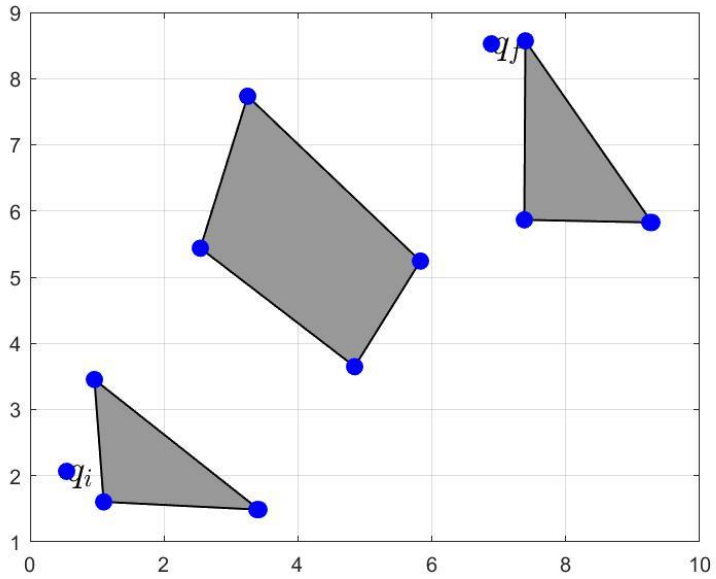
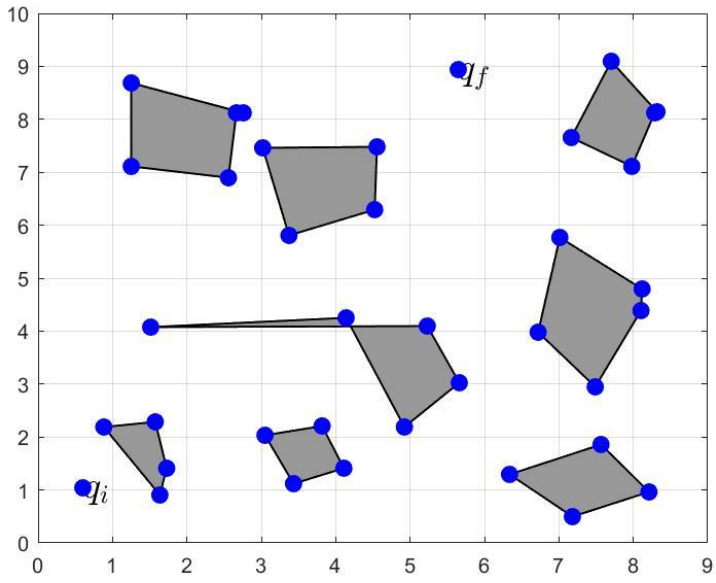
Hafeez Jimoh

MAE 593A Robot Motion Planning HW2

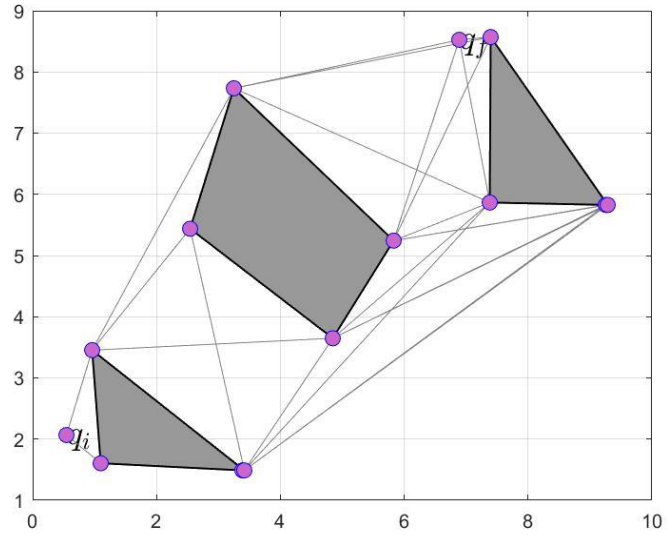
Obstacles with vertices and start position(qi) and goal position(qf) represented as nodes.

