Name: Aditya Nisal

Graph Search Algorithm Implementation

Dijkstra Algorithm:

Dijkstra's algorithm calculates the cost of reaching each node in the graph starting from the start node. The algorithm assigns the cost of reaching each node based on the cost of reaching the previous node. The priority queue is sorted based on the total cost of reaching the node, which is the sum of the cost of reaching the previous node and the cost of reaching the current node. The algorithm works by visiting the node with the lowest cost first, and adding its unvisited neighbors to the priority queue, until the goal node is reached.

Pseudocode:

1. Create a priority queue Q and add the start node to it
2. Assign the cost of reaching the start node to 0 and the cost of reaching all other nodes to infinity
3. Repeat the following steps until the priority queue Q is empty or the goal node is reached:
   a. Pop the node with the lowest cost from the priority queue Q
   b. If the node is the goal node, return the cost of reaching it
   c. For each neighbor of the current node:
      i. Calculate the cost of reaching the neighbor as the sum of the cost of reaching the current node and the cost of reaching the neighbor from the current node
      ii. If the cost of reaching the neighbor is less than the current cost of reaching the neighbor, update the cost and add the neighbor to the priority queue Q
4. If the goal node is not reached, return failure

Breadth First Search (BFS) Algorithm:

The Breadth-First Search (BFS) algorithm is a graph traversal technique that uses a queue, a FIFO (First-In-First-Out) data structure, to explore the graph. It starts by visiting the start node and exploring its neighbors, which are then added to the end of the queue. The algorithm then moves to the next node at the front of the queue, visiting its neighbors and adding them to the end of the queue. This process continues until the goal node is found. BFS is named so because it explores the breadth of the graph before going deeper, unlike DFS which visits the children of a node before exploring its siblings. BFS always results in the shortest path from the start node to the goal node, but it does not consider the heuristics or the cost to reach the goal node from the start node.

The pseudocode for the Breadth-First Search (BFS) algorithm is as follows:

1. Create a queue and add the start node to it.
2. Create a set to keep track of visited nodes.
3. While the queue is not empty: a. Dequeue the first node from the queue and call it the current node. b. If the current node is the goal node, return the path. c. For each unvisited neighbor of the current node: i. Mark the neighbor as visited. ii. Add the neighbor to the queue.
4. If the queue is empty and the goal node has not been found, return failure.

Depth First Search (DFS) Algorithm:

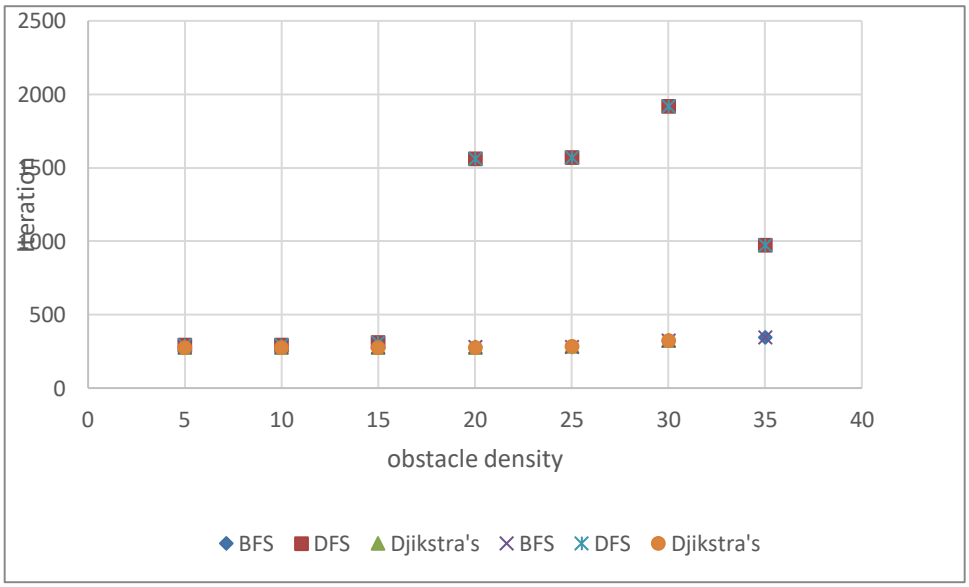
Name: Aditya Nisal

DFS (Depth-First Search) is a graph traversal algorithm used in motion planning. It works by exploring all the branches of a graph as far as possible before backtracking. The pseudocode for DFS is as follows:

1. Start at the root node.
2. Mark the current node as visited.
3. Repeat the following steps for each unvisited neighbor of the current node: a. Set the current node to the neighbor. b. Mark the current node as visited. c. Repeat the steps 1 to 3 for the current node.
4. If all neighbors of the current node are visited, backtrack to the parent node.
5. Repeat the steps 2 to 4 until the goal node is found or all nodes have been visited.

