Academic Journal of Science and Technology
ISSN: 2771-3032 | Vol. 20, No. 1, 2026

Research on the Optimization of Emergency Material Delivery Routes for Unmanned Aerial Vehicles

Xiaoyu Liu, Zhaoxin Zhou

School of Shipping, Shandong Jiaotong University, Weihai Shandong, 264200, China

Abstract: The optimization of the distribution path for emergency supplies on the port anchorage ships is a key link in the two-stage collaborative process of "operation allocation + route planning" for unmanned aerial vehicles (UAVs), directly influencing the feasibility and efficiency of the final distribution plan. To address this issue, this paper models the three-dimensional flight path of the UAV using B-spline curves, aiming to minimize the flight path length and penalize violations of constraints. A multi-constraint path optimization model is constructed. To efficiently solve this model, an improved particle swarm optimization (PSO) algorithm is designed. By optimizing the B-spline control points to generate smooth trajectories and introducing adaptive inertia weights, Levy flight, and failure restart mechanisms, the global search and local optimization capabilities are enhanced. Simulation experiments based on Python show that the proposed algorithm can effectively optimize the distribution path of emergency supplies for UAVs, generating shorter and safer flight trajectories. This research provides a feasible path optimization scheme and collaborative management strategy reference for UAV emergency supply distribution in complex marine environments.

Keywords: Two-stage Collaborative Process of "Operation Allocation + Route Planning"; Emergency Upplies; B-spline; PSO.

1. Introduction

The traditional research on path optimization focuses on the vehicle routing problem and has now developed a relatively complete theoretical framework and solution methods. With the development of low-altitude logistics networks and drone delivery scenarios, the optimization of drone flight paths has gradually become a research focus in the field of intelligent transportation. Drone flight path optimization occurs in complex environments, considering factors such as energy consumption, collision risks, and path length, and seeking the best path between the starting point and the destination [1]. This optimization process is crucial for the successful completion of tasks by drones, and the selection of path optimization algorithms is the key to finding high-quality paths. A good path optimization algorithm not only ensures the optimality of the path within the global scope but also enables drones to avoid dynamic obstacles in real time, thereby significantly improving the overall flight efficiency of drones.

Phung et al. proposed a spherical vector-based particle swarm optimization (SPSO) algorithm, which is based on spherical vectors, to address the path optimization problem of UAVs in complex environments[2]. Jayaweera et al. proposed a new type of UAV path optimization method based on artificial potential field (APF) to deal with the problem of UAVs flight interference by strong winds[3]. Han et al. proposed a set of grid-based optimization algorithm for indoor path optimization of UAVs to achieve the optimal path planning. Sonny et al. proposed a Q-learning algorithm and used a grid graph to promote the efficient path optimization of UAVs[4]. Yu et al. developed a hybrid grey wolf optimization algorithm and differential evolution (HGWODE) algorithm to solve the UAV path planning problem in disaster scenarios[6]. Tu et al. proposed a differential evolution combined marine predators algorithm (DECMPA) to solve the UAV path optimization problem in complex scenarios[7]. Wang et al. proposed an improved human evolution optimization algorithm (IHEOA) to optimize the flight path of UAVs[8].

In conclusion, researchers have continuously explored solutions to the problem of UAV flight paths by improving various heuristic algorithms. The improvements of these algorithms have provided ideas and methods for this study. At the same time, when dealing with the complex marine environment, other factors such as wind, marine vessels, and islands should be considered during the planning of UAV flight paths. Therefore, for the problem of UAV delivery of living supplies in the marine environment, further research and development of flight path optimization methods adapted to the marine environment are needed.

2. Model

2.1. Problem Description

In response to the emergency material distribution needs of ships at anchor, this study focuses on the three-dimensional path optimization problem of UAVs delivering supplies from the port distribution center to the target ships. Due to the difference between the actual flight distance and the two-dimensional projection, and considering the performance constraints of the UAVs and external environmental factors, the complexity of path planning has significantly increased. To achieve safe and efficient delivery tasks, this study optimizes the three-dimensional flight path.

Firstly, an environmental model based on continuous three-dimensional space is constructed, with the distribution center as the starting point and the coordinates of the target ship as the end point. Multiple three-dimensional cuboids are randomly generated to simulate static obstacles such as ships and no-fly zones to depict the real airspace restrictions. On this basis, a series of three-dimensional waypoints are defined to form the flight path, and B-spline curves are used for interpolation to generate a smooth and continuous trajectory that conforms to the maneuvering performance of the UAVs. The core of this modeling method is to optimize the path

composed of the starting point, end point, and intermediate control points while meeting the physical constraints, in order to achieve effective obstacle avoidance and minimize the flight distance.

2.2. Assumptions

To ensure the safe and efficient emergency material delivery by unmanned aircraft in the maritime environment and to clearly define the research boundaries of this study to reduce the complexity of model solution, the following basic assumptions were made when constructing the path optimization model:

(1) The obstacle information in the three-dimensional grid environment model is known and definite, and in a single path planning cycle, the environment is static and does not consider the sudden appearance or movement of obstacles;

(2) The unmanned aircraft flies at a constant speed along a straight line between adjacent two path points;

(3) Complex meteorological conditions such as wind, rain, and fog do not have a real-time impact on the flight path;

(4) No coverage of any mission points;

(5) Obstacles do not overlap with each other;

(6) Completely located within the expanded planning space..

