Hindawi
Journal of Robotics
Volume 2023, Article ID 6619841, 16 pages
https://doi.org/10.1155/2023/6619841

Research Article

Mobile Robot Path Planning Based on Improved Particle Swarm Optimization and Improved Dynamic Window Approach

Zhenjian Yang, Ning Li, Yunjie Zhang, and Jin Li

School of Computer and Information Engineering, Tianjin Chengjian University, Tianjin 300000, China

Correspondence should be addressed to Yunjie Zhang; zyj@tcu.edu.cn

Received 17 March 2023; Revised 25 April 2023; Accepted 3 May 2023; Published 18 May 2023

Academic Editor: Oscar Reinoso

To enable mobile robots to effectively complete path planning in dynamic environments, a hybrid path planning method based on particle swarm optimization (PSO) and dynamic window approach (DWA) is proposed in this paper. First, an improved particle swarm optimization (IPSO) is proposed to enhance the exploration capability and search accuracy of the algorithm by improving the velocity update method and inertia weight. Secondly, a particle initialization strategy is used to increase population diversity, and an addressing local optimum strategy is used to make the algorithm overcome the local optimum. Thirdly, a method of selecting navigation points is proposed to guide local path planning. The robot selects the appropriate navigation points as the target points for local path planning based on the position of the robot and the risk of collision with dynamic obstacles. Finally, an improved dynamic window approach (IDWA) is proposed by combining the velocity obstacle (VO) with the DWA, and the evaluation function of the DWA is improved to enhance trajectory tracking and dynamic obstacle avoidance capabilities. The simulation and experimental results show that IPSO has greater exploration capability and search accuracy; IDWA is more effective in trajectory tracking and dynamic obstacle avoidance; and the hybrid algorithm enables the robot to efficiently complete path planning in dynamic environments.

1. Introduction

In recent years, mobile robots have been applied in a wide range of fields. As a key technology of robotics, path planning has become a research hotspot in robotics [1]. There are two types of path planning methods for robots: global path planning and local path planning. Global path planning is based on the entire environmental information, such as the shape and size of obstacles, and their distribution on the map. On the other hand, local path planning has some unknown environmental information, such as the size and position of dynamic obstacles [2].

Currently, several algorithms have been applied to the global path planning of robots. The A$*$ algorithm [3] is a heuristic approach capable of theoretically determining the global optimal path. However, its computational complexity scales with the complexity of the environment, which can result in a considerable increase in execution time. The ant colony optimization (ACO) [4] is a heuristic algorithm that can effectively solve path planning problems. However, it suffers from limitations such as slow convergence speed and a tendency to converge towards the local optimum, which can impair its ability to find the global optimal solution. The genetic algorithm (GA) [5] is an intelligent algorithm known for its robustness and parallelism, and it is commonly used in path planning. However, it is susceptible to limitations such as early convergence, poor convergence path quality, and poor population diversity, which can restrict its effectiveness in finding the global optimal path. The slime mould optimization algorithm (SMOA) [6] is an intelligent algorithm that mimics the behavioral and morphological changes observed during the foraging process of slime mould. Although it can be used to solve path planning problems, it suffers from limitations such as slow convergence speed, and a tendency to converge towards the local optimum. The salp swarm algorithm (SSA) [7] is a recently developed intelligent algorithm that draws inspiration from the foraging behavior of the salp. While it exhibits strong performance in solving path planning problems, it suffers from drawbacks such as slow convergence speed, convergence towards local optimum, and poor population diversity. The particle swarm optimization (PSO) [8] is an intelligent algorithm that boasts several advantages, including a simple structure, few parameters, easy implementation, and fast convergence. However, the algorithm also has limitations such as low search accuracy, poor population diversity, and convergence towards the local optimum [9]. Existing literature shows that intelligent algorithms used for path planning often suffer from limitations such as low accuracy, poor diversity, and the local optimum. These limitations can restrict the ability of the algorithm in meeting the demands of global path planning. Consequently, many scholars have focused on improving these algorithms. Ji et al. [10] proposed improvements to the pheromone update strategy and state transfer strategy of the ACO to increase the randomness of the path selection by the ant colony, thereby enhancing the global exploration ability and reducing the probability of the algorithm trapping in the local optimum. Additionally, they introduced a local optimal escape strategy to effectively address the problem of local optimum in the ACO. Zhai and Feng [11] proposed a method to integrate simulated annealing into the GA, which can effectively mitigate the issue of the search process getting stuck in the local optimum. The approach also includes adaptive operators for crossover and mutation probabilities, which help to maintain population diversity and enhance the algorithm's global exploration capability. Wang et al. [12] proposed a learning technique based on orthogonal lensing opposition in SSA to enhance the algorithm's ability to escape the local optimum and find higher quality paths. Additionally, they 4introduced a scheme to adaptively adjust the number of leaders and a dynamic learning strategy to improve the global exploration capability and convergence speed. Wang et al. [13] enhanced the traditional SSA algorithm by incorporating inertia weight, designing a velocity clamping strategy, and designing an adaptive transformation parameter strategy. These modifications were aimed at achieving a better balance between the global exploration capability and local search accuracy of the algorithm. Agarwal and Bharti [14] proposed an improved version of the SMOA with improved slime mould weights. The proposed method maintains a specific perturbation rate in the early stage of the algorithm to avoid premature convergence towards the local optimum and a reduced perturbation rate in the later stage of the algorithm to accelerate convergence. Additionally, the authors designed a multiobjective fitness function that considers both path minimization and obstacle avoidance. This approach improves the performance of the algorithm and enables it to achieve better results. Zhang et al. [15] proposed improvements to the linearly decreasing inertia weight and learning factors in PSO to balance the exploration capability and search accuracy of the algorithm. The authors also introduced a measure of population diversity using fitness variance. However, the issue of premature and local optimum in the algorithm was not addressed. Zhong et al. [16] proposed an improved PSO that enhances the global exploration capability and diversity of the population. In the initialization phase, mutations were introduced to the particle positions to increase the search space coverage. Furthermore, a multi-particle competition mechanism was incorporated to promote population diversity in later stages of the algorithm. Wang et al. [17] proposed a more effective quantum swarm optimization algorithm that enhances both the population diversity and global exploration capabilities of the algorithm. Li et al. [18] proposed a modified PSO algorithm that utilizes hierarchical random disturbance to increase population diversity and improve the algorithm's local search capability by modifying particle positions. Additionally, a random positive feedback factor is included to further enhance the algorithm's performance. Tan et al. [19] proposed a method to determine the excellence of particles and populations by calculating their evolutionary capabilities. They then adjusted the inertia weight and learning factors based on the evolutionary ratio to improve the global exploration capability and convergence speed of the algorithm. The aforementioned improvements in PSO aim to enhance the exploration capability and search accuracy of the algorithm, yet they fail to address the issue of local optimum in PSO, leading to longer convergence path lengths. Therefore, this paper proposes an improved particle swarm optimization (IPSO) to address the limitations of PSO, including premature and local optimum, which can lead to long convergence path lengths. IPSO overcomes these drawbacks by improving the velocity update and inertia weight of the algorithm, as well as using an initialization strategy and addressing the local optimum strategy, resulting in a shorter convergence path length.

Theoretically, the aforementioned algorithms can obtain the shortest global path, but in practical environments, dynamic obstacles may render the obtained global path useless. In the context of dynamic environments, local path planning has good adaptability and can better handle dynamic obstacles. Some of the well-known local path planning algorithms include the BUG algorithm [20], the artificial potential field algorithm [21], the dynamic window approach (DWA) [22], etc. The DWA algorithm, which takes into account the dynamic constraints of the robot, is considered a practical local path planning method [23]. However, due to the absence of global path guidance, DWA is susceptible to getting trapped in the local optimum and is unable to avoid fast-moving obstacles [24].

Many scholars divide the path planning of robots into two phases: global path planning and local path planning. The global path planning phase aims to obtain a collision-free global path, which is used to guide local path planning. The local path planning phase is guided by the global path to obtain a path that avoids dynamic obstacles and satisfies the robot dynamics constraints [25]. Liu et al. [26] proposed a fusion algorithm that combines Jump $A*$ and DWA to improve the smoothness of the path and reduce the number of turn points. By improving the DWA, the efficiency of local path planning and obstacle avoidance is increased. Ji et al. [27] proposed a fusion algorithm that can be applied in complicated environments. The algorithm incorporates the weight information of the road surface into the heuristic function of the A ∗ algorithm, allowing the robot to avoid bumpy roads. Finally, DWA is utilized to accomplish local path planning to enhance the robot's ability to avoid dynamic obstacles. Jin et al. [28] proposed a hybrid algorithm that combines the A ∗ algorithm and the DWA. They added the target pose and orientation information to the A ∗ algorithm to obtain a more suitable global path for robot orientation. They improved the evaluation function of DWA to make the robot's orientation more adaptive to the curved global path and avoid dynamic obstacles. This algorithm is suitable for complicated environments and can effectively increase the robot's global path planning and obstacle avoidance capabilities. Yang et al.[29] proposed a hybrid algorithm that utilizes the global path generated by the improved ACO as the navigation information for local path planning. The authors improved the sampling window and evaluation function of the DWA to enhance the path tracking ability, dynamic obstacle avoidance ability, and motion stability of the robot. Jin et al. [30] improved ACO by dynamically adjusting the pheromone concentration to expedite convergence, followed by a secondary planning phase to reduce the global path length. Finally, the authors introduced various response strategies based on the motion trajectory of dynamic obstacles to enable successful avoidance of dynamic obstacles in the environment. In this paper, a fused path planning method is proposed that combines IPSO with IDWA. The proposed approach involves obtaining the global path by using IPSO and selecting navigation points in the global path to guide local path planning. IDWA is then used for local path planning and the avoidance of dynamic obstacles in the environment.

