ISSN 2959-6157

H2A algorithm and moving obstacle avoidance based on A* Algorithm

Fulin Ma[1], Binyu Yan[2], Hanyan Li[3]

[1]College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, 201403, China, 032150227@nuaa.edu.cn

[2]Pleasant Hill, Diablo Valley College, 94523, USA, yanbinyu132456@foxmail.com

[3]School of Logistics Engineering, Wuhan University of Technology, 430063, China, 1485290487@qq.com

Binyu Yan, Fulin Ma, Hanyan Li contributed equally to this work and should be considered co-first authors.

*Corresponding author email: 032150227@nuaa.edu.cn

Abstract:

The A* algorithm is widely used in the field of path planning for autonomous mobile robots. However, A* is a simple algorithm that cannot avoid moving obstacles. One of our research directions is to realize dynamic obstacle avoidance, which is based on A* algorithm. At the same time, we found that the A* algorithm is computationally inefficient. In contrast, the emerging nature-inspired algorithm outperforms the classical algorithm because of the reduced computational overhead. The nature-inspired algorithm is one of the most common heuristic algorithms. We reproduce a near-optimal algorithm that considers a single UGV approximate optimal path planning algorithm for obstacle avoidance in static environments. The algorithm uses two heuristic values, hence the name H2A (double heuristic algorithm). Our goal is to try to be able to avoid two moving obstacles at a time, the performance of the proposed algorithm is compared with the performance of the A* algorithm.

Keywords: Artificial Potential Fields, Dynamic Obstacle Avoidance, Autonomous Driving

1 Introduction

Path planning is a fundamental problem in various fields such as robotics, artificial intelligence, and video game development. One of the most popular and widely used algorithms for path planning is the A* (A-star) algorithm. Introduced by Peter Hart, Nils Nilsson, and Bertram Raphael in 1968, A* combines the strengths of Dijkstra's algorithm and Greedy Best-First-Search, making it both complete and optimal under certain conditions. The algorithm utilizes a best-first search approach and employs a heuristic to estimate the cost of the cheapest path from the start node to the goal node, significantly improving efficiency compared to exhaustive search methods.

On the other hand, we improve the underlying A* algorithm to H2A. And we have enhanced and optimized H2A in order to reduce running costs as much as possible alongside its real-world usability so that it can be integrated universally; we shall compare with A* for 16×16; 64×64 grid (2D maps) which is small size environment, the sort of comparison will show us which algorithm is better.

2 Literature Review

Obstacle avoidance path planning algorithms are divided into local dynamic path planning algorithms and global static path planning algorithms. Local dynamic path planning algorithms refer to situations where the device does not know the surrounding environment information during movement. The device may encounter changing environmental information while in motion and can only rely on sensors carried by the robot to perceive the surrounding environment. Based on the real-time situation of the surrounding environment, a dynamic model is established, and the CPU is used to analyze and calculate the optimal path to avoid obstacles. Global static path planning algorithms refer to situations where the device knows the surrounding environment before movement. By modeling the environment within the flight range and using some constraints and standards, an optimal path from the starting point to the target point that bypass obstacle is planned. Among global static path planning algorithms, the A* algorithm is currently one of the most widely used path planning algorithms.[1]

However, the traditional A* algorithm is primarily designed for static environments where obstacles are fixed and known in advance. In many real-world applications, such as autonomous vehicle navigation and dynamic game AI, the environment is often unpredictable and obstacles can move unpredictably. These dynamic conditions pres-

ent significant challenges for path planning algorithms, requiring them to adapt to changing environments in real-time to avoid collisions and ensure optimal path planning.

This paper addresses the limitations of the traditional A* algorithm in dynamic environments by proposing a modified version capable of handling moving obstacles. Our algorithm is able to maintain efficient and collision-free navigation. Through this enhancement, we aim to extend the applicability of the A* algorithm to a broader range of real-world scenarios where environmental changes is frequent and unpredictable.

3 Methodology

The traditional A* algorithm, based on cost functions, is one of the most effective heuristic search methods for global path planning in electronic maps. It combines the principles of greedy algorithms and Dijkstra's algorithm. The basic expression for the A* algorithm is:[2]

$$f(n) = g(n) + h(n)$$

f(n) represents the total cost at the current node,

g(n) is the actual cost from the start node to the current node,

h(n) is the estimated cost from the current node to the goal node.

