Essay

Drone Path Planning Based on an Improved Whale Optimisation Algorithm

Qixiang Nie , Guangxun Wang , Xinxing Shi , Xuechen Liang *

Posted Date: 26 March 2026

doi: 10.20944/preprints202603.2132.v1

Keywords: UAV; path planning; whale algorithm; rapidly expanding random tree-star; non-linear inertial weights

 doi:10.20944/preprints202603.2132.v1

Essay

Drone Path Planning Based on an Improved Whale Optimisation Algorithm

Qixiang Nie, Guangxun Wang, Xinxing Shi and Xuechen Liang *

School of Transportation Engineering, East China Jiaotong University, Nanchang 330013, Jiangxi, China
* Correspondence: 2023138086100103@ecjtu.edu.cn; Tel.: +86-155-5237-1749

Abstract

To address the issues of insufficient convergence performance and high sensitivity to local optima in the traditional Whale Optimization Algorithm (WOA) when handling 3D path planning tasks for unmanned aerial vehicles (UAVs), this paper proposes an improved UAV path planning algorithm based on the Whale Optimization Algorithm (R*WOA). Firstly, the global search capability and path optimisation mechanism of the Rapidly Expanding Random Tree Star (RRT*) algorithm are utilised to generate a high-quality initial population, thereby enhancing population diversity and the algorithm's global exploration capability; Secondly, the linear convergence factor of the traditional WOA is adjusted to a non-linear dynamic adjustment strategy based on the cosine function, enhancing global search capability in the early stages of iteration and local search capability in the later stages; simultaneously, a non-linear inertial weight is employed to modulate the position update mechanism of individuals, further enhancing the algorithm's local optimisation accuracy and convergence stability in the later stages of iteration. Finally, comparative experimental results on a basic test function set and in scenarios constructed using Digital Elevation Models (DEMs) demonstrate that R*WOA exhibits stable optimisation performance, capable of planning safer paths that are shorter in length and smoother in trajectory.

Keywords: UAV; path planning; whale algorithm;rapidly expanding random tree-star; non-linear inertial weights

1. Introduction

With the rapid development of drone technology, its applications are becoming increasingly widespread in fields such as meteorological observation and analysis, emergency rescue, traffic management, logistics and delivery, and disaster emergency management [1–3]. Among these, path planning plays a pivotal role in the autonomous decision-making and intelligent payload mission planning systems of drones, and has gradually evolved into a significant area of research [4,5].

Traditional path planning algorithms include the A* algorithm [6–8], the artificial potential field method [9–11] and the Rapidly Searching Random Tree (RRT*) algorithm [12–14], amongst others. As a heuristic search algorithm, the A* algorithm guides the search via an evaluation function and guarantees the discovery of the optimal path provided the heuristic function is sound. However, computational and storage overheads increase sharply with rising spatial complexity, and the algorithm struggles to handle complex terrain environments. The artificial potential field method models the target and obstacles as attractive and repulsive sources respectively, guiding the UAV in real time via the resultant force field. This generates naturally smooth paths and is computationally efficient, making it suitable for dynamic obstacle avoidance; however, it is prone to getting stuck in local minima and may fail in narrow passages or near obstacle-adjoining target points. Conversely, the RRT* algorithm retains the advantages of RRT, such as probabilistic completeness, whilst introducing a cost-optimisation mechanism: when a new node is inserted into the tree, the algorithm searches within its neighbourhood for a parent node with a lower cost to optimise the path of that node, and attempts

 doi:10.20944/preprints202603.2132.v1

to designate the new node as a better parent for existing nodes within the neighbourhood. Through iterative optimisation, as the number of samples approaches infinity, the algorithm converges to the global optimal path with probability 1; however, the paths it generates are typically suboptimal and winding rather than smooth, and the convergence speed is significantly influenced by the sampling strategy.

To better address the challenges of UAV path planning in complex environments and overcome the limitations of traditional path planning algorithms—such as insufficient optimisation capabilities and limited adaptability in complex scenarios—researchers have shifted their focus to swarm intelligence algorithms with global optimisation capabilities, such as genetic algorithms (GA) [15,16], Whale Optimization Algorithm (WOA)[17–19], Harris-Hawk Optimization Algorithm (HHO)[20–22] and other widely applied algorithms. By simulating the evolutionary mechanisms of natural populations, these algorithms continuously adjust search strategies and optimise exploration directions. Compared to traditional methods, swarm intelligence algorithms do not rely on specific problem structures or gradient information, and demonstrate greater robustness against the curse of dimensionality. By leveraging the co-evolutionary mechanisms of population individuals, these algorithms achieve a dynamic trade-off between global search and local exploration. They demonstrate significant advantages in solving complex constrained, non-linear and multi-peaked optimisation problems, providing efficient and feasible solutions for path planning problems in complex, multi-obstacle environments.

The Whale Optimization Algorithm (WOA), proposed by Mirjalili et al. [23] in 2016, is an emerging swarm intelligence optimisation method. Compared with other intelligent optimisation methods, WOA offers significant advantages such as simple parameter settings, a stable search mechanism and ease of implementation, making it a favoured choice among many researchers. Currently, WOA has been widely applied in the field of path planning[24–26]. However, when applied to the problem of three-dimensional path planning for unmanned aerial vehicles (UAVs), the algorithm still exhibits certain limitations. Specifically, within large-scale search spaces, as the computational complexity of the optimisation task increases significantly, the algorithm's solution accuracy declines markedly, making it difficult to plan high-quality globally optimal paths. More importantly, due to its relatively simple evolutionary mechanism, population diversity is rapidly lost during the iterative process, causing the algorithm to easily become trapped in local optima and thus rendering it unable to effectively handle complex and dynamic three-dimensional multi-obstacle scenarios.

Consequently, this paper proposes an improved Whale Optimization Algorithm (R*WOA) based on the Rapidly Expanding Random Tree Star (RRT*) algorithm for solving the UAV path planning problem. The main findings are summarised as follows:

- Firstly, the global search capability and path optimisation mechanism of RRT* are utilised to generate a high-quality initial population, thereby providing the algorithm with a superior starting point for the search.
- The linear convergence factor of the traditional WOA is adjusted to a non-linear dynamic adjustment strategy based on the cosine function, enhancing global search capabilities in the early stages of iteration and local search capabilities in the later stages; simultaneously, a non-linear inertial weight is employed to modulate the position update mechanism of individuals, further enhancing the algorithm's local optimisation accuracy and convergence stability in the later stages of iteration.

To verify the effectiveness of the proposed algorithm, the optimisation performance of the R*WOA algorithm was tested using standard test functions. Subsequently, a three-dimensional flight space for the UAV was established based on a Digital Elevation Model (DEM) to conduct path planning verification experiments. The results indicate that, compared with other improved algorithms, the R*WOA algorithm is capable of planning more optimal flight trajectories whilst ensuring the safety of the UAV's flight.

 doi:10.20944/preprints202603.2132.v1

2. Problem Modelling

This section will introduce the constraints and the definition of the objective function for the drone.

2.1. Constraints

2.1.1. Path Length Cost

To improve the operational efficiency of unmanned aerial vehicles (UAVs), this paper adopts path length minimisation as the optimisation objective. During autonomous flight, the UAV is pre-loaded with a list of waypoints it must pass through. Consequently, a single flight path comprises a number of waypoints, each of which contains three coordinate components, as shown in Figure 1. The cost function for path length is defined as follows:

$$F_1(X_i) = \sum_{j=1}^{n-1} \left\| \overrightarrow{P_{ij}P_{i,j+1}} \right\| \tag{1}$$

$\left\| \overrightarrow{P_{ij}P_{i,j+1}} \right\|$ represents the Euclidean distance between two waypoints.

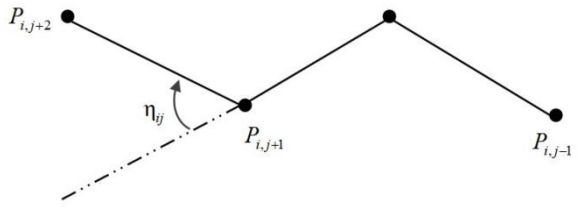


Figure 1. Schematic diagram of flight paths and their variables.