The main contributions of this paper are as follows: (1) an adaptive selection strategy is proposed to enhance the global exploration ability and search accuracy of PSO by enabling some particles to select the corresponding inertia weight and velocity update methods based on their performance. (2) A positive feedback factor is added to PSO to adjust the particle positions, enhance the search accuracy, and accelerate the convergence speed of the algorithm. (3) A particle initialization strategy is proposed to increase the diversity of the population and use the strategies of adding random perturbation and random crossover to make the algorithm more capable of addressing the local optimum, thereby enhancing the exploration ability of PSO. (4) IDWA is proposed to enhance global path guidance and obstacle avoidance by combining with VO and improving the evaluation function. (5) A hybrid path planning method is proposed to complete the path planning task for mobile robots in dynamic environments.

The structure of the remaining part of this paper is as follows: Section 2 describes the path planning problem and models the path planning problem and obstacles. Section 3 introduces the traditional PSO. Sections 4 and 5, respectively, introduce IPSO and IDWA. Section 6 introduces the process of algorithm fusion. Section 7 verifies the superiority and effectiveness of IPSO, IDWA, and the fusion algorithm through simulation experiments. Section 8 summarizes and concludes this study.

2. Problem Statement and Model Construction

2.1. Modelling of Path Planning. To ensure the safety of robots and reduce energy consumption, the goal of global path planning for mobile robots is to find a collision-free path with the shortest length in the environment [31]. Robot global path planning is usually seen as finding the set P of points generated by the robot moving through the environment, and the connection between adjacent points does not cross static obstacles. The set of points P can be expressed as the following equation:

$$P = \{S, p_1, p_2, \cdots, p_m, T\}, \tag{1}$$

where $(p_1, p_2, \cdots, p_m)$ is the sequence of points in the environment, S is the starting point, T is the target point, L is the length of the path, and L is calculated by the following equation:

$$L = \sum_{i=1}^{n-1} l_{(i,i+1)}, \tag{2}$$

where $l_{(i,i+1)}$ represents the distance from point i to point $i+1$ and n represents the number of elements of P.

To make the global path more suitable for mobile robots, this paper uses the third spline interpolation method to optimize the global path, so that the global path is smoother and the turning angle is smaller [32]. The method for generating a cubic spline curve path between points p_i and p_{i+1} is as follows.

(1) Generate m interpolation points between points p_i to p_{i+1} using cubic spline interpolation $(p_{i1}, p_{i2}, \cdots, p_{im})$.
(2) Connecting $(p_{i1}, p_{i2}, \cdots, p_{im})$ together to obtain a continuous path is called a cubic spline interpolation path.

The length $\mathrm{Cubl}_{(i,i+1)}$ of the cubic spline interpolation path between points p_i to p_{i+1} can be calculated from the following equation:

$$\mathrm{Cubl}_{(i,i+1)} = l_{(i,i1)} + \sum_{k=1}^{m-1} l_{(ik,ik+1)} + l_{(im,i+1)}. \tag{3}$$

The cubic spline interpolation path length L can be calculated from the following equation:

$$L = \sum_{i=1}^{n-1} \mathrm{Cubl}_{(i,i+1)}. \tag{4}$$

2.2. Modelling of Obstacle. To simplify the obstacle avoidance problem in the robot's working environment, it is common to consider the influence range of obstacles as the outer circle of the obstacle. Here the influence radius is R_{obs}. To ensure the safety of the robot, the safety distance d is added to the radius of influence of the obstacle, which depends on the size of the robot. d is added to the radius of influence of the obstacle, as shown in Figure 1.

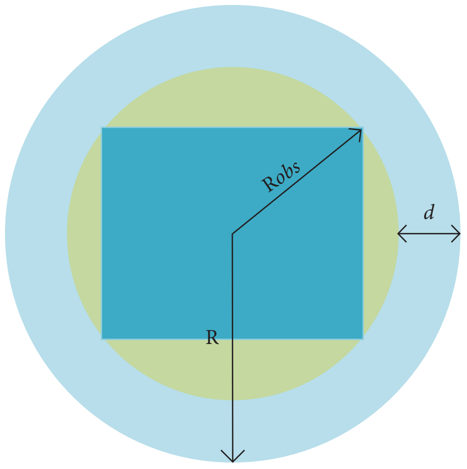


FIGURE 1: The actual radius of influence of the obstacle R.

The obstacle's real radius of influence R can be calculated by the following equation:

$$R = R_{obs} + d. \tag{5}$$

3. PSO

The fundamental concept of PSO is based on the cooperative learning of particles to achieve optimal solutions. The particle has two properties: velocity and position. The velocity determines the direction and size of the particle movement, and the position represents the solution to the problem. The particle updates its velocity by learning from the position of the global optimal particle and its historical optimal position and finds the optimal solution by updating the particle's velocity and position for multiple iterations. The velocity and position update equations of the particle i at the $t+1$ th iteration are calculated by equations (6) and (7), respectively.

$$\begin{aligned} V_i(t+1) &= \omega * V_i(t) + C_1 * R_1 * \left(P^i_{\text{best}} - X_i(t)\right) \\ &\quad + C_2 * R_2 * \left(G_{\text{best}} - X_i(t)\right), \end{aligned} \tag{6}$$

$$X_i(t+1) = X_i(t) + V_i(t+1), \tag{7}$$

where ω is the inertia weight, C_1 is the self-learning factor, which denotes the weight coefficient for learning towards its historical optimal position; C_2 is the global learning factor, which denotes the weight coefficient for learning toward the global optimal position; R_1 and R_2 are random numbers from 0 to 1; P^i_{best} is the historical optimal position of particle i; G_{best} is the current global optimal position; $V_i(t)$ denotes the velocity of particle i at moment t, and $X_i(t)$ denotes the position of the particle i at moment t.

To improve the search accuracy and global exploration ability of PSO, an improved IPSO is proposed in this paper.

4. IPSO

4.1. Adaptive Selection Strategy. The concept of inertia plays a crucial role in PSO, as it reflects a particle's ability to maintain its previous momentum. A higher inertia weight can enhance the global exploration capability of PSO, allowing particles to discover high-quality solution regions. Conversely, reducing the inertia weight can increase the local search ability of PSO, enabling particles to improve their search accuracy in a specific region [33]. However, the global and local search capabilities of PSO are contradictory due to this property.

To solve the above problems, many scholars have proposed adaptive inertia weights to balance the exploration ability and search accuracy of the algorithm. For instance, Wang et al. [34] and Ding et al. [35] introduced adaptive inertia weight to adjust the position of followers, thereby enhancing the exploration ability and search accuracy of the SSA.

We proposed an adaptive inertia weight. Some particles will select the corresponding inertia weight based on their performance, calculated as in the following equation:

$$\begin{cases} \omega_i = \omega_{\min} + \left(\omega_{\max} - \omega_{\min}\right) * \dfrac{\left(F_i(t) - F_{\text{Gbest}}\right)}{\left(F_{\text{ave}} - F_{\text{Gbest}}\right)}, F_i(t) \geq F_{\text{ave}}, \\ \omega_i = \omega_{\min} + \left(\omega_{\max} - \omega_{\min}\right) * \dfrac{\left(F_{\text{ave}} - F_{\text{Gbest}}\right)}{\left(F_i(t) - F_{\text{Gbest}}\right)}, F_i(t) < F_{\text{ave}}, \end{cases} \tag{8}$$

where $\omega_{\min}$ is 0.4, $\omega_{\max}$ is 0.9, ω_i is the inertia weight of the particle i, $F_i(t)$ is the fitness of the particle i at time t, F_{ave} is the average fitness of the population, and F_{Gbest} is the fitness of the current global optimal particle. When the $F_i(t)$ is larger than F_{ave}, the particle i is considered to perform poorly, and the inertia weight coefficient of the particle is increased to find more high-quality solution regions. When the $F_i(t)$ is smaller than F_{ave}, the particle is considered to perform better, and its inertia weight coefficient is reduced to enhance the local search accuracy of the particle.

Traditional PSO solely computes the particle's velocity based on its historical optimal position and global optimal position, which cannot exploit the potential of each particle in the population. To address this issue, we proposed a velocity update strategy with adaptive selection, in which some particles can select the velocity update method based on their performance, which can be calculated by the following equation:

$$\begin{cases} V_i(t+1) = \omega_i * V_i(t) + C_1 * R_1 * \left(P^i_{\text{best}} - X_i(t)\right) \\ + \eta * C_2 * R_2 * \left(G_{\text{best}} - X_i(t)\right), F_i(t) > F_{\text{bad}}, \\ V_i(t+1) = \omega_i * V_i(t) + \eta * C_1 * R_1 * \left(P^i_{\text{best}} - X_i(t)\right) \\ + C_2 * R_2 * \left(G_{\text{best}} - X_i(t)\right), F_i(t) < F_{\text{good}}, \\ V_i(t+1) = \omega_i * V_i(t) + C_1 * R_1 * \left(P^i_{\text{best}} - X_i(t)\right) \\ + C_2 * R_2 * \left(G_{\text{best}} - X_i(t)\right), F_{\text{good}} \leq F_i(t) \leq F_{\text{bad}}, \end{cases} \tag{9}$$

where F_{bad} is the fitness threshold of the poorly performing particles and F_{good} is the fitness threshold of the well-performing particles, η is a random number from 0 to 1. The poorly performing particles tend to search around their historical best position to enhance their global exploration ability. The well-performing particles tend to search around the global best position to enhance local search accuracy and the algorithm's convergence speed. Particles with average performance learn from the historical optimal position and the global optimal position to find more high-quality solution regions.

