iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** V **Month of publication:** May 2023

DOI: https://doi.org/10.22214/ijraset.2023.51795

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue V May 2023- Available at www.ijraset.com

Visualising Path Finding Algorithms Application Development and Implementing

Ujwal G B[1], Chandan Gowda H N[2], Dr. Sheshappa S. N[3], Prof. Mr. Vitesh Gowda M[4]
[1,2] B E Students, [3] Associate Professor, [4] Assistant Professor, Department of Information Science & Engineering, Sir M. Visvevaraya Institute of Technology, Bengaluru, Karnataka, India.

Abstract: ***The Dijkstra algorithm, A* Search, Greedy Best-first Search, Swarm Search, Breadth-first, and Depth-first Search are some of the common algorithms used today. The pathfinding algorithm is used in this study to provide an overview of algorithms and how they are implemented. The user will also learn more about how different algorithms and programming in general work. Knowing these tactics will give them a fundamental understanding of how to create different navigational tools. The visualizer is a grid page containing a "start node" and a "end node." In order to enhance the overall picture and better understand how these pathfinding algorithms deal with our everyday problems, the spectator can add new characteristics like a maze, walls, and weights. To build a visualizer, a programmer needs a solid grasp of front-end programming languages and pathfinding techniques.***

Keywords: Dijkstra's, A*, Greedy Best-First Search, Swarm, Convergent Swarm, Bidirectional Swarm Breadth-First Search, Depth-First Search Algorithms.

I. INTRODUCTION

A notable observation and quote attributed to computer scientist Donald Knuth was, "An algorithm must be seen to be believed." With the aid of this gadget, we are carrying out the identical action.

Both experienced and new programmers can benefit from using this tool to better understand algorithms. It is a web application with a 19x55 grid with start and end nodes that may be placed anywhere on the grid by the user. Walls and weights can also be added by the user to build a realistic habitat. Choose an algorithm and use it to illustrate how the chevalier algorithm functions once you have constructed the grid. The speed of the algorithm can be investigated, and the user can watch how long it takes to complete a task.

II. LITERATURE REVIEW

A study of the literature is necessary to assess aspects that have not been covered in other studies. Many researchers try to comprehend different kinds of conclusions, and a literature review is necessary to improve earlier findings. The research and debate are built on a number of intriguing components from the present literature.

A significant area of mathematical theory (networks) is the mathematical analysis of the graph-based abstract interactions between objects. These structures can be used to simulate pairwise relationships in a number of real-world systems, despite the fact that their study is entirely theoretical. The finding of shortest paths is one of the most widely used applications in a variety of practical applications, including maps, robot navigation, texture mapping, typesetting in TeX, urban traffic planning, optimal pipelining of VLSI chips, subroutines in advanced algorithms, telemarketer operator scheduling, routing of telecommunications messages, approximating piecewise linear functions, network routing protocols (OSPF, BGP, RIP), and utilising arbitrage opportunities.

III.METHODOLOGY

The entire operation of the project is discussed in this section. How the project began, how it functions, and how the many phases of the project were completed, as well as the problems encountered at each level.

What is the purpose of the project?

A pathfinding algorithm's primary goal is to determine the shortest route between two places.

This project shows multiple pathfinding algorithms in action, as well as other things!

All of the algorithms on this project have been converted for a 2D grid, with a "cost" of 1 for 90 degree turns and 1 for moves from one node to another.

©IJRASET: All Rights are Reserved

A. *Data Structures*

The two types of data structures that are most frequently used in practise to describe graphs are adjacency lists and adjacency matrices. Both data structures are high-level arrays that are indexed by vertices; as a result, each vertice must have a distinct integer identification between 1 and V. In a formal sense, these integers are the vertices.

