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

Answer: Agent behaves just like dummy agent, which go to random direction or don't move. Besides, it violates traffic rules and crash into other cars. Yes, it can make it to the target after all.

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

Answer: I pick the following states:

lights: to give penalty if agent violates traffic rules

oncoming: left, right: to model the state of other cars at the intersection where the agent is. If the agent makes a crash, give it a negative reward.

next_waypoint: to see whether the agent is going towards the goal. If true, give it a positive reward, else, punish it.

I didn't choose the 'deadline' as a state variable. Because the goal of the agent is to reach the end as quick as possible without violating traffic laws. The 'deadline' variable plays little role here. Besides, if I adding the 'deadline' variable, the state space will be too large and sparse, which is quiet hard to be trained to converge.

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

Answer: The agent can finally reach the destination rather than randomly take action. The agent still makes some mistake at the beginning, for example circling around instead of going towards the destination. Because the delayed reward when reach destination makes less effect on the beginning of the action sequence, comparing to the end, result from the discount factor. However, actions like crash into other cars can hardly be seen, because immediate punishment can directly affect the Q table. So when facing the same state next time, the agent will choose the highest score action and the action may lead to punishment will not be chosen.

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: After trying 25 combinations, it comes out the LEARNING_RATE=0.7 and GAMMA=0.8 can result in best model, which reach the end for 10 times per 10 trial with highest score represents the fewest time of violating traffic rules. According to the Q-values algorithm, the higher LEARNING_RATE means learning fast, in other words, the current result has more weight comparing to old Q value. And higher GAMMA means higher discount factor, means the present Q-value is more relevant to the estimate of optimal future value. So that the delayed reward can effect longer towards the front of action sequences.

Considering the 100 times trial is small, besides to reach the destination, high value of

LEARNING_RATE and GAMMA is reasonable. It performs good.

The reward values I set are as following:
1, If agent go in right direction, reward+2
2, If agent go wrong direction, reward-5
3, If agent's action violet traffic light or crash, reward -8
4, If agent's facing a red light and choose to stay and wait, reward not change
5, If agent's not facing a red light but choose to stay, reward -3
6, If agent reaches end in deadline, reward+10

The trying result of different LEARNING_RATE and GAMMA. The first item in tuple is the times of reaching end in 10 trails, and the second is the total scores of 10 trials.
LEARNING_RATE : L GAMMA : G

	G=0.50	G=0.60	G=0.70	G=0.80	G=0.90
L=0.50	10,-21	9,-63	10,-102	9,-42	10,-21
L=0.60	10,-115	9,-112	9,-112	10,-55	10,11
L=0.70	9,-87	10,-2	10,-4	10,20	10,-32
L=0.80	9,-157	10,-112	10,-139	10,-174	10,17
L=0.90	8,-196	10,-12	10,7	10,-66	10,-147

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: It gets close to the optimal policy, but still makes some mistakes. On the one hand, agent may disobey traffic rules a few times. When the road is clear, it performs good, while facing other agents in a crossroad, it may crash. I think it due to the reward/punishment setting. I set punish for waiting, besides the states that my agent facing other agents is comparing rare, so my agent would choose to crash rather than wait. On the other hand, my agent cannot find the shortest way every time, instead, go around in circles. Again, I think the reason is the punishment of waiting makes it would rather go around than stay and wait. However, the punishment of waiting is reasonable. Without it, agent would stay for a few time rather than move.

So I think the improvement is tuning the reward more elaborately. I also think the 100 times of training is not enough, which directly result in the disobey and go around in circle. The LEARNING_RATE and GAMMA are ideal, which are chosen from enumeration.