AIMS Mathematics, 10(12): 30879–30904.
DOI: 10.3934/math.20251355
Received: 08 November 2025
Revised: 13 December 2025
Accepted: 24 December 2025
Published: 30 December 2025

https://www.aimspress.com/journal/Math

Research article

Research on mathematical optimization-driven A* search, DWA improvement, and intelligent robot path planning

Le Gao[1,2,*], Yuying Zhang[1], Pinjie Liu[1,*], Xiaoying Ou[1], Jinglong Cheng[1] and Ying Zhu[3]

[1] School of Computer Engineering, Guangzhou Huali College, Guangzhou 510000, China
[2] Center for Earth Environment and Earth Resources, Sun Yat-sen University, Zhuhai 519000, China
[3] School of Intelligent Manufacturing, Guangzhou Huali College, Jiangmen 529000, China

* **Correspondence:** Email: le.gao@nscc-gz.cn, liupinjie0922@163.com.

Abstract: Mobile robots encounter issues like low global search efficiency and insufficient static path safety in path planning within complex dynamic environments. This paper proposes a fusion strategy integrating a mathematically optimized improved A* algorithm (ImpA*) and an enhanced Dynamic Window Approach (ImpDWA). At the global planning layer, path quality and efficiency are improved through optimizations such as obstacle ratio quantification and dynamic weighting of heuristic functions. At the local planning layer, the DWA evaluation system is optimized by adding a target point cost sub-function and dynamically adjusting weights. At the fusion layer, dual-algorithm collaboration is achieved via global path segmentation and key sub-target transmission. MATLAB simulations show that the ImpA* algorithm significantly optimizes path length and runtime. The fusion algorithm (ImpA*-ImpDWA) achieves an obstacle avoidance success rate exceeding 96.5% in dynamic environments, with comprehensive performance superior to other mainstream schemes. It realizes the coordinated balance of core indicators including safety and smoothness, providing reliable support for autonomous robot navigation.

Keywords: mathematical optimization; improved A* algorithm; dynamic window approach; mobile robot; path planning
Mathematics Subject Classification: 90B06, 90C90

1. Introduction

With the in-depth advancement of intelligent manufacturing and industrial automation, intelligent mobile robots have demonstrated enormous application potential in industrial logistics [1], smart agriculture [2], and daily services [3] due to their capabilities in autonomous decision-making, collaborative control, and environmental adaptability. Path planning, as a key technology of the autonomous navigation system, directly determines the movement efficiency of robots in complex environments [4–6]. The core goal of this technology is to generate paths that meet optimality and safety requirements in complex environments through computer algorithms, and mathematical optimization is the core tool to achieve these two goals. The core objective of this technology is to construct objective functions and constraint conditions in complex environments through computer algorithms, thereby generating paths that satisfy optimality and security [7]. Mathematical optimization serves as the core tool for achieving these two goals. Mathematical optimization can reasonably schedule resources and reduce the volume of communication data [8]. It transforms the qualitative requirements of path optimization into quantifiable and solvable mathematical models, providing rigorous theoretical support for algorithm improvement.

According to the difference in environmental information, path planning can be divided into global path planning and local path planning. Improvements in both types of algorithms rely on mathematical optimization to achieve performance breakthroughs. Global path planning is based on the premise of fully known environmental information and aims to solve the theoretically optimal path [9]. Typical algorithms include the A* algorithm [10] and Dijkstra's algorithm [11]. Among them, the A* algorithm is widely used due to its simple principle and efficient solution, but its performance is highly dependent on heuristic function design and search strategies. The heuristic function weight of the traditional A* algorithm is fixed, which cannot dynamically adjust the balance between exploration and exploitation according to environmental complexity. This leads to problems such as many redundant search nodes and poor path smoothness in obstacle-dense areas [12,13]. To address this issue, numerous scholars have also adopted other algorithms for global path planning research. Dai et al. [14] proposed an LPA* algorithm and applied it to path planning. Through comparative experiments with the original LPA algorithm and A* algorithm, the results show that the LPA* can effectively shorten the travel distance of mobile robots and reduce time consumption. Li et al. [15] designed an improved Dijkstra's algorithm (IDA*). Compared with the traditional Dijkstra's algorithm, IDA* can significantly improve computational efficiency and save operation time. To target the computational load challenge caused by large-scale grid maps, Liu et al. [16] adopted an improved Theta* algorithm to generate optimized paths for each sub-region. Subsequently, through a path connection mechanism, the local paths of adjacent grids are integrated to ultimately form a complete global travel path. Local planning targets scenarios with unknown or dynamically changing environmental information, and its core goal is to generate collision-free paths in real time [17]. DWA [18] is a mainstream algorithm in this field. The traditional DWA algorithm samples the velocity space and selects the optimal velocity through an evaluation function. However, the weight coefficients of the evaluation function are fixed and cannot be adjusted according to dynamic information such as obstacle distance and target position, making it prone to falling into local minima traps [19,20]. To address this problem, this paper improves the DWA with mathematical optimization as the core method.

Despite the certain progress made in improving single algorithms through mathematical

optimization in existing research, mobile robots still face problems in complex dynamic environments, such as insufficient coordination between global paths and local obstacle avoidance, and difficulty in simultaneously meeting path optimization efficiency, smoothness, and safety. Therefore, this paper proposes a fused path planning strategy of ImpA*-ImpDWA based on mathematical optimization. The specific improvements are reflected in three aspects: First, at the global planning layer, the search efficiency of the A* algorithm is improved by introducing a quantitative model of environmental complexity (global obstacle occupancy ratio) and dynamic weight optimization of the heuristic function. Meanwhile, path smoothness optimization is achieved through geometric elimination rules of redundant nodes and curve fitting. Second, at the local planning layer, the DWA evaluation system is reconstructed by constructing a target point cost sub-function and dynamic weight optimization of the speed sub-function, alleviating the problems of local minima and target loss. Third, at the algorithm fusion level, the collaboration between the ImpA* and ImpDWA is realized through the mathematical segmentation of the global path, ensuring intelligent obstacle avoidance in dynamic environments. Finally, the effectiveness of the fused algorithm is verified through simulation experiments.

2. Materials and methods

2.1. Assumptions of path planning model and rationality verification

To clarify the applicable scenarios and boundary conditions of the algorithm, the path planning model proposed in this paper is constructed based on the following 3 core assumptions, whose rationality is verified through theoretical derivation and experimental data:

(1) Environmental gridding assumption: The robot's movement environment is discretized into uniform square grids with a size of 1m×1m. Grid states are defined using binary discrete values, where 1 represents an obstacle grid and 0 represents a passable grid. Rationality verification: Gridding is a standardized environmental modeling method in the field of path planning. The 1m×1m grid size is highly matched with the physical size of the mobile robot (0.3m in diameter) used in experiments. This size not only ensures the modeling accuracy of environmental details but also avoids the exponential growth of computational complexity caused by overly fine grids. In a 30×30 grid map, the environmental modeling error under this assumption is less than 0.2m, which is much lower than the precision threshold of robot motion control (±0.1m), fully meeting the environmental description requirements of path planning.

(2) Simplified robot kinematics assumption: The robot is assumed to be a two-wheel differential drive model. During movement, only linear velocity and angular velocity constraints are considered, while non-ideal interference factors such as sliding friction and uneven ground are ignored. Rationality verification: By adding strict velocity and acceleration/deceleration constraints to the model, it can accurately reflect the robot's actual movement capabilities. The specific constraint parameters are set as follows: maximum linear velocity 1m/s, maximum angular velocity 0.6rad/s, maximum linear acceleration 0.2m/s^2, and maximum angular acceleration 0.3rad/s^2. In the simulation environment, the trajectory prediction error based on this assumption is less than 5%, which not only simplifies the algorithm's calculation process and ensures real-time performance but also meets the core requirement of trajectory accuracy for path planning.

(3) Uniform linear motion assumption for dynamic obstacles: The motion trajectory of dynamic

obstacles is assumed to be uniform linear motion, and their velocity magnitude and direction remain constant within the sampling period. Rationality verification: In typical application scenarios such as warehousing workshops and smart factories, the motion states of dynamic obstacles such as personnel and AGVs can be approximated as uniform linear motion. In experiments, the velocity range of dynamic obstacles is set to 0.6–1.0m/s, which matches the walking speed of personnel in real scenarios (0.9–1.2m/s). By comparing the predicted and actual values of obstacle motion trajectories, the position error under this assumption is less than 0.05m within a 0.1s sampling period, which will not affect the effectiveness of obstacle avoidance decisions.

2.2. A Algorithm and its improvement*

2.2.1. Traditional A* algorithm

As a classic heuristic global path planning algorithm, the A* algorithm is an extension of Dijkstra's algorithm [21]. The algorithm uses nodes as the basic search unit and constructs a path from the start point to the target point by iteratively expanding nodes. Its operation mechanism is as follows: The current node is set as the parent node, and child nodes in its surrounding neighborhood are searched. The cost value of each child node is calculated according to preset rules, and the child node with the smallest cost value is selected as the new parent node. During the algorithm execution, node management is realized by maintaining an Open list and a Close list: the Open list stores child nodes to be expanded, and the Close list records nodes that have completed searching, thereby improving search efficiency.

The core advantage of the A* algorithm lies in introducing a heuristic function to guide the search direction and accelerating algorithm convergence by estimating the cost from the current node to the target point. The algorithm continuously iterates the node expansion and cost value calculation process until the target point is found or all reachable nodes are traversed. The evaluation function of the traditional A* algorithm is shown in Formula (1):

$$f(n) = g(n) + h(n). \tag{1}$$

Where, n is the current node; $f(n)$ is the total cost value from the start point to the target point; $g(n)$ is the actual cost value from the start point to the current node n; $h(n)$ is the estimated cost value from the current node n to the target point.

In practical applications, the distance measurement methods commonly used in heuristic functions include Chebyshev distance (h_1), Euclidean distance (h_2), and Manhattan distance (h_3) [22–24], whose geometric relationships are shown in Figure 1. The corresponding calculation formulas are shown in (2)–(4)

$$h_1(n) = max\{|x2 - x1|, |y2 - y1|\}, \tag{2}$$

$$h_2(n) = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}, \tag{3}$$

$$h_3(n) = |x2 - x1| + |y2 - y1|. \tag{4}$$

Where ($x1$, $y1$) are the coordinates of the start point, and ($x2$, $y2$) are the coordinates of the target node. The Euclidean distance formula is selected as the heuristic function in this paper.

