

CS440 Assignment 1

Fast Trajectory Replanning

Felipe Da Silva
Omar Waseem

February 28, 2025

1 Introduction

This report analyzes the performance of three pathfinding algorithms used in a **grid-based search problem** where an agent must navigate from a start position to a goal position while encountering unknown obstacles. The implemented algorithms are:

- Repeated Forward A*
- Repeated Backward A*
- Adaptive A*

Each algorithm was tested on a **100x100 gridworld**, and their execution times were measured to compare efficiency.

2 Experimental Setup

The gridworld environments were generated using a **Depth-First Search (DFS)** based maze generation algorithm with randomized obstacle placement. The agent was allowed to move **only in four cardinal directions** (north, south, east, and west). The algorithms assumed all unknown cells to be unblocked until discovered otherwise, updating their knowledge dynamically as they traversed the grid.

The heuristic function used in all A* searches was the **Manhattan Distance**, defined as:

$$h(s) = |x_s - x_g| + |y_s - y_g| \quad (1)$$

where (x_s, y_s) and (x_g, y_g) represent the coordinates of the current state and the goal state, respectively.

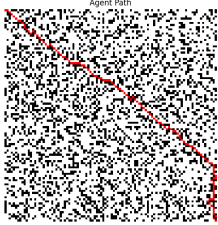
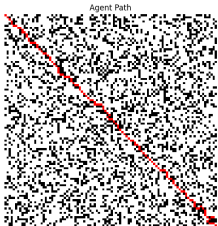
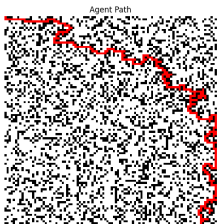
Algorithm	Runtime (seconds)	Path Visualization
Repeated Forward A*	0.011489	
Repeated Backward A*	0.011477	
Adaptive A*	0.498182	

Table 1: Execution times for each algorithm on a 100x100 grid.

3 Results and Observations

The table below summarizes the runtime of each algorithm:

3.1 Comparison of Forward and Backward A*

The results show that **Repeated Forward A*** and **Repeated Backward A*** performed almost identically in terms of execution time. This is expected because both algorithms utilize the same heuristic function and explore the grid in a similar fashion, with only a difference in their search direction.

3.2 Adaptive A* Performance

Adaptive A* had a significantly **higher runtime** compared to the other two algorithms. This is because Adaptive A* updates heuristic values dynamically,

leading to additional computations. While this can improve performance in future searches, it introduces an overhead that is evident in this test case.

4 Conclusion

From the experimental results, we conclude the following:

- Repeated Forward A* and Repeated Backward A* have **almost identical efficiency**, making them interchangeable depending on problem constraints.
- Adaptive A* has a **higher computational cost**, making it less efficient in this setting despite its theoretical advantages.

Future work could explore modifications to Adaptive A* to optimize its heuristic updates, reducing unnecessary computations while retaining its benefits.