Robotica (2025), **43**, pp. 3594–3614
doi:10.1017/S0263574725102610

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Research on mobile robot path planning based on multi-strategy improved ant colony algorithm

Xinghua Wang[1], Jie Wang[1], Jiawen Cao[1] and Ruijin Sun[2]

[1]School of Mechanical Engineering, Shanghai Institute of Technology, Shanghai, China
[2]School of Engineering Innovation, Shanghai Institute of Technology, Shanghai, China
Corresponding author: Xinghua Wang; Email: xhwang526@163.com

Received: 17 December 2024; **Revised:** 8 August 2025; **Accepted:** 7 September 2025;
First published online: 1 October 2025

Keywords: ant colony algorithm; path planning; mobile robot

Abstract

Aiming at the issues of traditional ant colony algorithm (ACO) in mobile robot path planning, including initial search blindness, susceptibility to local optima, and slow convergence, this paper proposes a multi-strategy improved ant colony algorithm (MS-ACO). Firstly, dynamic non-uniform distribution of initial pheromones is implemented by integrating the repulsive field from artificial the potential field method. Secondly, the heuristic information is enhanced to improve global search capability while constraining unnecessary path turns. Thirdly, an improved pheromone update strategy is developed by adopting distinct updating mechanisms for different evolutionary phases. Finally, dynamic parameter adaptation is achieved through optimized weight coefficients and volatility coefficients that coordinate with the pheromone update strategy, better aligning with the iterative characteristics of ant colony optimization. Experimental results demonstrate that MS-ACO effectively addresses the limitations of traditional ACO. Under identical experimental conditions, it achieves a 30.4% reduction in path length, 37.8% decrease in pathfinding time, and 71% fewer turns compared to conventional methods, verifying the feasibility and superiority of the proposed algorithm.

1. Introduction

As a critical domain in mobile robotics research, path planning technology forms the foundation for robots to navigate accurately from start to finish while completing tasks [1]. Over decades of development, researchers worldwide have proposed numerous effective path planning methodologies, including the A* algorithm [2], artificial potential field (APF) [3], neural networks [4], genetic algorithms [5], fuzzy logic [6], and particle swarm optimization (PSO) [7]. While these approaches have achieved notable success in mobile robot navigation, they exhibit inherent limitations when handling complex environments [8].

The ant colony optimization (ACO) algorithm, a classic heuristic inspired by natural ant foraging behavior, offers distinct advantages such as positive feedback reinforcement, robustness, distributed computation, and parallelism. Its compatibility with other heuristic algorithms further enhances optimization potential. Consequently, ACO has been widely adopted in mobile robot path planning in recent years [9]. However, deeper investigations reveal persistent challenges, including premature convergence to local optima, excessive iterations, and slow convergence rates.

To address the aforementioned limitations, numerous scholars have conducted extensive research. For example, Xu et al. [10] proposed an improved multi-step ACO algorithm that integrates the random sampling region concept of Informed RRT* into the adaptive multi-step ACO. This enhanced algorithm enables robots to identify high-quality paths more efficiently in path planning tasks, reduces search space size, and improves planning efficiency and accuracy. However, in complex environments, the shortest paths identified may not be optimal. Li et al. [11] modified pheromone update rules in ACO

© The Author(s), 2025. Published by Cambridge University Press.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

by applying Darwinian evolutionary theory, thereby accelerating iteration speed and reducing computational time. Nevertheless, their approach neglected the kinematic constraints of mobile robots. Niu et al. [12] designed a guided heuristic search strategy to facilitate the ant colony's transition from initial disordered exploration phases. Concurrently, a reward-punishment mechanism was introduced to accelerate algorithmic convergence, with redundant inflection points being eliminated through greedy algorithms. However, the algorithm demonstrates limited advantages in total computational time when applied to low-complexity maps. Pan et al. [13] replaced conventional eight-neighborhood searches with dynamically expanded neighborhood methods, reducing path lengths and turning points. However, convergence iterations remained comparable to the original ACO as problem scales increased. Song et al. [14] developed an adaptive ACO by modifying pheromone distribution rules and implementing adaptive volatilization coefficients, which improved algorithmic efficiency but yielded suboptimal solution quality. Wang et al. [15] integrated Monte Carlo metrics of feasible nodes into sigmoid functions to calculate path optimality probabilities (Monte Carlo factors) for ant decision-making, significantly enhancing path planning quality and overall performance, albeit with practical implementation limitations. Sun et al. [16] employed fruit fly optimization for rapid grid map pre-searching to initialize pheromone distributions for ACO, followed by global path planning with ACO. Path smoothing techniques were further applied to minimize turning angles and cumulative rotation, substantially improving search speed and precision. Tan et al. [17] hybridized ACO with PSO, utilizing PSO to train initial parameters for optimal path planning. However, this approach lacked comprehensive consideration and validation of practical algorithm implementation.

To overcome ACO's inherent weaknesses – local optima entrapment, blind initial exploration, and insufficient global awareness – this paper proposes methods to optimize the traditional ACO algorithm, including improved initial pheromone distribution, modified heuristic functions, and optimized pheromone updating methods, to enhance the efficiency of path planning and improve the results of path planning.

2. Traditional ant colony algorithm

2.1. Basic principles

The ACO algorithm originated from the pioneering work of Italian scholar Dorigo M., who first proposed the concept in 1992 [18]. Inspired by the foraging behavior of ants, the algorithm simulates how ants deposit volatile pheromones along their paths while searching for food. The pheromone concentration is inversely proportional to the path length – shorter paths accumulate higher pheromone levels, thereby attracting more ants. Through this positive feedback mechanism, a collective reinforcement process occurs, ultimately leading to the establishment of a globally optimal shortest path through repeated iterations by the ant population.

2.2. State transition probability

In the traditional ACO algorithm, ants navigate based on the pheromone concentration of subsequent nodes. During the t-th iteration, the transition probability P_{ij}^{k} that ant k to moves from node i to node j is defined as:

$$P_{ij}^{k} = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}(t)\right]^{\beta}}{\sum_{j=allos_k}\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}(t)\right]^{\beta}}, j \in allow_k \\ 0, j \notin allow_k \end{cases} \tag{1}$$

$$\eta_{ij}(t) = \frac{1}{d_{ij}} \tag{2}$$

where α is the weight coefficient of pheromones, β is the weight coefficient of the heuristic function, $\tau_{ij}(t)$ is the pheromone concentration function, and $\eta_{ij}(t)$is the distance heuristic function. $allow_k$is

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

the set of next feasible nodes, and d_{ij} is the Euclidean distance between current node i and candidate node j.

2.3. Pheromone emission and renewal

In the traditional ACO algorithm, the pheromone concentration at node i undergoes evaporation governed by the evaporation coefficient ρ ($0 < \rho < 1$) as the ants move from node i to node j. Simultaneously, the pheromone concentration at node j is augmented. The updated pheromone concentration at time $t+1$ is formulated as:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t, t+1) \tag{3}$$

$$\Delta\tau_{ij}(t, t+1) = \begin{cases} \dfrac{Q}{L_k}, \text{The path passed by ant } k \text{ in this iteration} \\ 0, \text{else} \end{cases} \tag{4}$$

where Q is the pheromone intensity coefficient, $\Delta\tau_{ij}(t, t+1)$ is the amount of pheromone deposited on the path from node i to node j by ant k in this iteration, and L_k is length of the path traversed by ant k in this iteration.

3. Improved of traditional ant colony algorithm

3.1. Optimization of initial pheromone distribution

In traditional ACO algorithms, the initial pheromone concentration is uniformly distributed [19], resulting in blind exploratory behavior during the early search phase, slow initial convergence, and increased computational time costs. To resolve these issues, this study integrates the APF method to establish a pheromone distribution reference system while preserving the autonomous search characteristics of ACO. Specifically, leveraging the principle that the straight-line distance between two points is the shortest, a path is generated from the start point by circumventing obstacles using repulsive potential fields and ultimately reaching the target point. The integration with APF offers dual advantages for path planning:

1. Goal-oriented guidance: The suboptimal paths generated by APF provide directional cues, reducing purely stochastic exploration;
2. Obstacle avoidance: The repulsive field characteristics of APF inherently reflect critical environmental features.

However, the APF-generated path serves solely as a reference baseline for pheromone distribution rather than a final solution. This design choice stems from two inherent drawbacks of APF: susceptibility to local minima and insufficient path smoothness.

To balance guidance and diversity, this study proposes a dynamic pheromone allocation strategy, where the pheromone increment at path nodes is inversely proportional to their distance from obstacles, as formulated in Eq. (5). This design ensures that critical turning points farther from obstacles receive stronger pheromone reinforcement, while maintaining exploration potential for other nodes.

