Hindawi
Complexity
Volume 2021, Article ID 8993173, 13 pages
https://doi.org/10.1155/2021/8993173

WILEY | Hindawi

Research Article

A Novel Particle Swarm Optimization Algorithm Based on Reinforcement Learning Mechanism for AUV Path Planning

Haoqian Huang and Chao Jin

College of Energy and Electrical Engineering, Hohai University, Nanjing 211100, China

Correspondence should be addressed to Haoqian Huang; qhbhhq@163.com

Received 28 June 2021; Revised 1 September 2021; Accepted 27 November 2021; Published 26 December 2021

Academic Editor: Hassan Zargarzadeh

In order to solve the problems of rapid path planning and effective obstacle avoidance for autonomous underwater vehicle (AUV) in 2D underwater environment, this paper proposes a path planning algorithm based on reinforcement learning mechanism and particle swarm optimization (RMPSO). A feedback mechanism of reinforcement learning is embedded into the particle swarm optimization (PSO) algorithm by using the proposed RMPSO to improve the convergence speed and adaptive ability of the PSO. Then, the RMPSO integrates the velocity synthesis method with the Bezier curve to eliminate the influence of ocean currents and save energy for AUV. Finally, the path is developed rapidly and obstacles are avoided effectively by using the RMPSO. Simulation and experiment results show the superiority of the proposed method compared with traditional methods.

1. Introduction

Autonomous underwater vehicle (AUV) has now become a hotspot area in recent years, especially the multi-AUV system due to the high parallelism, robustness, and collaboration of high efficiency [1–6]. The path planning and obstacle avoidance of AUV are the fundamental issues in the path planning research field. Therefore, a path planning algorithm is applied to plan an effective path and to avoid obstacles autonomously in complex underwater environment, and it is still an open challenging issue in AUV [7–9].

There are a number of achievements reported about studies of path planning in underwater environments. The AUV path planning algorithm can be divided into local path planning and global path planning according to the environment information. Local path planning, like rolling window algorithm [10] and artificial potential field method [11, 12], aims to avoid the obstacles quickly when a robot's sensor detects the surrounding obstacles. Global path planning, such as A* algorithm [13, 14], fast search algorithm [15], Dijkstra algorithm [16], probabilistic roadmaps [17], estimation of distribution algorithm (EDA) based approach [18], is a kind of path planning method used when the map environment is known.

Particle swarm optimization (PSO) is a well-known evolutionary algorithm that is generally regarded as an effective optimization method to solve path planning problems [19]. Roberge [20], Yan [21], and Kang [22] have adopted the PSO algorithm for obstacle avoidance and trajectory optimization of AUV path planning. Simulation experiments show that the PSO algorithm has excellent robustness and a fast convergence speed. However, the local optimal solution is obtained for the PSO algorithm. It is fast for the convergence speed of the algorithm in the initial stage of the search, and in the later stage of the search, the convergence speed of the algorithm is slow. Thus, a few improvements of this method have emerged. A cubic spline optimization algorithm based on an improved PSO algorithm is proposed to address multi-AUV path search problems. The path search is regarded as the parameter optimization of a particular cubic spline. In this manner, the convergence of the algorithm can be significantly improved [23, 24]. Compared with the standard PSO algorithm, the new hybrid PSO-LPM algorithm [25] for an AUV can find better trajectories and successfully implement real-time avoidance of static obstacles and moving obstacles. However, the effect of ocean current is not considered in these algorithms. Zeng [26] employed the quantum particle swarm

optimization (QPSO) algorithm for the path search of AUVs. Although the navigation of AUV in ocean current is studied, the problem of convergence rate of the algorithm still exists. The problem of convergence rate is that the convergence rate is not considered for the exiting PSO algorithm while the ocean current influence on AUV is studied. However, for the proposed RMPSO in the manuscript, not only is the ocean current influence on the AUV convergence speed considered, but also the convergence speed is improved substantially, so the path planning is acquired quickly.

This paper proposes an improved particle swarm optimization algorithm based on the reinforcement mechanism (RMPSO) integrating reinforcement learning mechanism with particle swarm optimization algorithm. The inertia weight ω can determine the speed of convergence of the particle swarm algorithm. The value of ω is reduced linearly with time for the traditional PSO which cannot adjust the value of ω adaptively. The introduction of the reinforcement mechanism can adjust the value of ω adaptively and make the algorithm converge faster. The contributions and novelties of the proposed algorithm can be summarized as follows.

(1) In order to make the model of environment more accurate and reduce the amount of calculation, a convex polygon obstacle is built firstly, and then the area of obstacle is extended to form a dangerous area. The MAKLINK phase-free network diagram is constructed by using the dangerous area.

(2) In order to save energy consumption of AUV, an ocean current evaluation function is proposed and it is combined with velocity synthesis method. According to ocean current evaluation function, the new fitness function is introduced in the proposed RMPSO to implement obstacle avoidance and estimate the statement of particles.

(3) By combining reinforcement learning mechanism with PSO, the parameter of particle swarm optimization can be adjusted adaptively according to the surrounding environment. The simulation results demonstrate the effectiveness of the proposed method compared with traditional methods.

The rest of the paper is organized as follows. Section 2 introduces the AUV underwater path planning problem and the thinking of applying PSO to the MAKLINK diagram. Section 3 presents the RMPSO algorithm for path planning and energy saving. Simulation results and experimental tests are provided in Section 4. Finally, the conclusions and future work are given in Section 5.

2. Problem Statement and Preliminaries

Our mission is to conduct a detailed investigation for the suspicious target point through the AUV in the underwater environment. The underwater environment is harsh and complex with a number of obstacles and ocean currents. The algorithm of path planning and obstacle avoidance, after the AUV receives the information of targets, is discussed in detail. The underwater environment is modeled as shown in Figure 1. Because the sizes of AUV and target appear very small in the vast ocean, both of them are regarded as mass points and their shapes are neglected. In Figure 1, the black area indicates obstacles and the red pentagram represents the suspicious target. There are some arrow lines indicating the ocean current. In this environment, the ocean current velocity is generally less than 1 knot, which is in the region where the AUV can be balanced to reach the target directly. A constant ocean current model is introduced to represent the ocean current effect on AUV.

In order to guarantee that the AUV accomplishes the task and gets access to the target successfully, it is necessary to ensure that obstacles can be avoided and the influence of ocean current on AUV can be eliminated. Therefore, the RMPSO algorithm is proposed, and it can deal with ocean current and plan a safe trajectory rapidly.

2.1. Modeling of Underwater Environment. Since the obstacles in the marine environment are basically irregular concave convex polygons, the MAKLINK graph theory for environmental model is applied to model the obstacles more accurately.

2.1.1. Modeling of Obstacles. To avoid obstacles effectively when the AUV works, a convex polygon obstacle model is established by using the Graham algorithm [27] according to the boundary information of the obstacle. The dotted line is shown in Figure 2(a). The area of obstacle is extended, and this extended distance is $d_{\min}$ m. The solid line range is shown in Figure 2(a), the space between the dotted line and the solid line is named Danger Zone. Lastly, the distance between AUV and obstacle meets $d \geq d_{\min}$.

2.1.2. Modeling of Two-Dimensional Map. Assume that there are m obstacles in the map and the ith obstacle O_i has n vertices. The mathematical model of obstacle O_i can be expressed as follows:

$$O_i = \left[(x_{i1}, y_{i1}), (x_{i2}, y_{i2}), \ldots, \left(x_{ij}, y_{ij}\right)\right]. \quad (1)$$

m, n are the numbers of obstacles and vertices, respectively; (x_{ij}, y_{ij}) means that the ith obstacle is the jth vertex coordinate, $i = 1, 2, 3, \ldots, m$ and $j = 1, 2, 3, \ldots, n$.