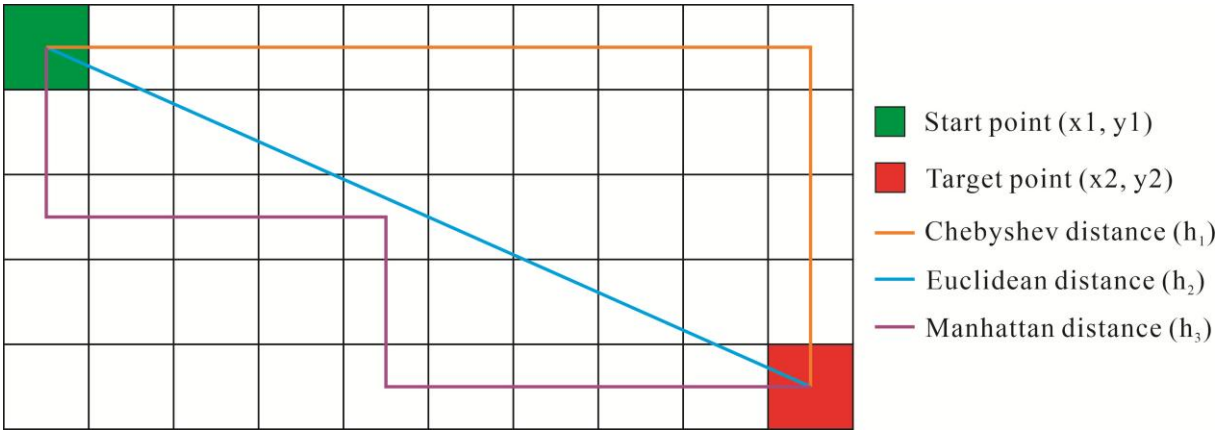

Figure 1. Schematic diagram of the 3 distances.

2.2.2. Heuristic function optimization

The coefficient of the heuristic function h(n) in the traditional A* algorithm is usually a constant, which makes the weight unable to be dynamically adjusted according to environmental characteristics during path planning and limits the further improvement of search efficiency. This paper proposes a dynamic weight adjustment strategy to balance the contradiction between search space and search efficiency by real-time optimizing the weight coefficient of the heuristic function. The specific implementation is as follows:

(1) Distance-adaptive weight adjustment mechanism: When the current node is far from the target point, increase the weight coefficient of h(n) to reduce the search space and improve global search efficiency. When the current node is close to the target point, decrease the weight coefficient of h(n) to expand the local search space, ensuring the algorithm can safely avoid obstacles.

(2) Environmental complexity-aware weight adjustment mechanism: To quantitatively describe the complexity of the map environment, the obstacle occupancy ratio O is introduced as an environmental characteristic parameter. Its mathematical expression is shown in Formula (5):

$$O = \frac{N}{m * n}. \tag{5}$$

Where N is the number of obstacle grids in the map area, and m and n are the horizontal and vertical grid numbers of the map area respectively. The weight adjustment rule is designed as follows: when the obstacle occupancy ratio O in the environment where the robot is located is low, increase the weight of $h(n)$ to improve search efficiency; when O is high and obstacles are dense, decrease the weight of $h(n)$ to expand the search range and avoid falling into local optimality.

The improved method of dynamic weighting for the heuristic function is shown in Formulas (6)–(9):

$$\omega = e^{\frac{d_1}{d_2}} - \frac{1}{2}(1-e^{-O}), \tag{6}$$

$$d_1 = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}, \tag{7}$$

$$d_2 = \sqrt{(x2 - x0)^2 + (y2 - y0)^2}, \tag{8}$$

$$f(n) = g(n) + [e^{\frac{d_1}{d_2}} - \tfrac{1}{2}(1\text{-}e^{-O})]\ h(n). \tag{9}$$

Where ω is the dynamic weight coefficient, (*x0, y0*) are the coordinates of the start point, d_1 is the distance from the current node to the target point, and d_2 is the distance from the start point to the target point. By introducing the dynamic weight ω, the problems of low search efficiency and easy falling into local optimality of the traditional A* algorithm are effectively solved.

2.2.3. Optimized search neighborhood

The selection of the search neighborhood has a key impact on path planning efficiency and path quality [25]. Common search neighborhoods in path planning include 4-neighborhood, 8-neighborhood, and 16-neighborhood, whose specific structures are shown in Figure 2(a). The 4-neighborhood search has a turning angle of 90° and a maximum step size of 1, resulting in poor path smoothness; the 8-neighborhood search has a turning angle of 45° and a maximum step size of $\sqrt{2}$, with significantly improved path smoothness; the 16-neighborhood search has a turning angle of 22.5° and a maximum step size of $\sqrt{10}$, achieving the optimal smoothness. However, the computational complexity increases exponentially with the increase of neighborhood dimensions, which is prone to a sharp increase in planning time in large-scale maps. Considering the balance between path smoothness and computational efficiency, the 8-neighborhood is selected as the search direction in this paper.

In the traditional 8-neighborhood A* algorithm, the movement costs of the 8 child nodes of the parent node are set to fixed values and do not change with the search direction. This results in an insignificant difference in the total path cost between expanding toward the target point and expanding in other directions. This mechanism tends to store a large number of redundant nodes in the Open list, which not only reduces search efficiency but also may cause the algorithm to fall into local optimality. To address this, this paper introduces a turning cost term into the cost function (as shown in Figure 2(b)). By increasing the cost of turning operations, the algorithm is guided to preferentially expand paths in straight or near-straight directions, reducing unnecessary turns, thereby reducing the number of redundant nodes in the Open list, improving search efficiency, and optimizing path smoothness. The improved cost function is shown in Formula (10):

$$f(n) = g(n) + [e^{\frac{d_1}{d_2}} - \tfrac{1}{2}(1\text{-}\ e^{-O})]\ h(n)\ - K D\ \cos\theta. \tag{10}$$

Where K represents the weight coefficient with a value range of (0, 1), which is used to adjust the proportion of global turning cost; θ denotes the angle formed by the target node-parent node-child node; D is the distance from the parent node to the target node—the farther the child node is from the target point, the greater the turning cost.

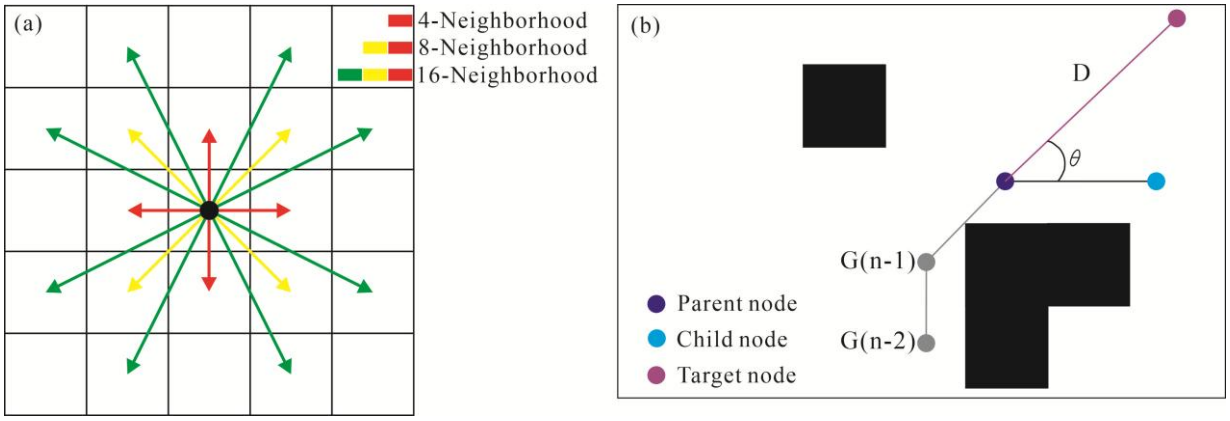

Figure 2. Optimized neighborhood search: (a) Schematic diagram of neighborhood search; (b) Schematic diagram of distance and angle.

2.2.4. Redundant node optimization

In complex environments, the path planned by the traditional A* algorithm often contains a large number of redundant nodes, which may also affect the motion safety of mobile robots. To address this, this paper proposes an A* redundant node elimination strategy. By removing redundant inflection points in the path, a smooth and safe trajectory is generated, and its principle is shown in Figure 3(a). Let N_0 be the start point and N_T be the target point. The path planned by the traditional A* algorithm is expressed as (N_0, N_1, N_2, N_3, N_4, N_5, N_6, N_7, N_T), which contains multiple redundant nodes. The specific optimization strategy is as follows:

For non-adjacent nodes N_i and N_j（$j > i+1$） in the path, if the following two conditions are met: first, the Euclidean distance $d(N_i, N_j)$ between the two points is less than the cumulative distance of the connections between all adjacent nodes from N_i to N_j in the path; second, the line segment $\overline{N_i N_j}$ has no collision with obstacles in the environment, and the vertical distance d_{min} to the nearest obstacle is not less than the preset safety radius r, then all nodes between N_i+1 and N_j-1 can be determined as redundant nodes and removed from the path. Through this operation, the path will only retain the start node, key inflection points, and target node. For example, in Figure 3(a), the retained path after the first round of optimization is (N_0, N_3, N_8, N_5, N_T). Iterative repetition finally generates the optimized path (N_0, N_3, N_T). If the line segment collides with an obstacle, the detection of the current non-adjacent nodes is skipped, and the judgment on the next group of nodes is continued.

