EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

.

A hybrid enhanced A* and DWA approach for dynamic path planning of manufacturing logistics robots

Xiaoming Zhang[1,2*]

[1]Modern Police Technology and Equipment Research Center of China People's Police University, Langfang, Hebei 065000, China

[2]Hebei Provincial Key Laboratory of Low-altitude Safety Control Technology, Langfang, Hebei 065000, China

Abstract

Smart manufacturing and logistics robots require integrated navigation capabilities. Optimized global routing, fast obstacle avoidance, and stable trajectory generation are all necessary. In this paper, we develop a hybrid path planner. An enhanced A* algorithm is combined with the Dynamic Window Approach (DWA). We intend to satisfy such practical demands. Four key aspects are addressed to optimize the conventional A* algorithm. Evaluation function optimization, neighborhood search optimization, Floyd based path smoothing, and segmented path processing. Path smoothing is also enhanced. These modifications cut computational costs. Invalid node expansion is reduced. Sudden steering changes are mitigated as well. Comparative simulations on two test maps were carried out. The superiority of our improved method was validated. The results show that path turns drop by 33.3% and 37.5%, respectively. Search time is reduced by 22.2% and 31.3%, respectively. Unlike pure static planning strategies, the hybrid framework delivers faster system response. Improved real-time obstacle avoidance capability is also achieved. Consequently, it fits well with practical industrial scenarios. Automated material handling and intelligent warehouse logistics in smart factories are covered.

Keywords: manufacturing logistics robots, enhanced A* algorithm, DWA algorithm, Path planning

Received on 28 July 2026, accepted on 09 September 2026, published on 22 September 2026

doi: 10.4108/eetsis.14220

1. Introduction

In recent years, mobile robots have found increasingly broad application in service operations, healthcare, logistics, and other related fields. As a key element of autonomous navigation, path planning is meant to identify a viable trajectory so that the robot can reach its intended destination. In complex operational settings, robots often have only incomplete knowledge of the surrounding environment, and therefore they need robust planning strategies together with effective mechanisms for obstacle avoidance [1]. Therefore, investigating path planning strategies for mobile robots is of both theoretical interest and practical relevance [2]. In current research, scholars generally sort path planning methods by observing the motion traits of surrounding obstacles. Such a classification system yields two major research branches in this field. One branch handles path planning problems in completely static environments, and the other branch concentrates on effective avoidance against dynamically moving obstacles [3]. In academic communities, static environment planning is conventionally known as global path planning. A large set of classical algorithms fall within this scope, including Dijkstra's algorithm [4], D* [5], A* [6], ant colony optimization [7], Rapidly exploring random trees (RRT) [8], and particle swarm optimization [9]. In contrast, the research direction of dynamic obstacle avoidance belongs to local path planning. It serves as a key technical solution for real-time obstacle evasion. Through literature review, we found that the dynamic window approach (DWA) [10], artificial potential fields [11], and genetic algorithms [12] are

*Corresponding author. Email: zhangxiaoming@cppu.edu.cn

the most commonly adopted tools to tackle such dynamic planning tasks. Among the global planners, A* is fairly widely adopted because its heuristic search strategy can guide route discovery in an efficient manner. Nevertheless, the method can suffer from inefficient node expansion and tends to generate paths with an excessive number of directional changes. Such drawbacks have motivated a considerable body of research aimed at enhancing A* through various optimization strategies.

Manufacturing logistics robots have drawn substantial attention as a research topic within robotics. Both theoretical significance and practical application value can be attached to this area. For complex logistics scenarios, developing a path planning method that is efficient, accurate, and safe still constitutes an important challenge. Point-to-point path planning for these robots is not entirely the same as that for general mobile robots. In most cases, logistics robots are deployed in areas with high pedestrian density and are required to respond to emergency events under rather strict safety constraints. During their movement, they must retain a strong capacity for obstacle avoidance, and at the same time, they should not pose any threat to the safety of nearby pedestrians.

Recent work has fused A* with DWA to overcome the limits of single-algorithm planning. This has led to various hybrid frameworks, all aiming for better performance. Xu et al. [13], for instance, extended the A* search to 48 directions. That broadens the search and shortens paths to some degree, but we found that per-node computation rises, which lowers overall efficiency. Min et al. [14], by contrast, took a different tack. They added curvature constraints to the cost function, reducing unnecessary turns and producing smoother routes. Nevertheless, redundant nodes still remain, and that restricts the optimization gain. As for Baik et al. [15], they used a multi factor weighting that accounts for collision risk, path length, and terrain adaptability. Their results show improved adaptability, yet the method also gives longer average paths, lower efficiency, and more tortuous trajectories.

The DWA algorithm sees wide use in dynamic obstacle path planning, thanks to its computational efficiency and real-time stability. Still, it does have certain weak points. One is that it cannot handle moving obstacles very well. Another is that its performance shifts noticeably with parameter changes. Several groups have tried to fix these flaws. Yi et al. [16], for example, merged an optimized social force model with DWA to predict pedestrian paths. That works, but it is computationally heavy. It also cannot adjust to individual movement variations in complex scenes. Li et al. [17] used TOACA key nodes to steer a modified DWA. This does improve dynamic avoidance, yet it still has clear issues high resource use, strong reliance on the global path, and sensitivity to parameters. Zhang et al. [18] embedded a fuzzy control scheme into DWA. Their method adjusts evaluation weights on the fly and works well under normal conditions. But in dense obstacle settings, the gain shrinks. Safety margins drop, and search time rises sharply.

Overall, existing optimized A* and DWA algorithms still suffer from unresolved technical defects that restrict real-world deployment. Practical robot operating environments are highly complex and filled with dynamic obstacles, where single global planning algorithms cannot satisfy the strict real-time and safety requirements of robot navigation. Accordingly, A*-DWA hybrid planning has become a mainstream research direction to address such challenges. Even so, current hybrid systems still perform poorly under extreme environmental complexity. To further promote hybrid planning performance, Zhang et al. [19] optimized the A* exploration logic prior to DWA integration. The enhanced A* algorithm reduces search ranges and removes redundant waypoints, effectively enhancing the hybrid system's overall operational performance. However, the fixed static heuristic weights limit the algorithm search efficiency and adaptability in complex, dynamically changing working environments.

From the above analysis, it becomes clear that existing path planning methods still have some limitations. These include excessive node expansion, redundant turning points, and insufficient local maneuverability. As a result, logistics efficiency may be reduced, and safety risks could increase. To address these issues, we propose in the current work a hybrid planning framework that is specifically designed for manufacturing logistics robots. This framework combines the A* search algorithm with the dynamic window approach (DWA). At the global planning level, we introduce a dynamically adjusted cost function, together with a goal oriented five-neighborhood exploration strategy. These measures are intended to boost search efficiency and to cut down on unnecessary node expansions. In addition, we apply a Floyd based optimization procedure to get rid of superfluous turning points along the path. The refined global trajectory is then fed into the DWA module. In this way, real time collision avoidance can be achieved, even when the environment is dynamic. Therefore, the main contribution of this work is a logistics oriented planning strategy that improves search efficiency, path compactness, and execution stability for manufacturing logistics robots.