Performance Plot:

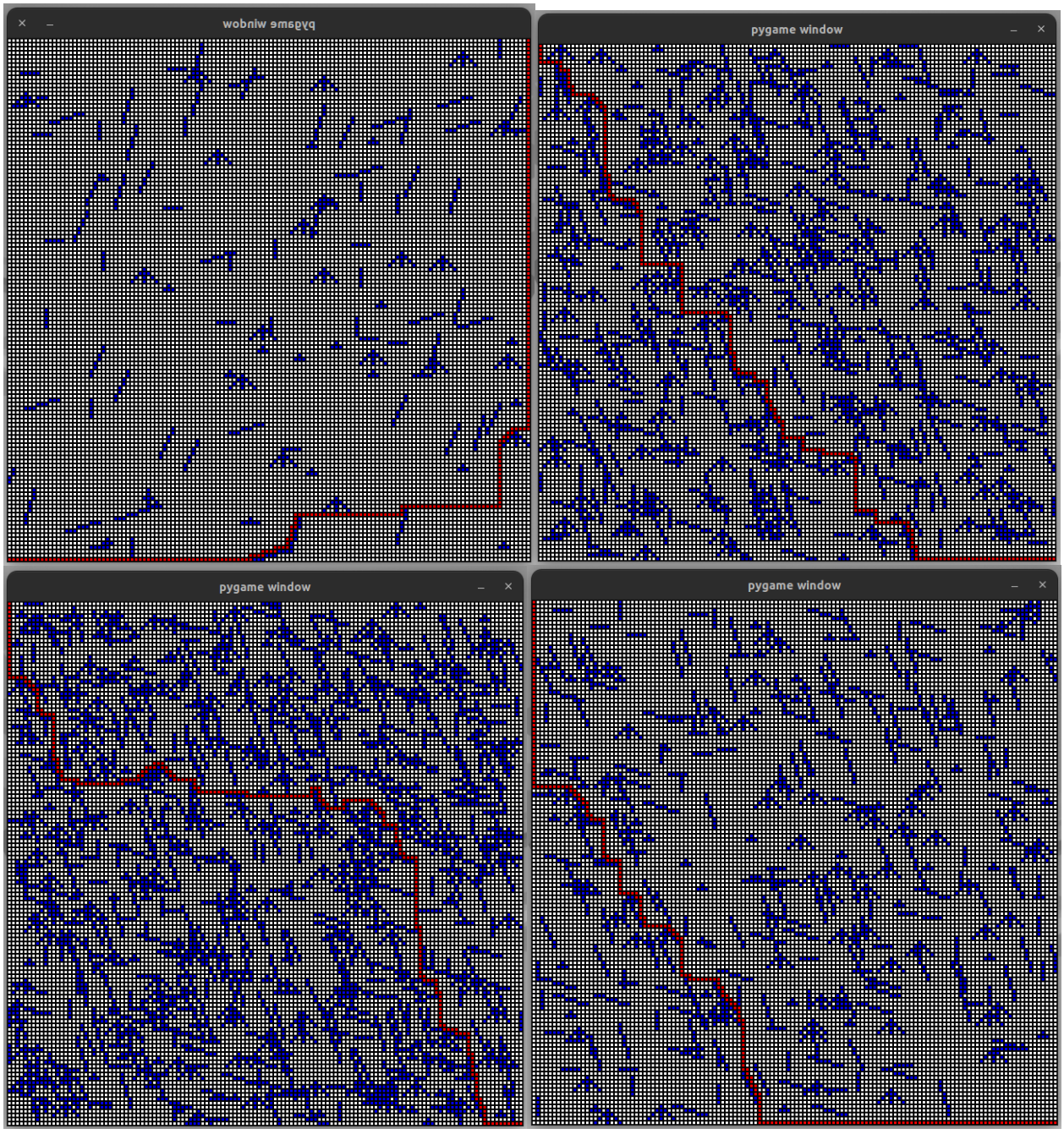
The graph shows the comparison of the number of iterations required by different algorithms to traverse a grid world. It is observed that DFS has the longest traversal time, while BFS and Dijkstra's Algorithm have a similar traversal time. As the density of obstacles increases, the number of steps needed to reach the goal increases significantly for DFS but only moderately for BFS and Dijkstra. The algorithms could not complete the traversal when the obstacle density reached 40% for BFS and DFS and 35% for Dijkstra.