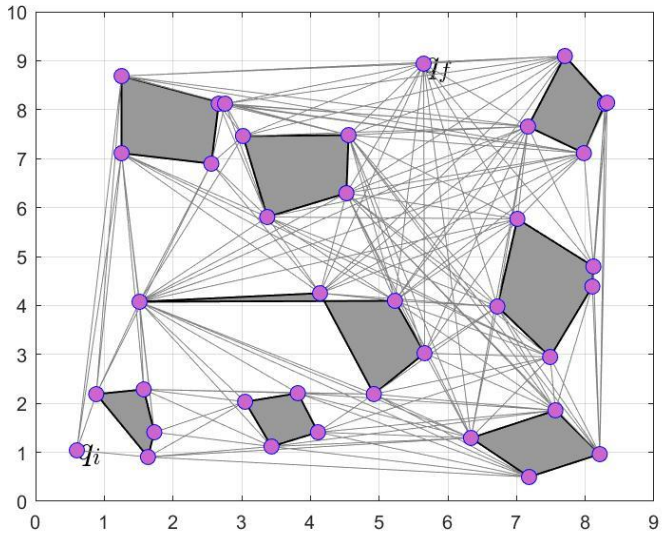




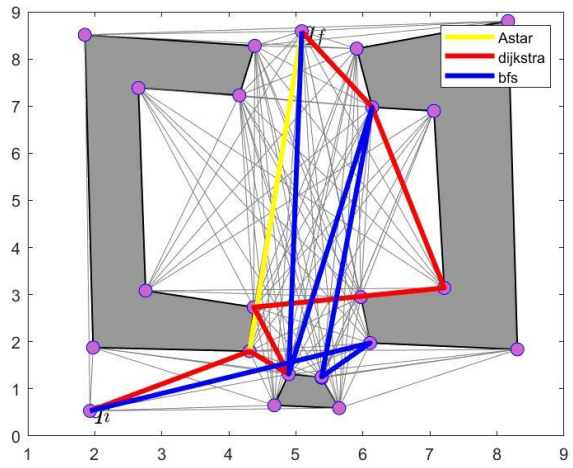
Visibility graph

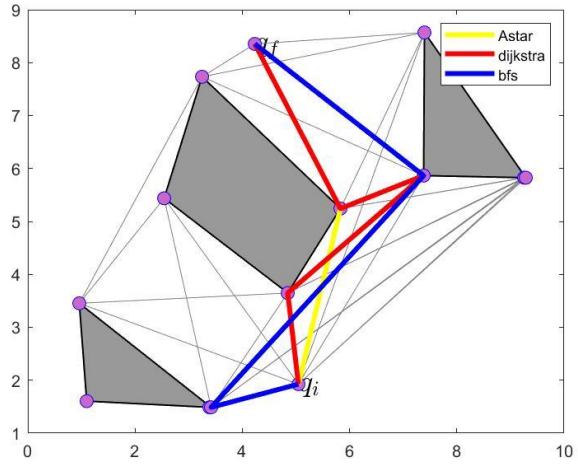
The visibility graph is generated by connecting all nodes together with edges. However only nodes that are in the Free space of the environment make up edges. Any two or nodes that intersect the obstacle when connected is not considered in this case. The visibility graph makes use of a function called discretizeLine() . This functions discretize a line between two points based on a step size and return discrete computed points between the nodes. Each of these points is tested using the isFree function from the previous assignment and if any of the point that belongs to a particular line tests positive, such edge represented as a line is not included in the visibility graph.





Results on different obstacles scenario for BFS, Dijkstra, and Astar algorithm.





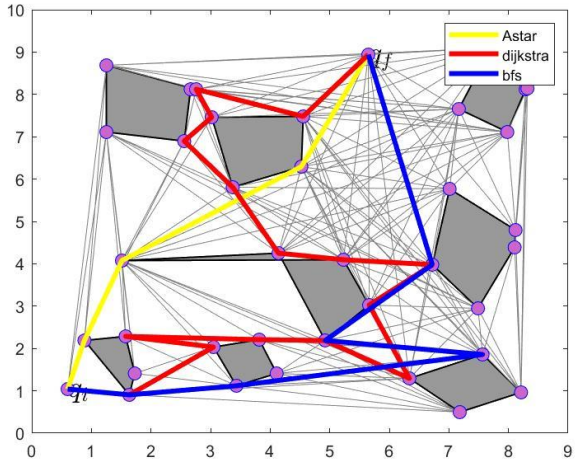


Table 1: Table comparing computation time

	3 Obstacle(12 nodes)	3 Obstacles(22 nodes)	8 Obstacles (35 Nodes)
Visibility Graph	1.403035 seconds.	11.245403 seconds.	19.490460 seconds.
BFS	0.006732 seconds	0.004977 seconds.	0.005615 seconds.
Dijkstra	0.014118 seconds.	0.015774 seconds.	0.017168 seconds.
Astar	0.042512 seconds.	0.046566 seconds.	0.048118 seconds.

It can be seen from the above that BFS consumes less time to compute a path as compared to Dijkstra and Astar. The time it takes to compute a visibility graph is largely impacted by the number of vertices of obstacle and number of obstacles itself. This is due to the discretization step and checking if a path between two nodes is collision free .

Running the code:

There is a file named main.m. Once it is ran, the start and goal location of the path planning task needs to specified and the path would be displayed in a figure. To specify new obstacle coordinates, main_obstacles file need to be run first. The number of obstacles to experiment with needs to be specified first in numbObs variable in the file and then one can specify obstacles coordinates in a matlab figure. A video showing the process the code being run is shown here for 3 obstacles and here for 8 obstacles.