2.3. Model Building

The core objective of path optimization is to minimize the total flight distance while ensuring that the path meets the obstacle avoidance, minimum flight altitude, and airspace boundary constraints. To achieve this, this study has constructed a target function that includes distance terms and constraint penalty terms. The distance term represents the total length of the flight path; the constraint penalty term penalizes situations where the path points conflict with the safe area of obstacles, are below sea level, or exceed the planned spatial boundaries. By assigning a larger weight to the penalty terms, the optimization algorithm will prioritize searching for feasible paths that satisfy all constraints, and then find the shortest path among them.

In the three-dimensional path optimization stage, this study uses B-spline curves as the parametric representation of the trajectory. B-spline curves are defined by a set of spatial anchor points called control points, and their shape is determined by the positions of these control points. In this stage, the optimization variables are directly defined as the three-dimensional coordinate set of these control points.

Let for a given path (such as from node i to node j), the set of control points corresponding to the trajectory be:

$$C=\left\{c_0,c_1,c_2,\cdots,c_M\right\},\ \ c_m=\left(x_i,y_i,z_i\right)^T\in R^3$$

Among them, c_0 and c_M represent the coordinate positions of each pair of ships in the emergency material distribution sequence for each unmanned aircraft, which are determined by the preset operation allocation results. The intermediate control point $\left\{c_1,c_2,\cdots,c_{M-1}\right\}$ is the decision variable to be optimized. The algorithm adjusts the coordinates of the intermediate control point through iterative processes, indirectly optimizing the flight trajectory. Moreover, to facilitate the calculation of the trajectory length and detection of constraint violations, the B-spline curve needs to be discretized into a set of finite spatial points for evaluation. These points are called trajectory sampling points.

Given a set of control points C, first generate a continuous trajectory function $P\left(u;C\right)$ through B-spline interpolation, where $u\in\left[0,1\right]$ is the normalized curve parameter. Then, uniformly select N values in the parameter space:

$$u_k=\frac{k}{N-1},k=0,1,\cdots,N-1$$

And calculate the corresponding spatial coordinates, obtaining the trajectory sampling point set:

$$P\left(C\right)=\left\{p_0,p_1,p_2,\cdots,p_{N-1}\right\},p_k=\left(x_k,y_k,z_k\right)^T$$

Finally, based on the trajectory sampling points, the distance term and the penalty term for violating constraints in the optimization objective function are approximately calculated.

2.3.1. Distance Item

The distance term in the path optimization aims to minimize the actual flight distance of the UAV in three-dimensional space. In the two-stage optimization framework of this study, the upper-level operation allocation determines the sequence of vessel visits for each UAV, forming a closed loop starting from the port distribution center, visiting each anchor vessel in sequence, and returning. Based on the visit sequence determined by the upper-level operation allocation, the three-dimensional path optimization stage needs to plan a safe and smooth three-dimensional flight path for each pair of adjacent nodes (including the starting point, customer point, and endpoint) in the visit sequence.

In the optimization of three-dimensional flight paths, the calculation of the flight path is not simply the straight-line distance between the starting point and the target point. Instead, it is the total length of the three-dimensional flight path generated by the algorithm and interpolated by B-spline curves. Specifically, the algorithm first optimizes a set of three-dimensional control points, and then uses these control points as the basis to generate a continuous and smooth spatial curve using B-spline interpolation. This curve starts from the given starting points of the two ships, passes through each control point in sequence, and finally reaches the target point. The calculation of the flight path involves densely sampling this continuous curve, obtaining a series of path points, and then accumulating the three-dimensional Euclidean distances between adjacent sampled points N to obtain an approximate total sum. Based on this, the mathematical expression of the unmanned aircraft's flight distance $L\left(C\right)$ is defined as follows:

$$L\left(C\right)=\sum_{k=0}^{N-2}\sqrt{\left(x_{k+1}-x_k\right)^2+\left(y_{k+1}-y_k\right)^2+\left(z_{k+1}-z_k\right)^2}$$

2.3.2. Penalty for Violating Constraints

(1) Obstacle Collision Penalty

Let the obstacle set consist of O rectangular flight-free zones. Each obstacle $o\in O$ is defined as:

$$Z_o=\left[x_o,x_o+l_o\right]*\left[y_o,y_o+w_o\right]*\left[z_o,z_o+h_o\right]$$

Among them, $\left(x_o,y_o,z_o\right)$ represents the coordinates of the smallest corner point, and $\left(l_o,w_o,h_o\right)$ represents the dimension vector. At the same time, to prevent the drone from

colliding with obstacles, a safety distance $s_{safe} > 0$ is defined, and the expanded obstacle area is defined as follows:

$$Z_o^{\text{inflated}} = \left[x_0 - s_{safe},\ x_o + l_o + s_{safe} \right] * \left[\mathrm{y}_o - s_{safe},\ y_o + w_o + s_{safe} \right] * \left[\mathrm{z}_o - s_{safe},\ z_o + h_o + s_{safe} \right]$$

Here, $*$ represents the Cartesian product, meaning that this region consists of all points (x, y, z) that satisfy conditions $x_o - s_{safe} \le x_k \le x_o + l_o + s_{safe}$, $y_o - s_{safe} \le y_k \le y_o + w_o + s_{safe}$ and $z_o - s_{safe} \le z_k \le z_o + h_o + s_{safe}$.

