Udacity – Car self's Diving Nanodegree Path Planning Project

Author: Bertrand Cabot

Goal : This note is a reflection on how to generate paths in my project.
I follow in the main line what we learn in the courses.

1. **I create one precise waypoints reference**. In order to transform s, d frenet coordinates in global x,y coordinates.
   I Load up map values from system (who has a low resolution of 30 m). I use the spline library to interpolate values. For the d axe, I calculate a theta angle which gives a direction for d axe. I interpolate it. And I center my waypoints referential in the middle of lane #1. So, I minimize error of translation between frenet referential and global referential.

2. I **memorize at each step the last state of trajectory** sent to the system. State means here: s, d positions, velocities and accelerations. So I keep more information than ones sent by the system. So I can have a smooth trajectory at each step of the play without gap between 2 times. Here, I could use object implementation to do that. But, I just use a vector container.

3. I determine **a target position and a target velocity** according to state of ego vehicle and position of other vehicle.
   If state is "KL", "LCL", or "LCR", I search the nearest ahead vehicle (in a 50 m window) in the concerned lane and I target position and velocity of this one. If there is no vehicle in the 50m window, I target a 70m ahead position and a velocity close to speed limit.

If state is “PLCL” or “PLCR”, I search the nearest vehicle in a +/- 20 m window in the destination lane. Then I detect the presence or not of a vehicle between 20m and 50m ahead in the destination lane. If there are, I target the position and velocity of nearest vehicle; if there are not, I target a 70m ahead position and a velocity close to speed limit in order to overpass the nearest vehicle.

4. I generate **alternative trajectories** with goal error defined by a random function and with different time to reach the goal. Then for each start/goal couple, I generate a trajectory with the polynomial Jerk Minimizing Trajectory function.
   For each time gaps, I generate 4 alternatives trajectories, due to the limitation of my system. But I would like to generate about 10 trajectories. ”10” seems to me a good number for alternatives trajectories.

5. Finally, **I choose the best trajectories** regarding to different checking properties :
   - time to reach goal
   - goal gap on s axe
   - goal gap on d axe
   - celerity of trajectory
   - collision
   - security buffer on s axe
   - security buffer on d axe
   - stays on the 3 lane road
   - exceed the speed limit
   - velocity always positive on the s axe
   - d lateral velocity under a threshold
   - exceed maximum acceleration on s axe
   - exceed maximum acceleration on d axe
   - exceed maximum jerk on s axe
   - exceed maximum jerk on d axe

- exceed a comfort threshold of jerk on s axe
- exceed a comfort threshold of jerk on d axe

I have configured the weight of cost by testing and by thinking about the priority of each property.

Limitation:

Sometimes, I can have some collision because of the behavior of the other vehicle. Or sometimes, if vehicle is on the lane #2, I could touch the limit of the road. However, I can reach 10 miles or more without incident.

And I am limited by the slowness of my system or the heaviness (lack of efficiency) of my functions. I would like to test 10 alternative trajectories by time step. But I can't : the trajectory is so not upgrade before the end of the previous sent one. That triggers some jerk limitation incident.