The working environment of AUV can be modeled as

$$E = [B, O_i, \ldots, O_M], \quad (2)$$

where B represents the environmental boundary.

A 2D map with AUV is shown in Figure 2(b), and there are four polygon obstacles, so m is set as 4, where yellow and black area represent Danger Zone and obstacles, respectively. The dotted line in Figure 2(b) represents the phase-free network graph based on MAKLINK graph theory. The MAKLINK graph algorithm is introduced, shown as Algorithm 1.

2.2. Thinking in PSO. The PSO, as a kind of evolutionary algorithm, imitates birds' foraging behavior to solve the optimization problem [28]. Particles optimize themselves

(1) **Initialize**: connecting the vertices of convex polymorphism to each other and forming an aggregate L
(2) **for** find the shortest line k in the aggregate L
(3) **repeat**:
(4) **if** k passes through the obstacles **then**
(5) select the next line in the aggregate L
(6) **else**
(7) **if** the two outer angles formed by the line k and the corresponding convex polygon boundary are more than 180° **then**
(8) **if** the angle between k and other candidate lines exceeds 180° **then**
(9) select the next line in the aggregate L
(10) **else** delete other lines of the vertex and keep the shortest line k
(11) **else** select the next line in the aggregate L
(12) **Until**: All vertices are traversed and the phase-free network graph is constructed

Algorithm 1: MAKLINK graph algorithm.

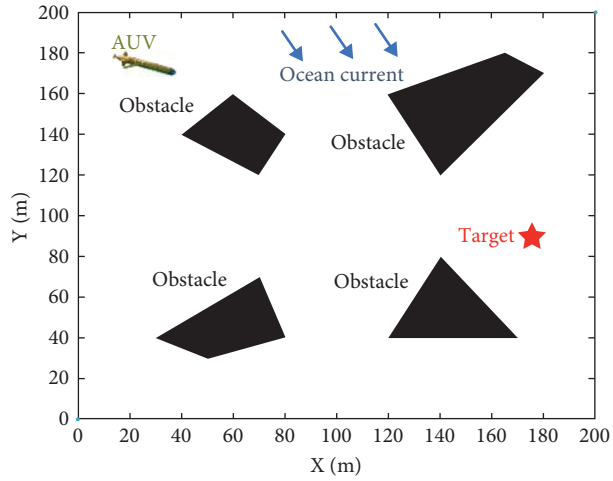


Figure 1: Complex marine environment.

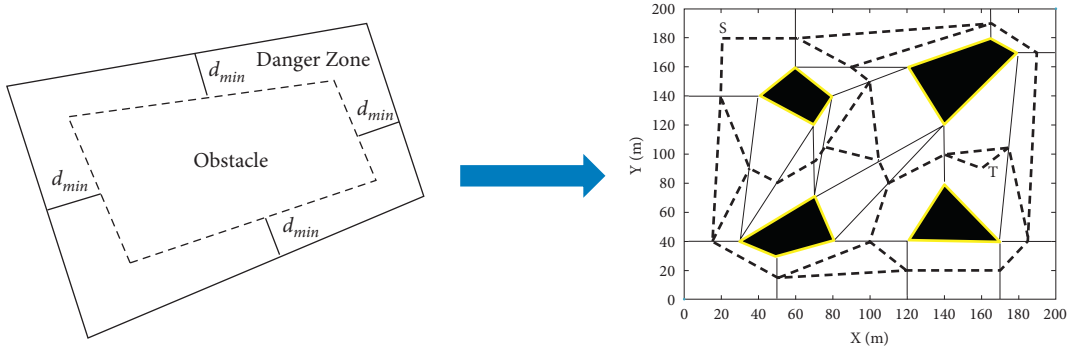


Figure 2: Modeling of underwater environment. (a) Modeling of obstacle. (b) Modeling of 2D map.

continuously by tracking two extremums in each iteration. The first extremum is the optimal solution found by each particle itself, which is called individual extreme value. The second extremum is the optimal solution found by all particles at present, which is called global extremum value. The location of the particle will be close to the two peak positions, namely, individual extreme value and global extremum value, and the optimal position will be searched.

Assume that there is a d-dimensional search space, and there are two kinds of attributes for each particle, namely, current position X_{id} and velocity V_{id}, X_{id} and V_{id} are given:

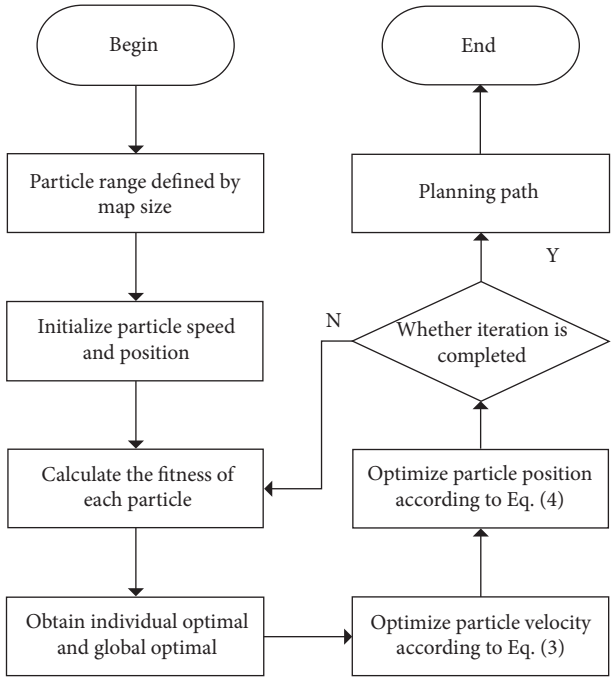


Figure 3: PSO algorithm applied in path planning.

$$V_{\mathrm{id}}(t+1) = \omega V_{\mathrm{id}}(t) + C_1 r_1(t)\left(P_{\mathrm{id}}(t) - X_{\mathrm{id}}(t)\right) + C_2 r_2(t)\left(P_{g\,d}(t) - X_{\mathrm{id}}(t)\right), \tag{3}$$

$$X_{\mathrm{id}}(t+1) = X_{\mathrm{id}}(t) + V_{\mathrm{id}}(t+1), \tag{4}$$

where V_{id} is similar to P_{id}, X_{id}, and $P_{g\,d}$; V_{id} is the dth dimension of the ith particle's velocity. It is limited to the interval $[-V_{\max}, +V_{\max}]$ to avoid the explosion of the particles. P_{id} and $P_{g\,d}$ represent the individual extreme value and the global extremum value, respectively. Coefficients r_1 and r_2 are two pseudorandom scalar values. The superscript in (3) denotes the tth iteration. The acceleration coefficients C_1 and C_2 are 2 for almost all applications. The factor ω is the inertial weight, and this inertial weight plays the role of balancing the global search (large inertial weight) with the local search (small inertial weight). The performance of the algorithm is improved by adaptation parameter during the optimization process significantly. The flow chart of the PSO algorithm applied in path planning is shown in Figure 3.

3. Main Algorithm

In order to further explain how the PSO algorithm is improved into RMPSO, mathematical models and formulas with some discussions are given.

