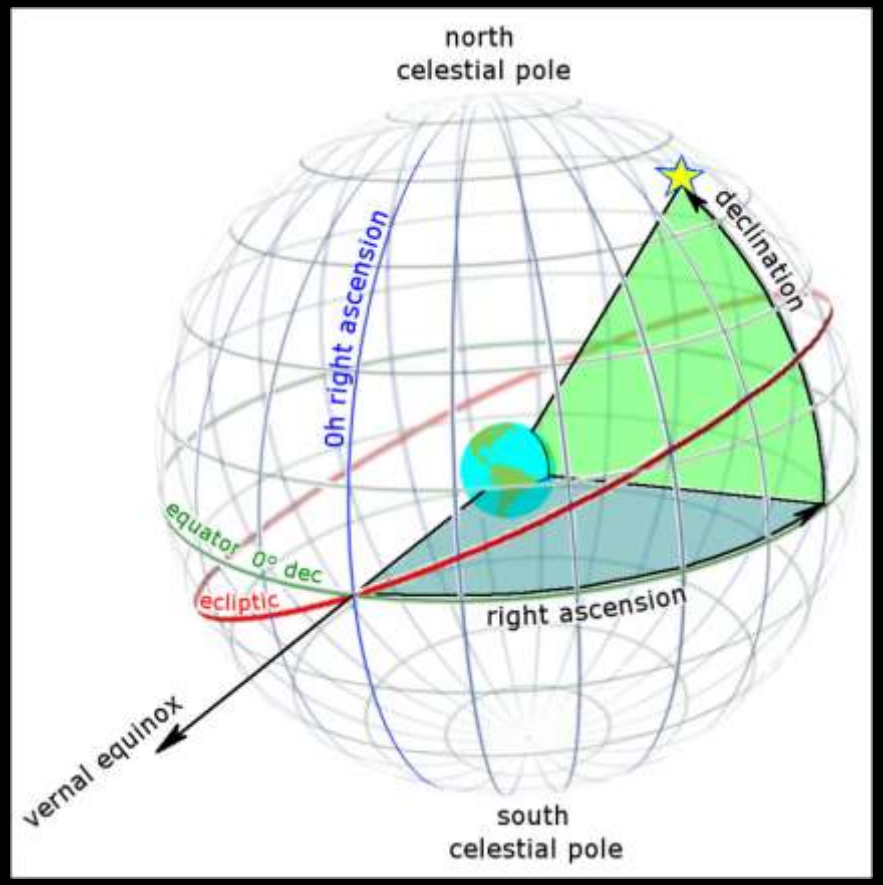

Navigating
the Cosmos

Introduction

Space Travel~~~

Pin a star in the sky



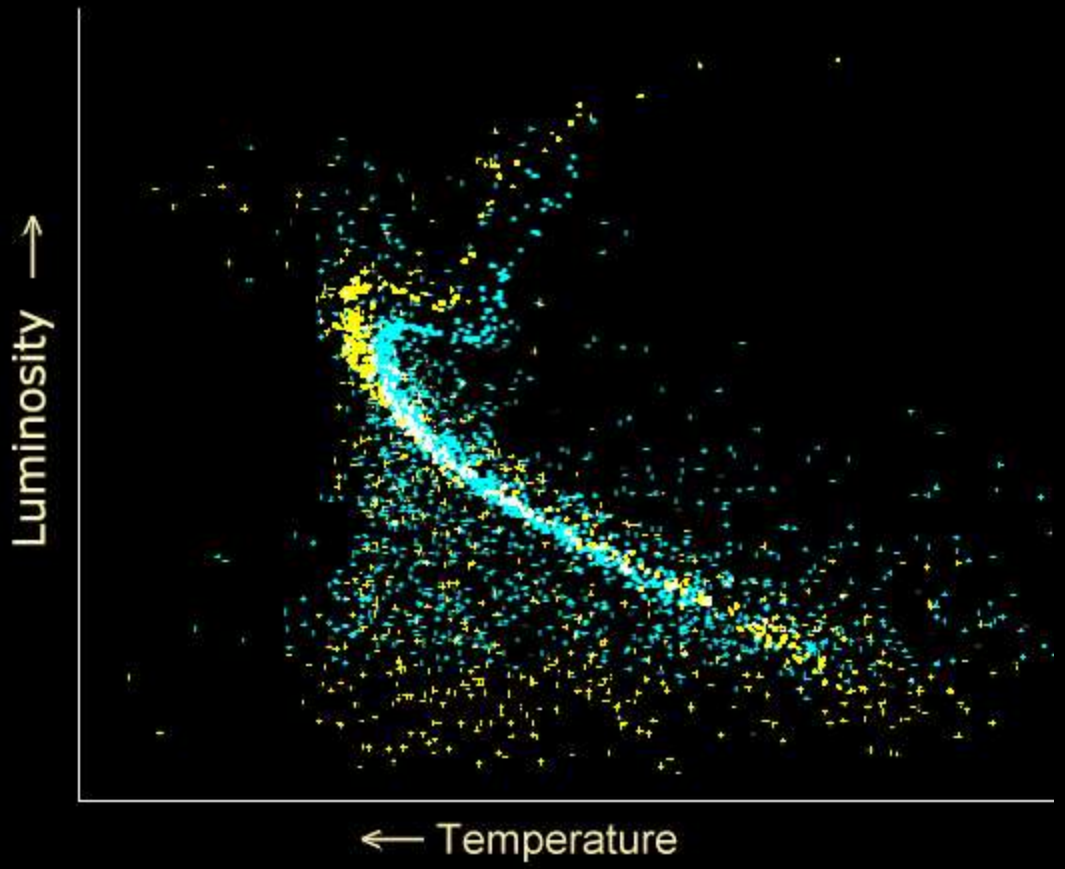
Three numbers to pin a point:

1. Right Ascension
2. Declination
3. Distance

