Part 3 - Forward vs. Backward A Analysis*

Implementation

We implemented both Repeated Forward A* and Repeated Backward A* algorithms for the Fast Trajectory Replanning problem. Both implementations break ties among cells with the same f-value in favor of cells with larger g-values.

Experimental Results

Our experiments on 20 randomly generated mazes of size 101×101 with approximately 30% blocked cells showed significant performance differences between the two algorithms:

1. Expanded Cells:

- Forward A*: Average of 2,121.40 cells expanded
- Backward A*: Average of 11,544.95 cells expanded
- Ratio (Forward/Backward): 0.18x

Forward A* expanded only 18% as many cells as Backward A*, making it substantially more efficient in terms of search space exploration.

2. Path Length:

- Forward A*: Average path length of 164.60
- Backward A*: Average path length of 155.30

While Backward A* found slightly shorter paths on average, Forward A* was more computationally efficient.

3. Runtime:

- Forward A*: Average runtime of 0.0259 seconds
- Backward A*: Average runtime of 0.1366 seconds
- Ratio: Forward A was approximately 5.3x faster*

Runtime correlates with the number of expanded cells, confirming Forward A*'s efficiency advantage.

Detailed Analysis

Looking at the individual maze results, we observe:

1. Consistency of Performance Advantage:

- Forward A* consistently expanded fewer cells than Backward A* across all 20 mazes.
- Even in the worst case for Forward A*, it still expanded 15% fewer cells than Backward A*.

2. Correlation with Maze Complexity:

- The advantage of Forward A* appears more pronounced in complex mazes.
- In mazes where both algorithms expanded many cells (e.g., Maze 15: 5,304 vs. 58,208 cells), the ratio was smaller (0.09x), showing a greater advantage for Forward A*.
- In simpler mazes (e.g., Maze 9: 257 vs. 320 cells), the ratio was closer to 1 (0.80x).

Explanation of Observations

The dramatic difference in performance can be explained by several factors:

1. Fog of War Effect:

- Forward A* starts from the agent's position, where it has more information about the environment.
- Backward A* starts from the target, where it has no initial knowledge of obstacles.
- Backward A* wastes more time exploring invalid paths due to missing information.

2. Information Gain During Search:

- Forward A* builds paths from known (agent's surroundings) to unknown.
    - More informed early decisions
    - Focused exploration in promising directions
    - Less need for recomputation when obstacles are discovered
- Backward A* works from unknown (target) to known (agent), leading to inefficiencies.

3. Heuristic Effectiveness:

- The Manhattan distance heuristic may be more accurate when estimating distance from the agent than from the target.

4. Path Quality Trade-off:

- Backward A* sometimes finds slightly shorter paths, but at a huge computational cost.

- The minor 5.6% improvement in path length does not justify the 5.3x slower performance.

Conclusion

For repeated A search in a fog of war scenario*, Forward A* significantly outperforms Backward A* in terms of computational efficiency:

- 5.3x faster
- 82% fewer cells expanded
- Only 5.6% longer paths (on average)

Thus, in applications where agents navigate partially observable environments, searching from the agent to the goal is preferable, especially when:

- Computational resources are limited
- Real-time performance is important

This result: when navigating an unfamiliar environment, it is more efficient to plan a path starting from where you have the most information (your current surroundings) rather than working backward from a destination with limited knowledge