Evaluating the A* Algorithm Performance in Obstacle Avoidance through Simulation

1st JaeHyeon Sung
School of Electical Engineering
University of Ulsan
Ulsan, Korea
jhn6446@gmail.com

2nd KangHyun Jo
Dept. of Electrical, Electronic and Computer Engineering
University of Ulsan
Ulsan, Korea
acejo@ulsan.ac.kr

***Abstract*—In autonomous driving, path planning is essential and inevitable. Therefore, this study recognizes the need for research on path planning. The research aims to evaluate the performance of obstacle avoidance algorithms generated through the A* algorithm. The evaluation will be conducted by comparing the node cost variations based on the presence or absence of obstacles. It is observed that paths adjacent to obstacles are mainly generated, indicating the necessity of parameters related to obstacles. With sufficient data on paths, promising results are anticipated.**

***Index Terms*—A* algorithm, Path planning**

I. Introduction

The idea of this algorithm is to select the node with the smallest value(F), which is the sum of the distance(G) from the starting point and the remaining distance(H: Heuristic) to the destination, for any node, and search to the destination. The important thing here is to appropriately estimate and set the remaining distance(H) to the destination. And you have to manage nodes with two lists, an open list, and a closed list. An open list is a list adjacent to the node currently under investigation and is a set of nodes whose selection possibilities are open as the shortest distance. A closed list is a set of nodes that have been inspected and do not need to be inspected again.

II. Related work

Each node stores values $G(n)$, $H(n)$, and $F(n)$, along with its own node address and the address of parent node from which it is accessed. Instead of using a regular list for the open list, a priority queue is employed to prioritize nodes with the lowest F value, representing the most promising path. The closed list is maintained as a regular queue.

The overall flow begins by placing the starting node into the open list. The following steps are then repeated:

1) Extract a node from the open list and examine all reachable nodes.
2) If the examined node is in the closed list, ignore it.
   a) If the node is in the open list, update its information with the lower F value.
   b) If the node is neither in the open nor closed list, insert it into the open list.
3) Move the node extracted in step 1 to the closed list.

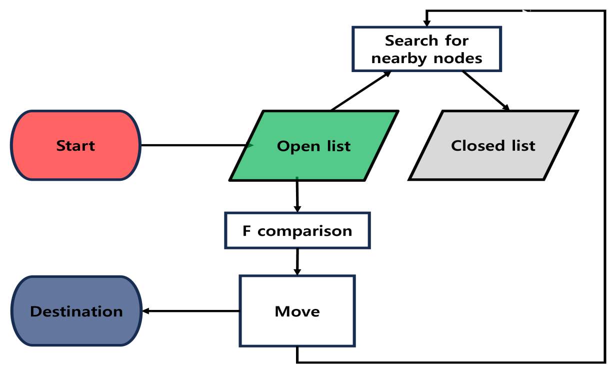


Fig. 1: A* algorithm process

4) If the target node enters the open list, stop the search. Otherwise, continue exploring from the target node to the starting node, tracing the path by finding parent nodes. If the open list becomes empty before the target node is reached, it indicates that no path exists.

Path scoring is achieved using the following Eq. (1). $G(n)$ is the cost of moving from the start point A to the current square, using the path generated so far. $H(n)$ is the estimated cost of moving from the current square to the destination $B(n)$. This estimate considers only horizontal and vertical movements, ignoring obstacles and diagonal motions. $F(n)$ represents the total cost, combining the actual cost $G(n)$ and the estimated cost $H(n)$ of reaching the current square. $G(n)$ represents the cost of moving to a specific square from the start point, using the path created so far. H can be estimated using various methods. Here, the Manhattan method is used, which considers only horizontal and vertical movements and disregards obstacles to calculate the estimated distance from the current square to the target square. $F(n)$ is calculated by summing up $G(n)$ and $H(n)$. To continue the search, the node with the lowest F value in the "open list" is selected.

III. Proposed work

This study aims to investigate how the A* algorithm estimates the optimal path based on the given equation when there are and aren't obstacles. The values assigned to each node according to the mentioned equation will be compared between the obstacle-avoiding path and the path without obstacles to evaluate avoidance performance.