4.2. Positive Feedback Factor. Li et al. [18] adds a positive feedback factor to the traditional PSO, which increases the local search capability of PSO by adjusting the position of the particles based on their fitness after iteration compared with that before iteration. In this paper, we improve the positive feedback factor proposed by [18], as in the following equation:

$$\begin{cases} X_i(t+1) = X_i(t+1) + \theta_i, F_i(t+1) \geq F_i(t), \\ X_i(t+1) = X_i(t+1) + \theta_i, F_i(t+1) < F_i(t) \,\&\, F_i(t+1) \geq F_{\text{ave}}, \\ X_i(t+1) = X_i(t+1) + \dfrac{1}{2} * \text{Rand} * \theta_i, F_i(t+1) < F_i(t) \& F_i(t+1) < F_{\text{ave}}, \end{cases} \tag{10}$$

where θ_i is the positive feedback factor of the particle i, Rand is a random number from 0 to 1, and t is the number of current iterations. When $F_i(t+1)$ is lower than $F_i(t)$ but higher than F_{ave}, the particle i moves θ_i length to speed up its entry into the high-quality solution region. When $F_i(t+1)$ is lower than $F_i(t)$ and lower than F_{ave}, particle i moves $1/2 * \text{Rand} * \theta_i$ length increasing the local search accuracy by reducing the extra step added due to the positive feedback factor while accelerating the entry into the high-quality solution region.

4.3. Initialization Strategy. In PSO, the diversity of the population plays a crucial role in determining the quality of the solution obtained. However, as the algorithm runs, the diversity of the population tends to decrease, especially in the later stages of the algorithm when particles cluster together, leading to premature convergence. To improve the diversity of the population, this paper uses an initialization strategy that classifies particles into excellent-performing, average-performing, and poor-performing based on their fitness values. At the beginning of each iteration, the initialization operation is performed on some of the poorly performing particles to increase the diversity of the population and prevent the algorithm from maturing prematurely.

4.4. Addressing the Local Optimum Strategy. Trapping in the local optimum has been a shortcoming of PSO that is difficult to overcome. Therefore, this paper proposes a method to address local optimum by adding a random perturbation and random crossover.

If the global optimal particle stays the same for a long period, the algorithm is judged to be in the local optimum, and a random perturbation of length δ is added to the global optimal particle. Then, check whether the fitness after adding the perturbation is lower than the fitness before adding the perturbation. If the fitness is lower, the perturbed particle is saved as a new global optimal particle. Otherwise, the perturbation is discarded, and the same operation is repeated in the next iteration. The δ can be calculated by the following equation:

$$\delta = \beta * \text{Rand}, \tag{11}$$

where β is the value assigned based on the F_{Gbest}. β is 0.2 if the current global optimal particle is in the high-quality solution region. The β is 0.5 if the current global optimal particle is in the inferior solution region. The process of adding random perturbations to make the algorithm address the local optimum is shown in Figure 2.

Besides adding random perturbations, a random crossover strategy is designed to make the algorithm address the local optimum.

After judging that the algorithm is trapped in the local optimum, replace a point in the global optimal particle with a point at the corresponding position of a random particle. The random crossover process is shown in Figure 3.

5. IDWA

5.1. Fusion with VO. VO has the ability to predict whether a robot will collide with dynamic obstacles and has been widely used in recent years for robot avoidance of dynamic obstacles [36]. The principle of VO prediction collision is shown in Figure 4 where V_{rob} and V_{obs} are the velocities of the robot and the dynamic obstacle, respectively. V_r is the relative velocity of the robot and the dynamic obstacle, and R_{rob} and R_{obs} are the radius of the robot and the dynamic obstacle, respectively.

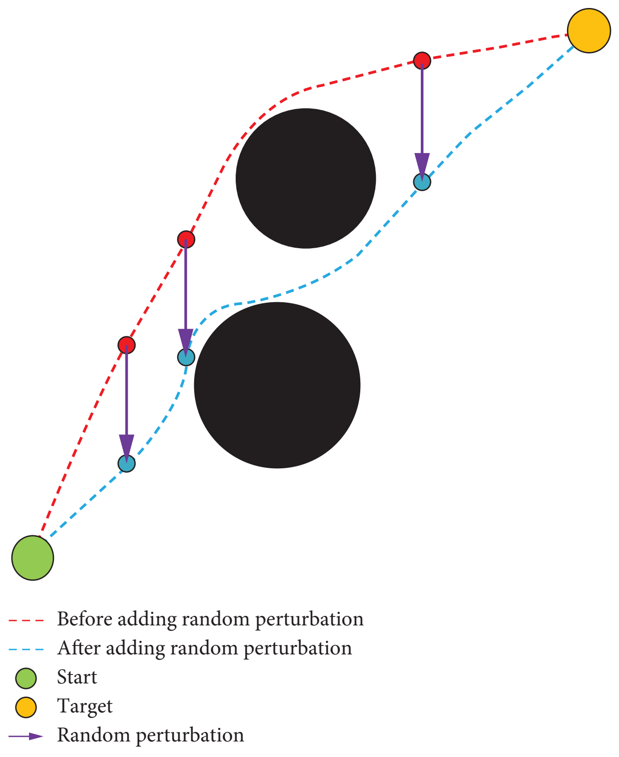
The relative velocity V_r of the robot to the obstacle can be calculated by the following equation:

$$V_r = V_{\text{rob}} \oplus - V_{\text{obs}}, \tag{12}$$

where $\oplus$ is the Minkowski sum. Define a ray starting from the robot center P_r along the V_r as the following equation:

$$\lambda(P_r, V_r) = \{P_r + V_r t | t \geq 0\}, \tag{13}$$

where t denotes time, so ray $\lambda(P_r, V_r)$ represents the real-time relative position of the robot and the obstacle if the robot and the obstacle maintain their motion without changing. Define the set V_r as the relative collision cone (RCC), the mathematical representation of the RCC is as in the following equation:



Before adding random perturbation
After adding random perturbation
Start
Target
Random perturbation

Figure 2: The process of adding random perturbation.

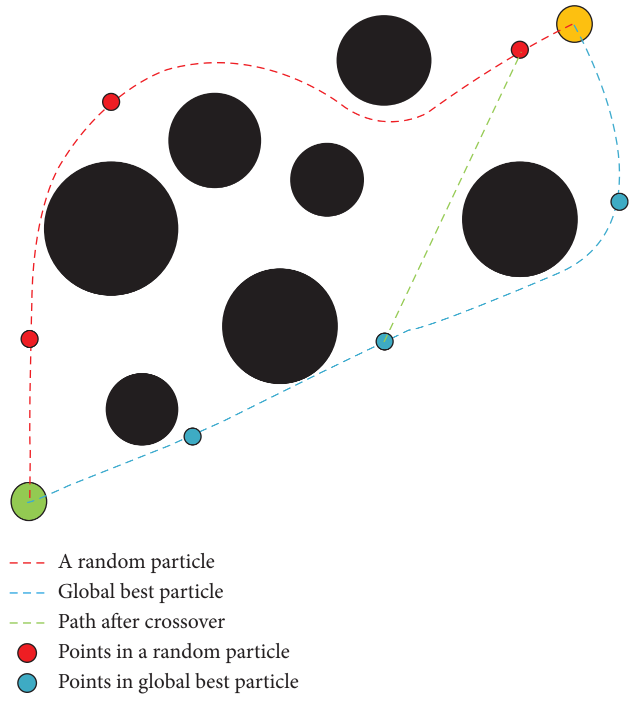


Figure 3: The process of random crossover.

$$\mathrm{RCC} = \left\{ V_r \middle| \lambda(P_r, V_r) \cap \text{Obstacle} \neq \varnothing \right\}, \tag{14}$$

where Obstacle is the influence range after the obstacle radius R_{obs} expansion R_{rob}.

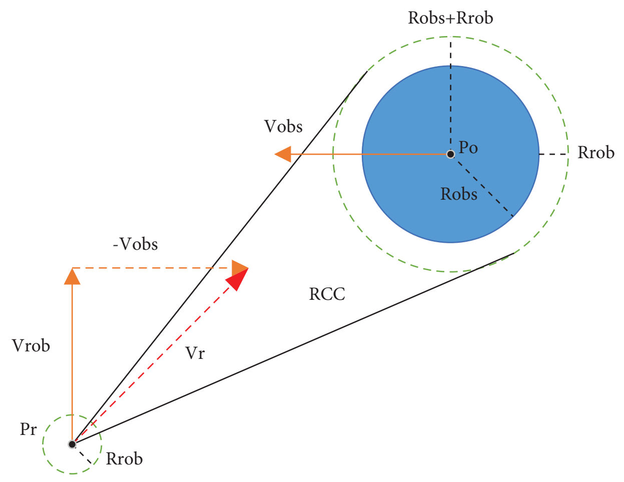


Figure 4: The principle of VO prediction collision.

If $\lambda(P_r, V_r)$ is within the RCC, it indicates the risk of collision between the robot and the dynamic obstacle. In this paper, we use a fusion of VO and IDWA for dynamic obstacle avoidance.

5.2. Improving the heading (v, ω) *Function.* The heading(v, ω) function of conventional DWA is calculated by the following equation:

$$\mathrm{heading}(v, \omega) = 180 - \alpha, \tag{15}$$

where α is the angular deviation of the robot orientation at the end of the trajectory from the target point. The traditional DWA heading(v, ω) function may cause the situation in Figure 5. It can be seen that the robot is in a better state after one unit of time following trajectory 1 than after one unit of time following trajectory 2. However, the calculation results based on the heading(v, ω) function shows that trajectory 2 has a higher score, which is unreasonable.

To solve this problem, we proposed an improved Heading(v, ω) function. When the robot approaches the target point, the heading(v, ω) function no longer calculates the angle between the robot's orientation at the end of the trajectory and the target point but instead calculates the angle between the robot's orientation after moving one unit of time and the target point, as shown in Figure 6.

5.3. Improving the dist (v, ω) *Function.* When modelling for the static obstacles, the influence range of the static obstacles is increased based on the size of the robot, so they do not collide when the global path is close to the obstacles. The dist(v, ω) function causes the robot to select the trajectory movement away from the static obstacles, which not only weakens the guidance effect of the global path on local path planning but also increases the path length of local path planning, as shown in Figure 7.