3.1. Fitness Function. The evaluation of particles position is determined by fitness function; then, the PSO algorithm optimizes the position of particles according to fitness value of particles. The traditional fitness function is given:

$$F(i) = d(i), \tag{5}$$

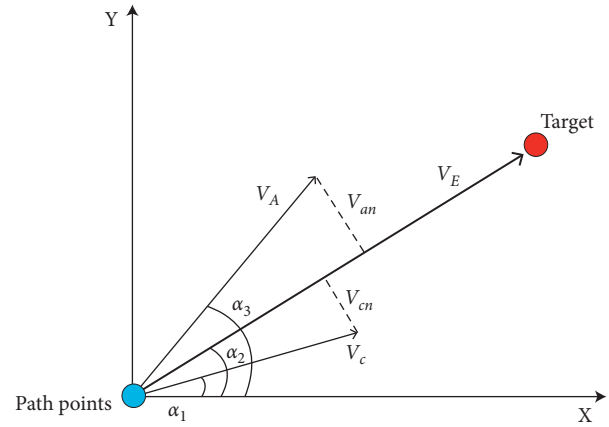


Figure 4: Velocity synthesis method.

where

$$d(i) = \sum_{d=1}^{D+1} \sqrt{\left(x_{\mathrm{id}} - x_{\mathrm{id}-1}\right)^2 + \left(y_{\mathrm{id}} - y_{\mathrm{id}-1}\right)^2}. \tag{6}$$

Equation (5) gives the Euclid distance of the particle generation path, where x_{id} and y_{id} represent the ith particle in the dth dimension.

(1) Penalty function: The penalty function $P(i)$ is introduced in (7) to make sure that AUV is not close to obstacles and is defined as follows:

$$P(i) = \begin{cases} 0, & \text{paths are not blocked by obstacles,} \\ E, & \text{paths are blocked by obstacles,} \end{cases} \tag{7}$$

where E is a positive constant and it is far greater than the fitness value of other particles. When obstacles are not traversed by a path formed by particles, $P(i) = E$. Otherwise, $P(i) = 0$. The fitness function is updated:

$$F(i) = d(i) \cdot 0.5 + P(i) \cdot 100, \tag{8}$$

where 0.5 and 100 can be obtained by adjusting parameters of program. Equation (8) effectively prevents the particles from producing obstacles when the PSO algorithm iterates.

(2) Ocean current evaluation function: After obstacles can be avoided by AUV successfully, in the next step, the ocean current evaluation function combined with the velocity synthesis approach [29] is designed to estimate the influence of ocean current on AUV.

As shown in Figure 4, a moving coordinate system is established. A velocity vector V_c in the moving coordinate system represents the ocean current speed affecting the AUV. V_E and V_A represent the vectors of synthesis and AUV's speed. The angles between $V_A/V_E/V_C$ and X-axis are defined as α_3, α_2, α_1, respectively.

Where V_{cn} is the vertical component of ocean current on V_E, and Van is the vertical component of AUV on V_E. When V_{cn} and V_{an} are opposite, the side effects of ocean current

will be canceled, namely, $V_A \sin(\alpha_3 - \alpha_2) = V_C \sin(\alpha_2 - \alpha_1)$. When the ocean current speed is known, the ocean current can be used for path planning by adjusting the speed of AUV and the direction of each road section. The desired speed V_E and path angle α_3 of AUV can be figured out:

$$V_A = \frac{V_C \cdot \sin(\alpha_2 - \alpha_1)}{\sin(\alpha_3 - \alpha_2)},$$
$$\alpha_3 = \arcsin\left(\frac{V_C \sin(\alpha_2 - \alpha_1)}{V_A}\right) + \alpha_2. \tag{9}$$

Assume that the speed of AUV is known, then the PSO generates the absolute value of the difference between the expected angle and the actual angle of each path, and the absolute value is $|\alpha_3(\mathrm{id}) - \alpha(\mathrm{id})|$. Considering that each path length will affect the energy consumption of AUV, the angle difference and path length are considered comprehensively, and the ocean current evaluation function of each path segment is $|\alpha_3(\mathrm{id}) - \alpha(\mathrm{id})| \cdot 0.3 + X(\mathrm{id}) \cdot 0.1$. The path ocean current evaluation function generated by each particle can be defined as follows:

$$\alpha(i) = \sum_{d=1}^{D+1} \left(\left|\alpha_3(\mathrm{id}) - \alpha(\mathrm{id})\right| \cdot 0.3 + X(\mathrm{id}) \cdot 0.1\right), \tag{10}$$

where $\alpha_3(\mathrm{id})$, $\alpha(\mathrm{id})$ and $X(\mathrm{id})$ are the expected angle, actual angle, and path length of the dth segment path generated by the ith particle, respectively. The dimension of particles is D, which indicates the number of path points generated by each particle in the map. The total number of path points including starting point and ending point is $d+2$, and the number of path segments generated is $d+1$. When the ocean current is taken into consideration, the fitness function is updated:

$$F(i) = d(i) \cdot 0.5 + C \cdot 100 + \alpha(i) \cdot 0.216. \tag{11}$$

According to (11), not only are obstacles effectively avoided, but also the influence on AUV of ocean current is taken into account.

3.2. PSO Weight Updating Function. The traditional PSO algorithm has fast convergence speed in the early stage and slow convergence speed in the later stage, and it cannot adjust ω adaptively. The increase of ω appropriately can improve the global search ability of the algorithm, and the decrease of ω improves the local search ability. Therefore, a reinforcement learning mechanism [30, 31] is integrated with PSO to overcome the shortcomings of the PSO algorithm and plan the optimal path rapidly.

(1) Reinforcement learning mechanism: As shown in Figure 5, $F(i)$ is the adaptation value function that can calculate the fitness value of each particle according to the environment information, and $F(i)$ can also calculate the global optimal solution and local optimal solution of the particle population. Then, the inertia weights $\omega(i)$ of the particles are adjusted adaptively according to $F(i)$. When the global optimal solution, local optimal solution, and inertia weights $\omega(i)$ are obtained, the velocities and positions of particles are updated according to (1) and (2). These steps mentioned above are repeated to optimize the inertia weights $\omega(i)$. The adaptation value function $F(i)$ is defined as (11) and $\omega(i)$ is defined as (13).

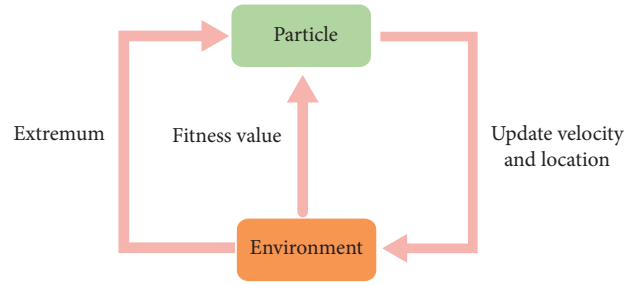


Figure 5: Application of reinforcement mechanism to PSO.

The inertia weight update formula of the traditional PSO algorithm is shown as (12). T is the number of current iterations, and the inertia weight $\omega(i)$ decreases linearly with the increase of T. Compared with the inertia weight formula of the traditional PSO, the inertia weight for RMPSO algorithm can be adjusted adaptively according to the surrounding environment, so the convergence rate of RMPSO increases.

$$\omega(i) = \frac{\omega(i) - (\omega(i) - 0.3)}{T}. \tag{12}$$

(2) PSO weight updating function: Based on the above-mentioned reinforcement mechanism, (13) is introduced into PSO weight updating function to improve the convergence speed and accuracy of the PSO algorithm. The following equation is defined:

$$\omega(i) = 0.3 * \exp((F(i) - 100) * 0.03 - 0.63), \tag{13}$$

where 0.3, 0.03, and 0.63 can be obtained by adjusting parameters of program; $\omega(i)$ represents the weight of the ith particle, and this weight has a positive relationship with $F(i)$ in (13). It is necessary to improve the global optimization ability when the fitness of particles becomes larger. At the same time, the particle's weight will be increased according to (13), which can improve the capability of global optimization. When the particle fitness value becomes smaller, leading smaller $\omega(i)$, the local optimization ability becomes stronger.

The velocities V_{id} of particles in the PSO algorithm are updated as

$$V_{\mathrm{id}}(t+1) = \omega_i(t) V_{\mathrm{id}}(t) + C_1 r_1(t)\left(P_{\mathrm{id}}(t) - X_{(t)}(t)\right) + C_2 r_2(t)\left(P_{\mathrm{gd}}(t) - X_{\mathrm{id}}(t)\right). \tag{14}$$

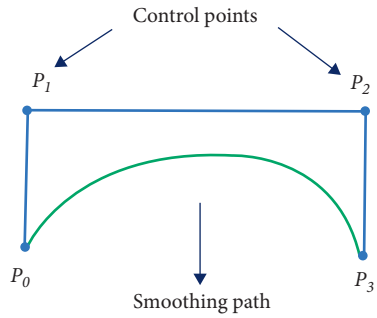


Figure 6: Bezier curve.

3.3. Bezier Curve. A Bezier curve of degree n can be represented as

$$R(t) = \sum_{i=0}^{n} PB_{i,n}(t), \quad t \in [0,1], \tag{15}$$

where t indicates the normalized time variable; $P_i = [x_i, y_i]^T$ is the coordinate vector of the ith control point with x_i and y_i in the X and Y coordinates, respectively. $B_{i,n}$ denotes the Bernstein basis polynomials, representing the base function in the expression of a Bezier curve and given by

$$\begin{aligned} B_{i,n}(t) &= C_n^i t^i (1-t)^{n-i} \\ &= \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i}, \quad i = 0, 1, \ldots, n. \end{aligned} \tag{16}$$

From (15) and (16), the parameter equation for each control point can be generated as follows:

$$P(t) = P_0(1-t)^3 + 3P_1(1-t)^2 t + 3P_2(1-t)t^2 + P_3 t^3, \tag{17}$$

where t is the range of [0, 1]. The Bezier curve is invariance under translation and rotation, and this is called geometry invariance property. In addition, the Bezier curve starts at the starting point ($t = 0$) and stops at the ending ($t = 1$). In other words, $P_0 = R(0)$ and $P_n = R(1)$. The Bezier curve has some control points which form a control polygon as shown in Figure 6.

By sampling four equidistance points in the polyline generated by the RMPSO algorithm, these four points are P0, P1, P2, P3. A series of dense points $P(t)$ are obtained from (17), and then the smooth curve is obtained by connecting these points.

3.4. The Whole Procedure of the Proposed Algorithm. The flow of the improved PSO algorithm is shown in Algorithm 2. The whole path planning algorithm is a loop procedure that is repeated until an AUV gets close to the target. The AUV moves to its corresponding position, and ocean currents and obstacles are taken into consideration at the same time. The procedure is performed through iterations to calculate the optimal solution of path cost.

In Figure 7, the whole path planning process includes the model of underwater environment, path planning, and path optimization. In order to express the ocean environment information integrally, the Graham algorithm is presented to construct polygon obstacle model, and a 2D map is built based on MAKLINK graph theory.

Firstly, a 2D map is established by MAKLINK theory and the Graham algorithm. Then, a suboptimal path is planned based on the Dijkstra algorithm. When the RMPSO is used to continuously optimize the suboptimal path through iteration, the obstacles will be avoided effectively based on MAKLINK phase-free network graph. Finally, in order to further optimize the path, the Bezier curve is used to smooth the optimized path.

4. Experiment Results and Analysis

The effectiveness of the RMPSO algorithm is demonstrated, and simulations are carried out with different ocean environment. At the same time, the path lengths of ant colony algorithm (ACO), PSO, Dijkstra, and RMPSO in the same map are compared. The algorithm convergence rates of the ACO, PSO, and RMPSO are also compared. The value of $d_{\min}$ is set as 1.875 m, which is three times the AUV's length. The number of iterations and the population size are set as 150 and 80; the dimension and the maximum particle speed are set as 8 and 3; the learning factors C1 and C2 are set as 2 and 2, respectively.

4.1. Validity Simulations. Figure 8 illustrates the environment model and the initial path planning route. The underwater workspace is designed as 200m × 200 m. The starting position of AUV is (20, 180), and the target position is (160, 90), which are represented by "Start" and "Target," respectively. The yellow area and the black area are the Danger Zone and obstacles, respectively. The initial path is formed using the Dijkstra algorithm and is shown in green in Figure 8. The nodes through which the route passes are S, P1, P2, P3, P4, P5, P12, P12, P11, T, respectively. It can be clearly seen that the green path along the dotted line is not the optimal path and its length is 252.26 m.

In Figure 8, the Dijkstra algorithm determines the node, and it also determines the real line where the node is located, so each segment of the path can move in the real line and the path line will not intersect with obstacles when moving. The positions of P1, P2, P3, P4, P5, P12, P13, P11 are adjusted continuously to determine the shortest path of each segment, and finally the optimal path is obtained.

4.1.1. Result in Stationary Environment. Based on the Dijkstra algorithm, the final path optimization is performed by employing the algorithm RMPSO. The red route in Figure 9(a) is the final path optimization path result. The AUV get close to the edge of the Danger Zone to avoid the obstacle, and the path length is only 176.2 m, which is shorter than the length of initial path planning. Figures 9(b) and 9(c) show the convergence trend of the RMPSO algorithm which has converged to the optimal value in the 35th iteration of the algorithm.

(1) **Initialize**: the particles' positions, velocity (V), the number of particles (P), maximum number of iterations (N), the global optimum (gbest), the individual optimum (pbest (i))
(2) **for** maximum number of iterations (N)
(3) **for** all particles (i)
(4) Update the position and V according to (14)
(5) Calculate the fitness value of particles F (i) according to (11)
(6) **if** (F (i) < pbest (i)) **then**
(7) pbest (i) = F (i)
(8) **end if**
(9) **if** (pbest (i) < gbest) **then**
(10) gbest = pbest (i)
(11) **end if**
(12) update ω (i) according to (13)
(13) **end for**
(14) **end for**

Algorithm 2: RMPSO algorithm.

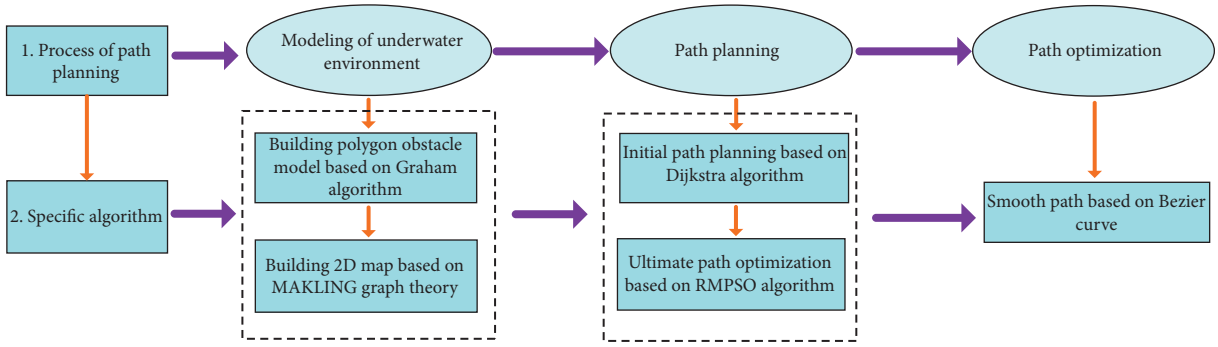


Figure 7: The whole path planning process.

