Implement a basic driving agent

Question 1: In your report, mention what you see in the agent's behavior. Does it eventually make it to the target location?

Answer: It took lots of time for the agent to get to the right place.

Identify and update state

Question 2: Justify why you picked these set of states, and how they model the agent and its environment.

Answer: I picked inputs and next_waypoint, and I combined them to be a state. The inputs include the current road situation including lights, oncoming, left and right. If the car cause an accident or go through the red light, then it will be punished. If the car obey the traffic rule and get closer to the destination, then it will be awarded. It's important for the car to learn the traffic rules in this way. The next_waypoint tells the agent how to get closer to the destination, which is essential for the agent to take right action.

Implement Q-Learning

Question 3: What changes do you notice in the agent's behavior?

Answer: At first the agent failed to get to the target location. After making trails for several times, the agent can reach the destination. The agent behaved better than pick random choice. It took fewer time to arrive at the destination than before.

Enhance the driving agent

Question 4: Report what changes you made to your basic implementation of Q-Learning to achieve the final version of the agent. How well does it perform?

Answer:
I randomly set gamma to be 0.6 and alpha to be 0.5. Then I fixed the value of gamma and adjusted alpha from 0.5 to 1.0. When alpha equals 0.7, I get the highest successful rate (successful time/total trails). So I set alpha to be 0.7 and adjusted the value of gamma. When gamma equals 0.3, I got the highest successful rate (97%).
Finally, I set the learning rate alpha to be 0.7 and set the discount factor gamma to be 0.3. I also set an exploration probability so that the agent will have chance to find a new way to get to the destination. If the agent always choose the best action for current situation, it will lose the chance to find a better route. I also added a counter to count the number of trials and set it to be 5. I used the first 5 trials to explore different ways and allow the agent to make mistakes and then the agent should choose the action which maximize the q-value.

Experiment results:

Condition: gamma: 0.6, alpha:0.5~1.0

alpha 1.0, gamma 0.6: 88%

alpha 0.9, gamma 0.6: 93%
alpha 0.8, gamma 0.6: 94%
alpha 0.7, gamma 0.6: 94%
alpha 0.6, gamma 0.6: 91%
alpha 0.5, gamma 0.6: 88%

Conculsion: alpha = 0.7

Condition: alpha:0.7, gamma:0.1~0.8

alpha = 0.7, gamma 0.8: 88%
alpha = 0.7, gamma 0.7: 93%
alpha = 0.7, gamma 0.6: 93%
alpha = 0.7, gamma 0.5: 95%
alpha = 0.7, gamma 0.4: 95%
alpha = 0.7, gamma 0.3: 97%
alpha = 0.7, gamma 0.2: 96%
alpha = 0.7, gamma 0.1: 95%

Conculsion: gamma = 0.3

Question 5: Does your agent get close to finding an optimal policy, i.e. reach the destination in the minimum possible time, and not incur any penalties?

Answer: If the agent find the optimal policy, the ratio of successful times (reach the target destination) in total trails should be very high. Meanwhile, the agent will take less steps to make it and get less penalties and more awards. From the experiment data, which is a PDF file included in the zipfile called trail_data_alpha0.7_gamma0.3, the agent failed 3 times in 100 trails and kept getting rewards. I think the agent has found an optimal policy.