A Study on the Performance of the A-Star Algorithm with Various Heuristics in Grids and Graphs

Preprint · November 2023

DOI: 10.13140/RG.2.2.31372.49287

2 authors:

Sai Prasad Gudari
SRM Institute of Science and Technology
2 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Dr. Vadivu G
SRM Institute of Science and Technology
74 PUBLICATIONS **290** CITATIONS
SEE PROFILE

A Study on the Performance of the A-Star Algorithm with Various Heuristics in Grids and Graphs

Sai Prasad Gudari
Department of Computing Technologies
SRM Institute of Science and Technology
Chennai, India
gudarisaiprasadgudari@gmail.com

Dr.Vadivu G
Department of Computing Technologies
SRM Institute of Science and Technology
Chennai, India
vadivug@srmist.edu.in

Abstract—Informed Search Algorithms such as A-star uses different Heuristics compared to traditional uninformed search algorithms but choosing suitable heuristic can impacts the performance and efficiency of the algorithm in that particular problem space. This study explores the applications of different heuristics in A-star Algorithm in various environments such as grid-based and graph-based. This study is based on the comparative analysis of most-commonly used heuristics such as Manhattan, Euclidean and Chebychev providing the insights on performance in diverse contexts. Using this systematic evaluation, this research seeks to get the insights of the dynamics which influence the heuristic performance, and offering practical guidance on the selection of heuristic to improve the A star algorithm efficiency in various environments. Ultimately this paper selects two different environments grid and graph and three Heuristics Manhattan, Euclidean and Chebychev and test the performance of these three heuristics and provide detail explanation

Keywords—A-star Algorithm, pathfinding, heuristic, Manhattan distance, Euclidean distance, Chebychev Distance, Environment, Grid, Graph, Nodes and Edges, Heuristic Selection, Artificial Intelligence

I. INTRODUCTION

Pathfinding's primary objective is to identify the most efficient route between two distinct points. Among the algorithms developed for this purpose, the A* (A-star) algorithm stands out due to its efficiency and effectiveness. This algorithm, created by Peter Hart, Nils Nilsson, and Bertram Raphael in 1968, has been widely adopted in diverse fields such as transportation, maze-solving, robotics, and artificial intelligence, thanks to its stellar performance and accuracy[2]. The A* algorithm consistently manages to locate the shortest possible path, provided one exists.

II. APPLICATIONS OF A-STAR ALGORITHM

A. Transportation

In the realm of transportation, the A* algorithm powers GPS navigation systems, enabling the calculation of the quickest route between two locations[2].

B. Robotics

In the context of robotics, the A* algorithm facilitates autonomous robots in navigating complex environments[2].

C. Video Game Industry

The A* algorithm's impact extends to the dynamic and immersive world of the video game industry, where it serves as a cornerstone for efficiently guiding characters through complex game environments. Its versatility in handling diverse scenarios and optimizing pathfinding has made it a staple in game development.

D. Real-World Applications

This adaptability and efficiency also carry over to real-world applications, empowering autonomous vehicles, robotic systems, and other technologies to navigate intricate terrains and bustling city streets[8].

E. Research Gap

Despite the widespread use of these heuristics, there hasn't been a thorough comparison study of how well they function in various contexts. There is a particular shortage of investigation into grid-based and graph-based environments, which are frequently used in applications like video game creation and autonomous vehicle navigation.

F. Objectives of Our Research

Our work fills this gap by giving a thorough comparative evaluation of the Manhattan, Euclidean, and Chebyshev heuristics' performance in these contexts when used with the A* algorithm. The environments selected provide a thorough knowledge of the potential benefits and drawbacks of each heuristic by representing a wide range of scenarios found in practical applications.

G. Significance of Heuristic Selection

Through this investigation, we hope to shed light on the frequently undervalued significance of heuristic selection and its effects on the efficiency and performance of the A* algorithm. We aim to provide practitioners and researchers with insightful information, enabling them to make educated heuristic decisions suited to specific problem areas.

H. Future Findings

These insights could pave the way for more efficient pathfinding solutions, enhancing the functionality and performance of myriad applications reliant on the A* algorithm. We hope that our findings will stimulate further research on this intriguing topic.

III. METHODOLOGY

This research is conducted to evaluate the performance of A-star algorithms with three commonly used heuristics

Manhattan, Euclidean and Chebychev. This study is conducted in two different environments grid-based and graph-based with increasing sized and probability of obstacles. This research is aimed to provide the insights of how the heuristics choice influences algorithm efficiency and performance in diverse scenarios.