A. Equations

For performance evaluation, this work utilize the following formula:

$$F(n) = G(n) + H(n) \tag{1}$$

$$H = |x_1 - x_2| \ + |y_1 - y_2| \tag{2}$$

The $F(n)$ value for each node is calculated using the following formula Eq. (1). The $H(n)$ value is determined using the Manhattan distance method Eq. (2), which calculates the distance without considering obstacles. The $G(n)$ value is calculated as the distance from the starting point to the current node. By comparing the values of each node, the shortest path is identified. As a simple example, consider the following diagram.

IV. Experiment

As shown in Fig. (2), the A* algorithm calculates the variables for each node and performs comparisons with adjacent nodes. In the case of the right node of the blue node, the smallest value of F is indicated through the comparison.

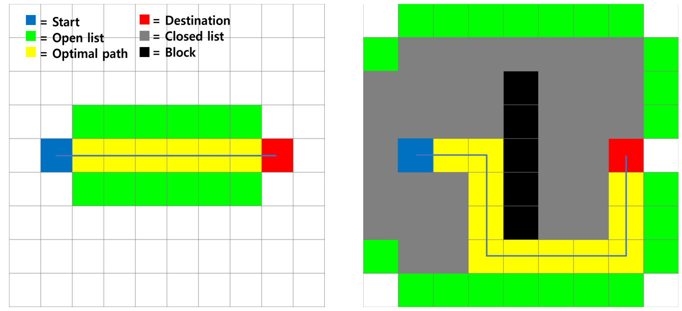
A* algorithm evaluates the performance by comparing the cost of each variable (G, H, F) based on the presence or absence of obstacles. Here, the distance between two adjacent nodes is set to 10, and the distance between two nodes on a diagonal is set to 14 for integer operations. The experiment is conducted to observe the F cost of each node and understand the impact of obstacles on path generation.

H : 60 G : 14 F : 74	H : 50 G : 10 F : 60	H : 40 G : 14 F : 54			
H : 50 G :10 F : 60	H : ∞ G : ∞ F : ∞	H : 30 G : 10 F : 40			Destination
H : 60 G : 14 F : 74	H : 50 G : 10 F : 60	H : 40 G : 14 F : 54			

Fig. 2: A* Algorithm Calculation Process

V. Result

In Fig. (3) result, it is evident that the path is generated in the direction of the smallest $F(n)$ value among the neighboring nodes. The algorithm considers only the starting point and the destination, enabling continuous forward movement until an obstacle is encountered between them. It is observed that the path is generated adjacent to the obstacles.



(a) Non obstacle (b) Exist obstacle

Fig. 3: Results with and without obstacles

This is a result of considering only the positions of the starting point and the destination until encountering an obstacle. Since the path is generated based on the $F(n)$ value, it appears necessary to introduce parameters related to obstacles.

VI. Conclusion

Through this study, it was observed that the path is generated in the direction of the smallest $F(n)$ value among the neighboring nodes. While the path generally follows a decreasing trend in $F(n)$, encountering obstacles leads to an increase in $F(n)$. However, the path selection continues to favor the direction with the smallest value among the surrounding nodes, even if it is higher than the current $F(n)$ value. As a result, when obstacles are positioned between the starting point and the destination, the path may be generated adjacent to the obstacles. This phenomenon cannot be considered beneficial for obstacle avoidance, prompting the need to explore alternative solutions to address this issue in future planning.

Based on this study, we will conduct research in a direction where path planning for obstacle avoidance is effectively utilized by increasing the $F(n)$ value when encountering obstacles. As the $F(n)$ value increases, the path generation will avoid areas near obstacles, which is expected to contribute to more stable path planning.

Acknowledgment

This result was supported by "Regional Innovation Strategy (RIS)" through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(MOE)(2021RIS-003)

References

[1] P. E. Hart, N. J. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," in IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100-107, July 1968.

[2] C. Ju, Q. Luo and X. Yan, "Path Planning Using an Improved A-star Algorithm," 2020 11th International Conference on Prognostics and System Health Management (PHM-2020 Jinan), Jinan, China, 2020, pp. 23-26.