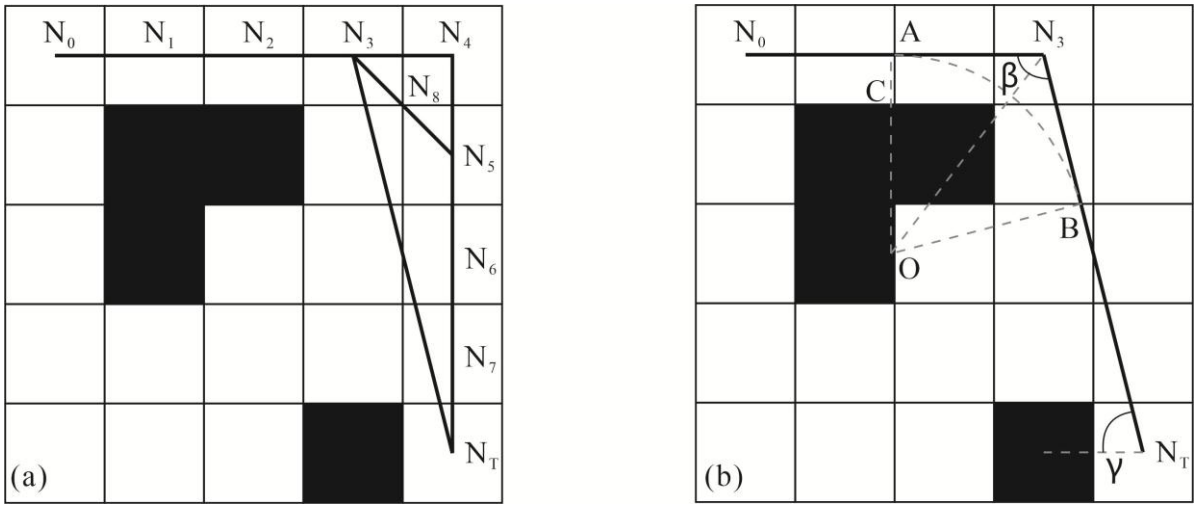

Figure 3. Schematic diagrams of redundant node optimization and path smoothing optimization: (a) Redundant node optimization; (b) Path node optimization after circular arc processing.

2.2.5. Path smoothing optimization

After redundant node optimization, the path (N_0, N_3, N_T) has achieved significant improvements in length, number of inflection points, and safety, but the problem of insufficient path smoothness still exists. In the field of path smoothing, there are various mature optimization methods, such as the B-spline curve method [26], cubic spline curve method [27], B-spline curve method [28], and circular arc transition method [29]. This paper adopts the circular arc transition smoothing method for further path optimization, and its principle is shown in Figure 3(b).

Given the start point $N_0(a_1, b_1)$, inflection point $N_3(a_2, b_2)$, and target point $N_T(a_3, b_3)$, where OA and OB are the safety radius R, the derivation of relevant mathematical relationships is shown in Formulas (11)–(13):

$$y_1 = k_1 x_1 - \frac{a1b2-b1a2}{a2-a1}, \tag{11}$$

$$y_2 = k_2 x_2 - \frac{a3b2-b3a2}{a2-a3}, \tag{12}$$

$$\alpha = \arctan k_1. \tag{13}$$

Where k_1 is the slope of the straight line N_0N_3, k_2 is the slope of the straight line N_3N_T, and α is the angle between N_0N_3 and the horizontal plane. According to geometric relationships, the relationship between the chord length and radius of the circular arc segment is shown in Formula (14):

$$l_{AN3} = R \tan\frac{\beta}{2}. \tag{14}$$

Where β is the angle between line segment N_0N_3 and line segment N_3N_T; l_{AN3} is the distance from the inflection point N_3 to the tangent point A of the circular arc. The distance l_{AN0} from the

start point N_0 to the tangent point A is shown in Formula (15):

$$l_{AN0} = \sqrt{(a2 - a1)^2 + (b2 - b1)^2} - l_{AN3}. \tag{15}$$

When $l_{AC} \geq r$ and $\beta \geq 90°$, the smoothing optimization is performed; otherwise, it is skipped. By combining Formulas (11)–(13), the coordinates of the tangent point $A(A_x, A_y)$ are obtained as shown in Formula (16):

$$\begin{cases} A_x = a1 + \cos\alpha\, l_{AN0} \\ A_y = \frac{b2 - b1}{a2 - a1} A_x - \frac{a1b2 - b1a2}{a2 - a1} \end{cases} \tag{16}$$

Similarly, the coordinates of the other tangent point $B(B_x, B_y)$ can be obtained, and the coordinates of the circular arc center $O(Ox, Oy)$ are shown in Formula (17):

$$\begin{cases} O_x = A_x + \sin\alpha\, R \\ O_y = A_x - \cos\alpha\, R \end{cases} \tag{17}$$

In summary, the expression of the optimized trajectory for the mobile robot is shown in Formula (18):

$$y = \begin{cases} \frac{b2 - b1}{a2 - a1} x - \frac{a1b2 - b1a2}{a2 - a1}, x \leq A_x \\ \sqrt{R^2 - (x - O_x)^2} + O_y, A_x < x < B_x \\ \frac{b3 - b2}{a3 - a2} x - \frac{a2b3 - b2a3}{a3 - a2}, x \geq B_x \end{cases} \tag{18}$$

2.3. DWA algorithm and its improvement

The DWA is a local path planning algorithm based on the robot's kinematic model, which exhibits excellent real-time obstacle avoidance and motion control capabilities in dynamic environments [30]. The algorithm uses the velocity space (v, ω) as the core to characterize the robot's motion state. By combining different linear velocities v and angular velocities ω, it simulates and predicts the robot's possible future motion trajectories. Subsequently, each trajectory is comprehensively and meticulously scored according to a carefully preset evaluation function, and the optimal trajectory is accurately selected therefrom, providing strong guidance for the robot to make precise motion decisions.

2.3.1. Kinematic model

To accurately simulate the robot's motion trajectory, the DWA algorithm constructs a mathematical model based on kinematic principles. Assuming that the robot's linear velocity at time t is v_t, angular velocity is ω_t, yaw angle is θ_t, and velocity sampling period is Δt, its kinematic model is shown in Formula (19):

$$\begin{cases} x(t+1) = x(t) + v_t \Delta t \cos\theta_t \\ y(t+1) = y(t) + v_t \Delta t \sin\theta_t \\ \theta_{t+1} = \theta_t + \omega\Delta t \end{cases} \tag{19}$$

2.3.2. Velocity sampling

In practical application scenarios, the movement speed of mobile robots is subject to the dual constraints of their own conditions and complex surrounding environmental factors, mainly reflected in linear velocity, angular velocity, and acceleration.

(1) Self-velocity constraints: Limited by the robot's hardware configuration, the robot's movement speed has clear maximum and minimum values. These two extreme values jointly determine the size of the dynamic window, and their expression is shown in Formula (20):

$$v_m = \{(v, \omega) | v \in [v_{min}, v_{max}], \omega \in [\omega_{min}, \omega_{max}]\}. \tag{20}$$

Where v_{min} and v_{max} represent the minimum and maximum linear velocities of the robot, respectively, and ω_{min} and ω_{max} represent the minimum and maximum angular velocities of the mobile robot, respectively. This constraint condition defines a clear boundary for the robot's speed range at the hardware level, ensuring that the robot's motion always remains within a reasonable range that the hardware can withstand.

(2) Motor acceleration/deceleration constraints: During the acceleration or deceleration process of the mobile robot, its speed change is strictly limited by the acceleration/deceleration capacity of the motor. This constraint restricts the current speed and the speed at the next sampling moment within the range allowed by the acceleration. The acceleration constraint formula is shown in Formula (21):

$$v_d = \{(v, \omega) | v \in [v_c - a_d \Delta t, v_c + a_i \Delta t], \omega \in [\omega_c - a_n \Delta t, \ \omega_c + a_m \Delta t]\}. \tag{21}$$

Where v_c and ω_c are the current linear velocity and angular velocity of the robot, respectively, a_i and a_d are the maximum linear acceleration and linear deceleration of the robot, respectively, a_m and a_n are the maximum angular acceleration and angular deceleration of the robot, respectively, and Δt represents the sampling time.

(3) Braking Distance Constraint: To ensure that the robot can stop in time before a collision hazard occurs during its movement, the braking distance constraint clearly defines the relationship between the speed range and the minimum distance to the obstacle. The braking speed constraint formula for the robot is shown in Formula (22) as follows:

$$v_a = \left\{(v, \omega) \middle| v \leq \sqrt{2\, dist(v, \omega)\, a_d}, \omega \leq \sqrt{2\, dist(v, \omega)\, a_n}\right\}. \tag{22}$$

Where *dist(v, ω)* is the distance from the current trajectory to the nearest obstacle. By reasonably limiting the speed, this constraint ensures the robot has sufficient braking distance when encountering obstacles, effectively avoiding collision accidents. In summary, the speed range of the mobile robot can be briefly expressed as (23):

$$v_r = v_m \cap v_d \cap v_a. \tag{23}$$

This formula comprehensively considers self-velocity constraints, motor acceleration/ deceleration constraints, and braking distance constraints, determining the feasible speed range of the mobile robot in actual motion and laying a foundation for subsequent trajectory deduction and optimal evaluation.

2.3.3. Optimized evaluation function

After completing velocity sampling, the DWA algorithm comprehensively scores the motion trajectories corresponding to multiple sets of candidate velocities based on the evaluation function, and then selects the optimal trajectory. The evaluation function adopted by the traditional DWA algorithm is shown in Formula (24):

$$G(v,\omega) = \sigma[\alpha \cdot head(v,\omega) + \beta \cdot dist(v,\omega) + \lambda \cdot vel(v,\omega)]. \quad (24)$$

Here, *dist(v, ω)* is the distance from the simulated trajectory to the nearest obstacle; vel(v, ω) is the speed corresponding to the simulated trajectory; σ is a smoothing function; α, β, and λ are the weighting coefficients of the respective cost sub-functions. Where the calculation formula of head(v, ω) is shown in Formula (25):

$$head(v,\omega) = 180° - \theta. \quad (25)$$

Here, θ is the angle between the end of the robot's simulated trajectory and the direction of the target point; The higher the speed, the higher the score of *vel(v, ω)*; the smaller the θ, the higher the score of *head(v, ω)*; the farther the simulated trajectory is from the nearest obstacle, the higher the score of dist(v, ω), and the safer the trajectory.

