**Write Up for Smart Cab project**

4/22/2016

Behavior under random action

When I direct the agent to perform randomly, the red car moves randomly, without regard to the destination. I left the simulation for 5 minutes and the red car eventually reached the destination.

Starting state variables and justification

For initial state variables I chose the following:

‘location’: location of the agent

‘destination’: destination of the agent

‘heading’: direction of the agent

‘light’: direction of the light (green if direction same as agent)

‘oncoming’: True if there is traffic oncoming

‘left’: True if there is traffic on the left

‘next\_waypoint’: next waypoint for the planner

‘location’ and ‘destination’ are necessary in order to map routes. ‘heading’ is needed in order to connect actions to movement on the board. ‘light’, ‘oncoming’, and ‘left’ are necessary to check whether an action is valid. ‘right’ is not included because it has no bearing on whether an action is valid. ‘next\_waypoint’ is necessary because it can bestow a reward.

Behavior after implementing QLearn while choosing best action

Gamma (discount) = .5

Alpha (learn rate)= .2

When I directed the agent to always choose the best action based on Q value the agent eventually decided to always choose not to act and receive a reward of 1.

Behavior after implementing QLearn while choosing best action with some randomness

Gamma (discount) = .5

Alpha (learn rate)= .2

Epsilon (explore rate)= .2

Deadline enabled

I added an epsilon for exploration in my next attempt. This does considerably better. The agent chooses to act this time. Unfortunately I’m not close to a feasible policy and the agent will still take random steps even after the ideal path has been discovered because the epsilon is constant.

Changed state variables

At this point I decided I had too many possible states. So I decided to compress the states into their useful information. The new states are:

‘next\_waypoint’: next waypoint for the planner

‘Light’: whether the light is green or not

I realized that the planner has already mapped the ideal route for the agent so I compressed all the location information into next\_waypoint. I used the light as a separate variable ignoring traffic because adding in the traffic patterns of the simulation has no positive effect on agent learning.

Behavior changing state variables

Gamma (discount) = .5

Alpha (learn rate)= .5

Epsilon (explore rate)= .1

I now have a learner that learns a feasible policy very quickly. It generally reports a success rate above 90% in repeated trails of n=100.

Discussion of ideal strategy

This simulation creates a rather interesting and unrealistic optimal policy. If the goal is to maximize reward, the agent will not proceed to the destination immediately. Instead it would reach the destination right before the deadline. It would follow the waypoints when possible without reaching the destination and go in circles otherwise. The learning agent on the other hand generally reaches the destination as quickly as possible.

Thoughts for further improvements

I can probably set the discount to zero because future states should not really affect the q score of the state action pairs. I would also like to remove some of the randomness of the exploration. Since there are not very many states it is very feasible to try each one. We could make an exploration function that accomplishes exactly that and once it has finished it would cease exploring. This simulation seems to be a perfect scenario for Epsilon decay, especially in relation to Q.

Sources:

Q-learning implemented using functions from this example: https://studywolf.wordpress.com/2012/11/25/reinforcement-learning-q-learning-and-exploration/