A. Grid Environment

The grid environment represents a 2d space with equal distance nodes like a chess board or pixel-based map. Obstacles are randomly placed throughout the grid at random points with different probabilities to emulate the real-world scenarios. Our aim is to find the optimal path between these obstacles using these three heuristics Manhattan, Euclidean and Chebychev and provide the insights of their performance and efficiency

B. Graph Environment

In graph environment nodes and edges are explicitly defines, allowing for more abstract representation of connections. This environment might represent a 2d map and nodes represent cites and edges could represent the roads. One special in graph environment is certain rows and columns, designated as highways are assigned lower edge weights and other rows and columns are treated as local roads and are assigned higher edge weights[6]. This provides a more realistic representation of real-world scenarios where some roads allow for faster travel than the other. Even with in this environment, the obstacles are placed randomly with increasing probabilities. In this study we will find how these heuristics react with these scenarios.

C. Heuristics

Heuristics play an important part in A-star (A*) algorithm by providing an estimate of the cost from a given node to the goal, steering the search towards plausible paths. The formula for A-star combines the actual cost to reach the current node often denoted as g(n) with the heuristic estimation of the cost from current node to the goal often denoted as h(n). The sum $f(n) = g(n)+h(n)$ represents the total estimated cost of the cheapest solution through the node. This informed guidance differentiates A* from uninformed search strategies, enabling it to often find solutions more efficiently. The heuristic should ideally not exaggerate the real cost of reaching the goal from a given state; a heuristic that follows this concept guarantees that A* will discover an optimal path[5]. A* balances the trade-off between exploration of the search space and direct pursuit of the objective by leveraging the predictive potential of heuristics, making it one of the most effective and commonly used pathfinding algorithms in numerous areas.

Manhattan Distance:

Manhattan Distance, also known as city block distance and it is used the calculate the distance between two points in a grid-based path. By considering only vertical and horizontal movements and no diagonal movement allowed. The name derives from the grid layout of most streets in Manhattan, which forms square block.

Mathematically:

Let's say if we take points in a 2d place $p1(x1,y1)$ and $p2(x2,y2)$ and the distance between these two points can be calculated as combing the absolute values of differences between x points and y points.

$$d(p1p2) = |x1 - x2| + |y1 - y2| .$$

Euclidean Distance:

The Euclidean distance represents the shortest distance between two points in a 2D or 3D space. It is the length of the line segment connecting the two points and is often used in plane geometry and trigonometry. This heuristic assumes that the agent can move in any direction, including diagonals, making it suitable for environments where diagonal movements are permitted.

Mathematically, for two points $(p1(x1,y1))$ and $(p2(x2,y2))$ in a 2D plane, the Euclidean distance is computed as:

$$d(p1,p2) = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}$$

Chebyshev Distance:

Chebyshev distance, named after Pafnuty Chebyshev, is a metric that defines the distance between two points in a grid, allowing for diagonal movement. It is especially relevant in chessboard-like environments where eight possible movement directions exist (including diagonals). It essentially captures the maximum of the absolute difference in x and y coordinates.

Mathematically, for two points $(p1(x1,y1))$ and $(p2(x2,y2))$, the Chebyshev distance is:

$$d(p1,p2) = max(|x2 - x1|, |y2 - y1|)$$

IV. Experiment Design

For both environments, the A* algorithm was executed with each of the three heuristics. Several variables were systematically altered to study their impact on the performance of the heuristics:

Obstacle Probabilities:

The environments were subjected to different obstacle probabilities, ranging from 0.1 to 0.3. This was done to assess the heuristics' resilience and efficiency in scenarios of varying complexity. The introduction of obstacles at varying probabilities simulates real-world challenges, where paths may be obstructed or altered due to unforeseen circumstances or changes in the environment.

Environment Sizes:

The size of the environments was another variable of interest, with sizes ranging from a minimal 50x50 grid/ graph to a much larger 500x500 configuration. By altering the size, we aimed to understand how the sheer scale of an environment might influence the performance of the A* algorithm with different heuristics. Larger environments naturally present more potential paths and increased computational challenges, and observing heuristic performance under these conditions provides valuable insights into their scalability and efficiency.