The choice of the heuristic function h(n) greatly impacts the search efficiency of the A* algorithm. Common choices for the heuristic function h(n) include Manhattan distance, Euclidean distance, and Chebyshev distance. Among these, Euclidean distance involves the most computation but yields the highest quality paths. Therefore, this paper uses Euclidean distance to calculate the estimated cost h(n). On the basis of the A* algorithm, we add a time dimension to realize the addition of moving obstacles. We put the time dimension on the z-axis, and realize the change of obstacles at different moments by outputting map grid at different times. Then, at each moment, the algorithm will identify the current obstacle and avoid it, which will be reflected in our results section. We also add potential energy field in our algorithm to make it prioritize avoiding obstacles.[3]

H2A is a fast path-planning algorithm for Unmanned Ground Vehicles (UGVs) in 2D grid-worlds. The UGV itself is assumed to be associated with one grid cell and can move in any of the eight directional motions on a planar surface. This special movement lets the UGV to more accurately search space around it. The core advantage of the H2A algorithm is that it can simultaneously consider and optimize two important performance metrics: first, the path length, which is the total distance that the UGV needs to travel to reach the end point from the starting point; and second, the distance to obstacles, which is related to the safety and obstacle avoidance ability of the UGV during the navigation process. Through such dual-objective optimization, the H2A algorithm is able to calculate the shortest travel path while ensuring the safety of the UGV. In addition, and with the objective of increasing even more its algorithm execution efficiency as a learning strategy in each iteration the H2A instead to generate only one possible solution for cells it generates two cell solutions. This approach dramatically decreases the overall running time of the algorithm, since with each iteration (for n-fold cross-validation) two steps are made in direction to optimal path for optimization instead just one. The algorithm can realize near-optimal path planning solution, so that the navigation efficiency of UGV and speed of task execution are improved.

We build this model in MATLAB to simulate it, which will be illustrated in detail in the next section.

4 Results

In this chapter, we conducted simulation experiments using MATLAB to investigate the obstacle avoidance algorithm. In this scenario, we set up a square map with a side length of 40. On this map, we randomly generate some fixed obstacles that do not move over time. Meanwhile, we also randomly generate a certain number of moving obstacles, which will move randomly around. As is shown in figure 1, the position of moving obstacles changes in different time. To illustrate the changes clearly, the figures shows the position in TIME 1 and TIME 4. However, they will not cover the existing boundaries, the initial point, or the target point. The number of obstacles can be set in the programme. In the simulation, the graph will display the positions of the obstacles and the robot at the current moment. Based on the positions of the obstacles in the next moment, the car will choose its direction of movement. Using the A* algorithm, it will record the selectable positions. When the robot encounters a dead end, it will reselect the optimal solution from the recorded positions and continue exploring until it finds the target or no selectable positions are left.

Considering the actual collision impact [4], you can clearly see in the figure 2 that the robot intentionally moves around the obstacles rather than sticking close to them. This is because we have added the artificial potential field method [5], which makes the car more inclined to choose paths away from obstacles when calculating the cost function. However, we do not want the robot to choose a much longer path just to stay away from obstacles. Therefore, the penalty for the robot sticking close to obstacles will only account for a small portion of the car's path selection

criteria.

In Figure 3, we compare the computation time required for A* algorithm and H2A algorithm under the same map. We randomly generated different maps and calculated the average time required for different maps to more accurately describe the efficiency of H2A algorithm. It can be clearly seen that H2A algorithm use less time to obtain the result. In Figure 4, as the scale of the map gets larger and larger, obstacles are more and more, the superiority of H2A algorithm becomes more obvious.