Parameter sensitivity analysis (Figure 1) reveals that the algorithm achieves optimal performance when the coefficient falls within the range of 0.2C–0.3C. Excessively high pheromone concentrations (>0.3C) improve early convergence speed by 21% but degrade final solution quality, whereas overly low concentrations (<0.2C) diminish the guidance effect. The initial pheromone increment is empirically determined as 0.25C, with the initial pheromone distribution formalized in Eq. (6).

$$\Delta\tau_{APF} = \frac{0.25C}{(1+d_i)} \tag{5}$$

$$\tau_{ij}(0) = \begin{cases} C + \Delta\tau_{APF}, (i, j)\epsilon \text{ The path of the line segment} \\ C, \text{else} \end{cases} \tag{6}$$

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

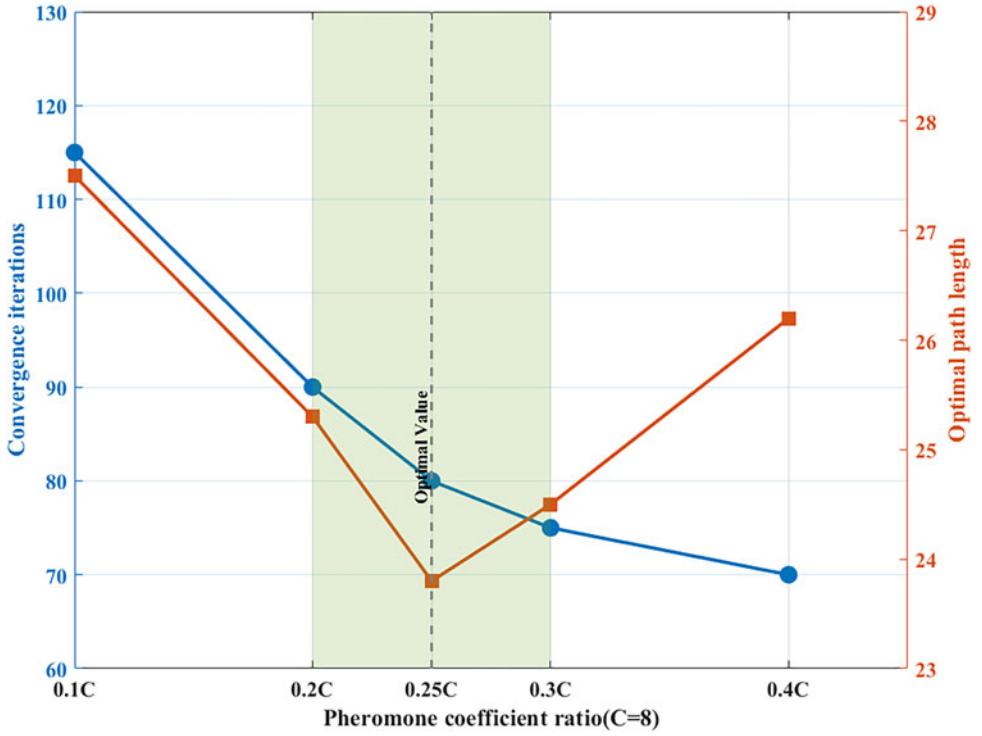


Figure 1. *Effect of pheromone coefficient on algorithm performance.*

where C is constant, set to 8 as adopted in [20]; d_i is the Euclidean distance from node i to the nearest obstacle.

3.2. Optimization of heuristic information

In traditional ant colony algorithms, the heuristic function η_{ij} is defined as the reciprocal of the distance between the current node i and the candidate next node j, as shown in Eq. (2). However, this formulation neglects the global guidance from candidate nodes to the target node, resulting in insufficient exploration of the solution space and susceptibility to local optima. To resolve this issue, this study draws inspiration from the bidirectional evaluation concept of the A* algorithm's heuristic function (Eq. (7)) and innovatively proposes a hybrid heuristic function (Eq. (8)) that integrates global path guidance and suppresses unnecessary turns.

$$f(n) = g(n) + h(n) \tag{7}$$

$$\eta_{ij}^{'}(t) = \frac{\varphi}{d_{ij}} + \frac{\omega}{d_{jg}} + \frac{\gamma}{(1+\Delta\theta)} \tag{8}$$

where $f(n)$ is the total cost from the start point to the end point; $g(n)$ is the accumulated cost from the start point to the current point n; $h(n)$ is the estimated cost from the current point n to the goal point; $\eta_{ij}^{'}(t)$ is improved heuristic information function; d_{ij} is the Euclidean distance between the current node i and the next node j; d_{jg} is the Euclidean distance from the next node j to the target point g; $\Delta\theta$ is the turning angle between the current path segment and the previous one (must satisfy $|\Delta\theta| \leq \theta_{\max}$, where $\theta_{\max}$ is the robot's maximum allowable turning angle); φ, ω are dynamic weights, where $\varphi = \frac{i_{ter}}{Max_{iter}}$, $\omega = 1 - \varphi$; γ is dynamic penalty coefficient, where $\gamma = \gamma_0 * (1 - \frac{i_{ter}}{Max_{iter}})$, where γ_0 set to 0.5 through extensive simulation experiments.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

The optimized model enhances global navigation capacity while preserving local path optimization capabilities through the introduction of a goal-oriented factor. Although Eqs. (7) and (8) share superficial similarities in form, their functional roles are fundamentally distinct: the *f(n)* in the A* algorithm serves as a precise cost evaluation function, rigorously calculating the exact cumulative cost from start to goal, while the η_{ij} in this work acts as a probabilistic heuristic factor, guiding probabilistic path selection rather than deterministic cost minimization. This improvement is not a direct transplantation of A* mechanisms but rather establishes a dynamic weight adjustment model that shifts algorithmic focus across iterations. Early Phase: Prioritizes global exploration (dominated by ω) to escape local optima and map the environment. Later Phase: Gradually intensifies local refinement (dominated by φ) to optimize path smoothness and precision. This time-varying mechanism effectively mitigates the "line-of-sight limitations" inherent to traditional A* algorithms in complex obstacle environments. Furthermore, the integration of steering angle suppression (via $|\Delta\theta| \leq \theta_{max}$) ensures generated paths comply with the kinematic constraints of mobile robots by penalizing abrupt directional changes.

3.3. Optimization of pheromone update strategy

To resolve the balance between convergence speed and global search capability in traditional ACO algorithms, this study proposes a dual-phase dynamic pheromone update strategy based on convergence state monitoring. The core innovation lies in constructing an adaptive reward-penalty system with dual-evaluation mechanisms, where intelligent phase partitioning is achieved through a dynamic path quality evaluation function, rather than relying on simplistic fixed-ratio division.

In the implementation, the phase partitioning is dynamically adjusted based on two metrics: the path diversity index (ξ) and the convergence rate factor (ε), as formulated in Eqs. (9) and (10). The algorithm transitions to the convergence phase when $\xi < 0.2$ and $\varepsilon < 0.01$. This dynamic partitioning mechanism offers stronger theoretical justification compared to conventional fixed-iteration-ratio partitioning methods. The parameter settings are informed by the order parameter criteria from phase transition theory in complex systems, which quantifies critical thresholds for state transitions.

The innovative mechanism proposed in this study is structured as follows: during the exploration phase (Phase I), a normal distribution-based probability-weighted update method is employed to reinforce paths whose lengths fall within the interval $[\mu\text{-}\sigma, \mu+0.5\sigma]$, where μ denotes the mean length and σ represents the standard deviation. This asymmetric interval design effectively retains exploration potential for suboptimal solutions. In the convergence phase (Phase II), a reward-penalty coupling mechanism is established wherein not only the optimal path is reinforced, but paths exceeding $\mu+\sigma$ in length are subjected to penalty imposition through a pheromone decay factor λ. This dual-phase mechanism strategically balances exploration-exploitation dynamics while enhancing solution quality through differential treatment of path candidates.

$$\xi = 1 - \left(\frac{N_{best}}{Max_{iter}} \right) \tag{9}$$

$$\varepsilon = \frac{\left| L_t^{best} - L_{t-1}^{best} \right|}{L_{t-1}^{best}} \tag{10}$$

where N_{best} is the optimal path visitation count; L_t^{best} is current-generation optimal solution; and L_{t-1}^{best} is previous-generation optimal solution.

Phase I (Exploration Phase) ($\xi \geq 0.2$ or $\varepsilon \geq 0.01$)

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} * \Phi(L^k) \tag{11}$$

where $\Phi(L^k)$ is Gaussian weight function, defined as: $\exp\left(-\frac{(L^k-)^2}{2\sigma^2}\right) * I_{[-\sigma,+0.5\sigma]}(L^k)$; $I_{[a,b]}(.)$ is Indicator function, where $L^k \in [a, b]$, otherwise 0.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

Phase II (Convergence Phase) ($\xi < 0.2$ and $\epsilon < 0.01$):

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}^{best} - \lambda \sum_{L^k > \mu+\rho} \Delta\tau_{ij}^{k} \tag{12}$$

where $\Delta\tau_{ij}^{best}$ is pheromone increment of the global optimal path, defined as $\Delta\tau_{ij}^{best} = \frac{Q_{best}}{L_{best}}$ ($Q_{best} = 1.5Q$ is the elite reinforcement coefficient); λ is dynamic penalty decay factor, defined as $\lambda = 0.5 + 0.3tanh(\frac{\sigma}{\mu})$, where tanh (.) denotes the hyperbolic tangent function, ensuring $\lambda \in (0.5,0.8)$.