B. *Adjacency Lists*

The most common data structure for storing graphs is the adjacency list. An adjacency list is a group of lists, each of which contains one of the neighbours (or out-neighbors, if the graph is directed) of a vertice. In undirected graphs, each edge uv is saved twice—once in the tail u's neighbour list and once in the v's neighbour list. In directed graphs, however, each edge uv is only saved once. For both kinds of graphs, the total amount of space needed for an adjacency list is O(V + E).

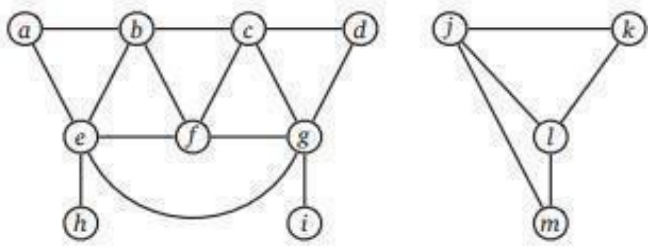
There are several ways to encode these neighbour lists, but the typical implementation uses a straightforward singly-linked list. By scanning a node's neighbour list, we may list the (out-)neighbors of a node v in O(1 + deg(v)) time. In a similar vein, determining if uv is an edge can be done in O(1 + deg(u)) time by scanning u's neighbour list. If we scan the neighbour lists of both u and v simultaneously and stop either when we discover the edge or when we reach the end of a list, we can reduce the time for undirected graphs to O(1 + mindeg(u), deg(v)).

C. *Adjacency Matrices*

The other typical data structure for graphs is the adjacency matrix, which was created by Georges Brunelin. A graph's adjacency matrix, or A[1... V, 1... V], is a V V matrix of 0s and 1s, with each element indicating whether a particular edge is present in G. This matrix is typically represented as a two-dimensional array. If the graph is undirected, A[u, v]:= 1 for all vertices u and v, and if the graph is directed, A[u, v]:= 1 for all vertices u and v, if and only if uv 2 E.

The adjacency matrix for undirected graphs is always symmetric, meaning that A[u, v] = A[v, u] for all vertices u and v. This is because uv and vu are essentially different names for the same edge and the diagonal members of A[u, u] are all zeros. Directed graph adjacency matrices may or may not be symmetric, and their diagonal members may or may not be zero.

	a	*b*	*c*	*d*	*e*	*f*	*g*	*h*	*i*	*j*	*k*	*l*	*m*
a	0	1	0	0	1	0	0	0	0	0	0	0	0
b	1	0	1	0	1	1	0	0	0	0	0	0	0
c	0	1	0	1	0	1	1	0	0	0	0	0	0
d	0	0	1	0	0	0	1	0	0	0	0	0	0
e	1	1	0	0	0	1	1	1	0	0	0	0	0
f	0	1	1	0	1	0	1	0	0	0	0	0	0
g	0	0	1	1	1	1	0	0	1	0	0	0	0
h	0	0	0	0	1	0	0	0	0	0	0	0	0
i	0	0	0	0	0	0	1	0	0	0	0	0	0
j	0	0	0	0	0	0	0	0	0	0	1	1	1
k	0	0	0	0	0	0	0	0	0	1	0	1	0
l	0	0	0	0	0	0	0	0	0	1	1	0	1
m	0	0	0	0	0	0	0	0	0	1	0	1	0



D. *Picking an Algorithm*

Choose an algorithm from the "Algorithms" drop-down menu. It's important to keep in mind that some algorithms are weighted while others are not. Unweighted algorithms do not take turns or nodes into account, but weighted algorithms do.

Additionally, not every algorithm guarantees the shortest path.

E. *Meet the Algorithm*

1) Dijkstra's Algorithm(weighted): The first pathfinding algorithm, ensures the shortest path. The "source node" is one of the nodes that Dijkstra's Algorithm uses to determine the shortest route between that node and every other node in the graph. In order to determine the path that minimises the overall distance (weight) between the source node and all other nodes, this technique leverages the edge weights.

