In your report, mention what you see in the agent's behavior. Does it eventually make it to the target location?
When agent given unlimited time to reach destination:

- **mention what you see in the agents behavior**
  - The agent has no state, no action, and reward of 0.0 at every time step.
- **does he make it to target location**
  - Initially there is no q training being implemented with the agent. Nor any other forms of getting to the target location. Overall since the agent has no action, it remains still and only 'reaches' target destination, if it is randomly selected to start there.

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

- My state set includes the traffic light, right incoming traffic, left incoming traffic and the next_waypoint which is the variable that determines the direction to the destination. The next waypoint is important because it refers to the heading according to the current location. To further optimize, I include the traffic light and incoming traffic to take into account traffic laws. I avoided the deadline in the state to keep the state space
  at a reasonable level.

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

- The behavior of the agent improves since it states 'Primary agent has reached destination!' with higher and higher frequency from its first run. Initially the agent does make random movements if and before reaching it's destination. It's learning to follow the next_waypoints and training laws, although the agent acts slightly sporadically due to the epsilon greedy algorithm included. Furthermore it chooses actions based on a epsilon random chance, else the max arg portion of the algorithm. Therefore there's epsilon chance that the agent will always choose an arbitrary action.

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?

- First I configured the two variables in the q learning equation. For the discount, I knew reaching the destination has a very high reward, so I gave it a high value of .99. Next, I configured the step size a few times until I was getting more than 7 successes. Finally I introduced the e-greedy algorithm which improved the frequency of him reaching his destination. The latter is due to the agent utilizing both exploration and exploitation. First I tried .2 for the epsilon variable and introduced randomness that led to average 2 errors in the last ten trials. Next I tried .3 for epsilon which actually increased the errors. When I changed the epsilon to .1, I noticed the last ten trials usually has no errors. Therefore I set epsilon as .1.

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?

- The agent get's close to finding an optimal policy. This is due to him learning from his environmental state. He has high efficiency and reaches his destination successfully an average 8 out of 10 times. In the last ten trials the agent does not

follow all the traffic laws consistently. Although the agent is making less errors. For example in the first few trials, the agent makes around 80 while the last few trials the agent averages around 30 for the penalty variable. The penalty errors could be a cause of the epsilon greedy algorithm where the agent is able to explore randomly versus only following the q algorithm. Finally the agent constantly takes indirect routes. Again this could be because of the epsilon greedy algorithm theres a 10 percent probability that the agent chooses his action randomly.