Robotica (2025), **43**, pp. 4429–4456
doi:10.1017/S0263574725102932

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Application of improved ant colony algorithm fusing Bresenham and direction factor in mobile robot path planning

Shuai Wu[1,2], Zibo Huang[2], Zijing Ye[1,2], Yapeng Wang[1], Xu Yang[1] and Sio-Kei Im[3]

[1]Faculty of Applied Sciences, Macao Polytechnic University, Macao, China
[2]School of Artificial Intelligence, Dongguan City University, Dongguan, China
[3]Macao Polytechnic University, Macao, China
Corresponding author: Yapeng Wang; Email: yapengwang@mpu.edu.mo

Received: 2 June 2025; **Revised:** 25 October 2025; **Accepted:** 7 November 2025; **First published online:** 5 December 2025

Keywords: mobile robot; path planning; ACO; Bresenham; curvature

Abstract

The traditional ant colony optimisation (ACO) algorithm, when applied to mobile robot path planning, faces several challenges: slow convergence, susceptibility to local optima, and the generation of paths with excessive turning points, all of which reduce the robot's operational efficiency. To overcome these shortcomings, this paper proposes a targeted set of improvements designed to enhance algorithm performance and increase the practicality and efficiency of path planning. First, we introduce an initial pheromone enhancement mechanism based on the Bresenham algorithm. By augmenting pheromone concentration along the approximate straight-line path from the start to the goal, ants are guided to explore in the optimal direction, thereby significantly accelerating convergence. Second, we integrate a directional continuity factor into the path selection probability: by using vector dot products to strengthen the bias toward consistent directions and by coupling this with a curvature-based pheromone reward that favours straighter segments, we ensure smoother, more direct paths. Finally, we apply a spring-model-based smoothing strategy as a post-processing step to the paths generated by the ant colony, reducing path complexity and the number of turns to guarantee efficient and reliable robot motion. To validate the performance of the improved algorithm, we conduct comparative experiments on a MATLAB platform against other enhanced ACO variants reported in the literature. The results demonstrate that our proposed algorithm significantly outperforms these existing methods across all performance metrics, exhibiting superior path planning capabilities.

1. Introduction

In modern industrial production, mobile robots, as intelligent equipment integrating terrestrial devices, underwater exploration systems, and aerial unmanned platforms, are seeing a continuously increasing application value [1]. Their core function, path planning, enables robots to autonomously search for the optimal path from a starting point to an endpoint, achieving the shortest distance, minimal time, and collision-free safety [2]. However, trajectory accuracy directly determines the operational efficiency of robots, making it a critical bottleneck in technological development [3]. Particularly in complex environments with multiple obstacles, path planning must address not only obstacle avoidance but also the kinematic constraints of the robot, posing significant challenges to path generation [4]. As the core of navigation control systems, path planning plays a vital role in enabling the autonomy and intelligence of mobile robots [5].

Path planning is the core of enabling efficient autonomous navigation for mobile robots, directly determining their application capabilities in complex environments. To address the numerous challenges in path planning, current research has intensified, giving rise to various algorithms for optimising path selection. Broadly speaking, path planning algorithms can be divided into two main categories:

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

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

traditional algorithms and meta-heuristic algorithms. Traditional algorithms include the artificial potential field (APF) method [6], Dijkstra's algorithm [7, 8], and the rapidly exploring random tree (RRT) algorithm [9]. Meta-heuristic algorithms encompass the grey wolf optimiser (GWO) [10], particle swarm optimisation (PSO) [11], and ant colony optimisation (ACO) [12], among others.

The APF method is widely favoured in path planning due to its algorithmic simplicity and computational efficiency. However, its limitations, such as susceptibility to local extrema and weak adaptability to dynamic environments, restrict its effectiveness in complex scenarios. Ren et al. proposed an improved artificial potential field (IAPF) method, which optimises PF calculations by leveraging local path information and introduces formulas for obstacle repulsion, target attraction, and resultant forces, effectively addressing the local optima problem and enhancing path planning efficiency [13]. Yang et al. developed an obstacle avoidance trajectory planning method for a dual-robot system based on an IAPF algorithm. By establishing interference and collision criteria based on kinematic analysis, they mitigated the issue of excessive virtual PF forces in traditional APF, significantly reducing trajectory solving time and improving convergence speed [14]. Abdel-Rahman et al. proposed a hybrid path planning approach combining the global optimisation capabilities of an IAPF with the real-time adaptability of the A* algorithm, overcoming the local minima problem of traditional APF. Experimental results showed a 50% improvement in path planning speed compared to the A* algorithm, along with the ability to find shorter paths [15].

Dijkstra's algorithm employs a greedy strategy to incrementally determine the shortest paths from a starting point to all nodes. When combined with heap optimisation, it significantly enhances computational efficiency, making it particularly suitable for solving non-negative-weight path problems in sparse graphs. However, when dealing with negative-weight graphs or dynamically changing graph structures, the algorithm requires special handling or frequent recalculations, which not only increases time complexity but also leads to high space consumption due to the storage of priority queues and distance arrays. Zhou et al. proposed an optimised Dijkstra algorithm that integrates real-time node occupancy and time-window conflict evaluation mechanisms to generate conflict-free, near-optimal paths, effectively avoiding channel congestion and achieving efficient multi-task distribution [16]. Luo et al. introduced an extended Dijkstra algorithm that models surface environments using Delaunay triangulation, converting triangular meshes into two-dimensional navigable channels to solve for optimal paths. Simulation results showed significantly improved path accuracy compared to the traditional Dijkstra algorithm [17]. Nan et al. proposed a Dijkstra algorithm based on adaptive resolution grid optimisation, which markedly improved path planning efficiency while substantially reducing algorithm debugging time [8].

The RRT algorithm, with its random sampling mechanism and dynamic tree structure expansion, can quickly construct a path search framework in highly complex environments. However, the generated paths often lack geometric continuity, and their quality is rarely globally optimal. To address these issues, Chi et al. proposed an autonomous motion planning algorithm, AM-RRT*, based on the RRT. By incorporating a target-point attraction PF to enhance sampling directionality and integrating a dynamics model with cubic B-spline curve optimisation, their experiments demonstrated significantly improved path smoothness and adaptability to dynamic environments compared to the traditional RRT algorithm [18]. Ye et al. introduced an improved IKB-RRT algorithm, which optimises APF guidance, motion control parameter sampling, and dual-tree path optimisation to tackle challenges in orchard environments. Testing showed reduced path length and planning time, along with significantly enhanced path smoothness, providing an efficient path planning solution for orchard robots [19]. Li et al. proposed an RRT algorithm optimised with adjustable probability, sampling regions, and Dijkstra's algorithm (APSD-RRT). The APS-RRT algorithm rapidly generates feasible paths, which are then pruned and smoothed using Dijkstra's algorithm. Experiments validated its superior performance over traditional RRT regarding planning efficiency, path quality, and computational cost balance [20].

In practical applications of mobile robots, traditional path planning algorithms, despite their well-established foundation, struggle to overcome technical limitations such as low computational efficiency, challenging parameter tuning, and susceptibility to local optima. Compared to traditional algorithms, meta-heuristic algorithms exhibit distinct technical advantages: their robust global optimisation capabilities enable precise handling of complex, dynamic, and highly constrained environments; dynamic

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

strategy adjustment mechanisms allow real-time adaptation to environmental changes; parallel computing architectures significantly enhance algorithm execution efficiency; and the integration of random search and diversity maintenance strategies effectively avoids local optima traps. Collectively, these features enable high-precision, high-quality path planning, providing a new technical pathway for the intelligent development of mobile robots.

The GWO, inspired by the hierarchical hunting and prey-encircling behaviours of grey wolves, demonstrates superior global search and rapid convergence performance in high-dimensional function optimisation and complex engineering problems. Luo et al. proposed the EWB-GWO algorithm, which enhances population diversity, convergence speed, and iteration efficiency of GWO by integrating Tent-Sine chaotic mapping, butterfly optimisation strategies, and Bresenham line detection techniques. Experimental results showed that it outperforms similar algorithms in path length, smoothness, and time complexity, providing an efficient solution for mobile robot path planning in complex environments [21]. Zhao et al. introduced the Golden Sine Grey Wolf Optimisation (GSGWO) algorithm, which improves GWO by optimising the initial population with chaotic mapping, incorporating golden sine convergence strategies, and adapting to environmental models. Simulation experiments demonstrated its superiority over traditional GWO in convergence speed, accuracy, and path stability [22]. Gao et al. proposed the Turning Point Grey Wolf Optimisation (TPGWO) algorithm, which enhances GWO's initial population and convergence performance through crossover mutation, roulette wheel selection, and nonlinear convergence factors. Simulation results indicated better convergence, stability, and accuracy than traditional GWO and PSO in patrol robot path planning [23]. Zhou et al. developed an enhanced GWO algorithm (NI-GWO) by optimising trajectory evaluation, convergence factors, and position updates, while integrating the dynamic window approach (DWA) for dynamic obstacle avoidance. This significantly reduced path length and planning time, enhancing UAV obstacle avoidance capabilities [24].

The PSO algorithm, inspired by the foraging and collaborative behaviours of bird flocks [25], exhibits efficient search performance in dynamic environments and multi-objective optimisation problems. However, it suffers from issues such as susceptibility to local optima and slow later-stage convergence. Lin et al. proposed a hybrid PSO-SA algorithm that integrates PSO with simulated annealing to optimise AGV path planning. By introducing dynamic weight adjustments and local search strategies, the traditional PSO algorithm is improved. Simulation experiments demonstrated its superiority over traditional PSO in convergence speed, path length optimisation, and collision-free smooth path generation [26]. Li et al. developed an improved PSO-based optimal energy path planning method, incorporating a parameter-adaptive mechanism based on deep deterministic policy gradients to reduce UAV energy consumption in mountainous terrain [27]. Additionally, Li et al. proposed an enhanced PSO algorithm for optimising collaborative path allocation in SMT machine robots. Incorporating inertia weight updates and fuzzy control learning factors improves global search capabilities. Experimental results showed faster convergence, shorter paths, and reduced energy consumption compared to traditional PSO and genetic algorithms [28]. Gao et al. introduced an improved PSO algorithm for apple-picking robot path planning, optimising obstacle avoidance trajectories of the end effector through dynamic velocity weight adjustments. Experiments indicated faster convergence, reduced collisions, and improved picking efficiency [29].

The ACO algorithm, inspired by the foraging behaviour of ants, excels in solving complex problems such as robot path planning and task sequencing, demonstrating outstanding adaptability. However, in large-scale scenarios, its efficiency is low, and it is prone to premature convergence, which limits its performance. In recent years, numerous researchers have proposed various improvement strategies to address these shortcomings, significantly enhancing the algorithm's effectiveness. Song et al. proposed a path planning strategy integrating an improved ACO algorithm with the DWA. By optimising pheromone initialisation, heuristic functions, and update mechanisms, it generates shorter paths with fewer turning points in static environments, while the enhanced DWA enables dynamic obstacle avoidance [30]. Miao et al. introduced an adaptive ACO algorithm (IAACO), incorporating angle guidance factors, obstacle exclusion factors, and adaptive pheromone update rules to improve the real-time performance, stability, and global optimisation capabilities of indoor robot path planning [31]. Shuai et al.

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

addressed the limitations of traditional ACO, such as stagnation and premature convergence, by proposing a hybrid PS-ACO algorithm. This method combines ACO's pheromone update rules with the local and global search mechanisms of PSO, thereby improving search efficiency and convergence performance [32]. Zhang et al. developed an adaptive improved algorithm based on population information entropy (AIACSE), which optimises population diversity, initial pheromone distribution, and adaptive parameter adjustments, significantly enhancing path planning performance [33]. Zhao proposed an improved ACO algorithm that incorporates non-uniform initial pheromone distribution, direction-guided selection strategies, optimised pheromone coverage and update mechanisms, and segmented pheromone evaporation coefficients. This approach not only improves path search efficiency and smoothness but also effectively balances convergence speed and search capability, significantly enhancing the performance of mobile vehicles in path planning tasks [34]. Zhang et al. tackled the issues of poor diversity and slow convergence in ACO by proposing a dynamic multi-role adaptive collaborative ACO algorithm (MRCACO). Experimental validation in robot path planning confirmed its superiority over traditional algorithms, significantly outperforming the ant colony system (ACS) and max-min ant system (MMAS) [35]. Gui et al. addressed the slow convergence and suboptimal path planning performance of ACO by proposing an improved ACO algorithm (MsAACO). This method integrates direction-guided strategies and adaptive heuristic functions to optimise node selection and path smoothness, significantly enhancing the convergence efficiency and stability of the ACO algorithm [36].

Although previous studies have improved the ACO algorithm and achieved promising results, its optimisation performance still has room for enhancement. This paper conducts an in-depth analysis of the algorithm's limitations in complex scenarios, including weak environmental adaptability, slow convergence speed, and insufficient path smoothness. Four innovative improvement strategies are proposed to further enhance the algorithm's performance.

