

A Multi-objective Path Planning Algorithm Based on a A* and Ant Colony

Hualiang Zhang*

College of Information Engineering, Shenyang University of Chemical Technology, Shenyang 110142, China

* Corresponding author: Hualiang Zhang (Email: 18695976587@163.com)

Abstract: To solve the problems of target conflicts and difficulty in dynamic obstacle avoidance in multi-objective path planning, this paper proposes a hybrid optimization scheme combining the A* algorithm and ant colony optimization (ACO). First, the A* algorithm quickly generates an initial feasible path from the start to the end point, and this path enhances the pheromone distribution of ACO to guide optimization. If the optimized path length is reduced by more than 5%, the optimized result is adopted; otherwise, the original A* path is retained. When the A* algorithm has no solution, ACO performs global path planning to ensure effectiveness and optimality. Experimental results show that in complex scenarios with static, dynamic obstacles and multiple target points, the hybrid algorithm achieves efficient, safe, and dynamically responsive multi-objective dynamic path planning, with an obstacle avoidance success rate of 94%. It significantly improves comprehensive performance and robustness, balancing computational efficiency and path quality, and provides reliable theoretical support and practical solutions for robot multi-objective path planning algorithms.

Keywords: Multi-Objective, A* Algorithm, Ant Colony Algorithm, Path Planning.

1. Introduction

With the rise of artificial intelligence technology, intelligent robots have been widely applied in logistics and distribution, drone inspection, autonomous driving, and other fields[1]. In these applications, robots or agents often need to reach multiple target points, making path planning one of the most important and fundamental elements in task planning[2]. When obstacles exist, real-time obstacle avoidance planning algorithms generate obstacle-avoiding paths based on the positions of these intermediate points to ensure safe and stable arrival at the target locations[3]. Therefore, multi-objective path planning has become a core demand. Currently, research on path trajectory algorithms mainly focuses on several key areas, such as adaptability to complex environments and multi-objective optimization.

Existing trajectory planning methods can be roughly divided into three categories: traditional algorithms, intelligent optimization algorithms, and reinforcement learning algorithms. Traditional algorithms include A*, RRT, and others. The A* algorithm may experience a significant increase in computational complexity and memory consumption in high-dimensional spaces, so it is often used in low-dimensional scenarios. Additionally, A* incorporates a heuristic function, enabling fast search speed, which is affected by the heuristic function[4]. The RRT algorithm can adapt to complex high-dimensional environments for exploration, but its probability-based exploration leads to low efficiency and generates paths with many redundant nodes[5]. Traditional algorithms efficiently solve path planning problems through certain rules and logic. A can quickly find the optimal path in the known map, while RRT is good at quickly exploring feasible paths in high-dimensional complex space, both of which have the characteristics of reliability and efficiency. Traditional trajectory planning methods rely on accurate mathematical models, but the nonlinearity, time-variability, and uncertainty of robot systems make it difficult to meet practical needs. A single objective can be well

optimized using existing trajectory planning methods, but multi-objective optimization in a single scenario often causes degradation of other objectives and conflicts among multiple objectives[6].

Due to conflicts among multiple objectives, optimization is necessary, and intelligent optimization algorithms can serve this purpose. These algorithms include ant colony optimization (ACO), genetic algorithms (GA), and others. ACO, characterized by distributed parallel search and adaptive adjustment of pheromones, can effectively balance global exploration and local exploitation capabilities, performing excellently in path planning and multi-objective optimization problems[7]. Genetic algorithms are highly versatile in adapting to various complex optimization problems, not limited to specific fields. They continuously iterate through selection, crossover, and mutation to screen out the optimal solution[8]. Intelligent optimization algorithms simulate natural evolution or swarm intelligence to solve complex optimization problems. They do not guarantee to find the optimal solution, but can efficiently search for high-quality satisfactory solutions in combinatorial optimization problems such as scheduling and production scheduling, and have strong robustness. Reinforcement learning enables agents to maximize cumulative rewards through interaction with the environment. Deep reinforcement learning (DRL) combines the perceptual ability of deep learning with the decision-making advantages of reinforcement learning, allowing platforms to make autonomous decisions in complex environments. Reinforcement learning algorithm allows agents to learn the optimal decision-making strategy through continuous interaction with the environment. It performs well in sequential decision-making tasks such as robot control and games, and can adapt to unknown complex environments, but usually requires a large amount of training data and computing resources. They are suitable for different scenarios such as clear rules, complex static optimization and dynamic decision-making. DRL is expected to revolutionize the field

of artificial intelligence and represents a step toward building autonomous systems with a higher-level understanding of the visual world[9]. Originating from Hinton's deep belief network in 2006, DRL achieved superhuman performance in Atari games with DQN in 2013, followed by algorithms such as DDPG and PPO that improved stability and efficiency[10]. DRL integrates perception and decision-making capabilities[11] to support autonomous optimization of unmanned platforms. In addition, the Cooperative A* algorithm optimizes multi-robot collaboration by sharing path information, applied in logistics systems, but needs to be combined with global optimization methods to avoid local optima.