3.4. Parameter optimization

In traditional ant colony algorithms, critical parameters – including the pheromone weight coefficient, heuristic function weight coefficient, and volatilization coefficient – are typically predefined as fixed constants. This static configuration leads to inherent limitations in balancing the exploration-exploitation trade-off during the search process. To resolve this issue, this section proposes a dynamic parameter adaptation strategy, which optimizes these coefficients by integrating the phase-specific characteristics of the pheromone update mechanism described in the preceding section. This adaptive approach ensures that parameter values are dynamically aligned with the distinct requirements of the exploration and convergence phases, thereby enhancing algorithmic flexibility and solution quality.

1. Optimization of Weight Coefficients α and β: The pheromone weight coefficient α determines the influence of pheromone concentration on path selection by ants, while the heuristic function weight coefficient α governs the impact of heuristic factors (e.g., distance) on path selection. In the early stages of the algorithm, when pheromone concentrations across paths exhibit minimal disparity, prioritizing the heuristic-based distance metric as the primary factor for node selection is critical. Concurrently, reducing the pheromone weight coefficient helps mitigate premature convergence to suboptimal paths. In later stages, after iterative updates amplify pheromone differentiation between high- and low-quality paths, elevating the pheromone weight coefficient becomes advantageous to leverage accumulated search experience, thereby accelerating convergence toward optimal solutions. Simultaneously, diminishing the heuristic weight coefficient reduces noise from distance-based biases, enabling more focused exploitation of pheromone-guided information. Therefore, this paper adjusts the weight coefficients as shown in Eqs. (13) and (14):

$$\alpha = \begin{cases} \frac{1}{2}\alpha_0, \xi \geq 0.2 \text{ or } \varepsilon \geq 0.01 \\ \frac{3}{2}\alpha_0, \xi < 0.2 \text{ and } \varepsilon < 0.01 \end{cases} \tag{13}$$

$$\beta = \begin{cases} \frac{3}{2}\beta_0, \xi \geq 0.2 \text{ or } \varepsilon \geq 0.01 \\ \frac{1}{2}\beta_0, \xi < 0.2 \text{ and } \varepsilon < 0.01 \end{cases} \tag{14}$$

Where: $\alpha_0 = 3$, $\beta_0 = 5$. The initial value was adopted from article [20] and determined using the controlled variable method.

2. Optimization of the Evaporation Coefficient ρ: The evaporation coefficient plays a dual role in balancing exploration and exploitation. A smaller evaporation coefficient enhances solution diversity by preserving pheromone trails across paths, thereby expanding the search space and mitigating premature convergence. However, this comes at the cost of slower convergence due to delayed elimination of suboptimal paths. Conversely, a larger evaporation coefficient accelerates convergence by rapidly diminishing pheromone levels on inferior paths, but risks trapping the algorithm in local optima by prematurely reducing path diversity. To address this trade-off, this paper introduces a phase-dependent evaporation rate adjustment strategy:

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

Initial phase: The evaporation rate is reduced ($\rho = 0.5\rho_0$) t to maintain path diversity and avoid local optima by allowing broader exploration.

Late-stage: The evaporation rate is increased ($\rho = 1.5\rho_0$) to accelerate convergence by intensifying pheromone accumulation on high-quality paths, synergistically integrated with the phase-specific pheromone update mechanisms proposed earlier. The specific adjustments are formulated as detailed in Eq. (15):

$$\rho = \begin{cases} \frac{1}{2}\rho_0, \xi \geq 0.2 \text{ or } \varepsilon \geq 0.01 \\ \frac{3}{2}\rho_0, \xi < 0.2 \text{ and } \varepsilon < 0.01 \end{cases} \tag{15}$$

where $\rho_0 = 0.7$. The initial value was adopted from article [20] and determined using the controlled variable method.

3.5. The flowchart of MS-ACO algorithm

Through the above analysis, the ant colony algorithm with four improvement strategies is applied to the mobile robot path planning. The algorithm flowchart and specific steps are shown in Figure 2:

4. Environmental modeling

Environment modeling for motion maps plays a critical role in mobile robot path planning, as it directly affects the positional accuracy and operational reliability of robotic navigation. Current mainstream environmental modeling methods include topological methods [21], which exhibit strong environmental dependency and high computational costs; visibility graph methods [22], which suffer from excessive memory consumption and generate non-smooth paths; and grid-based methods [23], distinguished by their structural simplicity, reliability, adaptability, and precision [24]. Given these comparative advantages, this study adopts a grid-based approach to construct the mobile robot's operational environment. The environment is discretized into grids where white grids represent feasible regions for robot navigation and black grids denote obstacle zones. The obstacle grids are subjected to an inflation process, as shown in Figure 3. The environment map is depicted in Figure 4.

The relationship between the grid index i and its corresponding coordinates is defined by Eq. (16):

$$\begin{cases} x_i = l \times \left[mod\,(i, A) - \frac{l}{2} \right] \\ y_i = l \times \left[A + \frac{1}{2} - ceil\frac{i}{A} \right] \end{cases} \tag{16}$$

where x_i is the x-coordinate of the i-th grid; y_i is the y-coordinate of the i-th grid; i is the grid index; l is edge length of each grid cell; A is the grid dimension; mod is modulo operation; and ceil is ceiling operation.

5. Simulation and experimental verification

5.1. Definition of evaluation metrics

To quantify algorithm performance, the following evaluation metrics are adopted:

1. Path Length: Defined as the total Euclidean distance traveled by the robot from the start to the goal. In the grid map, where each grid cell has an edge length of 1, the path length is calculated as:

$$L = \sum_{k=1}^{n} \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2} \tag{17}$$

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

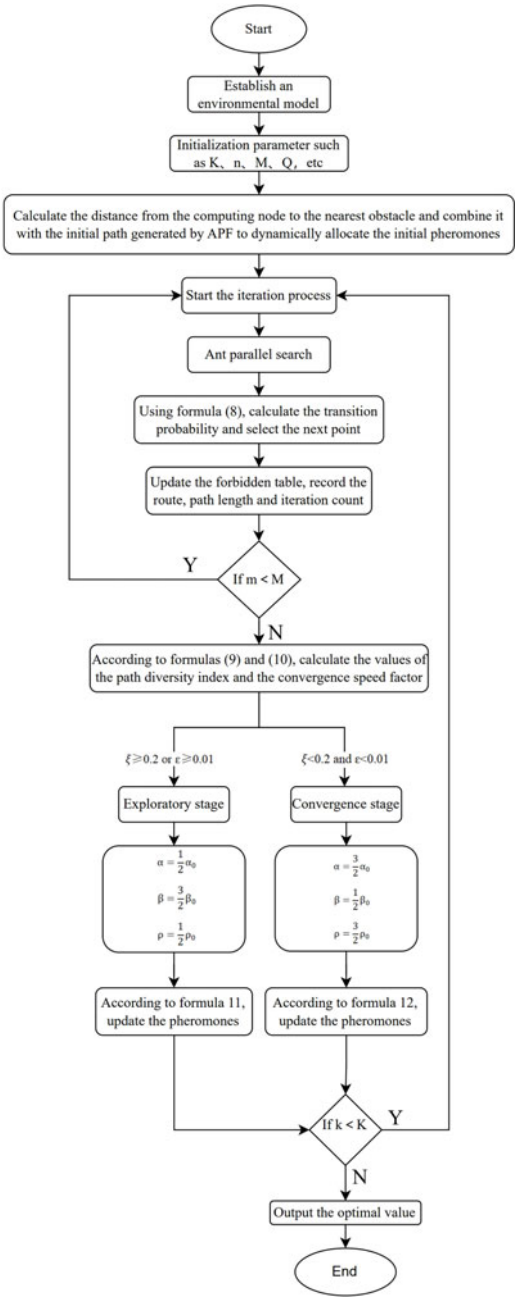


Figure 2. *The flowchart of MS-ACO algorithm.*

where (x_k, y_k) denotes the coordinates of the k-th node in the path, and n is the total number of path nodes.

2. Number of Turns: Calculated by detecting directional changes in consecutive movements along the path. Specifically, a turn is counted if the robot's movement direction (e.g., east, south, west, north, or diagonally) between node i and node j differs from the direction between node j and node k.

3. Search Time: Measured using the system clock during experiments to record the computational time required to generate a feasible path.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

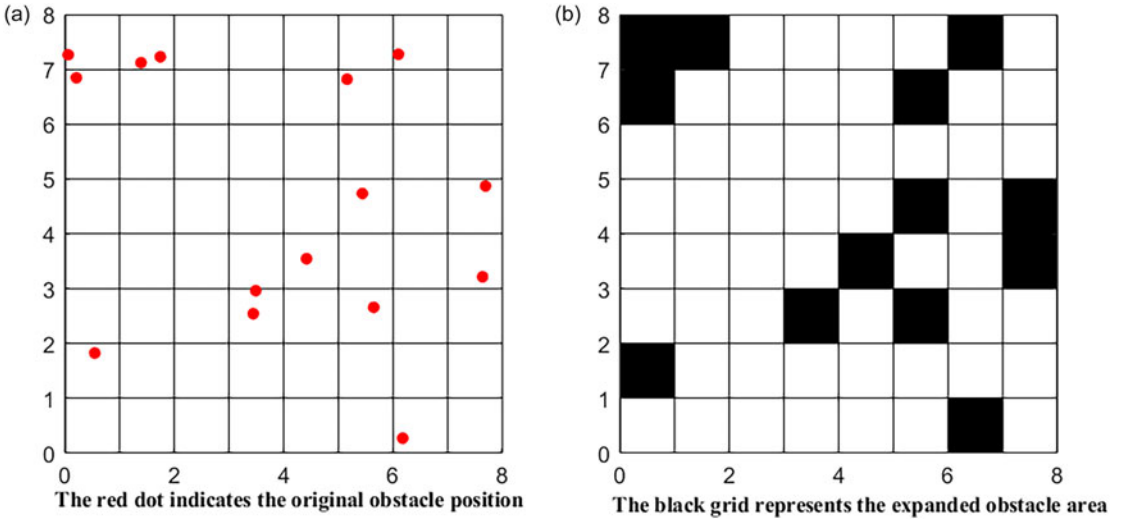


Figure 3. *Obstacle grid expansion processing diagram.*

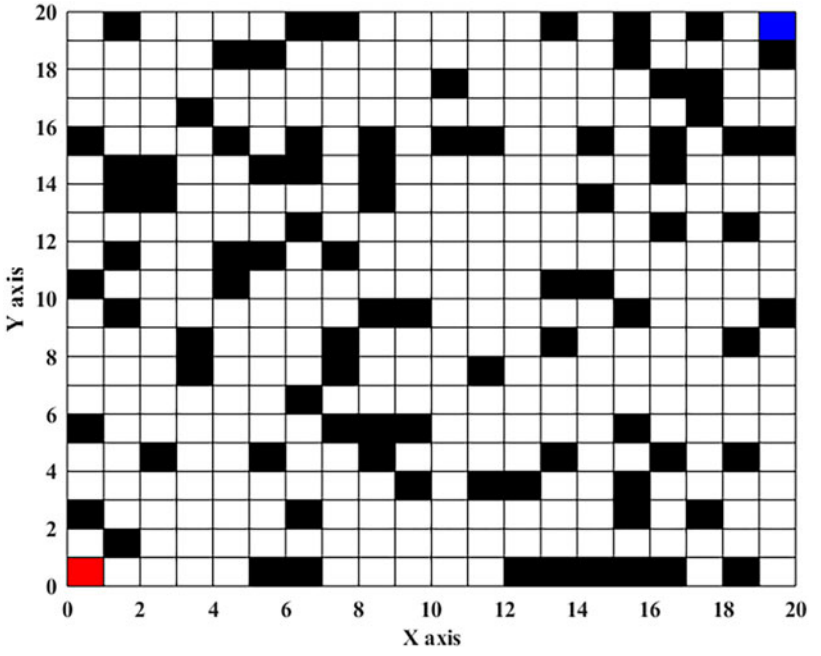


Figure 4. *Raster map with obstacles.*

5.2. Simulation

To validate the feasibility of the proposed optimized algorithm (MS-ACO), simulation experiments were conducted on a Windows 11 system using MATLAB R2021a. Path planning simulations were performed on grid-based maps of dimensions 10 × 10, 20 × 20, and 30 × 30, comparing the performance of four algorithms: the traditional ACO, the Improved ACO algorithm from reference [25] (JACO), the Enhanced ACO algorithm from reference [26] (IACO), the recently proposed Beluga Whale Optimization (BWO) algorithm [27], and the proposed MS-ACO algorithm. In the simulation experiments, the starting coordinates for ants are set to (1,1), with the goal coordinate fixed at (m, m), where m denotes the edge length of the square map.

The initial parameters for the optimized algorithm proposed in this paper are shown in Table I.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

Table I. *Initial parameter of algorithm.*

Parameter name	Parameter value	Parameter name	Parameter value
Number of iterations	100	Number of ants	50
α	3	β	5
ρ	0.7	Q	1

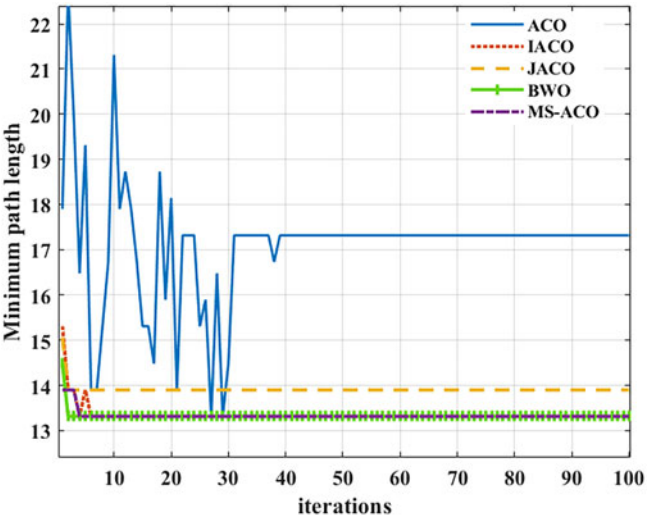


Figure 5. *10 × 10 convergence curve comparison diagram.*

As illustrated in Figure 5, in the 10 × 10 simulated map, all four algorithms achieve convergence after iterative optimization. The traditional ACO algorithm converges after 39 iterations with a converged path length of 17.3137, significantly lagging behind the improved algorithms. The JACO algorithm requires only 2 iterations to converge, yet attains a suboptimal path length of 13.8995, indicating room for further refinement. In contrast, the IACO, BWO, and MS-ACO algorithms demonstrate superior search capabilities: they converge after 6, 2, and 4 iterations, respectively, achieving the shortest converged path length of 13.3137.

Combined with the results in Figure 6, all three improved algorithms – JACO, IACO, and MS-ACO – demonstrate varying degrees of success in circumventing the local optima traps inherent to the traditional ACO algorithm within the relatively simple 10 × 10 grid environment. And BWO is also capable of finding the optimal path. A brief explanation of the path-trajectory diagrams: the routes shown are not the converged paths after 100 iterations, since in some later experiments, convergence was not achieved within 100 iterations. In the traditional ACO diagram, the second-best path found during those 100 iterations is displayed, whereas in the diagrams for the three optimized algorithms and BWO, the best path found during those 100 iterations is shown.

Further analysis of Figure 5 reveals that while JACO achieves the fastest convergence speed (2 iterations), its path quality is constrained by insufficient exploration capability due to premature convergence in early iterations. In contrast, the proposed MS-ACO algorithm integrates three key innovations: dynamic non-uniform initialization of pheromones based on environmental features; a dual-phase strategy prioritizing comprehensive exploration before pursuing rapid convergence; and dynamic parameter adaptation aligned with the dual-phase pheromone update mechanism. These enhancements enable MS-ACO to maintain path optimality (achieving the shortest path length of 13.3137), improving convergence speed by 33.3% compared to IACO (4 iterations vs. 6 iterations). However, due to the exploration phase in the MS-ACO algorithm, its convergence speed lagged behind BWO on the 10 × 10 map.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