For each combination of heuristic, obstacle probability, and environment size, the primary metrics of interest were the execution time and path length, nodes expanded and memory used. The execution time offers a direct measure of the algorithm's speed, while the path length serves as an indicator of the quality or efficiency of the identified path.

V. Results and Analysis

A. *Graph Environment*

In this section we'll analyze the behaviour of three heuristics in the graph environment, focusing on how each heuristic performs as probability and size increases.

1. With Increasing Obstacle Probabilities

TABLE I
PERFORMANCE WITH INCREASING PROBABILITIES IN GRAPH

Heuristics	***Probabilities***		
	0.1	0.1	0.3
Chebychev	0.008645	0.002809	0.001167
Euclidean	0.007776	0.002434	0.000993
Manhattan	0.007945	0.002518	0.001247

All heuristics exhibit a **decreasing trend** in the number of nodes expanded with increasing probabilities.

The differences in nodes expanded among the heuristics are subtle, with **Manhattan** expanding slightly fewer nodes at each probability.

Chebyshev and **Euclidean** heuristics, while having almost identical numbers, see Chebyshev expanding a tad more nodes at each probability.

TABLE II
NO OF NODES EXPANDED WITH INCREASING PROBABILITIES IN GRAPH

Heuristics	***Probabilities***		
	0.1	0.1	0.3
Chebychev	588.578947	195.675676	88.000000
Euclidean	586.815789	195.297297	87.914286
Manhattan	583.815789	194.810811	87.828571

All heuristics exhibit a **decreasing trend** in the number of nodes expanded with increasing probabilities.

The differences in nodes expanded among the heuristics are subtle, with **Manhattan** expanding slightly fewer nodes at each probability.

Chebyshev and **Euclidean** heuristics, while having almost identical numbers, see Chebyshev expanding a tad more nodes at each probability.

TABLE III
MEMORY CONSUMPTION WITH INCREASING PROBABILITIES IN GRAPH

Heuristics	***Probabilities***		
	0.1	0.1	0.3
Chebychev	393.263158	241.081081	192.914286
Euclidean	430.315789	241.081081	192.914286
Manhattan	405.052632	243.459459	192.914286

All heuristics show a **decreasing trend** in memory consumption with increasing probabilities.

Euclidean heuristic starts with the highest memory consumption at 0.1 probability, but by 0.2 and 0.3 probabilities, its consumption aligns with Chebyshev's.

Manhattan heuristic's consumption is between the other two at 0.1 probability, but it slightly exceeds Euclidean's consumption at 0.2 probability.

2. With Increasing Graph Sizes:

We'll visualize the relationship between graph size and the three metrics (execution time, nodes expanded, and memory consumption) for each heuristic.

Execution Time vs Graph Size:

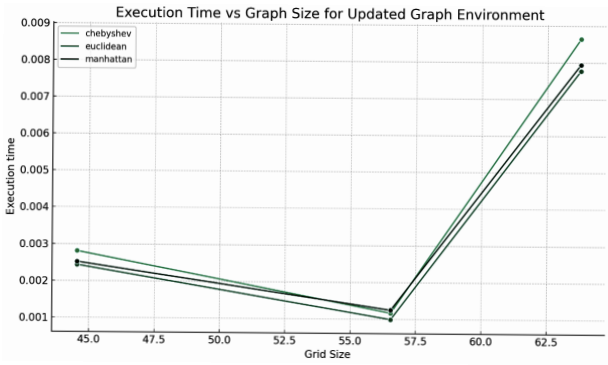

Fig. 1. Execution Time with increasing obstacle probabilities in graph

Chebyshev Heuristic: The execution time shows a consistent increase as the graph size grows. This indicates that as the graph becomes more complex, the heuristic takes longer to find a solution.

- **Euclidean Heuristic**: Similar to Chebyshev, there's an upward trend in execution time with graph size. However, the rate of increase appears slightly steeper, especially in larger graphs.
- **Manhattan Heuristic**: This heuristic's execution time rises with graph size but remains lower than the other two heuristics, emphasizing its efficiency in the graph environment.