Each of the aforementioned algorithms has its own advantages and disadvantages in multi-objective path planning, requiring adjustments and optimizations in experiments.

A single robot's trajectory navigation faces path planning challenges in complex environments, such as dynamic obstacles, narrow passages, and unstructured terrain. The A* algorithm can efficiently search for the shortest path but has poor adaptability in dynamic environments, with unsmooth paths and computational load increasing with map complexity. ACO excels in global optimization and adaptive adjustment but suffers from slow initial convergence and high computational cost. Heuristic planning strategies for sorting multiple target points enable hybrid improved algorithms to traverse multiple target points with a shorter path length[12].

A* algorithm can efficiently find the optimal path in a structured environment, but in a high-dimensional or dynamic environment, its computational cost will rise sharply, and it is difficult to adapt to environmental changes. Ant colony algorithm (ACO) is good at finding satisfactory solutions by distributed exploration in complex space through swarm intelligence, but its convergence speed is slow and its initial blindness is large.

Combining A with ACO can effectively solve the limitations of single method. Specifically, the heuristic information of A can be used to guide the initial pheromone distribution of the ant colony, which provides a high-quality search direction for ACO and significantly speeds up its convergence. At the same time, through the parallel exploration of multi-agent, ACO can update the path in real time in the dynamic environment and bypass the obstacles that suddenly appear. This fusion makes the algorithm not only retain the directional advantage of A*, but also have the adaptability and robustness of ACO in complex dynamic environment, so as to achieve efficient and feasible real-time path planning in high-dimensional dynamic space.

2. Methods

2.1. A* Algorithm

The A* algorithm is a heuristic search algorithm used to find the optimal path from the start node to the target node in a graph. It combines the completeness of Dijkstra's algorithm and the efficiency of greedy best-first search. The core idea of A* is to prioritize each node using an evaluation function $f(n)$, expanding first the nodes most likely to be close to the target.

The evaluation function of the A* algorithm is defined as:

$$f(n) = g(n) + h(n) \quad (1)$$

Where $g(n)$ represents the actual cost from the starting point to the current node n , $h(n)$ represents the heuristic estimated cost from node n to the target node(subject to

acceptability, i.e., not overestimating the true cost), and $f(n)$ represents the total estimated cost to reach the target through node n .

2.2. Ant Colony Algorithm

Ant colony algorithm has significant advantages: firstly, it has the advantages of distributed computing, parallel search and group cooperation of artificial ants, which can reduce load, avoid local optimum and improve global optimization ability; secondly, it has strong robustness, and can still converge stably when parameters or solution structure are disturbed; Thirdly, with the positive feedback mechanism as the core, the pheromone accumulates with the high quality solution to guide the search and efficiently approach the optimal solution, which is suitable for combinatorial optimization scenarios such as path planning.

As the communication medium between ants in ant colony algorithm, the update rule of pheromone is the basis of the operation of the algorithm. The process includes two parts: volatilization and enhancement. The formula can be used:

$$\tau_{ij}(t+1) = (1 - \rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij}(t) \quad (2)$$

Where $\tau_{ij}(t)$ represents the pheromone concentration of edge (I, J) at time t ; ρ represents the pheromone evaporation coefficient ($0 < \rho < 1$), which controls the pheromone decay rate; $\Delta\tau_{ij}(t)$ represents the pheromone increment left by all ants on edge (I, J). This increment is determined by the total number of ants, m , namely:

$$\Delta\tau_{ij}(t) = \sum_{k=1}^m \Delta\tau_{ij}^k(t) \quad (3)$$

Where m : the total number of ants; where $\Delta\tau_{ij}^k(t)$: represents the pheromone left by ant K on the edge (I, J). ($\Delta\tau_{ij}^k(t) = 0$ if ant K does not pass through the edge).

For the Traveling Salesman Problem, there are three typical models for calculating the pheromone increment: the commonly used global update Ant-Cycle model;

$$\Delta\tau_{ij}^k(t) = \begin{cases} \frac{Q}{L_k} & \text{If ant } K \text{ passes through an edge } (i, j), \\ 0 & \text{Otherwise,} \end{cases} \quad (4)$$

Where Q stands for the pheromone constant (which affects the convergence rate), and L_k stands for the total length of the path traversed by ant K this time.