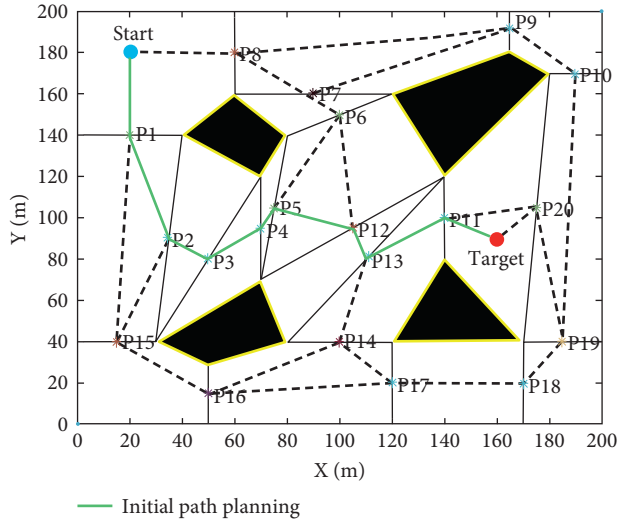


Figure 8: Initial path planning (Dijkstra algorithm).

4.1.2. Result in Ocean Current Environment. The serious influence of ocean currents in the complex underwater environment may lead to AUV mission failure. It is assumed that the AUV speed is 1.5 m/s, the current speed is 0.3 m/s, and the direction is -75°. In Figure 10(a), the red solid line is the optimized path of the RMPSO algorithm under the influence of ocean currents. When the influence of ocean currents is considered comprehensively, the position of the path point is changed in each section of the path, and a tortuous path is formed.

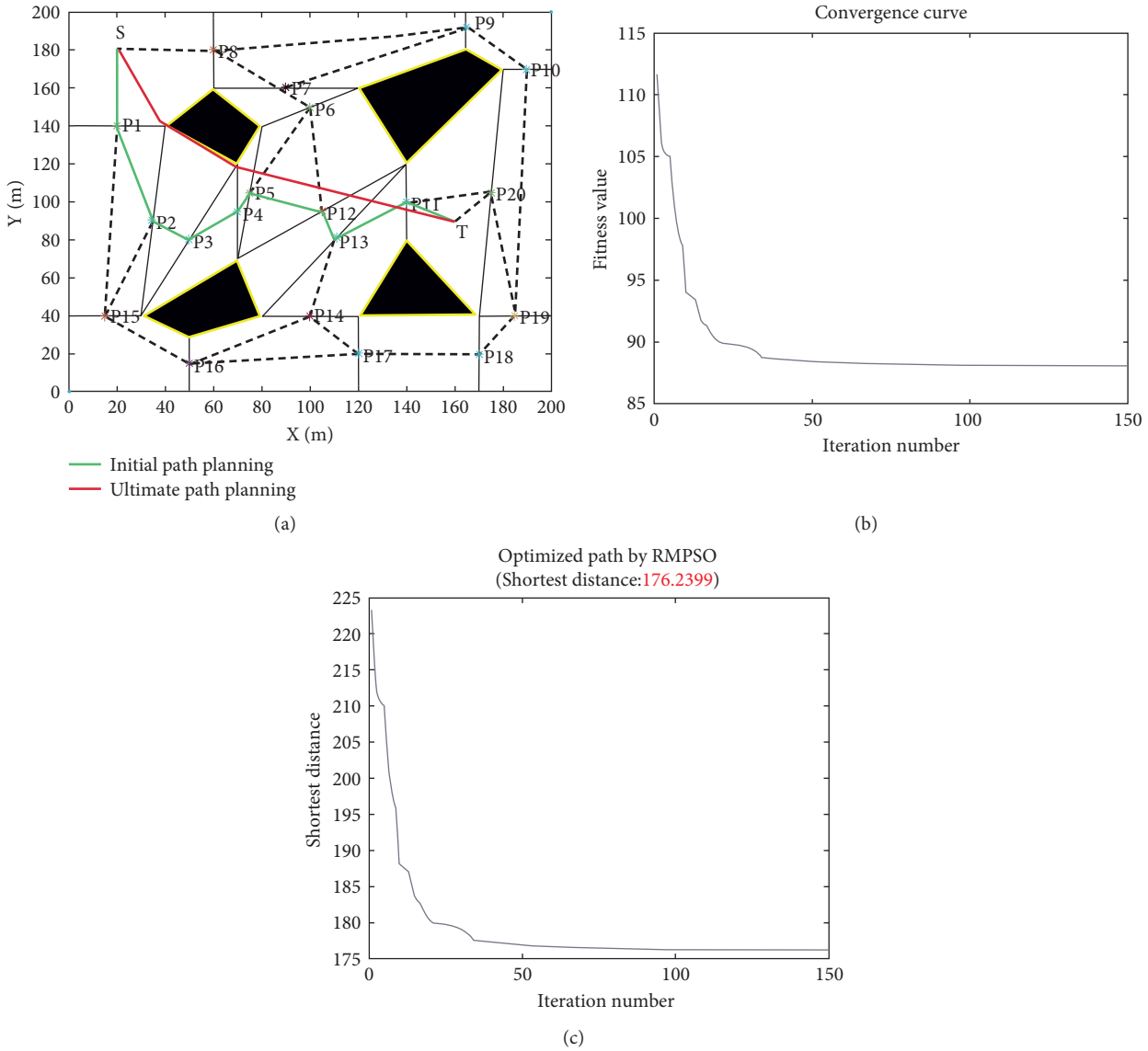


Figure 9: Optimized path through RMPSO. (a) Ultimate path planning (RMPSO). (b) Fitness convergence curve (RMPSO). (c) Path distance convergence curve (RMPSO).

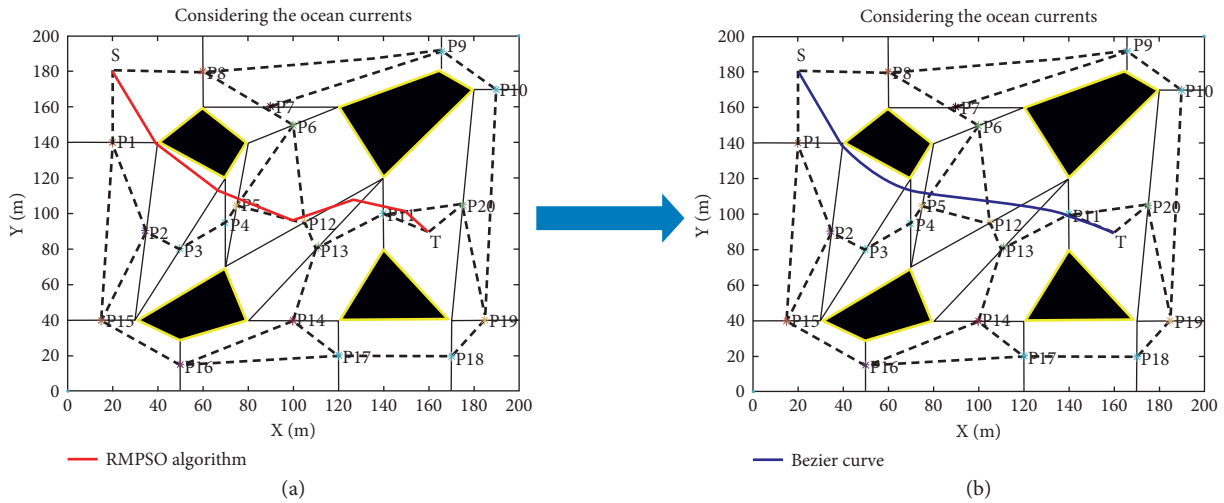


Figure 10: The influence of ocean currents is taken into account during path planning. (a) Path optimized by the RMPSO algorithm. (b) Path smoothed by the Bezier curve.

In order to save AUV energy, the Bezier curve smoothing path is used to reduce the number of turning points. Shown as the solid blue line path in Figure 10(b), the smoothed path has no obvious turning points.