1. An initial pheromone optimisation mechanism based on the Bresenham algorithm is introduced to enhance pheromone concentration along an approximate straight-line path from the start to the endpoint, guiding ants to prioritise exploration in the optimal direction and effectively improving algorithm convergence speed.
2. A direction continuity factor is incorporated into the path selection probability, leveraging vector dot products to enhance the preference for consistent directions, ensuring path smoothness.
3. A reward mechanism based on curvature pheromone increments is integrated, applying additional rewards to straighter paths to strengthen the ants‘ tendency to select low-curvature paths. This enhances path smoothness and fluidity, generating more efficient and natural robot motion trajectories.
4. A path-smoothing strategy based on the spring model is introduced. This strategy optimises the paths generated by the ACO algorithm through the synergistic effects of spring forces, smoothing forces, and direct connection forces, effectively reducing path complexity and the number of turns.

Based on these improvement strategies, this paper proposes an enhanced ACO algorithm (BCD-ACO). The remainder of the paper is organised as follows: Section 2 describes the path planning environment modelling method; Section 3 reviews the traditional ACO algorithm and the Bresenham algorithm; Section 4 provides a detailed introduction to the proposed improvement strategies; Section 5 conducts parameter optimisation and simulation experiments; and Section 6 summarises the research conclusions.

2. Additional formatting guidelines

In the process of path planning, modelling the map environment is a crucial preliminary step. Adopting an appropriate environmental modelling method not only significantly enhances the efficiency of path search algorithms but also provides strong support for the accuracy and practicality of planning results.

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

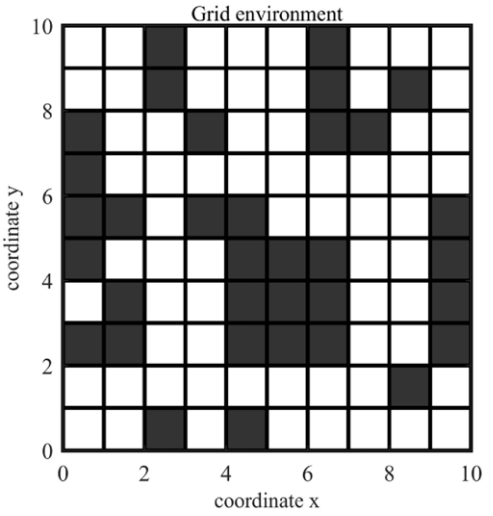


Figure 1. *10*10 grid environment map.*

Currently, commonly used map environment modelling methods include the visibility graph method, the grid map method [37], and the topological map method [38]. These modelling approaches have distinct characteristics, making them suitable for different application scenarios and algorithmic requirements. Among them, the grid map method divides the environment into regular grid cells, representing spatial information in a discretised manner. This approach facilitates computation and handling of complex terrains, making it widely applied in fields such as robot navigation.

In this study, the raster graph method is utilised for environmental modelling, and a raster map with a size of N×N is constructed to achieve the basic environmental representation of path planning. In this modelling process, to ensure the safety and reliability of the path, it is necessary to expand the obstacles. Specifically, if a certain grid has any contact with an obstacle (e.g., partial coverage or adjacency), then this grid is regarded as an obstacle grid and marked as an impassable area. This processing method effectively expands the boundaries of obstacles and enhances the ability to avoid obstacles on the planned path.

The coordinate system of a grid map takes the lower left corner of the map as the starting point and the upper right corner as the ending point. Each small grid in the map is a square, and the side length is uniformly set to 1. The passability of the grid is distinguished by colour: white grids indicate passable areas, allowing paths to cross; the black grid indicates the obstacle area, which is completely off-limits. According to this setting, the centre point coordinates of each grid can be calculated by Eq. (1), and the grid environment is shown in Figure 1.

$$\begin{cases} x_i = 1^* \left(\mathrm{mod}\left(i, MM\right) - 0.5\right) \\ y_i = 1^* \left(NN + 0.5 - \mathrm{ceil}\left(i/NN\right)\right) \end{cases} \tag{1}$$

where x_i and y_i represent the abscissa and ordinate of the position of the ith grid, M and N, respectively, represent the vertical direction (number of rows) and the horizontal direction (number of columns) of the grid matrix, "mod" represents the modular operation function, and "ceil" represents an upward rounding function.

3. Basic algorithm

3.1. The ACO algorithm before improvement

ACO algorithm is a typical representative of meta-heuristic algorithms, inspired by the collaborative behaviour of ant colonies foraging in nature [39]. Research has found that when ants search for food, they release pheromones along the way, forming chemical traces that can be perceived by other ants. The

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

paths passed multiple times become more attractive due to the accumulation of pheromones, constituting a positive feedback effect. The natural volatilisation of pheromones maintains a dynamic balance, enabling the ant colony to strike a balance between exploring new routes and optimising existing ones. ACO algorithm simulates this process by constructing solutions through artificial ants and updating pheromones in iterations to optimise the objective function.

In the ACO algorithm, each ant selects the next path based on the pheromone concentration and heuristic information. The probability that ant k chooses to go to node j at node i is as follows:

$$P_{ij}^{k}(t) = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\eta_{ij}(t)\right]^{\beta}}{\sum_{j \in \text{allowed}_k} \left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\eta_{ij}(t)\right]^{\beta}}, & \text{if } j \in \text{allowed}_k \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where α serves as the pheromone heuristic factor and is used to adjust the weight of pheromone concentration in the decision-making process; β, as an expectation heuristic factor, influences the expectation of ants in choosing paths; $\eta_{ij}(t)$ represents the heuristic information from the ith grid to the jth grid at time t, which is calculated by Eq. (3); and $allowed_k$ includes all the subsequent nodes that ant k can reach during its path exploration process:

$$\eta_{ij}(t) = \frac{1}{d_{ij}} \tag{3}$$

When ants complete a round of search iterations, the update of pheromones is indispensable. At the next moment, the pheromone concentration on the path will be recalculated and adjusted according to a specific update equation. The specific update equation is as follows:

$$\tau_{ij}(t+1) = (1-\rho)\,\tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{4}$$

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{M} \Delta\tau_{ij}^{k}(t) \tag{5}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{1}{L_k} \\ 0 \end{cases} \tag{6}$$

where ρ is the pheromone volatilisation factor, L_k is used to represent the length of the path walked by the kth ant, and $\Delta\tau_{ij}(t)$ represents the total release of pheromones on the path from the ith grid to the jth grid.

3.2. Bresenham algorithm

The Bresenham algorithm is a classic computer graphics algorithm used to draw straight lines on discrete grids, such as pixel grids. It was proposed by Jack E. Bresenham in 1962 and has been widely used due to its high efficiency and simplicity. The prominent feature of this algorithm is that it uses integer operations and avoids floating-point operations, making it highly suitable for computing environments with limited resources.

The core of the Bresenham algorithm is to select the pixel closest to the true line by controlling the error. Given a starting point (x_0, y_0) and an endpoint (x_n, y_n), the algorithm decides at each step whether to move horizontally or diagonally to approximate the line path. As shown in Figure 2.

Assume a starting point (x_0, y_0) to an endpoint (x_n, y_n), where $x_0 < x_n$, $y_0 < y_n$, and the slope is calculated by Eqs. (7), (8), and (9):

$$m = \frac{\Delta y}{\Delta x} \leq 1 \tag{7}$$

$$\Delta x = x_n - x_0 \tag{8}$$

$$\Delta y = y_n - y_0 \tag{9}$$

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

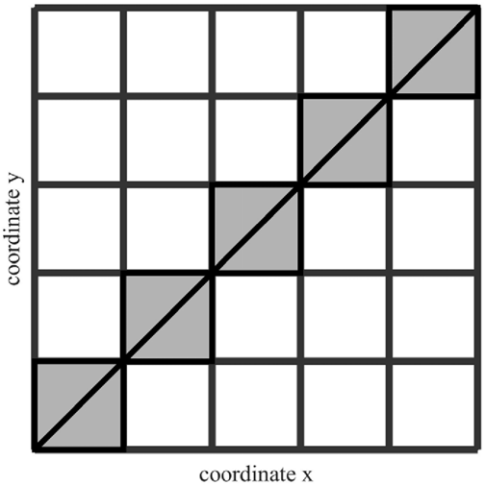


Figure 2. *Bresenham straight line.*

First, the error term e is initialised, with the calculation equation as shown below:

$$e = 2\Delta y - \Delta x \tag{10}$$

Next, the current pixel (x, y) is drawn. If $e < 0$, node $(x + 1, y)$ is selected and the error term e is updated using Eq. (11); if $e > 0$, node $(x + 1, y + 1)$ is selected and the error term e is updated using Eq. (12):

$$e = e + 2\Delta y \tag{11}$$

$$e = e + 2\,(\Delta y - \Delta x) \tag{12}$$

Finally, this iteration process is repeated until $x = x_n$. In this way, the Bresenham algorithm ensures that the generated pixel path visually approximates the true line while maintaining efficiency.

4. The improved ACO algorithm

4.1. Bresenham pheromone enhancement strategy

In the traditional ACO algorithm, the pheromone matrix is typically initialised uniformly, which may lead to low search efficiency in complex environments. To improve search efficiency, the Bresenham algorithm can be used to generate a path from the starting point to the endpoint that closely approximates a straight line, and the pheromone concentration can be enhanced on the edges along this path. This way, ants will prioritise exploring along the approximate straight-line direction in the early stages of the search, reducing ineffective searches and accelerating the algorithm's convergence speed.

First, use the Bresenham algorithm to generate a raster path between the starting point and the endpoint that closely approximates a straight line. Then, enhance the pheromone concentration of this path according to Eq. (13), as shown in Figure 3.

$$\tau_{ij} = \tau_0 \cdot D \tag{13}$$

where τ_0 is the initial pheromone value of the grid cell and D is the enhancement factor, typically $D > 1$, used to increase the pheromone concentration on the Bresenham straight-line path.

Initialising pheromones using the Bresenham algorithm effectively guides ants to select paths close to a straight line in the early stages of the search. This method significantly improves the search efficiency

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

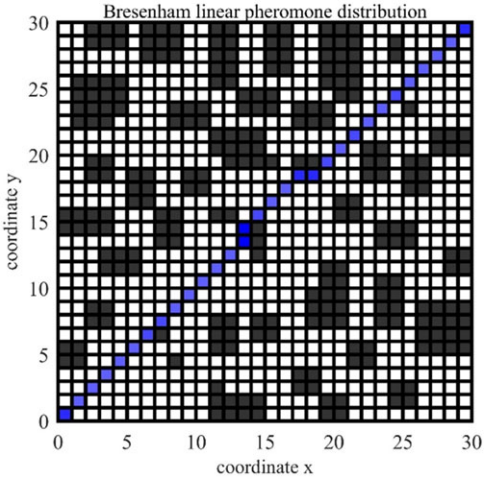


Figure 3. *Bresenham straight line.*

and convergence speed of the ACO algorithm, particularly in the field of path planning. By combining the simplicity and efficiency of the Bresenham algorithm with the global optimisation capabilities of ACO, this strategy demonstrates strong potential in practical applications.

4.2. Introduction of the direction continuity factor

The traditional ACO algorithm guides ants to find optimal paths using pheromones and heuristic information. However, in complex environments, the generated paths may contain numerous sharp turns or unsmooth trajectories, particularly in applications like robot path planning, which increases the difficulty and cost of practical execution.

To improve path smoothness, this study introduces a direction continuity factor to encourage ants to prioritise paths consistent with the previous movement direction. This approach quantifies directional consistency and adjusts the state transition probability, resulting in paths that better meet continuity and smoothness requirements, reducing abrupt turns.

The core idea of the direction continuity factor is to leverage the relationship between the ant's current movement direction and the direction of candidate path nodes, increasing the probability of selecting paths aligned with the previous direction. The specific implementation is based on the following principle: if the ant's previous direction is close to the next direction (i.e., the angle between them is small), the path is considered more optimal. Directional similarity is calculated as the ratio of the vector dot product to the vector magnitudes, which is then converted into a probability weight. The improved state transition probability equation is as follows:

$$P(i,j) = \frac{\left[\tau(i,j)\right]^{\alpha} \cdot \left[\eta(j)\right]^{\beta} \cdot \left[s_{ij}\right]^{\omega}}{\sum_{k \in \text{allowed}} \left[\tau(i,k)\right]^{\alpha} \cdot \left[\eta(k)\right]^{\beta} \cdot \left[s_{ik}\right]^{\omega}} \tag{14}$$

where s_{ij} is the direction continuity factor, calculated by Eq. (15) and ω is the direction continuity weight parameter, used to adjust the strength of the direction continuity factor's influence.

$$s_{ij} = \exp\left(\frac{\cos\theta}{2}\right) \tag{15}$$

where $cos\theta$ is the direction cosine, representing directional similarity using the cosine of the angle between vectors, calculated by Eqs. (16), (17), and (18).

$$\cos\theta = \frac{\mathbf{v}_{prev} \cdot \mathbf{v}_{current}}{|\mathbf{v}_{prev}| \cdot |\mathbf{v}_{current}|} \tag{16}$$

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

where v_{prev} is the previous step direction vector, calculated by Eq. (17), $v_{current}$ is the current step direction vector, calculated by Eq. (18), $v_{prev} \cdot v_{current}$ is the vector dot product, and $|v_{prev}| \cdot |v_{current}|$ is the product of the vector magnitudes.

$$\mathbf{v}_{prev} = \left(x_{current} - x_{prev}, y_{current} - y_{prev}\right) \tag{17}$$

$$\mathbf{v}_{current} = \left(x_{next} - x_{current}, y_{next} - y_{current}\right) \tag{18}$$

where x_{prev} and v_{prev} are the x-coordinate and y-coordinate of the node preceding the current node; x_{next} and v_{next} are the x-coordinate and y-coordinate of the next selectable node from the current node; and $x_{current}$ and $v_{current}$ are the x-coordinate and y-coordinate of the current node.

By introducing a direction continuity factor to quantify the consistency of movement direction, the state transition probability of the traditional ACO algorithm is modified, significantly improving path smoothness. This method effectively reduces unnecessary turns in complex environments, making the generated paths better aligned with practical requirements.

4.3. Pheromone update affected by path curvature

The traditional ACO algorithm is modelled on the foraging behaviour of ant colonies, guiding the search process through a dynamic pheromone updating mechanism. Specifically, ants deposit pheromones along the paths they traverse, and the concentration of pheromones increases on shorter or more frequently chosen paths. This, in turn, attracts more ants to these paths, forming a positive feedback loop that iteratively optimises the solution. The algorithm typically uses path length or cost as a single optimisation objective, gradually approaching the global optimum through repeated iterations. However, in practical applications such as robot path planning or vehicle navigation, focusing solely on path length may result in routes with numerous sharp turns or high-curvature segments. These features not only increase execution difficulty but can also lead to greater energy consumption and reduced system stability, posing potential safety risks. Therefore, it is necessary to incorporate geometric constraints such as path smoothness or curvature into the optimisation objective framework.

The pheromone update influenced by path curvature is an improvement to the traditional ACO algorithm. By introducing the factor of path curvature into the pheromone update process, ants are encouraged to choose paths with lower curvature (i.e., smoother paths), thereby enhancing the quality and practicality of the resulting paths. The underlying principle is as follows: the curvature is quantified by calculating the angle formed by each set of three consecutive points along the path. The smaller the curvature, the smoother the path. More pheromone is deposited on paths with lower curvature to guide subsequent ants to preferentially select these paths. The improved pheromone update equation is shown in Eq. (19):

$$\Delta\tau_{ij}^{k} = \frac{1}{L_k} \cdot \left(1 + \frac{C}{2}\right) \tag{19}$$

where the path curvature influences the value of C; the smaller the curvature and the smoother the path, the larger the value of C, which is calculated according to Eq. (20).

For three consecutive points P_{i-1}, P_i, and P_{i+1} on a path, the vectors $v_1 = P_i - P_{i-1}$ and $v_2 = P_{i+1} - P_i$ are calculated, and the sum of the absolute values of the cosine of the angles between these vectors is then computed:

$$C = \sum_{i=2}^{n-1} \left| \frac{v_1 \cdot v_2}{|v_1| \cdot |v_2|} \right| \tag{20}$$

By introducing a path curvature factor, the pheromone update becomes associated with the curvature of the path. The straighter the path, the higher the curvature value (C-value), and consequently, the greater the pheromone increment. This mechanism improves upon the traditional pheromone updating method by guiding the algorithm to favour paths with lower curvature and smoother trajectories.

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

As a result, it significantly enhances the practicality and adaptability of the generated paths in complex scenarios such as robot navigation.

4.4. Path smoothing strategy based on the spring model

The core objective of robot path planning is to find a path that not only avoids obstacles but also maintains smoothness and efficiency. However, this task becomes particularly challenging in environments with dense obstacles. In such scenarios, the path planning algorithm must navigate around obstacles while minimising the overall path length and curvature, all the while ensuring the path remains feasible. For example, in an environment filled with obstacles, the robot may need to make frequent turns or detours, which can increase the path length and introduce sharp turns or unnecessary deviations. Traditional path planning algorithms often struggle to balance obstacle avoidance, smoothness, and efficiency simultaneously, resulting in suboptimal paths.

To address the aforementioned limitations, many researchers in recent years have proposed improvements to path planning algorithms by incorporating various path smoothing techniques. Poy et al. introduced an enhanced particle swarm optimisation (EPSO) algorithm, which combines a novel path planning scheme with a cubic Bézier curve-based trajectory smoothing method. This approach reduces the tortuosity of the generated path and avoids frequent turning [40]. Mohaghegh et al. proposed a smoothing particle swarm optimisation with improved potential field (SPSO-IPF), which integrates PSO with a potential field (PF) method to produce smoother and safer paths [41]. Cao et al. developed a multi-objective path smoothing algorithm based on node adjustment and ramp smoothing. Their method filters and moves nodes in the original path using a path point adjustment algorithm. Then, B-spline curves are used to smooth discontinuities at turns, and control points are adjusted to ensure obstacle avoidance while satisfying the maximum curvature constraint [42].

Existing path-smoothing methods proposed by scholars have achieved significant results in improving the smoothness of paths generated by meta-heuristic algorithms, reducing path inflection points, and avoiding frequent turns. However, these methods still suffer from limitations in local optimisation. To address the shortcomings of traditional path smoothing strategies in global smoothness control and redundant node elimination, this study innovatively introduces the spring model into the field of path planning, proposing a path optimisation method based on dynamic principles. The core idea of the spring model is to treat nodes on the path as mass points connected by virtual springs between adjacent nodes. By simulating the stretching and compressive forces of the springs, the node positions are dynamically adjusted, ultimately achieving continuous path optimisation and autonomous elimination of redundant nodes. The specific operations are as follows.

For each node on the path (except the fixed start and endpoints), the model calculates the resultant force acting on it.

First, the spring force between node i and its previous node i-1, as well as the spring force between node i and its next node $i+1$, is calculated as follows:

$$\vec{F}_1 = \left(\|x_i - x_{i-1}\| - \frac{1}{2} \right) \cdot \frac{x_{i-1} - x_i}{\|x_{i-1} - x_i\|} \tag{21}$$

$$\vec{F}_2 = \left(\|x_{i+1} - x_i\| - \frac{1}{2} \right) \cdot \frac{x_{i+1} - x_i}{\|x_{i+1} - x_i\|} \tag{22}$$

where x_i represents the coordinates of the current node, x_{i-1} represents the coordinates of the previous node of node i, and x_{i+1} represents the coordinates of the next node of node i.

Then, the smoothing force and the direct connection force are calculated as follows:

$$\vec{F}_3 = \frac{1}{2} \left(x_{i+1} - 2x_i + x_{i-1} \right) \tag{23}$$

$$\vec{F}_4 = \frac{1}{2} \left(\frac{x_{i+1} + x_{i-1}}{2} - x_i \right) \tag{24}$$

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

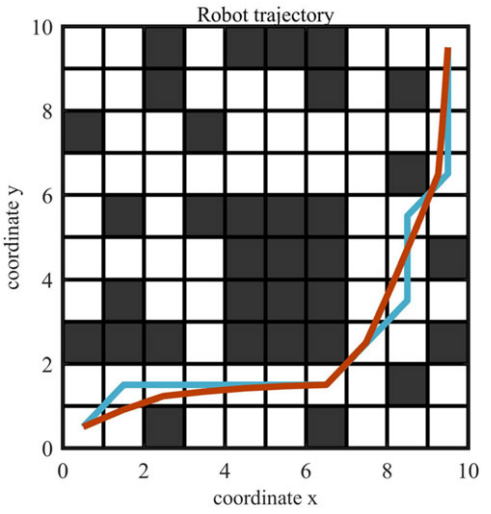


Figure 4. *Path optimisation graph based on spring model.*

Finally, the sum of all the forces is calculated as follows:

$$\vec{F}_{total} = \vec{F}_1 + \vec{F}_2 + \vec{F}_3 + \vec{F}_4 \tag{25}$$

The calculated resultant force acting on node i is substituted into the position update Eq. (26), and the new coordinates of the node are obtained:

$$x_i^{new} = x_i + \frac{1}{20} \cdot \vec{F}_{total} \tag{26}$$

The spring force and direct connection force encourage nodes to move toward a straighter path, significantly reducing path length, while the smoothing force reduces sharp corners in the path. Through the above formula calculations and multiple iterations, the nodes on the path are gradually adjusted, ultimately resulting in a shorter and smoother path, generating a more natural trajectory. As shown in Figure 4, the blue path represents the original path, and the red path represents the smoothed path.

The path smoothing strategy based on the spring model optimises the paths generated by the ant colony algorithm through the synergistic effects of spring forces, smoothing forces, and direct connection forces. The optimisation process integrates iterative updates and collision detection, ensuring that the path is both shortened and smoothed while avoiding obstacles. This method is highly practical in robot path planning, effectively enhancing path quality.

4.5. Steps and processes of the BCD-ACO algorithm

The enhanced algorithm begins its process by leveraging the Bresenham algorithm to establish initial pheromone trails strategically laid out to steer ants along a search trajectory that closely resembles a straight line. This initialisation phase ensures that the ants‘ early exploration efforts are directed efficiently toward their intended destination, setting a solid foundation for subsequent refinements. Following this, the algorithm focuses on improving the smoothness of the path through a series of repeated search iterations, integrating a direction continuity factor, a mechanism designed to maintain consistent movement directions. This factor reduces sudden shifts or jagged transitions in the ants‘ paths, resulting in a more cohesive and fluid route.

Next, the algorithm adjusts the pheromone concentrations dynamically, considering the curvature of the paths that emerge during the ants‘ exploration. By favouring paths with gentler curves and fewer sharp bends, this pheromone update process encourages the selection of smoother trajectories, enhancing the overall quality and navigability of the routes chosen by the ants. Finally, the algorithm employs a

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

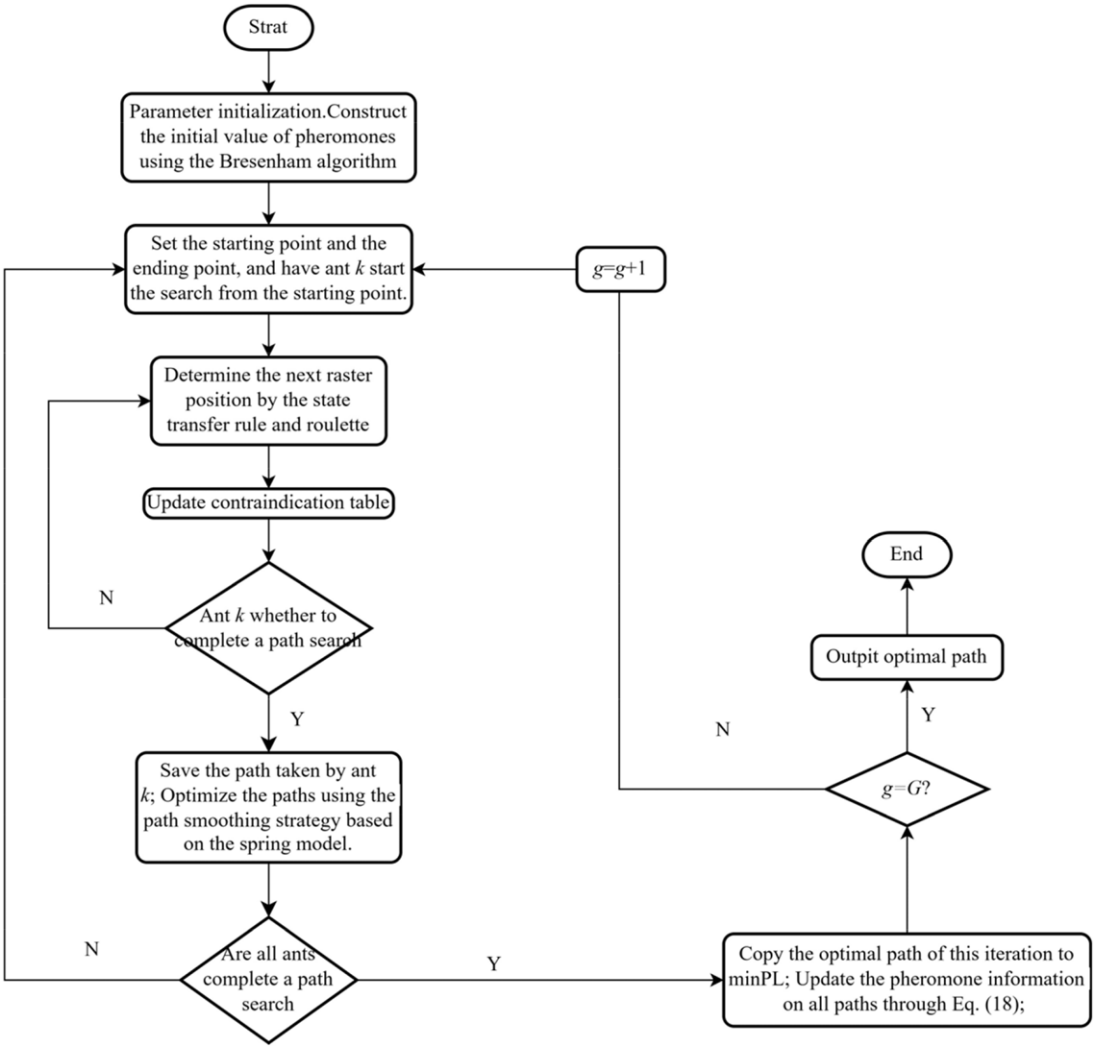


Figure 5. *Flowchart of the BCD-ACO algorithm.*

sophisticated path-smoothing strategy rooted in a spring model, which simulates elastic forces to refine the path further. This approach minimises irregularities and sharp turns, optimising the route for efficiency and safety while validating its practicality. Through this multi-step methodology, the algorithm ultimately produces a highly refined path that is not only efficient in terms of exploration but also smooth and secure for traversal. The flow chart of the BCD-ACO algorithm is shown in Figure 5. The pseudocode of the BCD-ACO algorithm is shown in Table I.

5. Experiment and analysis

In this study, MATLAB R2024b is used as the simulation platform to construct various grid map scenarios for mobile robots. Through simulation experiments, the overall performance of the improved ACO algorithm is systematically evaluated in complex path planning tasks. To fully verify the superiority and effectiveness of the proposed algorithm, both gradient experiments and comparative experiments are designed and implemented for comprehensive performance assessment. Specifically, the gradient experiments progressively introduce each improvement module of the algorithm to analyse the contribution

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

Table I. *Pseudo-code of BCD-ACO algorithm.*

Pseudo-code of BCD-ACO algorithm
01. Construct the environment model;
02. Initialise basic parameters;
03. Initialise pheromone with the Bresenham algorithm and enhance concentration via Eq. (12);
04. **For** g to G **do**
05. **For** k to M **do**
06. Place ant k at the starting point S;
07. **While** ant k has not reached the endpoint E **do**
08. Select the next grid position using Eq. (13);
09. **End While**
10. Save the path taken by ant k;
11. Apply the path smoothing strategy to optimise the path;
12. **End For**
13. Copy the best path of this iteration to *minPL*;
14. Update the pheromone on all paths using Eq. (18);
15. **End For**
16. Output the optimal path.

of each strategy and to explore the specific impact of each enhancement on the algorithm's performance. Meanwhile, the comparative experiments directly compare the proposed algorithm with other improved ACO algorithms and alternative path planning algorithms from existing research, quantifying improvements and advantages in key performance indicators such as convergence speed, path smoothness, and number of iterations. The simulation results demonstrate that each improvement module contributes positively to key performance metrics, leading to faster convergence, smoother trajectories, and reduced computational overhead across diverse grid scenarios. These findings confirm the proposed algorithm's strong potential for enhancing autonomous mobile robot navigation and underscore its applicability to real-world path planning challenges.

5.1. Parameter tuning

In the BCD-ACO algorithm, parameter tuning is crucial for performance optimisation. By optimising parameter settings, the algorithm can achieve a dynamic balance between exploration and exploitation, which helps enhance solution quality and search efficiency. Moreover, carefully tuned parameters significantly improve the algorithm's stability and reliability across multiple runs, reducing performance fluctuations caused by randomness. This enables the BCD-ACO algorithm to exhibit greater adaptability and robustness when addressing complex optimisation problems, providing strong support for efficient problem-solving in real-world applications. This study uses the grid environment from ref. [36] (as shown in Figure 6). A controlled variable method is applied to tune the parameters α, β, and ω sequentially. For each parameter value, 10 experiments are conducted to analyse its impact on key performance metrics, including the shortest path length (PL_{min}), longest path length (PL_{max}), average path length (PL_{mean}), standard deviation of path length (PL_{std}), best iteration count (I_{min}), worst iteration count (I_{max}), average iteration count (I_{mean}), and standard deviation of iteration count (I_{std}).

5.1.1. α parameter tuning

In the ACO algorithm, the parameter α functions as the pheromone influence coefficient, directly governing how strongly the existing pheromone trails steer each ant's decision-making when selecting the next node. By increasing α, one amplifies the weight of pheromone concentration in the probabilistic

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

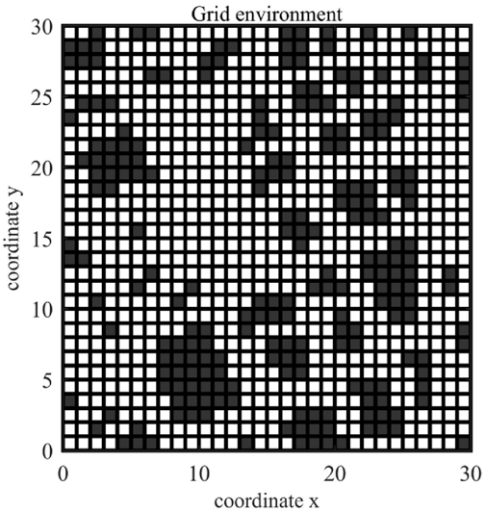


Figure 6. *Tuning environment diagra.*

transition rule, thereby encouraging ants to favour routes that have been heavily traversed in previous iterations. Conversely, a lower α reduces this bias, allowing heuristic information, such as distance or cost, to play a more prominent role. Since α fundamentally balances exploration and exploitation, its proper adjustment is essential for achieving rapid convergence without premature stagnation. For this reason, our tuning methodology is anchored first and foremost on carefully selecting the optimal value of α before proceeding to refine other algorithmic parameters. The tested range for α is {1, 2, 3, 4, 5}, while the other parameters are fixed at $K = 100$, $M = 50$, $\beta = 10$, $\rho = 0.3$, $\omega = 5$, and $D = 2$. The experimental results are shown in Figure 7.

Figure 7 illustrates the relationship between the α parameter and the variability of the algorithm's convergence behaviour. As the plot indicates, setting α to 4 yields the smallest standard deviation in the number of iterations required for convergence, suggesting a slightly more consistent performance across repeated runs. However, this reduction in variability is very slight when compared to the results obtained with $\alpha = 3$, and the difference is practically negligible in the context of our simulation experiments. More importantly, when α is set to 3, all other key performance metrics, including average convergence speed, total path length, and computational time, reach their minimum values. This combination of fast convergence, smooth path trajectories, and low-resource consumption demonstrates that $\alpha = 3$ provides the best overall balance of performance characteristics. Consequently, we adopt $\alpha = 3$ for the BCD-ACO algorithm to maximise search efficiency and ensure robust, high-quality path planning results.

5.1.2. β parameter tuning

In the ACO algorithm, the parameter β acts as the heuristic information influence coefficient, determining the weight given to the inverse distance (or other heuristic measure) when an ant selects its next node. Increasing β biases the algorithm toward shorter paths, while decreasing β strengthens the role of pheromone trails, promoting broader exploration. Therefore, once the optimal value of α has been established, we immediately tune β to further balance exploration and exploitation. The range of values tested for β is {8, 10, 12, 14, 16, 18, 20, 22}, with all other parameters fixed at: $K = 100$, $M = 50$, $\alpha = 3$, $\rho = 0.3$, $\omega = 5$, and $D = 2$. The experimental results are shown in Figure 8.

As shown in Figure 8, the minimum path length (PL_{min}) consistently reaches the lowest value of 41.46 across all tested values of parameter β. However, when β is set to 20, all performance metrics reach their lowest levels, indicating that the algorithm achieves the best overall performance under this condition. Therefore, the value of the parameter β is set to 20 to enhance the performance of the BCD-ACO algorithm, with the goal of reducing the energy consumption of the robot.

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

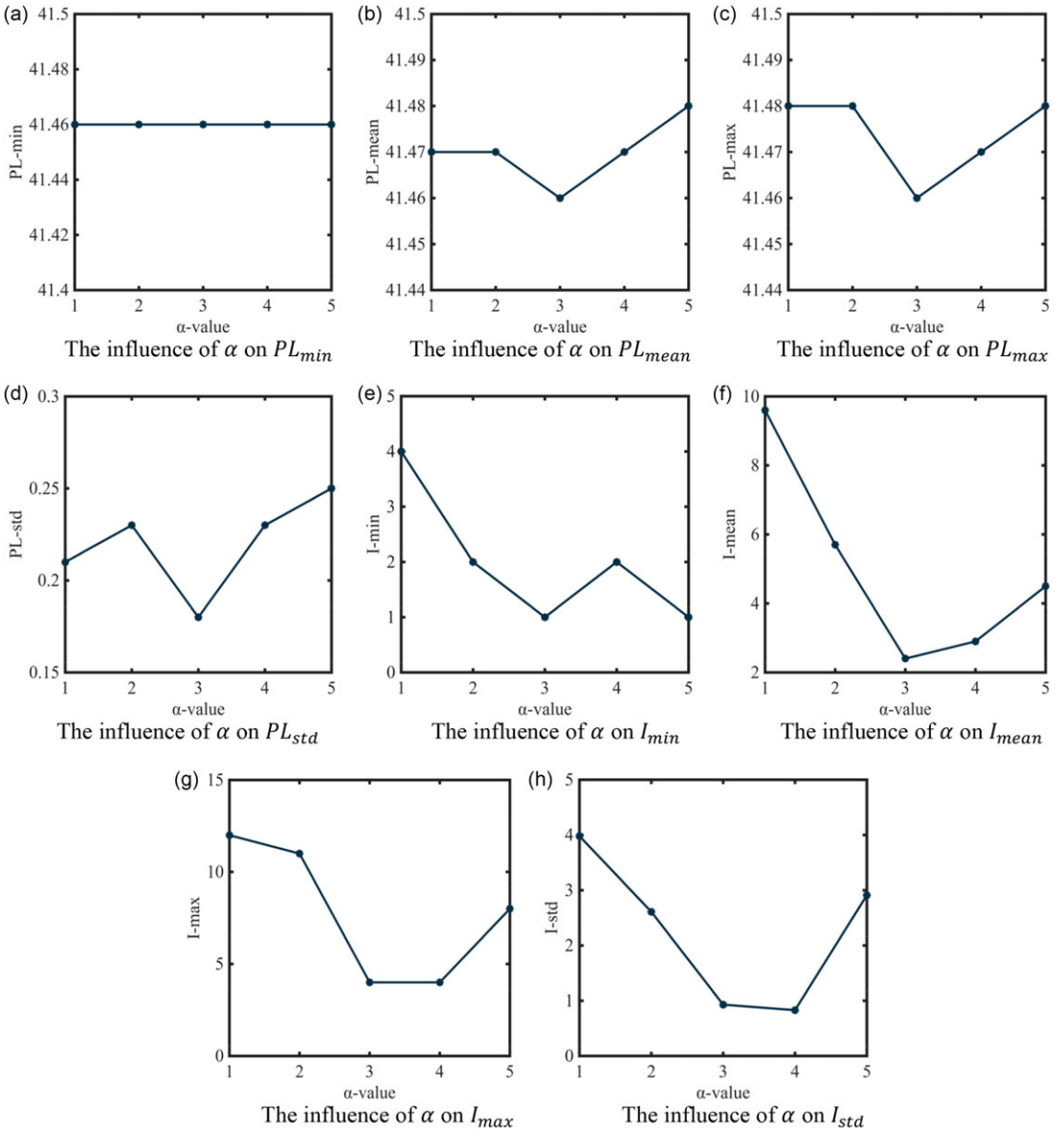


Figure 7. *The influence of different parameter* α *values on various indicators.*

5.1.3. ω *parameter tuning*

Parameter ω, which serves as the weight of the directional continuity factor, determines the strength of its influence. Tuning this parameter plays an important role in improving the smoothness of the paths generated by the algorithm. Therefore, it is necessary to optimise the value of ω .

The tested range for ω is $\{1, 2, 3, 4, 5\}$, with the other parameters fixed at: $K = 100$, $M = 50$, $\alpha = 3$, $\rho = 0.3$, $\beta = 20$, and $D = 2$. The experimental results are shown in Figure 9.

As shown in Figure 9, the minimum path length (PL_{min}) consistently reaches the lowest value of 41.46, and the maximum path length (PL_{max}) remains constant at 41.48 across all tested values of parameter ω. When ω is set to 3, the standard deviation of path length (PL_{std}) reaches its minimum; however, since PL_{std} remains within a narrow range of 0.23 to 0.27 regardless of the ω value, the impact of ω on this metric can be considered negligible. When ω is set to 5, all performance metrics except PL_{std} reach their

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

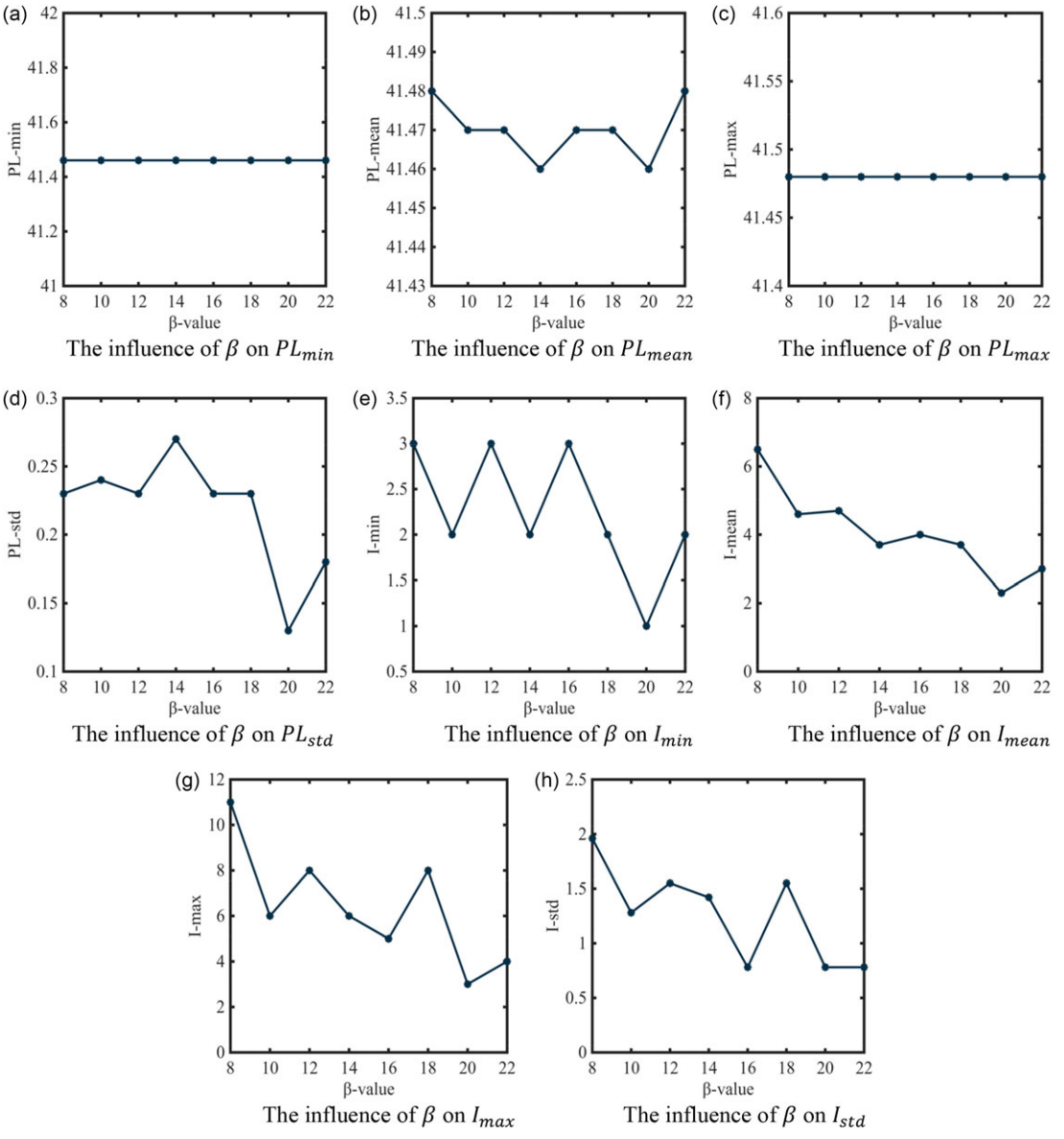


Figure 8. *The influence of different parameter β values on various indicators.*

lowest values, indicating that $\omega = 5$ offers the best overall performance. Therefore, the value of ω is set to 5 to enhance the smoothness of the paths generated by the BCD-ACO algorithm.

5.1.4. Final parameter value

Combined with the above parameter adjustment experiments, the values of each parameter in the BCD-ACO algorithm are shown in Table II.

5.2. Gradient experiment

In order to more precisely and quantitatively assess the individual contributions of each improvement strategy to the algorithm's performance, this paper designs and conducts a series of systematic gradient experiments in two different grid environments. The experiments employ a stepwise introduction

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

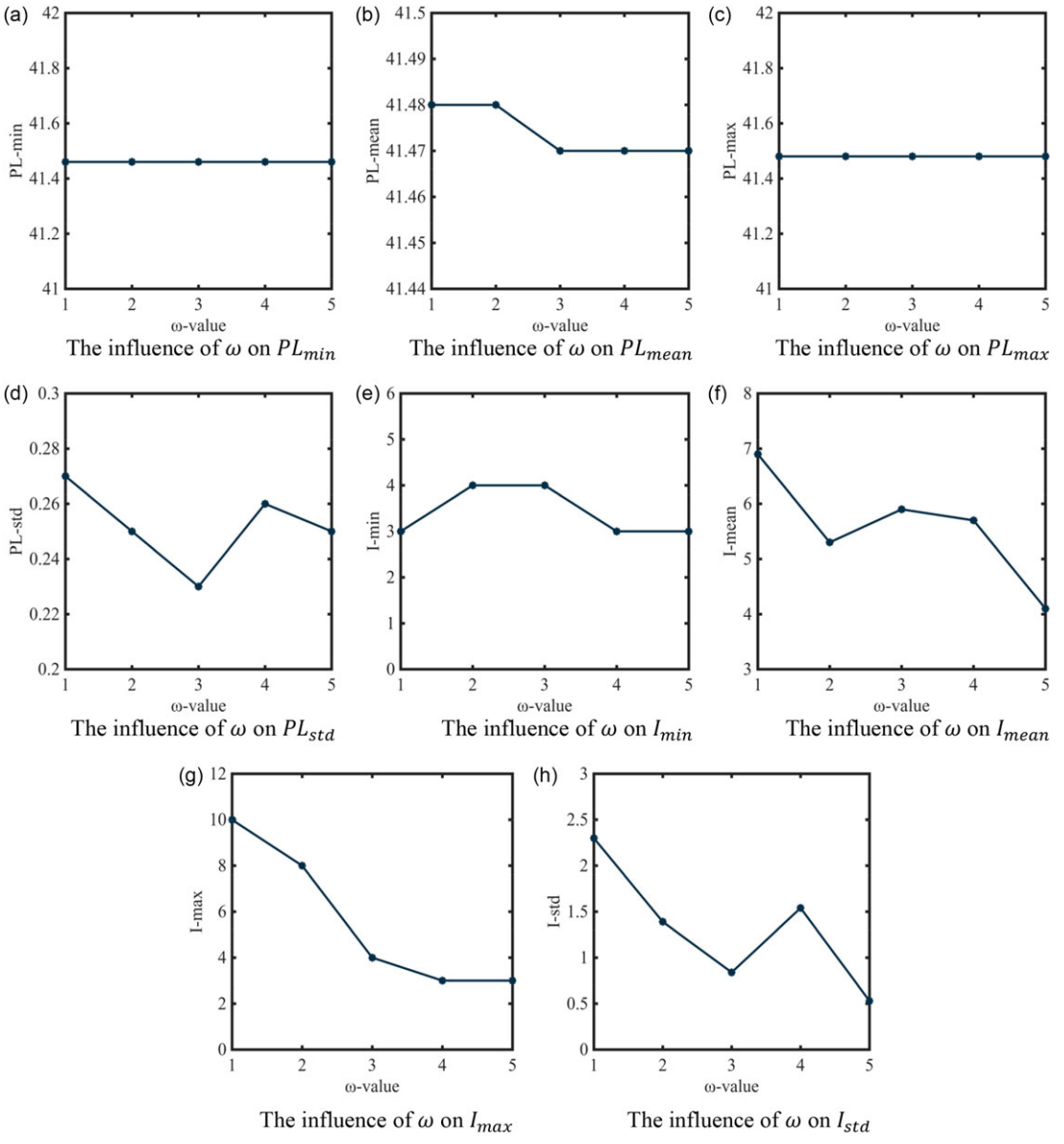


Figure 9. *The influence of different parameter ω values on various indicators.*

approach, incrementally adding the various improvement strategies proposed here to the classical ACO algorithm. Using this progressive method, we analyse and compare both the impact of each improvement strategy on algorithm performance and the synergistic effects arising from their combinations. The experimental results are shown in Figures 10 and 11. Here, ACO-1 denotes the classical ACO algorithm augmented with the Bresenham-based pheromone enhancement strategy; ACO-2 builds on ACO-1 by introducing a directional continuity factor; ACO-3 further extends ACO-2 by incorporating a path curvature-influenced pheromone update; and, finally, BCD-ACO adds to ACO-3 a spring model-based path smoothing strategy.

As shown by the results in Figure 10, the traditional ACO algorithm exhibits poor convergence, requiring approximately 60 iterations to converge. This indicates relatively low search efficiency and limited path optimisation capability in complex environments. In contrast, the ACO-1 algorithm, which

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

Table II. *Final parameter value table.*

Parameters	***K***	***M***	***α***	***β***	***ρ***	***ω***	***D***
Value	100	50	3	20	0.3	5	2

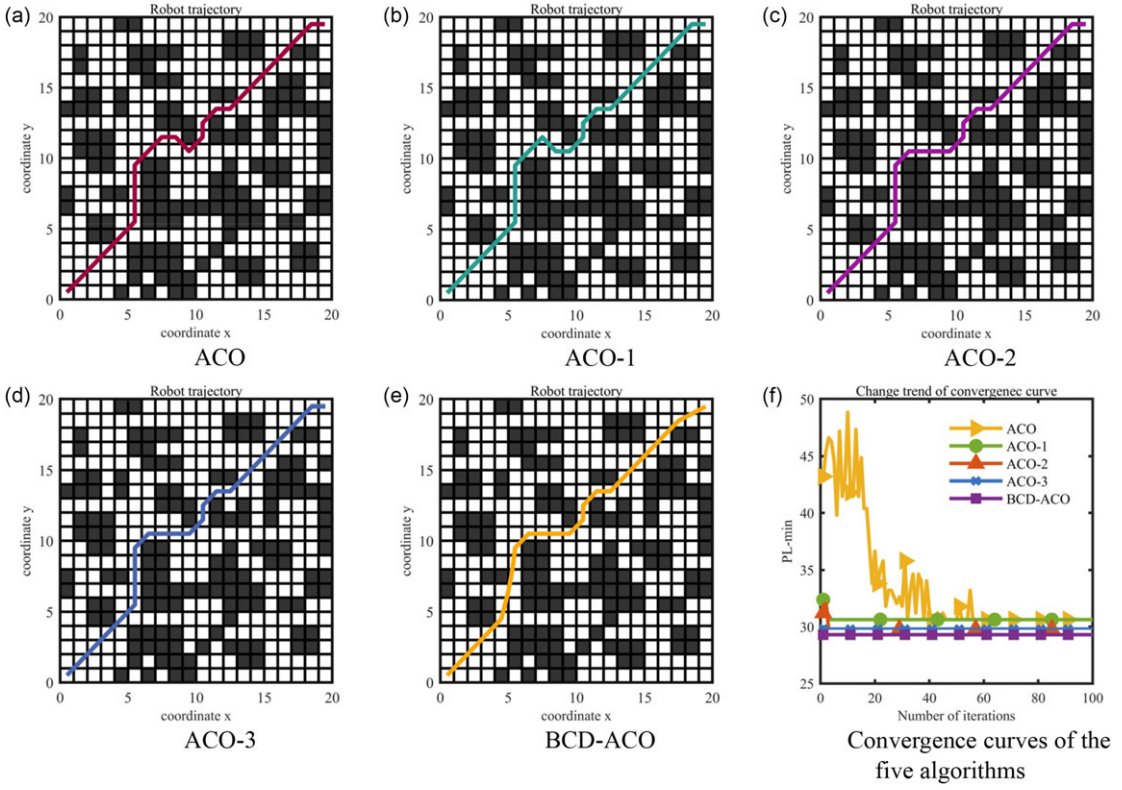


Figure 10. *Gradient experiment 1 comparison results.*

introduces the Bresenham pheromone enhancement strategy based on the traditional ACO, significantly improves convergence speed, converging in only three iterations and greatly increasing convergence efficiency. However, the paths generated by ACO-1 contain many sharp turns and lack smoothness, indicating that while this strategy improves search efficiency, it does not adequately account for the geometric characteristics of the path.

Building upon the ACO-1 algorithm, the introduction of the directional continuity factor in the ACO-2 algorithm effectively reduces the number of sharp turns in the path and improves its smoothness. This improvement demonstrates that the directional continuity factor can guide ants to preferentially select smoother paths during the search process, thereby enhancing the overall path quality.

Building upon the ACO-2 algorithm, the ACO-3 algorithm introduces pheromone updating influenced by path curvature, which further improves the convergence speed, achieving convergence in just one iteration. This result indicates that the pheromone update strategy based on path curvature dynamically adjusts the pheromone distribution, significantly enhancing the algorithm's global search capability and convergence efficiency.

Finally, based on ACO-3, the BCD-ACO algorithm further reduces path length and enhances overall path quality by incorporating a spring model-based path smoothing strategy.