TIME 1 TIME 4

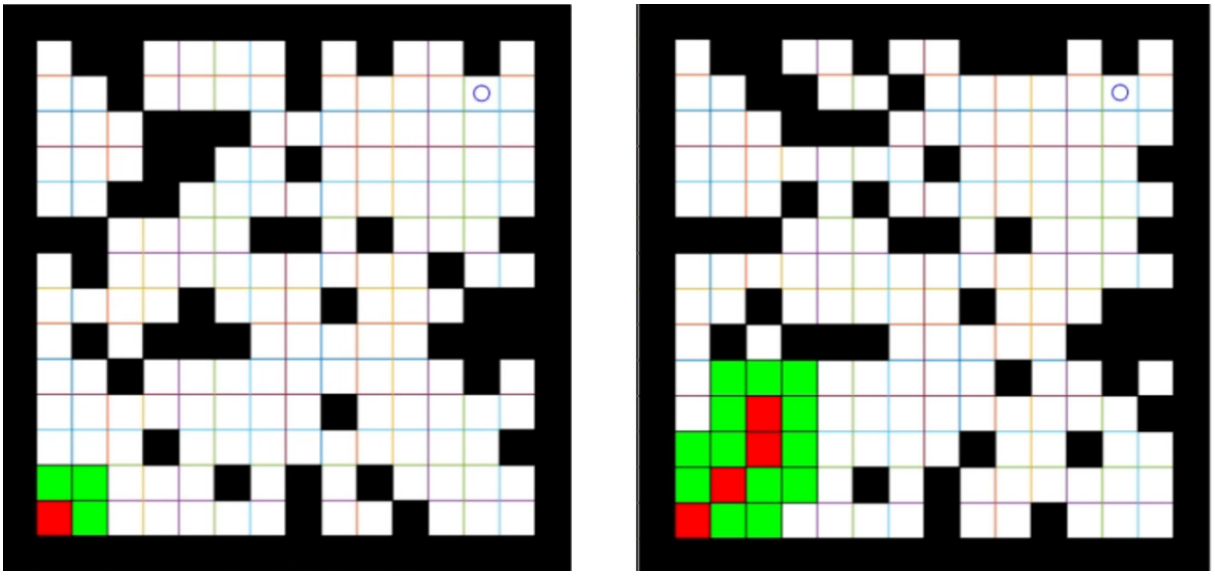


Figure 1 position of moving obstacles in different time

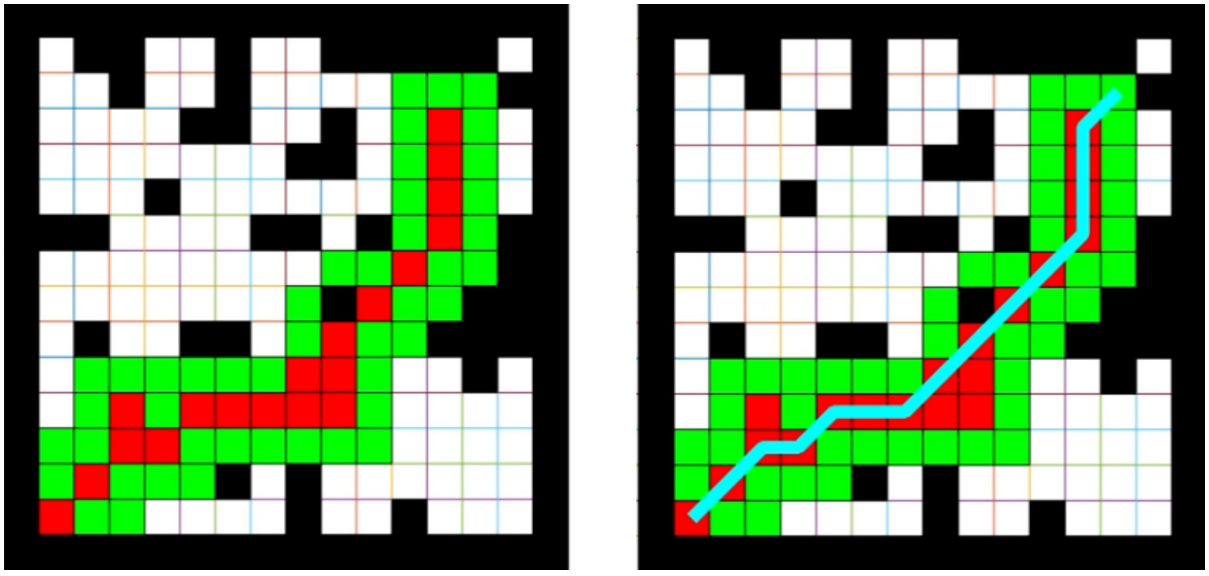


Figure 2 final trace of the robot

Map size 16×16

Add 100 random obstacles to the map

Get the running time

	1	2	3	4	5	6	7	8	9	10	Average
A*	0.3262 s	0.3216 s	0.3351 s	0.3175 s	0.3281 s	0.3096 s	0. 3063 s	0.3112 s	0.3114 s	0.3210 s	0.3188 s
H2A	0.3466 s	0.3608 s	0.3676 s	0.2990 s	0.2987 s	0.3015 s	0.3002 s	0.2954 s	0.3035 s	0.2967 s	0.3170 s