To tackle the issues in existing A*-DWA hybrids too many expanded nodes, excessive turns, and poor dynamic adaptability. We propose a new planning framework specifically for manufacturing logistics robots. Our novelty is not a single heuristic but three combined enhancements to the standard A* algorithm.

Introduce a dynamic weight that changes with the remaining distance to the goal. When far away, we apply a larger weight to accelerate exploration; as the robot gets closer, we reduce the weight to prioritize optimality. This adaptive balance cuts redundant expansions and outperforms fixed-weight strategies.

Adopt a goal directed five neighborhood search that prunes exploration to only five directions aligned with the current heading toward the target. If those are blocked, the algorithm falls back to the remaining directions, preserving completeness. This reduces per-node evaluation from eight to five successors, speeding up planning at the cost of a slight loss in strict global optimality a trade off we consider acceptable for time critical logistics tasks.

Apply a Floyd based smoothing procedure that removes redundant waypoints and eliminates sharp turns by checking line of sight connectivity between non adjacent nodes under obstacle clearance constraints. The resulting global path is more compact and smoother, which facilitates stable tracking and reduces steering adjustments during local DWA execution.

These three modules are designed to work together, balancing global optimality, computational efficiency, and trajectory smoothness in dynamic environments. The refined global path then serves as a reference for the DWA local planner, which samples velocity space and selects the best trajectory via a weighted cost function for real time collision avoidance.

2. Enhanced A* Algorithm and DWA Algorithm

2.1 Principle of the A* Algorithm

For the implementation of the A* pathfinding algorithm, we have made use of a graph based modeling scheme. The main objective here is to find the shortest path that is feasible, from a start node to a target node, both of which are predefined in the graph. As has been reported in earlier studies [20], the A* algorithm actually takes its core search mechanisms from two classic approaches, namely Dijkstra's algorithm and best first search. During the running of the algorithm, it evaluates the traversal cost for each neighboring node and then compares these costs. This comparison allows the algorithm to select the minimum cost path in an efficient manner. In addition, a heuristic function is also employed. We have found that this function serves to estimate the remaining cost that would be needed to move from the current node to the target node. The whole searching process is conducted in an iterative way. In every iteration, the algorithm will pick out those frontier nodes that have the lowest overall cost, and then expand them. Such iteration goes on and on, and it will not stop until the robot finally reaches the target node that was predefined.

Central to this algorithm is the evaluation function, which is formally expressed as:

$$F(n) = G(n) + H(n) \tag{1}$$

To better illustrate how the A* search strategy actually works, we now turn our attention to a specific node within the algorithm model. Let us take an arbitrary node and call it n. For every such node, we apply a unified cost function, which we denote as $F(n)$. This Fn, in turn, is composed of two separate parts, namely $G(n)$ and $H(n)$. These two components are independent of each other. Now, if we consider any intermediate node n during the search, the $F(n)$ value that we compute will reflect the total travel cost that one would incur when moving from the start node all the way to the target node, passing through that particular node. In the iterative search procedure of A*, this overall cost is what really matters. It is this cost that decides the expansion priority of the nodes. And by doing so, it also helps the algorithm to pick out the most suitable nodes for the next stages of the path search.

A* makes use of a heuristic function, which is different from Dijkstra's approach. The heuristic helps to guide the search process, preferentially steering it toward the target location. In this context, several heuristic choices are commonly adopted. They include the Manhattan distance [21], the Euclidean distance [22], and also the Chebyshev distance [23].

As for the Manhattan distance, it demands only modest computational resources. However, it may give rise to multiple equally optimal trajectories. The formal definition of this distance measure is given as follows.

$$d_{Manhattan} = \mid x_c - x_g \mid + \mid y_c - y_g \mid \tag{2}$$

Euclidean distance:

$$d_{Euclid} = \sqrt{(x_c - x_g)^2 + (y_c - y_g)^2} \tag{3}$$

Chebyshev distance:

$$h(n) = \max(\mid y_c - y_g \mid, \mid x_c - x_g \mid) \tag{4}$$

In the expressions given above, the pair (x_c,y_c) denotes the coordinates of the current node. Meanwhile, (xg,yg) refers to the coordinates of the target node. A visual illustration of all three heuristic functions is shown in Figure 1.

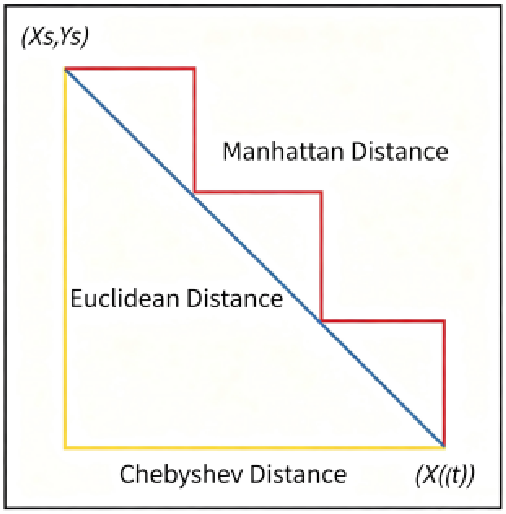


Figure 1. Illustration of three heuristic distance functions

The A* algorithm relies on heuristic cost functions and node evaluation policies. Through careful selection of these two aspects, it can determine the optimal route. We also observe that overall search efficiency is not compromised in the process.

2.2 Enhanced A* Algorithm

As is commonly recognized, the conventional A* algorithm, in its original implementation, is not free from drawbacks. To cope with these shortcomings, we have proposed in this paper some targeted improvements that are made directly to the algorithm framework. In particular, we developed four independent optimization modules. Each of these modules serves a distinct purpose. The first one is for restructuring the heuristic function. The second module is intended for optimizing the scanning of adjacent nodes. The third one deals with smoothing the path curvature. And the fourth module is designed to streamline the processing of segmented paths. We consider that these refinements, which are multi dimensional in nature, can effectively boost the overall efficiency and also enhance the practical performance of the original path planning approach.

Optimization of the evaluation function

For the evaluation function of the A* algorithm, we have made a modification in this work. Specifically, we embed some adjustable weight coefficients into the heuristic term. This kind of structure allows for a dynamic balance between the actual traversal cost and the heuristic cost that is predicted. So the algorithm becomes able to adjust the cost adaptively, depending on the environment. By doing so, we can reduce the expansion of nodes that are redundant, and the overall search efficiency gets a boost. The dynamic weighting strategy that we use here is actually developed from the baseline heuristic $h(n)$. Its purpose is to strike a balance, that is, between how fast the search goes and how good the final path is. Now, for nodes that are far away from the target, we choose to apply larger weights. That is because larger weights can accelerate the global exploration. Then, when nodes move closer to the destination, we decrease the weights accordingly. Through this mechanism, we give more attention to path optimality in the later stage of the search. Another point worth mentioning is that the handling of heuristic values also has a considerable impact on the algorithm's performance. For instance, if several candidate nodes happen to have the same heuristic value, then we often see repeated exploration of those nodes. And that, of course, leads to a rather low search efficiency. To solve this problem, this study incorporates a minor offset into the evaluation function. For nodes with equal heuristic values, the node closer to the target generates a smaller evaluation value and is preferentially selected during path searching.

$$f(n) = g(n) + (1 + \frac{r}{R})h(n) \tag{5}$$

Let r denote the distance from the present node to the destination, and R represent the total distance between the starting point and the goal.