2.1.2. Cost of Security Threats

To ensure that the drone arrives safely at its destination, the constraints imposed by obstacles within the flight space must be fully taken into account when planning the route. The safety threat cost of the drone varies according to the risk levels of different areas. For the path segment $\left\| \overrightarrow{P_{ij}P_{i,j+1}} \right\|$, its threat cost is related to the distance d_k from that path segment to C_k. Consider the drone's diameter D and the danger radius S of the collision zone. The threat cost F_2 is calculated by iterating over all waypoints P_{ij} and the obstacle set K:

$$F_2(X_i) = \sum_{j=1}^{n-1} \sum_{k=1}^{K} T_k\left(p_{ij}\overrightarrow{P_{i,j+1}}\right)$$

$$T_k\left(p_{ij}\overrightarrow{P_{i,j+1}}\right) = \begin{cases} 0, & \text{if } d_k > S + D + R_k \\ (S + D + R_k) - d_k, & \text{if } D + R_k < d_k \leq S + D + R_k \\ \infty, & \text{if } d_k \leq D + R_k \end{cases} \tag{2}$$

Here, k denotes the set of obstacles, and each hazard is modelled as a cylindrical region, where C_k is the centre coordinate of the projection and R_k is the radius (as shown in Figure 2).

 doi:10.20944/preprints202603.2132.v1

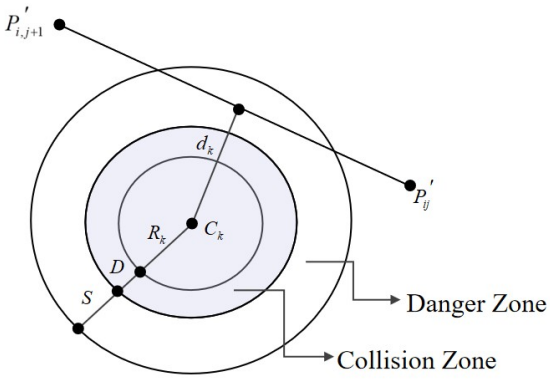


Figure 2. Obstacle model diagram.

2.1.3. Cost of Flight Altitude

During mission execution, the UAV should maintain a highly stable flight attitude in order to significantly reduce energy consumption. Let the upper and lower limits of the flight altitude be $h_{\max}$ and $h_{\min}$ respectively, and let the current flight altitude of the UAV be h_{ij}; then, the flight altitude cost corresponding to waypoint P_{ij} can be expressed as:

$$H_{ij} = \begin{cases} \left| h_{ij} - \dfrac{h_{\max} + h_{\min}}{2} \right|, & \text{if } h_{\min} \le h_{ij} \le h_{\max} \\ \infty, & \text{otherwise} \end{cases} \tag{3}$$

From Equation (3), it can be seen that H_{ij} serves to penalise height values that exceed the permitted range, whilst constraining the drone to fly near the average altitude. By summing the H_{ij} values for the n waypoints along a single flight path, the total altitude cost F_3 is obtained, expressed as:

$$F_3(X_i) = \sum_{j=1}^{n} H_{ij} \tag{4}$$

2.1.4. Smoothing Cost

In addition to ensuring a smooth flight, path planning must also minimise changes in turn angles and climb/descent angles as much as possible; this requirement is crucial for enhancing flight safety.

Let $\vec{k}$ be the unit vector along the z-axis, as shown in Figure 3. The angle between the two consecutive path segments $\overrightarrow{P'_{ij}P'_{i,j+1}}$ and $\overrightarrow{P'_{i,j+1}P'_{i,j+2}}$ projected onto the horizontal plane Oxy is the turning angle ϕ_{ij}; the projection vectors can then be calculated as follows:

$$\overrightarrow{P'}_{ij}\overrightarrow{P'}_{i,j+1} = \vec{k} \times \left(\overrightarrow{P_{ij}P_{i,j+1}} \times \vec{k} \right) \tag{5}$$

 doi:10.20944/preprints202603.2132.v1

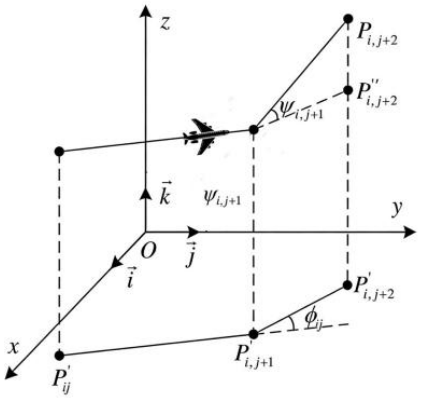


Figure 3. Schematic diagram of cost smoothing.

The turning angle is calculated as follows:

$$\phi_{ij} = \arctan\left(\frac{\left\|\overrightarrow{P'_{ij}P'_{i,j+1}} \times \overrightarrow{P'_{i,j+1}P'_{i,j+2}}\right\|}{\overrightarrow{P'_{ij}P'_{i,j+1}} \cdot \overrightarrow{P'_{i,j+1}P'_{i,j+2}}}\right) \tag{6}$$

The climb angle ψ_{ij} refers to the angle formed between the path segment $\overrightarrow{P_{ij}P_{i,j+1}}$ and its projection onto the horizontal plane, $\overrightarrow{P'_{ij}P'_{i,j+1}}$. It is calculated as follows:

$$\psi_{ij} = \arctan\left(\frac{z_{i,j+1} - z_{ij}}{\left\|\overrightarrow{P'_{ij}P'_{i,j+1}}\right\|}\right) \tag{7}$$

The formula for calculating the smoothing cost is as follows:

$$F_4(X_i) = a_1 \sum_{j=1}^{n-2} \phi_{ij} + a_2 \sum_{j=1}^{n-1} \left|\psi_{ij} - \psi_{i,j-1}\right| \tag{8}$$

In this context, a_1 and a_2 are the penalty factors for the turn angle and the climb angle, respectively.

2.2. Objective Function

It is necessary to comprehensively consider the constraints on path X_i in terms of optimality, safety and feasibility; based on this, the total cost function can be defined as follows:

$$F(X_i) = \sum_{k=1}^{4} b_k F_k(X_i) \tag{9}$$

Here, b_k represents the weighting coefficient, and X_i is the decision variable, which is a list consisting of n waypoints, i.e. $P_{ij} = (x_{ij}, y_{ij}, z_{ij})$, and satisfies $P_{ij} \in \mathcal{O}$ (where $\mathcal{O}$ denotes the operational space of the UAV). $F_1(X_i)$ to $F_4(X_i)$ correspond sequentially to path length cost (1), safety threat cost (2), flight altitude cost (4) and smoothness cost (8).

3. Improving the Whale Algorithm

The improvements to WOA proposed in this paper consist of two main parts: during the initialisation phase, RRT* is employed to initialise particle positions, relying on its uniform sampling property to ensure initial population diversity; During the evolutionary iteration phase, the linear convergence factor is adjusted using a non-linear dynamic adjustment strategy based on the cosine function, enhancing global search capabilities in the early stages of iteration and local search capabilities in the later stages; simultaneously, a non-linear inertial weight is employed to modulate the position

 doi:10.20944/preprints202603.2132.v1

update mechanism of individuals, further enhancing the algorithm's local optimisation accuracy and convergence stability in the later stages of iteration. The improved strategies are detailed below.

3.1. Basic Whale Optimization Algorithm (WOA)

The Whale Optimization Algorithm (WOA) is an emerging class of intelligent optimisation methods. By simulating the hunting behaviour of humpback whale pods in the wild, the algorithm comprises three key stages: encircling prey, bubble-net fishing, and searching for prey, as illustrated in Figure 4. In the WOA, the position of each humpback whale represents a potential solution; by continuously updating the whales' positions within the solution space, the algorithm ultimately arrives at the global optimum.

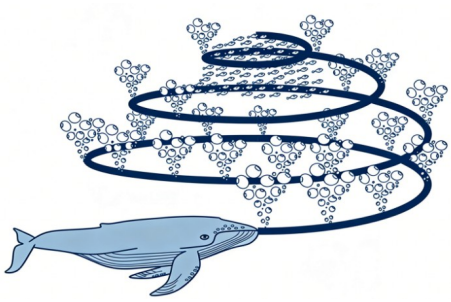


Figure 4. Schematic diagram of cost smoothing.