However, the coefficient of the speed cost sub-function in the traditional evaluation function is a constant, which makes the algorithm prone to falling into local optimality in complex environments. To address the above problem, this paper makes two improvements to the evaluation function: first, introduce a dynamic weight coefficient for the speed cost sub-function *vel(v, ω)*, enabling it to dynamically adjust the weight ratio according to the distance between the robot and obstacles—increase the weight to improve operational efficiency when far from obstacles, and decrease the weight to ensure safe obstacle avoidance when close to obstacles; second, add a target point cost sub-function *distg(v, ω)* to the cost *G(v, ω)*, which effectively shortens the motion path length by quantifying the proximity between the path and the target point. The improved *G(v, ω)* is shown in Formula (26):

$$G(v,\omega) = \sigma[\alpha \cdot head(v,\omega) + \beta \cdot dist(v,\omega) + \lambda \cdot \ln(e + \frac{dist(v,\omega)}{d_2})vel(v,\omega) + \gamma\, dist_g(v,\omega)]. \quad (26)$$

Here, γ is the weighting coefficient of $dist_g(v,\omega)$. The calculation formula of *distg(v, ω)* is shown in Formula (27):

$$dist_g(v,\omega) = \frac{1}{\sqrt{(x_g - x_m)^2 + (y_g - y_m)^2}}. \quad (27)$$

Here, (x_m, y_m) are the coordinates of the end of the trajectory, (x_g, y_g) are the coordinates of the target point, The smaller the distance between the end of the trajectory and the target point, the higher the score of $dist_g(v,\omega)$, and vice versa.

3. Algorithm fusion

This paper proposes a mathematical optimization strategy that integrates the ImpA* algorithm with the ImpDWA. The workflow of this fused algorithm is shown in Figure 4. By integrating global planning and dynamic optimization mechanisms, this optimized fusion strategy enables mobile

robots to achieve efficient path planning in complex environments.

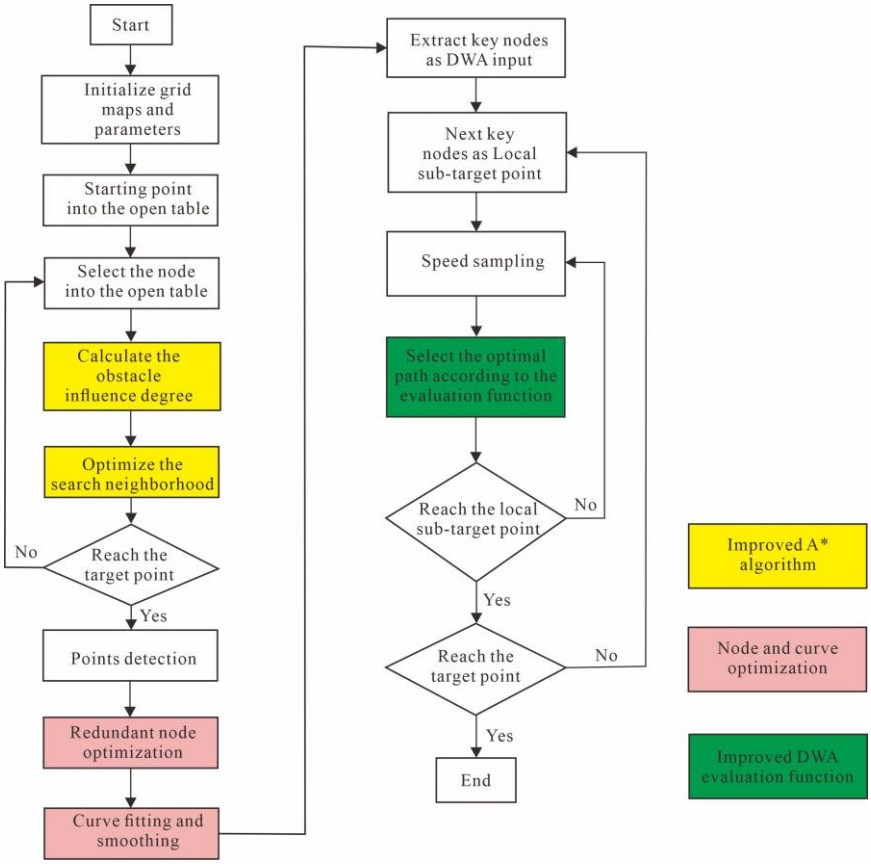

Figure 4. The flow of the fusion algorithm.

First, the ImpA* algorithm is adopted for global path planning. During the node search process, obstacle influence factors and dynamic weight coefficients are introduced, allowing the algorithm to complete preliminary obstacle avoidance while guiding towards the target point, significantly reducing computational redundancy in subsequent local planning. To address issues such as insufficient smoothness and dense inflection points in paths generated by the traditional A* algorithm, curve fitting technology is further used for secondary path optimization, ultimately obtaining a smooth global path that meets the robot's kinematic constraints.

Subsequently, key nodes are extracted from the optimized global path as input for the local sub-targets of the ImpDWA algorithm. Based on the robot's kinematic model, multiple sets of candidate motion trajectories are generated in the locally perceived environment. Finally, combined with the direction guidance of the global path and the improved evaluation function, a multi-dimensional comprehensive evaluation of candidate trajectories is performed: the global path constraint ensures the direction consistency of local planning, and the dynamic weight adjustment mechanism and target point cost sub-function in the evaluation function realize multi-objective optimization of trajectory safety, motion efficiency, and path length. The optimal motion trajectory is then selected to complete the closed-loop control of algorithm fusion.

Under this fusion framework, the ImpA* algorithm undertakes the function of global path topology construction, providing macro navigation guidance for the robot; the ImpDWA algorithm

focuses on the local area between adjacent key nodes, achieving real-time obstacle avoidance for unknown static and dynamic obstacles. The two form a collaborative closed loop through the transmission of key sub-targets, effectively improving the comprehensive path planning performance of mobile robots in complex environments.

4. Experimental results and discussion

To verify the effectiveness of the fused algorithm proposed in this paper, simulation experiments were conducted using MATLAB R2022b. The experimental environment was built on the Windows 10 operating system with hardware configuration of an i7-13620H processor and 16 GB memory, ensuring the stability and reliability of experimental data. The environmental parameters are as follows: grid size of 1m×1m; evaluation function coefficients α=0.05, β=0.2, γ=0.2, η=0.1; constrained maximum linear velocity of 1m/s, maximum angular velocity of 0.6rad/s, maximum linear acceleration of 0.2m/s^2, maximum angular acceleration of 0.3rad/s^2; velocity range of dynamic obstacles of 0.6–1.0m/s.

4.1. Simulation experiment analysis of the global algorithm

Under the unified grid map environment setting, global path planning simulations were performed using Dijkstra's algorithm, traditional A* algorithm, LPA* algorithm, IDA* algorithm, Theta* algorithm, and ImpA* proposed in this paper. Two specifications of grid maps were set up in the experiments: a 30×30 map (obstacle coverage rate of 22.22%) and a 20×20 map (obstacle coverage rate of 20%). Among them, black areas represent obstacles, white areas represent passable areas, and gray areas represent the algorithm's search space. For both maps, the start point coordinates are set as (1, y-coordinate of the end point) and the end point coordinates as (x of the map size, 1). Specifically, the 30×30 map has a start point of (1, 31) and an end point of (31, 1); the 20×20 map has a start point of (1, 21) and an end point of (21, 1), so as to simulate static environments with different complexities.

As shown in Figures 5 and 6, all six algorithms can plan global paths in both 30×30 and 20×20 grid maps, but there are significant differences in path quality and search efficiency. Dijkstra's algorithm generates paths with many inflection points and low search efficiency. Although the traditional A* algorithm and IDA* algorithm perform better than Dijkstra's algorithm, they still have the problem that the path is close to the vertices of obstacles (Figures 5(a), (b), (d) and 6(a), (b), (d)), which is prone to collision risks in narrow channel environments. The LPA* algorithm has high search redundancy in static environments (Figures 5(c) and 6(c)), and while the Theta* algorithm can generate smoother paths, it has a longer running time (Figures 5(e) and 6(e)). In contrast, the ImpA* algorithm proposed in this paper, through dynamic weight coefficient adjustment, search neighborhood optimization, and path smoothing strategies, successfully avoids obstacle vertices (Figures 5(f) and 6(f)), reserves sufficient turning space for the mobile robot, and significantly improves obstacle avoidance safety. At the same time, sharp-turn paths are replaced by smooth curves, effectively enhancing the continuity of motion trajectories. This indicates that the multi-dimensional optimization strategy of the ImpA* algorithm can provide a better trajectory planning solution for robot movement in complex environments.

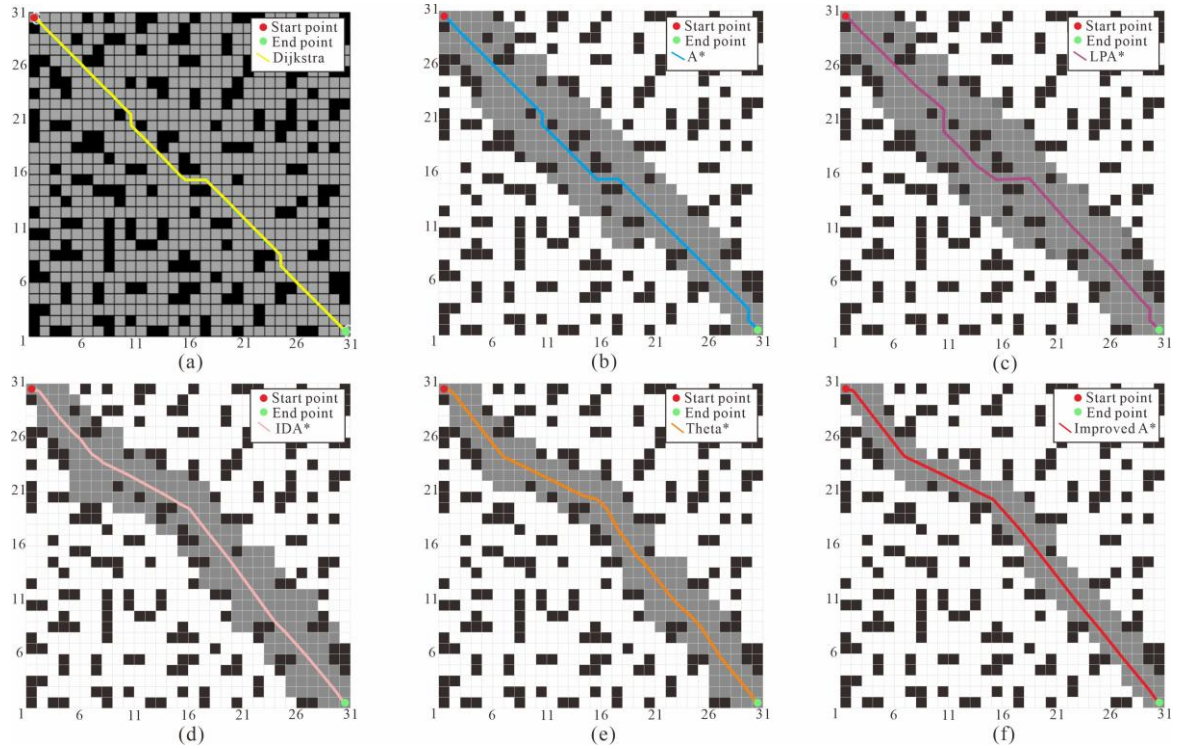

Figure 5. Simulation results of different algorithms in 30×30 map.

