Nathan George
Smart Cab Project Report
7-13-2016

Implement a basic driving agent

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

It eventually does make it to the target location, but each movement is random, of course. It doesn't take into account the reward from it's actions, so it will sometimes run red lights, violate right-of-way, and often does not follow the waypoints.

Identify and update state

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

I picked these for the set of states:
next_waypoint: [None, 'left', 'right', 'forward']
light: ['red', 'green']
'oncoming': [None, 'left', 'right', 'forward']
'left': [None, 'left', 'right', 'forward']
'right': [None, 'left', 'right', 'forward']

The next_waypoint models where the car should go according to the planner, and helps the agent determine which moves to make to most efficiently reach the target. Following the waypoints gives a reward, so that helps the agent learn. The light, oncoming, left, and right traffic variables model the agent's immediate surroundings and help it make good driving choices. The light helps it learn to not run red lights (since it is penalized for running a red light), the oncoming, left and right traffic variables teach it how to obey traffic laws (since it is penalized for violating right-of-way). For example, the agent should not turn left if there is an oncoming car, and it will be penalized if it does so.

Implement Q-Learning

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

The agent appears to learn how to drive efficiently after the first route is completed. It follows the waypoints almost all of the time. However, it still makes some nonsense moves, such as not following the waypoints, breaking traffic rules, and sometimes not moving when it should. The penalties from running red lights and breaking right-of-way laws help it avoid those actions, while the rewards for following the waypoints help it follow the proper path.

Enhance the driving agent

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?

I tuned the alpha and gamma parameters by running trials, then varying alpha from 0.1 to 0.9. I ran 10 trials for each alpha, then took the maximum of the total penalty score (which is negative) to find the best alpha. I did the same thing for gamma (setting alpha the best alpha I just found), and then the same thing for alpha with gamma set as the best found gamma. I found 0.2 to be best for alpha (barely, compared with 0.8). The best combination I found was alpha = 0.1, gamma = 0.8. See 'tuning alpha

and gamma.txt' for tabulation of the scores. If I were to go back and do this again, I would report the total score (including positive rewards) and probably try to maximize that.

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?

Yes, it basically finds the optimal policy with a few exceptions. A mistake is made in about 10% of the last 10 trials. I compared the waypoint direction to the action direction, and found it almost always is identical, with a few exceptions near the end of the 100 trials. In about 10% of the last 10 trials, about one of the moves is non-ideal (action does not match waypoint). See the test1.txt, test2.txt, test3.txt files for logs.