1) Encircling Prey.In the initial optimization phase, the global optimal solution remains unknown, and population individuals gradually narrow the search range by approaching the current best individual. Let N denote the population size, d the search space dimensionality, the current best individual position be $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_d^*)$, and the position of the i-th individual be $\mathbf{x}_i = (x_{1,i}, x_{2,i}, \ldots, x_{d,i})$. Then, the position update rule for the k-th dimensional component is formulated as:

$$x_{k,i+1} = x_k^* - AD_k \tag{10}$$

$$D_k = \left| cx_k^* - x_{k,i} \right| \tag{11}$$

Where: $x_{k,i+1}$ denotes the position component in the kth dimension after the $i+1$th iteration; D_k denotes the distance between the current individual and the optimal individual in the kth dimension; $|\cdot|$ denotes the absolute value operation; the coefficients A and c are defined as:

$$A = 2ar_1 - a \tag{12}$$

$$c = 2r_2 \tag{13}$$

The convergence factor a decreases linearly from 2 to 0 as the number of iterations increases; A takes values in the interval $[-a, a]$; r_1 and r_2 are random numbers in the interval $[0, 1]$, that is

$$a = 2 - \frac{2t}{T_{\max}} \tag{14}$$

t is the current iteration number, and $T_{\max}$ is the maximum number of iterations.

2) Bubble-net hunting. This stage simulates the behaviour of humpback whales as they deploy a bubble net and spiral upwards to close in on their prey, using a logarithmic spiral trajectory to perform a detailed local search. The position update formula for the kth-dimensional component is:

$$x_{k,i+1} = x_k^* - D_{k1} e^{bl} \cos(2\pi l) \tag{15}$$

$$D_{k1} = \left| x_k^* - x_k^i \right| \tag{16}$$

 doi:10.20944/preprints202603.2132.v1

Where: b is a constant that controls the shape of the logarithmic spiral, typically taken as a fixed value (e.g. $b = 1$); l is a random number uniformly distributed in the interval $[-1, 1]$, used to adjust the step size and direction of the spiral; D_{k1} is the distance component between the current individual and the optimal individual in the kth dimension.To strike a balance between exploration and exploitation, the algorithm uses a random probability p to decide whether to execute an encirclement or a bubble-net attack strategy; the position update rule is standardised as follows:

$$X_{k,i+1} = \begin{cases} x_k^* - AD_k & p < 0.5 \\ x_k^* - D_{k1} e^{bl} \cos(2\pi l) & p \geq 0.5 \end{cases} \tag{17}$$

where: p is a random number in the interval $(0, 1)$, and $A \in [-1, 1]$.

3) Search for prey. When $|A| \geq 1$, the algorithm ceases to converge towards the current optimal individual and instead selects an individual at random from the population to serve as a guide, in order to avoid getting stuck in a local optimum. Let the position of the randomly selected individual be $\boldsymbol{x}_{\text{rand}} = (x_{1,\text{rand}}, x_{2,\text{rand}}, \ldots, x_{d,\text{rand}})$, then the position update formula is:

$$x_{k,i+1} = x_{k,\text{rand}} - AD_k \tag{18}$$

$$D_k = \left| cx_{k,\text{rand}} - x_{k,\text{rand}} \right| \tag{19}$$

This mechanism ensures population diversity by expanding the search range and enhancing the algorithm's ability to explore the global search space.

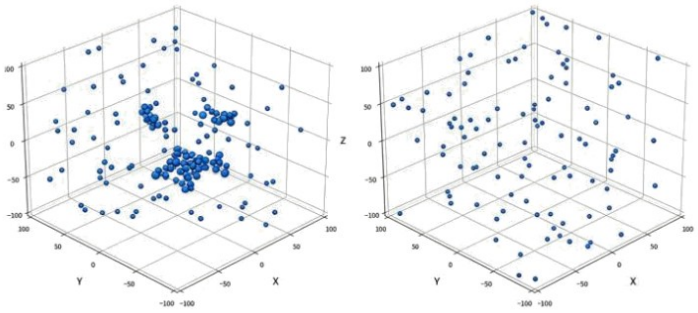
*3.2. R*WOA*

3.2.1. RRT* Initialisation of the Population

In swarm intelligence optimisation algorithms, the quality of the initial population is a key factor influencing the algorithm's convergence and solution accuracy. Traditional random initialisation methods are prone to particle clustering due to sampling randomness, leading to a lack of population diversity and the risk of premature convergence. Given the sensitivity of the Whale Optimization Algorithm (WOA) to initial positions, this paper introduces a Rapid Search Tree (RST) strategy for population initialisation. RRT* generates high-quality initial nodes—which serve as the initial solutions for the Whale Optimization Algorithm (WOA)—by combining random sampling with tree structure expansion and a path optimisation mechanism. Its ability to rapidly traverse the search space prevents individual clustering, enhances the diversity of initial solutions, strengthens the algorithm's global search capability, accelerates convergence, and reduces the risk of local optima traps.

To verify the impact of different initialisation strategies on population distribution characteristics, an initial population of 100 individuals was generated within the three-dimensional search space [-100, 100]^3, as shown in Figure 5: The population generated by traditional random initialisation methods exhibits a pronounced spatial clustering phenomenon, with some individuals overlapping in position and distribution density showing significant regional variations. This non-uniform distribution weakens the diversity level of the initial population, thereby limiting the algorithm's global exploration capability. In contrast, the RRT*-based initialisation strategy achieves more uniform spatial coverage within the three-dimensional space, effectively enhancing the diversity quality of the initial population and providing more favourable initial conditions for the algorithm to escape local optima.

 doi:10.20944/preprints202603.2132.v1



(a) Random initialisation of the population (b) RRT* initialisation of the population

Figure 5. Map of the initial population distribution

The specific steps for initialising a whale population using RRT* are as follows:

1) Generate a target point at random. Within the three-dimensional search space, a target point coordinate G_{rand} is generated at random via uniform sampling, which serves as the guiding direction for the expansion of the path tree. Where:

$$G_{\text{rand}} = (x_{\text{rand}}, y_{\text{rand}}, z_{\text{rand}}) \tag{20}$$

where: $x_{\text{rand}}, y_{\text{rand}}, z_{\text{rand}}$ are coordinates generated randomly in three-dimensional space.

2) Select the nearest node. Within the currently constructed randomly expanded tree structure, determine the tree node G_{near} closest to the random sampling point G_{rand} using a distance metric.

$$G_{\text{near}} = \arg\min_{G \in T} \left\| G G_{\text{rand}} \right\| \tag{21}$$

In the formula: T denotes the set of trees, i.e. the nodes that have already been generated; $\left\| G G_{\text{rand}} \right\|$ denotes the Euclidean distance between the point G and G_{rand}.

3) Expand the node. Extend the nearest node G_{near} by a certain distance σ in the direction of a random point G_{rand}, generating a new node G_{new}:

$$G_{\text{new}} = G_{\text{near}} + \sigma \frac{G_{\text{rand}} - G_{\text{near}}}{\left\| G_{\text{rand}} - G_{\text{near}} \right\|} \tag{22}$$

In the equation: σ is the expansion step size.

4) Collision detection. Check whether the new node G_{new} lies within the search space and does not collide with any obstacles, thereby ensuring the node's validity.

$$\forall o \in O, \quad \left\| G_{\text{new}} - o_{\text{pos}} \right\| > o_{\text{radius}} \tag{23}$$

Where: O denotes the set of obstacles, o_{pos} denotes the centre coordinates of an obstacle, and o_{radius} denotes the safety radius of an obstacle.

5) Select the optimal parent node. Search for all neighbouring nodes within the neighbourhood of the new node G_{new}, calculate the path cost from each neighbouring node to the new node, and select the node with the lowest cost as the parent of the new node.

$$G_{\text{parent}} = \arg\min_{G \in N(G_{\text{new}}, r)} \left(\text{cost}(G) + \left\| G - G_{\text{new}} \right\| \right) \tag{24}$$

 doi:10.20944/preprints202603.2132.v1

6) Path reconnection. Attempt to reconnect nodes within the neighbourhood via the new node G_{new}. If a better path is found, update the parent node and the path cost to optimise the tree structure.

$$\text{if new\_cost}(G) < \text{cost}(G), \text{ then } \begin{cases} G_{\text{parent}}(G) = G_{\text{new}} \\ \text{cost}(G) = \text{new\_cost}(G) \end{cases} \tag{25}$$

7) Generate whale individuals. For each valid new node G_{new} generated, use its position as the initial solution for an individual in the WOA, denoted by X_i. The position of this individual is:

$$X_i = G_{\text{new}} = (x_i, y_i, z_i) \tag{26}$$

For each valid new node G_{new}, its position coordinates (x_i, y_i, z_i) serve as an initial solution for the WOA. Repeat the above steps until the required number of population individuals has been generated.