The experimental results show that the Bresenham-based pheromone enhancement strategy endows the algorithm with highly efficient search capability, the directional continuity factor improves path smoothness, the curvature-influenced pheromone update further accelerates convergence, and the spring

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

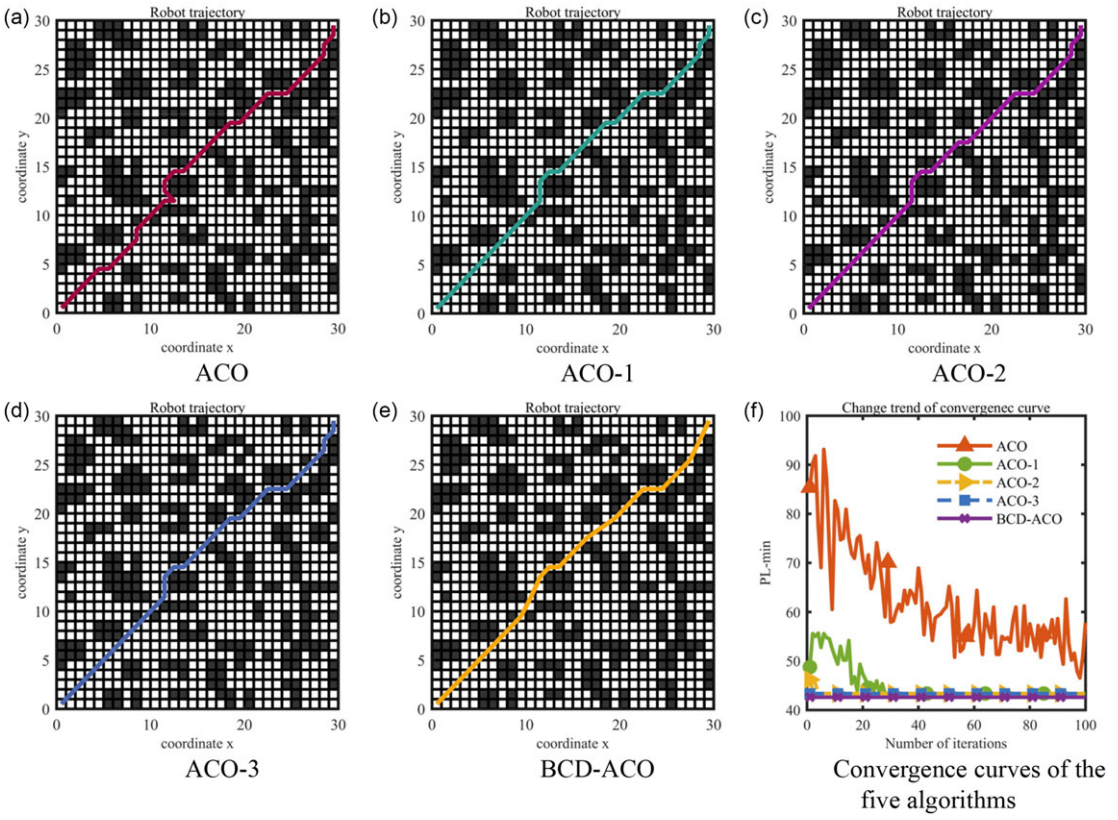


Figure 11. *Gradient experiment 2 comparison results.*

model-based path smoothing strategy achieves final optimisation in both path length and quality. The incremental introduction of each strategy demonstrates clear cumulative effects and synergy. By integrating all these improvements, the BCD-ACO algorithm excels in convergence speed, path smoothness, and overall path length.

To further validate the coordination and performance contributions of the improvement strategies in more complex environments, this study conducted a second gradient experiment in a 30×30 grid environment. Compared to the first experiment's 20×20 grid, the new environment is larger and more complex, posing stricter challenges to the algorithm's search efficiency and path optimisation capability.

The traditional ACO algorithm performs poorly in the 30×30 grid environment, struggling to converge and generating paths that are relatively long with numerous sharp turns. This reflects its limitations in complex environments, indicating that the traditional ACO algorithm lacks sufficient search capability and path optimisation ability when dealing with large-scale, complex grids, making it difficult to meet the demands of efficient path planning.

The ACO-1 algorithm, which incorporates the Bresenham pheromone enhancement strategy, significantly improves the algorithm's performance. The convergence speed is greatly increased, requiring only 30 iterations to converge, and the planned path length is noticeably shorter compared to the traditional ACO algorithm. This improvement demonstrates that the Bresenham pheromone enhancement strategy effectively optimises pheromone distribution, thereby enhancing the algorithm's search efficiency and path quality in complex environments.

Building upon the ACO-1 algorithm, the introduction of the directional continuity factor in the ACO-2 algorithm further enhances performance, with convergence accelerating to just four iterations.

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

The addition of the directional continuity factor effectively reduces sharp turns in the path and improves its smoothness, indicating that this strategy guides ants to choose better path directions in complex environments, thereby balancing convergence speed and path quality.

Based on the ACO-2 algorithm, the ACO-3 algorithm incorporates pheromone updating influenced by path curvature, demonstrating even more outstanding performance by converging to the optimal solution in just one iteration. This result shows that the pheromone update strategy influenced by path curvature dynamically adjusts pheromone distribution, further strengthening the algorithm's global search capability and convergence efficiency, enabling it to quickly adapt to the challenges of complex environments.

Finally, based on the ACO-3 algorithm, the BCD-ACO algorithm, which incorporates a spring model-based path smoothing strategy, further compresses the path length, significantly enhances overall path quality, and demonstrates its unique advantages in path optimisation.

Overall, the second series of stepwise experiments confirms the effectiveness and synergy of each improvement strategy in larger, more complex grid environments. The Bresenham-based pheromone enhancement lays the groundwork for efficient search; the directional continuity factor improves path smoothness; the curvature-influenced pheromone update accelerates convergence; and the spring model-based path smoothing strategy further shortens the path length. As the final optimised version, the BCD-ACO algorithm integrates all of these enhancements and, in terms of convergence speed, path smoothness, and path length, significantly outperforms the traditional ACO algorithm.

5.3. Comparative experiment

To comprehensively validate the superior performance of the proposed BCD-ACO algorithm in path planning tasks, this study designs a series of comparative experiments that systematically contrast BCD-ACO with multiple existing path planning algorithms. The experiments include, as ant-colony-based benchmarks, the IAACO algorithm from ref. [31], the PS-ACO algorithm from ref. [32], the AIACSE algorithm from ref. [33], the IACO algorithm from ref. [34], the MRCACO algorithm from ref. [35], the MsAACO algorithm from ref. [36], and the classical ACO algorithm, in order to assess BCD-ACO's improvements in convergence speed, path quality, and adaptability to complex environments. Additionally, to further verify BCD-ACO's competitiveness among a broader set of path planning methods, the study incorporates Q-learning, breadth-first search (BFS), depth-first search (DFS), the A* algorithm, and Dijkstra's algorithm as baseline comparisons. These algorithms span heuristic search, reinforcement learning, and graph-search paradigms, representing diverse technical approaches within the path planning field. All tests are conducted on the grid maps used in ref. [36]; performance data for IAACO, PS-ACO, AIACSE, MRCACO, IACO, and MsAACO are cited directly from their original publications to ensure fairness and consistency. Each algorithm is evaluated under uniform conditions with 20 repeated runs per environment, comparing metrics such as path length, convergence speed, and number of turning points. By benchmarking across grid environments of varying scale and complexity, the experiments aim to elucidate the performance advantages of BCD-ACO and reveal its potential value in convergence efficiency, path length reduction, smoothness, and environmental adaptability.

5.3.1. Comparative experiment 1

Comparative experiment 1 conducts a comprehensive comparison of the BCD-ACO algorithm with multiple existing algorithms in a 20×20 grid environment. The comparison includes the IAACO algorithm from ref. [31], the MsAACO algorithm from ref. [36], the traditional ACO algorithm, DFS, BFS, Q-learning, A* algorithm, and Dijkstra's algorithm. The experimental results are presented in Figures 12 and 13, as well as in Table III.

The experimental results shown in Figure 12 clearly demonstrate that the BCD-ACO algorithm excels in path smoothness. Compared to other improved ACO algorithms in the literature, the paths planned by

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

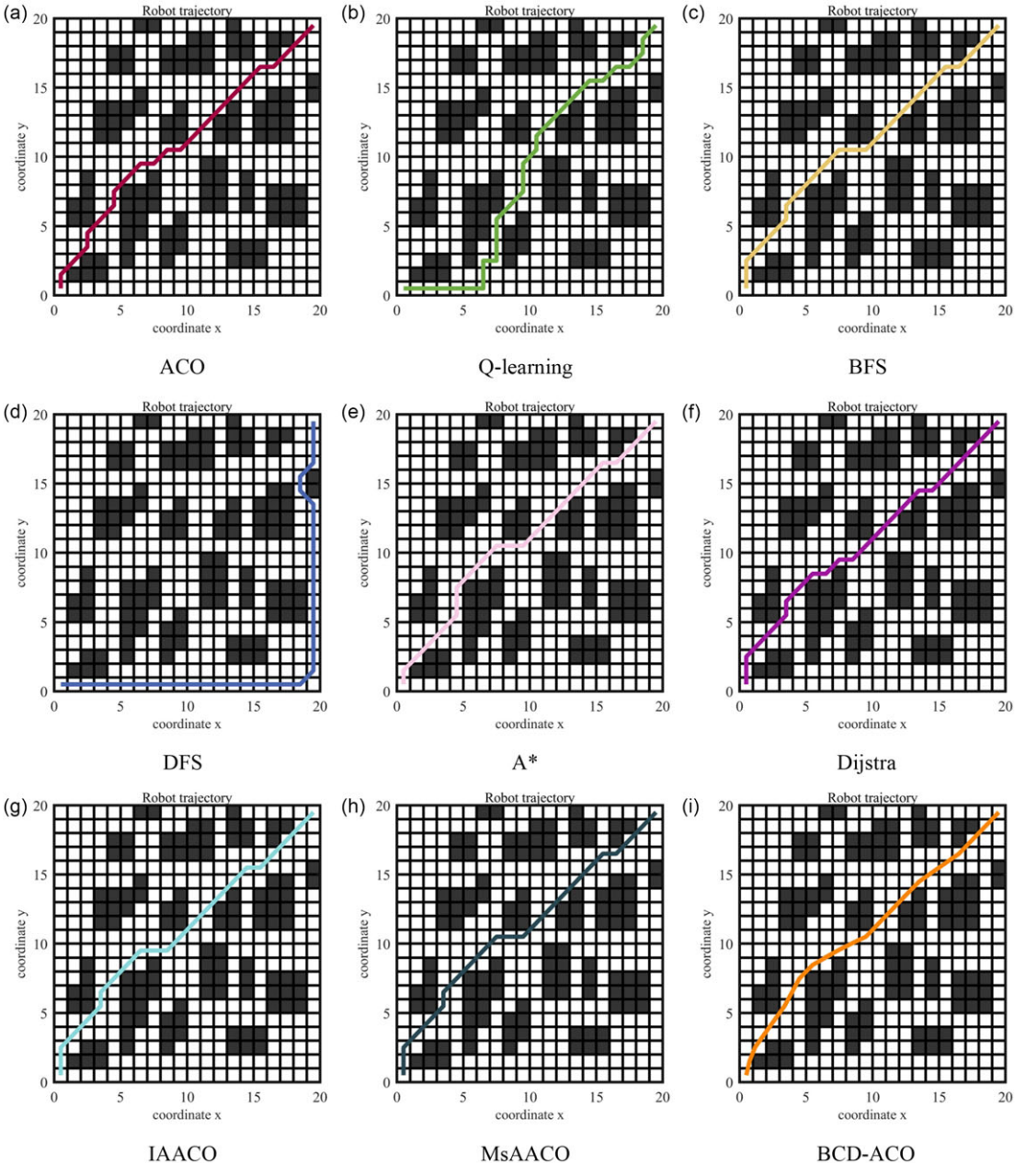


Figure 12. *Comparative experiment 1 path comparison diagram.*

BCD-ACO are significantly smoother. This improvement not only reduces sharp turns and unnecessary fluctuations but also enhances the stability and safety of the robot during movement.