And we will convert it into cartesian coordinate system.

We choose Earth to be our origin.

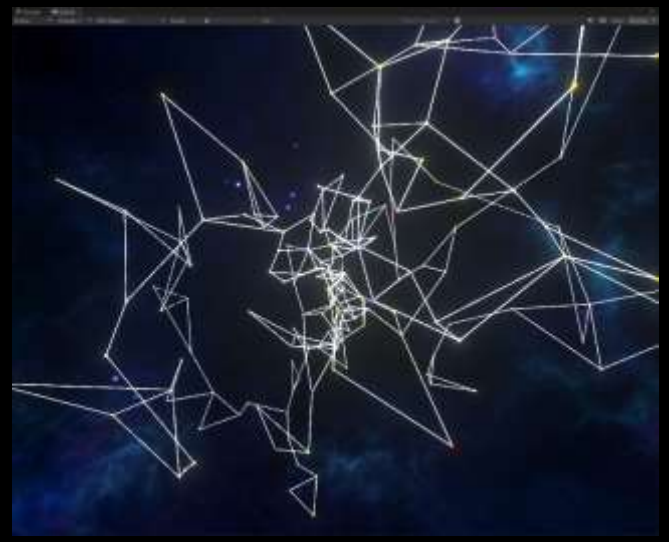
Star Luminosity and Color

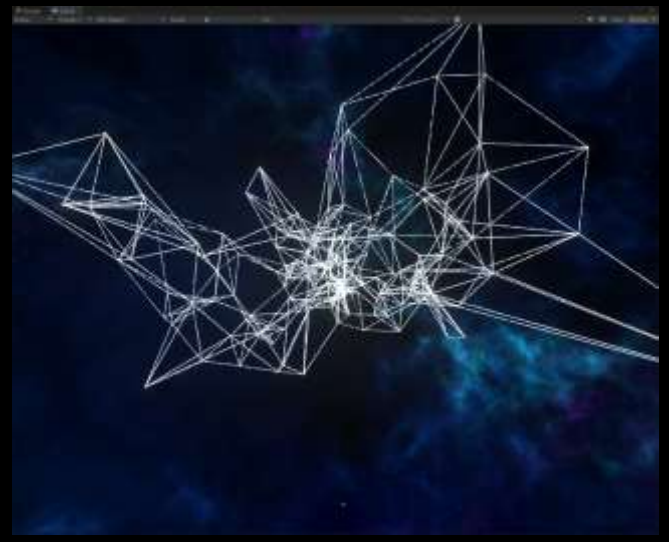


Luminosity is calculated based on:
1. Magnitude in G-band
2. Parallax (related to distance)

