DELHI TECHNOLOGICAL UNIVERSITY

Pathfinding Visualizer

DATA STRUCTURES AND ALGORITHM

(IT-353)

PROJECT REPORT

SUBMITTED BY :

HARSH PANCHAL 2K18/EP/031

ASHWAJIT SINGH 2K18/EP/018

Under the supervision of

Ms. GEETANJALI BHOLA

ACKNOWLEDGEMENT

We have taken efforts in this project. However, it would not have been possible without the support and help of many individuals and organizations. We would like to extend my sincere thanks to all of them.

We are highly indebted to DTU and especially to Ms. GEETANJALI BHOLA our Faculty Supervisor, for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

Last but not least, we would like to thank our respective families for their endless support and help throughout the duration of the work.

Thank you

ABSTRACT

Pathfinding is the search for an optimal path from a start location to a goal location in a given environment. In Artificial Intelligence pathfinding algorithms are typically designed as a kind of graph search. These algorithms are applicable in a wide variety of applications such as computer games, robotics, networks, and navigation systems. The performance of these algorithms is affected by several factors such as the problem size, path length, the number and distribution of obstacles, data structures and heuristics.

Pathfinding is closely related to shortest path problem, thus the definition of pathfinding is finding the optimal path from a given start node(s) to goal node(g) in the given graph(G), where optimal refers to the shortest path, low-cost path, fastest path or any other given criteria. Pathfinding can generally be divided into two categories: SAPF, that is Single Agent Pathfinding, to generate a path for one agent and MAPF that is Multi-Agent Pathfinding, to generate the path for more than one agent.

CONTENTS

INTRODUCTION

Pathfinding or pathing is the plotting, by a computer application, of the shortest route between two points. Pathfinding is the search for an optimal path from a start location to a go allocation in a given environment. These algorithms are applicable in a wide variety of applications such as computer games, robotics, networks, and navigation systems. The performance of these algorithms is affected by several factors such as the problem size, path length, the number and distribution of obstacles, data structures and heuristics.

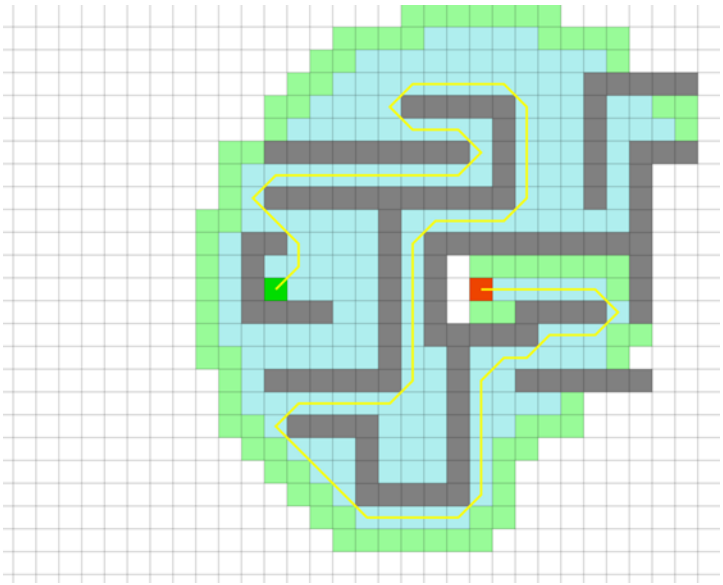
In this project we are using two different algorithm Dijkstra's , A* Algorithm.

One of the most popular path finding algorithm is known as the Dijkstra's Shortest Path First algorithm. It is a solution to the single-source shortest path problem in graph theory.

A Star algorithm is a best first graph search algorithm that finds a least cost path from a given initial node to one goal node. Uses heuristic to guide search which means it is too optimistic, estimating the cost to be smaller than it actually is.