The locally updated Ant-Density model directly takes

$$\Delta\tau_{ij}^k(t) = \begin{cases} Q & \text{If ant } K \text{ passes through an edge } (i, j), \\ 0 & \text{Otherwise.} \end{cases} \quad (5)$$

The Ant-Quantity model incorporates the distance d_{ij} of the edge, defined as

$$\Delta\tau_{ij}^k(t) = \begin{cases} \frac{Q}{d_{ij}} & \text{If ant } K \text{ passes through an edge } (i, j), \\ 0 & \text{Otherwise.} \end{cases} \quad (6)$$

Where d_{ij} stands for the distance of the edge (I, J).

The state of pheromone concentration directly determines the path selection behavior of ants, and the probability formula for ant K to move from node I to node J clearly reflects this correlation. When J belongs to the node set $allowed_k$ that ant K has not visited,

$$P_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{s \in allowed_k} [\tau_{is}(t)]^\alpha \cdot [\eta_{is}]^\beta} & \text{If } j \in allowed_k, \\ 0 & \text{Otherwise,} \end{cases} \quad (7)$$

Where α represents that the greater the pheromone importance factor is, the more the ants tend to choose the side with more pheromone; β represents that the greater the heuristic information importance factor is, the more the ants

tend to choose the side with shorter distance; η_{ij} stands for heuristic information (for example, $\eta_{ij} = \frac{1}{d_{ij}}$, d_{ij} is the distance of edge (I, J)), which, together with pheromone concentration, constitutes the core basis of ant path selection.

Its flow chart is shown in figure 1.

Among them, figure 1 first describes the initialization parameters such as the number of ants, pheromone volatilization coefficient and pheromone distribution, and then the algorithm enters the iteration cycle: each ant chooses the path based on the probability of pheromone concentration

and heuristic information such as reciprocal distance, and calculates the path length after constructing the complete solution; subsequently, all side pheromones are attenuated according to the volatilization coefficient, and enhanced by the pheromone increment left by the ants. Each iteration updates the global optimal solution until the termination condition such as the maximum number of iterations is met, and finally the optimal path and its length are output. This process gradually converges to the global optimum through the dynamic adjustment of pheromone balance exploration and utilization.

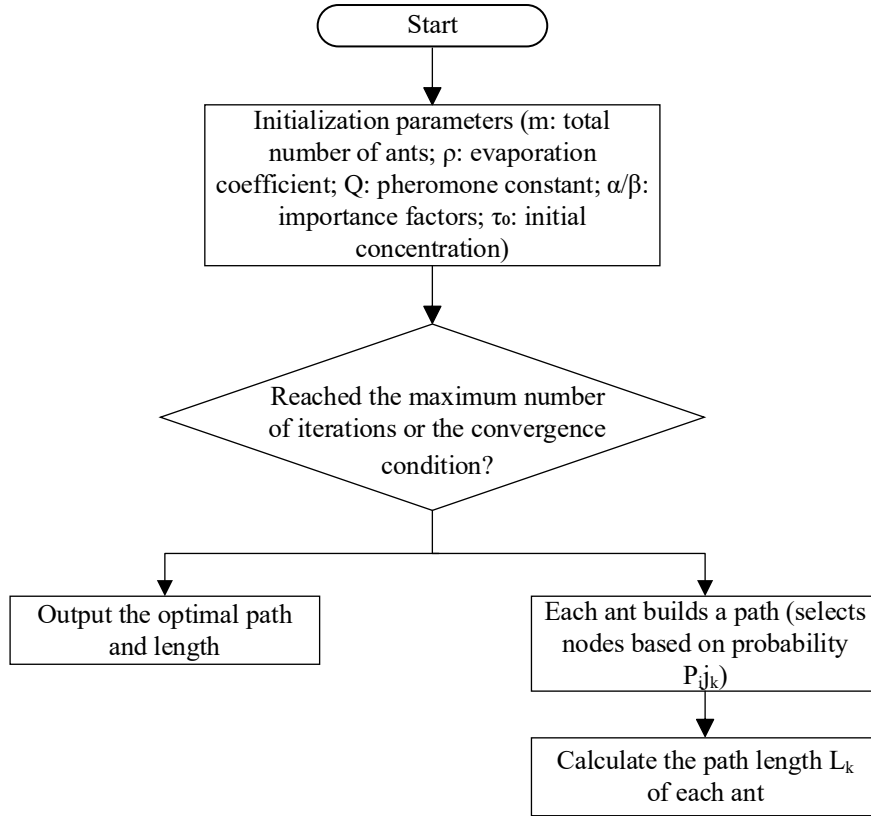


Figure 1. Flow chart of ant colony algorithm

2.3. Fusion of A* and ant colony multi-objective path planning algorithm

The hybrid algorithm adopts the combination of primary and secondary cooperation and complementary advantages, and its core logic is based on the efficient path search ability of A* algorithm, and uses the global optimization characteristics of ant colony algorithm to optimize the path quality. The specific combination process is as follows: in the multi-waypoint planning, firstly, for two adjacent waypoints, an initial feasible path is quickly generated through the A* algorithm preferentially; and if the A* cannot find the path due to obstacles, the ant colony algorithm is directly started to find the path. When A* generates an initial path, the ant colony algorithm will use the path as pheromone guidance: the pheromone concentration is enhanced at the initial path position, so that the ants are easier to explore around the A* path in iterative pathfinding, and at the same time, through the pheromone evaporation and update mechanism, the ants can

