Implementation of Agricultural Path Planning with Unmanned Ground Vehicles (UGV) based on Enhanced A* Algorithm

2023 12th International Conference on Modern Circuits and Systems Technologies (MOCAST) | 979-8-3503-2107-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/MOCAST57943.2023.10176428

Antonios Chatzisavvas
Electrical & Computer Engineering
University of Western Macedonia
Kozani, Greece
achatzisavvas@uowm.gr

Malamati Louta
Electrical & Computer Engineering
University of Western Macedonia
Kozani, Greece
louta@uowm.gr

Minas Dasygenis
Electrical & Computer Engineering
University of Western Macedonia
Kozani, Greece
mdasyg@ieee.org

***Abstract*—The A* algorithm is well-known for its use in numerous applications, including robots and GPS systems, for the purpose of route planning. The algorithm, despite its usefulness, has several restrictions regarding its operational efficiency and the length of its paths. This article makes a suggestion for improving the standard A* algorithm in order to overcome these constraints. Compared to the A* algorithm's standard performance, the findings showed that the improved algorithm reduce the amount of time needed for route planning by 9.11% on average, while also cutting the path length by 9.29% on average. The A* algorithm's performance can be significantly improved with the help of the method that we propose, both in terms of its operational efficiency and the length of its paths.**

Keywords — UGV, agricultural path planning, A* algorithm, enhanced A* algorithm, unmanned ground vehicles

I. INTRODUCTION

The employment of robots has increased dramatically over the last several decades, especially in circumstances that are thought of as commonplace. Industrial mobile robots that are have become more popular more quicker than those that are fixed or traditional. Autonomous navigation is a critical capability found in unmanned ground vehicles (UGV). Developing autonomous navigation that is trustworthy, safe, and successful is one of the most popular yet, challenging research subjects, in the field of robotics. The four main difficulties that emerge during navigation are localization, perception, route planning and motion control [1]. One may argue that, out of all these factors, route planning is the one that is most important for the navigational processes. A common technique in route planning research is symmetry.

The A* approach, which is based on graph search, is used for route planning. The present node serves as the center of a symmetrical scan of the surrounding nodes throughout the search process. Bidirectional search is a search methodology in addition to being a symmetric route search tool. If this symmetric search strategy is used, the algorithm's efficiency might be improved by a factor of two. Path planning refers to the act of selecting a route across a particular area that is devoid of potential hazards. Given how chaotic real life is most of the time, this process may be challenging [2]. Whether or not an autonomous mobile robot can perform reliable and effective navigation, depends on the path that has been planned. As a result, route planning is a crucial component of mobile robot navigation. Route planning research is now receiving more attention than ever before as a direct consequence of the rise of mobile robot applications.

Depending on how they are utilized, route planning algorithms may be classified into a wide range of groups. Diagram search-based schedulers contain, among others, the Dijkstra algorithm [3], the A-star (A*) technique, and the state lattice algorithm [4]. One kind of research that uses sampling-based research is the Random Rapid Exploration Trees (RRT) [5]. Other forms of path planning include artificial intelligence-based path planning, path planning based on vision, and other technologies.

By taking into consideration the knowledge of the obstacles that are present in the static environment, the A* algorithm is employed to calculate the shortest path feasible. The selection of viable node pairs and subsequent assessment of the shortest route based on the acquired feasible node pairs make up the two layers of the issue of determining the shortest path in a given static environment [6]. Only by utilising the knowledge that derived from the preceding layer this issue can be addressed. The approach cannot be utilized in dynamic settings since it is inefficient and cannot be employed in such circumstances because a dynamic environment does not meet any of the aforementioned requirements. We chose the A* algorithm since it is one of the main algorithms used in modern real-time route planning applications for static settings. Recent work focuses on the algorithm and seeks to discover further advances in efficiency and performance.

The effectiveness and resilience of route planning algorithms are their two most crucial features. Resilience is crucial for mobile robots since it signifies the algorithm's reliability and is a need for secure, stable, and quick movement. One of the key requirements for good assignment performance in mobile robots is the algorithm's efficacy for path planning and route search. Efficiency takes into account the algorithm's speed. This study examines possible improvements to the algorithm's effectiveness and strategies to make it more resilient. In order to reduce the number of orthogonal turns, it is necessary to find solutions to the research challenges of minimizing the algorithm's execution time, the amount of search nodes, as well as the avoidance of collisions.