2) A* Search (weighted): One of the most popular path-finding and graph traversal methods is the A* Search algorithm. This method is quite effective and ingenious. A* is a best-first search algorithm that is informed, meaning it is written in terms of weighted graphs. Starting from a particular starting node, it seeks to discover a path to the specified target node that incurs the lowest cost (least amount of time, least amount of distance, etc.). It accomplishes this by keeping track of a tree of paths leading from the start node and extending each of those paths by one edge until its termination requirement is met.
3) Greedy Best-First Search (weighted): A faster, more heuristic-intensive variant of A* does not guarantee the existence of the shortest path.
4) Swarm Algorithm (weighted): A mixture of Dijkstra's and A*.
5) Convergent Swarm Algorithm (weighted): The Swarm algorithm that is quicker and more heuristic-focused.
6) Bidirectional Swarm Algorithm (weighted): Swarm from both sides.
7) Unweighted Breath-First Search: A fantastic algorithm that ensures the shortest path.
8) Unweighted Depth-first search: A poor pathfinding technique that does not guarantee the shortest path.

F. Adding Walls

To add a wall, click on the grid.
A road cannot pass through walls because they are impenetrable.
More on visualisation: To visualise algorithms and perform other tasks, use the navbar buttons.
From the navbar, you can clear the current path, clear walls and weights, clear the entire board, and change the visualisation speed.
To return to this tutorial, go to the upper left corner of your screen and select "Pathfinding Visualizer."

IV. OBJECTIVE

1) It can be used as an electronic learning aid to learn about algorithms.
2) It is used to determine the shortest path.
3) It is utilised in the telephone system.
4) In IP routing, it is used to find the Open Shortest Path First.
5) It is used in geographical maps to discover map locations that refer to graph vertices.
6) We can create a GPS system that will direct you to the appropriate destinations.
7) BFS uses search engine crawlers to create indexes. It finds all links in the original page to get new pages, starting with the source page.
8) BFS is used to find all neighbour nodes in peer-to-peer networks like BitTorrent.
9) People that utilise wireless technology expect high data rates for Voice, Video, and other applications that exceed Gigabytes per second. There are numerous standards for exceeding GB/s data speeds. MIMO (Multi input Multi output) is one of the standards. To discover the shortest partial Euclidian distances, MIMO uses the K-best Technique (a Breadth-First Search algorithm).

V. RESULT ANALYSIS

Six development phases were assigned to this project. These stages deal with every part of the project, from data collection to processing to user output.

1) The first part is the creation of the graph matrix.
2) Including obstacles and event listeners.
3) Include the algorithms for graphs.
4) Include the feature for pathfinding.
5) Enhance the user interface and design.
6) Timer capabilities were added. The project is now available for the user to utilise after completing all of these stages.

Now that the visualizer is complete, let's use it. Choose an algorithm of your choice to start with.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue V May 2023- Available at www.ijraset.com