Pathfinding- Overview

Pathfinding refers to computing an optimal route in a given map between the specified start and goal nodes. It is an important topic in the area of Artificial Intelligence with applications in fields such as GPS, Real-Time Strategy Games, Robotics, logistics while implemented in static or dynamic or real-world scenarios . Recent developments in pathfinding lead to more improved, accurate and faster methods and still captivates the researcher's attention for further improvement and developing new methods as more complex problems arise or being developed in AI . A great deal of research work is done in pathfinding for generating new algorithms that are fast and provide optimal path since the publication of the Dijkstra algorithm in 1959. Most of the research work is validated using experimental data. Therefore, the research must provide reliable and accurate information as experiments are very volatile.



Pathfinding Problem Definition:

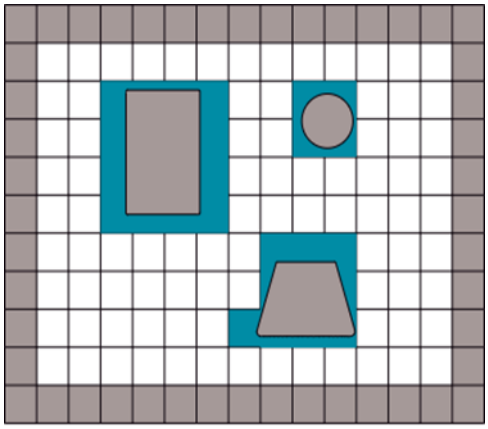
Pathfinding is closely related to shortest path problem, thus the definition of pathfinding is finding the optimal path from a given start node(s) to goal node(g) in the given graph(G), where optimal refers to the shortest path, low-cost path, fastest path or any other given criteria. Pathfinding can generally be divided into two categories: SAPF, that is Single Agent Pathfinding, to generate a path for one agent and MAPF, that is Multi-Agent Pathfinding, to generate the path for more than one agent. In this paper, we only consider the single-agent pathfinding problem in a static environment, which means the map does not change as the agent moves. Pathfinding has applications in different fields and it is hard to consider all the application areas, so in this paper, only video game applications are used and in 2D environments.

Map Representations:

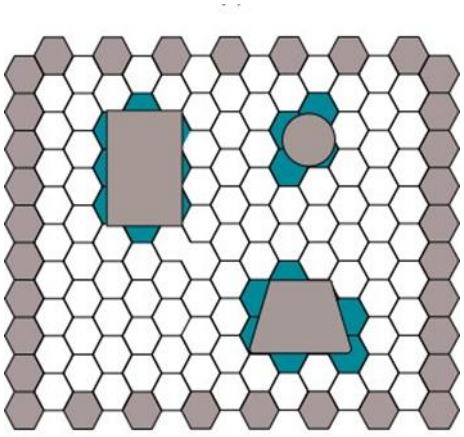
Pathfinding is used in a wide variety of areas and usually implemented on different maps that are generated to test pathfinding algorithms. The widely popular maps are implemented using a grid-based graph, set of nodes and edges, representations in the algorithm. Usually, a grid is superimposed over a map and then the graph is used to find the optimal path . Most widely used representations are square tile grid which can either be accessed as a 4-way path or 8- way path. Both have their own advantages and disadvantages. Grids are considered simple and easy to implement and are commonly used by researchers. Other representations are Hexagonal grid, Triangular grid, Navigation Mesh, and Waypoints.

Tile Grids:

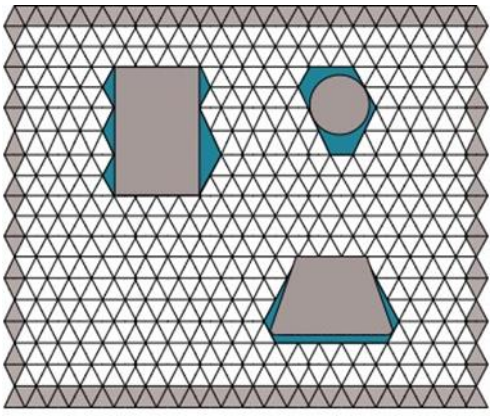
The composition of the grid includes vertices or points that are connected by edges. Basically, grids uniformly divide the map world into smaller groups of regular shapes called “tiles”. The movement in square tile grids can either be 4-way (no diagonal movement) or 8-way (diagonal movement). The second most widely used grid representations are Hexagonal grids . Hexagonal grids are like square grids with the same properties and take less search time and reduced memory complexities . Triangular grids are not popular among game developers and researchers but some methods are proposed to reduce the search effort and time consumption.