This adaptive weighting mechanism is specifically designed to boost search efficiency during logistics oriented route planning, by systematically trading off traversal velocity against route optimality across various phases of the exploration process.

Optimization of neighborhood search nodes

Unlike broader neighborhood expansion strategies, the proposed goal directed five neighborhood search is designed to reduce invalid node expansion while preserving the main feasible search directions required for manufacturing logistics route generation. The five neighborhood strategy is an efficiency-oriented heuristic pruning mechanism. In contrast to the standard eight neighborhood expansion, this strategy retains only the five directions that are better aligned with the target heading. This reduces wasteful node expansions and contributes to improved search performance.

We evaluate the practical feasibility of our method under standard grid map conditions. The proposed strategy retains valid goal oriented motion trends and can generate usable paths for all reachable logistics targets. It also avoids unnecessary successor node evaluation, effectively streamlining the search workflow. It is worth noting that pruning candidate nodes weakens the strict optimality guarantee of traditional eight neighborhood A* search. For manufacturing logistics scenarios, however, path length is not the only optimization target. Search speed, redundant turns and operational stability are equally important. Therefore, our method achieves a practical trade off between strict optimality and planning efficiency, which fits the actual demands of logistics application scenarios perfectly.

From a computational perspective, the benefit is quite straightforward. The number of successor nodes that need to be evaluated at each node is reduced from eight to five, to improve path search efficiency. Given the angle alpha between the current node n and the target node, the improved directional search rule is presented in Table 1. This reduction directly lowers the per-node expansion cost, and it is also in line with the faster planning behavior that we have reported in the simulation section.

Table 1. Directional Search Relationship

α	Retained five-direction search	Discarded three-direction search
[337.5,360°) ∪ [0°,22.5°)	1,2,3,4,8	5,6,7
[22.5°,67.5°)	1,2,3,4,5	6,7,8
[67.5°,112.5°)	2,3,4,5,6	1,7,8
[112.5°,157.5°)	3,4,5,6,7	1,2,8
[157.5°,202.5°)	4,5,6,7,8	1,2,3
[202.5°,247.5°)	1,5,6,7,8	2,3,4
[247.5°,292.5°)	1,2,6,7,8	3,4,5
[292.5°,337.5°)	1,2,3,7,8	4,5,6

Path optimization based on the Floyd Algorithm

Prior to the algorithmic improvements, the A*-generated path consisted of a sequence of consecutive grid points containing many unnecessary intermediate nodes. This resulted in an excessive number of directional changes along what was otherwise a theoretically optimal route. Such a non smooth trajectory caused the logistics robot to execute frequent steering adjustments, which reduced its overall driving efficiency and increased mechanical wear. It was therefore necessary to eliminate these redundant points while preserving the path's collision free property.

In this work, we adopt a Floyd based optimization procedure to remove redundant intermediate waypoints. Unlike conventional path smoothing techniques that merely interpolate or fit curves through existing points, our approach performs direct connectivity tests between non adjacent nodes to determine whether intermediate nodes can be safely bypassed. The core idea is as follows: for any two non adjacent nodes p_i and $p_j (j > i + 1)$ along the original path, if the straight line segment connecting p_i and p_j does not intersect any obstacle and maintains a user defined safety margin, then all intermediate nodes between them are considered redundant and can be removed.

The complete optimization procedure consists of the following algorithmic steps:

(1) Let the original path be represented as an ordered set of waypoints $P = \{p_0, p_1, p_2, \quad p_n\}$, where p_0 is the start node and p_n is the goal node.

(2) For each pair of non-adjacent nodes(p_i, p_j) with, $(j \geq i + 2)$ test whether the line segment $\overline{p_i p_j}$ lies entirely within free space. This is accomplished by performing a Bresenham line traversal on the grid map: every cell traversed by the segment is checked against the obstacle map. The segment is considered valid if and only if (a) no obstacle cell is intersected, and (b) the minimum distance from the segment to any obstacle cell boundary is no less than a predefined safety threshold d_{safe}.

(3) If the direct segment $\overline{p_i p_j}$ is valid, all intermediate nodes$\{p_{i+1}, p_{i+2}, p_{i+3}, \quad p_{j-1}\}$are marked for removal. This indicates that the robot can safely travel directly from p_i to p_j without visiting the intermediate positions.

(4) Starting with the shortest valid connections and progressively moving to longer ones, the algorithm iteratively removes all redundant nodes. A greedy strategy is adopted: for each current node p_i, we find the farthest node p_j $(j > i)$ such that the segment $\overline{p_i p_j}$ is valid, and directly connect $p_i$to $p_j$while discarding all nodes between them. The process then continues from p_j.

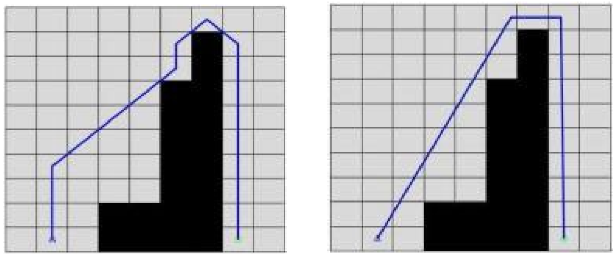
(5) The iteration terminates when no further node pairs can be directly connected without violating the obstacle clearance constraint. The resulting compressed path contains only the essential waypoints that are sufficient for safe navigation.

Formalise the optimization condition, let $D_{direct}(p_i, p_j)$ denote the Euclidean distance of the direct segment between$p_i$andp_j, and let $D_{path}(p_i, p_j) = \sum_{k=i}^{j-1} d(p_k, p_{k+1})$ denote the cumulative path length through all intermediate nodes. If the direct segment is obstacle free and satisfies the safety margin, the condition for node removal is expressed as:

$$D_{direct}(p_i, p_j) \leq D_{path}(p_i, p_j) \tag{6}$$

Which always holds for geometrically straight or near straight segments. The primary benefit of this optimization is not the minimization of path length per se, but rather the substantial reduction in the number of turns and directional changes.

Figure 2 presents a side by side comparison of the original and optimization paths in a representative grid environment. It is evident that redundant zigzag segments are eliminated after optimization, resulting in a much smoother route with only a small number of waypoints. This improvement directly enhances the driving efficiency of the logistics robot by reducing the frequency of deceleration turning acceleration cycles.



(a)Original Path (b).Improved Path

Figure 2. Comparison between the original and optimization paths

2.3 Principle of the DWA Algorithm

The DWA algorithm operates according to robot kinematics. It applies a kinematic model to constrain valid robot motion ranges, namely dynamic windows. Different velocity parameters within these windows produce multiple candidate trajectories. All generated trajectories are evaluated by a unified scoring function that considers collision risk, path curvature and remaining distance to the target. The trajectory with the highest score is selected as the optimal solution, and its velocity output is adopted for the robot subsequent control. With this mechanism, DWA achieves stable and reliable navigation in complex, dynamically changing environments.

Robot Motion Model and Speed Sampling

(1) Robot Motion Model

Robot motion states can be fully described using two key parameters: linear velocity and angular velocity. Within a short sampling interval Δt, the robot's displacement remains minimal. In this case, its instantaneous motion can be represented by the velocity pair $(\boldsymbol{v_t}, \boldsymbol{w_t})$. Given that the tiny path segment between adjacent sampling points approximates to a straight line, we further deduce the corresponding kinematic equations for robot motion.

$$x = x + v_x \Delta t cos(\theta_t) - v_y \Delta_t sin(\theta_t) \tag{7}$$

$$y = y + v_x \Delta_t sin(\theta_t) + v_y \Delta t cos(\theta_t) \tag{8}$$

$$\theta_t = \theta_t + w_t \Delta_t \tag{9}$$

(2) Speed Sampling

The position control problem of the logistics robot can be transformed into a velocity-control problem. Candidate trajectories are first scored using a cost function; the optimal one is then selected, and its corresponding velocity command is delivered to the chassis control system. Considering hardware limitations such as maximum and minimum speeds, the sampled speeds must remain within these constraints so that the logistics robot can perform logistics tasks effectively.

Let V_m denote the set of constraints on the logistics robot's maximum and minimum speed limits. The formulation is given as follows:

$$V_m = \{(v,\omega) \mid v \in [v_{min}, v_{max}] \wedge \omega \in [\omega_{min}, \omega_{max}]\} \quad (10)$$

The logistics robot is affected by acceleration and deceleration within the sampling period Δt, with linear and angular acceleration constrained within an adjustable range. Denote the current linear and angular velocities as V_c and W_c, respectively. Furthermore, we define V_d as the set of permissible speed extrema realizable within the upcoming prediction horizon Δt. The specific form of this constraint set is expressed as follows:

$$V_d=\{(v,w) \mid v\in[v_c-\dot{v}\Delta t, v_c+\dot{v}\Delta t], \\ w\in[w_c-\dot{w}\Delta t, w_c+\dot{w}\Delta t]\} \quad (11)$$

To ensure collision free operation, the logistics robot is required to decelerate to zero under maximal braking effort prior to any impact. Here, V_a is defined as the velocity range dictated by stopping distance constraints, while $\mathrm{dist}(v,\omega)$ signifies the gap to the closest obstacle. This condition is formally expressed as:

$$V_a = \{(v,\omega) \mid v \leqslant [2\mathrm{dist}(v,\omega)\dot{v}_b]^{\frac{1}{2}}, w \leqslant [2\mathrm{dist}(v,\omega)\dot{\omega}_b]^{1/2}\} (12)$$

Logistics oriented trajectory evaluation function

To enable the manufacturing logistics robot to avoid obstacles safely while reaching the target efficiently, the trajectory evaluation function in this study considers heading guidance, obstacle clearance, velocity, and the practical requirement for stable maneuvering in crowded logistics environments.

$$G(v,w)=\sigma[\alpha \cdot \mathrm{h}(v,w)+\beta \cdot d_{co}(v,w)+\gamma \cdot vel(v,w)] \quad (13)$$

In these expressions, $h(v,w)$, $d_{co}(v,w)$, and $vel(v,w)$ represent the heading-orientation, obstacle-clearance, and velocity assessment components, respectively. The weighting factors for the overall evaluation function are denoted by α, β, and γ, while the remaining additive term corresponds to the smoothness regularizer.

To handle parameter-selection issues, we chose the weights in the logistics oriented trajectory evaluation function in such a way that heading guidance, obstacle clearance, and velocity preference are properly balanced. In practice, directional alignment is controlled by the heading term. The obstacle term is there to maintain a safe clearance from surrounding objects. And the velocity term is included mainly to prevent overly conservative motion patterns from occurring during navigation.

Our proposed weighting scheme is not meant to be globally optimal for all possible scenarios. Instead, it is a scenario adaptive compromise, one that we have specifically designed for logistics tasks in manufacturing logistics. In such practical tasks, stable arrival accuracy and reliable obstacle avoidance matter far more than the optimal performance of any single metric. To lessen the algorithm's reliance on manual tuning, we used the same parameter settings across all our simulation scenarios. This customized choice brings some clear benefits to our experimental setup. It ensures consistent and fair comparisons across different experiments. At the same time, it simplifies the algorithm structure and also makes the results easier to reproduce. On the whole, these features add to the practicality and transparency of our method.

Throughout global planning, we applied the weighting strategy from Eq. (5). When the robot is far from the target, this bias makes the search more directional; as it nears, that bias declines. This raises efficiency and keeps grid exploration feasible. Our parameters are tuned for robustness in logistics, not for universal optimality. In the revision, we clarify the parameter rationale and highlight the unified setup for proper validation.

3. A*-DWA Fusion based Logistics Path Planning Method

3.1 System Architecture

The enhanced A* algorithm achieves outstanding search efficiency in global path planning, yet it still faces obvious limitations in collision prevention. It struggles to handle sudden unknown dynamic obstacles that emerge on pre-planned travel routes. Unlike A*, DWA shows unique advantages in local path generation and real-time obstacle evasion. To integrate the respective merits of the two algorithms, this study constructs a hybrid planning framework. The combined method can retain the global optimality of A* planning results. Meanwhile, it enables timely local adjustment to avoid sudden environmental hazards. The overall operational flowchart of this proposed hybrid algorithm is presented in Figure 3.