The following is a list of the major contributions that this paper makes:

- This article presents a suggestion for a new and enhanced version of the A* algorithm by including the stages and function that optimize the first version of the A* technique. The smoothing and the extension distance are both

used in order to achieve the goal of improving the overall resilience of the route.

- We use a method known as two-way search, as well as a procedure for heuristic optimization and expanding the distance, in order to make the algorithm more efficient. This allows us to find solutions more quickly. In order to put the enhanced A* algorithm through its paces, it is evaluated in conjunction with a number of different algorithms.
- According to the findings, the revised version of the A* algorithm has significantly enhanced efficiencies regarding its performance and robustness.

This work's remaining sections are structured as follows: Section II presents similar study articles. In Section III, we describe our enhanced A* algorithm. In Section IV, we explain the experimental effects. Finally, in Section V, we summarise our study's impacts.

II. Relative Work

A geometric model of a given complexity must be processed by the algorithm in order to construct a route for a robotic system. Google Maps and different traffic routing approaches often employ the Dijkstra algorithm and its variations. Modern methods for usage in static situations are provided as A* and its derivatives [7].

It is feasible to utilize the A* technique to draw the shortest route on a map; however, in order to do so, the method has to iterate over each node of the route and choose the alternative that will result in the lowest possible route cost. Due to this, the approach performs a substantial number of computing and takes a significant amount of time completing calculations. Moreover, the algorithm's effectiveness decreased as the magnitude of the map increased [8], which is a consequence of the aforementioned.

The A* algorithm has been the subject of extensive study, with a primary emphasis on obstacle avoidance, a variety of application situations, and improving the system's performance as it develops. A common route planning approach involving graph traversal is the A* algorithm. A* uses a heuristic function to direct its search in the direction of the states with the shortest paths. The A* algorithm has been extensively used in a variety of industries, including the transportation sector [9]. Since then, developments in artificial intelligence have allowed for improvements to be made to the A* algorithm, one of which is the planning of routes for automated guided vehicles (AGVs) or unmanned surface vehicles (USVs), as well as robot route planning. As the A* approach is more straightforward and employs fewer search nodes than certain other route planning algorithms, it is better suited to more constrained situations [10].

Before moving on with its search, the A* algorithm has to get some guidance from a heuristic function so that it can choose which way to travel in order to discover the shortest path on a map. The most essential components of the heuristic procedure are the Manhattan distance, the Euclidean distance, the Chebyshev distance, and the diagonal distance [11]. In order to finish the work, a heuristic function could be used to guide the execution of an algorithm all the way through the calculation or execution time.

A solution that recommends employing an A* optimization strategy that has been enhanced in two different ways [12] is presented as an answer to the issues of slow explore rate and poor algorithm performance. To begin, weights are assigned to the evaluation procedure to confirm that the heuristic function can be dependent upon. This was done to ensure that the function could be trusted. After that, the maps are brought up to date by adding a cluster of nodes that are organized around a certain focal point on the map. If the collection of nodes contains nodes that represent difficulties, then this node is regarded as an untrustworthy point, and as a result, it will not be sought. As a consequence of these modifications to the procedure, the A* algorithm may become a more effective computational tool. Nevertheless, the optimization strategy is only one of many possible tactics, and using it results in an increase in the amount of computation that the program is required to carry out. In addition to that, the optimization strategy being discussed here does not take a thorough approach.

Some researchers focused their attention on storing traversed nodes and suggested an enhanced A* storage array strategy [13]. This action was taken in order to achieve the aim of improving the performance of the A* algorithm. When accessing a particular element of the array, the storage mechanism requires scanning the sequence number in order to locate that element. This is done in order to locate the element. In contrast to the traditional A* approach, which requires traversing through numerous nodes in order to find the needed element, this may be completed with only a single operation by using a more efficient algorithm. This method does not result in an improvement to the algorithm's route design; all it does is optimize the storage and access of nodes. Moreover, it does not result in a decrease in the amount of processing that is needed by the algorithm.