4.2. Comparison Studies. In order to more clearly compare the differences between the ACO algorithm [32], Dijkstra algorithm [33], PSO algorithm, and RMPSO algorithm in terms of convergence speed and total path length, the ocean current influence is not considered for the above algorithms based on the environment.

4.2.1. Result of the ACO Algorithm. Path No. 4 in Figure 11 shows the optimization results of the ant colony algorithm, and it can be seen that the path is not the shortest path and that the total path length is 188.86 m.

4.2.2. Result of the PSO Algorithm. Path No. 1 in Figure 11 shows the optimization results of the particle swarm algorithm, and it can be seen that the path is almost the same as those of the RMPSO algorithm, with a total path length of 179.38 m.

4.2.3. Comparison and Analysis. Figure 12 shows the comparison of the convergence speed of three optimization algorithms, namely, the ant colony algorithm, PSO algorithm, and RMPSO algorithm. It can be seen that the red curve representing RMPSO algorithm converges much faster than the other two optimization algorithms for the same number of convergences. Table 1 shows the specific values of convergence of the three optimization algorithms. Because the ant colony algorithm relies on pheromone concentration to look for paths and there exists a lot of instability, so it starts to converge only after the 56th iteration. The inertia weight of the PSO algorithm decreases linearly with the number of iterations and cannot take into account the current particles' condition, so it starts to converge only in the 45th iteration. The RMPSO can adjust the inertia weights according to the surrounding environment, and the algorithm starts to converge in the 22nd iteration, which is much better than the other two algorithms.

With the same number of iterations, the RMPSO algorithm converges to the optimal value in the 22nd iteration, while the ant colony and particle swarm algorithm converge to the optimal value only in 45th and 56th iterations, respectively.

Table 2 shows the comparison of the total path lengths of the four optimization algorithms. The Dijkstra algorithm has the longest path which is 252.26 m, while the final path length of the RMPSO algorithm converges to only 176.20 m. Obviously, the RMPSO algorithm is not only shorter than the standard particle swarm algorithm in terms of path length, but also much faster than the other two optimization algorithms in terms of convergence speed.

4.2.4. Comparison Based on Raster Maps. Raster maps have the advantage of being able to simply model and easily validate new algorithms. Therefore, we apply RMPSO to raster maps and compare it with the PSO and A∗ algorithms.

Figure 13 shows the paths planned by three algorithms, with black squares as obstacles, PSO algorithm for line 1, RMPSO algorithm for line 2, and A* algorithm for line 3. Through Table 3, we can learn that the path length of RMPSO planning is better than those of the PSO and A* algorithms, where RMPSO has 3.75 m more than A*. Because the A* algorithm relies on the size of the adaptation value of each raster in the raster map, there are many corners in the planned path due to the calculation of each raster, so the path length of the A* algorithm is more than those of both PSO and RMPSO.

4.3. Real Experiment and Results. To further test the performance of the proposed RMPSO algorithm in a real environment, the underwater experiment platform is established, shown as Figure 14. The underwater experiment platform is mainly composed of underwater vehicle [34]. The size of the underwater vehicle is shown in Figure 14, with a length of 0.625 m, a width of 0.457 m, and a height of 0.326 m; 8 thrusters are used for driving to achieve free cornering. The underwater vehicle has a maximum power of 1500 W, weighing 20 Kg, and can be equipped with cameras, DVL, and other sensors. The experiment was conducted on a lake with an area of about 3000 square meters.

The underwater vehicle platform is composed of a control host CPU model i5, 8G RAM, 120G SSD with Windows 10 operating system. It is also equipped with a 15″ high brightness LED screen, which can display real-time underwater images, attitude, underwater vehicle depth, temperature, and other information. The ground control platform controls the vehicle operation through cables. The experiment was conducted on the lake surface to verify the algorithm to facilitate observation of the experiment. The real-time position of the underwater vehicle is obtained through the control platform and aerial photography, and then the trajectory of the underwater vehicle is plotted.

As shown in Figure 15, the convex polygonal obstacle endpoints are created by floating balls, and then the MKALINK 2D environment model is completed on the computer. The blue and green pentagons are the starting and terminal points, respectively. The black polygon is the obstacle, and the yellow area is the Danger Zone.

The vehicle speed is set as 1.5 m/s in the experiment. In Figures 16, Figure 16(a) shows the trajectory of the optimized algorithm-controlled robot operation. The yellow line is the path planned by the ant colony algorithm, and the blue line is the path planned by the RMPSO algorithm. It can be seen that although the ant colony algorithm effectively avoids obstacles, the path length is longer than that of the RMPSO algorithm. Figure 16(b) shows the planning path of the RMPSO algorithm considering the case with the influence of ocean currents, and the red curve is the path map after Bezier curve optimization. In the experimental process, the control board can largely reduce the frequent control of the robot through this path and

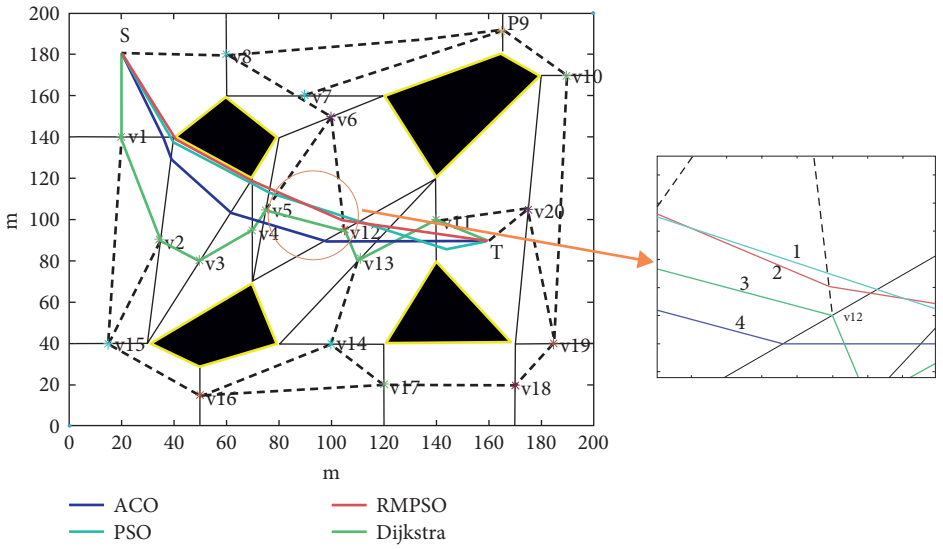


Figure 11: Optimization paths.

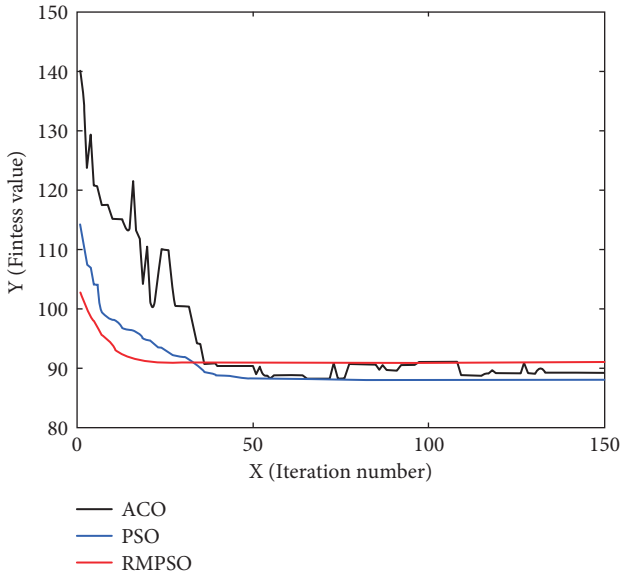


Figure 12: Comparison of convergence speed of the three iterative algorithms.