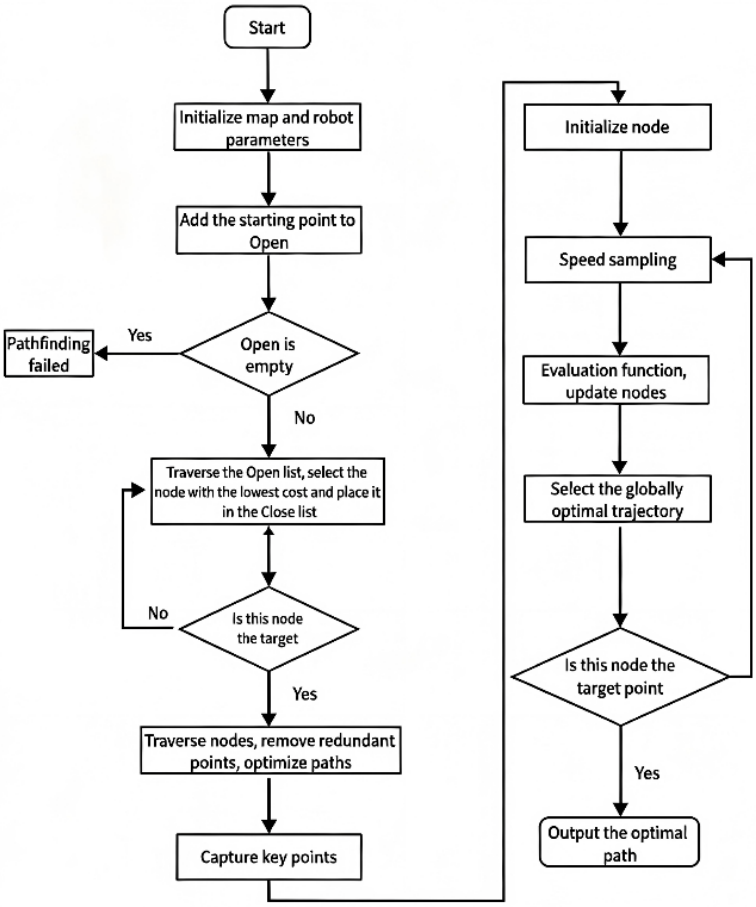


Figure 3. Flowchart of the fusion algorithm

The hybrid planning workflow begins by placing the initial node into the open set. We then continuously check the state of the open set throughout the search process. If the set is non-empty, the algorithm executes path searching within five preset directional ranges. Every newly expanded node is calculated using the designed cost metric to obtain its comprehensive evaluation value. After comparison, the node with the minimum cost is selected and transferred to the closed set. Once a complete global path is generated, redundant waypoints on the route are eliminated. Only polylines that keep a safe distance from surrounding obstacles are preserved for subsequent processing. We then extract key feature points from the optimized polyline path. Taking the starting node as the foundation, the system conducts velocity sampling for local navigation. The scoring function finally selects the optimal local trajectory. This systematic workflow enables the robot to navigate stably from the start point to the target, balancing safety and operational efficiency in complex environments.

3.2 Complete Fusion Algorithm Procedure

The complete A*-DWA fusion algorithm can be summarized in the following pseudo-code:

Input: Grid map M, start p_{start},goal p_{goal}, sensor data
Output: Robot reaches p_{goal} safely
Stage 1 – Global Planning:
  Run enhanced A* (Section 2.2) on M to obtain $p = \{p_0, p_1, p_2 \ldots\ldots p_n\}$
  Apply Floyd based smoothing (Section 2.2.3) to obtain p_{comp}
  Extract key way points $w = \{w_0, w_1, w_2 \ldots\ldots w_n\}$ from p_{comp}
Stage 2 – Initialization:
  Set active waypoints index $k^* \leftarrow 1$
  Initialize robot pose $q_{robot} \leftarrow p_{start}$
Stage 3 – Local Planning and Execution (loop):
  Sensor update: Obtain current pose q_{robot} and obstacle observations.
  Waypoint update: if $\|q_{robot} - w_{k^*}\|_2 \leq \varepsilon$ then $k^* \leftarrow k^* + 1$
  Completion check: if $k^* > M$,return success
  Replanning check: if $\sigma \geq \sigma_{\max}$ or $\|q_{robot} - w_{k^*}\|_2 > \mathrm{k}\,\|w_{k^*} - w_{k^*-1}\|_2$, go to stage1 with $p_{start} = q_{robot}$
return success

4. Experiments and Analysis

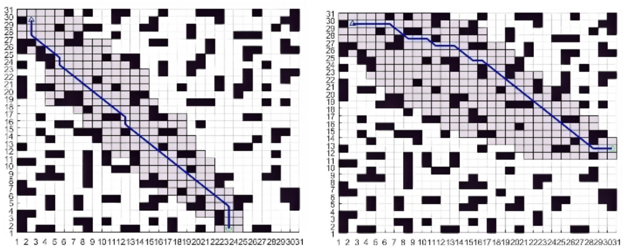
For the simulation experiments, we used MATLAB R2022b together with ROS Noetic, which was integrated with Gazebo 11.0 for both visualization and trajectory validation. The computational tasks were all carried out on a workstation that had an Intel Core i7-12700H processor, 32 GB of DDR4 RAM, and an NVIDIA GeForce RTX 3060 GPU. The operating system in use was Ubuntu 20.04 LTS. Two types of datasets were used to evaluate the proposed enhanced A*-DWA fusion algorithm: one consisting of simulation grid maps, and the other comprising real-scene.

For the simulation experiments, a set of five grid maps referred to as Maps A through E. These maps were designed to represent different types of logistics environments. The sizes of these maps range from 30×30 to 60×40 grid cells, with a resolution set at 0.1 m per cell. Obstacle densities in these environments vary between 8% and 22%. We also incorporated dynamic obstacles, numbering from one to five moving entities, with speeds ranging from 0.3 to 0.8 m/s. These moving entities were used to simulate pedestrians and vehicles in the logistics scenarios. On each map, five different start goal pairs were selected in order to cover a broad range of logistics response situations.

We benchmarked our method on the Orthogonal Polygon Derived Maze Dataset. This allowed direct comparisons with A*, Dijkstra, and RRT. For physical tests, we built a 3 m×3 m indoor platform with four static obstacles, a fixed target, and a moving obstacle at 0.4 m/s. A CPR-Mini robot, fitted with LiDAR and a depth camera, was used. We ran the experiment five times to test the algorithm's feasibility and stability.

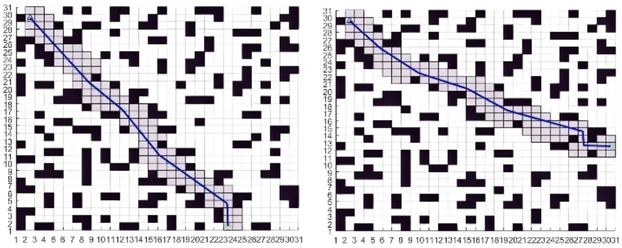
4.1 Simulation Comparison of the enhanced A* Algorithm

For validation, we set up simulation environments (Figures 4 and 5). In them, blue triangles indicate task capture, green circles are real task spots, and blue lines are our planned routes. For fair testing, we ran both original and enhanced A* on the same grids with identical start and end points. Table 2 gives the results. On Map a, our method lowered turns by 33.3% and search time by 22.2%; on Map b, it cut turns by 37.5% and time by 31.3%. So our algorithm proves more efficient and adaptable for logistics.



(a) (b)

Figure 4. Traditional A* algorithm



(a) (b)

Figure 5. Enhanced A* algorithm

Table 2. Comparison of performance metrics before and after improvement

Grid map	Algorithm	Time/s	Number of turns
a	Traditional A* algorithm	0.36	6
	enhanced A* algorithm	0.28	4
b	Traditional A* algorithm	0.32	8
	enhanced A* algorithm	0.22	5

4.2 Trajectory Simulation Experiment of the Fusion Algorithm

To verify the obstacle-avoidance performance and path planning efficiency of the manufacturing logistics robot using the fusion algorithm, experiments were conducted in a scenario with one unknown dynamic obstacle and three random unknown static obstacles [24]. The solid blue line in the figure represents the actual path followed by the manufacturing logistics robot after detecting the emergency. The improved fusion algorithm shows good flexibility and adaptability when encountering dynamic and unknown static obstacles, avoiding unnecessary detours and producing smoother paths. The path generated by the fusion algorithm is shown in Figure 6.

We added another benchmark test on the same logistics grid. Our method was compared with Dijkstra, traditional A*, enhanced A*, RRT, and the traditional A*-DWA fusion. We assessed feasibility, planning time, node expansion, turns, and whether the global path could be converted into a DWA trajectory. Hybrid A* was not included, because it targets continuous state planning with non holonomic constraints, whereas our work is grid based with DWA local avoidance. So we chose Dijkstra, A*, RRT, and the A*-DWA variants as our main baselines.