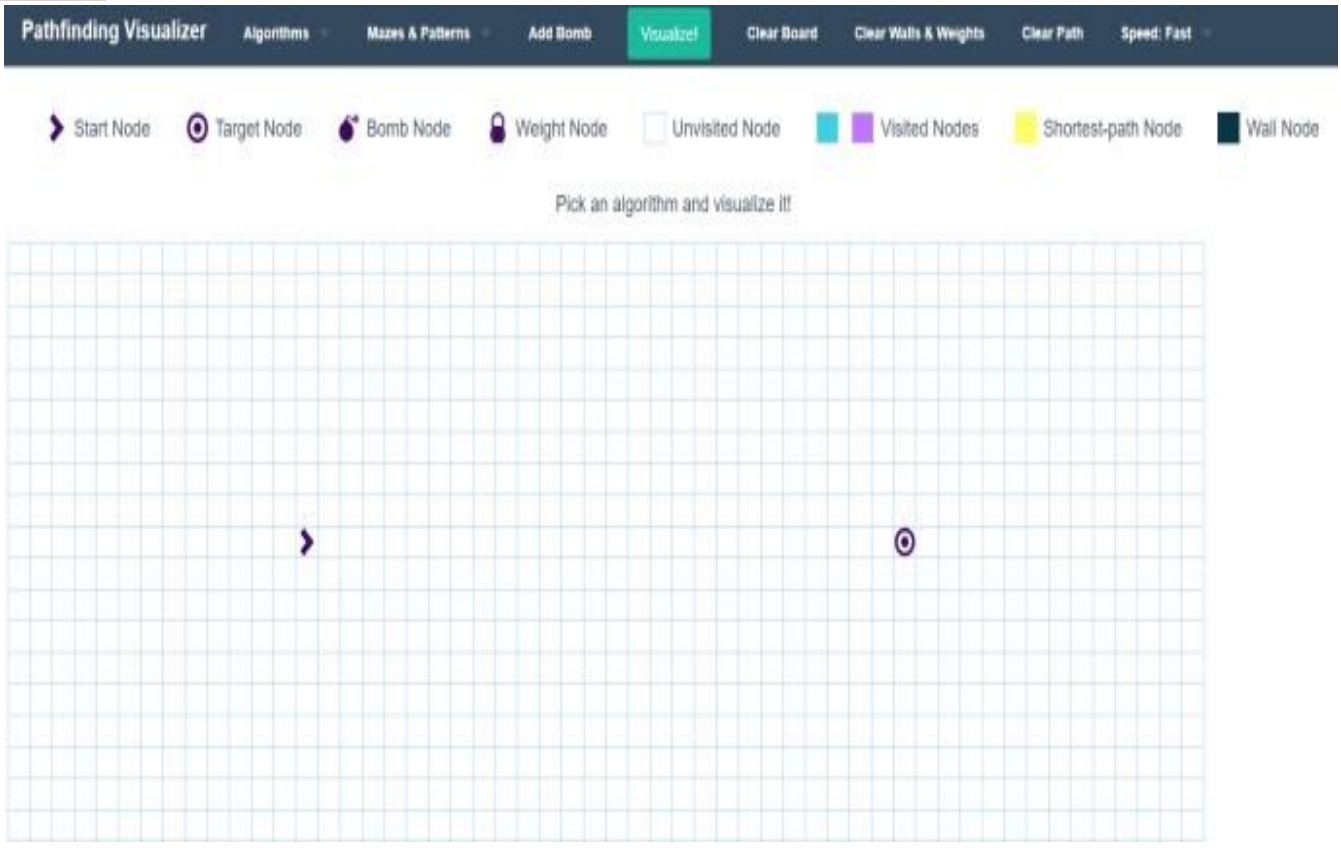


Fig 1. Now include walls and weights.(optional)