(a)



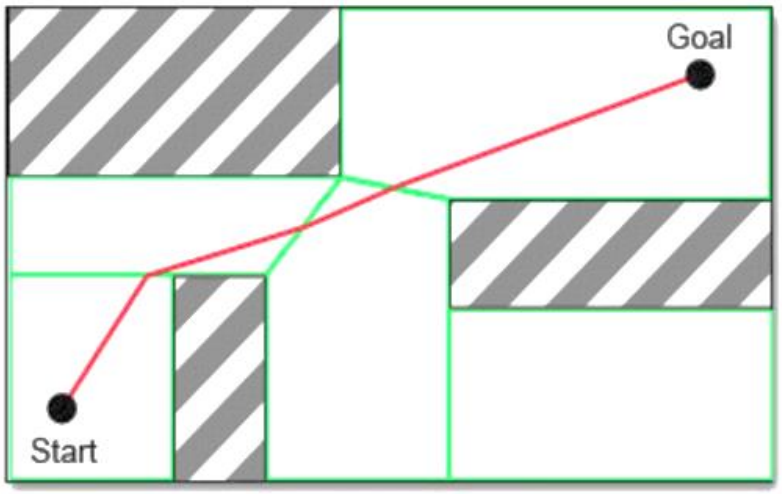
(b)

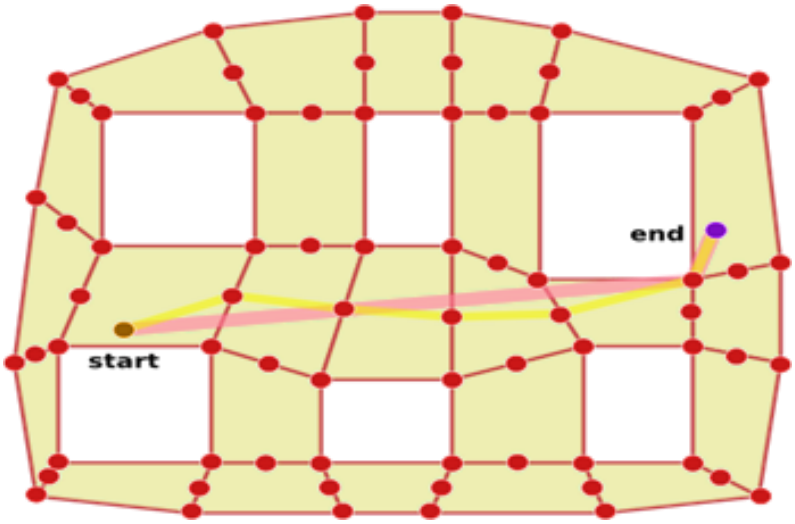


(c)

Navigation Mesh:

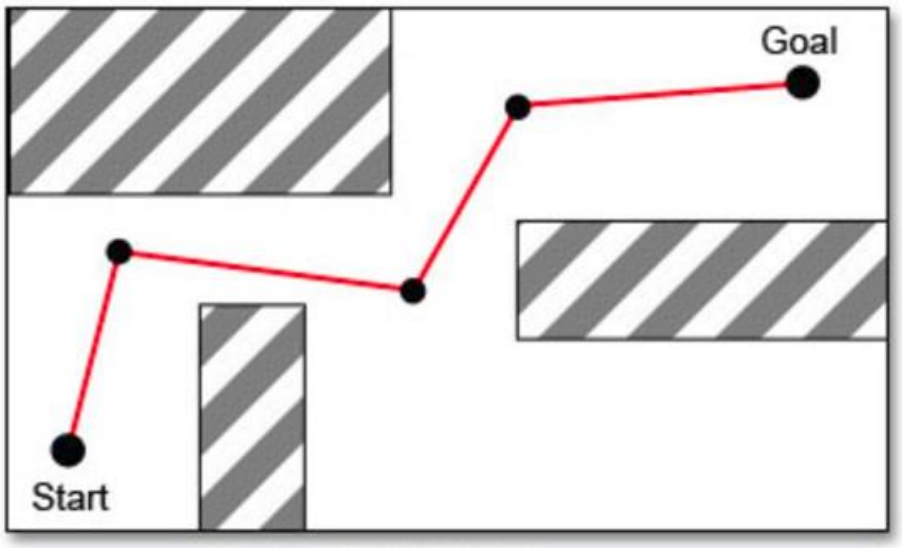
A navigation mesh is a connected graph of convex polygons, where the polygon is a node in a graph, also known as navmesh. Polygons represent a walkable area, thus movement in any direction is possible within the polygon. The map is pre-processed to generate nav-mesh and then the path can be found by traversing polygons (from polygon consisting start point to polygon consisting goal point). The benefits of using navigation mesh are that it reduces the number of nodes in the graph as the large walkable area can be represented as a single convex polygon, reduces the memory required to store pre-processed map, and increases the speed of pathfinding .