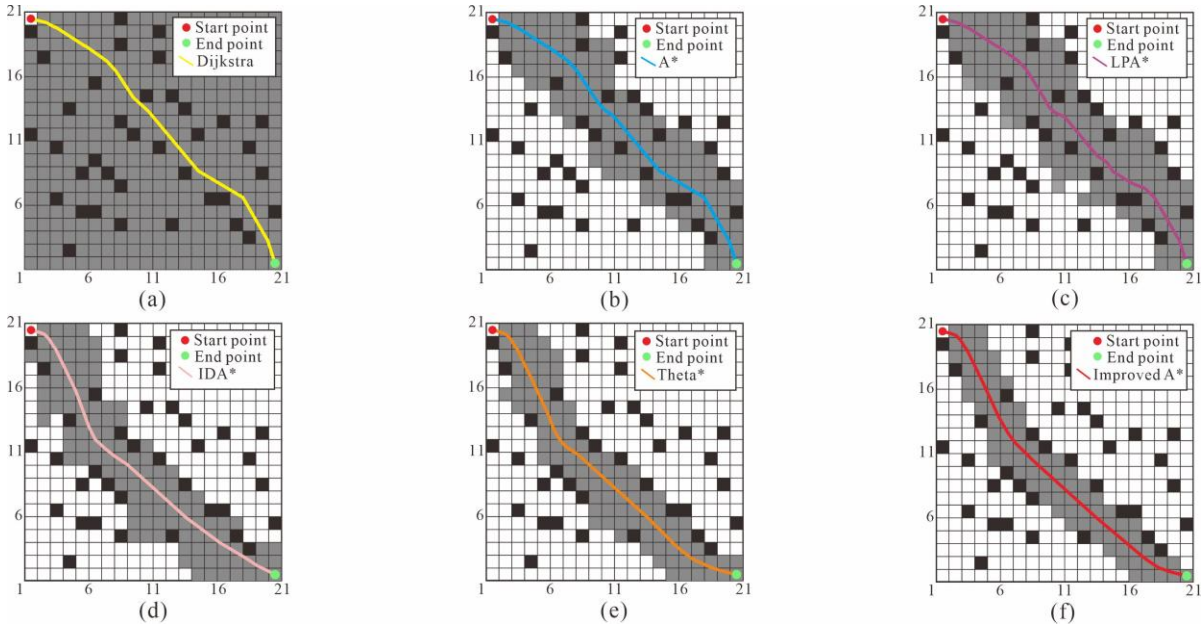

Figure 6. Simulation results of different algorithms in 20×20 map.

Quantitative analysis results further confirm the superiority of the improved algorithm. In the 30×30 map (Table 1), the path length planned by the ImpA* algorithm in this paper is 41.81m, which is 0.90%, 0.88%, 0.99%, 0.81%, and 0.40% shorter than that of Dijkstra's algorithm (42.19m), traditional A* algorithm (42.18m), LPA* algorithm (42.23m), IDA* algorithm (42.15m), and Theta* algorithm (41.98m), respectively. The path running time of the proposed optimization method is

83.54s, which is 26.71%, 4.72%, 5.13%, 4.44%, and 2.59% less than that of the above algorithms, respectively. The number of search nodes is only 129, which is much lower than the search numbers of the other five comparative algorithms.

Table 1. Comparison of simulation results of different algorithms in 30×30 map (global planning).

Map size	Algorithm	Length/m	Moving time/s	Search nodes
30×30	Dijkstra [11]	42.19	113.98	700
	A* [10]	42.18	87.68	213
	LPA* [14]	42.23	88.06	235
	IDA* [15]	42.15	87.42	198
	Theta* [16]	41.98	85.76	162
	ImpA* (Ours)	41.81	83.54	129

The above data indicate that the ImpA* algorithm in this paper can significantly reduce the search space through multi-strategy collaborative optimization in static environments with different complexities. While ensuring path safety and smoothness, it significantly improves operational efficiency and reduces dependence on computing resources, demonstrating excellent comprehensive performance in global path optimization.

In the 20×20 map (Table 2), the path length of the ImpA* algorithm proposed in this paper is 27.65m, which is 0.97%, 0.90%, 1.00%, 0.82%, and 0.43% shorter than that of the other five comparative algorithms, respectively. The path running time of the improved A* algorithm is 55.12s, which is 26.60%, 4.64%, 4.83%, 4.44%, and 2.51% less than that of the other five comparative algorithms, respectively. The number of search nodes is 83, which is also significantly lower than that of the other five algorithms.

Table 2. Comparison of simulation results of different algorithms in 20×20 map (global planning).

Map size	Algorithm	Length/m	Moving time/s	Search nodes
20×20	Dijkstra	27.92	75.10	380
	A*	27.90	57.80	125
	LPA*	27.93	57.92	132
	IDA*	27.88	57.68	118
	Theta*	27.77	56.54	98
	ImpA* (Ours)	27.65	55.12	83

4.2. Simulation experiments analysis of local and fused algorithm

To verify the performance of the optimized ImpDWA local planning and fused algorithm proposed in this paper, based on the aforementioned 30×30 and 20×20 grid maps, 2 unknown static obstacles and 1 random dynamic obstacle with a speed of 0.8–1.0m/s were introduced respectively to construct a dynamically complex environment. Static obstacles simulate sudden fixed barriers during robot movement, and dynamic obstacles simulate moving targets in the scene, which is close to the complexity of real operating scenarios. The traditional DWA algorithm and the ImpDWA local obstacle avoidance algorithm proposed in this paper were selected to fuse with Dijkstra, A*, LPA*,

IDA*, Theta*, and ImpA* respectively to form 12 fusion schemes. Each scheme was tested through 120 simulation experiments, with average path length, running time, and obstacle avoidance success rate as evaluation indicators to comprehensively assess the dynamic environment adaptability and practical performance of the fused algorithm. The specific experimental analysis results are as follows.

4.2.1. Analysis of the effectiveness of local algorithm improvement

Taking the global path as the reference trajectory, the independent local obstacle avoidance performance of the traditional DWA and the ImpDWA was compared. The results show that ImpDWA solves the problem of trajectory oscillation of the traditional DWA algorithm when dynamic obstacles approach by dynamically adjusting the constraint coefficients of the speed window and angle window, and optimizing the proportion of obstacle avoidance safety weight in the evaluation function. In the 30×30 dynamic obstacle map, the traditional DWA responds slowly to dynamic obstacles and usually travels along the edge of obstacles, which increases the risk of collision with dynamic obstacles (Figures 7(a) and 7(c)). In contrast, ImpDWA predicts the movement trend of dynamic obstacles in advance and adjusts the steering angle to achieve obstacle avoidance without retracement (Figure 7). In the 20×20 dynamic obstacle map environment (Figure 8), the traditional DWA results in obstacle avoidance paths close to the edge of obstacles (safety distance < 0.3m) due to the fixed angle window, while ImpDWA dynamically expands the safety margin through obstacle distance feedback (safety distance ≥ 0.5m), significantly reducing the collision risk.

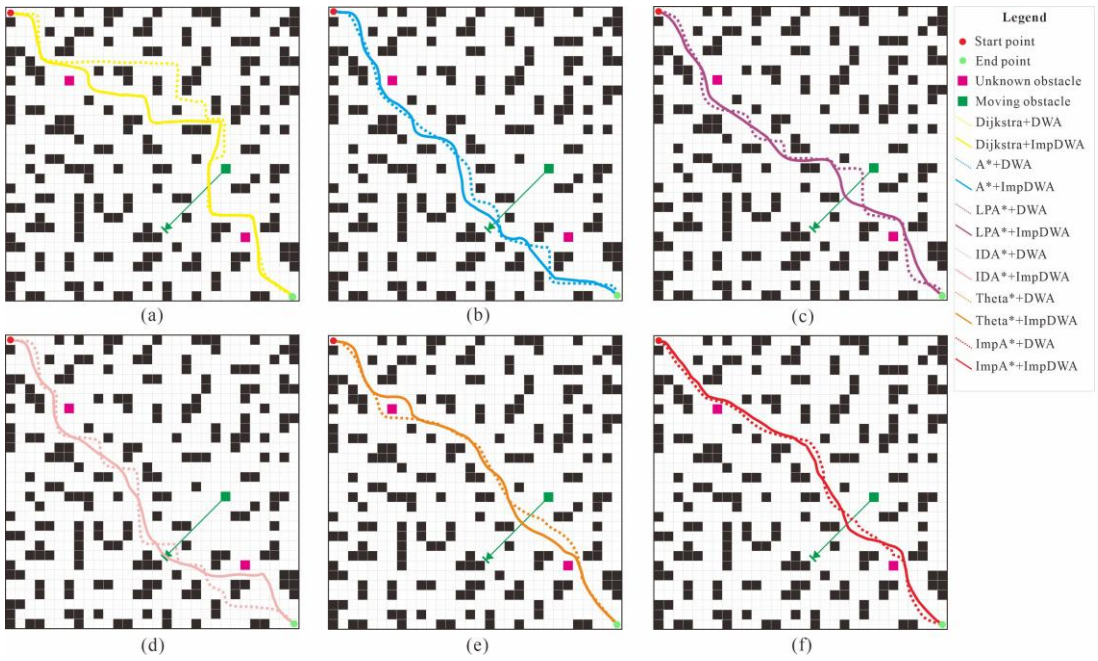

Figure 7. Obstacle avoidance trajectory comparison of fusion algorithms in 30×30 dynamic map.