To address these problems, the dist(v, ω) function has been improved as the following equation:

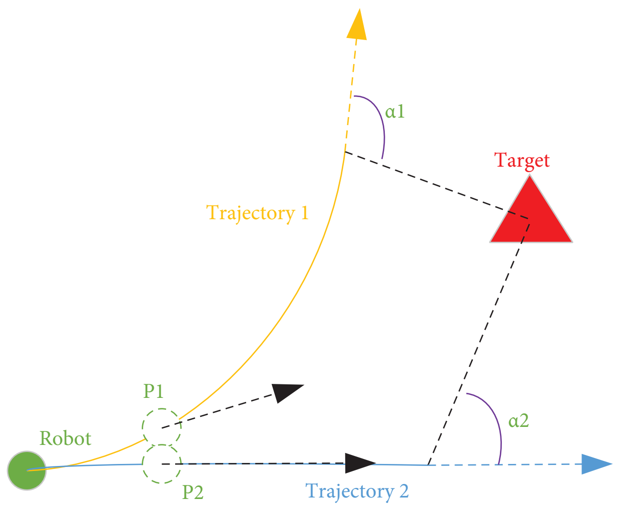


FIGURE 5: Analysis of unreasonable heading (v, ω) function.

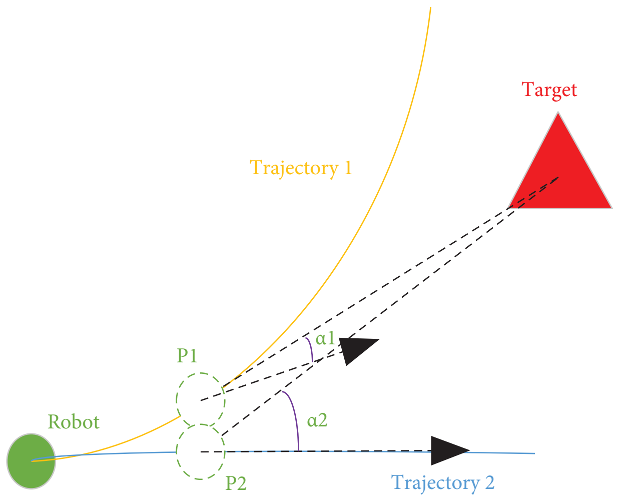


FIGURE 6: Improved Heading (v, ω) function.

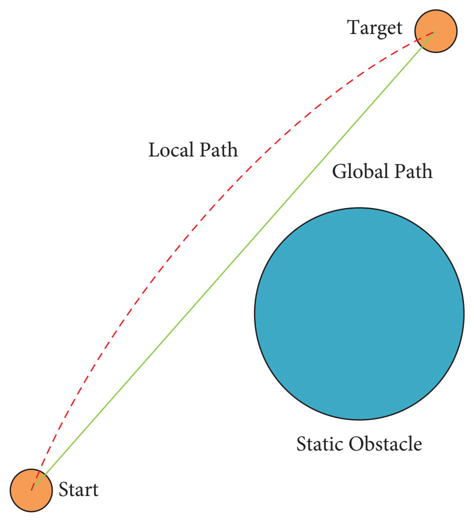


FIGURE 7: Analysis of unreasonable dist (v, ω) function.

$$\text{Dist}(v,\omega) = \begin{cases} 1, \sqrt{(x_{\text{rob}} - x_{\text{obs}})^2 + (y_{\text{rob}} - y_{\text{obs}})^2} > R, \\ 0.1, \sqrt{(x_{\text{rob}} - x_{\text{obs}})^2 + (y_{\text{rob}} - y_{\text{obs}})^2} \le R, \end{cases} \tag{16}$$

where x_{rob}, x_{obs}, y_{rob}, and y_{obs} represent the horizontal and vertical coordinates of the robot and the static obstacle, respectively, and R is the radius of influence of the obstacle. The improved Dist (v, ω) function allows the robot to select a trajectory that is close to but without collision with static obstacles, which enhances the guidance effect of the global path on local path planning.

5.4. Adding the vh(v, ω) Function. To make IDWA more effective in avoiding dynamic obstacles, the $vh(v, \omega)$ function is added to the evaluation function to calculate the change in angle difference with the obstacle before and after robot movement. The $vh(v, \omega)$ function can be calculated by the following equation:

$$\begin{cases} vh(v,\omega) = |VA(t+1) - VA(t)|, \\ VA(t) = 0, VA(t) = \pi, \\ vh(v,\omega) = VA(t+1) - VA(t), \\ VA(t) \ne 0, VA(t) \ne \pi, \end{cases} \tag{17}$$

where $VA(t)$ and $VA(t+1)$ denote the angular difference between the velocity of the robot and the velocity of the dynamic obstacle before and after motion, respectively. After adding the $vh(v, \omega)$ function, the algorithm will select trajectories for the robot that increase the angular difference between the velocity of the robot and the velocity of the dynamic obstacle, as shown in Figure 8.

The evaluation function of IDWA is stated in the following equation:

$$\begin{aligned} G(v,\omega) = \sigma(&\alpha\,\text{Heading}(v,\omega) + \beta\,\text{Dist}(v,\omega) \\ &+ \gamma\text{vel}(v,\omega) + \lambda vh(v,\omega)), \end{aligned} \tag{18}$$

where σ is a normalization operation. The α, β, γ, and λ are the weighting factors of Heading (v, ω), Dist (v, ω), vel (v, ω), and $vh(v, \omega)$, respectively. The λ is calculated as in the following equation:

$$\begin{cases} \lambda = \frac{3}{2} * \alpha, & \text{is collision} = 1, \\ \lambda = 0, & \text{is collision} = 0, \end{cases} \tag{19}$$

where is collision is the judgment result of VO, and the value of is collision is 1 if the robot will collide with the dynamic obstacle, otherwise is collision is 0.

6. Algorithm Fusion

To complete the path planning of mobile robots in dynamic environments, this paper proposes a hybrid algorithm combining IPSO and IDWA. In the proposed hybrid algorithm, multiple navigation points are identified in the

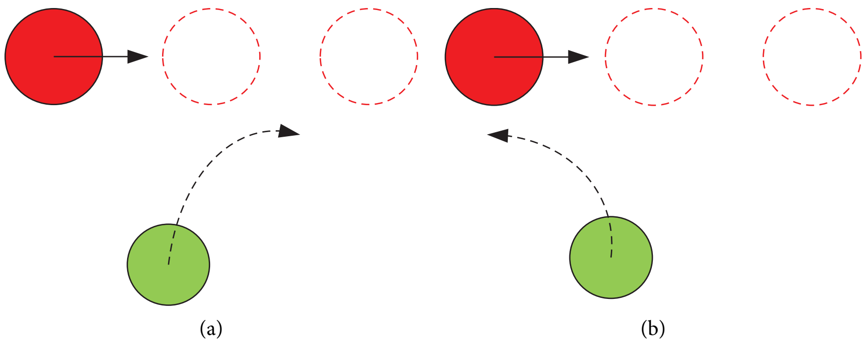


Figure 8: Comparison of the effect of dynamic obstacle avoidance. (a) The Conventional DWA dynamic obstacle avoidance. (b) Add $vh(v, \omega)$ function dynamic obstacle avoidance.

global path generated by IPSO. The robot selects the most suitable navigation point as the target point for IDWA based on its current position and the risk of collision with obstacles and iteratively updates its target point until reaching the global target point. Selecting navigation points is crucial for IDWA. If the selected navigation point is too far away from the robot, it may not effectively guide the robot, while if the selected navigation point is too close to the robot, it may not effectively avoid dynamic obstacles. For this reason, we proposed a method for selecting navigation points on global paths.

This paper uses the cubic spline interpolation to optimize the global path, assuming there are Num points in the global path, and the robot's location is the i th point. If the distance of the robot from dynamic obstacles does not reach the obstacle avoidance distance or reaches the obstacle avoidance distance but the robot will not collide with dynamic obstacles, the $i + n$ th point will be chosen as the navigation point for IDWA. If the distance of the robot from dynamic obstacles is less than the obstacle avoidance distance and the robot will collide with the dynamic obstacle, the $i + kn$ th point will be chosen as the navigation point for IDWA. Where the n depends on the velocity of the robot, and the k depends on the obstacle avoidance distance and the size of the dynamic obstacle.

The steps of the hybrid algorithm are as follows.

Step 1: Modelling of the robot's working environment.

Step 2: Get the global path by using IPSO.

Step 3: Divides the global path into multiple navigation points.

Step 4: Use the above method to select the navigation points in turn as the target points for local path planning, and use IDWA to complete the local path planning.

Step 5: The robot reaches the global target point, and the path planning ends.

The flow chart of the fusion path planning method is shown in Figure 9.

7. Simulation Experimental Analysis

To determine the effectiveness of the IPSO, IDWA, and hybrid algorithm, we use the MATLAB 2021a software to conduct simulation experiments on a computer with Win 10 and 8GB RAM.

7.1. Simulation Experimental of IPSO.

7.1. Simulation Experimental of IPSO. To determine that IPSO has strong exploration ability and search accuracy, this paper conducts simulation experiments and data comparisons with traditional PSO, [18] improved particle swarm optimization (IPSO1), and [19] improved particle swarm optimization (IPSO2) in $10 * 10$ and $15 * 15$ environments, respectively.

To minimize the effect of the randomness of the intelligent algorithm, 50 simulation experiments are conducted in two environments, and the average path length and shortest path length of the four algorithms are compared.

The average path length and shortest path length can reflect the exploration ability and search accuracy of the algorithm, respectively. The coordinates of the start and target points of Environment 1 and Environment 2 are (0, 0), (10, 10) and (0, 0), (15, 15), respectively. The start and target points are represented by yellow squares and green squares, respectively. The relevant parameters set in this paper are as follows: the number of populations is 50, the maximum number of iterations is 300, and the learning factor $C_1 = C_2 = 1.5$, the inertia weight $\omega_{\min} = 0.4$, and $\omega_{\max} = 0.9$.

The simulation experimental data of the four algorithms in environment 1 and environment 2 are described in Tables 1 and 2, respectively.