8) Population assembly. Repeat the above steps until the required number of population individuals has been generated. Ultimately, the whale population Q consists of these nodes generated via RRT*:

$$Q = X_1, X_2, \cdots, X_v \tag{27}$$

In the equation: v is the population size.

3.2.2. Adjusting the Convergence Factor for Non-Linearity

In the standard Whale Optimization Algorithm (WOA), the position-updating behaviour of the humpback whale is governed by the control parameter A, the value of which directly determines the algorithm's search mode: when $|A| \geq 1$, the algorithm performs a global random search with a 50% probability to expand the solution space; when $|A| < 1$, it switches to local exploration, focusing on the current optimal region for refined optimisation. This control mechanism, based on a linear convergence factor, has significant shortcomings when addressing complex multi-peaked optimisation problems: the linearly decreasing convergence factor a results in insufficient time for global exploration in the early stages of the algorithm, making it difficult to fully traverse the solution space; in the later stages, the fixed local exploration step size fails to adapt dynamically to multi-stage search requirements, ultimately leading the algorithm to converge prematurely to a local optimum. To optimise the balance between global exploration and local exploration, this paper proposes a cosine-type piecewise convergence factor a', as shown in Equation (28). Additionally, other non-linear convergence factors are introduced for comparison to validate the effectiveness of the improved strategy.

$$a' = \begin{cases} a_f + (a_o - a_f)\dfrac{1 + \left|\cos\left(\dfrac{(t-1)\pi}{T_{\max} - 1}\right)\right|^{\varepsilon}}{2}, & 0 < t \leq 0.5T_{\max} \\ a_f + (a_o - a_f)\dfrac{1 - \left|\cos\left(\dfrac{(t-1)\pi}{T_{\max} - 1}\right)\right|^{\varepsilon}}{2}, & 0.5T_{\max} < t \leq T_{\max} \end{cases} \tag{28}$$

$$\begin{cases} a = 2 - 2\dfrac{t}{T_{\max}} \\ a_1' = (a_o - a_f)\cos\left(\dfrac{t}{T_{\max}}\right) \\ a_2' = a_o - (a_o - a_f)\left(\dfrac{t}{T_{\max}}\right)^2 \end{cases} \tag{29}$$

In the equation: a_o is the value of the initial convergence factor, set to 2; a_f is the value of the final convergence factor, set to 0; ε is the decay exponent, where $0 < \varepsilon \leq 1$.

The curves showing how different convergence factors vary with the number of iterations are shown in Figure 6. The original linear convergence factor a exhibits a strictly linear decreasing trend; its excessive rate of decline in the early stages leads to insufficient global search coverage, whilst its

 doi:10.20944/preprints202603.2132.v1

fixed step size in the later stages makes it difficult to adapt to the multi-stage requirements of complex optimisation problems. The cosine-type convergence factor a_1' maintains a high value in the early stages of iteration, thereby extending the duration of global search; however, its decay is too slow in the later stages, making it prone to search stagnation. The quadratic-type convergence factor a_2' decays gradually in the early stages and accelerates in the later stages; whilst it ensures exploration in the early stages, the step size becomes too large in the later stages, making it prone to missing the optimal solution and becoming trapped in a local optimum.

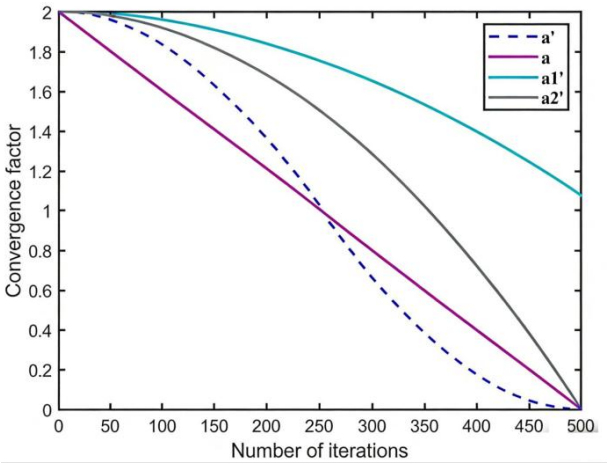


Figure 6. Graphs showing the variation of convergence factor curves for different types

The cosine-type convergence factor a' proposed in this paper decreases gradually in the early stages of iteration, allowing it to maintain a high value for an extended period, thereby effectively prolonging the duration of global search and ensuring that the algorithm thoroughly traverses the solution space; in the later stages of iteration, it gradually decreases, precisely controlling the local search step size, avoiding the omission of the optimal solution, and significantly improving the accuracy of local optimisation. Consequently, replacing the linear convergence factor with a non-linear cosine-type convergence factor effectively balances the requirements of global exploration and local exploitation, prevents premature convergence of the algorithm, and significantly improves the search performance and solution accuracy of WOA.

3.2.3. Non-Linear Inertial Weighting Strategy

During the local exploration phase in the later stages of the standard Whale Optimization Algorithm (WOA), the search range tends to contract excessively, causing the algorithm to become trapped in local optima. This makes it difficult to escape the local extremum region, ultimately resulting in the failure to find the global optimum. Furthermore, when dealing with complex high-dimensional optimisation problems, the algorithm converges rather slowly, often requiring a large number of iterations to obtain a satisfactory solution, thereby limiting its efficiency. To address these shortcomings, this paper introduces a non-linear inertial weighting strategy to improve the basic WOA, aiming to balance global exploration and local exploitation capabilities, thereby enhancing the algorithm's convergence speed and optimisation accuracy.

The inverse tangent function, used as the non-linear adjustment strategy for the inertial weight, is expressed as follows:

$$w = w_{\min} + (w_{\max} - w_{\min}) \arctan\left(\frac{\pi t}{2T_{\max}}\right) \tag{30}$$

The new position update formula is:

$$x_{k,i+1} = w x_k^* - AD_k \tag{31}$$

 doi:10.20944/preprints202603.2132.v1

$$x_{k,i+1} = wx_k^* - D_k e^{bl} \cos(2\pi l) \tag{32}$$

By introducing a non-linear inertial weight into the WOA, which is applied during the position update phases of the enveloping and spiral ascent stages, specifically, in the early stages of the iteration, the adaptive weight factor w is set to a larger value to enhance the algorithm's global optimisation capability and expand the search range within the solution space; As the iteration process continues, the rate of increase of w follows a decreasing trend. This strategy helps enhance the algorithm's ability to escape local optima, thereby improving convergence accuracy and ultimately obtaining higher-quality optimal solutions.

$w_{\max}$ and $w_{\min}$ correspond to the upper and lower bounds of the non-linear adaptive weight factor, respectively; $T_{\max}$ denotes the maximum number of iterations, and t denotes the current iteration number. Following extensive experimentation, this paper sets these parameters to $w_{\max} = 0.82$ and $w_{\min} = 0.01$.Figure 7 shows the dynamic trend of the adaptive weighting factors.The process for the improved Whale Optimisation Algorithm (R*WOA) is shown in Figure 8. The specific steps are as follows:

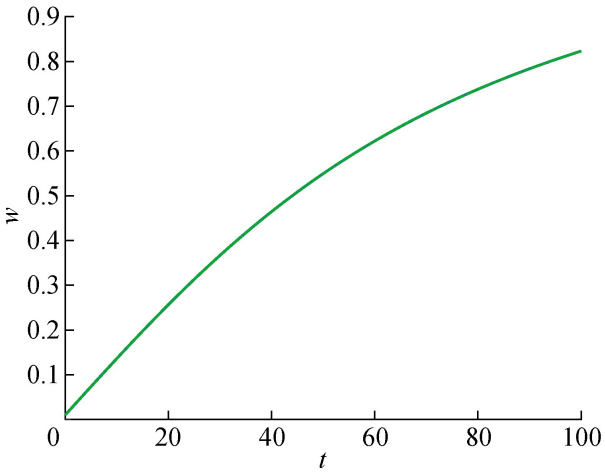


Figure 7. Graph of coefficient changes

 doi:10.20944/preprints202603.2132.v1

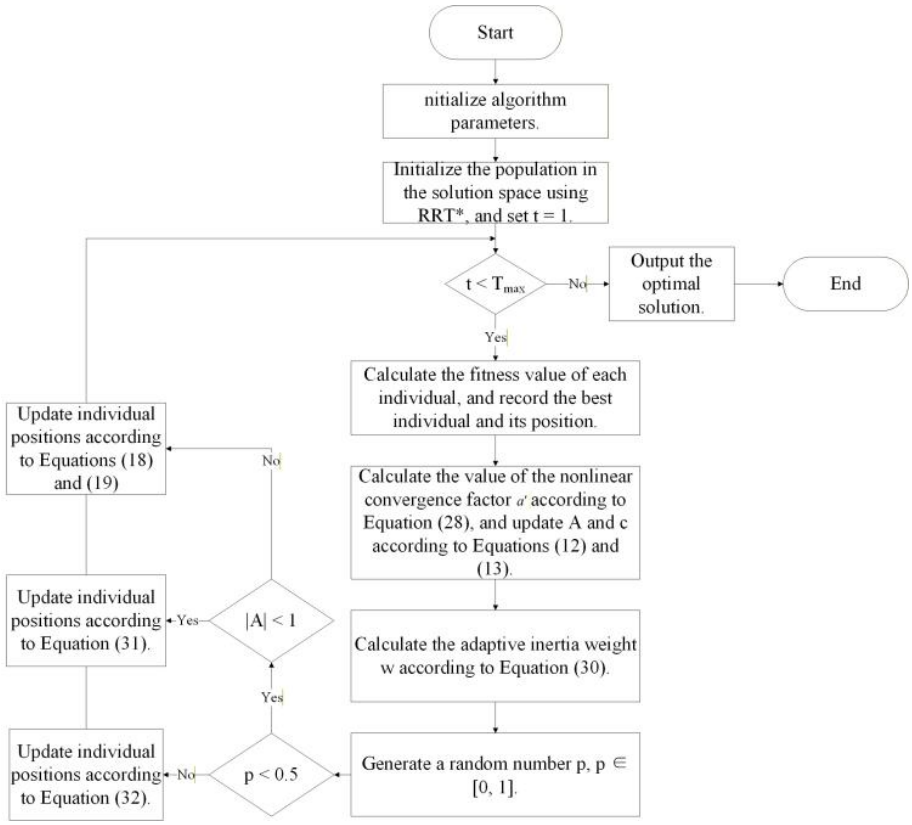


Figure 8. R*WOA Algorithm Flowchart

4. Algorithm Verification

This paper conducts a comparative analysis of the R*WOA algorithm against the WOA, HHO and GA algorithms to verify the performance advantages of the R*WOA algorithm. Furthermore, a UAV flight environment is constructed using a Digital Elevation Model (DEM) to carry out simulation experiments and performance evaluations of the R*WOA algorithm's trajectory planning capabilities.

4.1. Testing of Standard Functions

To evaluate the optimisation performance of the R*WOA algorithm, this study selected 10 benchmark test functions for optimisation testing. Table 1 lists the dimensions and search space ranges of each function, where: $F_1 \sim F_6$ are single-peaked test functions with a unique global optimum, primarily used to assess the algorithm's convergence speed and solution accuracy; $F_7 \sim F_{10}$ are multi-peaked test functions, containing multiple local optima, and are primarily used to test the algorithm's ability to escape local optima and balance global exploration with local exploitation in complex solution spaces.

 doi:10.20944/preprints202603.2132.v1

Table 1. Benchmark functions.

Type	Function	Dimension	Range	Optimum
Single-peak function	$f_1(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
	$f_2(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	$[-30, 30]$	0
	$f_3(x) = \sum_{i=1}^{n} i \cdot x_i^2$	30	$[-10, 10]$	0
	$f_4(x) = \sum_{i=1}^{n} i \cdot x_i^4 + \text{rand}$	30	$[-1.28, 1.28]$	0
	$f_5(x) = \sum_{i=1}^{n} (x_i + 0.5)^2$	30	$[-100, 100]$	0
	$f_6(x) = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} 0.5 i x_i)^2 + (\sum_{i=1}^{n} 0.5 i x_i)^4$	30	$[-5, 10]$	0
multi-peak function	$f_7(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)]$	30	$[-5.12, 5.12]$	0
	$f_8(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	30	$[-32.678, 32.678]$	0
	$f_9(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]$	0
	$f_{10}(x) = 418.9829n - \sum_{i=1}^{n} x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	$[-500, 500]$	0

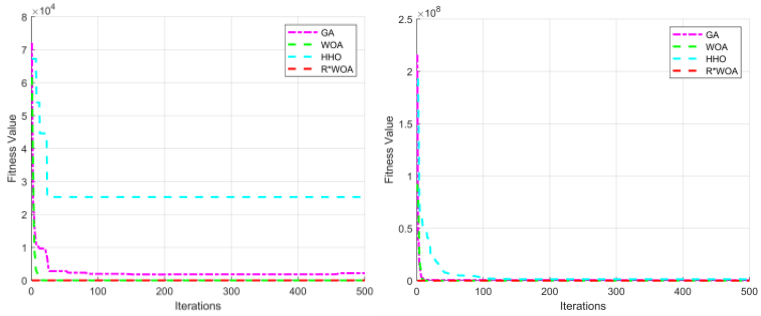
In this study, the dimension of the benchmark test functions was uniformly set to 30 dimensions. The population size for each algorithm is set to 30, and the maximum number of iterations is set to 500. To reduce the randomness of the tests and enhance the persuasiveness of the validation results, each algorithm is run independently 500 times on 10 standard test functions, using the best solution (B), worst solution (W), mean (M) and standard deviation (Std) as evaluation criteria. The resulting data are shown in Table 2. Furthermore, to observe the behaviour of the algorithms during the iterative process, one experiment was randomly selected from each benchmark function, and the resulting fitness curves are shown in Figure 9.

 doi:10.20944/preprints202603.2132.v1

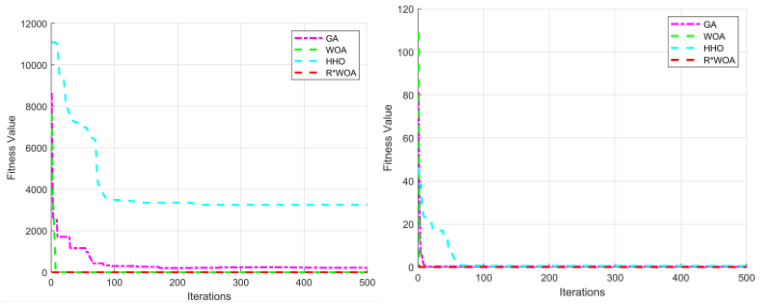
Table 2. Numerical experiments on benchmark functions.