At the same time, the obstacle avoidance success rate of ImpDWA in each fused algorithm is improved compared with the traditional DWA. In the 30×30 and 20×20 maps, Dijkstra-ImpDWA

after local optimization is 3.3% and 2.5% higher than Dijkstra-DWA respectively; A*-ImpDWA is 2.5% higher than A*-DWA in both maps; LPA*-ImpDWA is 4.1% and 4.2% higher than LPA*-DWA respectively; IDA*-ImpDWA is 1.6% and 1.7% higher than IDA*-DWA respectively; Theta*-ImpDWA is 2.5% higher than Theta*-DWA in both maps; the proposed ImpA*-ImpDWA is 2.5% higher than ImpA*-DWA in both maps.

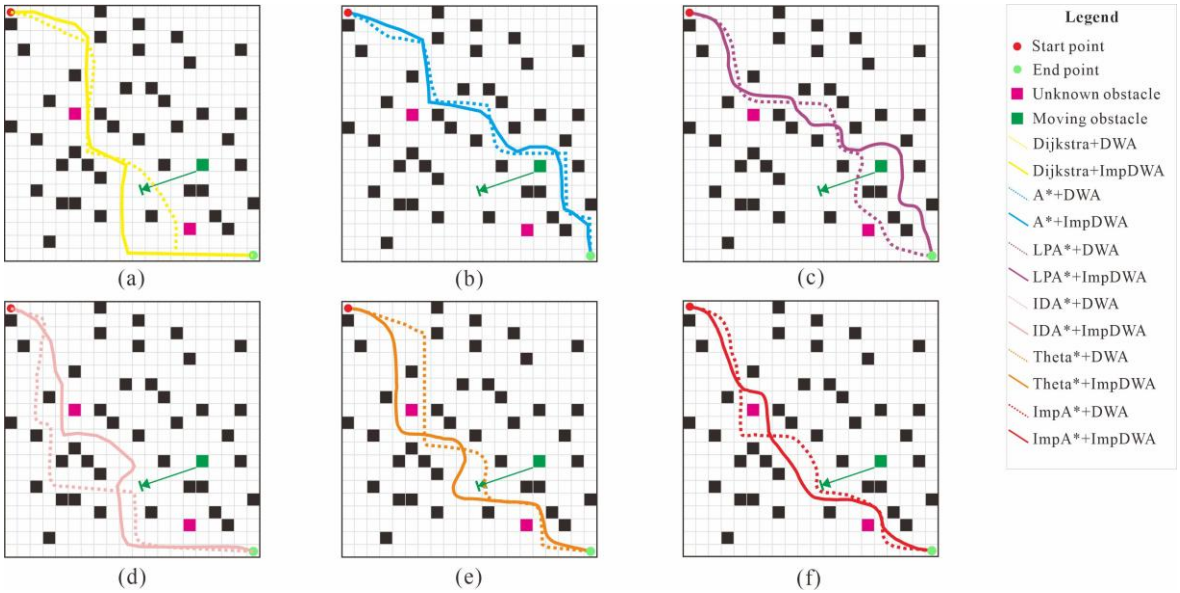

Figure 8. Obstacle avoidance trajectory comparison of fusion algorithms in 20×20 dynamic map.

4.2.2. Comprehensive performance comparison of fused algorithms

After fusing the DWA local algorithm (before and after improvement) with the six global algorithms respectively, the comprehensive performance in the dynamic environment shows significant differences as follows:

(1) Obstacle avoidance success rate

The quantitative data in Tables 3 and 4 indicate that the ImpA*-ImpDWA fusion scheme achieves the optimal obstacle avoidance success rate. Among all fusion schemes, the ImpA*-ImpDWA fusion scheme based on the mathematically optimized improved algorithm proposed in this paper achieves 96.7% and 97.5% obstacle avoidance success rates in the two map sizes, respectively.

In the fusion schemes based on the traditional DWA, even when combined with high-performance global algorithms such as ImpA*, Theta*, and IDA*, the obstacle avoidance success rate is still not higher than 95%. For example, in the 30×30 map, ImpA*-DWA is 94.2%, Theta*-DWA is 93.3%, and IDA*-DWA is 91.7%; in the 20×20 map, ImpA*-DWA is 95%, Theta*-DWA is 94.2%, and IDA*-DWA is 92.5%. The main reason is that the traditional DWA has insufficient ability to predict dynamic obstacles, making it prone to collisions with sudden unknown obstacles at the inflection points of the global path.

Table 3. Comparison of simulation results of fusion algorithms in 30×30 dynamic map.

Map size	Algorithms	Length/m	Moving time/s	Obstacle avoidance rate
30×30	Dijkstra-DWA	45.86	124.75	86.7%
	Dijkstra-ImpDWA	44.92	119.20	90.0%
	A*-DWA	44.73	93.84	91.7%
	A*-ImpDWA	43.95	90.96	94.2%
	LPA*-DWA	44.85	94.18	89.2%
	LPA*-ImpDWA	44.12	91.54	93.3%
	IDA*-DWA	44.67	93.66	91.7%
	IDA*-ImpDWA	43.88	90.78	93.3%
	Theta*-DWA	44.21	92.10	93.3%
	Theta*-ImpDWA	43.56	89.96	95.8%
	ImpA*-DWA	43.78	89.46	94.2%
	ImpA*-ImpDWA (Ours)	43.27	87.12	96.7%

Among the fusion schemes based on ImpDWA, except for the proposed ImpA*-ImpDWA, the obstacle avoidance success rates of other fusion algorithms such as Theta*-ImpDWA, IDA*-ImpDWA, LPA*-ImpDWA, and A*-ImpDWA are 95.8%, 93.3%, 93.3%, and 94.2% respectively (in the 30×30 map). However, due to the low global path search efficiency of Dijkstra-ImpDWA, the response of the local algorithm is delayed, resulting in an obstacle avoidance success rate of only 90%. This highlights the importance of performance matching between global and local algorithms: an inefficient global algorithm will limit the response speed of local obstacle avoidance, and even when combined with an improved local algorithm, it is difficult to achieve the optimal effect.

Table 4. Comparison of simulation results of fusion algorithms in 20×20 dynamic map.

Map size	Algorithms	Length/m	Moving time/s	Obstacle avoidance rate
20×20	Dijkstra-DWA	30.65	82.95	88.3%
	Dijkstra-ImpDWA	29.87	79.40	90.8%
	A*-DWA	29.72	62.74	92.5%
	A*-ImpDWA	29.35	61.26	95.0%
	LPA*-DWA	29.81	62.96	90.0%
	LPA*-ImpDWA	29.28	61.02	94.2%
	IDA*-DWA	29.68	62.60	92.5%
	IDA*-ImpDWA	29.21	60.80	94.2%
	Theta*-DWA	29.43	61.70	94.2%
	Theta*-ImpDWA	29.05	60.44	96.7%
	ImpA*-DWA	29.12	59.62	95.0%
	ImpA*-ImpDWA (Ours)	28.93	58.88	97.5%

(2) Path length

Data in Tables 3 and 4 show that the path length of the ImpA*-ImpDWA fusion scheme is 43.27m in the 30×30 map and 28.93m in the 20×20 map, both being the shortest among all fusion

schemes.

Compared with traditional global-local fusion schemes (e.g., Dijkstra-DWA), the path length of ImpA*-ImpDWA is shortened by 5.2%~5.61%. One reason is that the globally optimized ImpA* algorithm generates a smoother global path with low redundancy, reserving better adjustment space for local obstacle avoidance. The other reason is that the trajectory sampling strategy of ImpDWA avoids the detour problem of traditional DWA caused by obstacle avoidance; by dynamically adjusting velocity and angle, it maximally adheres to the globally optimal trajectory while evading obstacles.

When comparing schemes with the same global algorithm but different local algorithms, taking ImpA*-DWA and ImpA*-ImpDWA as examples. The introduction of ImpDWA shortens the path length by an average of 0.19–0.51m, verifying the role of the improved local algorithm in reducing trajectory redundancy. When the same local algorithm is combined with different global algorithms, schemes using ImpA* as the global layer generally have shorter path lengths than those using other global algorithms, reflecting the fundamental advantage of the improved ImpA* algorithm in global path planning.

(3) Running time

Factors affecting the planning time include global path search efficiency, the number of local trajectory samples, and data interaction delay between the two. The running time of the ImpA*-ImpDWA fused algorithm is 87.12s in the 30×30 map, which is 3.16%~30.16% shorter than that of other schemes on average; in the 20×20 map, it is 58.88s, 2.58%~29.02% shorter than other schemes on average. The main reasons are as follows:

At the global layer, ImpA* significantly reduces the number of search nodes through search neighborhood optimization and dynamic weight adjustment. Compared with the traditional A* algorithm, the number of search nodes is reduced by 39.44% and 33.60% in the 30×30 and 20×20 grid maps, respectively, reserving more sufficient computing time for the local algorithm.

At the local layer, ImpDWA reduces the number of invalid sampling points by optimizing the trajectory sampling window, and simplifies the calculation logic of the evaluation function, thereby reducing the time overhead of local planning.

4.2.3. Performance ranking of fused algorithms

Based on the comprehensive evaluation of the three indicators, the performance ranking of the 12 fusion schemes from best to worst in the dynamic environment is as follows: ImpA*-ImpDWA ＞ Theta*-ImpDWA ＞ A*-ImpDWA ＞ ImpA*-DWA ＞ LPA*-ImpDWA ＞ IDA*-ImpDWA ＞ Theta*-DWA ＞ A*-DWA ＞ LPA*-DWA ＞ IDA*-DWA ＞ Dijkstra-ImpDWA ＞ Dijkstra-DWA.