The shortest paths of the four algorithms in Environment 1 and Environment 2 are shown in Figures 10 and 11, respectively.

The simulation experimental data demonstrate that the average path length and shortest path length obtained by IPSO are superior to the other three algorithms in both environments. This is attributed to the superior exploration ability and search accuracy of IPSO.

To determine the effectiveness of the address local optimum strategy and the faster convergence speed of IPSO, the convergence curves in the two environments are shown in Figure 12(a) and Figure 12(b), respectively. In this experiment, the number of populations is 150, the maximum number of iterations is 500, and the learning factor $C_1 = C_2 = 1.5$, the inertia weight $\omega_{\min} = 0.4$, and $\omega_{\max} = 0.9$.

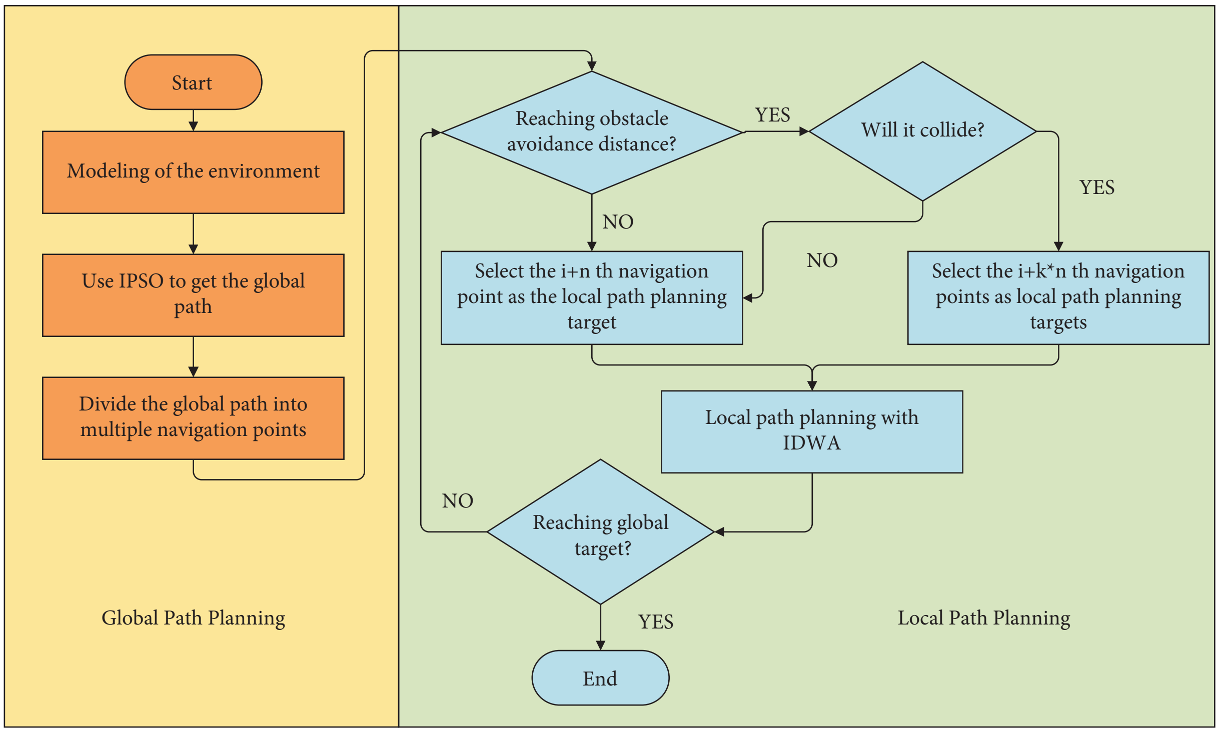

Start
Modeling of the environment
Use IPSO to get the global path
Divide the global path into multiple navigation points
Global Path Planning
Reaching obstacle avoidance distance?
YES
Will it collide?
YES
NO
NO
Select the i+n th navigation point as the local path planning target
Select the i+k*n th navigation points as local path planning targets
Local path planning with IDWA
NO
Reaching global target?
YES
End
Local Path Planning

Figure 9: Flow chart of the fusion algorithm.

Table 1: Simulation experimental data of environment 1.

Environment 1	Average path length (m)	Shortest path length (m)
PSO	15.7687	14.9860
IPSO1 [18]	15.5512	14.7632
IPSO2 [19]	15.4805	14.6825
IPSO	**15.0966**	**14.5374**

The bold values are the smallest among the four groups, representing the average path length and the shortest path length, respectively.

Table 2: Simulation experimental data of environment 2.

Environment 2	Average path length (m)	Shortest path length (m)
PSO	22.4382	21.8782
IPSO1 [18]	22.2542	21.8041
IPSO2 [19]	22.2667	21.6564
IPSO	**22.0474**	**21.2866**

The bold values are the smallest among the four groups, representing the average path length and the shortest path length, respectively.

The convergence curves in Figure 12 demonstrate that IPSO outperforms PSO, IPSO1, and IPSO2 in terms of convergence speed and convergence path length in environments 1 and 2. The convergence path length of IPSO in environment 1 and environment 2 are reduced at the 343th and 387th iteration, respectively, proved the effectiveness of the address local optimum strategy. The convergence path length in two environments is shown in Table 3.

7.2. Simulation Experimental of IDWA. To determine the obstacle avoidance capability of IDWA, the simulation experiments of avoiding dynamic obstacles were compared using DWA and IDWA, respectively.

The radius of the robot in this experiment is 0.1 m, the maximum linear velocity is 1 m/s, the maximum acceleration is 0.2 m/s^2, the maximum rotational velocity is 40/s, the maximum rotational acceleration is 40/s^2, the initial

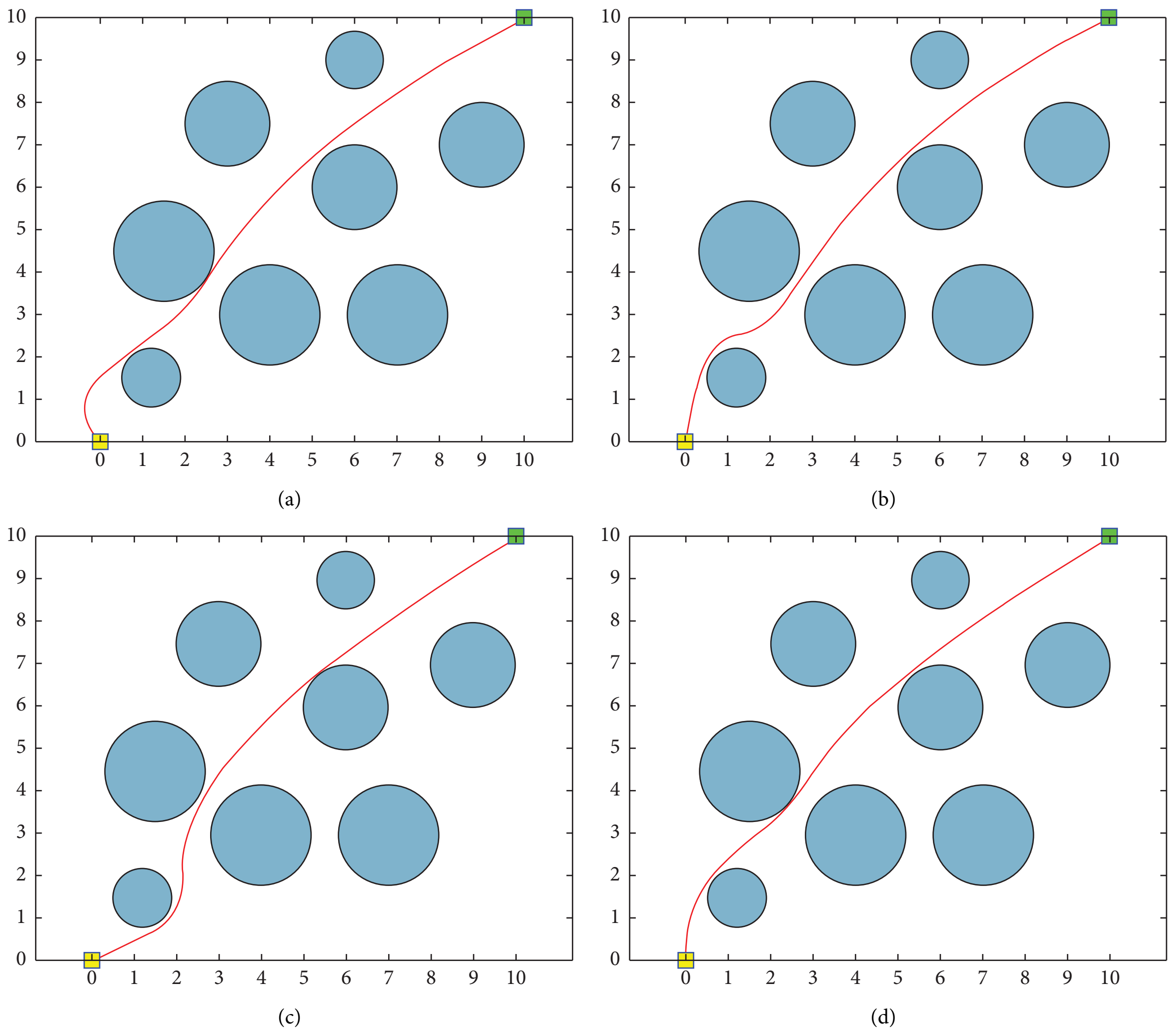


Figure 10: The shortest path of four algorithms in environment 1. (a) PSO in environment 1. (b) IPSO1 in environment 1. (c) IPSO2 in environment 1. (d) IPSO in environment 1.