Waypoints:

A waypoint can be defined as a point along the path which can be marked manually or can be automatically computed. The purpose of waypoints is to minimize the path representation as the shortest path can be pre-computed between any two points. Therefore, certain optimization techniques are developed to compute the path using waypoints. The main advantage of waypoints can be in a static world as the map does not change, so the shortest paths between two waypoints can be pre-computed and stored, reducing the time to calculate the final path after execution.



Algorithms :

For finding a path between two nodes in a given graph a search algorithm is required. Many search algorithms have been developed for graph-based pathfinding. Pathfinding algorithm generally finds the path by expanding nodes and neighboring nodes according to some given criteria. Pathfinding algorithms can be broadly divided into two categories: Informed and Uninformed pathfinding Algorithms. In this project we use Dijkstra & A* aglorithms

INFORMED PATHFINDING ALGORITHM AND HEURISTICS :

As the name suggests informed means having prior information about the problem space before searching it. Informed search refers to the use of knowledge about the search space like problem map, estimated costs, an estimate of goal location. Thus, the algorithm utilizes this information while searching a path and it makes pathfinding fast, optimal and reduces memory usage in node expansion . Various algorithms that fall under this category are A* and many more. These algorithms use different heuristic functions or uniform cost function to utilize the 7 information of the search problem.

The following heuristic functions, used by these algorithms, are discussed briefly here:

Manhattan distance: It is considered as a standard heuristic for the square grid, defined as the sum of the absolute difference between the start and goal position cartesian coordinates. In pathfinding, the Manhattan distance is the distance between start node to goal node when the movement is restricted to either vertical or horizontal axes in a square grid.

The heuristic function is given below: $h(x) = |x1 – x2| + |y1 – y2|$

Octile Distance: Octile distance is the distance between two points when diagonal movement is possible along with horizontal and vertical. The Manhattan distance for going 3 up and then 3 right will be 6 units whereas only 3 units diagonally (octile distance).

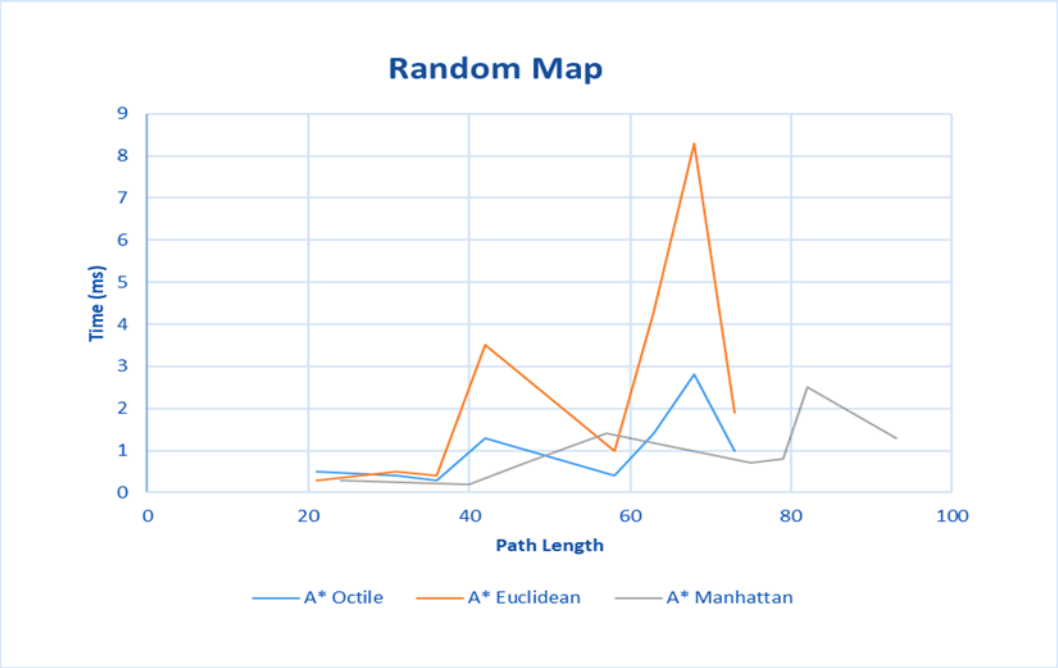
The function of octile distance is given below: $h(x) = \max((x1 – x2), (y1 – y2) + (\sqrt{2} -1) * \min((x1 – x2), (y1 – y2))$