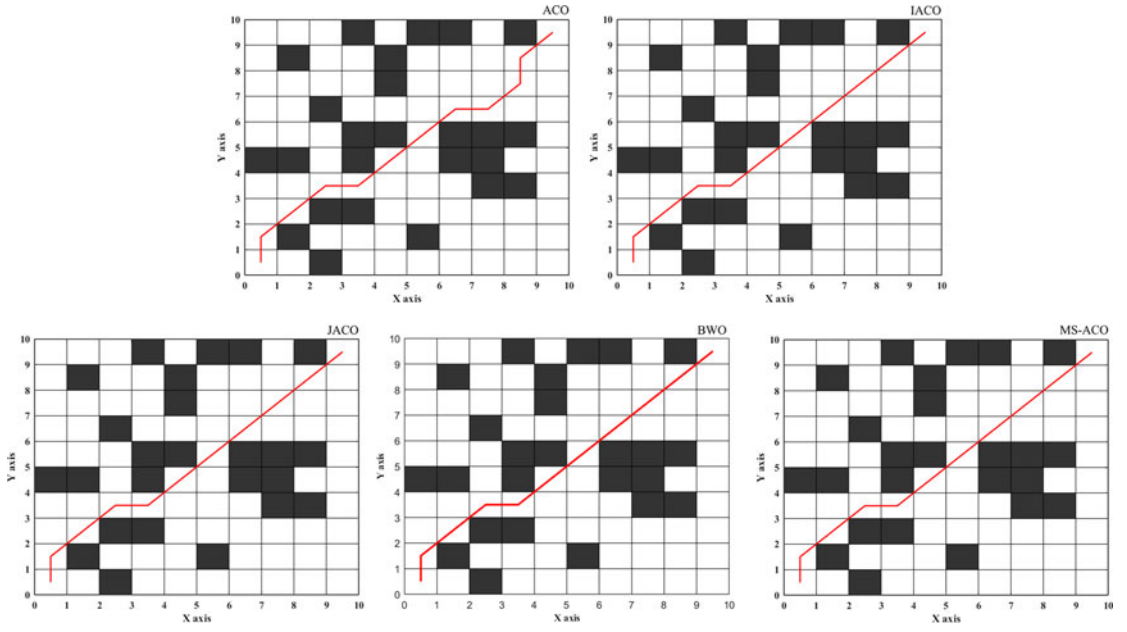


Figure 6. *10 × 10 path trajectory comparison diagram.*

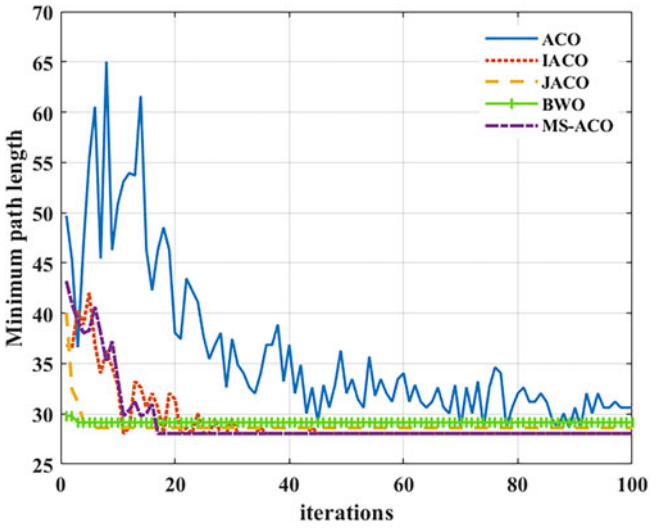


Figure 7. *20 × 20 convergence curve comparison diagram.*

As shown in Figure 7, in the 20 × 20 simulation map, only four algorithms achieve convergence. The traditional ACO algorithm fails to converge within 100 iterations, *d* exhibiting significant degradation in path planning capability in larger-scale environments. Among the improved algorithms, JACO and BWO maintain rapid convergence, requiring only 5 and 3 iterations to converge. However, its converged path length of 28.6274 and 29.1416 remains suboptimal, with 2.1% and 3.9% discrepancy from the optimal solution. Both IACO and MS-ACO demonstrate superior search performance, converging to the optimal path length of 28.0416. Notably, MS-ACO achieves this result in 17 iterations, achieving a 2.65× speed improvement over IACO (45 iterations).

Combined with the results in Figure 8, the multi-strategy enhancements of MS-ACO exhibit significant advantages in the 20 × 20 grid map:

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

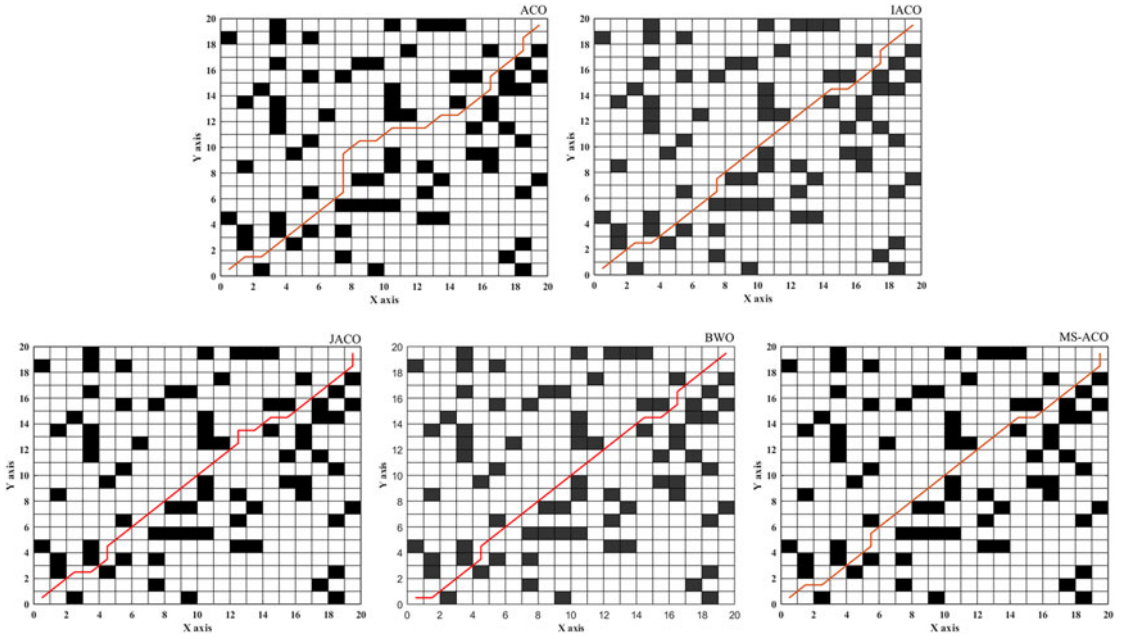


Figure 8. *20 × 20 path trajectory comparison diagram.*

1. Exploration Phase Effectiveness: The inclusion of an explicit exploration phase effectively mitigates premature convergence issues observed in JACO, ensuring thorough search space coverage before exploitation.
2. Convergence Acceleration: The non-uniform initial pheromone distribution based on environmental features, combined with the convergence-oriented phase, substantially improves algorithmic efficiency.
3. Turn Suppression Mechanism: By penalizing directional changes in the heuristic function, MS-ACO generates paths that better align with the kinematic constraints of mobile robots, enhancing practical applicability.

Compared to the 10 × 10 environment map, the performance advantages of MS-ACO become more pronounced in the 20 × 20 environment, further validating the feasibility of its hybrid strategy. These results underscore the superiority of integrating prior-guided initialization and dynamic parameter adaptation over static parameter frameworks.

As illustrated in Figure 9, in the 30 × 30 simulation map, the convergence capabilities of the algorithms exhibit substantial divergence.

Traditional ACO and IACO fail to converge within 100 iterations, highlighting the limitations of complex environments on algorithmic convergence. JACO and BWO retain their rapid convergence trait, achieving convergence in 9 and 3 iterations with suboptimal path lengths of 46.5269 and 43.7838, but their suboptimality gap reaches 7.3% and 1% compared to the global optimum. MS-ACO demonstrates exceptional path planning performance, converging to the optimal path length of 43.3553 at the 18th iteration. Notably, its convergence efficiency achieves a 6× improvement over IACO (which converges at the 108th iteration under a 200-iteration limit).

Combined with the results in Figure 10, the advantages of MS-ACO's multi-phase collaborative mechanism become even more pronounced in the 30 × 30 grid map. Key insights include:

1. Exploration Phase Superiority: The explicit exploration phase enables MS-ACO to overcome the local optima barriers that restrict JACO, ensuring global search robustness in highly complex environments.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