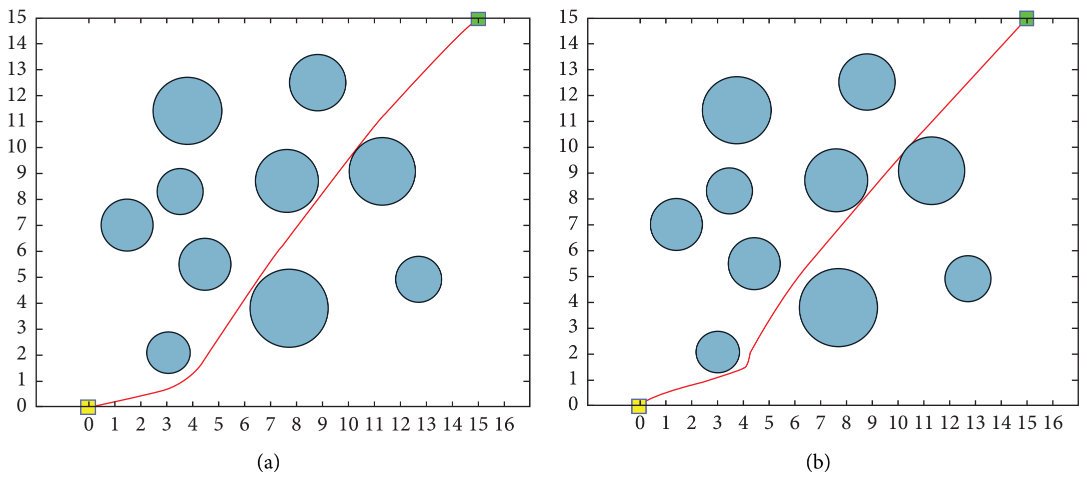


Figure 11: Continued.

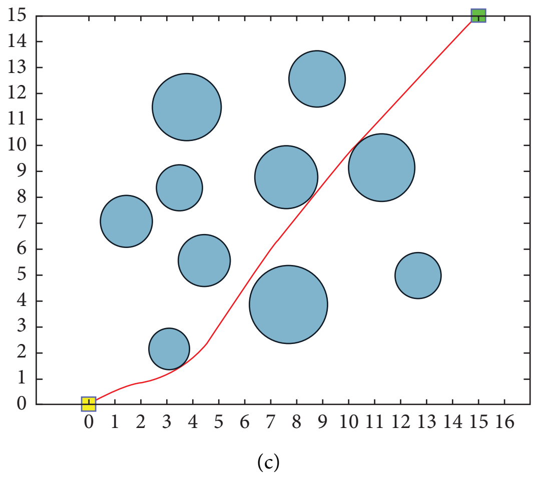
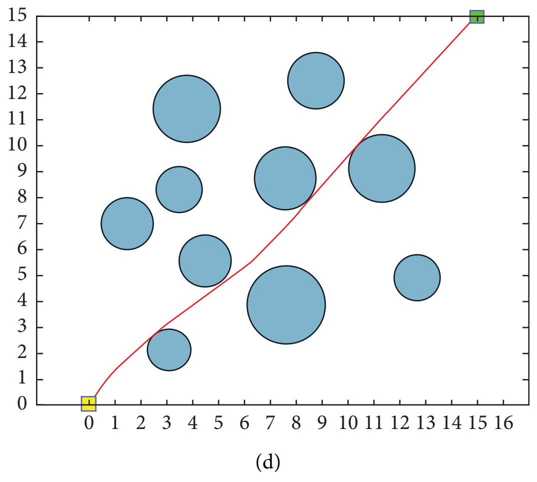



Figure 11: The shortest path of four algorithms in environment 2. (a) PSO in environment 2. (b) IPSO1 in environment 2. (c) IPSO2 in environment 2. (d) IPSO in environment 2.

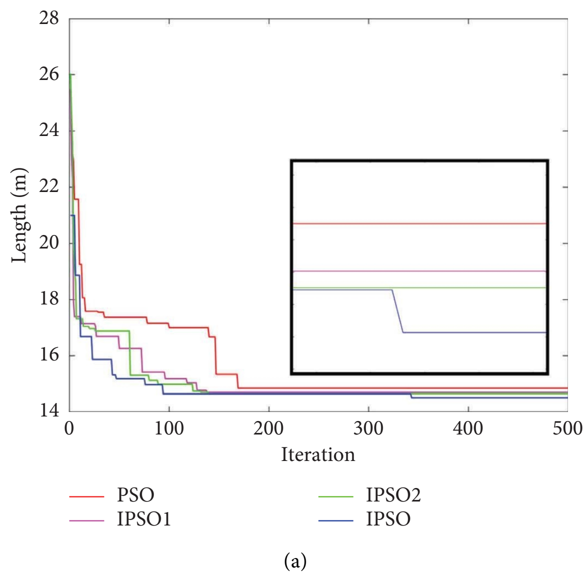
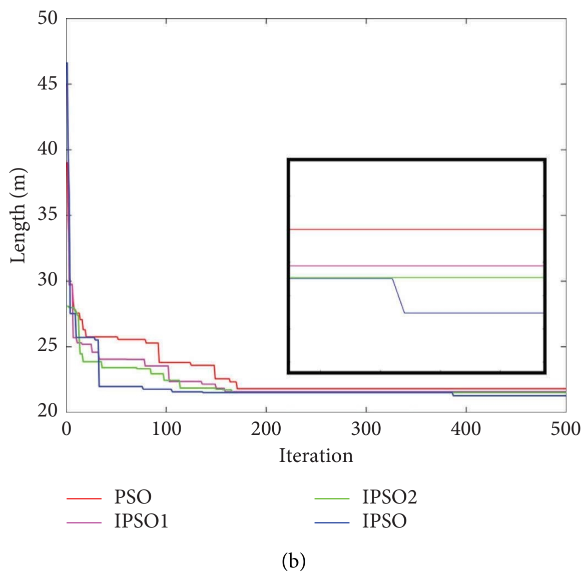



Figure 12: The convergence curves in two environments. (a) Convergence curve of environment 1. (b) Convergence curve of environment 2.

Table 3: Convergence path length and number of convergence iterations in environment 1 and environment 2.

	Convergence path length in environment 1 (m)	Number of convergence iterations in environment 1	Convergence path length in environment 2 (m)	Number of convergence iterations in environment 2
PSO	14.8496	169	21.7977	171
IPSO1 [11]	14.7031	138	21.5781	159
IPSO2 [12]	14.6513	140	21.5085	166
IPSO	**14.5134**	**94**	**21.2950**	**137**

The values in bold represent the best of the four sets of values.

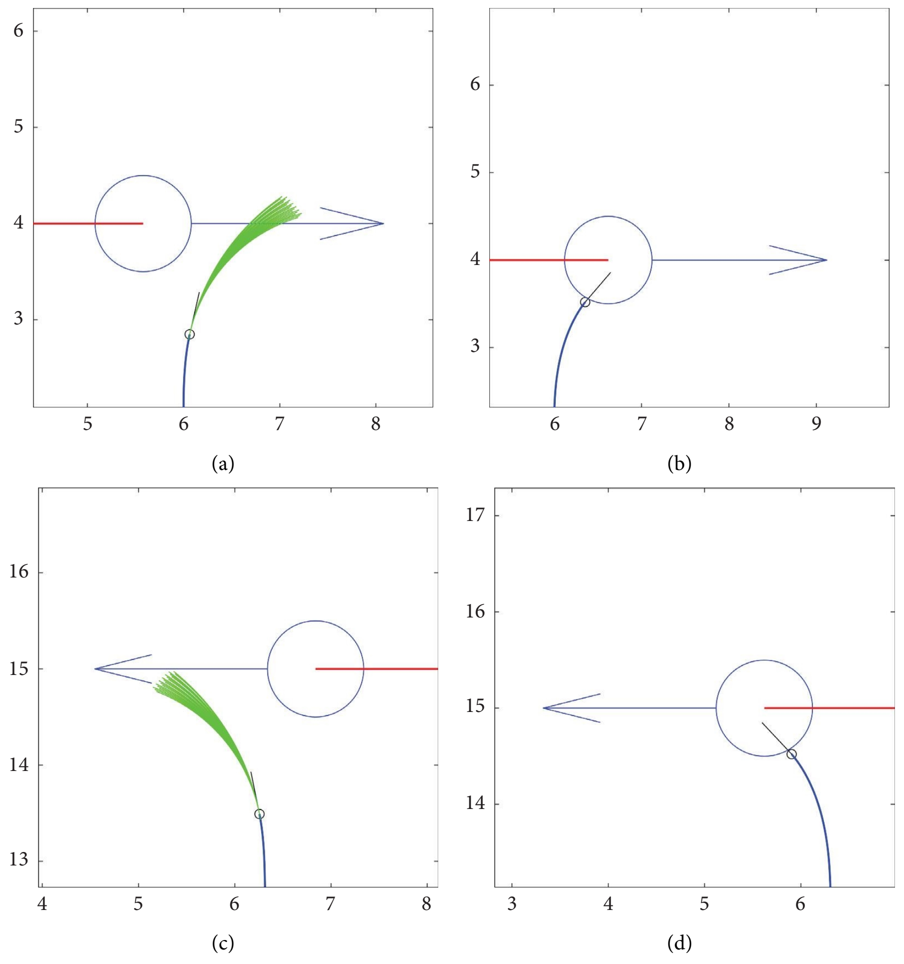


Figure 13: Results of DWA avoidance of dynamic obstacles. (a) Robot detects dynamic obstacle 1. (b) Robot avoidance of dynamic obstacle 1 failed. (c) Robot detects dynamic obstacle 2. (d) Robot avoidance of dynamic obstacle 2 failed.