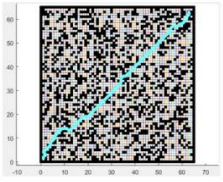
Figure 3 Comparison between Basic A* and H2A in different map scales

Map size 64×64

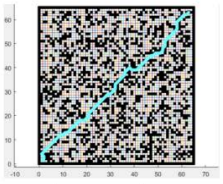
Add 1500 random obstacles to the map

Get the running time

e.g of A*:



e.g of H2A:



	1	2	3	4	5	6	7	8	9	10	Average
A*	2.5544 s	2.5934 s	2.5048 s	2.6261 s	2.6842 s	2.6237 s	2.5779 s	2.5581 s	2.4968 s	2.6092 s	2.5829 s
H2A	2.2075 s	2.2762 s	2.2915 s	2.3371 s	2.2416 s	2.3694 s	2.2515 s	2.2995 s	2.2772 s	2.2535 s	2.2805 s

Figure 4 Comparison between Basic A* and H2A in one large map scale

5 Discussion

The experimental results of our study demonstrate significant improvements in path planning performance when using the H2A algorithm compared to the traditional A* algorithm. Specifically, the H2A algorithm consistently achieved shorter running times and more efficient path-finding in environments with static obstacles. The integration of dual heuristics and dynamic obstacle avoidance allowed H2A to navigate complex scenarios with greater agility and precision. Our simulations showed that H2A could effectively predict and circumvent the paths of moving obstacles, reducing collision rates and enhancing overall navigation efficiency.

The traditional A* algorithm is primarily effective in static environments where obstacles are fixed and pre-known. However, it is unable to make any predictions in dynamic environments. The improved A* algorithm enables the algorithm to achieve obstacle avoidance and path planning in the presence of dynamic obstacles. Additionally, with the incorporation of the artificial potential field method, the vehicle can choose the shortest possible path while reducing the likelihood of collisions with obstacles.

6 Conclusion

In this study, we have presented a modified A* algorithm capable of effectively navigating environments with moving obstacles. By integrating dynamic replanning, predictive modeling of obstacle movements, and maintaining safety margins, our approach enhances the traditional A* algorithm's ability to adapt to real-time changes in dynamic environments. The experimental results demonstrate that our modified A* algorithm not only reduces collision rates but also maintains near-optimal pathfinding efficiency. This advancement broadens the applicability of the A* algorithm to a wider range of real-world scenarios, including autonomous navigation and dynamic game AI, where environmental changes are frequent and unpredictable. Future work will focus on refining the predictive model for more complex obstacle behaviors and conducting ex-

tensive tests in real-world applications to further validate the algorithm's robustness and efficiency.

By comparing the H2A and A * algorithms, we can conclude that under the same map range and the same number of obstacles, the running time of H2A is shorter than that of A *, and as the map expands and the number of obstacles increases, the advantage of H2A's running time becomes more and more obvious, significantly lower than that of A *.

Acknowledgement

Binyu Yan, Fulin Ma, Hanyan Li contributed equally to this work and should be considered co-first authors.

References

[1] Gao Jiuzhou, Xu Weifeng. Autonomous obstacle avoidance path planning of UAV based on improved A-star algorithm[J]. Technology and market,2024,31(01):38-43.

[2] Pan Shiying. Research on AUV path planning based on improved A* algorithm [J]. Technology of equipment manufacturing,2022, (11):49-52.

[3] Miu Yinjun, Shi Wei. Research on route planning of unmanned vehicles based on improved A-star algorithm [J]. Computer knowledge and technology, 2024, 20 (03): 4-7. DOI: 10.14004/j.cnki.ckt.2024.0161.

[4] Chen, Shenghao, et al. "Improved A-star Method for Collision Avoidance and Path Smoothing." *2023 IEEE International Conference on Control, Electronics and Computer Technology (ICCECT)*. IEEE, 2023.

[5] Xiaodong, Zhuang, et al. "Optimal path planning in complex environments based on optimization of artificial potential field." *Robot* 25.6 (2003): 531-535.