Number of Nodes Expanded vs Graph Size:

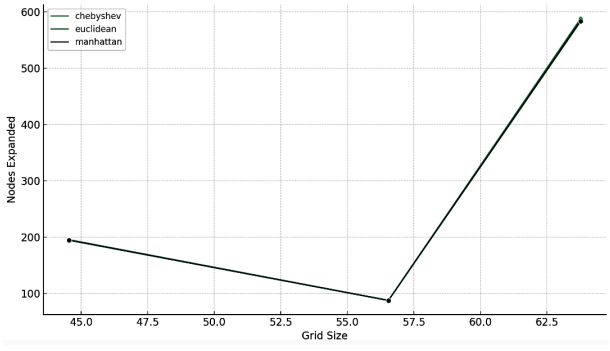

Fig. 2. No. Of Nodes Expanded with increasing obstacle probabilities in graph

- **Chebyshev Heuristic**: The number of nodes expanded grows with the graph size, suggesting a direct approach in searching for the solution.

• **Euclidean Heuristic**: There's also an increase in nodes expanded with graph size, but the growth rate is less pronounced compared to Chebyshev.

• **Manhattan Heuristic**: Consistently expands the fewest nodes across varying graph sizes, further highlighting its direct approach.

Memory Consumption vs Graph Size:

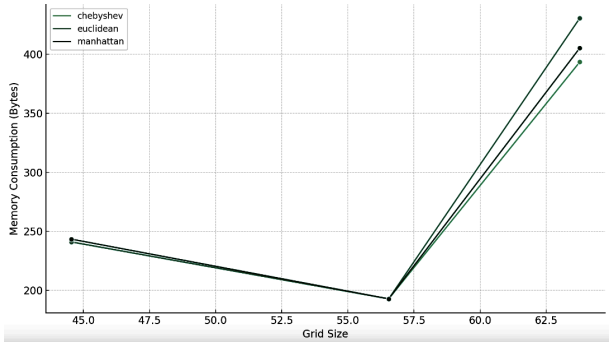

Fig. 3. Memory Consumption with increasing obstacle probabilities in graph

- **Chebyshev Heuristic**: Memory consumption starts high for smaller graph sizes but shows a declining trend as the graph size increases.
- **Euclidean Heuristic**: Displays a similar pattern to Chebyshev in terms of memory consumption, starting high and reducing with graph size.
- **Manhattan Heuristic**: Starts with the highest memory consumption but sees a significant reduction as the graph size grows, making it more memory-efficient for larger graphs compared to the other two heuristics.

Key Insights:

- **Manhattan** heuristic continues to be both time-efficient and direct in its approach in the graph environment, as evidenced by its lower execution time and fewer nodes expanded. However, it requires more memory in smaller graph sizes.
- The **Chebyshev** and **Euclidean** heuristics demonstrate a direct relationship between graph size and both execution time and nodes expanded. Their memory consumption patterns differ slightly, with Euclidean generally consuming more memory across varying graph sizes.

B. Grid Environment

We will analyze the behavior of the three heuristics in the updated grid environment by focusing on how each heuristic performs as the probability and grid size increases.

1. With Increasing Probabilities

TABLE IV.
PERFORMANCE WITH INCREASING PROBABILITIES IN GRID

Heuristics	***Probabilities***		
	0.1	0.1	0.3
Chebychev	0.012045	0.008456	0.003218
Euclidean	0.012685	0.008585	0.003261
Manhattan	0.004056	0.002170	0.000893

From the numbers, we can observe:

- All heuristics show a **decreasing trend** in execution time with increasing probabilities.
- The **Manhattan** heuristic continues to have the fastest execution time across all probabilities.
- **Chebyshev** and **Euclidean** heuristics have similar execution times, with Euclidean being slightly slower at each probability.

TABLE V.
NO OF NODES EXPANDED WITH INCREASING PROBABILITIES IN GRID

Heuristics	***Probabilities***		
	0.1	0.1	0.3
Chebychev	629.526316	424.702703	190.228571
Euclidean	524.105263	352.324324	152.400000
Manhattan	303.000000	157.000000	72.857143

- All heuristics show a **decreasing trend** in the number of nodes expanded with increasing probabilities.
- The **Manhattan** heuristic consistently expands the fewest nodes, highlighting its efficiency in navigating through the grid.
- **Chebyshev** heuristic, while reducing the nodes expanded with increasing probability, still expands more nodes than the other two heuristics across all probabilities.

TABLE VI
MEMORY CONSUMPTION WITH INCREASING PROBABILITIES IN GRID

Heuristics	***Probabilities***		
	0.1	0.1	0.3
Chebychev	1832.842105	3817.729730	1215.314286
Euclidean	2522.315789	3658.810811	771.885714
Manhattan	4980.631579	3340.324324	541.714286

- The **Manhattan** heuristic starts with the highest memory consumption at 0.1 probability but shows a significant reduction in consumption as the probability increases.
- Both **Chebyshev** and **Euclidean** heuristics show an initial increase from 0.1 to 0.2 probabilities, followed by a drop in memory consumption at 0.3

probability. However, Chebyshev's consumption at 0.2 probability exceeds that of Euclidean.