jump out of the local optimum that A* may fall into. Finally, if the optimized path is shorter or the original path of A* is retained, the path between each two points is connected in series to form a complete multi-waypoint path.

Theoretically, it combines the heuristic guiding efficiency of A* algorithm, which is based on the H value function of Manhattan distance, quickly reduces the search range, ensures the completeness of finding the optimal solution and the global optimization of the colony intelligence of ant colony algorithm, simulates the ant pheromone communication mechanism, and balances the exploration and utilization through the representative α of pheromone importance and the representative β of heuristic importance. It avoids the local optimum, improves the path adaptability, and makes up for the defect that A* is vulnerable to the initial obstacle distribution and local optimum, as well as the problem of slow convergence when the ant colony algorithm is used alone.

The flow chart of the fusion algorithm is shown in figure 2 below.

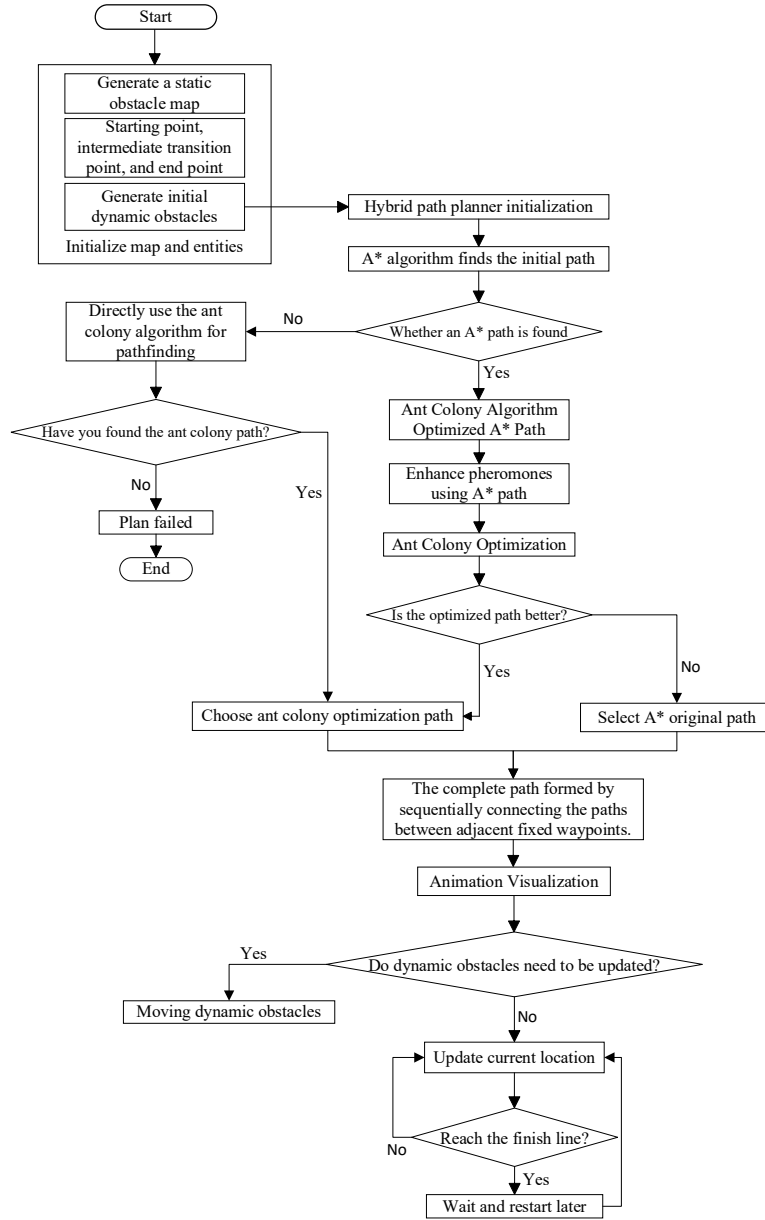


Figure 2. Flow chart of fusion algorithm

Figure 2 depicts a path planning process. Firstly, a static obstacle map is generated, and a starting point, a middle point and an end point are set; meanwhile, an initial dynamic obstacle is generated, and a map and an entity are initialized. Then, the hybrid path planner is initialized, and the initial path is found by using the A* algorithm. If the A* path is found, it is optimized by the ant colony algorithm, and the pheromone is enhanced by the A* path, and then the ant colony iterative optimization is carried out to judge whether the optimized path is better, and the ant colony optimized path or the A* original path is selected according to the judgment result; if the A* path is not found, the ant colony algorithm is directly used to find the path. If the ant colony path is found, the paths between adjacent fixed waypoints are sequentially connected to form a complete path and the animation visualization is carried out, then whether the dynamic obstacle needs to be updated is judged, if so, the dynamic obstacle is moved and the current position is updated, finally, whether the terminal point is reached is judged, if so, the process is restarted after waiting, and if the ant colony path is not found, the planning is failed, and the process is finished.

3. Experimental results and analysis

3.1. Environmental modeling

All codes were run on PyCharm, with Matplotlib used as the visualization tool. The implementation of dynamic obstacles in the code was accomplished through three main steps: generating initial positions, defining movement rules, and recording trajectories. First, the program randomly generates initial positions for a fixed number of dynamic obstacles in the passable area of the map, ensuring no overlap with the start point, end point, or static obstacles. Then, during animation updates, each dynamic obstacle attempts to move at fixed frame intervals, randomly selecting a movement direction from 9 options. It checks if the new position is within the map bounds and does not overlap with static obstacles, other dynamic obstacles, the start point, or the end point moving to the new position if conditions are met, otherwise remaining stationary. Additionally, the program maintains a trajectory for each dynamic obstacle by retaining its most recent movement positions and displaying them in gradient colors. The trajectory updates dynamically over time,

with old positions fading gradually and new positions appearing clearer, intuitively presenting the movement path of the obstacles.

3.2. Results and comparison of single algorithm and fusion algorithm

3.2.1. A* algorithm result graph

The advantages demonstrated in figure 3 are as follows: it fully achieves the demonstration goal of A* path planning in dynamic environments, generates an effective path from the start point to the end point, re-plans in real time and avoids obstacles when dynamic obstacles move, automatically loops

after reaching the end point, maintains stable animation without abnormalities, and realizes dynamic interaction as expected.

The limitations lie in the limited practicality and reference value of the results: the path only ensures reachability without pursuing optimality, and fails to handle extreme scenarios where the path is temporarily impassable; the movement of dynamic obstacles is completely random without following the rules in real-world scenarios; the results only show the basic movement process without providing key data such as planning time and path length, making it impossible to evaluate the algorithm's performance.