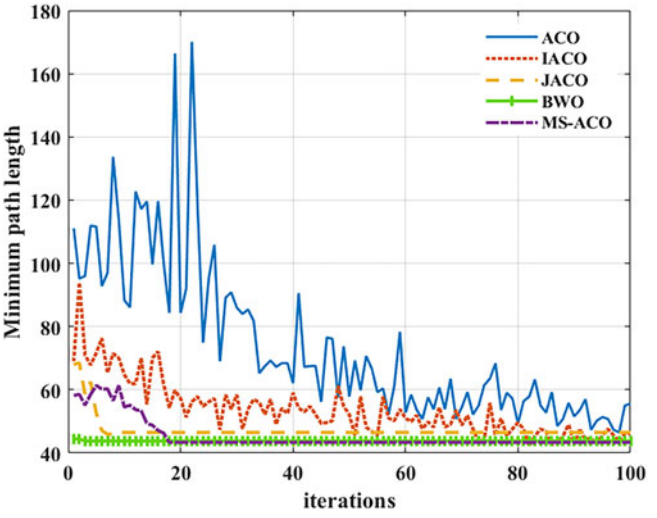


Figure 9. *30 × 30 convergence curve comparison diagram.*

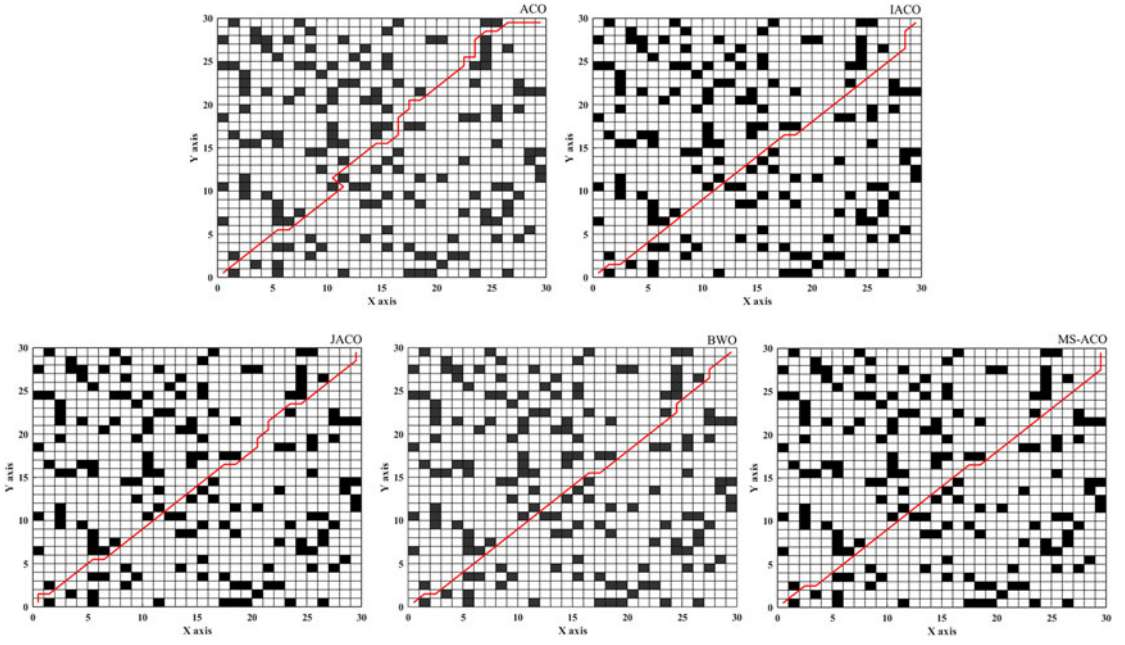


Figure 10. *30 × 30 path trajectory comparison diagram.*

2. Efficiency Optimization: The non-uniform initial pheromone distribution and convergence-phase strategies synergistically enhance search efficiency, achieving rapid convergence while maintaining solution quality.
3. Path Smoothness: By suppressing unnecessary turns in the heuristic function, MS-ACO generates paths with minimized directional changes, aligning with the kinematic constraints of mobile robots for practical deployment.

Comparative Analysis: Relative to smaller-scale maps (10 × 10 and 20 × 20), MS-ACO's performance gains in the 30 × 30 environment are markedly amplified. Specifically, it converges to the optimal path length (43.3553) with 6× faster efficiency than IACO, while JACO exhibits a 7.3% suboptimality

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

Table II. *Comparison of algorithm results.*

Map size	Algorithm type	Convergence iterations	Convergence path length	Number of turns
10 × 10	ACO	39	17.3137	7
	JACO	2	13.8995	3
	IACO	6	13.3137	3
	BWO	2	13.3137	3
	MS-ACO	4	13.3137	3
20 × 20	ACO	—	—	14
	JACO	5	28.6274	10
	IACO	45	28.0416	8
	BWO	3	29.1416	7
	MS-ACO	17	28.0416	7
30 × 30	ACO	—	—	18
	JACO	9	46.5269	13
	IACO	108	43.3553	6
	BWO	3	43.7838	8
	MS-ACO	18	43.3553	5

The data with the horizontal lines in the table represent the results obtained by increasing the DACO iteration count to 200, as it could not converge within the original 100 iterations; in the table, the dashes (—) represent that no results were obtained after 100 iterations.

gap. This scalability underscores the algorithm's ability to adaptively balance exploration depth and exploitation intensity across varying environmental complexities.

The simulation results (as shown in Figures 5-10 and Table II) demonstrate that the proposed MS-ACO exhibits significant advantages in path planning performance. Compared to the traditional ACO algorithm, MS-ACO achieves multi-dimensional improvements across all tested map environments:

1. Path optimization: In the 10 × 10 map, the optimal path length is reduced by 23.1% compared to ACO.
2. Path smoothness: In the 30 × 30 complex terrain, the number of path turns is reduced to 29.4% of ACO's value, aligning better with the kinematic constraints of mobile robots.
3. Convergence efficiency: In the 10 × 10 map, the iteration count is reduced by 89.7% relative to ACO.

Comparative analysis reveals that while MS-ACO's convergence speed is slightly slower than JACO – a trade-off inherent to its exploration phase – it maintains superiority in path quality and smoothness, with these advantages becoming more pronounced in higher-complexity environments. Notably, MS-ACO consistently obtains theoretical optimal paths (matching IACO's performance) across all tested map scales (10 × 10 to 30 × 30), while achieving 33%, 62.2%, and 83.3% reductions in convergence iterations for the respective map sizes. These results validate the effectiveness of MS-ACO's balanced exploration-convergence mechanism, which harmonizes solution optimality with computational efficiency through adaptive phase transitions and dynamic parameter tuning.

In summary, the proposed MS-ACO effectively resolves the limitations of traditional ACO algorithms, including blind initial exploration, slow convergence rates, and susceptibility to local optima. Through the synergistic integration of multi-strategy optimizations – such as dynamic parameter adaptation, dual-phase exploration-convergence decoupling, and motion-constrained heuristic design – the algorithm achieves simultaneous enhancements in global search capability and convergence efficiency. Compared with the latest path planning algorithms, the proposed method also demonstrates significantly enhanced search performance.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

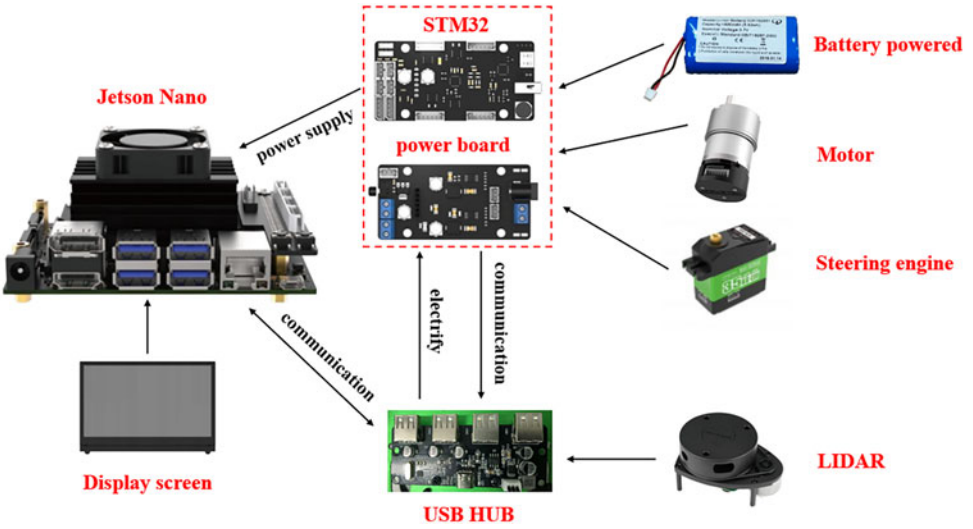


Figure 11. *Each module connectiondiagram.*

5.3. Experimental verification

The experimental validation was conducted on a robotics platform with the NVIDIA Jetson Nano as the central controller. The platform comprises key hardware components, including gimbal servo motors and a LiDAR module, integrated with a mainboard, expansion board, and peripheral devices. The connection diagram of the mainboard, expansion board, and hardware modules is illustrated in Figure 11.

Global mapping using LiDAR on the robotic platform involves three core stages:

1. Preprocessing (Data Optimization and Feature Handling)

The LiDAR emits pulsed laser beams and calculates distances to obstacles via Time-of-Flight or phase difference measurements upon receiving reflected signals, generating raw point cloud data.

A point cloud comprises discrete "points," each containing angular (azimuth) and distance information. To enhance data quality, the following optimizations are performed:

1. Noise Filtering: Statistical filtering is applied to remove isolated noise points, and invalid long-range data is truncated using a threshold (e.g., maxUnange = 5.0).
2. Motion Distortion Compensation: Inertial measurement unit or odometry data is integrated to correct point cloud stretching caused by robotic motion during scanning.
3. Sensor Calibration: Invalid regions (e.g., blind zones) are filtered based on LiDAR mounting parameters and calibration results.

This preprocessing pipeline ensures high-fidelity environmental representation for subsequent mapping and navigation tasks.