The key conclusions are summarized as follows:

(1) The performance improvement of the local algorithm is crucial to the overall effect of the fusion scheme. Compared with the traditional DWA, ImpDWA can significantly improve the obstacle avoidance success rate, shorten the path length and planning time, and is a key link in optimizing the performance of the fused algorithm;

(2) The efficiency and path quality of the global algorithm directly determine the upper limit of the fusion scheme. Even when combined with an improved local algorithm, inefficient global algorithms (e.g., Dijkstra) are still difficult to meet the real-time requirements of dynamic environments. In contrast, the collaboration between high-performance global algorithms (e.g., ImpA*, Theta*) and ImpDWA can maximize the fusion advantages;

(3) The ImpA*-ImpDWA fused algorithm proposed in this paper achieves collaborative optimization of the safety and smoothness of the global path and the dynamic adaptability of local obstacle avoidance. In dynamic environments, it demonstrates comprehensive performance with an obstacle avoidance success rate of 96.7%~97.5%, the shortest path length, and optimal planning time. This verifies the effectiveness and practicality of the fused algorithm, providing a reliable solution for the autonomous navigation of mobile robots in complex dynamic environments.

4.3. Performance verification of the fused algorithm in large-scale dynamically complex scenarios

The previous experiments have fully confirmed the effectiveness of the proposed ImpA*-ImpDWA fused algorithm. In small-to-medium-scale dynamically complex scenarios (20×20 and 30×30 grids), it achieves optimal performance across three core indicators: obstacle avoidance success rate, path length, and running time. To further verify the stability and adaptability of the algorithm in larger-scale scenarios with higher obstacle density, comparative experiments were designed based on a 100×100 large-scale grid environment. The specific experimental setup and result analysis are as follows:

4.3.1. Experimental scenario and parameter design

Continuing the basic parameter system of previous experiments, the settings were optimized for the characteristics of large-scale scenarios: (1) Scenario scale: A 100×100 grid map was adopted, with the grid size remaining 1m×1m; (2) Obstacle configuration: 40 unknown static obstacles were randomly distributed in the map, and 15 dynamic obstacles were introduced with a moving speed range of 0.8-1.0m/s. The motion trajectories of dynamic obstacles were random and irregular, simulating a complex environment with multi-target dynamic interference in large-scale scenarios; (3) Comparative algorithms: The ImpA*-ImpDWA proposed in this paper was selected for comparison with 5 other fused algorithms, including Dijkstra-ImpDWA, A*-ImpDWA, LPA*-ImpDWA, IDA*-ImpDWA, and Theta*-ImpDWA, to ensure the consistency and comparability of the experiments; (4) Experimental scheme and evaluation indicators: Each fused algorithm was tested through 120 independent simulation experiments. The core evaluation indicators included average path length, average running time, and obstacle avoidance success rate. The average value of multiple experiments was adopted to eliminate random errors.

4.3.2. Experimental results and quantitative analysis

The simulation results are shown in Figure 9. Table 5 presents the comparative data of different fused algorithms, and the quantitative comparison fully verifies the superiority of the proposed algorithm in large-scale scenarios:

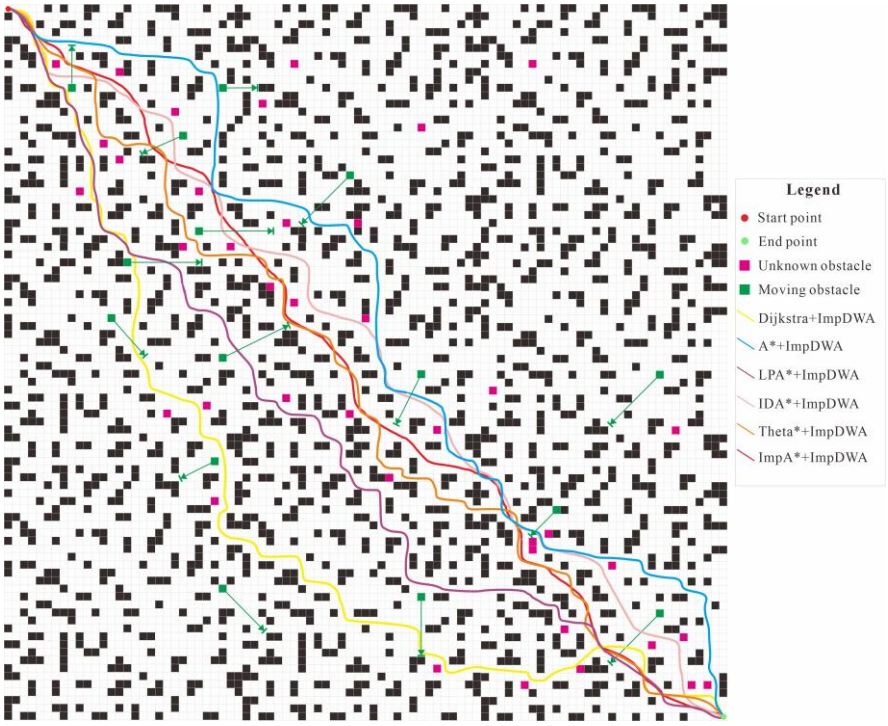

Figure 9. Obstacle avoidance trajectory comparison of fusion algorithms in 100×100 dynamic map.

Table 5. Comparison of simulation results in 100×100 dynamic map.

Map size	Algorithms	Length/m	Moving time/s	Obstacle avoidance rate
100×100	Dijkstra-ImpDWA	172.86	417.37	79.3%
	A*-ImpDWA	159.74	353.59	89.6%
	LPA*-ImpDWA	156.21	356.24	91.8%
	IDA*-ImpDWA	153.52	350.68	89.1%
	Theta*-ImpDWA	155.48	348.42	94.2%
	ImpA*-ImpDWA (Ours)	149.23	336.71	96.5%

Based on the data in table 5, the result analysis are as follow:

(1) Obstacle avoidance success rate: The ImpA*-ImpDWA fused scheme achieves an obstacle avoidance success rate of up to 96.5% in the 100×100 large-scale, multi-obstacle scenario, which is 2.3 percentage points higher than the second-best performer Theta*-ImpDWA (94.2%) and 17.2 percentage points higher than the worst-performing traditional fused scheme Dijkstra-ImpDWA (79.3%). This indicates that even in large-scale scenarios with expanded obstacle distribution and increased dynamic interference, the proposed fused algorithm still exhibits strong adaptability to complex environments and anti-interference capabilities.

(2) Average path length: The average path length planned by ImpA*-ImpDWA is 149.23m, the shortest among all comparative schemes. Compared with the traditional fused scheme Dijkstra-ImpDWA (172.86m), it is shortened by 23.63m (a reduction ratio of 13.67%); compared with the high-performance comparative scheme Theta*-ImpDWA (155.48m), it is shortened by 6.25m (a reduction ratio of 4.02%). This result confirms the effectiveness of the dynamic heuristic function and neighborhood search optimization of the ImpA* algorithm: in the massive grid nodes of

large-scale scenarios, it can quickly focus on the optimal path direction and reduce invalid searches.

(3) Average running time: The average running time of the fused algorithm is only 336.71s, which is significantly shorter than other comparative schemes: 75.66s less than Dijkstra-ImpDWA (412.37s, a reduction ratio of 18.3%); 11.71s less than Theta*-ImpDWA (348.42s, a reduction ratio of 3.36%); even compared with IDA*-ImpDWA, the running time is shortened by 13.97s (a reduction ratio of 3.98%). The core reason is that the ImpA* algorithm greatly reduces the number of search nodes in large-scale scenarios through optimized pruning strategies, thereby lowering the computational overhead of global path planning.

4.3.3. Cross-scenario adaptability and research extension

We conducted a comprehensive analysis of the experimental results across different scenarios: small-to-medium-scale grids (20×20 and 30×30) and a large-scale grid (100×100). The proposed ImpA*-ImpDWA fused algorithm delivers excellent performance in small-to-medium-scale environments. In dynamically complex settings with larger scales and higher obstacle densities, it still maintains key comprehensive advantages: high obstacle avoidance success rate, shortest path length, and optimal running time. This performance consistency highlights the algorithm's outstanding cross-scenario adaptability. This feature provides reliable experimental support and technical guarantees for the practical application of the algorithm in dynamic scenarios such as large-scale warehouse robot scheduling.

It is worth noting that the cross-system applicability and multi-scenario adaptability of an algorithm are core dimensions for measuring its research value, as well as an extended direction of the previous experimental conclusions. To further expand the application boundaries of the algorithm, future research will conduct simulation verification for more typical real-world scenarios, such as humanoid motion decision-making systems in UAV trap environments [31], autonomous obstacle avoidance tasks of unmanned surface vessel swarms in marine environments [32], and path planning systems for high-rise building fire rescue [33]. Through multi-scenario and multi-task verification, the robustness and practicality of the algorithm will be continuously optimized, providing a generalized solution for autonomous navigation problems in different fields.

5. Conclusions

From the perspective of mathematical optimization, this paper constructs a hybrid path planning framework. It deeply integrates the ImpA* algorithm and ImpDWA. In the global planning phase, three key optimizations are adopted. First, environmental complexity is quantified through modeling. Second, heuristic functions use dynamic weight adjustment. Third, path smoothing strategies are applied. These measures enhance both the rationality of the global path and its search efficiency. In the local obstacle avoidance phase, the DWA evaluation system is upgraded. A target-oriented cost term is added, and weight coefficients are dynamically adjusted. This solves critical problems of traditional methods. For instance, it avoids local minima and improves target guidance. The dual algorithms are collaboratively designed. They form a closed-loop mechanism of global optimization-local correction. Simulation experiments provide clear verification. The proposed strategy outperforms existing mainstream schemes significantly. Advantages are observed in path quality, planning efficiency, and dynamic obstacle avoidance reliability. It offers theoretical value and technical support for autonomous robot navigation in complex environments.

This research still has limitations. Algorithm performance is only verified via the MATLAB simulation platform. Practical applications involve various uncertain factors. These include sensor measurement noise, positioning errors, uneven ground, hardware response delays, and changes in environmental illumination. None of these factors are fully considered in the current study. Future research will focus on two key aspects. First, it will complete the transplantation and deployment of the fusion algorithm on a physical robot platform. Hardware adaptation and debugging will also be carried out. This opens up the transformation path from simulation verification to practical application. Second, a real and complex experimental environment incorporating multi-source interference is constructed to systematically evaluate the algorithm's adaptability in other environmental scenarios.