Color is calculated based on the Temperature, using the formula:
Blue photometer - Red photometer

Star Graph Builder





Algorithm (KNN algorithm):

1.Node Identification: Each data point is treated as a node in the graph.

2.Distance Calculation: Calculate the distance between each node and every other node in the dataset.

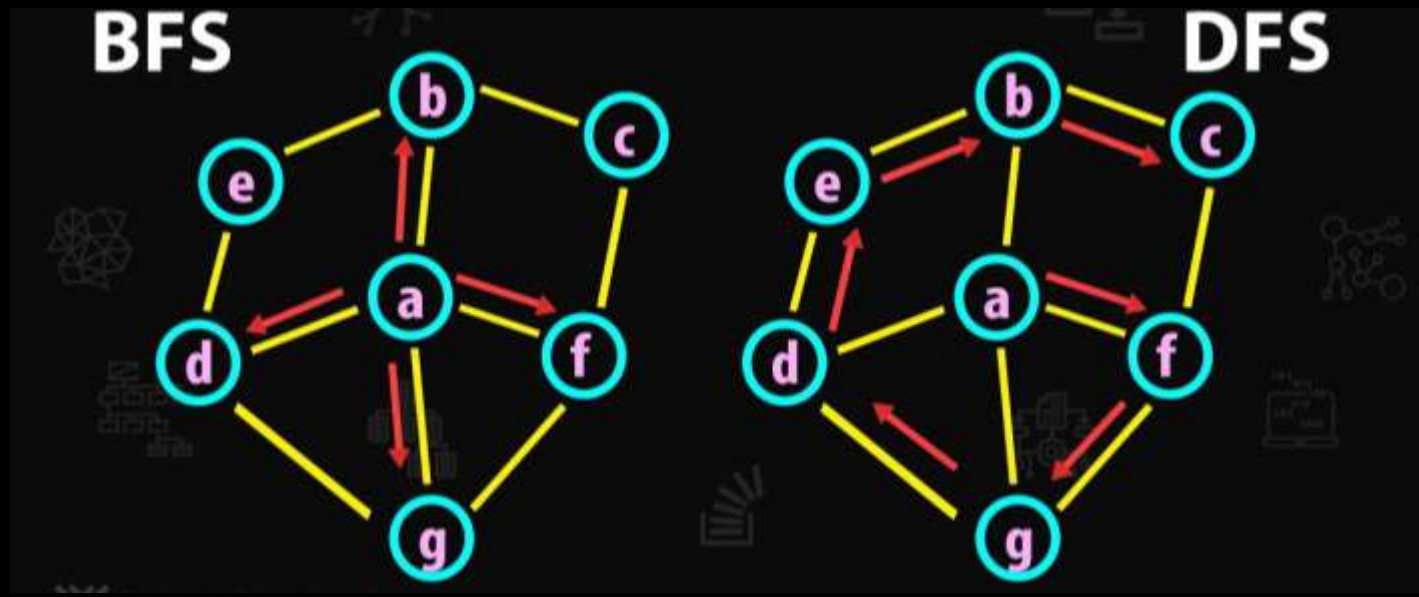
3.Neighbor Selection: For each node, identify the closest K neighbors.

4. Edge Creation: Create edges between each node and its K neighbors.

Modification:

1. Path connectivity: Once the graph found that this node is connected to the whole graph, it will not try to connect this one.
2. Data storage: Store edges, vertices, and neighbor list to make the calculation of other algorithms more convenient.
3. Visualization of the graph! Using the Unity Engine Material System to do the work.

