Self-Driving Car: Path Planning

1 Objectives

The car should be able to successfully cruise in traffic without colliding with other vehicles while performing occasional lane-changing or speed adjustment in its lane. Car motion jerk and acceleration should be confined below a certain bound and speed should not exceed the designated limit (i.e., 50 mph). Furthermore, lane changing should be fast enough to prevent off-lane detection (in which case typically other vehicles lose sight of the car and potentially collide with it).

2 Method

The idea behind my solution is to create a general function that plots trajectories from one lane to another in order to provide the planner with a mechanism to implement its "intentions". This function can thereafter be used by a higher-level planner, which, based on traffic in close proximity of the car, makes decisions in the behavioral layer on whether the car should keep or change its lane. To do this, the planner generates a number of candidate trajectories (plans) and chooses the one that minimizes a cost function involving criteria such as collision, acceleration and speed.

2.1 Generating a Curve for Lane-Changing

Consider the car at a world position (x_1, y_1) with speed υ_1, orientation θ_1 and track coordinates (s, d) and suppose that the target position is the center of a lane at (s_2, d_2). To define the motion well, we assume that the velocity vector at the end of the maneuver will be parallel to the s-axis. Thus, it is necessary to specify the speed υ_2 along the s-axis. Another degree of freedom that remains to define the motion is the distance traveled in the s-axis Δs.

The resulting spline is designed in *world coordinates*. Although it is possible to generate a spline in s-d coordinates, it would however produce a discontinuous curve in world coordinates, due to the use of linear segments to interpolate between the waypoints of the track. Thus, instead of generating a dense set of points in s-d coordinates, we instead obtain a sparse set of 3-5 points, which are then transformed into real-world coordinates. These transformed points are then interpolated by a spline parametrized by time.

To generate the sparse s-d point-set, we assume that the vehicle is constantly accelerating (or decelerating) in the s-direction. Thus, its s-coordinate through time will be given by:

$$s(t) = \frac{1}{2}at^2 + \upsilon_1 t + s_1 \tag{1}$$

Note that, although υ_1 may not entirely be along the s-acis, it is reasonable to assume at this stage that the car is moving only along the s-axis.

Having specified a target speed υ_2 and a distance position s_2 in s, we may obtain the duration of the motion:

$$T = \frac{2\left(s_2 - s_1\right)}{\upsilon_1 + \upsilon_2} \tag{2}$$

Now, using eq. (2) we may compute the acceleration in the s-axis:

$$a = \frac{2\left(s_2 - s_1 - u_1 T\right)}{T^2} \tag{3}$$

With the above in place, it is now easy to generate the sparse set of s-d points by taking points $s_i \in (s_1, s_2)$ where $i = 1, 2, \ldots, n$ such that:

$$s_i = s_1 + i\frac{s_2 - s_1}{n} \tag{4}$$

We thereafter solve for the respective time instances t_i [1] from eq. (1). The d-coordinates of the s-d points are chosen uniformly between the starting and ending coordinates in d:

$$d_i = d_1 + i\frac{d_2 - d_1}{n} \tag{5}$$

[1]Note that $s\left(0\right) = s_1$.

We thereafter transform each pair (s_i, d_i) into a pair of world coordinates (x_i, y_i) and generate *2 splines* parametrized by time using the spline tool over the following pairs of points:

$$x\,(t) \quad = interpolate[(t_0, x_0), (t_1, x_1), \ldots, (t_n, x_n)] \tag{6}$$

$$y\,(t) \quad = interpolate[(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)] \tag{7}$$

Note that in practice, to achieve tangent matching at the boundary, we simply generate extra s-d points in close proximity of the finishing point, while in the beginning, we simply push a few of the points that were not previously executed by the simulator for continuity [2].

2.2 Choosing the Optimal Plan

The planner generates multiple splines using combinations of various end-speeds and distances in the s-axis. In particular, the speed values are multiples of the speed limit,

$$\{0.1\upsilon^*, 0.2\upsilon^*, \ldots, 1.2\upsilon^*, \}$$

, while distances are,

$$\{10, 15, 20, 25, 30, 40, 50, 70, 90, 110, 120\}$$

Each combination defines the duration and of the overall motion and the acceleration a in the s-axis. The planner penalizes acceleration severely (by a factor of 5) when the target lane is the same as the current lane to reduce jerk and acceleration alerts. When changing lanes however, acceleration is mildly penalized (by a factor of 0.5), while long durations are also penalized (factor of 1 in the cost function) to encourage faster lane changing with less jerk and acceleration.

2.3 Behaviors

The behavior of the planner is simple. It either keeps its lane or decides to change lane, if one is available. The main criterion that imposes lane change is the speed level. If the speed level stays below 75% of the speed limit for

[2]The fact that **the simulator actually returns** the list of points that were not previously reached should be more explicitly stated in the project description.

140 consecutive simulator feedback cycles, then a lane change is issued. The planner then examines the neighboring lanes and if one is empty, it sets it as the target lane; in any other case, then target lane remains the same and the low-speed counter is reset.