2. Matching (Alignment of Local Data with Global Map)

The core objective is to determine the robot's pose (position and orientation) within the global map using scan matching techniques, aligning the current frame's point cloud with the existing map. Different algorithms employ distinct methodologies:

1. Gmapping utilizes a particle filter framework based on Rao-Blackwellized particle filtering, where each particle independently maintains a map hypothesis. The algorithm calculates matching scores via scanMatch and balances accuracy with computational load by adjusting the particle count.

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

2. Hector directly aligns point clouds through Gauss-Newton optimization, eliminating reliance on odometry but requiring high-frequency LiDAR data. Robustness is enhanced using multi-resolution map representations.
3. Cartographer constructs local submaps and optimizes pose graphs via the Ceres solver, while loop closure detection mitigates cumulative errors for long-term consistency.

These approaches highlight the trade-offs between computational efficiency, environmental adaptability, and robustness in real-time SLAM applications

3. Map Fusion (Global Consistency Maintenance)

Map fusion, as a core component of simultaneous localization and mapping technology, is implemented through three critical hierarchical processes:

1. Occupancy Grid Update: The system dynamically adjusts grid confidence levels via probabilistic models and adaptively expands map boundaries to accommodate environmental dynamics.
2. Loop Closure Optimization: Cartographer employs the branch-and-bound algorithm for global matching between submaps to ensure topological consistency. Gmapping suppresses accumulative errors in large-scale scenarios by modulating particle filter weights.
3. Autonomous Exploration and Expansion: Integrating frontier detection algorithms with Rapidly-exploring Random Tree (RRT) path planning, the system drives dynamic map growth through active frontier exploration.

This multi-layered framework ensures global consistency, scalability, and real-time adaptability in SLAM applications, balancing computational efficiency with environmental fidelity.

Based on an analysis of algorithm-scenario compatibility, Gmapping demonstrates superior mapping accuracy and global consistency in small-scale environments due to its lightweight architecture and real-time responsiveness. Therefore, this study adopts Gmapping as the core algorithm for environmental modeling, enabling the construction of high-precision and globally consistent maps that satisfy the requirements for mobile robot navigation in spatially constrained scenarios. This selection aligns with the dual objectives of computational efficiency and topological fidelity, ensuring reliable performance in real-world deployments. The initial mapping state and final mapping results are shown in Figure 12. The experimental environment is shown in Figure 13.

Robot Pose Estimation During Motion: Adaptive Monte Carlo Localization (AMCL). AMCL, a core algorithm in the ROS navigation stack, enables probabilistic localization in 2D environments through particle filtering. Its principles and workflow are outlined as follows:

1. Particle Initialization: Initially, a large set of particles (e.g., 1000) is randomly generated across the map space. Each particle encodes a pose hypothesis – position (x, y) and orientation (θ) – and is assigned an initial weight. If the robot's initial position is known (e.g., calibrated via RVIZ's 2D Pose Estimate tool), particles are concentrated near this pose to accelerate convergence.
2. Motion Model Update (Prediction Phase): As the robot moves, each particle's position is updated using odometry data (linear velocity v, angular velocity ω):

$$\begin{cases} x(t+1) = x(t) + v^* \cos(\theta(t)) * \Delta t \\ y(t+1) = y(t) + v^* \sin(\theta(t)) * \Delta t \\ \theta(t+1) = \theta(t) + \omega * \Delta t \end{cases} \tag{18}$$

motion noise (e.g., odometry error parameters $S_{rr}, S_{rt}, S_{tr}, S_{tt}$) is injected to simulate real-world kinematic uncertainty.

1. Observation Model Update (Correction Phase): LiDAR scan data are matched against the static map to compute the likelihood of each particle's pose. Particles with higher consistency between

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

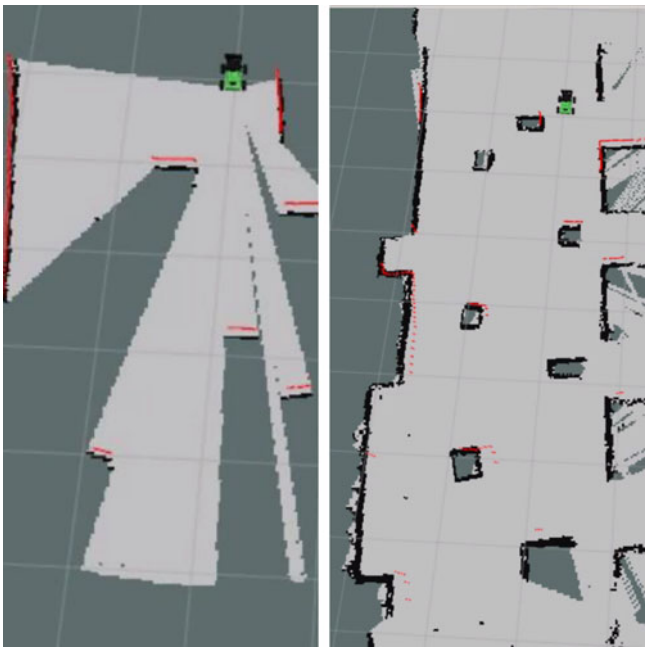


Figure 12. *Initial state of drawing and result of drawing.*



Figure 13. *Experiment site diagram.*

their predicted poses and map obstacles receive increased weights, while mismatched particles are penalized. Adaptive resampling (e.g., KLD-sampling) dynamically adjusts the particle population to mitigate particle degeneracy and maintain localization robustness.

2. Pose Estimation Output: The final estimated pose is derived as the weighted average of all particles, accompanied by a covariance matrix quantifying localization uncertainty.

This probabilistic framework ensures real-time pose estimation under sensor noise and environmental dynamics, enabling reliable navigation in partially observable or dynamic settings.

Target Distance Computation During Robot Motion: The target distance calculation is based on the globally planned path and dynamically updated using real-time localization data to determine the remaining distance. The core workflow is as follows:

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press



Figure 14. *Experimental process of traditional ACO.*



Figure 15. *Experiment process of MA-ACO.*

1. Global Path Planning: Inputs: (a) Start point (current robot position obtained via AMCL localization). (b) Goal point (user-specified target coordinates). (c) Static map (grid map generated by SLAM algorithms). Output: A global path represented as a sequence of discrete waypoints, where the total path length serves as the initial estimate of the target distance.
2. Real-Time Distance Update: Inputs: (a) Global path (sequence of discrete waypoints). (b) Real-time localization data (current robot position from AMCL or odometry). Computational Logic: (a) Match the nearest waypoint: Identify the waypoint on the global path closest to the robot's current position. (b) Remaining distance calculation: Iteratively sum the Euclidean distances between consecutive waypoints from the matched point to the goal:

$$surplus\ distance = \sum_{i=k}^{n} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \tag{19}$$

Dynamic Update: If the robot deviates from the global path (e.g., due to localization errors or environmental changes), global path replanning is triggered to update the remaining distance.

The experimental procedures for ACO and MS-ACO are illustrated in Figures 14 and 15, with the corresponding results depicted in Figure 16. To ensure statistical reliability, 30 experimental trials were conducted for each algorithm using the robotic platform, and the performance metrics are summarized in Table III.

Synthesizing the experimental results above, in relatively simple map environments, the traditional ACO algorithm still exhibits excessive, unnecessary turns during pathfinding. In contrast, under identical

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

Table III. *Comparison of the experimental results.*

Parameter name	Path length/m			Pathfinding time/s			Number of turns
	Optimal length	Average	Standard deviation	Optimal value	Average	Standard deviation	
ACO	9.78	11.67	1.03	43	45.08	1.05	7
MS-ACO	8.12	8.12	0	28	28.00	0	2

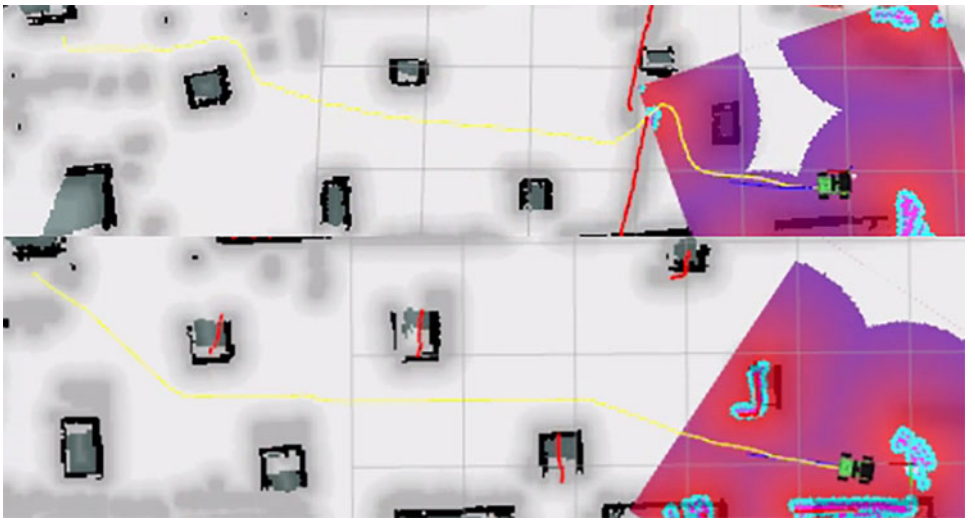


Figure 16. *Experimental results of ACO and MS-ACO.*

conditions, the proposed MS-ACO algorithm successfully completes path planning tasks, generating smoother paths with significantly fewer turns and no local optima entrapment, demonstrating its practical feasibility. According to the data in Table III, MS-ACO outperforms ACO across key metrics:

1. Path length: Reduced by 30.4%;
2. Pathfinding time: Decreased by 37.8%;
3. Number of turns: Minimized by 71%.

These substantial improvements in path quality, computational efficiency, and navigation smoothness highlight MS-ACO's ability to optimize robotic path planning, further validating its effectiveness and superiority over conventional approaches in real-world applications.