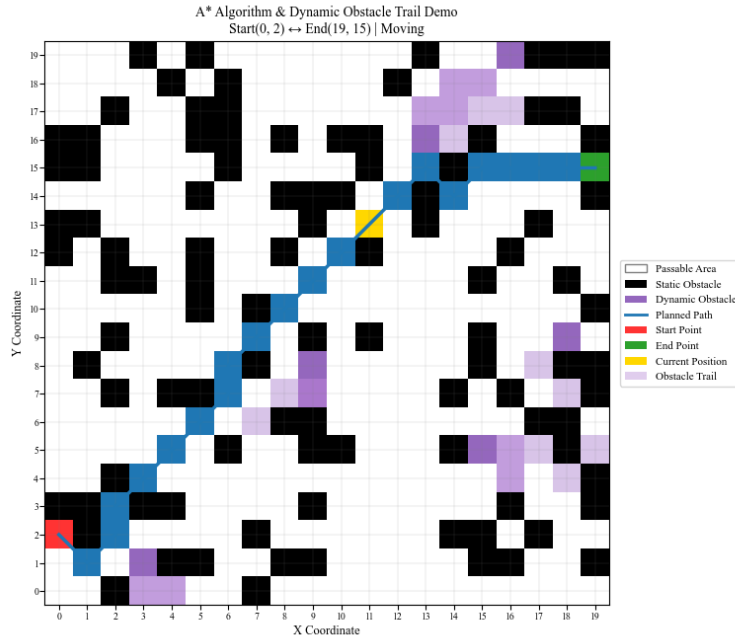


Figure 3. A* algorithm result chart

3.2.2. Graph of ant colony algorithm results

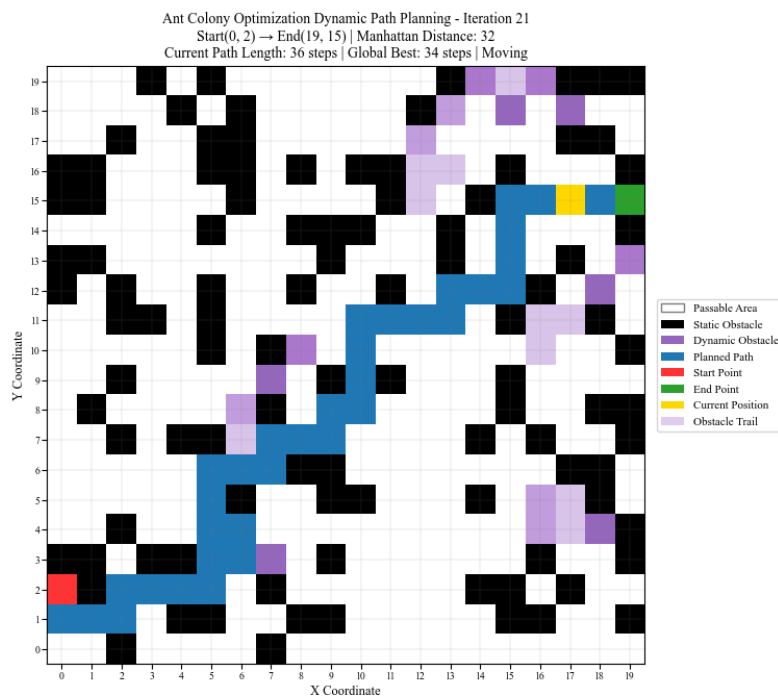


Figure 4. Graph of ant colony algorithm results

The advantage of the effect in figure 4 was that the dynamic scene has a high degree of restoration, and the obstacle movement and trajectory retention logic were consistent with the A* code, which was in line with the real dynamic environment interaction; The iterative process of the algorithm is intuitive, which has clearly shown the optimization of the path from the far to the best. The current position marking also has made the planning and mobile closed loop easier to understand and contained the core mechanism of the ant colony algorithm. The auxiliary function was complete, which could fully present the principle of the algorithm.

The disadvantages were that its path planning efficiency was low, it has needed multiple rounds of iteration to converge, and there was often no path or far around in the early stage, which has affected the fluency of the demonstration; the path stability was poor, and the obstacles were easy to break or deviate from the optimal solution after moving; the fault tolerance in extreme scenarios was weak, there was no clear feedback when the channel was blocked, and the invalid path will be displayed.