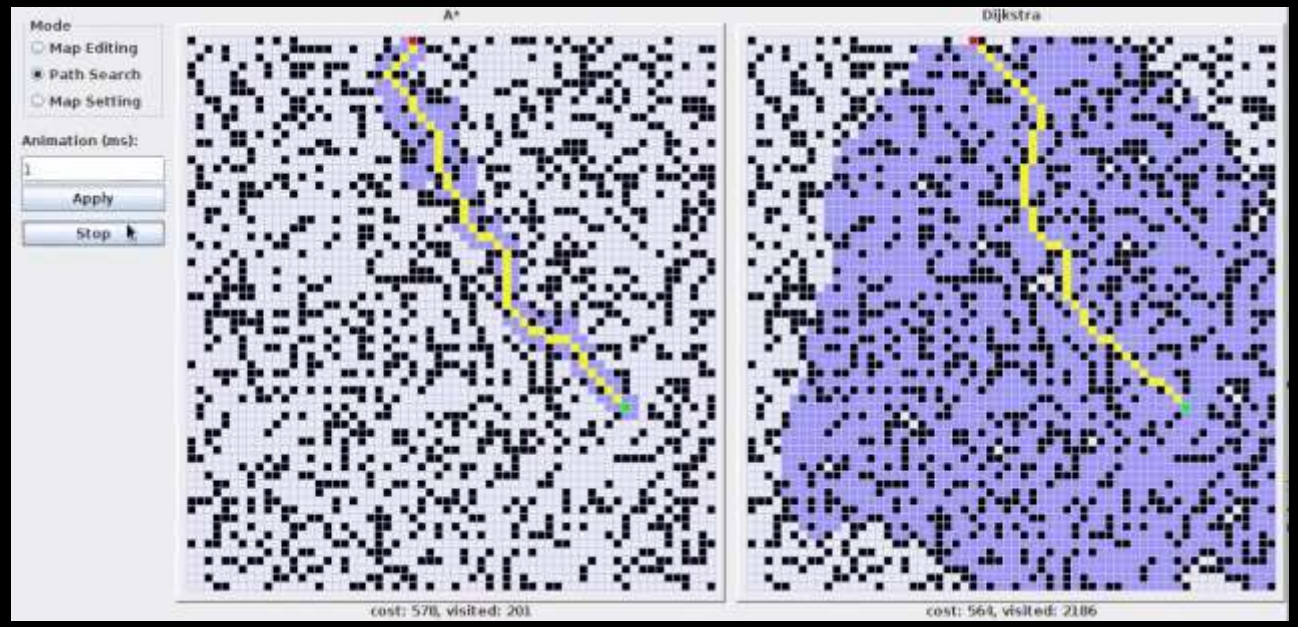
Star Graph Navigation



DFS is a graph traversal method that explores as far as possible along each branch before backtracking. It uses a stack (either explicitly or via recursion) to keep track of the next vertex to visit.

BFS explores all neighbors at the current depth before moving on to nodes at the next depth level. It uses a queue to manage the set of next vertices to explore.

Star Graph Navigation



A* combines the best features of Dijkstra's algorithm and a heuristic-based search to efficiently find the shortest path to a goal by estimating the costs from the current node to the end using a heuristic.

Dijkstra's algorithm methodically calculates the shortest path from a starting node to all other nodes in a graph using a priority queue to explore the least costly paths first.

Star Graph Navigation

Algorithm	Time Complexity	Note
Depth-First Search (DFS)	$O(V + E)$	V is vertices, E is edges; worst-case when all vertices and edges are explored.
Breadth-First Search (BFS)	$O(V + E)$	Similar to DFS, suitable for unweighted graphs.
A* Search	$O((V + E) \cdot \log V)$	Assumes a good heuristic; $V \log V$ arises from maintaining the priority queue.
Dijkstra's Algorithm	$O((V + E) \cdot \log V)$	Uses a priority queue (e.g., binary heap).