To ensure the safety of the UAV during three-dimensional flight, this study employs an obstacle collision penalty function to conduct collision detection on the path sampling points. This function quantifies the collision risk by determining whether each sampling point is within the expanded boundary of the obstacle: if a point exceeds the extended range of the obstacle in any of the x, y, or z dimensions, it is considered safe; otherwise, a collision is counted. The collision penalty value is the total number of violations of all path points and is combined with the path length through weighted combination to form a composite objective function, thereby converting the rigid obstacle avoidance constraint into a controllable and optimizable soft constraint. This method enables the algorithm to autonomously balance the path economy and flight safety, and by setting a higher penalty coefficient, ensures that the obstacle avoidance requirements are given priority. Thus, the obstacle penalty function is as follows:

$$I_{obs}(p_k) = \begin{cases} 1, & \begin{array}{l} x_o - s_{safe} \le x_k \le x_o + l_o + s_{safe} \\ \text{AND } \mathrm{y}_o - s_{safe} \le y_k \le y_o + w_o + s_{safe} \\ \text{AND } \mathrm{z}_o - s_{safe} \le z_k \le z_o + h_o + s_{safe} \end{array} \\ 0, & else \end{cases}$$

Then the obstacle collision penalty is the total number of violations of the path points:

$$P_{obs}(C) = \sum_{k=0}^{N-1} I_{obs}(p_k)$$

(2) Flight Altitude Penalty

This study introduces a flight altitude penalty mechanism to ensure that the drone's trajectory remains above a safe altitude, avoiding collisions with the sea surface or low-altitude obstacles. This mechanism checks the vertical coordinates of the path sampling points against a threshold. If the altitude is below the preset safety benchmark, it is considered a violation. The penalty value accumulates the number of all violation points and is jointly optimized with the path length through weighted combination. This method transforms the rigid altitude constraint into an optimizable penalty term, enabling the algorithm to adaptively balance between path length and flight safety, thereby enhancing the safety of low-altitude flight in complex environments. The flight altitude function is as follows:

$$I_{height}(p_k) = \begin{cases} 1, & z_k \le 0 \\ 0, & else \end{cases}$$

Then the highly restrictive punishment is:

$$P_{height}(C) = \sum_{k=0}^{N-1} I_{height}(p_k)$$

(3) Boundary Exceedance Penalty

Let the three-dimensional planning space be denoted as $W \subseteq R^3$, and its boundary is defined as:

$$W = \left[x_{\min}, x_{\max} \right] * \left[y_{\min}, y_{\max} \right] * \left[z_{\min}, z_{\max} \right]$$

The boundary values are determined by $p_{\min} = (x_{\min}, y_{\min}, z_{\min})$ and $p_{\max} = (x_{\max}, y_{\max}, z_{\max})$.

The boundary-exceeding penalty function is used to ensure that the drone's path remains within the predefined three-dimensional space. This mechanism independently checks each path point by setting the maximum allowable thresholds for each coordinate direction. If any dimension coordinate exceeds the limit, it is considered a violation. The penalty value counts the number of violations and combines it with the main objective function through weighting, thereby converting the hard spatial constraints into controllable and optimizable soft constraints. This design enables the algorithm to optimize the path quality while dynamically ensuring spatial feasibility, enhancing the stability and compliance of path planning in complex environments. The penalty function for exceeding boundaries is:

$$I_{\text{bound}}(p_k) = \begin{cases} 1, & x_k > x_{\max} \text{ OR } \mathrm{y}_k > y_{\max} \text{ OR } \mathrm{z}_k > z_{\max} \\ 0, & else \end{cases}$$

Then the highly restrictive punishment is:

$$P_{bound}(C) = \sum_{k=0}^{N-1} I_{bound}(p_k)$$

The penalty for violating the constraints is the weighted sum of the above three penalties, that is

$$P(C) = \left(P_{obs}(C) + P_{height}(C) + P_{bound}(C) \right) = \left(\sum_{k=0}^{N-1} I_{obs}(p_k) + \sum_{k=0}^{N-1} I_{height}(p_k) + \sum_{k=0}^{N-1} I_{bound}(p_k) \right)$$

2.3.3. Path Optimization Model under Multiple Constraints

Based on the above analysis, the UAV departs from the starting point of the port distribution center and successively passes through the customer points served by a certain UAV for the purpose of minimizing the distance term and the penalty term for violating constraints during the delivery

process. J multi-constraint condition-based UAV path optimization model is constructed. Assuming that the total cost function for the UAV to complete one cargo delivery is A, then its expression is:

$$\begin{aligned}\min J &= L(C)+\lambda * P(C)\\ &= \sum_{k=0}^{N-2}\sqrt{(x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2+(z_{k+1}-z_k)^2}\\ &+\lambda *\left(\sum_{k=0}^{N-1} I_{obs}(p_k)+\sum_{k=0}^{N-1} I_{height}(p_k)+\sum_{k=0}^{N-1} I_{bound}(p_k)\right)\end{aligned}$$

Here, it is referred to as the "λ penalty coefficient".