2. With Increasing sizes

We'll visualize the relationship between grid size and the three metrics (execution time, nodes expanded, and memory consumption) for each heuristic.

Execution Time vs Grid Size:

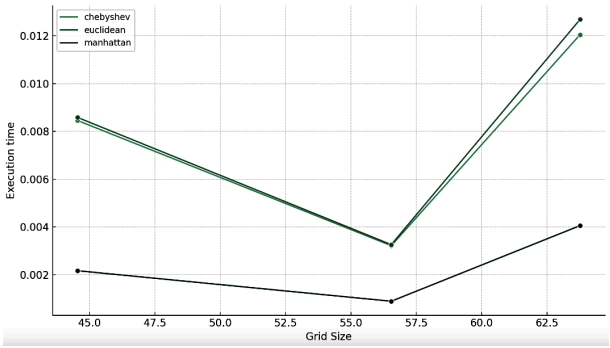

Fig. 4. Execution Time with increasing obstacle probabilities in grid

- **Chebyshev Heuristic**: The execution time exhibits a gradual increase as the grid size grows, indicating that as the environment becomes more complex, the heuristic requires more time to navigate.
- **Euclidean Heuristic**: This heuristic also shows a gradual increase in execution time with grid size. It starts off slightly slower than Chebyshev but appears to maintain a more consistent rate of growth.
- **Manhattan Heuristic**: This heuristic remains the most efficient in terms of execution time across varying grid sizes, with its execution time being considerably lower than the other two heuristics.

Number of Nodes Expanded vs Grid Size

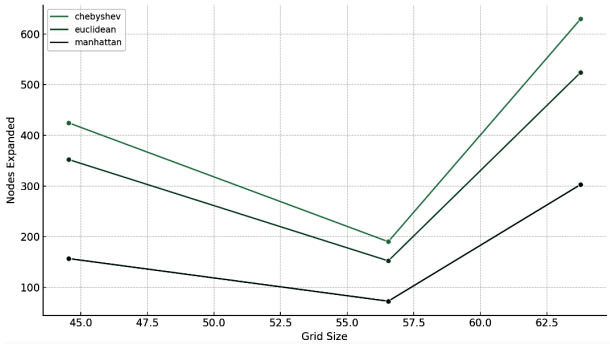

Fig. 5. Number of Nodes Expanded with increasing obstacle probabilities in grid

- **Chebyshev Heuristic**: The number of nodes expanded grows with the grid size, aligning with the observed increase in execution time.
- **Euclidean Heuristic**: Similarly, the number of nodes expanded increases with grid size, but the growth rate seems less steep compared to Chebyshev.
- **Manhattan Heuristic**: This heuristic consistently expands the fewest nodes, further emphasizing its efficiency in the grid environment.

Memory Consumption vs Grid Size

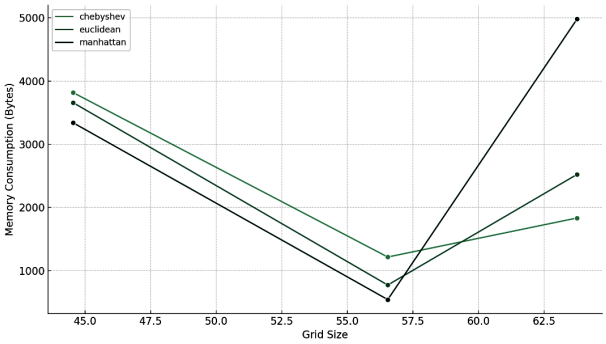

Fig. 6. Memory Consumption with increasing obstacle probabilities in grid

- **Chebyshev Heuristic**: Memory consumption starts off high for smaller grid sizes and shows a declining trend as the grid size increases.
- **Euclidean Heuristic**: Its memory consumption follows a similar trend to Chebyshev, starting off high and reducing with grid size. However, Euclidean's consumption is slightly higher across all grid sizes.
- **Manhattan Heuristic**: Manhattan starts with the highest memory consumption, but its consumption reduces considerably as the grid size grows, making it more memory-efficient for larger grids compared to the other heuristics.