6. Conclusion

This study focuses on the optimization of traditional ant colony algorithms in mobile robot path planning, proposing a MS-ACO. By integrating the APF method to implement repulsive force detour planning between start and target points and dynamically adjusting pheromones based on environmental characteristics, the algorithm achieves more rational initial pheromone distribution, thereby enhancing the effectiveness of initial search phases. The evaluation function from the A* algorithm is introduced to incorporate the distance from the next selected node to the target into the heuristic information function, complemented by a turning penalty term. This enhancement ensures both global optimization capability and compliance with mobile robots' motion constraints. A phased pheromone update strategy is implemented, distinguishing between exploration and convergence phases with differentiated

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

reward-punishment mechanisms, which collectively improve global search capability and convergence speed. Dynamic adjustment mechanisms for pheromone weight coefficients, heuristic function weights, and volatilization coefficients are established in accordance with phase-specific characteristics, further enhancing the algorithm's adaptability and overall performance.

The simulation experiments of the ACO, JACO, IACO, BWO, and MS-ACO algorithms are conducted in grid maps of different sizes. The results demonstrate that MS-ACO outperforms other algorithms in terms of path length and inflection points, with its advantages being particularly pronounced in complex environments. Physical robot experiments further validate that MS-ACO achieves significant optimization in path length, pathfinding time, and number of turns compared to traditional ACO, confirming its effectiveness and feasibility. The MS-ACO algorithm exhibits strong performance in mobile robot path planning, effectively addressing issues such as ACO's susceptibility to local optima in complex environments and slow convergence speed, thereby demonstrating high application value. However, this study currently focuses on path planning in static environments. Future research should further investigate path planning in dynamic obstacle environments to enhance the algorithm's applicability in more complex scenarios.

Author contribution. Jie Wang designed the path planning algorithm, drafted the manuscript, and conducted the experiments; Xinghua Wang provided corrections and revisions to the article; Ruijin Sun collected the experimental data.

Financial support. This work was supported by the National Natural Science Foundation of China (No. 51901133). Introduction of Talents Scientific Research Start-up Project (No. YJ2021-49). Shanghai Higher Education Young Teacher Training Support Program (No. ZZ202212056).

Competing interests. All authors disclosed no relevant relationships.

Ethical approval. None.

References

[1] Y. Liu, Y. Du, T. Zhang and J. Bai, "Research on indoor mobile robot path planning based on genetic mechanism improved ant colony algorithm," *J. Beijing Inst. Graph. Commun.* **30**(2), 57–62 (2022).
[2] X. Tang, H. Zhou and T. Xu, "Obstacle avoidance path planning of 6-DOF robotic arm based on improved A* algorithm and artificial potential field method," *Robotica* **42**(2), 457–481 (2024).
[3] M. Ding, X. Zheng, L. Liu, J. Guo and Y. Guo, "Collision-free path planning for cable-driven continuum robot based on improved artificial potential field," *Robotica* **42**(5), 1350–1367 (2024).
[4] M. Singh and D. Parhi, "Path optimisation of a mobile robot using an artificial neural network controller," *Int. J. Syst. Sci.* **42**(1), 107–120 (2011).
[5] A. Karami and M. Hasanzadeh, "An adaptive genetic algorithm for robot motion planning in 2D complex environments," *Comput. Elect. Eng.* **43**(C), 317–329 (2015).
[6] S. Pradhan, D. Parhi and A. Panda, "Fuzzy logic techniques for navigation of several mobile robots," *Appl. Soft Comput.* **9**(1), 290–304 (2009).
[7] G. Li and W. Chou, "Path planning for mobile robot using self-adaptive learning particle swarm optimization," *Sci. China Inf. Sci.* **61**(5), 052204 (2018).
[8] S. Zhang, J. Pu, S. N. and F. Sun, "A review of the application of ant colony algorithm in mobile robot path planning," *Comput. Eng. Appl.* **56**(8), 10–19 (2020).
[9] X. Chen, Y. Kong and X. Fang, "A fast two-stage ACO algorithm for robotic path planning," *Neural Comput. Appl.* **22**(2), 313–319 (2013).
[10] F. Xu and Y. Sun, "Mobile robot path planning based on multi-step ant colony algorithm," *Mod. Mach. Tools Autom. Manuf. Tech.* **2023**(6), 18–21 (2023).
[11] T. Li and H. Zhao, "Path optimization for mobile robot based on evolutionary ant colony algorithm," *Control Decis.* **38**(3), 612–620 (2023).
[12] Q. Niu, X. Dong and Y. Fu, "AGV path planning improvement based on heuristic strategy of ant colony algorithm," *Comput. Integr. Manuf. Syst.* **4**(86), 1–19 (2023).
[13] H. Pan, L. Ao and J. Lu, "Mobile robot path planning based on dynamic extended neighborhood ant colony algorithm," *Trans. Chin. Soc. Agric. Mach.* **55**(2), 423, 432+449 (2023
[14] P. Song, W. Gao and Y. Zhang, "Research on off-road vehicle path planning based on improved ant colony algorithm," *Comput. Simul.* **40**(10), 200–204+325 (2023).

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press

[15] T. Wang, L. Wang and D. Li, "Monte Carlo-based improved ant colony optimization for path planning of welding robot," *J. King Saud Univ. Comput. Inf. Sci.* **35**(2023), 101603 (2023).
[16] J. Sun, Y. Yu and L. Xin, "Research on Path Planning of AGV Based on Improved Ant Colony Optimization Algorithm," *33rd Chinese Control and Decision Conference (CCDC)* (2021) pp. 7567–7572.
[17] Y. Tan, J. Ouyang, Z. Zhang, Y. Lao and P. Wen, "Path planning for spot welding robots based on improved ant colony algorithm," *Robotica* **41**(3), 926–938 (2023).
[18] M. Dorigo, V. Maniezzo and Colorni, "The ant system: an autocatalytic optimizing process Technical Report 91-O16," Milan, Italy, Dipartimento di Elettronica, Politecnicl di Milano (1991).
[19] F. Li, W. Shi and C. Xun, "Mobile navigation design for ROS robots," *Autom. Instrum.* **42**(12), 56–61+66 (2021).
[20] D. Zhang, R. Luo, Y. Yin and S. Zou, "Multi-objective path planning for mobile robot in nuclear accident environment based on improved ant colony optimization with modified A," *Nucl. Eng. Technol.* **55**(5), 1838–1854 (2023).
[21] G. Han, "Automatic parking path planning based on ant colony optimization and the grid method," *J. Sens.* **2021**, 1–10 (2021).
[22] T. Lv and M. Feng, "A smooth local path planning algorithm based on modified visibility graph," *Mod. Phys. Lett. B* **31**(19-21), 1740091–1740097 (2017).
[23] K. Zhang, Y. Zhang and G. Lu, "Loop closure detection based on local semantic topology for visual SLAM system," *Robot* **41**(5), 649–659 (2019).
[24] M. Zhang, Y. Zhang and H. Zhong, "Path planning for coal mine rescue robots based on improved A algorithm," *Coal Field Geol. Explor.* **50**(12), 185–193 (2022).
[25] Y. Huang, Q. Jiang and N. Xun, "Research on robot path planning using an improved ant colony algorithm," *Mech. Des. Manuf.* **2023**(12), 194–198 (2023).
[26] T. Zhou and W. Wei, "Mobile robot path planning based on an improved ACO algorithm and path optimization," *Multimed. Tools Appl.* **2024**(19), 370–394 (2024).
[27] C. Zhong, G. Li and Z. Meng, "Beluga whale optimization: A novel nature-inspired metaheuristic algorithm," *Knowl.-Based Syst.* **251**, 109215 (2022).

Cite this article: X. Wang, J. Wang, J. Cao and R. Sun (2025). "Research on mobile robot path planning based on multi-strategy improved ant colony algorithm", Robotica **43**, 3594–3614. https://doi.org/10.1017/S0263574725102610

https://doi.org/10.1017/S0263574725102610 Published online by Cambridge University Press