Project Report

Implement a Basic Driving Agent

QUESTION: Observe what you see with the agent's behavior as it takes random actions. Does the smartcab eventually make it to the destination? Are there any other interesting observations to note?

The agent will make silly decision in intersections. Instead of approaching its destination, it is more likely to drive away or bump into other agents. There is a few case that the agent makes it to the destination in allotted time, but the probability is negligible. This agent is vulnerable to traffic jam, when all the other three ways have a cab coming, It will take a long time for it to make a decision, just like what a road novice in real life will do in an intersection.

Inform the Driving Agent

QUESTION: What states have you identified that are appropriate for modeling the smartcab and environment? Why do you believe each of these states to be appropriate for this problem?

The states I pick are traffic light state, the intentions of other agents on the three other ways, the next waypoint that the planner suggests, deadline excess. They are the states directly affects agent's reward in this environment. Below are reasons for each state:

- *Traffic Light: Traffic light indicates whether the agent should cross the line, the agent should first obey this rule. And the environment has relative penalty/reward for traffic light reaction, therefore it's possible to learn traffic light behavior.*
- *The Intensions of the other agents: This state is separated into three states in the code. In real life, other cars will tell us where they want to go by flashing*

their cornering lamps, so that we can make further decision than just based on the traffic light. The agent should learn how to react to different situations in intersection.

- The Next Waypoint: The next waypoint suggests the next way the agent should go in order to get to the destination as soon as possible and avoid risking bumping into other agents. For the agent to learn, the environment should assign penalty if the agent tries to go the way that's not suggested.
- Deadline Excess: A cab driver should pick route that similar to what the planner suggests, otherwise, it will waste gasoline and passenger's time which reduce satisfaction or probably, tips. The agent should take penalty if it wanders too much around the town. Deadline in real life is not concrete except the case in which the passenger is going for movie or meeting, etc. So it's better to make it a Boolean value that illustrates whether the agent is late. This is also better than continuous number so that it shrinks the states space in Q learning and increase the learning efficiency in other behaviors.

OPTIONAL: How many states in total exist for the smartcab in this environment? Does this number seem reasonable given that the goal of Q-Learning is to learn and make informed decisions about each state? Why or why not?

According to the statements in the previous question, there're 2 possible values for traffic light, 4x4x4 for 4 different intentions and 3 other agents, 4 possible next way points and true or false for deadline excess. So it is 1024 different states and 4096 (s, a) tuples for this Q-learning problem.
4096 is quite reasonable. Supposed we train the cab in a city where we will meet 10 intersections on a trip at average. Particularly, in a grid like city with heavy traffic, which provide sufficient information to learn with, the agent should converge quickly. My agent still meets a few new states on the 1000th trial sometimes, but it has quite converged before the 100th trial. So this number of states is possible to get the agent learned to make informed decisions.

Implement a Q-Learning Driving Agent

QUESTION: What changes do you notice in the agent's behavior when compared to the basic driving agent when random actions were always taken? Why is this behavior occurring?

I pick the next waypoint the planner offers as the exploration source in the training. Therefore at the beginning, the agent won't act the silly way the random one did, but still makes a lot of trouble to the traffic since all it wants is to get to the destination. After a couple of trips, the agent started to make less mistake in the intersection. More trips give better decisions. The agent makes a lot of mistakes at the beginning is because the Q table is all zero, the agent can recognize good or bad behavior. As the training goes on and the agent gets penalty for mistakes and reward for good moves, the Q table is gradually collecting these information for later use, and according to the table, the agent chooses the action that could give a best Q value, i.e. the agent tends to make better decision the situation has been seen before.

Improve the Q-Learning Driving Agent

QUESTION: Report the different values for the parameters tuned in your basic implementation of Q-Learning. For which set of parameters does the agent perform best? How well does the final driving agent perform?

The parameter I pick are gamma = 0.6, alpha = 0.1, n trials = 200. With this set of parameters, the Q learning is basically converged before the 100th trials with 50 agents on the map. After the 100th trials, The agent still make mistakes but the frequency is acceptable. The agent will explore the next waypoint the planner offers in priority, which make it easy to success when other issues is handle. The success rate is satisfactory overall.

QUESTION: 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? How would you describe an optimal policy for this problem?

Yes, it is close to finding the optimal policy, because so far, the agent can avoid most of the risk in the intersection and it will arrive in time. It's hard to say if it had made the time minimum due to different traffic conditions, but the success rate is good enough.

For the smartcab problem, an optimal policy should make the trip safe first, and simultaneously tries to approach the destination and reach there as soon as possible, just like what we do in real life. More strictly, the agent should be smart enough to make some decisions like avoiding going to the blocks with busy traffic or switch to a better plan when it saw heavy traffic.