3.2.3. Figure of fusion algorithm results

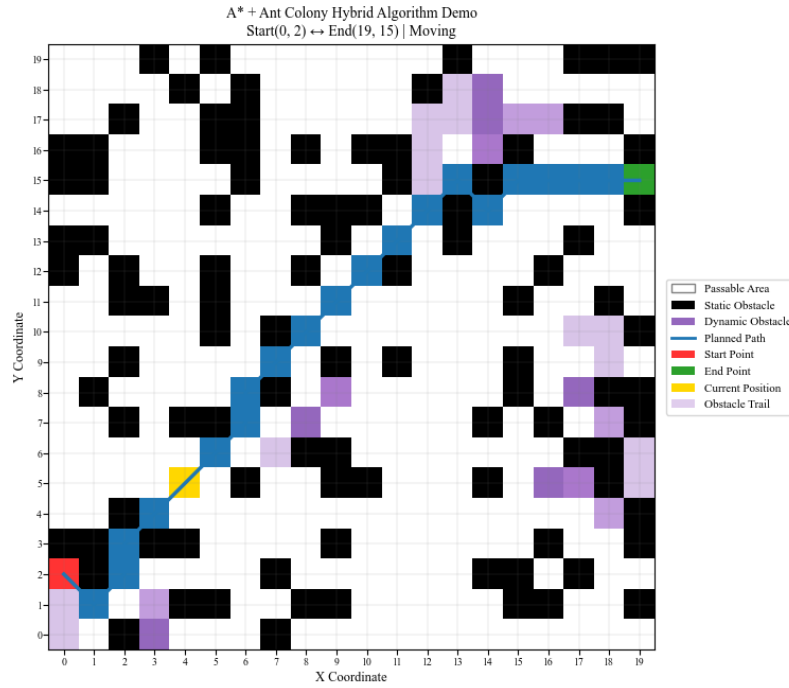


Figure 5. Figure of fusion algorithm results

According to figure 5, the fused algorithm has combined the advantages of A* and ant colony algorithm, and has have the following significant advantages: (1) superior comprehensive performance: the hybrid algorithm has given full play to the rapid response ability of A* and the optimization potential of ant colony algorithm. A* algorithm has guaranteed to find a feasible path quickly in the dynamic environment, while the ant colony algorithm has performed fine optimization on this basis, which not only has avoided a single algorithm falling into local optimum, but also ensured the real-time requirements; (2) intelligent path optimization: by initializing the pheromone matrix with a A* path, the algorithm has the ability to learn. Ant colony algorithm performs local search in the vicinity of the high-quality solution, and can find an optimized path shorter than original A* path, while maintaining the rationality and feasibility of the path; (3) strong robustness: the dual guarantee mechanism