Euclidean Distance: When any angle movement, not the grid directions (horizontal. vertical, diagonal), is allowed then the straight-line distance is the

shortest distance between any two points which is also known as Euclidean distance.

The function is given below: h(x) = sqrt (|x1 – x2| + |y1 – y2|)

In uniform cost search the next node is selected based on the cost so far, so the lowest cost node gets selected at each step. It is complete and optimal but not efficient as it takes lot of time to explore nodes.



Uninformed Pathfinding Algorithms :

Uninformed pathfinding refers to finding the path without any knowledge of the destination in the search space with only information about start node and adjacent nodes, also known as blind search. Thus, the algorithm blindly searches the space by exploring adjacent nodes to the current node. Breadth-first search, depth-first search, Dijkstra are some algorithms that fall under this category. Uninformed search is slow and consumes lots of memory in storing nodes as it searches whole space until the destination node is found. The uninformed pathfinding is also known as an undirected search approach, which simply does not spend any time in planning. It just explores the nodes that are connected with the current node and then explore their neighbor nodes and so on until finds the node marked as goal node.

Dijkstra's Algorithm :

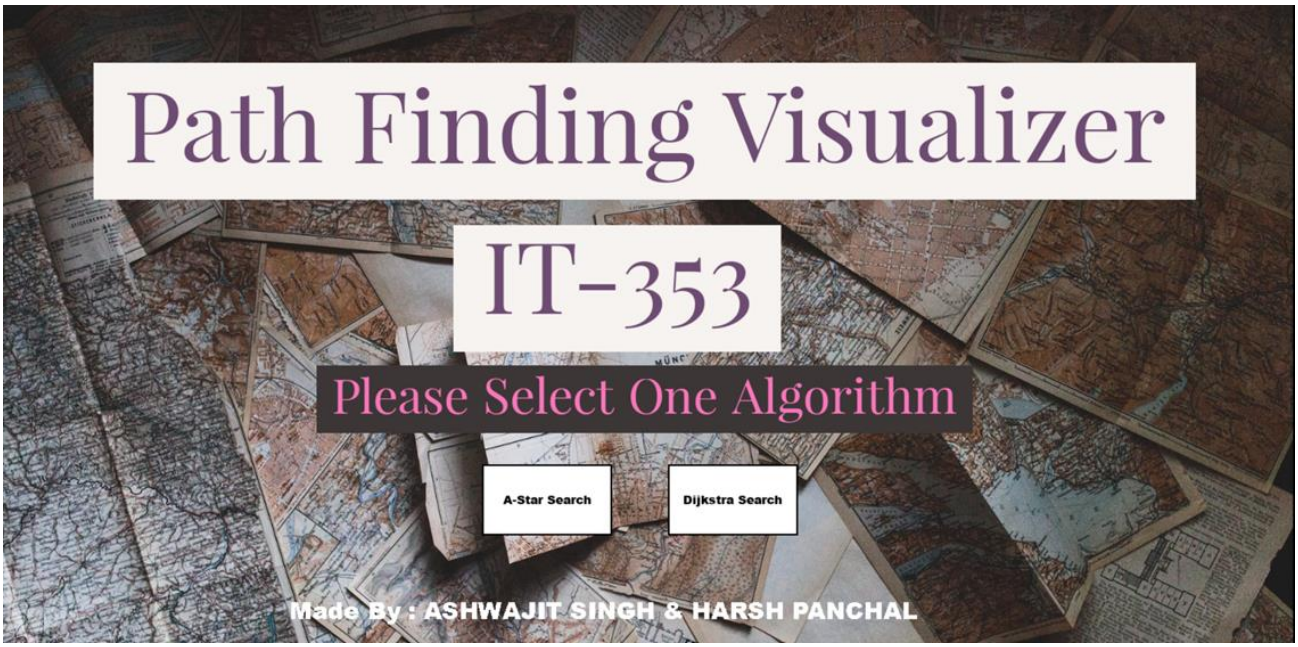
- It is a solution to the single-source shortest path problem in graph theory.
- It is a greedy method.
- It works on directes or non-directed graph
- Dijkstra's: It has one cost function, which is real cost value from source to each node: f(n)=g(n) It finds the shortest path from source to every other node by considering only real cost.
- It can be calculated by d[u] + c(u,v)<d[v]