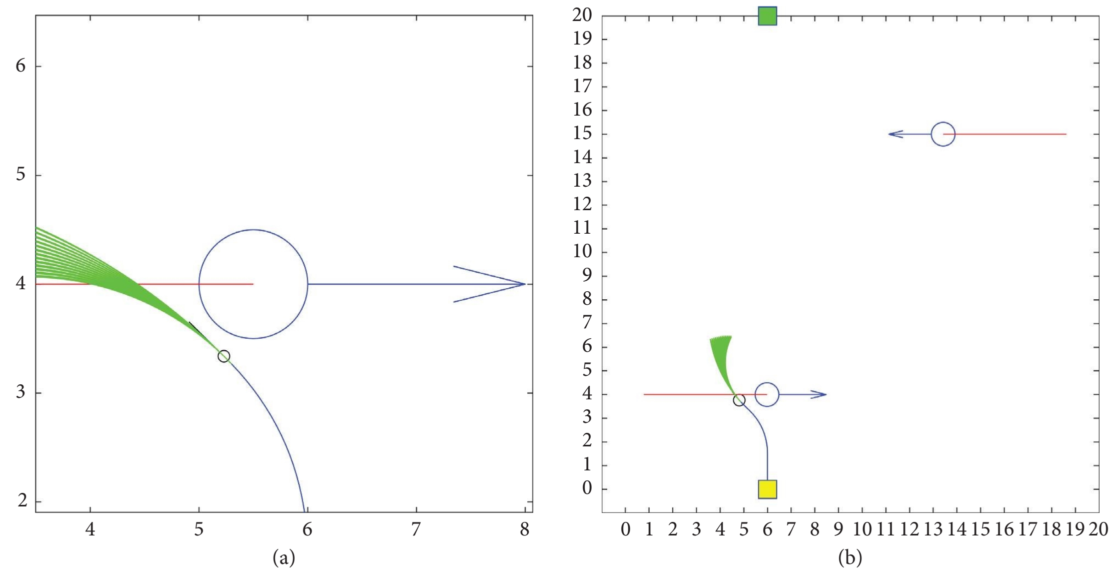
orientation angle is 90°, the radius of the dynamic obstacle is 0.6 m, and the velocity is 0.6 m/s. The results of the simulation experiments are shown in Figures 13 and 14, respectively.

The $\mathrm{dist}(v, \omega)$ function of conventional DWA drives the robot to turn to the side with the same direction of motion as the dynamic obstacle, and the distance of the robot from the dynamic obstacle is too close, leading to an empty set of velocity sampling space, resulting in obstacle avoidance failure, as shown in Figure 13.

If the robot is judged to be at risk of collision with the dynamic obstacle by using VO, the $vh(v, \omega)$ function in the IDWA evaluation function drives the robot to turn in the direction opposite to the dynamic obstacle's motion direction, which not only successfully avoids the dynamic obstacle but also shortens the path length increased by avoiding the dynamic obstacle, as shown in Figure 14.

7.3. Algorithm Fusion. The above experiments demonstrated the effectiveness of IPSO and IDWA. To further assess whether the fusion algorithm is effective for robot path planning in the dynamic environment, it will be verified in a dynamic environment of $25 * 25$.

In this experiment, the robot has a radius of 0.1 m, a maximum linear velocity of 1 m/s, a maximum acceleration of 0.2 m/s^2, the maximum rotational velocity of 40/s, a rotational acceleration of 40/s^2, and an initial orientation angle of 36°. The coordinates of the starting point are (0, 0) and the coordinates of the global target point are (23, 23), which are represented as yellow and green squares, respectively, the red $*$ is navigation point for IDWA. The radius of both dynamic obstacles is 0.6 m and the velocity is 0.8 m/s. The process and results of the fusion algorithm for robot path planning in dynamic environments are shown in Figure 15.



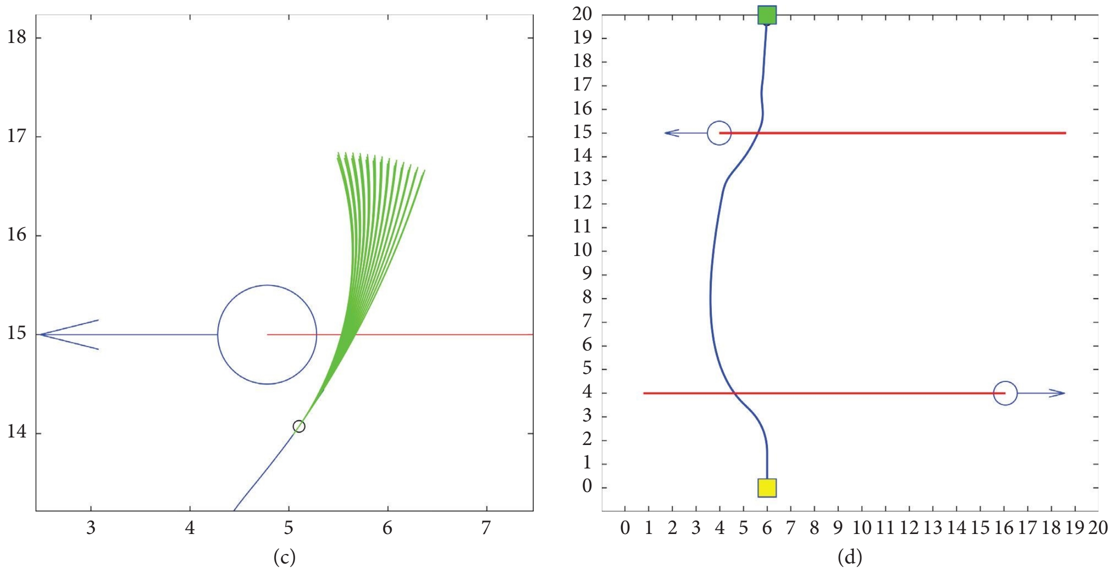


Figure 14: Results of IDWA avoidance of dynamic obstacles. (a) Robot detects dynamic obstacle 1. (b) Robot avoids dynamic obstacles 1. (c) Robot detects dynamic obstacle 2. (d) Robot avoids all dynamic obstacles.

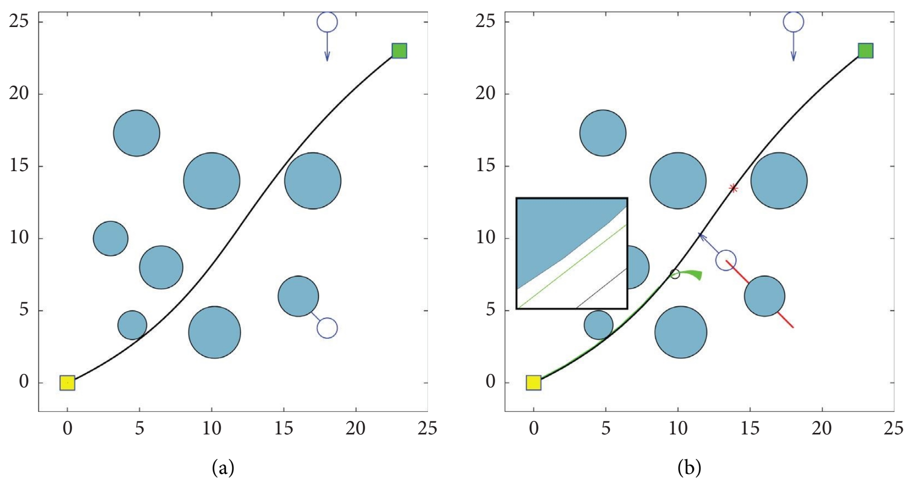


Figure 15: Continued.

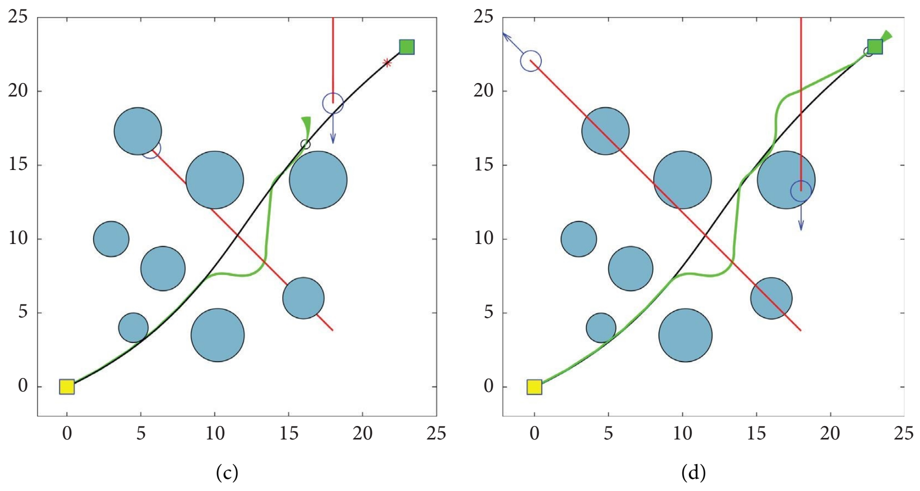


Figure 15: Fusion algorithm in the dynamic environment. (a) The global paths in the environment obtained using IPSO. (b) The fusion algorithm for the avoidance of dynamic obstacle 1. (c) The fusion algorithm for the avoidance of dynamic obstacle 2. (d) The fusion algorithm completes path planning in the dynamic environment.

The black line is the global path planned by IPSO, the green line is the trajectory of IDWA, and the red line is the trajectory of two dynamic obstacles.

From the experimental results, it can be seen that the global path and the local path fit closely when there are no dynamic obstacles. Under the direction of the global path, IDWA successfully completes the dynamic obstacle avoidance and path planning tasks.

8. Conclusion

This paper proposes a hybrid algorithm for the path planning of robots in dynamic environments.

To improve the traditional PSO, IPSO is proposed to enhance the exploration ability, search accuracy, and address the local optimum to obtain a shorter global path. Firstly, the exploration capability and search accuracy of PSO are increased using an adaptive selection strategy and a positive feedback factor. Secondly, the initialized particle strategy is used to increase population diversity. Finally, the strategy of adding random perturbation and random crossover is used to help the algorithm address the local optimum. In addition, the global path is optimized using cubic spline interpolation to make the global path smoother and have smaller turning angles. Simulation experiments prove that the IPSO has better exploration ability and search accuracy.