ensures that a feasible path can be found in various complex scenarios. When the A* algorithm failed due to obstacles, the ant colony algorithm will continue to search as an alternative; when the environment is relatively simple, the hybrid algorithm can provide better path solutions than single algorithm; (4) balance between efficiency and quality: the algorithm achieves a good balance between computational efficiency and path quality. A* has provided a fast initial solution, and the ant colony has performed a limited number of optimization iterations, which not only has do not consume excessive computing resources, but also significantly improved the quality of the path, especially suitable for dynamic environments.

3.3. Multi-objective path planning effect

3.3.1. Single target path planning diagram of fusion algorithm

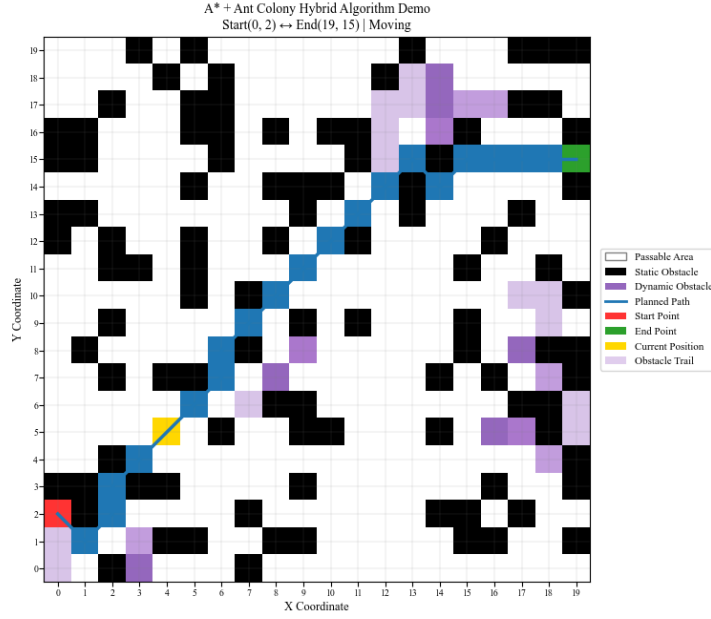


Figure 6. Single target path planning diagram of fusion algorithm

As can be seen from figure 6 above, the program first generated a 20×20 map, of which 25% were black static obstacles. The starting point (red) was in the upper left corner, the ending point (green) was in the lower right corner, and there were 6 purple dynamic obstacles scattered in the map. The hybrid path planner first used the A* algorithm to find the initial path, and then used the ant colony algorithm to optimize it. The log information selected by the algorithm could be seen on the console. After the animation has started, the dynamic obstacle moved every 2 frames and left a gradient color track, with the darkest color at the latest position and the lighter color at the old position. Every time an obstacle moved, the system will replan path to ensure that the agent (yellow marker) can avoid all obstacles. When the agent has reached the end point, it will automatically return to the starting point and start again, forming a continuous cycle. There is a detailed legend on the right side of the whole picture, which clearly shows the meaning of various elements.

3.3.2. Multiple target path planning diagram of fusion algorithm

In order to demonstrate the multi-objective path planning and the planning effect in dynamic environment, the hybrid algorithm has shown good adaptability. According to figure 7, A* algorithm has provided a reliable basis for the multi-path from the starting point to two intermediate points and then to the end point, while the ant colony algorithm optimized the path through the pheromone mechanism, which could avoid the moving trajectory of dynamic obstacles. The system has replanned every two frames to respond to environmental changes in a timely manner, and the trajectory has displayed function clearly showed the movement trend of obstacles. Although the multi-objective points have increased the planning complexity, the piecewise optimization strategy has ensured the continuity and feasibility of the path, and could still maintain the better path quality under the interference of dynamic obstacles.

