Using Pathfinding Algorithms in Unity to Enhance Game Optimization for An Agent

Intro to Artificial Intelligence
Term Project - Fall 2024

Justin Pelletier
Ryan Larrea

Project Goal

Project Goal

The object of the project is to showcase the two search algorithms, A* and Dijkstra's. We compare the two in terms of speed and memory usage. To showcase the algorithms we utilize a procedurally generated maze for both algorithms to search through.

Context

Pathfinding algorithms are important especially for video games.

Research Focus

A* and Dijkstra's are two search algorithms that can return the shortest path in an environment.

Background and Research Review

Pathfinding Algorithms

- **Dijkstra**: Guarantees shortest path, explores all nodes.
- **A***: Uses heuristics, faster and more focused.

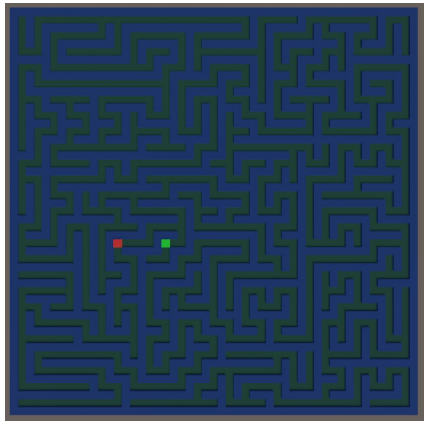
Key Related Research

- **Study 1:** Road networks, bidirectional Dijkstra rises above.
- **Study 2:** Comprehensive algorithm comparison.

Our Research

- Dynamic, real-time environment in Unity
- Focus on speed and memory.

Implementation



- Maze Creation

- A* and Dijkstra’s algorithms

Metrics For Evaluation

Speed

Memory

Node Exploration Visualization

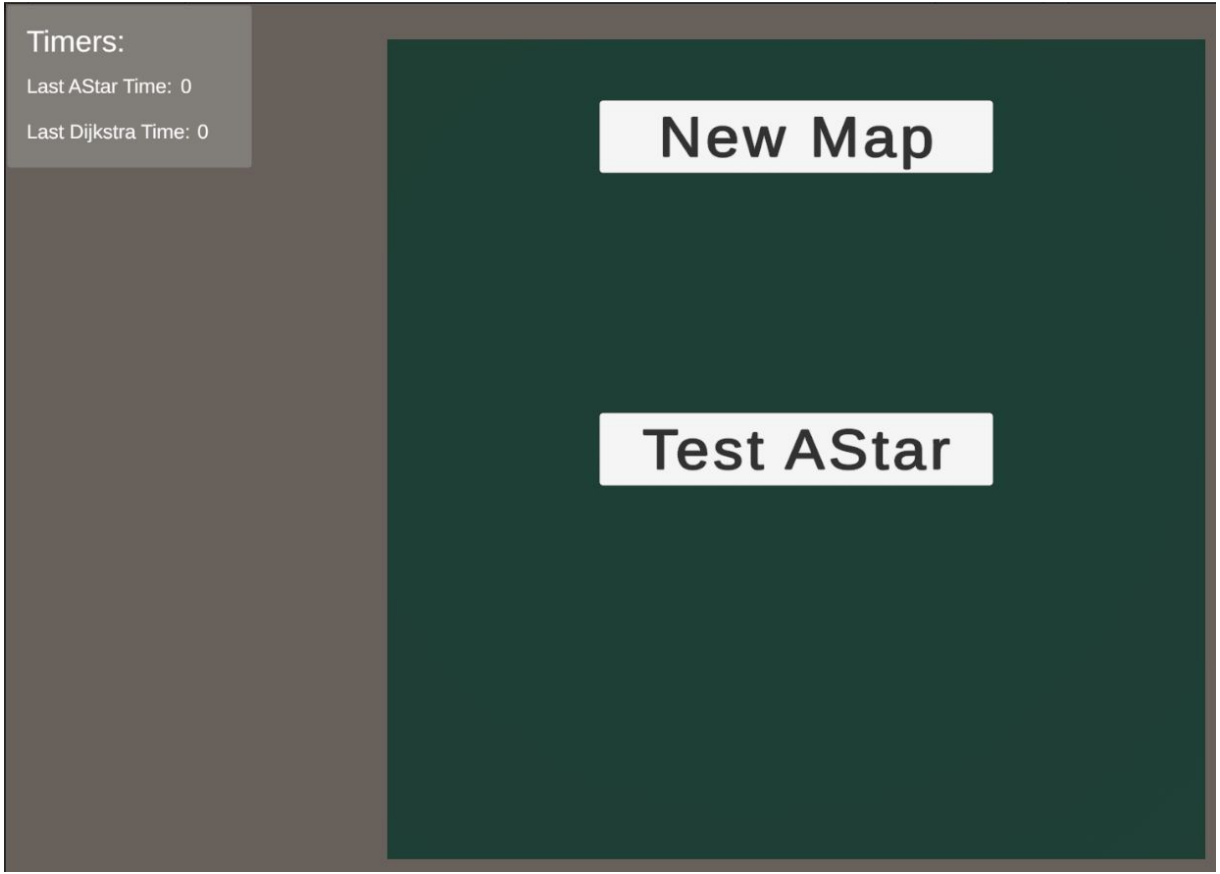
Interactivity

Timers

Regenerate Map

Test A*/Dijkstra

Visual Of The Algorithm



Demonstration

Results and Discussion

Key Findings

- **A*:** Faster, lower memory usage.
- **Dijkstra**: Thorough but slower.

Insight from Unity

- Randomly generated mazes highlight differences.
- Performance tracked through metrics using Unity's built in analytic tools.

Real-World Relevance

- A* excels in dynamic environment due to its heuristic approach.
- Dijkstra is best for non-heuristic setups.

Future Work

- Explore additional algorithms:
  - Bidirectional Dijkstra, IDA*, UCS, etc.
- Implement dynamic heuristics for A*.
- Implement a varying, more complex environments.
- Expand the scope and/or domain:
  - Robotics, traffic systems, other AI agents, etc.

Conclusion

We hypothesized that an A pathfinding algorithm performs more efficiently in terms of speed and memory usage compared to Dijkstra's algorithm.*

Using Unity we were able to support our initial hypothesis that the A*'s algorithm is more efficient.