A* Algorithm :

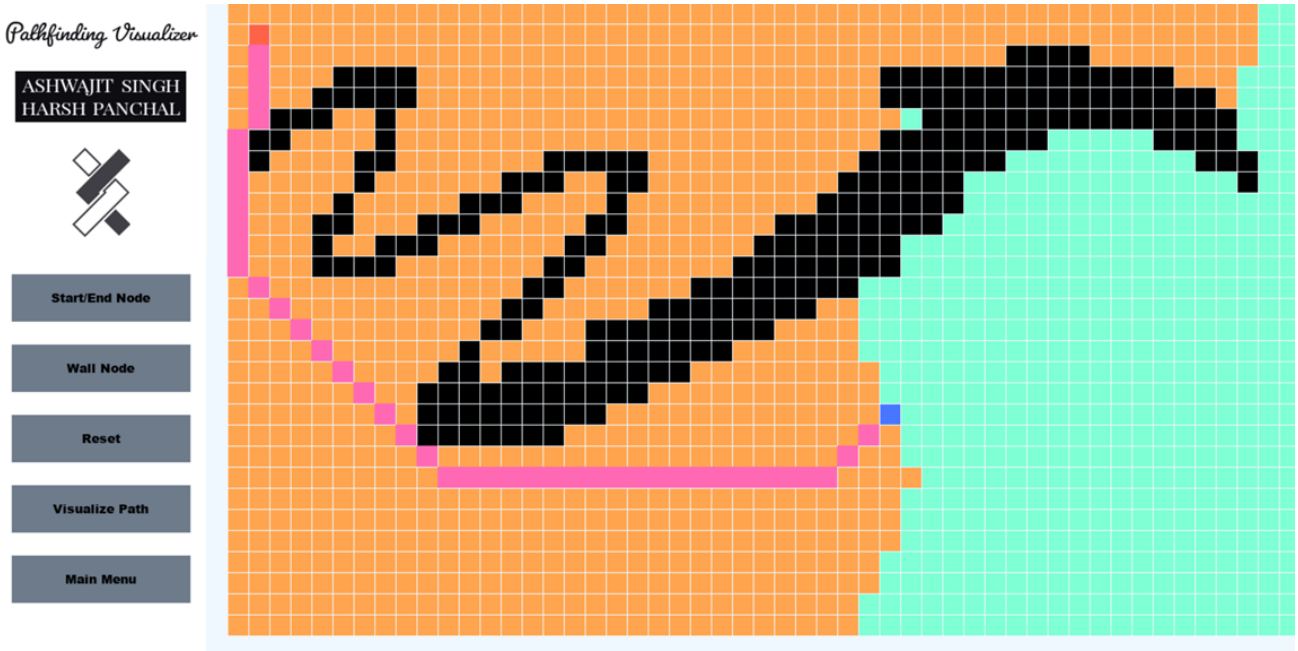
- A Star algorithm is a best first graph search algorithm that finds a least cost path from a given initial node to one goal node.
- Uses heuristic to guide search which means it is too optimistic, estimating the cost to be smaller than it actually is.
- A* is one of the many search algorithm that take an input, evaluates a number of possible paths and returns a solution.
- A* combines features of uniform-cost search and pure heuristic search to effectively compute optimal solutions
- A* evaluates nodes by combining g(n) and h(n) f(n) = g(n) + h(n)
- f(n) is called evaluation function.
- A* is both complete and optimal.