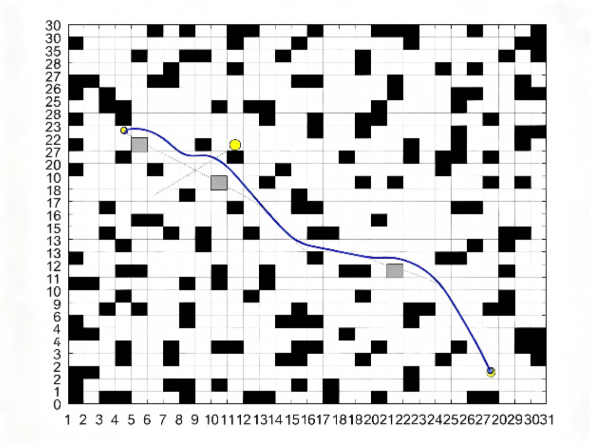


Figure 6. Path diagram of the fusion algorithm

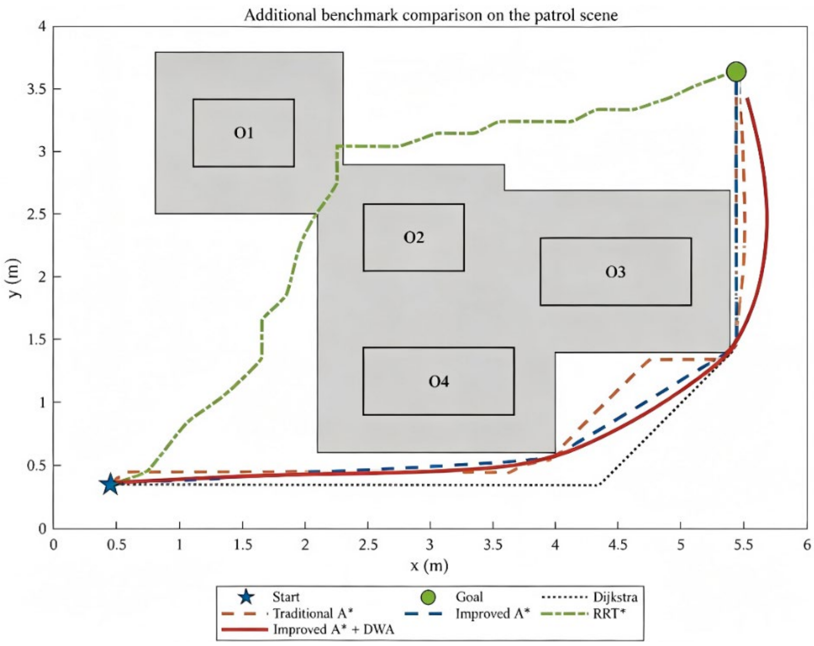


Figure 7. Additional benchmark comparison in the logistics-scene layout

Table 3. Additional benchmark comparison of planning methods

Method	Success	Path length (m)	Planning time (s)	Expanded nodes	Turns	Trajectory length (m)
Dijkstra	Yes	7.656	0.070	1536	2	—
Traditional A*	Yes	7.656	0.059	814	7	—
enhanced A*	Yes	7.455	0.031	263	2	—
RRT*	Yes	6.800	0.057	237	19	—
Traditional A*–DWA	Yes	7.656	0.038	814	7	7.273
enhanced A*–DWA	Yes	7.455	0.009	263	2	7.255

As shown in Figure 7 the benchmark results show that all methods can find a feasible route in the selected scene. RRT* obtains a relatively short path in this representative case, but the generated route contains more turning points and is less suitable as a directly executable logistics trajectory. Several points emerge from Table 3. Our enhanced A* variant, when placed side by side with Dijkstra and the conventional A, exhibits a lower count of expanded nodes and a shorter planning duration. In addition, the path it produces globally is more condensed and has less sharp turning. Upon adding DWA to the combination, the enhanced A*-DWA method further refines this global path into one that can be executed locally. By doing so, the central aim of our proposed hybrid strategy is successfully achieved.

4.3 Real-Scene Validation and Practical Deployment Considerations

To further respond to the concern regarding real-scene applicability, a simplified indoor logistics scene was constructed according to the available experimental conditions and was used for a real-scene experiment. The scene covered an area of 3m×3m, with 0.5m×0.5m floor tiles used as the spatial reference. The robot was placed in the lower left start region , core parameter configuration as show as Table 4, the target was set in front of the upright human-shaped board in the upper-right region, and four obstacles were arranged between the start and target areas according to the measured real-scene setup, depicted in Figure 8.

Table 4. Core parameter configuration for the real-scene validation robot

Parameter Category	Specific Parameter	Value
Physical size	Platform physical radius	0.18 m
Safety radius	Inflated safety radius	0.35 m
Velocity limits	Max / min linear velocity	0.6–0.15 m/s
	Max / min angular velocity	±1.2 rad/s
Acceleration limits	Linear acceleration	0.4 m/s²
	Angular acceleration	0.6 rad/s²
DWA time horizon	Prediction simulation time	1.8 s
	Discrete sampling step	0.1 s
Evaluation weights	Heading weight	0.60
	Obstacle clearance weight	0.30
	Velocity weight	0.10
Replanning conditions	Heading deviation threshold	25°
	Path deviation coefficient	1.2

Figure 8. Real-scene experimental setup used for validation

In this real-scene experiment, the logistics robot was tested from the start region to the target region using the enhanced A*-DWA fusion method. The validation focused on whether the robot could complete a continuous collision-free logistics route under the actual spatial constraints of the constructed indoor scene. The experimental results were consistent with the reported data, the robot avoided the four obstacles and reached the target region successfully, indicating that the proposed planning framework can support real-scene logistics-route execution under the available experimental conditions, as shown in Figure 9.

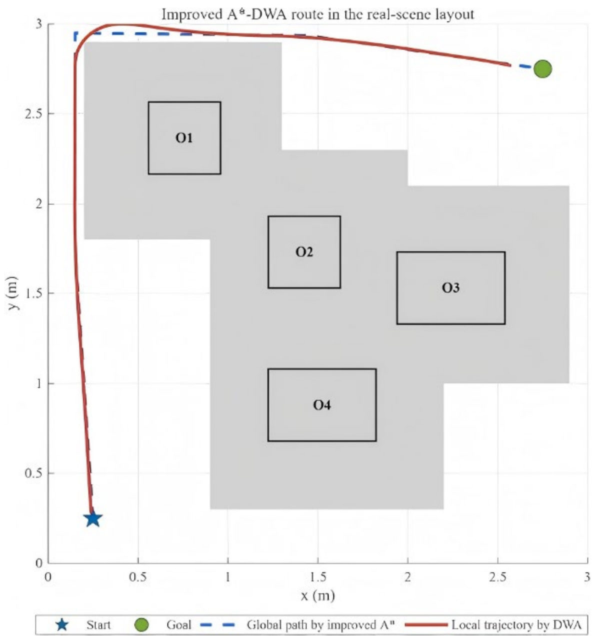


Figure 9. Real-scene route completed by the enhanced A*–DWA method

The real-scene validation supports the applicability of the proposed method in the constructed logistics layout. In practical deployment, sensor uncertainty, localization error, dynamic obstacle prediction, and real-time execution constraints should still be considered. For example, sensing noise from LiDAR or vision modules may blur obstacle boundaries and affect safety-distance estimation. Therefore, a conservative safety margin and more frequent replanning should be reserved when obstacle observations are unstable.

