**Brief Overview**

This path planning program uses an algorithm called Improved Q-Learning (IQL). A-Star uses a combination of heuristic value (h-value) along with distance traversed (g-value) to come up with a f-value that helps determine the path for a robot to take. IQL instead assigns each node a Q-value with the goal having the largest Q-value (in this case 100). Using the Q-values of each node as a guide, the robot is able to find a path from start to goal. IQL is an improvement over the Classical Q-Learning (CQL) algorithm. In CQL, a Q-value is given to each action that can be performed at a particular node. Therefore, if there are ‘n’ nodes with ‘m’ actions, then the algorithm needs to store ‘mxn’ Q-values. However, this is not the case in IQL. In IQL, a Q-value is stored only for the ‘best action’ at a particular node. Therefore, if there are ‘n’ nodes, the algorithm will only store ‘n’ Q-values. This is achieved with the help of boolean locks (L). If a Q-value at a node gets updated, then it need not get updated again and the boolean lock is set from 0 to 1.

**Important Points Regarding the Program**

The original algorithm assumed the robot could only go in the four cardinal directions, however, in this program the robot can move diagonally at a 45 degree angle as well. The enivroment takes a while to create the larger the enviornment is. For the 300x300 enviornment, it may take up to a minute for the enivornment to be created. The larger the enviroment, the harder it is to make out the individual nodes, ensure that you view the graph at maximum size so that you may better see the nodes. It is possible for a path not to be found especially if there are more obstacles or if the starting and/or goal index are situated in the corner of the enviornment.

**Hypothesis**

My hypothesis regarding the results is that the algorithm would take longer the larger the enviornment was as well as take longer if the start and goal points were further apart.

**Results**

Given below are the results of various program runs. The time taken includes the time taken to run the algorithm as well as the time taken to plot the path.



















**Interpretation of Results**

From the results, we can see that as the enivornment gets larger, the time taken also gets larger. However, it does not appear that the time taken increases with distance from the goal. Also, it does not appear that the obstacle percentage has any significant impact on the time either. However, an interesting thing that we can notice is that the graphs of each scenario resemble eachother. This means that for a partcular point, the algorithm behaves the same regardless of the environment size and the extra time taken is largely due to the extra time taken to assign Q-values for each node. Thus we can infer that the variation in times is likely due to how the program plots a particluar path and not due to the actual algorithm.

**Screenshots**

Here are some of the indices that were tested along with their respective graphs. The blue node represents the starting point and the green node represents the goal. The red color color is used to represent the nodes and path taken by the robot to go from start to finish.







































