Function	Stat.	GA	WOA	HHO	R*WOA
F1	B	8.26E+02	1.66E-101	3.25E+02	1.06E-161
	W	3.37E+03	2.57E-81	6.57E+04	6.11E-136
	M	1.73E+03	7.53E-84	1.37E+04	1.28E-138
	Std	4.47E+02	1.20E-82	1.19E+04	2.73E-137
F2	B	2.46E+04	4.86E-05	3.21E+03	1.41E-07
	W	5.93E+05	2.87E+01	2.97E+08	1.18E+00
	M	1.63E+05	2.82E+00	2.73E+07	2.13E-02
	Std	8.19E+04	7.54E+00	4.27E+07	6.43E-02
F3	B	9.03E+01	1.18E-100	1.09E+01	2.69E-162
	W	5.65E+02	3.04E-84	1.05E+04	3.79E-137
	M	2.37E+02	1.76E-86	1.93E+03	8.84E-140
	Std	7.06E+01	1.62E-85	1.69E+03	1.71E-138
F4	B	2.95E-02	7.04E-07	2.28E-01	1.24E-07
	W	1.17E+00	8.03E-03	1.29E+02	3.61E-04
	M	2.22E-01	5.89E-04	1.56E+01	5.48E-05
	Std	1.31E-01	8.89E-04	1.94E+01	5.23E-05
F5	B	6.24E+02	0.00E+00	3.68E+02	0.00E+00
	W	3.78E+03	0.00E+00	6.18E+04	0.00E+00
	M	1.69E+03	0.00E+00	1.60E+04	0.00E+00
	Std	4.61E+02	0.00E+00	1.26E+04	0.00E+00
F6	B	2.36E+02	3.07E-08	2.17E+02	1.32E-137
	W	1.08E+10	7.62E+02	2.11E+07	5.87E-98
	M	1.27E+09	2.82E+02	4.75E+04	1.55E-100
	Std	1.65E+09	1.84E+02	9.49E+05	2.67E-99
F7	B	1.26E+02	0.00E+00	7.79E+01	0.00E+00
	W	3.29E+02	0.00E+00	4.41E+02	0.00E+00
	M	2.01E+02	0.00E+00	2.27E+02	0.00E+00
	Std	2.65E+01	0.00E+00	4.94E+01	0.00E+00
F8	B	7.02E+00	4.44E-16	7.63E+00	4.44E-16
	W	1.11E+01	7.55E-15	2.06E+01	4.00E-15
	M	9.02E+00	2.87E-15	1.86E+01	7.35E-16
	Std	7.78E-01	2.18E-15	1.72E+00	9.76E-16
F9	B	7.34E+00	0.00E+00	2.88E+00	0.00E+00
	W	4.20E+01	7.15E-01	6.28E+02	0.00E+00
	M	1.66E+01	3.00E-03	1.43E+02	0.00E+00
	Std	4.21E+00	3.87E-02	1.21E+02	0.00E+00
F10	B	1.05E+04	3.82E-04	5.45E+02	3.83E-04
	W	1.27E+04	4.32E+03	1.00E+04	5.91E+02
	M	1.22E+04	1.59E+02	6.29E+03	1.46E+00
	Std	2.10E+02	4.45E+02	1.36E+03	2.64E+01

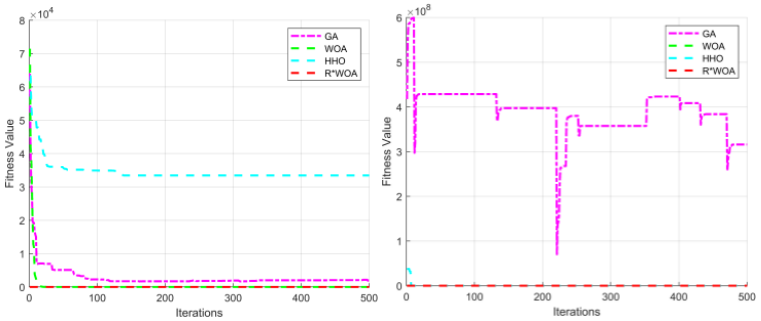
 doi:10.20944/preprints202603.2132.v1



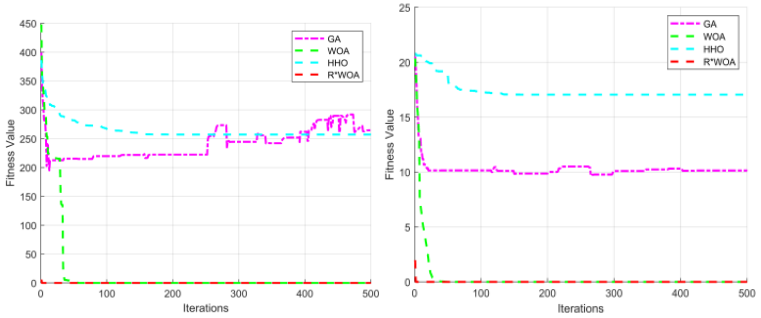
(**a**) **F**1 function fitness curve (**b**) **F**2 function fitness curve



(**c**) **F**3 function fitness curve (**d**) **F**4 function fitness curve



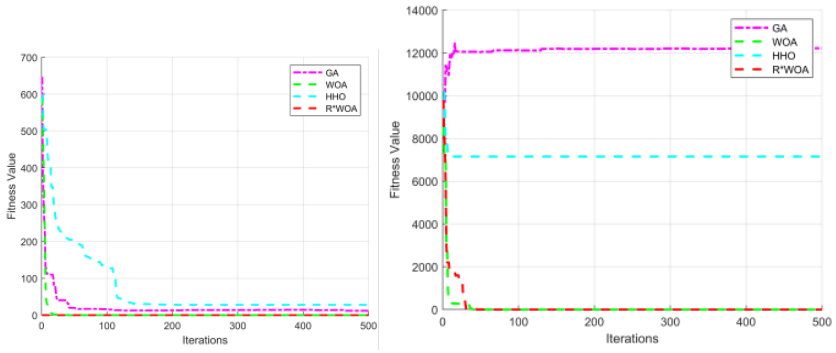
(**e**) **F**5 function fitness curve (**f**) **F**6 function fitness curve



(**g**) **F**7 function fitness curve (**h**) **F**8 function fitness curve

Figure 9. *Cont.*

 doi:10.20944/preprints202603.2132.v1



(**i**) **F**9 function fitness curve (**j**) **F**10 function fitness curve

Figure 9. Fitness curves based on reference functions.

F1∼F6 are unimodal functions, primarily designed to test the algorithms' ability to develop solutions; F7∼F10 are multimodal functions, which have multiple local optima, making it easy for intelligent algorithms to become trapped in local optima during the solution process; these are primarily designed to assess the algorithms' exploration capabilities. As shown in Table 1, in the single-peaked function tests F1∼F6, R*WOA significantly outperformed GA, WOA and HHO in key metrics such as the optimal value, mean and standard deviation, demonstrating exceptional optimisation capability and stability; In the multi-peaked function tests for F7∼F10, R*WOA similarly achieved optimal or near-optimal mean values for most functions, whilst maintaining good control over the standard deviation, indicating its excellent exploration capability and robustness in complex multi-peaked terrain. In particular, on challenging functions such as F4 and F6, R*WOA's convergence accuracy and stability were markedly superior to those of the comparison algorithms. In the F7 test, R*WOA performed on a par with WOA; however, in F10, R*WOA achieved a better mean value than WOA and HHO, with a smaller standard deviation, resulting in superior overall performance. Based on the numerical comparison results between R*WOA and the other three algorithms, it can be seen that R*WOA demonstrates significant performance advantages on both single-peak and multi-peak functions. R*WOA ranks highest overall in the standard test functions, possessing superior convergence accuracy and global optimisation capabilities, and demonstrating strong competitiveness.

Furthermore, in order to observe how the algorithm behaves during the iterative process, a single experiment was randomly selected from the benchmark test functions, and the resulting fitness curve is shown in Figure 9. As can be seen from Figure 9, the comparison algorithm suffers from issues such as poor initialisation quality and an imperfect search strategy during the optimisation process. This results in significant fluctuations in its fitness curve and relatively limited overall convergence performance, specifically manifested in insufficient dispersion of the initial solution set, slow convergence speed, low convergence accuracy, and a tendency to get stuck in local optima. In contrast, the R*WOA algorithm proposed in this paper employs an effective diversity preservation mechanism during the initialisation phase, whilst introducing enhanced global exploration and local exploitation strategies during the iterative process. Consequently, it is able to approach the global optimal solution more stably and accurately, whilst maintaining a relatively fast convergence rate.

4.2. Setting up the Experimental Environment

This study utilised Digital Elevation Model (DEM) data, which was discretised to construct a three-dimensional simulation space measuring $1000 \times 900 \times 400\,\mathrm{m}^3$, with the x-axis ranging from $[0, 1000]$ m, the y-axis ranges from $[0, 900]$ m, and the z-axis represents the actual elevation. The UAV's starting coordinates were set to $[200, 100, 150]$ m, the target coordinates to $[800, 800, 150]$ m, and the number of waypoints was set to 10.

Cylindrical obstacle models were superimposed onto the real terrain to generate four sets of reference scenarios with varying densities and complexities, simulating various threat conditions

 doi:10.20944/preprints202603.2132.v1

encountered during actual flight. The specific coordinates of the threat zones are shown in Table 3, and the UAV constraints are set as shown in Table 4. In the experiments, the maximum number of evolutionary generations was set to 200, the population size to 500, and each algorithm was run independently 30 times. By extracting the optimal solution, the worst-case solution, the mean and the standard deviation, the convergence performance and robustness of the algorithms in path planning were quantitatively evaluated. The optimal and worst-case solutions reflect the algorithm's ability to search for extrema, the mean characterises the overall optimisation effect, and the standard deviation is used to measure the algorithm's stability under different disturbance conditions. A comprehensive analysis of these metrics enables a systematic verification of the algorithm's superior convergence and strong robustness in three-dimensional path planning.

