# Path Planning Project

Report

Author: Aneeq Mahmood

Email: aneeq.sdc@gmail.com

Steering a car along the track using path planning techniques

## Generation of Path Points:

There are two main steps involved in driving the car on the highway. Firstly, in which lane to drive the car in, and secondly, if a certain lane is provide how to generate points so that the car follows the path.

**Choosing the Lane:**

This part is based on a finite state machine (FSM), who has only two states namely keep the lane (KL) or change lane left (CLL) /change lane right (CLR). There has been no explicit prepare lane change (PLC) state, which could make the car drive down to the speed of the new proposed lane and then to CLL or CLR. Reason being that other cars in the simulator can handle their speed individually, and they can react well (most of the time) to Ego's maneuvers. The only precaution made in the code is to check that there is enough path left for the car coming from behind in the new proposed lane, so that it can see Ego vehicle changing its lane. This avoids collision from behind which serves the purpose for the project.

The actual FSM works like this: Ego starts at 0 velocity in its own lane (Lane 1), and checks the path ahead over a distance LANE_HORIZON using the *check_fwd_behaviour()* function. If the distance is free from other cars, Ego picks up speed and reaches up to the reference velocity which is set to 45 MPH to completely avoid any speed violation. If there are vehicles ahead, then *check_costs()* function calculate the costs of staying in the same lane or changing to left or right.

**Cost Function:**

The cost function used here is $J = 1 - e^{\frac{-\Delta d}{\Delta s}}$

Where Δd is the lateral distance for changing a lane, and Δs is distance between Ego and a car front (or behind). In the FSM, only a single lane change is allowed, which implies that a car in lane 0 cannot change path directly to lane 2. The FSM will first see the feasibility to move to lane 1 and then if desired move to lane 2. Because of this limitation, Δd will always be approximately 4m which is the lane length. Because of its fixed nature, Δd is set to 1 in the simulation as it is affecting cost calculations for any state in FSM the same way.

**Analyzing Future Lane:**

In the *check_cost()* function, available distance in front and behind of Ego for the proposed lane is checked.. If this distance is greater than LANE_HORIZON, $\Delta s$ will be set to a large number, indicating that there is enough room for the car to change lane; otherwise the actual distance ahead and back is used to calculate the cost. This cost is then compared later, with the cost of staying in the same lane by using the distance between Ego and its nearest car ahead. Based on the smallest cost, Ego either remains in the lane and adjusts speed or changes the lane. Afterwards it picks up speed again to get to the reference velocity.

**Generating the Path:**

Once the future lane is decided, path points are generated using the spline algorithm discussed in the project Q&A video. The code is based on this video and generates 5 anchor points. The first two anchor points are from Ego's old and current position. The next three points are drawn ahead of the car in the same or new lane (based on the FSM). Using these anchors, spline algorithm is activated. Afterwards, over a distance of PATH_HORIZON, new future x-points are created using the car's speed and simulation tick interval (20ms), and spline creates the corresponding y-coordinates. Using a combination of these new points and path points from previous iteration, a path is produced.