In order to reduce the amount of time required for the A* algorithm to do calculations, an improved strategy has been proposed [11] and given the name A* time-efficient algorithm. In the approach that is the subject of this discussion, finding the importance of the heuristic procedure does not take place during the phase devoted to initialization; rather, this step takes place before the collision phase. The fact that this is the case contributes to the method's increased efficacy in terms of the amount of time it requires to be computed. This method of optimization does neither reduce the number of search nodes nor does it optimize the heuristic procedure; instead, it maximizes the potential of finding the value of the heuristic function. As a result, there is not a lot of space for improvement in terms of how efficiently things are done.

In contrast to similar efforts by other authors, we present an improved version of the A* algorithm in terms of average path length. In our work, we experimentally validate the results of running and using the enhanced A* algorithm which is shown to improve the path quality and time.

III. Implementation

The tried-and-true A* algorithm is the one to use for designing routes in unchanging environments.

$$f(a)=k(a)+p(a) \quad (1)$$

The above cost operation from the present point where the robot is to the point where the robot is to move is represented by k(a). p(a) is the heuristic procedure that is executed from the present node to the end where the robot is to move. The robot's position points are obtained through the sensors, and the path cost is extracted from the heuristic function, namely:

$$k(a) = \sqrt{(n\_y - r\_y)^2 + (n\_x - r\_x)^2} \quad (2)$$

$$p(a) = \sqrt{(n\_x - w\_y)^2 + (n\_x - w\_x)^2} \quad (3)$$

In the Equalizations (2),(3), $((n\_y) - (\mathrm{n}\_x))$ are the abscissa and ordinate of the current point $((r\_y) - (\mathrm{r}\_x))$ are the abscissa and ordinate of the starting point $((w\_y) - (\mathrm{w}\_x))$ are the abscissa and ordinate of the target point.

A. Evaluation function design

When the UGV constructs a non-linear path, it requires a certain amount of time to do the rotation operation. Hence, if you want to reduce the time it takes to operate the robot, decreasing the amount of time spent spinning is essential. According to the first eight nodes around the present node, the standard procedure known as A* calculates the cost value of the matching node. To reduce the time it takes the UGV to perform the turn we need to implement, the evaluation procedure described in this research involves a floating function. Calculating the line between the two ends of the evaluation gives the value of the function.

$$f(a)=k(a)+p(a)+t(a) \quad (4)$$

$$t(n) = \begin{cases} 0, & (x_n - x_p)^*(y_n - y_p) = 0 \\ k^*|p(a)|, & \text{other} \end{cases} \quad (5)$$

The parameter for the time factor is $^{k\sim\theta}$ and that is a general definition of the robot's angle of rotation. When the UGV's angle of rotation becomes more pronounced, the function's value rises. The precision of the cost function might be greatly improved by include the time variable, which would boost the efficacy of route planning [13].

B. Cost and weight improvement estimates

$$f(n) = k(a) + \left(\frac{r}{R} + 1\right) p(a) + t(a) \quad (6)$$

The distance from the point where the UGV is to the point designated as the target is denoted by k(a), p(a). The distance that the UGV starts moving to the first target point is denoted by p(a). The following is a list of the precise stages that the modified A* algorithm takes in order to find the shortest route even when there are obstacles in the way:

1) Initialize the start point as well as the endpoint. Also, the creation of a list of obstacles in the environment. Starting point to the target point.

2) Creation list of open and closed routes.

3) If the list of open points is empty, it means that the route was not found and will have to be recalculated.

4) Having defined the beginning location, search for nodes in the area around the starting point.

5) The node with the lowest value is found by referring to the result of the evaluation function.

6) The node is selected according to the closed list.

7) Combination nodes of the path are finally acquired and placed in the closed list.