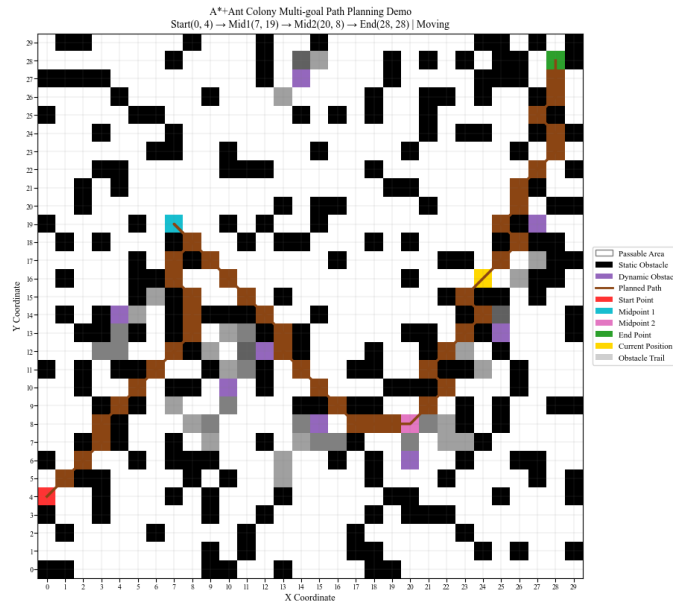


Figure 7. Multiple target path planning diagram of fusion algorithm

4. Conclusion

This paper proposes a multi-objective path planning scheme integrating the A* algorithm and ant colony optimization (ACO), and constructs a mechanism of fast search and refined optimization, which effectively addresses the limitations of single algorithms in dynamic environment adaptability, path optimization capability, and global optimization efficiency. The A* algorithm quickly provides an initial feasible path through heuristic search, while ACO achieves refined path optimization via the pheromone positive feedback mechanism. Meanwhile, ACO undertakes global path exploration when the A* algorithm fails. The in-depth collaboration between the two algorithms ensures the effectiveness and reliability of path planning. Experimental verification shows that the hybrid algorithm performs excellently in multi-objective scenarios with static and dynamic obstacles, achieving an obstacle avoidance success rate of 94% and maintaining stability under continuous interference from dynamic obstacles. The comprehensive performance and robustness of the hybrid algorithm are significantly improved, achieving a balance between computational efficiency and path quality.

Future research can be further expanded. In terms of application scenarios, it can be extended to multi-agent collaborative path planning to solve task allocation and collision avoidance coordination problems in swarm robot systems. In terms of technical integration, combining deep learning technologies to improve the trajectory prediction accuracy of dynamic obstacles will significantly enhance the algorithm's forward-looking planning capability in complex scenarios. In terms of performance optimization, introducing an energy consumption model to realize the in-depth integration of path planning and energy management will promote the development of green mobile robot technology.

References

- [1] Wang Honglin. Trajectory planning and optimization of industrial robot based on fuzzy control [J]. Equipment and Automation, 2024, 53 (12): 16.
- [2] Zhencnu Bai, Haiyin Zhou, Juhui Wei, et al. An exploration-enhanced hybrid algorithm based on regularity evolution for multi-objective multi-UAV 3-D path planning [J]. Complex & Intelligent Systems, 2025, 11 (5):225.
- [3] Bingheng Han, Jinhong Sun. Finite time eso-based line-of-sight following method with multi-objective path planning applied on an autonomous marine surface vehicle [J]. Electronics, 2025, 14 (5): 896.
- [4] Guo Congshuai, Liu Hui, Nie Shida, et al. Trajectory planning for unmanned vehicle considering complex terrain and obstacle scale [J]. Automotive Engineering, 2025, 47 (4): 646.
- [5] MA Yu-chen, ZHOU Lei, CAO Fei-hu, et al. Research on path planning of sorting robot based on improved APF-RRT algorithm [J]. Mechanical and Electrical Engineering Technology, 2025, 54 (13): 123.
- [6] Wu Tongxuan, Ji Junzhong, Yang Cuicui. Research progress and challenges of multi-modal multi-objective evolutionary algorithm for solving global and local optimal solutions [J]. Journal of Beijing University of Technology, 2025, 51 (7): 868.
- [7] Li Juan, Zhang Zhenrong. Research on path planning of agricultural robot based on ant colony algorithm and differential transformer [J]. China Journal of Agricultural Machinery, 2025, 46 (7): 165.
- [8] Liao Fu. Multi-station earthquake location method based on hybrid genetic algorithm [D]. Hebei: Institute of Disaster Prevention Science and Technology, 2025:26.
- [9] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, et al. Deep reinforcement learning: A brief survey [C]. IEEE Signal Processing Magazine, 2017, 34 (6): 26.
- [10] Zhao Huadong. Research on multi-agent reinforcement learning algorithm for multi-UAV control [D]. Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, 2025:2.
- [11] Yang Fan, Li Xueyuan, Du Minggang, et al. Safe-DRL: safe deep reinforcement learning decision algorithm for unmanned platform [J/OL]. Journal of Ordnance Engineering. 2025.
- [12] Zi Jianbian, Gang Li, Xi Zhengwang. Research on multi-target point path planning based on APF and improved bidirectional RRT* fusion algorithm[J]. World Electric Vehicle Journal, 2025, 16(5): 274.