To improve the disadvantages of DWA trapped in local optimums and insufficient obstacle avoidance, an IDWA is proposed. Firstly, the global path is used to provide global guidance for the robot, which improves the problem of DWA getting stuck at the local optimum due to the lack of global guidance. Secondly, the Heading(v,ω) and Dist(v,ω) functions are improved to enhance the guidance of the global path to the local path planning. Finally, VO is used to determine whether the robot will collide with dynamic obstacles, and the $vh(v,\omega)$ function is added to the evaluation function to improve the ability of the algorithm to avoid dynamic obstacles. The simulation experiments prove that IDWA can effectively complete the path planning of the robot in a dynamic environment with the guidance of the global path.

Data Availability

The path length and the number of convergence iterations used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper was financially supported by Zhenjian Yang's research team at the School of Computer and Information Engineering, Tianjin Chengjian University Tianjin City Enterprise Science and Technology Specialists Project (19JCTPJC50000).

References

[1] K. Hao, J. L. Zhao, B. B. Wang, Y. L. Liu, and C. Q. Wang, "The application of an adaptive genetic algorithm based on collision detection in path planning of mobile robots," *Computational Intelligence and Neuroscience*, vol. 2021, Article ID 5536574, 20 pages, 2021.

[2] B. Hichri, A. Gallala, F. Giovannini, and S. Kedziora, "Mobile robots path planning and mobile multirobots control: a review," *Robotica*, vol. 40, no. 12, pp. 4257–4270, 2022.

[3] Z. A. Zhang, Y. X. Wan, Y. Wang, X. Q. Guan, W. Ren, and G. Li, "Improved hybrid A* path planning method for spherical mobile robot based on pendulum," *International Journal of Advanced Robotic Systems*, vol. 18, no. 1, Article ID 172988142199295, 2021.

[4] C. W. Miao, G. Z. Chen, C. L. Yan, and Y. Y. Wu, "Path planning optimization of indoor mobile robot based on

adaptive ant colony algorithm," *Computers & Industrial Engineering*, vol. 156, Article ID 107230, 2021.

[5] R. Sarkar, D. Barman, and N. Chowdhury, "Domain knowledge based genetic algorithms for mobile robot path planning having single and multiple targets," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 7, pp. 4269–4283, 2022.

[6] B. N. Ornek, S. B. Aydemir, T. Duzenli, and B. Ozak, "A novel version of slime mould algorithm for global optimization and real world engineering problems," *Mathematics and Computers in Simulation*, vol. 198, pp. 253–288, 2022.

[7] X. B. Cheng, L. C. Zhu, H. H. Lu, J. Z. Wei, and N. Wu, "Robot path planning based on an improved salp swarm algorithm," *Journal of Sensors*, vol. 2022, Article ID 2559955, 16 pages, 2022.

[8] Z. H. Yu, Z. J. Si, X. B. Li, D. Wang, and H. B. Song, "A novel hybrid particle swarm optimization algorithm for path planning of UAVs," *IEEE Internet of Things Journal*, vol. 9, no. 22, pp. 22547–22558, 2022.

[9] Q. W. Qian, J. F. Wu, and Z. Wang, "Optimal path planning for two-wheeled self-balancing vehicle pendulum robot based on quantum-behaved particle swarm optimization algorithm," *Personal and Ubiquitous Computing*, vol. 23, no. 3-4, pp. 393–403, 2019.

[10] H. J. Ji, H. Hu, and X. G. Peng, "Multi-underwater gliders coverage path planning based on ant colony optimization," *Electronics*, vol. 11, no. 19, p. 3021, 2022.

[11] L. Z. Zhai and S. H. Feng, "A novel evacuation path planning method based on improved genetic algorithm," *Journal of Intelligent and Fuzzy Systems*, vol. 42, no. 3, pp. 1813–1823, 2022.

[12] Z. Wang, H. Ding, Z. Yang, B. Li, Z. Guan, and L. Bao, "Rank-driven salp swarm algorithm with orthogonal opposition-based learning for global optimization," *Applied Intelligence*, vol. 52, no. 7, pp. 7922–7964, 2022.

[13] Z. Wang, H. Ding, J. Wang et al., "Adaptive guided salp swarm algorithm with velocity clamping mechanism for solving optimization problems," *Journal of Computational Design and Engineering*, vol. 9, no. 6, pp. 2196–2234, 2022.

[14] D. Agarwal and P. S. Bharti, "Implementing modified swarm intelligence algorithm based on Slime moulds for path planning and obstacle avoidance problem in mobile robots," *Applied Soft Computing*, vol. 107, Article ID 107372, 2021.

[15] L. Zhang, Y. J. Zhang, and Y. F. Li, "Mobile robot path planning based on improved localized particle swarm optimization," *IEEE Sensors Journal*, vol. 21, no. 5, pp. 6962–6972, 2021.

[16] J. B. Zhong, B. Y. Li, S. X. Li, F. R. Yang, P. H. Li, and Y. Cui, "Particle swarm optimization with orientation angle-based grouping for practical unmanned surface vehicle path planning," *Applied Ocean Research*, vol. 111, Article ID 102658, 2021.

[17] L. Wang, L. L. Liu, J. Y. Qi, and W. P. Peng, "Improved quantum particle swarm optimization algorithm for offline path planning in AUVs," *IEEE Access*, vol. 8, pp. 143397–143411, 2020.

[18] W. Li, M. Tan, L. Wang, and Q. Z. Wang, "A cubic spline method combing improved particle swarm optimization for robot path planning in dynamic uncertain environment," *International Journal of Advanced Robotic Systems*, vol. 17, no. 1, Article ID 172988141989166, 2020.

[19] Z. Tan, H. G. Liang, D. Zhang, and Q. G. Wang, "Path planning of surgical needle: a new adaptive intelligent particle swarm optimization method," *Transactions of the Institute of Measurement and Control*, vol. 44, no. 4, pp. 766–774, 2022.

[20] M. Zohaib, S. M. Pasha, N. Javaid, A. Salaam, and J. Iqbal, "An improved algorithm for collision avoidance in environments having U and H shaped obstacles," *Studies in Informatics and Control*, vol. 23, no. 1, pp. 97–106, 2014.

[21] X. Yan, D. P. Jiang, R. L. Miao, and Y. L. Li, "Formation control and obstacle avoidance algorithm of a multi-USV system based on virtual structure and artificial potential field," *Journal of Marine Science and Engineering*, vol. 9, no. 2, p. 161, 2021.

[22] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, no. 3, pp. 463–497, 2015.

[23] M. Kobayashi and N. Motoi, "Local path planning: dynamic window approach with virtual manipulators considering dynamic obstacles," *IEEE Access*, vol. 10, pp. 17018–17029, 2022.

[24] T. Hossain, H. Habibullah, R. Islam, and R. V. Padilla, "Local path planning for autonomous mobile robots by integrating modified dynamic-window approach and improved follow the gap method," *Journal of Field Robotics*, vol. 39, no. 4, pp. 371–386, 2022.

[25] Z. Q. Jian, S. Y. Zhang, S. T. Chen, Z. X. Nan, and N. N. Zheng, "A global-local coupling two-stage path planning method for mobile robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5349–5356, 2021.

[26] L. S. Liu, J. X. Yao, D. W. He et al., "Global dynamic path planning fusion algorithm combining jump-A* algorithm and dynamic window approach," *IEEE Access*, vol. 9, pp. 19632–19638, 2021.

[27] X. Y. Ji, S. Feng, Q. D. Han, H. F. Yin, and S. W. Yu, "Improvement and fusion of A* algorithm and dynamic window approach considering complex environmental information," *Arabian Journal for Science and Engineering*, vol. 46, no. 8, pp. 7445–7459, 2021.

[28] Y. Jin, M. Yue, W. J. Li, and J. Y. Shangguan, "An improved target-oriented path planning algorithm for wheeled mobile robots," *Proceedings of the Institution of Mechanical Engineers - Part C: Journal of Mechanical Engineering Science*, vol. 236, no. 22, pp. 11081–11093, 2022.

[29] L. W. Yang, L. X. Fu, P. Li, J. L. Mao, and N. Guo, "An effective dynamic path planning approach for mobile robots based on ant colony fusion dynamic windows," *Machines*, vol. 10, no. 1, p. 50, 2022.

[30] Q. B. Jin, C. N. Tang, and W. Cai, "Research on dynamic path planning based on the fusion algorithm of improved ant

colony optimization and rolling window method," *IEEE Access*, vol. 10, pp. 28322–28332, 2022.

[31] H. Miao and Y. C. Tian, "Dynamic robot path planning using an enhanced simulated annealing approach," *Applied Mathematics and Computation*, vol. 222, pp. 420–437, 2013.

[32] J. F. Lian, W. T. Yu, K. Xiao, and W. R. Liu, "Cubic spline interpolation-based robot path planning using a chaotic adaptive particle swarm optimization algorithm," *Mathematical Problems in Engineering*, vol. 2020, Article ID 1849240, 20 pages, 2020.

[33] X. H. Liu, D. G. Zhang, J. Zhang, T. Zhang, and H. L. Zhu, "A path planning method based on the particle swarm optimization trained fuzzy neural network algorithm," *Cluster Computing*, vol. 24, no. 3, pp. 1901–1915, 2021.

[34] Z. S. Wang, H. W. Ding, J. J. Yang et al., "Orthogonal pinhole-imaging-based learning salp swarm algorithm with self-adaptive structure for global optimization," *Frontiers in Bioengineering and Biotechnology*, vol. 10, Article ID 1018895, 2022.

[35] H. W. Ding, X. G. Cao, Z. S. Wang et al., "Velocity clamping-assisted adaptive salp swarm algorithm: balance analysis and case studies," *Mathematical Biosciences and Engineering*, vol. 19, no. 8, pp. 7756–7804, 2022.

[36] M. Z. Peng and W. Meng, "Cooperative obstacle avoidance for multiple UAVs using spline_VO method," *Sensors*, vol. 22, no. 5, p. 1947, 2022.