Table 1: Comparison of convergence speed of the optimization algorithms.

Optimization algorithm	Iteration number/Total number	Convergence value	Convergence minimum
ACO	56/150	88.81	88.15
PSO	45/150	88.61	88.04
RMPSO	22/150	91.51	91.00

Table 2: Comparison of path distances.

Path planning algorithm	Path distance (m)
Dijkstra	252.26
ACO	188.86
PSO	179.83
RMPSO	176.20

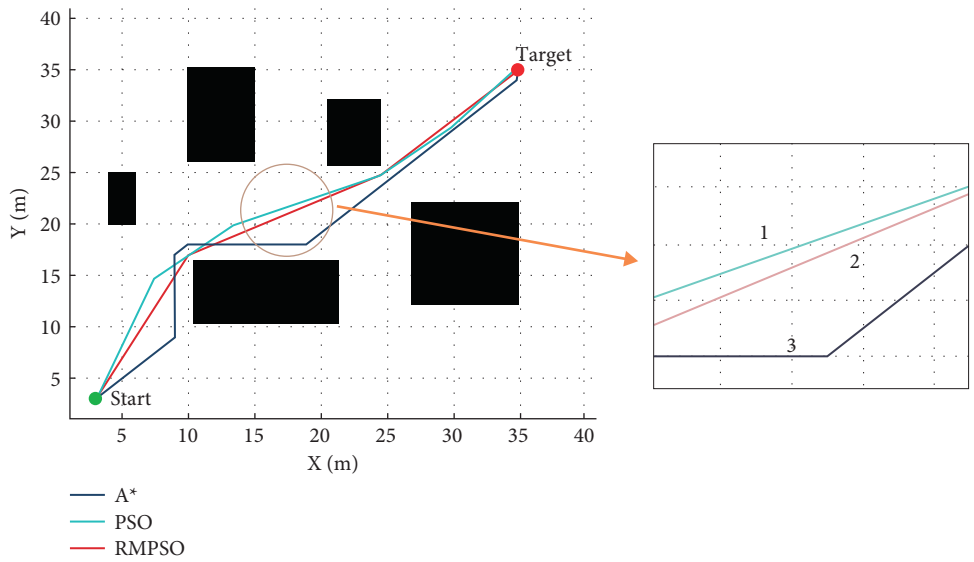


Figure 13: Trajectory comparisons for different algorithms based on raster maps.

Table 3: Comparison of path distances in raster maps.

Path planning algorithm	Path distance (m)
A*	50.53
PSO	47.29
RMPSO	46.78

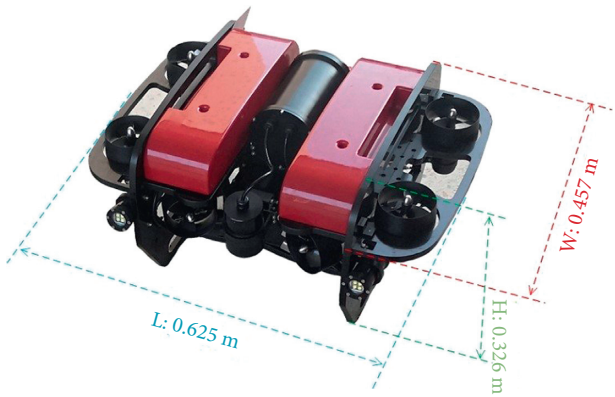


Figure 14: Underwater experiment platform.

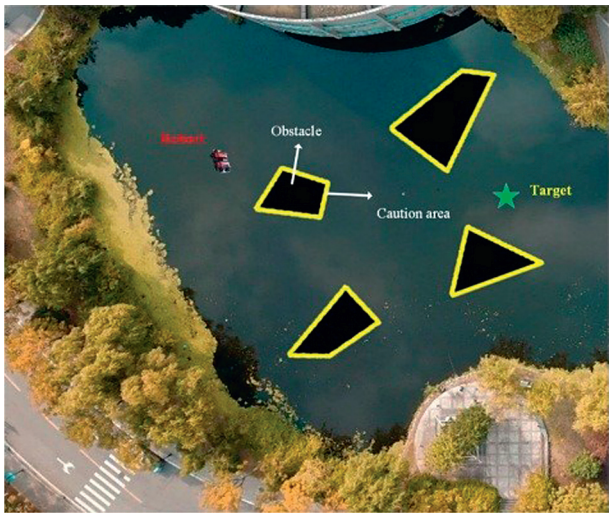
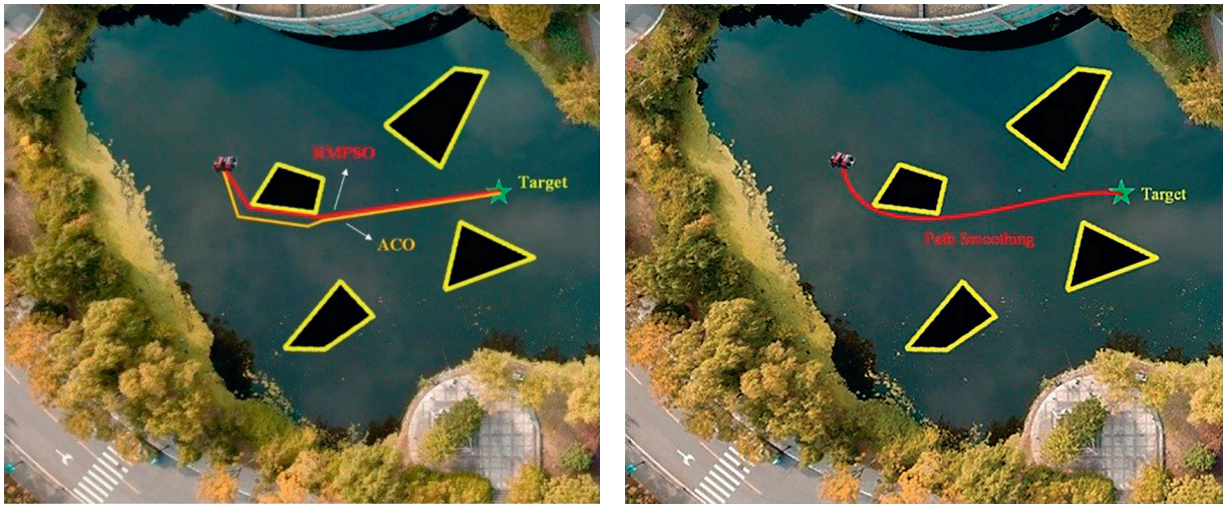


Figure 15: Experimental environment on the lake.



(a) (b)

Figure 16: Path planning in the lake. (a) Comparison of the ACO algorithm and RMPSO algorithm. (b) Bezier curve smoothing path.

also effectively avoid the obstacles. The stability of underwater vehicle is improved greatly by using the proposed RMPSO to obtain the shortest and smoothed trajectory, resulting in saving energy efficiently.

5. Conclusion and Future Work

We propose an improved particle swarm algorithm based on reinforcement mechanism combined with the ocean current model to solve the path planning problem and energy consumption problem of AUV. Simulation results demonstrate that the proposed RMPSO path planning algorithm can effectively avoid obstacles in the MAKLINK undirected network graph. By comparing various algorithms with RMPSO, the convergence speed of the proposed algorithm is found to be much faster, but there are a few disadvantages for the stability. In future research, the RMPSO algorithm will have practical applications in the fields of underwater search, rescue, and investigation. The statistical analysis will be used to prove the superiority of RMPSO, improve the stability of the algorithm, and use it to solve underwater 3D path planning problems [35].