To ensure the feasibility of the trajectory, this paper mainly considers the following constraints: obstacle avoidance constraint, altitude constraint and regional constraint. Since the trajectory is represented by B-spline curves, the constraint checks are performed at the discretized path points of the B-spline curves. Specifically, the B-spline curve is uniformly sampled into N points, and then each point is checked to see if it satisfies the following conditions:

$$I_{obs}(p_k) \notin \bigcup_{o\in O} \begin{aligned}&\left[x_o - s_{safe},\ x_o + l_o + s_{safe}\right] * \left[y_o - s_{safe},\ y_o + w_o + s_{safe}\right]\\ &* \left[z_o - s_{safe},\ z_o + h_o + s_{safe}\right]\end{aligned}$$

$$I_{height}(p_k) \geq 0$$

$$r_{\min} \leq I_{\text{bound}}(p_k) \leq r_{\max}$$

Equation (X) represents the objective function for the path optimization of the UAV. Equations (X) - (X) respectively represent the obstacle collision constraints, height constraints, and violation of spatial boundary constraints during the path optimization process.

3. Algorithm Design

3.1. Design of PSO Algorithm

Particle Swarm Optimization (PSO) is an evolutionary computing technique based on swarm intelligence. It was proposed in 1995 by Eberhart and Kennedy, who were inspired by the hunting behavior of bird flocks. This algorithm simulates the information sharing and collaborative search among individuals, driving the group to evolve from disorder to order in the solution space, thereby gradually approaching the optimal solution. Based on the principle of PSO, this study addresses the problem of UAV path optimization, constructs a path planning model under multiple constraints, and uses the PSO algorithm for solution. In the following text, the basic principle of the standard PSO will be first elaborated, followed by a detailed introduction to the improvement strategies adopted in this study. In Section 3.2, the basic principle of the original PSO will be deeply discussed; in Section 3.3, the improvement of the PSO algorithm will be focused on.

3.2. Principle of PSO Algorithm

Firstly, initialize the positions and velocities of the population, and calculate the fitness value of each particle. At the same time, initialize the individual historical optimal position and fitness value of each particle, as well as the global optimal position and fitness value of the entire population. During the iteration process, each particle updates its own velocity and position based on its individual historical optimal and global optimal values, recalculates the fitness, and updates the individual historical optimal and global optimal. The iteration continues until the maximum number of iterations is reached, and the global optimal solution and its fitness are output. The main steps of the PSO algorithm are as follows.

(1) Construction of the solution

In this study, the positions of control points in the drone path optimization are represented using a real number encoding method. Specifically, the flight path consists of the starting point, several control points, and the target point. The three-dimensional coordinates (x, y, z) of the control points are the variables to be optimized. During the algorithm initialization, the coordinate components of each control point are randomly initialized as a real number within the [0,1] interval, thereby forming a real number vector. For example, if the path optimization includes 3 control points, it is encoded as a length-9 real number vector, corresponding to the x, y, and z coordinates of the 3 points.

(2) Calculation of the initial fitness

For each particle's position vector, the decoded normalized coordinates are mapped to the actual geographical space, a three-dimensional control point sequence is constructed, and a continuous flight path is generated through B-spline interpolation. The total length of the path and the conflict penalty with obstacles are calculated, and the fitness value of each particle is obtained by weighted summation. The penalty term quantifies violations such as collision with obstacles, being below the safe height, and exceeding the planning boundary. By setting a larger weight for the penalty term, a compound fitness function is constructed, thereby guiding the optimization algorithm to preferentially search for safe and feasible paths.

(3) Recording of the initial optimal solution

The particle with the best fitness from the initial population is identified, and its position and fitness are recorded as the global optimal position and global optimal fitness, respectively. At the same time, the current position and fitness of each particle are used as its individual historical optimal position and fitness, establishing the initial optimal solution benchmark.

(4) Start the iterative loop

The algorithm enters the main iterative loop, starting from the first generation until the preset maximum number of iterations is reached. In each iteration, all particles in the population are updated successively, and the quality of the solution is gradually optimized.

(5) Update each particle

In the PSO algorithm, the velocity of the particle is determined by the inertia weight, the individual optimal solution, and the global optimal solution, while the position is adjusted based on the updated velocity. The update formulas for the velocity and position of the particle are as shown in (X) and (X).

$$v^{iter} = \omega * v_j^{iter-1} + c_1 * \left(gbest_j - x_j^{iter-1} \right) * r_1 + c_2 * \left(zbest_j - x_j^{iter-1} \right) * r_2$$

$$x_j^{iter} = x_j^{iter-1} + v^{iter}$$

In the formula, $iter$ represents the current iteration number; $iterations$ represents the maximum number of iterations; v^{iter} stands for the velocity of particle j in the $iter$ th generation; v_j^{iter-1} represents the velocity of particle j in the $iter$ th generation; ω is the inertia weight; c_1 and c_2 are learning factors that determine the degree to which particles rely on individual optimal solutions and global optimal solutions during the update process; r_1 and r_2 are random numbers; $gbest_j$ is the individual optimal solution of particle i; $zbest_j$ is the global optimal solution; x_j^{iter-1} represents the position of particle j in the $iter$ th generation.

(6) Evaluate the new position

Decode the updated particle positions into the actual control point sequence, generate the trajectory again through B-spline interpolation, calculate the total length of the new trajectory and the penalty for violating constraints, and obtain the fitness value of the new position.

(7) Update the optimal record

Compare the fitness value of the new position with the historical optimal fitness value of the particle individual. If it is better, update the individual optimal position and fitness; at the same time, compare it with the global optimal fitness value. If it is better, update the global optimal position and fitness. In the improved PSO, a failure restart mechanism may also be included: if a particle fails to improve the individual optimal position for several consecutive times, conduct a local search near the global optimal to escape from the local optimum.

(8) Record the convergence history

After each round of iteration, record the current global optimal fitness value in the history array to form a convergence curve, which is used for subsequent analysis of the optimization process and performance evaluation of the algorithm.