Key Insights:

- The **Manhattan** heuristic continues to stand out in terms of speed and efficiency, as evidenced by its consistently lower execution time and fewer nodes expanded. However, it requires more memory in smaller grid sizes.
- The **Chebyshev** and **Euclidean** heuristics show similar patterns in terms of execution time and nodes expanded, with Chebyshev generally expanding more nodes but running slightly faster than Euclidean.

CONCLUSION

The research aimed to investigate the performance of three heuristics: Chebyshev, Euclidean, and Manhattan, within the A* algorithm across two environments: Grid and Graph. These environments were further varied by size and probability configurations. The results were consistent in shedding light on the efficiency and behavior of each heuristic.

Manhattan consistently emerged as the most time-efficient heuristic across both environments. Its direct approach was evident in its tendency to expand fewer nodes, making it quicker in providing solutions. However, this efficiency was often at the cost of higher memory consumption, particularly in smaller grid and graph sizes.

Both **Chebyshev** and **Euclidean** heuristics showed comparable performances. They generally required slightly more time and expanded more nodes than Manhattan. In terms of memory consumption, while both started high, the trend showed a decrease as the complexity (size) of the environment increased. This indicates that for larger and

more intricate problems, these heuristics could be more memory-friendly.

FUTURE WORK

1. **Diverse Environments**: While this study focused on grid and graph environments, future research could explore more complex environments with various terrains or obstacles. This would add layers of realism and intricacy, making the analysis richer.

2. **Hybrid Heuristics**: Combining the strengths of different heuristics might lead to a more optimized approach. Research could focus on developing hybrid heuristics that maximize efficiency and minimize memory consumption.

3. **Parallel Processing**: Given the computational intensity of pathfinding algorithms, future studies could explore parallel processing or distributed computing methodologies to further expedite the A* algorithm's performance.

4. **Memory Optimization**: Given that memory consumption emerged as a significant metric, especially for the Manhattan heuristic, future work could delve deeper into techniques or algorithms that optimize memory usage during the search process.

5. **Real-world Applications**: Applying the findings to real-world scenarios, like navigation systems, robotics, or gaming, could be a valuable direction. Observing the heuristics' performance in practical applications would provide insights into their feasibility and efficiency in real-world contexts.

REFERENCES

1. Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100–107.
2. Russell, S. J., & Norvig, P. (2009). Artificial Intelligence: A Modern Approach (3rd ed.). Prentice Hall.
3. Dechter, R., & Pearl, J. (1985). Generalized best-first search strategies and the optimality of A*. Journal of the ACM (JACM), 32(3), 505-536.
4. Pohl, I. (1970). Heuristic search viewed as path finding in a graph. Artificial Intelligence, 1(3-4), 193-204.
5. O. Amine and M. Mohammed, "Predicting A* search algorithm heuristics using neural networks," *2021 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, Cape Town, South Africa, 2021, pp. 1-12, doi: 10.1109/ICECET52533.2021.9698700.
6. Dian Rachmawati and Lysander Gustin 2020 J. Phys.: Conf. Ser. 1566 012061
7. B. Cao, Z. Yang, L. Yu and Y. Zhang, "Research on the star algorithm for safe path planning," *2023 IEEE International Conference on Control, Electronics and Computer Technology (ICCECT)*, Jilin, China, 2023, pp. 105-109, doi: 10.1109/ICCECT57938.2023.10141167.
8. Erke S, Bin D, Yiming N, Qi Z, Liang X, Dawei Z. An improved A-Star based path planning algorithm for autonomous land vehicles. *International Journal of Advanced Robotic Systems*. 2020;17(5). doi:10.1177/1729881420962263
9. Liu, Xiang and Daoxiong Gong. “A comparative study of A-star algorithms for search and rescue in perfect maze.” *2011 International Conference on Electric Information and Control Engineering* (2011): 24-27.
10. Ariel Felner, Richard E. Korf, and Sarit Hanan. Additive pattern database heuristics. *Journal of Artificial Intelligence Research*, 22:279-318, 2004..
11. E. R. Firmansyah, S. U. Masruroh and F. Fahrianto, "Comparative Analysis of A* and Basic Theta* Algorithm in Android-Based Pathfinding Games," *2016 6th International Conference on Information and Communication Technology for The Muslim World (ICT4M)*, Jakarta, Indonesia, 2016, pp. 275-280, doi: 10.1109/ICT4M.2016.063.