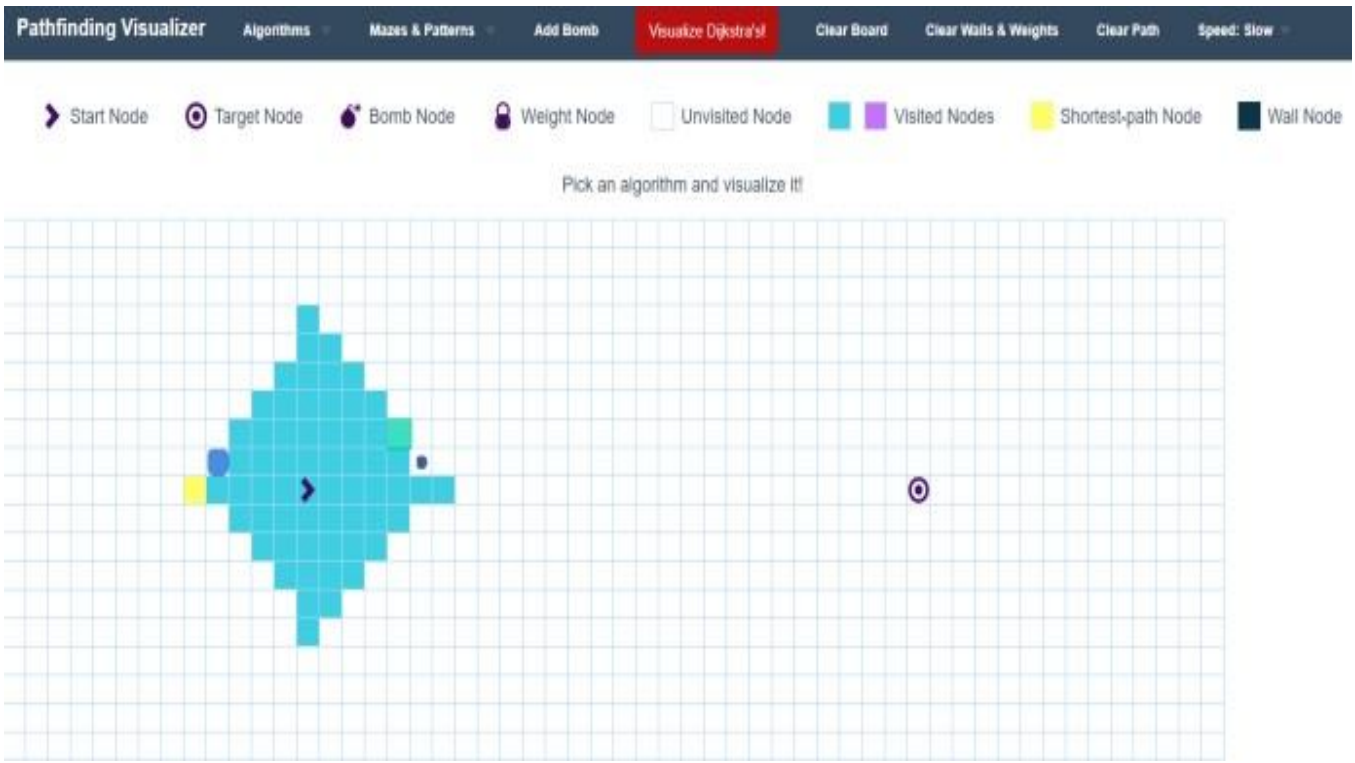


Fig 2. The algorithm you select will be visualised when you click "visualise."

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



Fig 3. The shortest path will be found after a short while, and it will be marked with a yellow line.