Table 3. Coordinates of the danger zone

Map1			**Map2**		
No.	Threat Center	Radius	No.	Threat Center	Radius
1	(200, 300, 150)	70	1	(200, 300, 150)	70
2	(600, 200, 150)	70	2	(600, 200, 150)	70
3	(700, 500, 120)	75	3	(700, 500, 120)	75
4	(420, 450, 150)	90	4	(420, 450, 150)	90
			5	(400, 150, 150)	60
Map3			**Map4**		
No.	Threat Center	Radius	No.	Threat Center	Radius
1	(200, 300, 150)	70	1	(200, 300, 150)	70
2	(600, 200, 150)	70	2	(600, 200, 150)	70
3	(700, 500, 120)	75	3	(700, 500, 120)	75
4	(420, 450, 150)	90	4	(420, 450, 150)	90
5	(400, 150, 150)	60	5	(400, 150, 150)	60
6	(600, 700, 150)	60	6	(600, 700, 150)	60
7			7	(900, 300, 150)	50
8			8	(900, 600, 150)	60

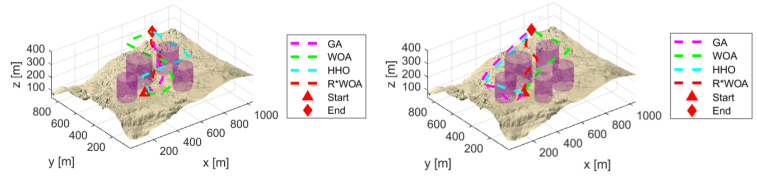
Table 4. Drone restriction settings

Parameter	**Value**
Safety margin distance D	1
Threat escape distance V	10
Maximum altitude $h_{\max}$, minimum altitude $h_{\min}$	200, 100
Maximum steering angle $\alpha_{\max}$, maximum climb angle $\beta_{\max}$	45°, 45°

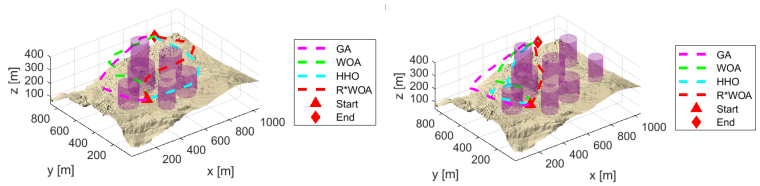
4.3. Analysis of Experimental Results

Figures 10 and 11 illustrate the path planning results of each algorithm in four complex scenarios, from a three-dimensional view and a top-down perspective respectively. The experimental results indicate that all algorithms were able to successfully plan complete, feasible paths from the start to the end point. However, the other comparison algorithms all exhibited varying degrees of convergence to local optima during the optimisation process. This directly resulted in the paths they planned being excessively long, lacking smoothness, and of significantly lower overall quality compared to the algorithm proposed in this paper, highlighting the gap in global optimisation and path quality optimisation capabilities under complex conditions.

 doi:10.20944/preprints202603.2132.v1

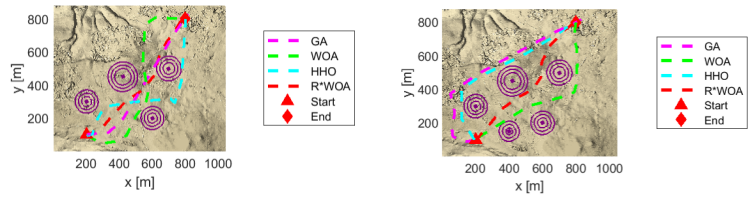


(**k**) **M**ap 1 (**l**) **M**ap 2

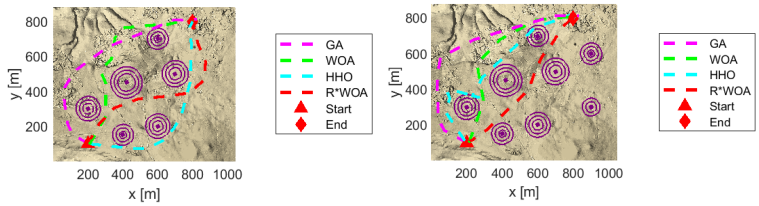


(**m**) **M**ap 3 (**n**) **M**ap 4

Figure 10. 3D views in 4 different scenarios.



(**a**) **M**ap 1 (**b**) **M**ap 2

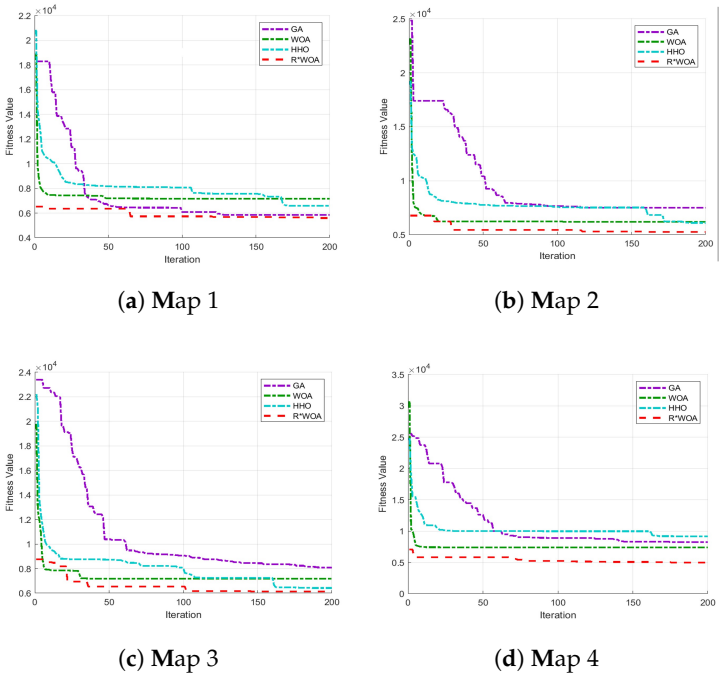


(**c**) **M**ap 3 (**d**) **M**ap 4

Figure 11. Comparison of convergence curves for different algorithms across four scenarios.

As shown in Figure 12, across four different simulation environments, all algorithms included in the comparison exhibited a tendency to converge by the 175th generation. However, the compared algorithms all suffered to varying degrees from issues such as poor initial distribution and slow convergence rates. In contrast, the R*WOA algorithm proposed in this paper demonstrated superior initial population diversity and achieved a faster convergence rate.

 doi:10.20944/preprints202603.2132.v1



(**a**) **M**ap 1 (**b**) **M**ap 2

(**c**) **M**ap 3 (**d**) **M**ap 4

Figure 12. Comparison of convergence curves for different algorithms across four scenarios.

Table 5 presents a comparison of the performance of the four algorithms across multiple test scenarios. The analysis indicates that the R*WOA algorithm proposed in this paper demonstrates a significant improvement in overall performance compared to traditional algorithms. In terms of convergence accuracy, the R*WOA algorithm outperforms the others in both key metrics—the mean and the optimal value—in the vast majority of scenarios. In particular, regarding the optimal value metric, the algorithm obtained the optimal solution in the majority of test scenarios, demonstrating that its improved strategy effectively enhances the algorithm's global search capability, helping it escape local optima and thereby obtain higher-quality solutions. This advantage is consistently demonstrated across multiple scenarios, indicating that the algorithm's improvements possess good general applicability.

Table 5. Experimental results in different scenarios.