As clearly shown in Figure 13, the proposed BCD-ACO algorithm exhibits a marked advantage in convergence performance. Compared to the traditional ACO algorithm, IAACO, and MsAACO algorithms, BCD-ACO not only achieves a substantial increase in convergence speed but also demonstrates significantly higher efficiency. Experimental data indicate that BCD-ACO consistently converges to the global optimal solution after just one iteration, fully reflecting its exceptional optimisation efficiency and operational stability. This outstanding performance not only considerably reduces overall computation

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

Table III. *Results of comparative experiment 1.*

	Path length			Iterations		
Algorithm	PL_{min}	PL_{mean}	PL_{std}	I_{min}	I_{mean}	**Turns times**
ACO	28.63	30.37	2.53	19	42.57	11
Q-learning	29.21	30.30	0.78	–	–	14
BFS	28.63	28.63	0	–	–	7
DFS	38.24	38.24	0	–	–	6
A*	28.63	28.63	0	–	–	7
Dijstra	28.63	28.63	0	–	–	9
IAACO	28.63	29.51	0.75	3	4	7
MsAACO	28.63	28.63	0	5	5.33	7
BCD-ACO	27.51	27.63	0.14	1	2.25	7

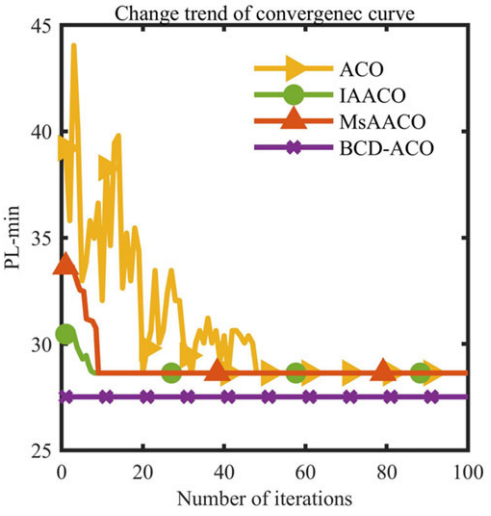


Figure 13. *Comparison of convergence curves of improved ant colony optimisation algorithms in comparative experiment 1.*

time and lowers computational resource consumption but also displays enhanced robustness and reliability when tackling various complex optimisation problems, further verifying the algorithm's broad applicability in real-world scenarios.

5.3.2. Comparative experiment 2

Comparative experiment 2 aims to systematically evaluate the performance of the BCD-ACO algorithm in a 30×30 grid environment and comprehensively compare it with several mainstream algorithms. Specifically, this experiment selects the PS-ACO algorithm proposed in ref. [32], the AIACSE algorithm from ref. [33], the MRCACO algorithm from ref. [35], the MsAACO algorithm from ref. [36], as well as traditional algorithms such as BFS, A*, and Dijkstra as comparison objects. These algorithms are representative and widely used in the field of path planning, which effectively demonstrates the superiority and adaptability of the BCD-ACO algorithm in complex environments.

During the experimental phase, to ensure fairness and reproducibility, all algorithms were subjected to multiple independent tests in the same 30×30 grid environment. By systematically collecting and analysing the resulting data, we conducted a comprehensive evaluation of each algorithm's performance across three key dimensions: path length, number of iterations, and optimisation capability. The aggregated results, presented as convergence curves in Figure 14 and numerical comparisons in Table IV,

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

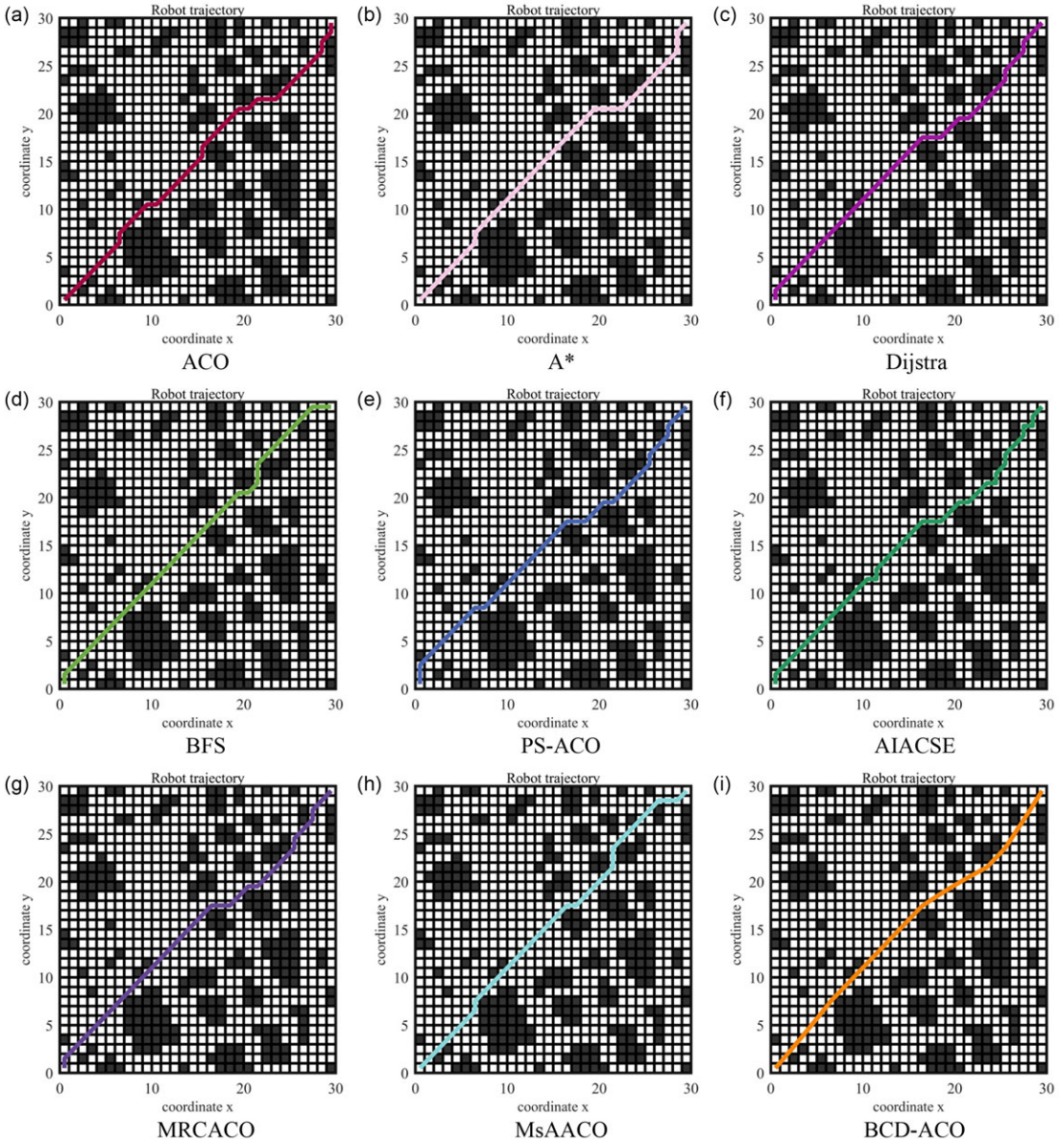


Figure 14. *Comparative experiment 2 path comparison diagram.*

demonstrate the significant advantages of the BCD-ACO algorithm in terms of path planning quality, computational efficiency, and global search performance when compared to the other algorithms.

The experimental results from Figure 14 and Table IV clearly demonstrate that the proposed BCD-ACO algorithm exhibits significant advantages across multiple key metrics in path planning, outperforming the other compared algorithms. Specifically, the paths generated by BCD-ACO show superior smoothness, featuring more continuous and fluent curves that effectively reduce sharp turns and unnecessary fluctuations. This improvement enhances the stability and safety of the robot during movement.

Furthermore, the experimental data indicate that BCD-ACO produces shorter path lengths, optimising movement efficiency while reducing energy consumption and time costs. Additionally, the number of turns in the paths is significantly decreased, resulting in a more streamlined path structure. This not

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

Table IV. *Results of comparative experiment 2.*

Algorithm	Path length			Turns times
	PL_{min}	PL_{mean}	PL_{max}	
ACO	42.77	48.63	2.54	13
A*	42.77	42.77	0	6
Dijstra	42.77	42.77	0	9
BFS	42.77	42.77	0	6
PS-ACO	43.36	45.42	1.79	11
AIACSE	44.53	44.53	0	17
MRCACO	42.77	42.77	0	9
MsAACO	42.77	42.77	0	7
BCD-ACO	41.46	41.56	0.12	6

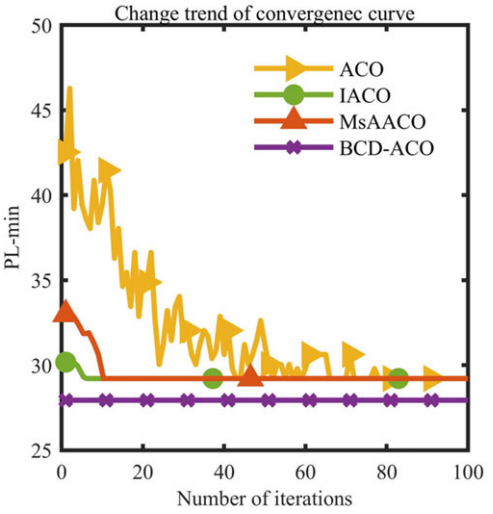


Figure 15. *Comparison of convergence curves of improved ant colony optimisation algorithms in comparative experiment 3.*

only further improves path smoothness but also minimises mechanical wear and control complexity caused by frequent turns during robot navigation.

5.3.3. Comparative experiment 3

Comparative Experiment 3 comprehensively compares BCD-ACO against several existing algorithms in a 20×20 grid environment. The algorithms compared are the IACO algorithm from ref. [34], the MsAACO algorithm from ref. [36], the classical ACO algorithm, the A* algorithm, and Dijkstra's algorithm. The experimental results are shown in Figures 15 and 16 and Table V.

Based on the experimental data from Figures 15, 16, and Table V, the proposed BCD-ACO algorithm demonstrates significant advantages across multiple core metrics in path planning, outperforming the other compared algorithms.

First, it exhibits excellent path smoothness, generating smooth and continuous paths that reduce abrupt turns and fluctuations, thereby enhancing the robot's movement stability and lowering potential

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

Table V. *Results of comparative experiment 3.*

	Path length			Iterations		
Algorithm	PL_{min}	PL_{mean}	PL_{std}	I_{min}	I_{mean}	**Turns times**
ACO	29.21	31.58	1.73	45	62.7	11
A*	29.21	29.21	0	–	–	9
Dijstra	29.21	29.21	0	–	–	9
IACO	29.21	–	–	5	–	10
MsAACO	29.21	29.21	0	5	5.33	7
BCD-ACO	27.85	27.94	0.15	1	2.35	6

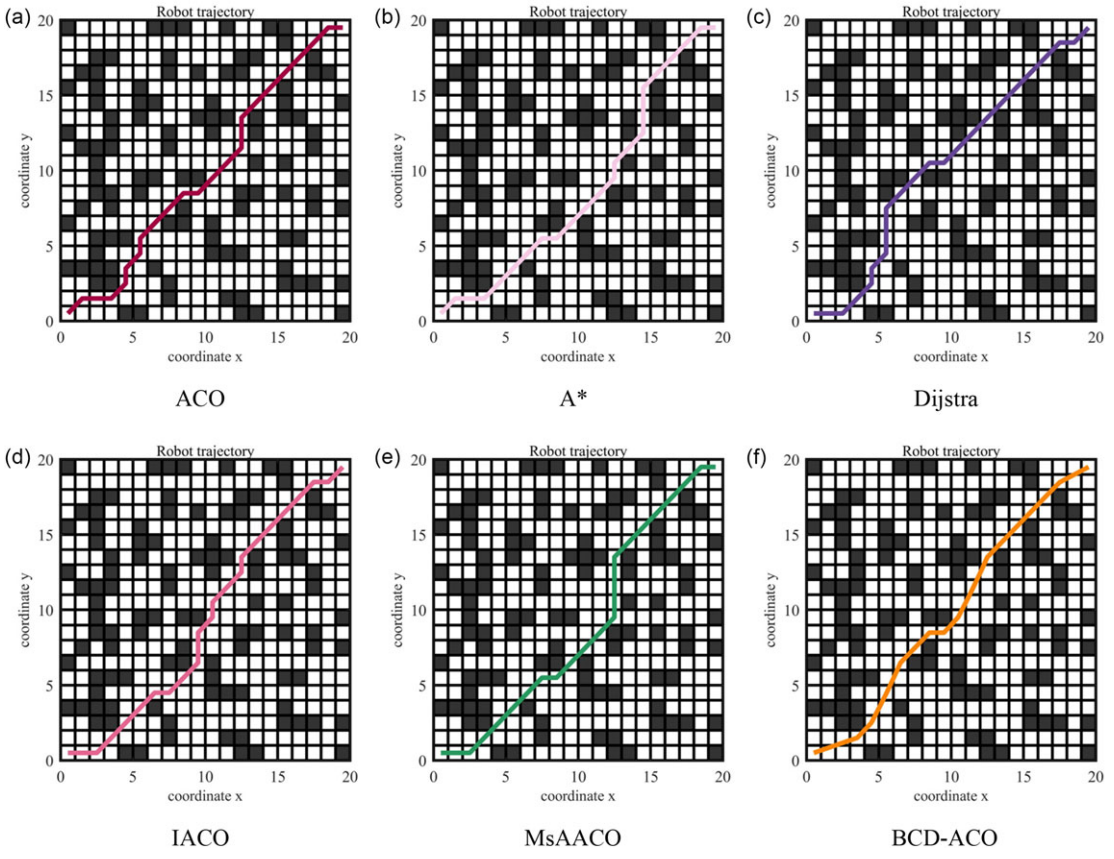


Figure 16. *Comparative experiment 3 path comparison diagram.*

collision risks. Second, the BCD-ACO algorithm plans shorter paths, optimising movement efficiency, which helps save energy and time, particularly beneficial in resource-constrained scenarios. Additionally, the number of turns in the paths is significantly reduced, resulting in a more streamlined path structure. This not only improves smoothness further but also decreases mechanical wear and control complexity caused by frequent turning.