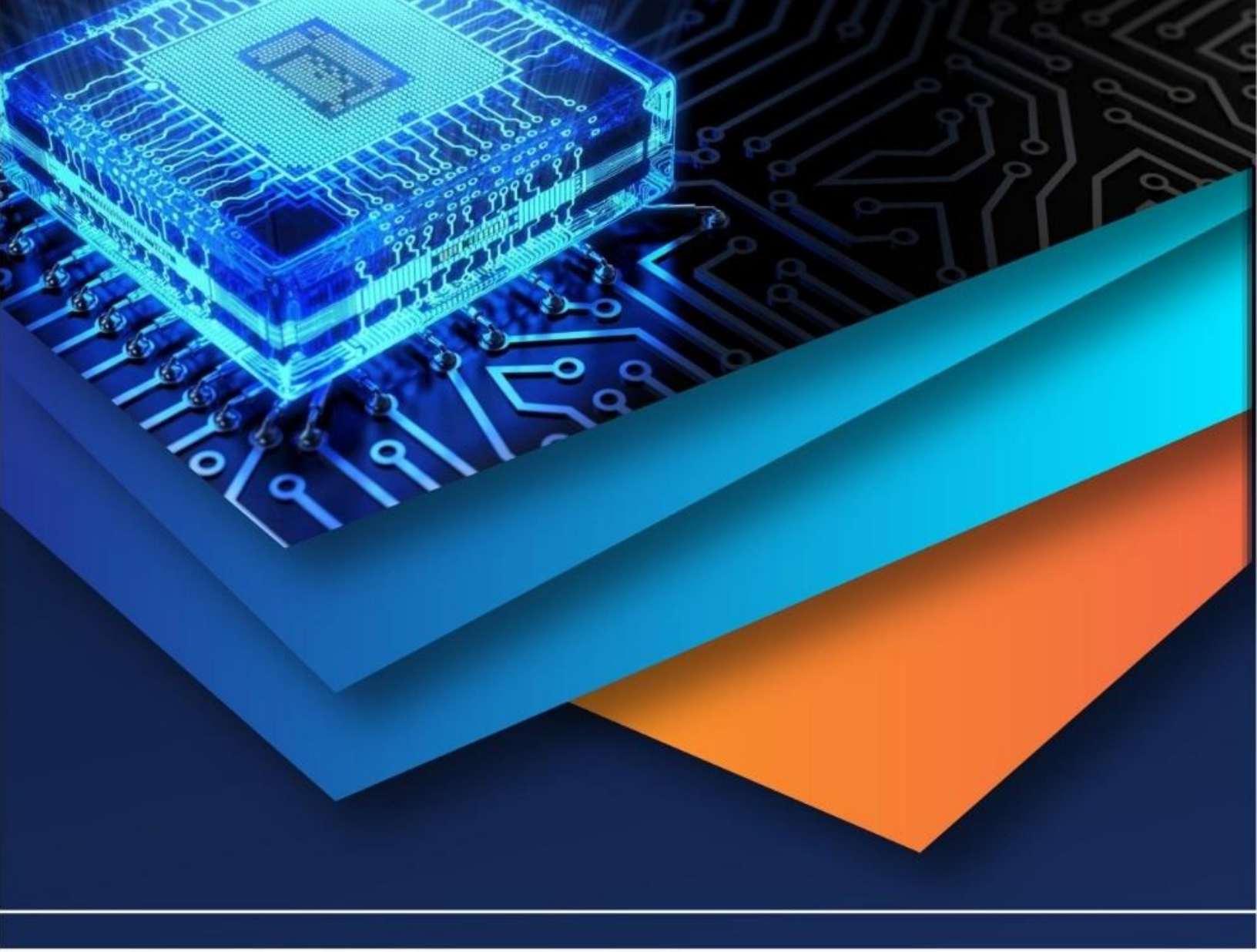
Future work : For the time being, the programme only has a small number of pathfinding algorithms; we'd like to add a lot more and visualise them in both 2-D and 3-D. We'd also like to offer more comparison tools for these methods.

VI. CONCLUSION

The pathfinding visualizer presented in this research was implemented and created entirely with free source off-the-shelf software. We made it simple to understand and learn about the algorithms by creating this project.

REFERENCES

[1] https://en.wikipedia.org/wiki/Algorithm#:~:text=Algorithms%20are%20always%20unambiguous%20and,automated%20reas oning%2C%20and%20other%20tasks.&text=As%20an%20effective%20method%2C%20an,language%20for%20calculatin g%20a%20function.
[2] https://www.geeksforgeeks.org/dijkstras-s hortest-path-algorithm-greedy-algo-7/
[3] https://www.geeksforgeeks.org/a-search-algorithm/
[4] https://www.geeksforgeeks.org/best-first- searchinformed-search/
[5] https://www.geeksforgeeks.org/breadth-firstsearch-or-bfs-for-a-graph/
[6] https://www.geeksforgeeks.org/depth-firstsearch-or-dfs-for-a-graph/
[7] https://www.w3schools.com/js/default.asp
[8] https://www.meta-chart.com/histogram
[9] Roles J.A. & ElAarag H. (2013). A Smoothest Path algorithm and its visualization tool. Southeastcon, In Proc. of IEEE, DOI: 10.1109/SECON.2013.6567453



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)