right lane if no car in the right lane and right lane is available
    c) decelerate velocity to avoid collision if change lane is invalid

2) when there is no car ahead
    a) back to the middle lane if not in the middle lane
    b) accelerate velocity to go fastest possible if not reach maximum speed limit and already in middle lane

### 2.3 Trajectory Estimation

the location estimation of trajectory points between the known waypoints is achieved by interpolating the position of those points, this can be done by fitting polynomials to waypoints, which use a polynomial function to interpolate the location of a new point, or another method introduced: spline fitting, which guarantees the generated function passes through every point.

First, the last two points of the previous trajectory points are used in conjunction three points at a far distance to initialize the spline interpolation. Note the coordinates are transformed (shift and rotation) to local car coordinates.

The new path starts with whatever previous path points were left over from the last cycle, then append new waypoints, until the new path has 50 total waypoints.

Using information from the previous path ensures that there is a smooth transition from cycle to cycle. But the more waypoints we use from the previous path, the less the new path will reflect dynamic changes in the environment. Ideally, we might only use a few waypoints from the previous path and then generate the rest of the new path based on new data from the car's sensor fusion information.

Lastly to notice, the speed change is used to increase/decrease velocity on every trajectory points instead of doing it for the complete trajectory.

## 3. Discussion

From the executed simulator, the car is driving alone the road with no collision, max acceleration and jerk are not exceeded, and it stays within its lane, except for the time between changing lanes, which addressed the project rubric requirement. However, when there is a car ahead and change lane condition is invalid, the ego car try to act as follow the car ahead but actually keeps accelerating and decelerating all the time, making it annoying and uncomfortable if it’s in the real world scenario, a more detailed and specific behavior strategy is expected to make the car auto follow with the same speed of the car in front. Moreover, the flexible safety distance parameter can be further fine-tuned according to various traffic situation to avoid missing the short time window of change lane.