Mapping and Pathfinding

Li, Qingfeng

My project aimed to be able to map an unknown environment and be able to find the shortest path between any two points using a robot. Most robots today have sensors which are reliable enough to detect obstacles but because of real world error, their movement is often not consistent. Due to the inconsistent movement the program used a simulated robot. The program has to be split into two parts in order to code two algorithms. First, DFS (depth first search) has to be implemented. It is chosen over BFS (breadth first search) since the movement of the robot much more inefficient when using BFS. To use DFS, the environment around the robot has to be turned into a tile system and then it can be converted to a tree of nodes. The map of nodes is then sent to the A* function. A* is the algorithm to find the shortest path between two points. Basically, it explores the most “promising node” at each turn until the end node is found. After that, the robot backtracks through the nodes until it has recreated the path. The reason I didn’t use A* directly instead of using DFS first was because of its inefficiency. That is due to the fact that the promising node is not always adjacent to the previously explored node. Another issue with using A* directly is that the robot will not explore the entire environment, and therefore not everything will be mapped.