(9) Repeat 4-8 until termination

Repeat the execution of steps 4 to 8 until the maximum number of iterations is reached or other termination conditions are met (such as fitness convergence or reaching a preset accuracy). At this point, the algorithm stops the optimization iteration.

(10) Decode and output the result

After the iteration termination, perform the final decoding of the global optimal position to obtain the actual control point coordinates, generate the optimal trajectory through B-spline interpolation. At the same time, encapsulate the trajectory point sequence, fitness value, convergence history, calculation time, etc. into a result object and return it, providing a complete solution for subsequent visualization, verification or practical application.

.

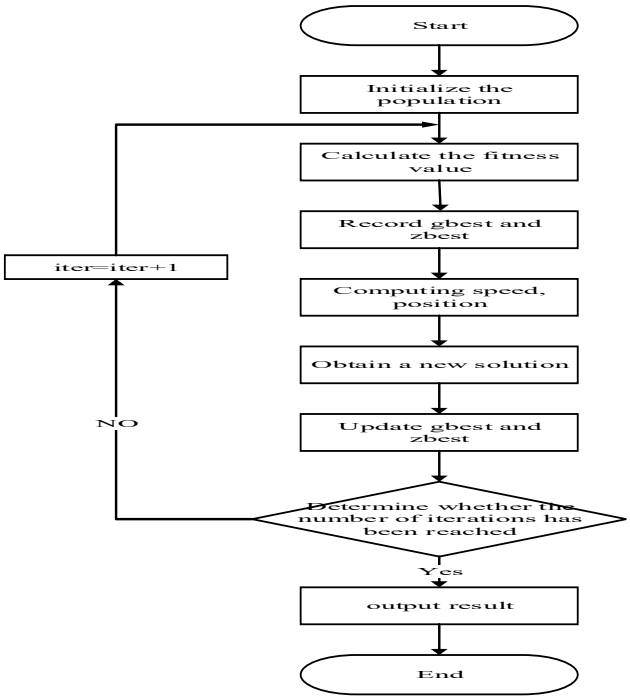


Figure 1. Overall flowchart of PSO algorithm

3.3. Improving the PSO Algorithm

Based on the traditional PSO algorithm, this section improves the algorithm performance by using an improved adaptive inertia weight, introducing Levy flight, and a failure restart mechanism, thereby finding a better global solution. The specific improvement methods are as follows.

(1) Adaptive Inertia Weight

In the improved PSO algorithm, the inertia weight adopts an adaptive adjustment strategy based on the particle's fitness to achieve a dynamic balance between global exploration and local exploitation. Specifically, for each particle, if its fitness is better than the population average, the inertia weight is linearly interpolated between its preset maximum and minimum values, allowing the better particles to maintain a stronger exploration ability; otherwise, a minimum inertia weight is directly assigned to prompt the poorer particles to conduct a more detailed local search. This mechanism adjusts dynamically with each iteration, promoting the algorithm to smoothly transition from global search to local optimization, thereby effectively preventing premature convergence and improving convergence accuracy. Specifically, the calculation method of the adaptive inertia weight is as follows:

$$\omega = \begin{cases} \omega_{\min} + \dfrac{(\omega_{\max} - \omega_{\min}) * f_j - f_{\min}}{f_{avg} - f_{\min}} & , \ f_j < f_{avg} \\ \omega_{\min} & , \ else \end{cases}$$

Among them, ω represents the adaptive inertia weight; $\omega_{\min}$ represents the minimum inertia weight; $\omega_{\max}$ represents the maximum inertia weight; f_j represents the fitness value of the j-th particle in the current iteration; $f_{\min}$

represents the minimum fitness value of all particles in the current population; $f_{\max}$ represents the maximum fitness value of all particles in the current population; f_{avg} represents the average fitness value of all particles in the current population.

(2) Incorporation of Levy Flight

In the improved PSO, Levy Flight is introduced into the velocity update mechanism as an intelligent disturbance strategy based on the heavy-tailed distribution. This mechanism calculates the scale parameter by calling a dedicated Levy Flight generation function and generates a random step vector following the Levy distribution. Specifically, the algorithm uses the Mantegna algorithm to generate Levy random numbers, and its mathematical expression is:

$$Levy(\beta)=\frac{u}{|v|^{\frac{1}{\beta}}}$$

Here, β represents a distribution parameter (typically set to 1.5), while u and v are two independent normal distribution random variables: $u \sim N(0,\sigma^2)$ and $v \sim N(0,1)$. The scale parameter σ is determined by formula (X), and Γ is the gamma function. This distribution characteristic manifests as a pattern combining a large number of short-distance movements with occasional long-distance jumps, which precisely meets the requirements of the optimization algorithm for the balance between global exploration and local exploitation.

$$\sigma=\left(\frac{\Gamma(1+\beta)\sin\left(\frac{\pi\beta}{2}\right)}{\beta\Gamma\left(\frac{1+\beta}{2}\right)2^{\frac{\beta-1}{2}}}\right)^{\frac{1}{\beta}}$$

