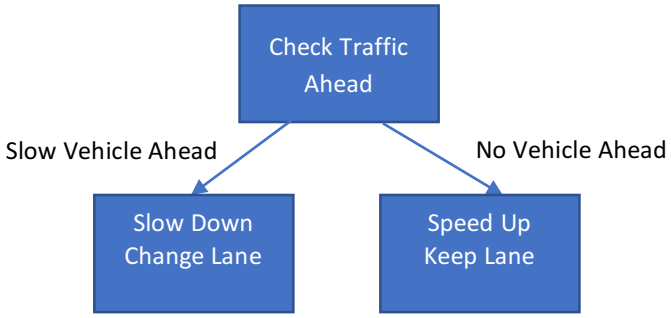
PATH PLANNING PROJECT WRITEUP

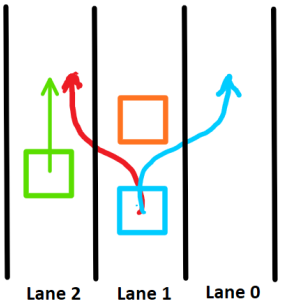
For the vehicle to navigate the virtual highway, the vehicle required a ***behavior planner*** and ***trajectory generation*** module. These modules are explained in the following sections.

BEHAVIOR PLANNER

The behavior planner determines whether the vehicle should accelerate, decelerate, or change lanes. The logic behind the behavior planner is captured in the decision tree below.



The behavior planner begins by analyzing traffic in the same lane as our vehicle (see the method ***checkTrafficAhead***). If no vehicle is ahead, then our vehicle will keep its current lane and accelerate until it reaches the target speed of 49.5 mph. Conversely, if a vehicle is detected and its speed is less than our desired target speed of 49.5 mph, then our vehicle will slow down to avoid a collision. Furthermore, our vehicle will attempt to change lanes according to the logic explained diagrammed below (see the method ***findNewLane***).

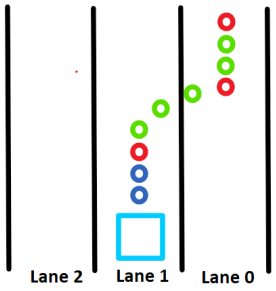


In the figure to the right, we see a bird's eye-view of a three-lane highway. Our vehicle (blue square) is in Lane 1 behind the slowly moving orange vehicle. Consequently, our vehicle determines that a lane change is necessary. Our vehicle examines each of the remaining lanes (Lanes 0 and 2) to determine which could accommodate a lane change. Our vehicle determines that Lane 2 is not suitable since the green vehicle in Lane 2 will occupy the same location as our vehicle upon completing the maneuver (see green and red arrows denoting vehicle paths). On the other hand, no other vehicle will occupy our projected final location in Lane 0 (see blue arrow) after the lane change. So, our vehicle will choose to change lanes into Lane 0.

TRAJECTORY GENERATION

The trajectory module generates points along the path that our vehicle should follow. To construct this path, we begin by adding all the points not yet visited from the previous path our vehicle was following (see method ***addPreviousPathPoints***). We then add new points sampled from a spline that extends the previous path and

passes through anchor points. The anchor points include the endpoint of the previous path as well as points 30, 60, and 90 meters away from that endpoint (see method ***getAnchorPoints***). The points sampled along the spline are spaced in a such a way that visiting one every 20 ms forces our vehicle to travel at a desired speed (see method ***interpolateSpline***).



The figure to the right illustrates the trajectory generation process in the setting of our vehicle (blue box) attempting a change from Lane 1 to Lane 0. The first points of the trajectory are the points remaining from the previous trajectory (blue circles). Next come the anchor points that define our spline (red circles); note how two of the anchor points are in lane 0 which ensures that we'll create a path that leads us into Lane 0. Once we fit the spline to the anchor points, we sample the spline to get the green circles which give us a smooth path from Lane 1 to Lane 0. Sampling the spline was achieved using the same linear approximation procedure described in the project walkthrough video.

VEHICLE IN ACTION

The image in Figure 1 shows our vehicle driving in the center lane with no traffic ahead. Consequently, our behavior planner instructs the vehicle to accelerate until it reaches the desired speed of approximately 49.5 mph. In Figure 2, our vehicle finds itself behind traffic and cannot change lanes due to the fast-approaching red vehicle. So, our vehicle decelerates to 36.5 mph in order to match the speed of the blue vehicle ahead and avoid a collision. In Figure 3, our vehicle again finds itself behind traffic but this time middle lane is free of traffic. So, our vehicle proceeds with generating a trajectory that will lead it to the middle lane.

Figure 1

Figure 2

Figure 3