Data Availability

The research data used to support the findings of this study are included within the article. The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Central Universities (B200202163), the National Natural Science Foundation of China (61703098), the Natural Science Foundation of Jiangsu Province (BK20160699), and the Fujian Provincial Key Laboratory of Coast and Island Management Technology Study (FJCIMTS2019-03).

References

[1] L. Paull, S. Saeedi, M. Seto, and H. Li, "AUV navigation and localization: a review," *IEEE Journal of Oceanic Engineering*, vol. 39, no. 1, pp. 131–149, 2014.

[2] D. Yang and P. J. Wu, "E-commerce logistics path optimization based on a hybrid genetic algorithm," *Complexity*, vol. 2021, Article ID 5591811, 10 pages, 2021.

[3] H. Huang, X. Chen, B. Zhang, and J. Wang, "High accuracy navigation information estimation for inertial system using the multi-model EKF fusing adams explicit formula applied to underwater gliders," *ISA Transactions*, vol. 66, pp. 414–424, 2017.

[4] H. Q. Huang, R. D. Shi, J. Zhou et al., "Attitude determination method integrating square-root cubature kalman filter with expectation-maximization for inertial navigation system applied to underwater glider," *Review of Scientific Instruments*, vol. 90, no. 9, 2019.

[5] H. Q. Huang, X. Y. Chen, Z. K. Zhou, H. Liu, and C. P. Lv, "Study on INS/DR integration navigation system using EKF/RK4 algorithm for underwater gliders," *Journal of Marine Science and Technology*, vol. 25, no. 1, pp. 84–95, 2017.

[6] W. C. Hou, Q. S. Lou, X. Wu, Y. Zhou, and G. Si, "Multi-objective optimization of large-scale EVs charging path planning and charging pricing strategy for charging station," *Complexity*, vol. 2021, Article ID 8868617, 17 pages, 2021.

[7] B. Wang, Z. Liu, Q. Li, and A. Prorok, "Mobile robot path planning in dynamic environments through globally guided reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6923–6936, 2020.

[8] K. Wang and K. H. Bae, "In-depth learning layout and path optimization of energy service urban distribution sites under e-commerce environment," *Complexity*, vol. 2021, Article ID 6665610, 11 pages, 2021.

[9] A. Belkadi, H. Abaunza, L. Ciarletta, P. Castillo, and D. Theilliol, "Design and implementation of distributed path

planning algorithm for a fleet of UAVs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 6, pp. 2647–2657, 2019.

[10] D. Gonzalez, J. Perez, V. Milanes, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2016.

[11] A. Chatterjee, "Motion planning approach that produces critical point-free configuration space," *Electronics Letters*, vol. 47, no. 19, pp. 1073–1075, 2011.

[12] Q. Yao, Z. Zheng, L. Qi et al., "Path planning method with improved artificial potential field-A reinforcement learning perspective," *IEEE Access*, vol. 8, pp. 135513–135523, 2020.

[13] L. d. S Costa and F. Tonidandel, "DVG+A∗ and rrt path-planners: a comparison in a highly dynamic environment," *Intelligent Robot Systems*, vol. 58, 2021.

[14] Z. Zhang, J. Wu, J. Dai, and C. He, "A novel real-time penetration path planning algorithm for stealth UAV in 3D complex dynamic environment," *IEEE Access*, vol. 8, pp. 122757–122771, 2020.

[15] J. Xu, Y. Chen, X. Yang, and M. Zhang, "Fast marching-based path generating algorithm in anisotropic environment with perturbations," *IEEE Access*, vol. 8, pp. 5256–5263, 2020.

[16] D. C. Yong, Z. X. Yu, J. Ya, and M. Sankaran, "Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment," *Applied Soft Computing*, vol. 12, no. 3, pp. 1231–1237, 2012.

[17] L. Kavraki, P. Svestka, J. C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 4, no. 12, pp. 566–580, 1996.

[18] R. D. Liu, Z. G. Chen, Z. J. Wang, and Z. H. Zhan, "Intelligent path planning for AUVs in dynamic environments: an EDA-based learning fixed height histogram approach," *IEEE Access*, vol. 7, pp. 185433–185446, 2019.

[19] Y. Mu, B. Li, D. An, and Y. Wei, "Three-dimensional route planning based on the beetle swarm optimization algorithm," *IEEE Access*, vol. 7, pp. 117804–117813, 2019.

[20] V. Roberge, M. Tarbouchi, and G. Labonte, "Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 132–141, 2012.

[21] Z. Yan, J. Li, Y. Wu, and G. Zhang, "A real-time path planning algorithm for AUV in unknown underwater environment based on combining PSO and waypoint guidance," *Sensors*, vol. 19, no. 1, pp. 20–34, 2019.

[22] H. I. Kang, B. Lee, and K. Kim, "Path planning algorithm using the particle swarm optimization and the improved Dijkstra algorithm," *IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application*, vol. 2, pp. 1002–1004, 2008.

[23] X. X. Wu, L. B. Guo, and J. Wang, "Mobile robot path planning algorithm based on particle swarm optimization of cubic splines," *Robot*, vol. 31, no. 6, pp. 556–560, 2009.

[24] D. Li, P. Wang, and L. Du, "Path planning technologies for autonomous underwater vehicles-A review," *IEEE Access*, vol. 7, pp. 9745–9768, 2019.

[25] Y. Zhuang, S. Sharma, B. Subudhi, H. Huang, and J. Wan, "Efficient collision-free path planning for autonomous underwater vehicles in dynamic environments with a hybrid optimization algorithm," *Ocean Engineering*, vol. 127, pp. 190–199, 2016.

[26] Z. Zeng, K. Sammut, L. Lian, F. He, A. Lammas, and Y. Tang, "A comparison of optimization techniques for AUV path planning in environments with ocean currents," *Robotics and Autonomous Systems*, vol. 82, pp. 61–72, 2016.

[27] M. Chardon and A. Moukrim, "The coffman--graham algorithm optimally solves UET task systems with overinterval orders," *SIAM Journal on Discrete Mathematics*, vol. 19, no. 1, pp. 109–121, 2005.

[28] P. K. Das and P. K. Jena, "Multi-robot path planning using improved particle swarm optimization algorithm through novel evolutionary operators," *Applied Soft Computing*, vol. 92, Article ID 106312, 2020.

[29] M. Chen and D. Zhu, "A workload balanced algorithm for task assignment and path planning of inhomogeneous autonomous underwater vehicle system," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 11, no. 4, pp. 483–493, 2019.

[30] P. Cui, W. S. Yan, R. X. Cui, and J. H. Yu, "Smooth path planning for robot docking in unknown environment with obstacles," *Complexity*, vol. 2018, Article ID 4359036, 17 pages, 2018.

[31] G. Q. Xia, Z. W. Han, B. Zhao, and X. W. Wang, "Local path planning for unmanned surface vehicle collision avoidance based on modified quantum particle swarm optimization," *Complexity*, vol. 2020, Article ID 3095426, 15 pages, 2020.

[32] G. Han, Z. Zhou, T. Zhang et al., "Ant-colony-based complete-coverage path-planning algorithm for underwater gliders in ocean areas with thermoclines," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8959–8971, 2020.

[33] A. Ammar, H. Bennaceur, I. Châari, A. Koubâa, and M. Alajlan, "Relaxed Dijkstra and A∗ with linear complexity for robot path planning problems in large-scale grid environments," *Soft Computing*, vol. 20, no. 10, pp. 4149–4171, 2016.

[34] Zfan-tech, "Datasheet of "ROV8T-AJ" underwater vehicle," http://www.zfan-tech.com/.

[35] M. Chen and D. Zhu, "Optimal time-consuming path planning for autonomous underwater vehicles based on a dynamic neural network model in ocean current environments," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 14401–14412, 2020.