In the improvement of the PSO algorithm, the Levy perturbation term is added to the velocity update formula (X) in the form of equation (X). This design has a dual regulating function: on the one hand, the fixed coefficient η controls the perturbation intensity to prevent excessive jumping that may disrupt the convergence; on the other hand, the time decay factor $1-\frac{iter}{iterations}$ causes the perturbation intensity to linearly decrease with the iterative process, ensuring that the algorithm initially enhances the global exploration ability with a larger Levy step size and escapes the local optimal region, while gradually weakening the perturbation and turning to local fine search in the later stage. This dynamic balanced perturbation mechanism significantly improves the global search efficiency and convergence accuracy of the algorithm in complex multi-modal optimization problems. The Levy flight perturbation term and the improved velocity calculation formula are as follows.

$$Levy = Levy(\beta)*\left(1-\frac{iter}{iterations}\right)$$

$$v_{new}^{iter}=\omega*v_j^{iter-1}+c_1*\left(pbest_j-x_j^{iter-1}\right)*r_1+c_2*\left(gbest_j-x_j^{iter-1}\right)*r_2+\eta*Levy$$

(3) Introduction of Restart Mechanism

In the improved PSO algorithm, a failure restart mechanism was introduced to address the issue of particles getting trapped in local optima. This mechanism records the consecutive number of times the individual optimal solution of the particle has not been updated. When this count exceeds the threshold, it is considered that the particle's search has stagnated. Subsequently, a multi-scale perturbation is applied within a gradually expanding neighborhood centered on the current global optimal solution, and new solutions are generated through a normal distribution to ensure they are within the feasible domain. If the new solution is better than the original individual optimal solution, the particle's state is updated and the failure count is reset. This method enables an incremental and refined search near the global optimal solution, effectively escaping local optima while maintaining the excellent structure of the solution, thereby significantly improving the search efficiency and convergence accuracy of the algorithm in complex spaces.

4. Numerical Experiment Analysis

In order to verify the correctness of the proposed model and the effectiveness of the improved algorithm, this study conducted a path optimization simulation experiment by using a set of preset job allocation scheme data, taking into account the customers' time-sensitive requirements.

4.1. Simulation Environment and Parameter Settings

Table 1. relevant parameter

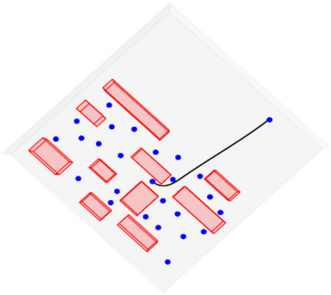
Description	Parameters	Numerical Value
B-spline interpolation points	C	3
B-spline trajectory sampling points	N	100
Collection of obstacles	O	15
Penalty coefficient	λ	10
Iterations	/	100
Population quantity	/	80
Levy coefficient	η	0.01
Learning factor	ε_1 ε_2	1.1425
Inertia weight	$\omega_{\min}$、$\omega_{\max}$	0.5/0.7
Safe distance	s_{safe}	0.05

To ensure the objectivity and reproducibility of the experimental results, this paper selects Python 3.10 as the simulation platform to model and solve the scenario of optimizing the emergency material distribution path for UAVs at the Lingang anchorage. In the environmental modeling aspect, a continuous three-dimensional space modeling method is adopted, where obstacles are abstracted as three-

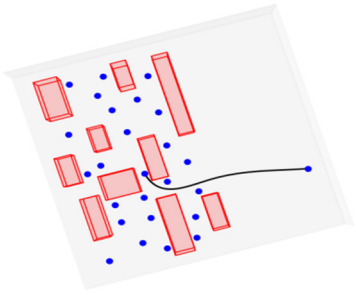
dimensional rectangular prisms with randomly generated positions and sizes, thereby constructing an offshore flight simulation environment containing static obstacles. The path planning uses B-spline curves to represent the trajectory, and smooth paths are generated by optimizing the coordinates of the control points. To ensure the consistency of each simulation experiment, all experiments are conducted under the same environmental configuration and parameter settings. The specific parameter settings are detailed in Table 1.

4.2. Example Results and Analysis Discussion

To verify the performance improvement of the Improved Particle Swarm Optimization Algorithm (IPSO) in the three-dimensional trajectory optimization problem of UAVs, this section designs and conducts a comparative experiment. The experiment takes the standard Particle Swarm Optimization Algorithm (PSO) as the benchmark and tests it on the same trajectory optimization model and problem instance, as shown in Figure 2 the effectiveness graph of IPSO and PSO algorithms. To ensure the fairness of the comparison and the reliability of the conclusion, the algorithms are evaluated from three dimensions: convergence accuracy, solution stability, and comprehensive optimization performance. Specifically, the minimum, maximum, and average objective function values found by the algorithms in multiple independent runs are recorded, and the standard deviation of the objective function is calculated. Through the comparative analysis of the above statistical indicators, the comprehensive effect of the IPSO algorithm after introducing adaptive inertia weight, Levy flight perturbation, and failure restart mechanism in enhancing the global search ability and avoiding premature convergence can be comprehensively evaluated. The intuitive display of the algorithm performance comparison is shown in Table 2.