C. Threshold setting

The distance from the moving route to the top of the static barrier is known as the “safety threshold”. Assuming that the current node is connected, the distance between the current node and the top of the static barrier along the line segment has to be more than or comparable to the protection threshold. If the segment that connects the previous node and the current node to the top of the static block has a length that is less than the safety threshold when it reaches the static obstacle, the node in question is removed from the route, and the next node in the route is picked. The process will continue until the safety threshold is achieved, which is when there is a larger space between the top of the static barrier and the segment line between the prior node and the present node. Once this point has been reached, the operation will come to an end.

Alterations are made to the path, beginning at the previous node, and the processes outlined above are repeated until the distance separating the starting and ending points corresponds with the criteria specified before. Since the present node serves no use, it will be removed shortly. The procedure is repeated up to the point when the distance between the starting place and the destination point satisfies the requirements that were presented before.

IV. Experimental findings

In order to improve the evaluation function, this research makes use of the algorithm A*, which allows it to acquire the best possible results for route planning in a number of different settings. The researchers that wrote up this study came up with the algorithm themselves. The simulation tests were developed in such a manner that both the traditional version of the algorithm A* as well as the updated version of the algorithm A* were used. This was done so that an evaluation of the superior algorithm's performance could be carried out.

We made use of a VELOS unmanned ground vehicle (UGV) so that we could achieve the objectives of our experimental studies. It is a mobile robot that was designed to collect photos while traveling autonomous across agricultural fields in order to diagnose diseases. It was discovered that barriers were rough locations that the height of the robot would make it difficult for it to approach. To confirm the start and end point, the simple algorithm A* is used to derive the three path planning experiments, where the start and end point of each experiment are the same.

Both algorithm was executed in an Intel NUC i5 16gb ram. Implementation was done in Python programming language. To compare the performance of algorithms, we use metrics such as execution time and path length. Execution time measures the time it takes for the algorithm to generate a path from the starting location to the goal location. Additionally, a shorter path length is preferred as it minimizes the distance the

robot needs to travel and reduces the risk of collisions with obstacles. The experiment findings are presented in Table I.

TABLE I. EXPERIMENTAL FINDINGS OF STANDARD A* ALGORITHM

Path planning experiment	Path Length (m)	Time (s)
1	14.41	142.78
2	13.68	142.42
3	12.17	142.16

In the same way as the standard A* algorithm experiment, the enhanced A* algorithm is used in the same way and run with the same conditions in order to have a comparison when conducting the three experiments. In this experiment, the same variables that were controlled in the prior one are used. Because of this, the soundness of comparing the data and findings shown in Table II is ensured, as well as the stability of the conclusions themselves.

TABLE II. EXPERIMENTAL FINDINGS OF ENHANCED A* ALGORITHM

Path planning experiment	Path Length (m)	Time (s)
1	13.74	133.16
2	12.46	129.31
3	11.24	127.24

Table III compares the results of route planning systems using the standard A* and the enhanced A* method. Table III illustrates that the enhanced A* algorithm suggested in this study has a route length that is 9.29% shorter on average than the standard A* technique and that the required amount of time has been reduced by 9.11% on average. Figures 1-2 show the average path length and the average time for the three path-planning experiments with the standard A* and enhanced A* algorithm.

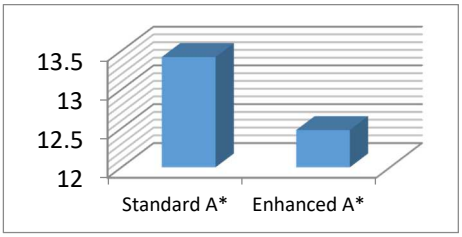

Figure 1.Comparison of the average path length for the three path planning experiments

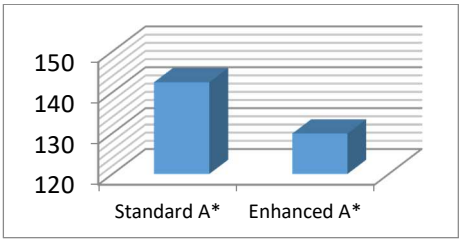

Figure 2. Comparison of the average time for the three path planning experiments

TABLE III. COMPARISON OF EXPERIMENTAL OUTCOMES OF STANDARD A* ALGORITHM AND ENHANCED A* ALGORITHM

Algorithm	Average Path Length (m) for three path-planning experiments	Average Time (s) for three path-planning experiments
Standard A*	13.42	142.45
Enhanced A*	12.48	129.90