FINAL PROJECT IMAGES :

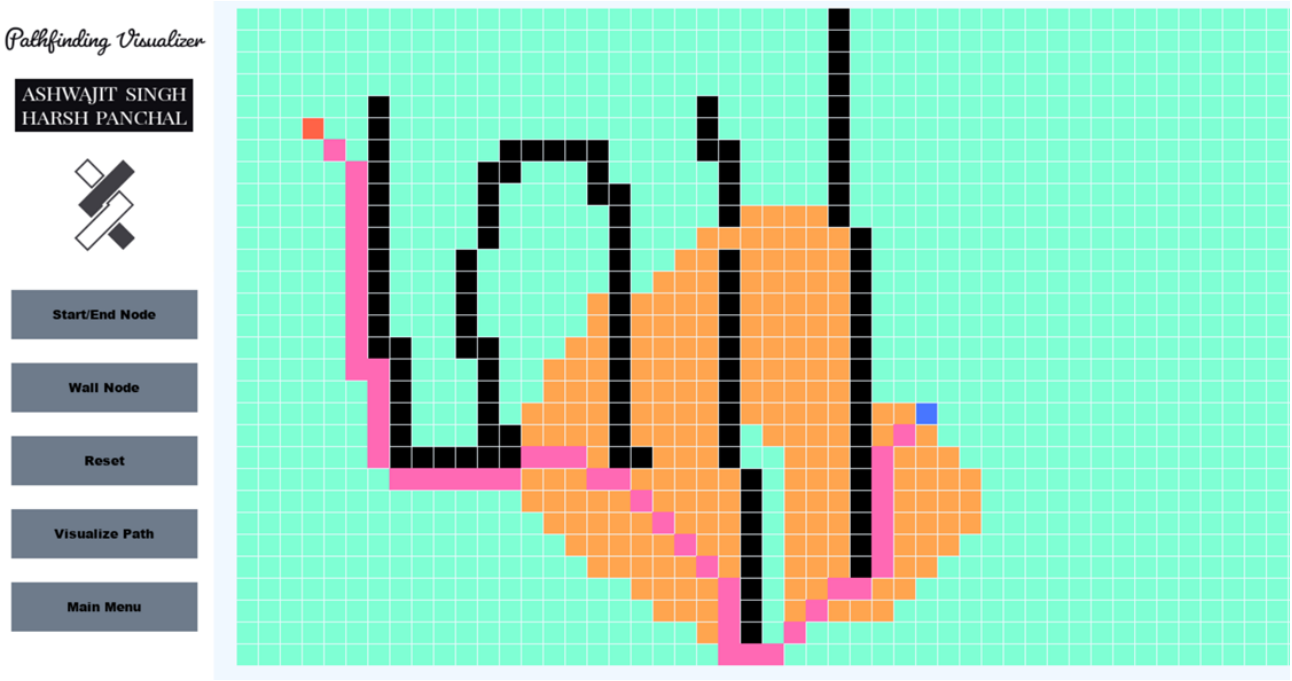
MAIN MENU :



Dijkstra Search :



A* :



CODE :

https://github.com/Harsh5chal/DSALGO.git

REFRENCES

- https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm
- https://www.geeksforgeeks.org/dijkstras-shortest-path-algorithm-greedy-algo-7/
- http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html
- https://scholar.uwindsor.ca/etd/8178
- https://scholar.uwindsor.ca/etd
- https://en.wikipedia.org/wiki/Pathfinding#:~:text=A%20common%20example%20of%20a,from%20the%20start%20is%20examined.&text=Dijkstra's%20algorithm%20fails%20if%20there%20is%20a%20negative%20edge%20weight.