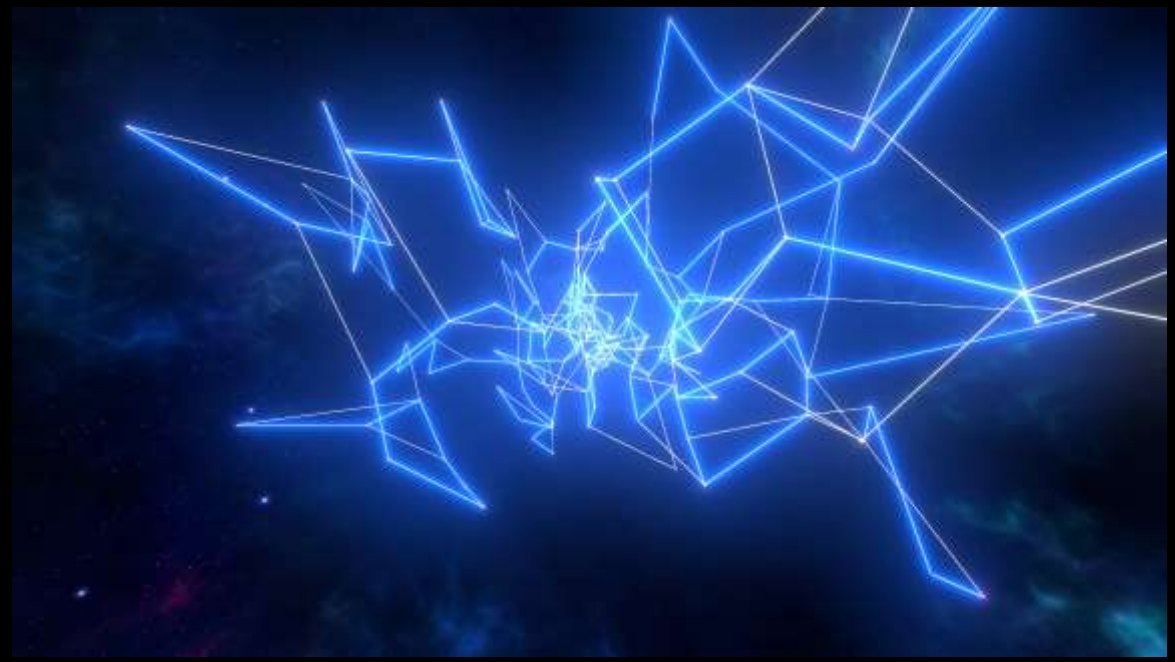
These are the algorithms we have implemented:

1. Depth-First Search
2. Breadth-First Search
3. A* Search
4. Dijkstra's Algorithm

Modification:

1. Unified Graph Struture
2. Reconstruct Path
3. Visualization and Debugging

Star Traversal



Finding the shortest path traversal all stars is an NP-Hard problem. So we use greedy algorithm to find an almost best solution.

This is the algorithm we made our selves. Here are the steps:

1.Initialization:

1. A random vertex is selected as the starting point.
2. This vertex is marked as visited, and it's added to the traversa path.

2.Finding the Nearest Unvisited Neighbor:

1. From the current vertex, the algorithm looks for the closest unvisited neighbor.
2. This neighbor is then marked as visited, added to the path, and becomes the new current vertex.

3.Continuation and Handling Dead-ends:

1. This process of moving to the nearest unvisited neighbor continues.
2. If a dead-end is reached (no unvisited neighbors), the algorithm uses the A* search to find a path to the nearest unvisited vertex. This path is followed until a new unvisited vertex is reached. (This could be optimized if we have enough time)

4.Termination:

1. The traversal ends when all vertices have been visited.

Future Work

Video Link: https://youtu.be/iuxMKl6uWbk

What we can do in the future

1. Add more traversal and navigation algorithms
2. Represent more realistic visual performance. Visualize how the algorithm finds the best path.
3. Trying to find a path when all stars in the graph are moving (Like in the dune)