Map	Stat.	GA	WOA	HHO	R*WOA
Map1	B	5.44E+03	6.03E+03	5.64E+03	4.96E+03
	W	6.79E+03	8.74E+03	8.51E+03	6.27E+03
	M	6.18E+03	7.24E+03	6.60E+03	5.61E+03
	Std	5.15E+02	8.31E+02	8.81E+02	3.65E+02
Map2	B	6.45E+03	6.46E+03	5.60E+03	4.96E+03
	W	7.08E+03	8.78E+03	8.79E+03	1.15E+04
	M	6.79E+03	7.36E+03	6.52E+03	6.70E+03
	Std	2.08E+02	7.39E+02	9.74E+02	1.80E+03
Map3	B	7.32E+03	5.93E+03	5.19E+03	5.27E+03
	W	1.36E+04	8.41E+03	9.93E+03	1.02E+04
	M	9.59E+03	7.25E+03	7.05E+03	6.59E+03
	Std	2.55E+03	8.18E+02	1.25E+03	1.47E+03
Map4	B	6.98E+03	5.65E+03	6.70E+03	4.98E+03
	W	1.40E+04	8.25E+03	8.79E+03	6.46E+03
	M	9.07E+03	7.11E+03	7.53E+03	5.81E+03
	Std	2.56E+03	8.92E+02	7.12E+02	5.03E+02

 doi:10.20944/preprints202603.2132.v1

In terms of algorithmic stability, R*WOA achieved the smallest standard deviation across multiple scenarios, indicating that the proposed improvement strategy not only enhances search accuracy but also improves the repeatability and reliability of the search process, which is of significant importance for practical engineering applications. In certain scenarios with higher complexity, R*WOA still has room for improvement in metrics such as worst-case performance, reflecting that the algorithm still holds potential for optimisation when handling specific complex constraints. Overall, the experimental results demonstrate that the R*WOA algorithm exhibits clear advantages over the comparison algorithms in terms of convergence accuracy and stability. This validates the effectiveness of the improvement strategy proposed in this paper and provides a more efficient solution for path planning problems.

5. Conclusions

To address the issues of poor convergence performance and susceptibility to local optima in the traditional Whale Optimization Algorithm (WOA) for 3D path planning of unmanned aerial vehicles (UAVs), an improved R*WOA algorithm is proposed. Firstly, the Rapidly Expanding Random Tree* (RRT*) algorithm is introduced to generate a high-quality initial population, thereby enhancing population diversity and global exploration capability. Secondly, a cosine-type non-linear convergence factor is designed to replace the linear strategy, balancing global exploration with local exploitation. Concurrently, a non-linear inertial weighting mechanism is adopted to enhance the accuracy of local optimisation and convergence stability in the later stages. Simulation validation was conducted on 10 standard test functions and four types of 3D scenes constructed based on digital elevation models. The results demonstrate that R*WOA outperforms GA, WOA and HHO in terms of convergence accuracy, stability and path planning quality, exhibiting good reliability and practicality. Future research will focus on real-time path planning in dynamic environments and the advancement of real-world scenario testing.

Author Contributions: methodology, Q.N.; validation, Q.N.; writing—original draft preparation, G.W.; writing—review and editing, X.L.; visualization, X.S. All authors have read and agreed to the published version of the manuscript.

Funding: Funding: This research received no external funding

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Solodov, A.; Williams, A.; Al Hanaei, S.; Goddard, B. Analyzing the threat of unmanned aerial vehicles (UAV) to nuclear facilities. *Security Journal* **2018**, *31*, 305–324.
2. Stuive, L.; Gzara, F. Airspace network design for urban UAV traffic management with congestion. *Transportation Research Part C: Emerging Technologies* **2024**, *169*, 104882.
3. Manju, K.; Manjunath, T. The Future of Autonomous Drones in Environmental Monitoring and Disaster Management. *Proceedings* **2025**.
4. Ali, H.; Xiong, G.; Haider, M.H.; Tamir, T.S.; Dong, X.; Shen, Z. Feature selection-based decision model for UAV path planning on rough terrains. *Expert Systems with Applications* **2023**, *232*, 120713. Elsevier.
5. Hu, G.; He, P.; Salam, M.A.; Wei, G. A novel framework for 4D UAV swarm path planning. *Applied Mathematical Modelling* **2025**, Article 116383.
6. Yang, L.; Tan, L.; Zhang, F. UAV 3D trajectory planning based on improved A* algorithm and differential evolution. *J. Netw. Intell.* **2023**, *8*, 1150–1163.
7. Wang, P.; Mutahira, H.; Kim, J.; Muhammad, M.S. ABA*–adaptive bidirectional A* algorithm for aerial robot path planning. *IEEE Access* **2023**, *11*, 103521–103529.
8. Xu, X.; Zeng, J.; Zhao, Y.; Lü, X. Research on global path planning algorithm for mobile robots based on improved A*. *Expert Systems with Applications* **2024**, *243*, 122922.
9. Yingqi, X.; Wei, S.; Wen, Z.; Jingqiao, L.; Qinhui, L.; Han, S. A real-time dynamic path planning method combining artificial potential field method and biased target RRT algorithm. In Proceedings of the Journal of Physics: Conference Series, Bristol, UK, 1 January 2021; Volume 1905, pp. 012015.

 doi:10.20944/preprints202603.2132.v1

10. Sheng, H.; Zhang, J.; Bai, T.; Wang, D. New multi-UAV formation keeping method based on improved artificial potential field. *Chin. J. Aeronaut.* **2023**, *36*, 249–270.
11. Suo, Y.; Chen, X.; Yue, J.; Yang, S.; Claramunt, C. An improved artificial potential field method for ship path planning based on artificial potential field—mined customary navigation routes. *J. Mar. Sci. Eng.* **2024**, *12*, 731.
12. Karaman, S.; Walter, M.R.; Perez, A.; Frazzoli, E.; Teller, S. Anytime motion planning using the RRT. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1478–1483.
13. Wang, B.; Ju, D.; Xu, F.; Feng, C. Bi-RRT*: an improved bidirectional RRT* path planner for robot in two-dimensional space. *IEEJ Trans. Electr. Electron. Eng.* **2023**, *18*, 1639–1652.
14. Wang, H.; Zhou, X.; Li, J.; Yang, Z.; Cao, L. Improved RRT* algorithm for disinfecting robot path planning. *Sensors* **2024**, *24*, 1520.
15. Yuan, J.; Liu, Z.; Lian, Y.; Chen, L.; An, Q.; Wang, L.; Ma, B. Global optimization of UAV area coverage path planning based on good point set and genetic algorithm. *Aerospace* **2022**, *9*, 86.
16. Roberge, V.; Tarbouchi, M.; Labonté, G. Fast genetic algorithm path planner for fixed-wing military UAV using GPU. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *54*, 2105–2117.
17. Yang, W.; Xia, K.; Fan, S.; Wang, L.; Li, T.; Zhang, J.; Feng, Y. A multi-strategy whale optimization algorithm and its application. *Eng. Appl. Artif. Intell.* **2022**, *108*, 104558.
18. Deng, H.; Liu, L.; Fang, J.; Qu, B.; Huang, Q. A novel improved whale optimization algorithm for optimization problems with multi-strategy and hybrid algorithm. *Mathematics and Computers in Simulation* **2023**, *205*, 794–817.
19. Yang, Q.; Liu, J.; Wu, Z.; He, S. A fusion algorithm based on whale and grey wolf optimization algorithm for solving real-world optimization problems. *Applied Soft Computing* **2023**, *146*, 110701.
20. Cao, Y.; Sun, Z. Multi-strategy optimization of HHO algorithm for path planning of warehouse robots. In Proceedings of the 2023 China Automation Congress (CAC); pp. 735–740.
21. Cai, C.; Jia, C.; Nie, Y.; Zhang, J.; Li, L. A path planning method using modified Harris hawks optimization algorithm for mobile robots. *PeerJ Comput. Sci.* **2023**, *9*, e1473.
22. Hussien, A.G.; Amin, M. A self-adaptive Harris hawks optimization algorithm with opposition-based learning and chaotic local search strategy for global optimization and feature selection. *Int. J. Mach. Learn. Cybern.* **2022**, *13*, No. 2, 309–336.
23. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Advances in Engineering Software* **2016**, *95*, 51–67.
24. Yu, B.; Fan, S.; Cui, W.; Xia, K.; Wang, L. A multi-UAV cooperative mission planning method based on SA-WOA algorithm for three-dimensional space atmospheric environment detection. *Robotica* **2024**, *42*, No. 7, 2243–2280.
25. Deng, X.; Wang, Y. Application of deep reinforcement learning-enhanced whale optimization algorithm in 3D UAV path planning. *Arabian Journal for Science and Engineering* **2026**, 1–16.
26. Liu, Z.; Li, S.; Xu, H. An improved whale migration optimization algorithm for cooperative UAV 3D path planning. *Biomimetics* **2025**, *10*, No. 10, 655.