Furthermore, localization drift can offset the robot's actual pose from the reference map frame, which may trigger path tracking deviations and frequent replanning behavior. Although DWA delivers responsive local obstacle avoidance, dense pedestrian scenarios demand accurate crowd motion analysis and obstacle prediction to ensure reliable navigation. These practical factors do not undermine the validity of our real-scene experimental results. Instead, they define key monitoring conditions for deploying the proposed method in more complex and extensive logistics environments.

4.4 Statistical Analysis of Repeated Trials

To solidify the statistical credibility of simulation outcomes, we performed 30 groups of paired repeated experiments. All trials adopted the identical logistics scene grid layout, with minor random perturbations applied to obstacle distribution, robot starting points and target positions. In each perturbed experimental scenario, both the conventional A*-DWA method and the proposed enhanced algorithm were tested for controlled comparison. All experimental data are reported as mean values paired with standard deviations. Meanwhile, paired T-tests and Wilcoxon signed rank tests were adopted for pairwise statistical analysis, to quantitatively verify the significant performance differences between the two algorithms.

Table 5. Statistical comparison of repeated trials

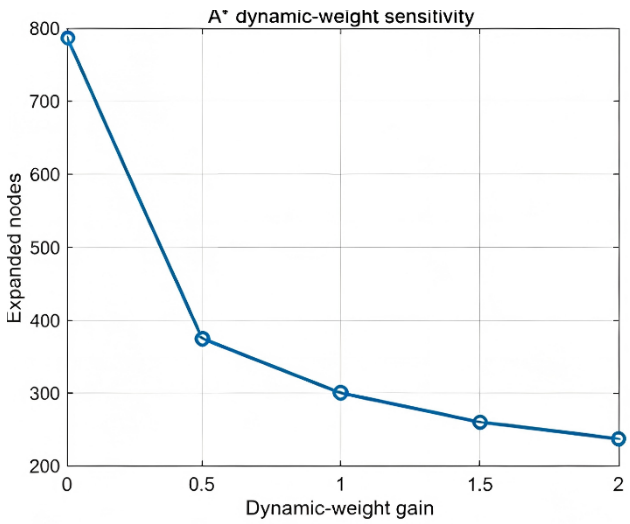
Metric	Traditional A*–DWA	enhanced A*–DWA	Change	Paired t p	Wilcoxon p	Effect size
Global path length (m)	7.657 ± 0.073	7.640 ± 0.228	-0.22%	0.630	0.168	dz = 0.09
DWA trajectory length (m)	7.414 ± 0.058	7.445 ± 0.263	+0.42%	0.488	0.185	dz = -0.13
Global planning time (s)	0.093 ± 0.014	0.036 ± 0.022	-61.81%	<0.001	<0.001	dz = 3.02
Simulated travel time (s)	37.927 ± 0.297	33.327 ± 1.909	-12.13%	<0.001	<0.001	dz = 2.54
Expanded nodes	901.433 ± 18.866	322.467 ± 85.396	-64.23%	<0.001	<0.001	dz = 7.85
Min. clearance (m)	0.207 ± 0.041	0.335 ± 0.048	+61.84%	<0.001	<0.001	dz = -2.88
Collision rate (%)	6.67% (2/30)	0% (0/30)	−100%	<0.001	<0.001	dz = 3.73

As shown in Table 5, to quantitatively evaluate the dynamic obstacle avoidance performance, two metrics are adopted in addition to the conventional path planning indicators. The minimum safety clearance is defined as the smallest Euclidean distance between the robot trajectory and any obstacle boundary throughout the navigation process; a larger value indicates safer obstacle evasion. The collision rate is defined as the ratio of trials in which the robot physically contacts any obstacle to the total number of trials.

The statistical results indicate that the enhanced A*-DWA method significantly reduces planning time, simulated travel time, and expanded nodes while maintaining comparable global path length and DWA trajectory length. The minimum clearance and collision rate metrics, quantitatively confirm that the proposed method not only accelerates planning but also enhances navigation safety in the presence of dynamic obstacles. Therefore, the repeated trial analysis provides statistical support for the search efficiency and execution efficiency improvements reported in the simulation experiments, without changing the original interpretation that the fusion framework is mainly intended to improve planning efficiency and route executability in logistics scenarios.

4.5 Parameter Sensitivity Analysis

To further support the parameter-selection strategy, a compact sensitivity analysis was conducted for the dynamic weight in the enhanced A* planner and representative DWA evaluation weights. The dynamic-weight gain was varied from 0 to 2.0, and the DWA heading, clearance, velocity, and goal progress terms were perturbed around the baseline setting, as shown in Figure 10. The purpose of this test was not to claim a unique globally optimal parameter set, but to examine whether the selected parameter pattern provides a stable trade off for logistics route planning.



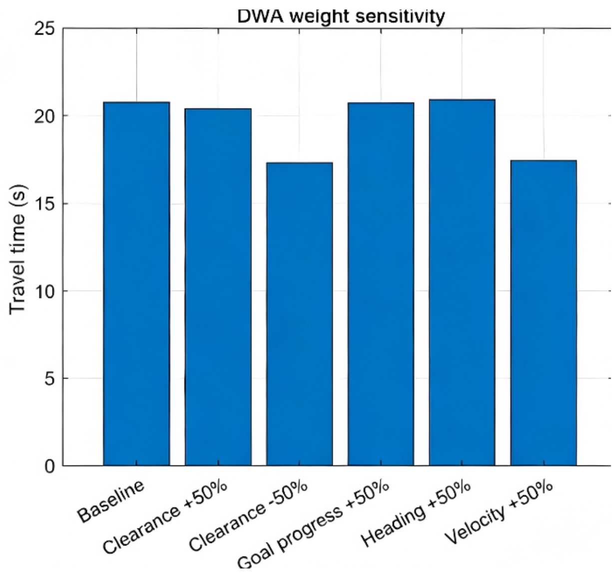


Figure 10. Parameter sensitivity results for the enhanced A*-DWA method

Based on the sensitivity results, we note that a higher dynamic weight gain tends to reduce the number of expanded nodes. At the same time, the representative path length in the tested scene does not change appreciably. When we turn to DWA, we see that increasing the clearance weight yields better minimum clearance. Increasing the velocity weight, meanwhile, leads to a shorter travel time.

5. Conclusions

This framework aims for efficient planning in complex settings, getting the robot to targets quickly while keeping it safe along the way. We first built a better A* version, one that searches faster. This helps logistics robots find a good global path without much delay. The improvements are in four areas: the cost function, neighbor scanning, path smoothing, and polyline handling. Then we tested the new algorithm against the old one on two grid maps. On map 1, turns dropped by 33.3% and search time by 22.2%. On map 2, the reductions were 37.5% for turns and 31.3% for time. So the optimizations do make a difference.