V. CONCLUSIONS

The A* algorithm is famous for path planning in many applications, including robotics and GPS systems. Despite its effectiveness, the algorithm has some operating efficiency and path length limitations. This article proposes enhancing the standard A* algorithm to address these limitations. The results showed that the enhanced algorithm reduced the time required for path planning by 9.11% on average and the path length by 9.29% on average, compared to the standard A* algorithm. The A* method's performance can be significantly improved with the help of the algorithm that has been suggested, both in terms of how efficiently it operates and how long its paths are.

ACKNOWLEDGMENT

This research has been co-financed by the EuropeanRegional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation (project code MIS 5047196).

REFERENCES

[1] Mohd. N. Zafar and J. C. Mohanta, "Methodology for Path Planning and Optimization of Mobile Robots: A Review," Procedia Computer Science, vol. 133. Elsevier BV, pp. 141–152, 2018.

[2] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," Robotics and Autonomous Systems, vol. 86. Elsevier BV, pp. 13–28, Dec. 2016.

[3] E. W. Dijkstra, "A note on two problems in connexion with graphs," Numerische Mathematik, vol. 1, no. 1. Springer Science and Business Media LLC, pp. 269–271, Dec. 1959.

[4] C. Zhang, D. Chu, S. Liu, Z. Deng, C. Wu, and X. Su, "Trajectory Planning and Tracking for Autonomous Vehicle Based on State Lattice and Model Predictive Control," IEEE Intelligent Transportation Systems Magazine, vol. 11, no. 2. Institute of Electrical and Electronics Engineers (IEEE), pp. 29–40, 2019.

[5] L. Zhang, Z. Lin, J. Wang, and B. He, "Rapidly-exploring Random Trees multi-robot map exploration under optimization framework," Robotics and Autonomous Systems, vol. 131. Elsevier BV, p. 103565, Sep. 2020.

[6] T. Dudi, R. Singhal, and R. Kumar, "Shortest Path Evaluation with Enhanced Linear Graph and Dijkstra Algorithm," 2020 59th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE). IEEE, Sep. 23, 2020.

[7] K. Karur, N. Sharma, C. Dharmatti, and J. E. Siegel, "A Survey of Path Planning Algorithms for Mobile Robots," Vehicles, vol. 3, no. 3. MDPI AG, pp. 448–468, Aug. 04, 2021.

[8] Ziqiang, W.; Xiaoguang, H.; Xiaoxiao, L. Overview of Global Path Planning Algorithms for Mobile Robots. Comput. Sci. 2021, 48,1–16.

[9] C. Liu, Q. Mao, X. Chu, and S. Xie, "An Improved A-Star Algorithm Considering Water Current, Traffic Separation and Berthing for Vessel Path Planning," Applied Sciences, vol. 9, no. 6. MDPI AG, p. 1057, Mar. 13, 2019.

[10] Y. Singh, S. Sharma, R. Sutton, D. Hatton, and A. Khan, "A constrained A* approach towards optimal path planning for an unmanned surface vehicle in a maritime environment containing dynamic obstacles and ocean currents," Ocean Engineering, vol. 169. Elsevier BV, pp. 187–201, Dec. 2018.

[11] J. Yao, C. Lin, X. Xie, A. J. Wang, and C.-C. Hung, "Path Planning for Virtual Human Motion Using Improved A* Star Algorithm," 2010 Seventh International Conference on Information Technology: New Generations. IEEE, 2010.

[12] Gao, Q.J.; Yu, Y.S.; Hu, D.D. Feasible Path Search and Optimization Based on Improved A* Algorithm. J. Civ. Aviat. Univ. China2005, 23, 42–45.

[13] J. Peng, Y. Huang, and G. Luo, "Robot Path Planning Based on Improved A* Algorithm," Cybernetics and Information Technologies, vol. 15, no. 2. Walter de Gruyter GmbH, pp. 171–180, Jun. 01, 2015.