Author contributions

Le Gao: Conceptualization, validation, formal analysis, writing-original draft preparation, supervision, project administration; Yuying Zhang: Software, visualization; Pinjie Liu: Methodology, investigation, data curation, writing-review and editing, funding acquisition; Xiaoying Ou: Software; Jinglong Cheng: Software; Ying Zhu: Software, resources. All authors have read and agreed to the published version of the manuscript.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This research was funded by "Department of Science and Technology of Guangdong Province, Pdjh2024a515", "Department of Education of Guangdong Province, 2024ZDZX1030" and "Guangdong Education Research Project, 24GYB119". Thanks for the support for the Guangzhou Huali College.

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. M. A. Ferreira, L. C. Moreira, A. M. Lopes, Autonomous navigation system for a differential drive mobile robot, *J. Test. Eval.*, **52** (2024), 841–852. https://doi.org/10.1520/JTE20230191
2. A. Amin, X. C. Wang, Y. N. Zhang, T. H. Li, Y. Y. Chen, J. M. Zhang, A comprehensive review of applications of robotics and artificial intelligence in agricultural operations, *Stud. Inform. Control*, **32** (2023), 59–70.
3. T. Chen, S. Q. Li, Z. P. Zeng, Z. H. Liang, Y. X. Chen, W. S. Guo, An empirical investigation of users' switching intention to public service robots: From the perspective of PPM framework, *Gov. Inform. Q.*, **41** (2024), 101933. https://doi.org/10.1016/j.giq.2024.101933

4. H. W. Qin, S. L. Shao, T. Wang, X. T. Yu, Y. Jiang, Z. H. Cao, Review of autonomous path planning algorithms for mobile robots, *Drones*, **7** (2023), 211. https://doi.org/10.3390/drones7030211
5. O. Misir, Dynamic local path planning method based on neutrosophic set theory for a mobile robot, *J. Braz. Soc. Mech. Sci. Eng.*, **45** (2023), 127. https://doi.org/10.1007/s40430-023-04048-6
6. Q. Zhang, J. Zhao, L. Pan, X. Wu, Y. Y. Hou, X. Q. Qi, Optimal path planning for mobile robots in complex environments based on the gray wolf algorithm and self-powered sensors, *IEEE Sens. J.*, **23** (2023), 20756–20765. https://doi.org/10.1109/JSEN.2023.3252635
7. L. Gao, J. Z. Zhang, J. X. Yu, X. Zhang, Z. Q. Zeng, BPA: A decentralized payment system that balances privacy and auditability, *AIMS Mathematics*, **9** (2024), 6183–6206. https://doi.org/10.3934/math.2024302
8. X. M. Liu, X. H. Chang, L. W. Hou, Attack-dependent adaptive event-triggered security fuzzy control for nonlinear networked cascade control systems under deception attacks, *Mathematics*, **12** (2024), 3385. https://doi.org/10.3390/math12213385
9. X. He, C. Guo, Research on multi-strategy fusion of the chimpanzee optimization algorithm and its application in path planning, *Appl. Sci.*, **15** (2025), 608. https://doi.org/10.3390/app15020608
10. B. Guo, Z. Kuang, J. H. Guan, M. T. Hu, L. X. Rao, X. Q. Sun, An improved A-star algorithm for complete coverage path planning of unmanned ships, *Int. J. Pattern Recogn.*, **36** (2022), 2259009. https://doi.org/10.1142/S0218001422590091
11. J. Jason, M. Siever, A. Valentino, K. M. Suryaningrum, R. Yunanda, Dijkstra's algorithm to find the nearest vaccine location, *Procedia Computer Science*, **216** (2023), 5–12. https://doi.org/10.1016/j.procs.2022.12.105
12. Y. Zhang, L. L. Li, H. C. Lin, Z. W. Ma, J. Zhao, Development of path planning approach using improved a-star algorithm in AGV system, *J. Internet Technol.*, **20** (2019), 915–924. https://doi.org/10.3966/160792642019052003023
13. C. G. Liu, Q. Z. Mao, X. M. Chu, S. Xie, An improved a-star algorithm considering water current, traffic separation and berthing for vessel path planning, *Appl. Sci.*, **9** (2019), 1057. https://doi.org/10.3390/app9061057
14. Y. Dai, W. J. Lv, S. K. Li, M. Y. Zong, Improving the Lifelong Planning A-star algorithm to satisfy path planning for space truss cellular robots with dynamic obstacles, *Robotica*, **43** (2025), 1243–1257. https://doi.org/10.1017/S0263574725000256
15. W. Z. Li, J. J. Liu, S. L. Yao, An improved Dijkstra's algorithm for shortest path planning on 2D grid maps, *2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, Beijing, China, 2019, 438–441. https://doi.org/10.1109/iceiec.2019.8784487
16. C. G. Liu, K. Zhang, Z. B. He, L. H. Lai, X. M. Chu, Clustering Theta* based segmented path planning method for vessels in inland waterways, *Ocean Eng.*, **309** (2024), 118249. https://doi.org/10.1016/j.oceaneng.2024.118249
17. M. Kobayashi, N. Motoi, Local path planning: dynamic window approach with virtual manipulators considering dynamic obstacles, *IEEE Access*, **10** (2022), 17018–17029. https://doi.org/10.1109/ACCESS.2022.3150036
18. R. Zhou, K. Zhou, L. N. Wang, B. R. Wang, An improved dynamic window path planning algorithm using multi-algorithm fusion, *Int. J. Control Autom. Syst.*, **22** (2024), 1005–1020. https://doi.org/10.1007/s12555-022-0495-8

19. M. Yao, H. G. Deng, X. Y. Feng, P. G. Li, Y. F. Li, H. Y. Liu, Improved dynamic window approach based on energy consumption management and fuzzy logic control for local path planning of mobile robots, *Comput. Ind. Eng.*, **187** (2024), 109767. https://doi.org/10.1016/j.cie.2023.109767
20. J. Moon, B. Y. Lee, M. J. Tahk, A hybrid dynamic window approach for collision avoidance of VTOL UAVs, *Int. J. Aeronaut. Space Sci.*, **19** (2018), 889–903. https://doi.org/10.1007/s42405-018-0061-z
21. M. E. Miyombo, Y. K. Liu, C. M. Mulenga, A. Siamulonga, M. C. Kabanda, P. Shaba, et al., Optimal path planning in a real-world radioactive environment: A comparative study of A-star and Dijkstra algorithms, *Nucl. Eng. Des.*, **420** (2024), 113039. https://doi.org/10.1016/j.nucengdes.2024.113039
22. L. Morin, P. Gilormini, K. Derrien, Generalized euclidean distances for elasticity tensors, *J. Elast.*, **138** (2020), 221–232. https://doi.org/10.1007/s10659-019-09741-z
23. X. Liu, W. T. Chen, L. Peng, D. Luo, L. K. Jia, G. Xu, et al., Secure computation protocol of Chebyshev distance under the malicious model, *Sci. Rep.*, **14** (2024), 17115. https://doi.org/10.1038/s41598-024-67907-9
24. C. D. Wang, J. L. Yang, B. Q. Zhang, A fault diagnosis method using improved prototypical network and weighting similarity-Manhattan distance with insufficient noisy data, *Measurement*, **226** (2024), 114171. https://doi.org/10.1016/j.measurement.2024.114171
25. Z. B. Zeng, H. Dong, Y. L. Xu, W. Zhang, H. C. Yu, X. P. Li, Teaching-learning-based optimization algorithm with dynamic neighborhood and crossover search mechanism for numerical optimization, *Appl. Soft Comput.*, **154** (2024), 111332. https://doi.org/10.1016/j.asoc.2024.111332
26. J. Ahmad, M. Wahab, Enhancing the safety and smoothness of path planning through an integration of Dijkstra's algorithm and piecewise cubic Bezier optimization, E*xpert Syst. Appl.*, **289** (2025), 128315. https://doi.org/10.1016/j.eswa.2025.128315
27. Y. D. Ji, Q. Q. Liu, C. Zhou, Z. J. Han, W. Wu, Hybrid swarm intelligence and human-inspired optimization for urban drone path planning, *Biomimetics*, **10** (2025), 180. https://doi.org/10.3390/biomimetics10030180
28. H. F. Bao, J. Fang, C. H. Wang, Z. B. Li, J. S. Zhang, C. S. Wang, Research on static/dynamic global path planning based on improved A* algorithm for mobile robots, *J. Robot.*, **2023** (2023), 5098156. https://doi.org/10.1155/2023/5098156
29. B. P. Wang, D. Y. Ju, F. Z. Xu, C. Feng, G .L. Xun, CAF-BRT*: A 2D path planning algorithm based on circular arc fillet method, *IEEE Access*, **10** (2022), 127168–127181. https://doi.org/10.1109/ACCESS.2022.3226465
30. M. Y. Wu, E. L. M. Su, C. F. Yeong, B. Dong, W. Holderbaum, C. G. Yang, A hybrid path planning algorithm combining A* and improved ant colony optimization with dynamic window approach for enhancing energy efficiency in warehouse environments, *PeerJ Computer Science*, **10** (2024), e2629. https://doi.org/10.7717/peerj-cs.2629
31. J. T. Chen, Q. Zhou, H. R. Ren, H. Y. Li, Partition and planning: a human-like motion decision for UAV in trap environment, *Sci. China Technol. Sci.*, **67** (2024), 1226–1237. https://doi.org/10.1007/s11431-023-2605-7
32. Y. H. Li, J. W. Ye, L. Gao, M. Cai, CoDAC: autonomous obstacle avoidance optimization for unmanned surface vehicle clusters via multi-modal dynamic perception and collaborative community detection, *IEEE Access*, **13** (2025), 134552–134569. https://doi.org/10.1109/ACCESS.2025.3593236

33. J. T. Chen, H. R. Ren, Q. Zhou, H. Y. Li, Fast unfolding-based indoor space partitioning and rapid complementary search planning for high-rise fire rescue, *IEEE T. Autom. Sci. Eng.*, **22** (2025), 18750–18760. https://doi.org/10.1109/TASE.2025.3590419

AIMS Press