These combined advantages make the BCD-ACO algorithm highly practical for path planning in fields such as robot navigation, logistics optimisation, and autonomous driving, providing robust support for efficient and safe solutions.

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

6. Conclusion

Path planning plays a crucial role in automation and robotics, directly impacting operational efficiency and task safety. However, traditional ACO algorithms often face challenges such as low search efficiency, slow convergence speed, and susceptibility to local optima, especially in complex environments or multi-objective scenarios.

The paper proposes BCD-ACO, an improved ACO algorithm that integrates multiple enhancement strategies to effectively address the traditional ACO's limitations, namely slow convergence, susceptibility to local optima, and excessive path turning points, in 2D grid-based mobile robot path planning, thereby improving motion efficiency and practical applicability. First, by incorporating a Bresenham-based pheromone boosting mechanism, initial pheromone levels are elevated along the approximate straight line path from start to goal, guiding ants to explore near-optimal directions and markedly accelerating convergence. Second, a directional continuity factor is introduced into the transition probability calculation using the dot product of successive movement vectors to favour directional consistency, while a curvature-informed pheromone increment rule encourages the formation of straighter routes; this dual optimisation ensures smoother paths and reduces sharp turns. Third, a spring model-based path smoothing strategy is applied as post-processing to further simplify the ant generated path and decrease turn frequency, enhancing the robot's motion stability and reliability. These strategies are seamlessly combined to form the core framework of BCD-ACO. To validate its performance, comprehensive comparative experiments were conducted in MATLAB against other state-of-the art ACO variants from the literature. Simulation results demonstrate that BCD-ACO significantly outperforms its peers in key metrics, including path smoothness, path length, number of turning points, and convergence efficiency, showcasing its superior path planning capability and providing a robust solution for efficient mobile robot navigation in complex environments.

Looking ahead, the BCD-ACO algorithm holds great potential for applications in mobile robot path planning, with its outstanding performance laying a solid foundation for further theoretical research and practical deployment. Future research can focus on several directions: first, to better handle dynamic environments and multi-robot cooperative scenarios, the algorithm's adaptability can be enhanced by incorporating real-time obstacle perception and multi-objective optimisation mechanisms, thereby improving robustness in complex, non-static settings. Second, integrating deep learning or reinforcement learning techniques can boost the algorithm's intelligent decision-making capabilities, enabling more efficient autonomous exploration and path optimisation in unknown environments. Additionally, exploring extensions of the algorithm to three-dimensional spaces or non-grid environments, such as UAV navigation or underwater robot path planning, can broaden its range of applications. At the same time, deeper integration with hardware platforms can optimise the algorithm's real-time computational efficiency to meet the requirements of low-power embedded devices. In-depth research along these lines will further enhance the performance of the BCD-ACO algorithm, providing stronger technical support for widespread applications in intelligent robotics, autonomous driving, and smart logistics, ultimately helping automation systems achieve higher levels of intelligence and efficiency.

Author contribution. Shuai Wu and Zibo Huang conceived the ideas, conducted the research, and wrote the article. Yapeng Wang provided systematic guidance throughout the research process. All authors collectively discussed the research findings and contributed to the revision and improvement of the final manuscript.

Financial support. This work was supported by the Guangdong Provincial Ordinary Universities Youth Innovative Talent Project (No. 2025KQNCX145), the Dongguan City University Young Teachers Development Fund Project (No. 2025QJ003Z, 2025QJ008Z), the College Students' Innovative Entrepreneurial Training Plan Program (No. S202513844004, S202513844022), and the Macao Polytechnic University (Grant No. RP/FCA-02/2025; Approval Code: fca.30e2.143f.c).

Competing interests. The authors declare that no conflicts of interest exist.

Ethical approval. Not applicable.

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

References

[1] H. Qin, S. Shao, T. Wang, X. Yu, Y. Jiang and Z. Cao, "Review of autonomous path planning algorithms for mobile robots," *Drones* **7**(3), 211 (2023).

[2] L. Liu, X. Wang, X. Yang, H. Liu, J. Li and P. Wang, "Path planning techniques for mobile robots: Review and prospect," *Expert Syst. Appl.* **227**(C), 120254 (2023).

[3] X. Liu, D. Jiang, B. Tao, G. Jiang, Y. Sun, J. Kong, X. Tong, G. Zhao and B. Chen, "Genetic algorithm-based trajectory optimization for digital twin robots," *Front. Bioeng. Biotechnol.* **9**, 793782 (2022).

[4] G. Gao, D. Li, K. Liu, Y. Ge and C. Song, "A study on path-planning algorithm for a multi-section continuum robot in confined multi-obstacle environments," *Robotica* **42**(10), 3324–3347 (2024).

[5] L. Zheng, W. Yu, G. Li, G. Qin and Y. Luo, "Particle swarm algorithm path-planning method for mobile robots based on artificial potential fields," *Sensors* **23**(13), 6082 (2023).

[6] Y. Zhang, "Improved Artificial Potential Field Method for Mobile Robots Path Planning in a Corridor Environment," **In:** *2022 IEEE International Conference on Mechatronics and Automation (ICMA)*, IEEE (2022) pp. 185–190.

[7] P. Maristany de las Casas, L. Kraus, A. Sedeño-Noda and R. Borndörfer, "Targeted multiobjective Dijkstra algorithm," *Networks* **82**(3), 277–298 (2023).

[8] G. Nan, Z. Liu, H. Du, W. Zhu and S. Xu, "Transmission line-planning method based on adaptive resolution grid and improved Dijkstra algorithm," *Sensors* **23**(13), 6214 (2023).

[9] S. Luo, M. Zhang, Y. Zhuang, C. Ma and Q. Li, "A survey of path planning of industrial robots based on rapidly exploring random trees," *Front. Neurorobot.* **17**, 1268447 (2023).

[10] H. C. Tsai and J. Y. Shi, "Potential corrections to grey wolf optimizer," *Appl. Soft. Comput.* **161**, 111776 (2024).

[11] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh and S. Mirjalili, "Particle swarm optimization: A comprehensive survey," *IEEE Access* **10**, 10031–10061 (2022).

[12] H. Heng, M. H. M. Ghazali and W. Rahiman, "Exploring the application of ant colony optimization in path planning for unmanned surface vehicles," *Ocean Eng.* **311**, 118738 (2024).

[13] C. Ren, F. Fu, C. Yin, Z. Yan, R. Zhang and Z. Wang, "Improved artificial potential field method based on robot local path information," *Int. J. Adv. Robot. Syst.* **21**(5), 17298806241278172 (2024).

[14] D. Yang, L. Dong and J. K. Dai, "Collision avoidance trajectory planning for a dual-robot system: Using a modified APF method," *Robotica* **42**(3), 846–863 (2024).

[15] A. S. Abdel-Rahman, S. Zahran, B. E. Elnaghi and S. F. Nafea, "Enhanced hybrid path planning algorithm based on APF and A-star," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **48**, 867–873 (2023).

[16] X. Zhou, J. Yan, M. Yan, K. Mao, R. Yang and W. Liu, "Path planning of rail-mounted logistics robots based on the improved Dijkstra algorithm," *Appl. Sci.* **13**(17), 9955 (2023).

[17] M. Luo, X. Hou and J. Yang, "Surface optimal path planning using an extended Dijkstra algorithm," *IEEE Access* **8**, 147827–147838 (2020).

[18] P. Chi, Z. Wang, H. Liao, T. Li, J. Tian, X. Wu and Q. Zhang, "AM-RRT*: An Automatic Robot Motion Planning Algorithm Based on RRT," **In:** *International Conference on Neural Information Processing*, Singapore, Springer Nature Singapore (2023) pp. 91–103.

[19] L. Ye, F. Wu, X. Zou and J. Li, "Path planning for mobile robots in unstructured orchard environments: An improved kinematically constrained bi-directional RRT approach," *Comput. Electron. Agr.* **215**, 108453 (2023).

[20] X. Li and Y. Tong, "Path planning of a mobile robot based on the improved RRT algorithm," *Appl. Sci.* **14**(1), 25 (2023).

[21] Y. Luo, Q. Qin, Z. Hu and Y. Zhang, "Path planning for unmanned delivery robots based on EWB-GWO algorithm," *Sensors* **23**(4), 1867 (2023).

[22] D. Zhao, G. Cai, Y. Wang and X. Li, "Path planning of obstacle-crossing robot based on golden sine grey wolf optimizer," *Appl. Sci.* **14**(3), 1129 (2024).

[23] Q. Zhang, X. Ning, Y. Li, L. Pan, R. Gao and L. Zhang, "Path planning of patrol robot based on modified grey wolf optimizer," *Robotica* **41**(7), 1947–1975 (2023).

[24] X. Zhou, G. Shi and J. Zhang, "Improved grey wolf algorithm: A method for UAV path planning," *Drones* **8**(11), 675 (2024).

[25] D. Wang, D. Tan and L. Liu, "Particle swarm optimization algorithm: An overview," *Soft Comput.* **22**(2), 387–408 (2018).

[26] S. Lin, A. Liu, J. Wang and X. Kong, "An intelligence-based hybrid PSO-SA for mobile robot path planning in warehouse," *J. Comput. Sci.* **67**, 101938 (2023).

[27] Y. Na, Y. Li, D. Chen, Y. Yao, T. Li, H. Liu and K. Wang, "Optimal energy consumption path planning for unmanned aerial vehicles based on improved particle swarm optimization," *Sustainability-BASEL* **15**(16), 12101 (2023).

[28] X. Li, B. Tian, S. Hou, X. Li, Y. Li, C. Liu and J. Li, "Path planning for mount robot based on improved particle swarm optimization algorithm," *Electronics* **12**(15), 3289 (2023).

[29] R. Gao, Q. Zhou, S. Cao and Q. Jiang, "Apple-picking robot picking path planning algorithm based on improved PSO," *Electronics* **12**(8), 1832 (2023).

[30] B. Song, S. Tang and Y. Li, "A new path planning strategy integrating improved ACO and DWA algorithms for mobile robots in dynamic environments," *Math. Biosci. Eng.* **21**(2), 2189–2211 (2024).

[31] C. Miao, G. Chen, C. Yan and Y. Wu, "Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm," *Comput. Ind. Eng.* **156**, 107230 (2021).

[32] B. Shuang, J. Chen and Z. Li, "Study on hybrid PS-ACO algorithm," *Appl. Intell.* **34**, 64–73 (2011).

[33] S. Zhang, J. Pu and Y. Si, "An adaptive improved ant colony system based on population information entropy for path planning of mobile robot," *IEEE Access* **9**, 24933–24945 (2021).

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press

[34] J. Zhao, D. Cheng and C. Hao, "An improved ant colony algorithm for solving the path planning problem of the omnidirectional mobile vehicle," *Math. Probl. Eng.* **2016**(1), 7672839 (2016).
[35] D. Zhang, X. You, S. Liu and H. Pan, "Dynamic multi-role adaptive collaborative ant colony optimization for robot path planning," *IEEE Access* **8**, 129958–129974 (2020).
[36] J. Cui, L. Wu, X. Huang, D. Xu, C. Liu and W. Xiao, "Multi-strategy adaptable ant colony optimization algorithm and its application in robot path planning," *Knowl-BASED Syst.* **288**, 111459 (2024).
[37] X. Dai, S. Long, Z. Zhang and D. Gong, "Mobile robot path planning based on ant colony algorithm with A* heuristic method," *Front. Neurorobot.* **13**, 15 (2019).
[38] E. Fabrizi and A. Saffiotti, "Augmenting topology-based maps with geometric information," *Robot. Auton. Syst.* **40**(2-3), 91–97 (2002).
[39] M. Dorigo, V. Maniezzo and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst. Man. Cybern. Part B (Cybernetics)* **26**(1), 29–41 (1996).
[40] Y. L. Poy, Z. Y. Loke, S. Darmaraju, C. H. Goh, B. H. Kwan, H. Liu and D. W. K. Ng, "Enhanced particle swarm optimisation for multi-robot path planning with Bezier curve smoothing," *Robotics* **13**(10), 141 (2024).
[41] M. Mohaghegh, H. Jafarpourdavatgar and S. A. Saeedinia, *New Design of Smooth PSO-IPF Navigator with Kinematic Constraints* (IEEE Access, 2024).
[42] N. Cao, G. Yi, S. Zhang and L. Qiu, "A multiobjective path-smoothing algorithm based on node adjustment and turn-smoothing," *Meas. Control* **56**(7-8), 1187–1201 (2023).

Cite this article: S. Wu, Z. Huang, Z. Ye, Y. Wang, X. Yang and S. Im (2025). "Application of improved ant colony algorithm fusing Bresenham and direction factor in mobile robot path planning", Robotica **43**, 4429–4456. https://doi.org/10.1017/S0263574725102932

https://doi.org/10.1017/S0263574725102932 Published online by Cambridge University Press