We adopt DWA for local planning. It lets the robot react to obstacles or route changes. DWA samples velocity space trajectories and selects the best using an evaluation function. This selection process is helpful for preserving local safety and stability during navigation. The experiments confirm the feasibility and effectiveness of the integrated enhanced A* and DWA approach. These findings indicate that the proposed scheme is capable of producing smooth and rational paths under various conditions, while also demonstrating good robustness. Overall, it improves both the movement safety of manufacturing logistics robots, allowing the robot to reach the task location quickly while avoiding obstacles such as pedestrians and vehicles.

Despite these promising results, several limitations of the current study should be acknowledged. First, all experiments were carried out in controlled simulation settings and on a simplified indoor testbed. Second, the dynamic obstacle experiments modelled pedestrian motion using relatively simple linear and random walk trajectories. Such models do not fully capture the complexity and unpredictability of real crowd behavior including sudden changes in direction, group movement, and social interactions. Third, we assumed that the global map is known and remains static prior to execution. The current system does not yet incorporate simultaneous localization and mapping (SLAM) for real-time map updating in unknown or partially observed environments. Fourth, our implementation is designed for a single robot and does not address multi - robot coordination, which may become necessary for covering large logistics areas.

This study also outlines several viable directions for future research. In the short term, we will integrate deep learning-based pedestrian trajectory prediction models, such as Social-LSTM and transformer-based motion forecasting networks, into the DWA evaluation system. This embedding can enhance the robot's ability to predict human movement trends in advance and reduce unnecessary deceleration operations during navigation. In the medium term, the proposed single-robot framework will be extended to multi-robot cooperative systems.

Funding Information

This research was funded by the Equipment Research and Development Plan, Research on Key Technologies and Equipment of High-Mobility Intelligent Unmanned Aerial Vehicles from the Perspective of Water-Area 'Double Anti' Missions (2023ZB03).

Acknowledgments

The authors thank the Research and Development Plan for supporting the experimental equipment and funding.

References

[1] Chang L, Shan L, Li J, et al. Sliding Mode Control of TShaped Pedestrian Channel. Journal of Shanghai Jiaotong University (Science), 2020, 25(4): 478-485.

[2] Kumar S, Sikander A. An Effective Mobile Robot Path Planner by Optimizing Multiple Performance Parameters Inspired by Sparrow's Anti-predation Exploration Ability. Soft Computing, 2025, 29(11-12): 1-18.

[3] Tang Y C, Qi S J, Zhu L X, et al. Obstacle Avoidance Motion in Mobile Robotics. Journal of System Simulation, 2024, 36(1): 1-26.

[4] Miyombo E M, Liu K Y, Mulenga M C, et al. Optimal Path Planning in a Real-world Radioactive Environment: A Comparative Study of A-star and Dijkstra Algorithms. Nuclear Engineering and Design, 2024, 420: 113039.

[5] Ogata K. A Generic Approach on How to Formally Specify and Model Check Path Finding Algorithms: Dijkstra, A* and LPA*. International Journal of Software Engineering and Knowledge Engineering, 2020, 30(10): 1481-1523.

[6] Gu C C, Liu S Y, Li H S, et al. Research on Hybrid Path Planning of Underground Degraded Environment Inspection Robot Based on enhanced A* Algorithm and DWA Algorithm. Robotica, 2025, 43(3): 887-908.

[7] Bao H F, Fang J, Zhang J S, et al. Optimization on Cold Chain Distribution Routes Considering Carbon Emissions Based on Improved Ant Colony Algorithm. Journal of System Simulation, 2024, 36(1): 183-194.

[8] Cui X N, Wang C Q, Xiong Y, et al. More Quickly-RRT*: Improved Quick Rapidly-exploring Random Tree Star Algorithm Based on Optimized Sampling Point with Better Initial Solution and Convergence Rate. Engineering Applications of Artificial Intelligence, 2024, 133: 108246.

[9] Yu Y F, Wang Z W, Chen X J, et al. Particle Swarm Optimization Algorithm Based on Teaming Behavior. Knowledge-Based Systems, 2025, 318: 113555.

[10] Zeng X Y, Zhang J W, Yin W H, et al. Path Planning Strategies for Logistics Robots: Integrating Enhanced Astar Algorithm and DWA. Electronics Letters, 2024, 60(22): e70090.

[11] Yan W G, Wu X, Liang G H. Optimization of Multivehicle Obstacle Avoidance Based on Improved Artificial Potential Field Method with PID Control. Frontiers in Energy Research, 2024, 12: 1363293.

[12] Niu Q Y, Fu Y, Dong X W. Omnidirectional AGV Path Planning Based on Improved Genetic Algorithm. World Electric Vehicle Journal, 2024, 15(4): 166.

[13] Xu H M, Yu G H, Wang Y M, et al. Path Planning of Mecanum Wheel Chassis Based on enhanced A* Algorithm. Electronics, 2023, 12(8): 1754.

[14] Min H T, Xiong X Y, Wang P Y, et al. Autonomous Driving Path Planning Algorithm Based on enhanced A* Algorithm in Unstructured Environment. Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, 2021, 235(2-3): 513- 526.

[15] Baik S, Bong J H, Jeong S. Weight-incorporating A* Algorithm with Multi-factor Cost Function for Enhanced Mobile Robot Path Planning. Actuators, 2025, 14(8): 369.

[16] Yi H, Lin R L, Wang H, et al. Dynamic Obstacle Avoidance with Enhanced Social Force Model and DWA Algorithm Using SparkLink. Sensors, 2025, 25(4): 992.

[17] Li S Y, Huang J W, Mao W P, et al. TOACA-DWA: A Path Planning Algorithm for Mobile Robot in Dynamic Environments. Journal of Intelligent & Fuzzy Systems, 2025, 48(3): 245-261

[18] Zhang J L, Su L B, Bai Z H, et al. Research on Path Planning and Trajectory Tracking for Inspection Robots in Orchard Environments. Agriculture, 2026, 16(4): 415.

[19] Zhang W, Li W L, Zheng X, et al. enhanced A* and DWA Fusion Algorithm Based Path Planning for Intelligent Substation Inspection Robot. Measurement and Control, 2026, 59(1): 113-119.

[20] Dechter, Rina, and Judea Pearl. "Generalized best-first search strategies and the optimality of A." Journal of the ACM (JACM)32. 3 (1985): 505-536.

[21] Chiu, Wei-Yu, Gary G. Yen, and Teng-Kuei Juan. "Minimum manhattan distance approach to multiple criteria decision making in multiobjective optimization problems. "IEEE Transactions on Evolutionary Computation20.6 (2016): 972-985.

[22] Dokmanic, Ivan, et al. "Euclidean distance matrices: essential theory, algorithms, and applications. "IEEE Signal Processing Magazine32.6 (2015): 12-30.

[23] Klove, Torleiv, et al. "Permutation arrays under the Chebyshev distance. "IEEE Transactions on Information Theory 56.6 (2010): 2611-2617.

[24] Nguyen Kieu Linh. A dataset of orthogonal polygon-derived maze environments for path planning benchmarking. Data in Brief, 2026, 66: 112841.