1. Implementation Overview

This project involved implementing two classical path planning algorithms Dijkstra's Algorithm and A* Search on a 2D occupancy grid. Each grid was read from a CSV file, where 0 indicates a free cell and 1 represents an obstacle. The A* algorithm was implemented using the Manhattan heuristic to maintain admissibility, while Dijkstra operated without any heuristic, expanding nodes based solely on cost.

2. Benchmark Results

Map	Algorithm	Avg Time (ms)	Avg Path Length	Success Rate (%)
map1_sparse	Dijkstra	11.25	42.80	100
map1_sparse	A*	0.25	38.60	100
map2_maze	Dijkstra	3.60	211.20	100
map2_maze	A*	3.58	317.60	100
map3_dense	Dijkstra	0.11	52.67	60
map3_dense	A*	0.20	58.00	20

3. Analysis

In this benchmark, A* consistently achieved faster computation times than Dijkstra due to its use of the Manhattan heuristic, especially evident in the `map1_sparse` results. It completed searches significantly faster while maintaining a 100% success rate. However, Dijkstra occasionally produced shorter paths, as seen in `map1_sparse` and more prominently in `map2_maze`.

In `map2_maze`, Dijkstra’s uniform exploration led to more optimal paths compared to A*, which followed a heuristic-driven route that resulted in longer paths. Despite similar runtimes, this shows how heuristics may reduce efficiency in maze-like environments by diverting the search from shorter but less obvious paths.

The `map3_dense` results highlight the impact of obstacle density. Dijkstra succeeded 60% of the time, whereas A* succeeded only 20%. This suggests that A* struggles more in dense environments where its heuristic may falsely guide it into blocked regions, while Dijkstra's exhaustive approach improves the chance of finding a narrow valid path.

Overall, A* is better suited for open or semi-structured maps where speed is crucial and the heuristic aligns with the goal. Dijkstra, while slower, offers more reliability in environments where heuristic shortcuts are risky. The results illustrate the balance between speed and path quality, and how algorithm choice should depend on map complexity.