(a) Diagram illustrating the effectiveness of the IPSO algorithm



(b) Diagram illustrating the effectiveness of the PSO algorithm

Figure 2. Effectiveness Diagram of Improved and Unimproved PSO Algorithm

Table 2. Analysis of PSO and IPSO Indicators

evaluation index	Minimum objective function value	Maximum objective function value	Average objective function value	Standard deviation of the objective function value
PSO	5.0396	5.8126	5.2511	0.2939
IPSO	4.7116	5.0411	4.8641	0.1396

From Figures 2 and Table 2, it can be concluded that after running the test algorithm 5 times, the minimum and average objective function values obtained by the IPSO algorithm are superior to those of the standard PSO algorithm. Specifically, the optimal objective function value of the IPSO algorithm is 4.7116, and the average objective function value is 4.8641. In contrast, the optimal objective function value of the PSO algorithm is 5.0396, and the average objective function value is 5.2511. This indicates that the IPSO algorithm has improved in solution accuracy and can find better trajectory planning solutions. Additionally, considering the stability aspect, the standard deviation of the objective function values of the IPSO algorithm is 0.1396, which is lower than 0.2939 of the standard PSO algorithm, demonstrating that the IPSO algorithm has better stability and is less affected by the initial random population. Therefore, the IPSO algorithm outperforms the standard PSO algorithm in terms of convergence accuracy, convergence speed, and stability, verifying the effectiveness of the proposed improvement strategies in this paper.

After verifying the effectiveness of the IPSO algorithm, and to demonstrate the feasibility of the "task allocation + path planning" two-stage collaborative optimization framework, this subsection applies the IPSO algorithm to a set of preset job allocation scheme data. The detailed flight distance comparison data can be found in Tables 3-4, while the visualization effect comparison of the optimized three-dimensional trajectory is presented in Figure 3.

Table 3. Flight Route Solution Set 1 for UAV Operations on Ships at the Inner Anchorage Area of Qingdao Port

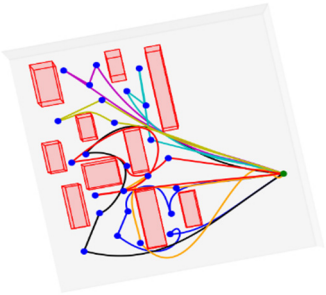
Sequence of drone operations	The flight distance of the UAV	
	IPSO	PSO
0-16-23-18-24-0	13.52	17.98
0-11-19-25-0	16.97	25.14
0-2-4-1-0	15.48	15.92
0-5-7-3-10-0	13.78	19.98
0-9-8-6-0	13.76	13.79
0-15-21-22-17-0	13.27	13.77
0-14-12-13-20-0	17.78	17.63
	104.56	124.21

Table 4. Flight Route Solution Set 2 for UAV Operations on Ships at the Inner Anchorage Area of Qingdao Port

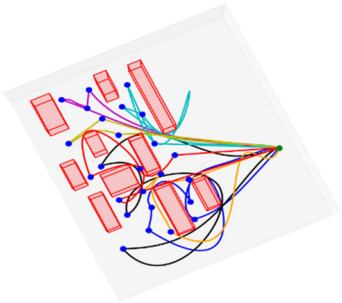
Sequence of drone operations	The flight distance of the UAV	
	IPSO	PSO
0-15-21-24-0	10.71	10.75
0-3-2-1-4-9-10-14-0	16.66	23.59
0-22-12-8-6-5-7-0	15.23	18.24
0-11-13-20-19-23-0	19.25	19.52
0-16-18-25-17-0	15.25	31.45
	77.1	103.55

By comparing and analyzing the flight distances of the drones in Tables 3 to 4, it can be seen that IPSO demonstrates significant advantages in the flight path planning of drones. In the simulation experiment of the anchorage area in Qingdao Port, the flight paths planned by the IPSO algorithm were significantly shorter than those of the PSO algorithm. This indicates that the IPSO algorithm has better global convergence and local optimization capabilities in path optimization.

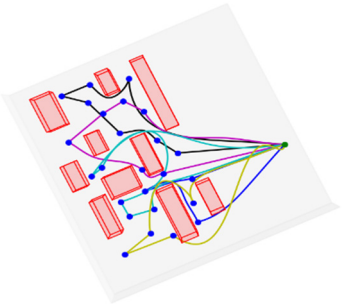
Specifically, in the experiment at the anchorage of Qingdao Port, for solution set 1, the flight distance of the drone planned by the IPSO algorithm was 104.56, while that of the PSO algorithm was 124.21, reducing the flight distance by 15.82%; in solution set 2, IPSO also performed well, with the flight distance reduced by 25.54% compared to the PSO algorithm.



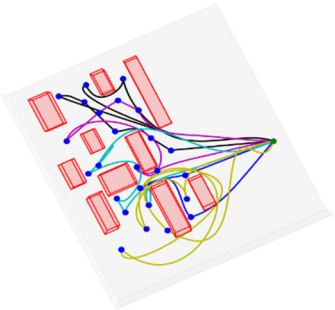
(a) Optimization solution set of IPSO path for Qingdao Port 1



(b) Optimization solution set of PSO path for Qingdao Port 1



(a) Optimization solution set of IPSO path for Qingdao Port 2



(b) Optimization solution set of PSO path for Qingdao Port 2

Figure 3. Path Optimization Diagrams of Improved and Unimproved PSO Algorithms

From the analysis in Figures 3(a) to (d), it can be seen that in the case of Qingdao Port, the comparison of the three-dimensional path optimization results of the IPSO and PSO algorithms shows:

In terms of the global optimality and economy of the path. Intuitively, the flight paths planned by the IPSO algorithm (Figures (a) and (c)) generally show a shorter and more direct overall trend in most scenarios. Compared with the PSO algorithm (Figures (b) and (d)), which often shows obvious detours or detours, the paths generated by IPSO are more efficient in connecting the starting point and the end point. This indicates that its global search ability is stronger, can better approach the theoretical shortest path, and directly reduces the flight energy consumption and time cost of the unmanned aircraft.

In terms of the precision and safety of local obstacle avoidance. In terms of the details of obstacle avoidance, the IPSO algorithm demonstrates higher planning intelligence. When the path approaches the surface of obstacles, it often maintains a more reasonable and consistent safe distance, and the path direction conforms to the contour of the obstacles but avoids being too close. In contrast, the results of some PSO algorithms show that the paths are too close to the edges of obstacles or have unnecessary oscillations at the corners of obstacles. This indicates that the local search mechanism of IPSO can handle constraint boundaries more precisely, generating safer flight paths.

In terms of the smoothness of the path and flight feasibility. The paths generated by the IPSO algorithm are generally smoother and more fluent. The trend of the sequence of flight points changes continuously, the curvature of the turns is natural, and it is more in line with the kinematics and dynamics constraints of the unmanned aircraft, which is conducive to the stable execution of flight control. While in some results of the PSO algorithm, it can be observed that the flight paths have relatively sharp corners or small oscillations. Such paths are geometrically achievable but will increase the difficulty of aircraft control and may lead to speed loss or additional energy consumption in actual flight.

5. Conclusion

This study focuses on the emergency material distribution scenario for ships in port anchorage areas, systematically exploring the problem of UAV path optimization. A two-stage collaborative optimization framework integrating "task allocation + trajectory planning" was proposed. By establishing a three-dimensional trajectory model based on B-spline curves and introducing a path optimization objective function under multiple constraints, the trade-off between trajectory length and safety was effectively balanced. To solve this model, an improved particle swarm optimization algorithm was designed and implemented, incorporating adaptive inertia weight, Levy flight perturbation, and failure restart mechanisms, significantly enhancing the algorithm's global search ability and convergence stability in complex three-dimensional environments.

The main contributions of this study can be summarized as

follows:

(1) A multi-constraint UAV trajectory optimization model suitable for complex marine environments was constructed. This model aims to minimize the path length and handles constraints such as obstacle avoidance, flight altitude, and airspace boundaries through a penalty function mechanism, providing a mathematical model foundation for the safe flight of UAVs in dynamic obstacle environments.

(2) An improved particle swarm optimization algorithm was proposed. Through adaptive inertia weight adjustment, Levy flight perturbation, and failure restart mechanisms, the algorithm's global exploration and local exploitation capabilities were significantly enhanced. Simulation experiments show that compared to the standard PSO algorithm, the improved algorithm has obvious advantages in convergence accuracy, stability, and path quality, and can generate shorter, smoother, and safety-compliant flight trajectories.

(3) The feasibility and effectiveness of the proposed model and algorithm were verified through multiple sets of simulation experiments in actual anchorage areas such as Qingdao Port. The UAV paths planned by the improved algorithm significantly reduced flight distance and improved distribution efficiency, confirming the practical application potential of this research in emergency material distribution tasks.

(4) This study provides a systematic solution for UAV path planning and collaborative operations in complex marine environments, having certain theoretical significance and practical reference value, especially applicable to scenarios with high timeliness requirements such as port emergency logistics and ship material replenishment.

However, this study still has certain limitations. Future work can be expanded in the following aspects:

(1) The current study assumes a static environment and known obstacle information. Future work can introduce dynamic obstacle models and robust path planning methods in uncertain environments.

(2) This study focuses on a single UAV departing from a single distribution center. Future work can be extended to multi-UAV collaboration and multi-distribution center scheduling system optimization problems.

(3) Meteorological factors such as wind, waves, and visibility in actual maritime environments significantly affect UAV flight. Future research can combine weather forecasts and real-time sensing data to construct an environment-adaptive path planning model.

(4) Further integration of online learning methods such as reinforcement learning can be combined to achieve autonomous re-planning and intelligent decision-making for UAVs in dynamic tasks and unexpected failures.

In conclusion, this study provides feasible ideas and method support for UAV path optimization in port anchorage area emergency material distribution through modeling and algorithm improvement, with certain theoretical innovation and practical application prospects.

References

[1] Aggarwal S, Kumar N. Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges[J]. Computer communications, 2020, 149: 270-299.

[2] Phung M D, Ha Q P. Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization[J]. Applied Soft Computing, 2021, 107: 107376.

[3] Jayaweera H M P C, Hanoun S. Path planning of unmanned aerial vehicles (UAVs) in windy environments[J]. Drones, 2022, 6(5): 101.

[4] Han B, Qu T, Tong X, et al. Grid-optimized UAV indoor path planning algorithms in a complex environment[J]. International Journal of Applied Earth Observation and Geoinformation, 2022, 111: 102857.

[5] Sonny A, Yeduri S R, Cenkeramaddi L R. Q-learning-based unmanned aerial vehicle path planning with dynamic obstacle avoidance[J]. Applied Soft Computing, 2023, 147: 110773.

[6] Yu X, Jiang N, Wang X, et al. A hybrid algorithm based on grey wolf optimizer and differential evolution for UAV path planning[J]. Expert Systems with Applications, 2023, 215: 119327.

[7] Tu B, Wang F, Han X. 3D path planning for UAV based on a hybrid algorithm of marine predators algorithm with quasi-oppositional learning and differential evolution[J]. Egyptian Informatics Journal, 2024, 28: 100556.

[8] Wang X, Zhou S, Wang Z, et al. An improved human evolution optimization algorithm for unmanned aerial vehicle 3D trajectory planning[J]. Biomimetics, 2025, 10(1): 23.