Name: Aditya Nisal

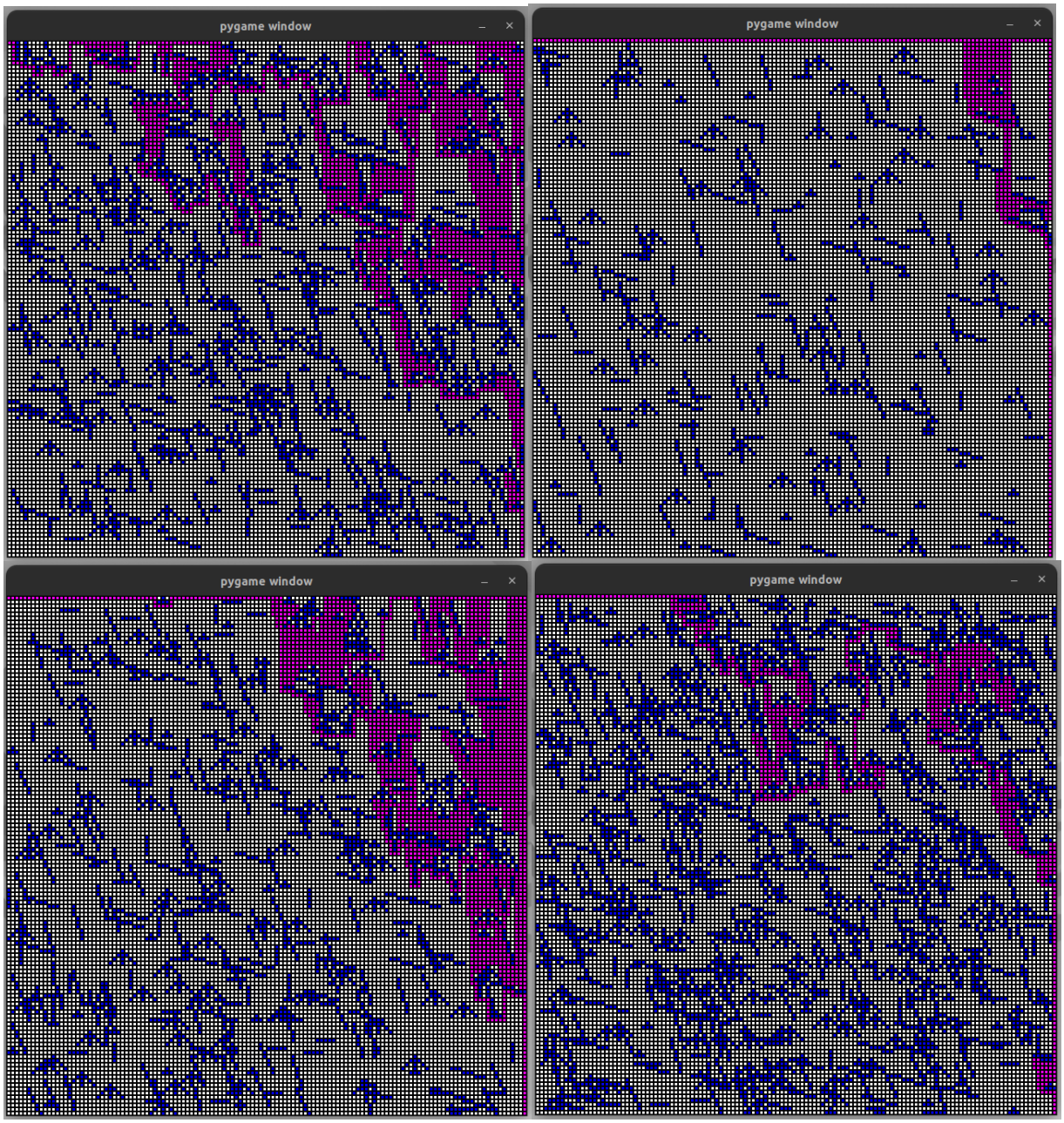
Images for different Planners:

BFS:



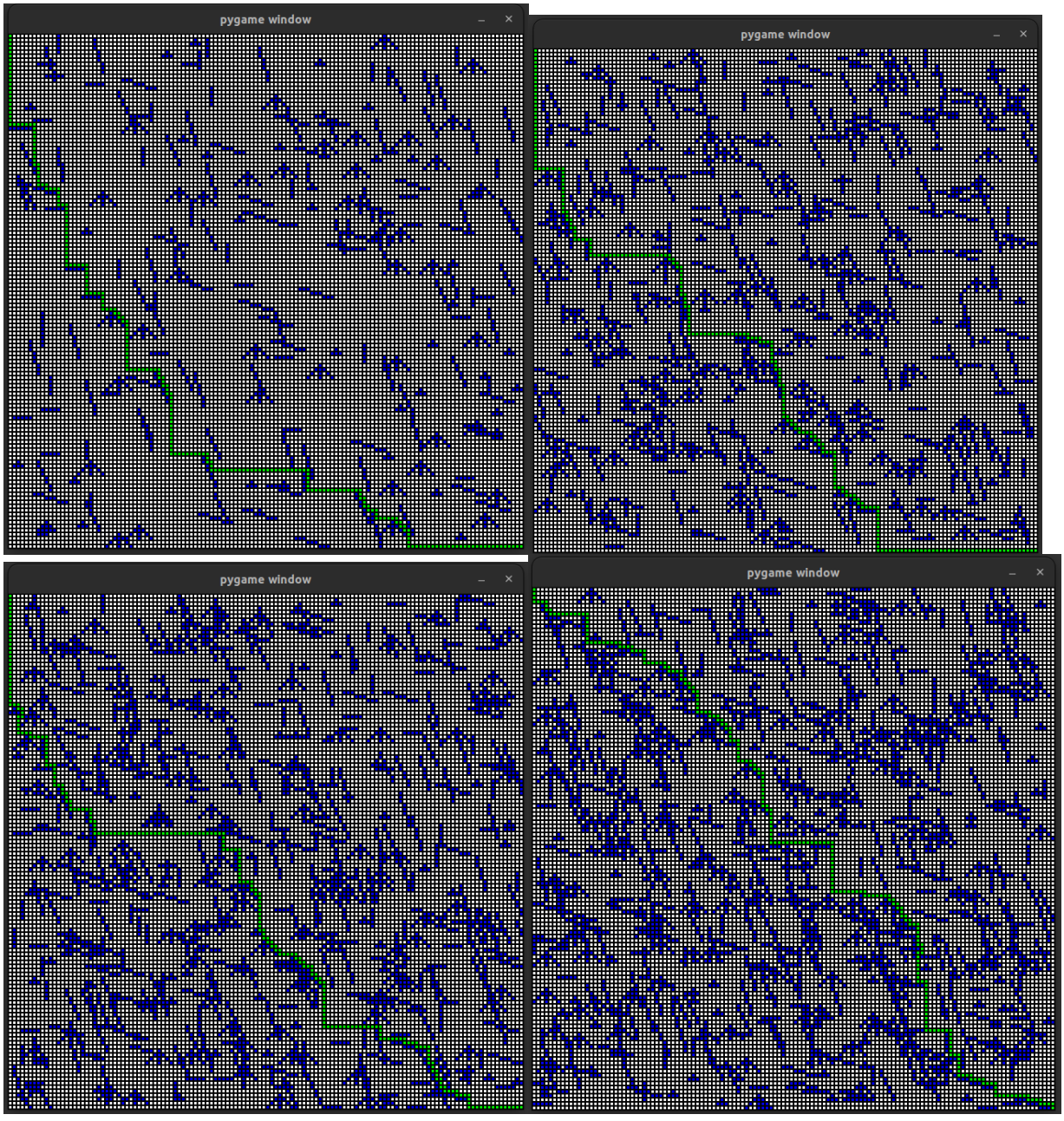
Name: Aditya Nisal

DFS:



Name: Aditya Nisal

Dijkstra’s Algorithm:



Name: Aditya Nisal

Random Planner: