DOI: 10.19884/j.1672-5220.202312007

A Modified Self-Adaptive Sparrow Search Algorithm for Robust Multi-UAV Path Planning

SUN Zhiyuan[1], SHEN Bo[1,2*], PAN Anqi[1], XUE Jiankai[1], MA Yuhang[1]
1. College of Information Science and Technology, Donghua University, Shanghai 201620, China
2. Engineering Research Center of Digitized Textile & Fashion Technology, Ministry of Education, Donghua University, Shanghai 201620, China

Abstract: With the advancement of technology, the collaboration of multiple unmanned aerial vehicles (multi-UAVs) is a general trend, both in military and civilian domains. Path planning is a crucial step for multi-UAV mission execution, it is a nonlinear problem with constraints. Traditional optimization algorithms have difficulty in finding the optimal solution that minimizes the cost function under various constraints. At the same time, robustness should be taken into account to ensure the reliable and safe operation of the UAVs. In this paper, a self-adaptive sparrow search algorithm (SSA), denoted as DRSSA, is presented. During optimization, a dynamic population strategy is used to allocate the searching effort between exploration and exploitation; a *t*-distribution perturbation coefficient is proposed to adaptively adjust the exploration range; a random learning strategy is used to help the algorithm from falling into the vicinity of the origin and local optimums. The convergence of DRSSA is tested by 29 test functions from the Institute of Electrical and Electronics Engineers (IEEE) Congress on Evolutionary Computation (CEC) 2017 benchmark suite. Furthermore, a stochastic optimization strategy is introduced to enhance safety in the path by accounting for potential perturbations. Two sets of simulation experiments on multi-UAV path planning in three-dimensional environments demonstrate that the algorithm exhibits strong optimization capabilities and robustness in dealing with uncertain situations.

Key words: multiple unmanned aerial vehicle (multi-UAV); path planning; sparrow search algorithm (SSA); stochastic optimization
CLC number: TP18 **Document code**: A
Article ID: 1672-5220(2024)06-0630-14

Open Science Identity (OSID)

0 Introduction

Unmanned aerial vehicles (UAVs) are widely used in military and civil fields[1-2]. Path planning is one of the key factors to ensure the smooth accomplishment of the UAV's mission. UAV path planning refers to the determination of a collision-free trajectory from the start to the end point that is suitable for the current scenario, taking into account various constraints and factors such as total length, environment, data, threat information and energy consumption[3]. As time goes by, the tasks performed by UAVs become more and more complicated. It is noticeable that a single UAV is gradually unable to meet the requirements of some tasks, thus making UAV collaboration a mainstream trend[4].

The commonly used multi-UAV path planning methods are conventional algorithms, graphic methods, heuristic algorithms, swarm intelligence algorithms and so on. Among them, the swarm intelligence algorithms are a new type of heuristic algorithms inspired by the group behavior of various organisms in nature. In the swarm intelligence algorithms, individuals share heuristic information mutually, exhibiting intelligent behavioral characteristics, which finally leads to a global optimum[5]. In contrast to traditional optimization algorithms, swarm intelligence algorithms are simple, parallel and adaptable, so they have received extensive attention in recent years in the field of path planning.

For instance, a genetic algorithm is applied in multi-constraint three-dimensional (3D) UAV path planning[6]. The encoding method is based on azimuthal deviation angles, enabling the determination of the optimal path through geometric computation. A comprehensively improved particle swarm optimization was used for 3D path planning for UAV formation[7]. The convergence speed and the global optimality of the algorithm are improved. Besides, the research applied the grey wolf optimizer to the 3D multi-UAV path planning[8]. After that, the authors improved the grey wolf optimizer for more efficient cooperative path planning for multi-UAVs in 3D complex environments by improving the initialization of the population, the update of individual positions and the decay factor[9]. The

Received date: 2023-12-28
Foundation items: National Natural Science Foundation of China (No. 62303108); Fundamental Research Funds for the Central Universities, China (No. CUSF-DH-T-2023065)
* Correspondence should be addressed to SHEN Bo, email: bo.shen@dhu.edu.cn
Citation: SUN Z Y, SHEN B, PAN A Q, et al. A modified self-adaptive sparrow search algorithm for robust multi-UAV path planning[J]. *Journal of Donghua University (English Edition)*, 2024, 41(6): 630-643.

pigeon-inspired optimization algorithm is also respectively used for the distributed flocking obstacle avoidance and cooperative path planning of multi-UAVs[3,10].

Other swarm intelligence algorithms used for path planning include ant colony optimization (ACO)[11-12] and particle swarm optimization (PSO)[13]. ACO algorithm adopts a positive feedback mechanism to facilitate the exchange of information among individuals, ultimately converging toward the optimal path through the collective efforts and collaboration of individual ants. However, ACO algorithm has poor local search ability and slow convergence speed, and the optimization performance is greatly affected by the parameter settings. PSO algorithm has a simple mechanism and high efficiency, and is less affected by the dimensionality of the problem. However, PSO algorithm is prone to get trapped in the local optimum, and the search performance is too dependent on the parameter settings. Sparrow search algorithm (SSA) is a novel swarm intelligence optimization algorithm proposed by Xue et al.[14] in 2020, and simulates the predatory mechanism and anti-predatory mechanism that sparrows are equipped within nature. Compared with other intelligent optimization algorithms, SSA has faster convergence speed and stronger optimization performance, which has made it promising for cooperative path planning of multi-UAVs[15].

Nevertheless, SSA also has its shortcomings. In SSA, producers are mainly used for a global search for optimal solutions, but the update mechanism of those producers with no predators around them leads to a small search range. Scroungers are responsible for moving closer to the optimal solution as well as increasing population diversity, but those scroungers at the edge of the population have a normally distributed update mechanism, leading individuals to converge toward zero. In real life, however, the optimal solution of most optimization problems is not at zero. In this paper, a modified self-adaptive SSA is proposed to address these challenges, emphasizing exploration in the early iteration and strengthening exploitation in the late iteration. In addition to optimization, it is necessary to tackle the uncertainty caused by positioning errors or modeling errors in path planning. Under these circumstances, in addition to the optimality of the solution, the robustness of the solution should also be taken into account[16]. Therefore, in this paper, a stochastic optimization (SO) strategy is proposed to enhance safety under uncertainty.

The primary contributions of this paper are outlined as follows.

1) A modified self-adaptive SSA is proposed to improve exploration and avoid performance degradation of non-origin optimization.

2) An SO strategy is presented to cope with uncertainties during the path planning tasks.

3) The simulation and the validation are performed on multi-UAV path planning applications for the modified self-adaptive SSA and SO strategy.

The remaining part of the paper is structured as follows. Section 1 builds a cooperative path planning model for multiple UAVs and represents a cost function to be optimized. Section 2 introduces a modified self-adaptive SSA and an SO strategy. In Section 3, the modified self-adaptive SSA has been tested using 29 standard benchmark functions from the Institute of Electrical and Electronics Engineers (IEEE) Congress on Evolutionary Computation (CEC) 2017 benchmark suite to demonstrate the algorithm's superiority. Section 4 describes the specific parameters of the simulation model and conducts simulation experiments with the modified self-adaptive SSA and the SO strategy. Section 5 presents the conclusions.

1 Modeling and Problem Formulation

Suppose that in a military strike mission, n UAVs are required to take off from separate starting points to the same strike point. The optimization objective for this multi-UAV coordination problem is to minimize the costs related to the flight length, the flight altitude, the flight corner and the time synchronization. During the flight of UAVs, some constraints such as the maximum corner of UAVs, the collisions between UAVs, threat zones and no-fly zones must be taken into account. The objectives and constraints will be formulated in this section.

1.1 Flight length cost

The flight length of UAVs is a critical metric in the context of UAV path planning. Shorter paths often perform more efficiently in the execution of tasks. In addition, the fuel consumption of UAVs is directly proportional to the flight length, and a shorter flight length results in lower fuel consumption cost. Assuming that the flight path of a single UAV consists of a line connecting m waypoints between the start and end points, the total flight length f_l of the n UAVs is[17]:

$$f_l = \sum_{j=1}^{n} \sum_{i=1}^{m+1} L_{i,j}, \tag{1}$$

where $L_{i,j}$ denotes the length of the ith flight segment of the jth UAV which consists of line segments between waypoints and start and end points. Figure 1 shows a schematic of a UAV flight path.

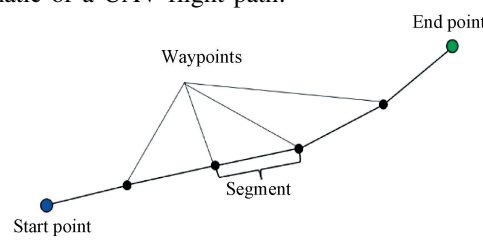

Fig. 1 Schematic of UAV flight path

1.2 Flight altitude cost

During the flight of UAVs, the variation in the flight altitude is a measure of the degree of flight stabilization. A stable flight would have fewer altitude changes. The

cost function of the flight altitude f_h can be represented as

$$f_h = \sum_{j=1}^{n} \sum_{i=1}^{m+1} |\Delta h_{i,j}|, \tag{2}$$

where $|\Delta h_{i,j}|$ denotes the altitude change of the jth UAV in the ith flight segment.

1.3 Flight corner cost and constraint

The ideal flight path for UAVs is a straightforward one. In actual flight, however, the flight path would be more or less curved. The curvature of the path comes from the steering of UAVs, which gives rise to the concept of the corner. Due to the performance limitations of UAVs, the corner magnitudes need to be set within a safe range of values. Besides that, the magnitude of the corners reflects the smoothness of the paths, in other words, smaller corners lead to smoother paths. Therefore, it is required that the supplementary angle $\varphi_{i,j}$ between adjacent trajectory segments (shown in Fig. 2) must not exceed the maximum corner $\varphi_{\max}$ and should be as small as possible, so the flight cost function f_d [16] and flight corner constraint function constraint$_d$ [17] are:

$$f_d = \sum_{j=1}^{n} \sum_{i=1}^{m} \varphi_{i,j}, \tag{3}$$

$$g_{i,j} = \begin{cases} 0, & \varphi_{i,j} < \varphi_{\max}, \\ 1, & \varphi_{i,j} > \varphi_{\max}, \end{cases} \tag{4}$$

$$\text{constraint}_d = \sum_{j=1}^{n} \sum_{i=1}^{m} g_{i,j}, \tag{5}$$

where $\varphi_{i,j}$ denotes the angle between the $(i+1)$th segment and the ith segment of the jth UAV's trajectory; $\varphi_{\max}$ is the maximum value of the corner; $g_{i,j}$ denotes exceeding the cornet limit.

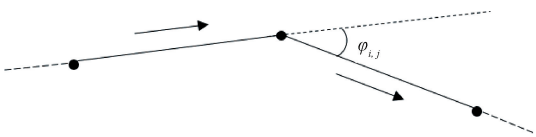

Fig. 2 UAV corner in a schematic diagram

1.4 Time synchronization cost

Given the importance of ensuring that multiple UAVs arrive at the same time, it is vital to implement time synchronization among them. In multi-UAV path planning, each UAV is assumed to have its maximum flight speed $v_{\max}$ and minimum flight speed $v_{\min}$. Based on the respective flight length of each UAV, the theoretical values of its longest flight time and shortest flight time can be calculated. Assuming the flight lengths of UAVs are $L_1, L_2, \cdots, L_j (j = 1, 2, \cdots, n)$, then the longest flight time and shortest flight time of UAVs are $[T_{1\_\min} = L_1/v_{\max}, T_{1\_\max} = L_1/v_{\min}], \cdots, [T_{j\_\min} = L_j/v_{\max}, T_{j\_\max} = L_j/v_{\min}]$. After synthesizing the flight time intervals of all UAVs, the upper and lower bounds of the overlapping time are defined as $[T_a, T_b]$, where

$$T_a = \max(T_{1\_\min}, T_{2\_\min}, \cdots, T_{j\_\min}), \tag{6}$$

$$T_b = \min(T_{1\_\max}, T_{2\_\max}, \cdots, T_{j\_\max}). \tag{7}$$

The larger the overlapping time, the better arrival synchronization among UAVs. Therefore, the time synchronization cost f_c is

$$f_c = 100(T_a - T_b). \tag{8}$$

1.5 Collision avoidance constraint

Collision avoidance between UAVs is a necessary consideration for safety in flight. In this work, it is specified that the distance between the waypoints of the paths generated by different UAVs cannot exceed a certain value. It is specifically expressed as follows[18]:

$$c_{i,j,i_1,j_1} = \begin{cases} 0, & \| x_{i,j} - x_{i_1,j_1} \| > d_{\min}, \\ 0.5, & \text{otherwise}, \end{cases} \tag{9}$$

$$\text{constraint}_c = \sum_{j=1}^{n} \sum_{i=1}^{m} \sum_{\substack{j_1=1 \\ j_1 \neq j}}^{n} \sum_{\substack{i_1=1 \\ i_1 \neq i}}^{m} c_{i,j,i_1,j_1}, \tag{10}$$

where $x_{i,j}$ denotes the position of the jth UAV in the ith flight segment; c denotes the collision penalty; $d_{\min}$ is the shortest distance allowed between UAVs; constraint$_c$ is the total account of the collision situation.

1.6 Threat zone

UAVs executing tasks in complex military environments may be subject to attacks from the enemy, including radar, anti-aircraft artillery and missiles, which are collectively called threats. The above threats are expressed in 3D as a rotating parabolic whose radius is related to the height h at which it is located. The radius r_h at a height h is[9]

$$r_h = \sqrt{(r_{\max}^2 \cdot h)/h_{\max}}, \tag{11}$$

where $h_{\max}$ represents the maximum height; $r_{\max}$ represents the maximum radius at the maximum height. UAVs are required not to pass through the area of the rotating parabolic, or they are extremely vulnerable to attack, which might lead to mission failure. For a more precise determination, a path is considered infeasible when a waypoint below height h or a k-section point of the segment enters the radius r_h, which can be presented as

$$d_{i,j} = \begin{cases} 1, & (x_{i,j} - a)^2 + (y_{i,j} - b)^2 < r_h^2, \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

$$\text{constraint}_t = \sum_{j=1}^{n} \sum_{i=1}^{km} d_{i,j}, \tag{13}$$

where (a, b) is the coordinate of the region center projected on the XOY plane; constraint$_t$ is the constraint function reflecting the extent of UAV penetration into the threat zone.

1.7 No-fly zone

Influenced by the environment, such as strong convective weather, there will be an area where flight is not possible. Once in this area, UAVs may be at risk of

being destroyed. The no-fly zone $e_{i,j}$ is represented as a cylindrical area with radius R and is determined in the same way as the threat area:

$$e_{i,j}=\begin{cases}1, & (x_{i,j}-a)^2+(y_{i,j}-b)^2<R^2 \quad \text{and} \quad h<h_{\max},\\ 0, & \text{otherwise},\end{cases} \tag{14}$$

$$\text{constraint}_e=\sum_{j=1}^{n}\sum_{i=1}^{km}e_{i,j}, \tag{15}$$

where constraint_e is the constraint function of UAV entering the no-fly zone.

1.8 Total cost function

The total cost function comprises both the objective functions to be optimized and the constraints. All objective functions are summed up by the weight ω. The weight can be defined according to the importance and preference of different objectives. Constraints are embedded in the total cost function according to the penalty function method, where an extremely large coefficient η is imposed when the constraint is violated. In summary, the total cost function can be expressed as

$$F_{\min}=\omega_1 f_l+\omega_2 f_h+\omega_3 f_d+\omega_4 f_c+\eta(\text{constraint}_d+\text{constraint}_c+\text{constraint}_t+\text{constraint}_e). \tag{16}$$

2 Modified Self-Adaptive SSA and SO Strategy

This part will first introduce the mathematical model of the basic SSA. Its shortcomings are specifically analyzed according to its updated formulation. Then, a modified self-adaptive SSA and an SO strategy are proposed.

2.1 Basic SSA

In nature, sparrows are strong in foraging and early warning, which is the inspiration for SSA. In SSA, the sparrow individuals with better fitness values are prescribed as scroungers and the remaining individuals as followers, while a certain percentage of the sparrow individuals in the group are selected as guards, and these individuals change their positions when danger is perceived.

The population is initialized at the beginning of the algorithm and the fitness value is evaluated for each individual. Then the individuals are sorted according to the results of the fitness value. The population is categorized into producers and scroungers, and warning mechanisms are introduced by generating guards among the population. The producers, the scroungers and the guards update the positions of sparrow individuals in accordance with their respective rules, which are modeled mathematically as follows.

2.1.1 Producers

$$X_{i,j}^{t+1}=\begin{cases}X_{i,j}^{t}\cdot\exp\left(\dfrac{-i}{\alpha\cdot iter_{\max}}\right), & R_2<ST,\\ X_{i,j}^{t}+Q\cdot L, & R_2\geqslant ST,\end{cases} \tag{17}$$

where $X_{i,j}^{t}$ is the value of the ith individual in dimension j at iteration t; $iter_{\max}$ is the maximum number of iterations; α, R_2, Q, L and ST are predefined control parameters of SSA.

In producer populations, if R_2 is lower than the safety threshold ST, it means that the current individual is in a safe position without any predators around it. At this point, the producer can search for food around the current location. When R_2 is higher than the safety threshold ST, the sparrow has discovered the predator and needs to move elsewhere to find food.

2.1.2 Scroungers

$$X_{i,j}^{t+1}=\begin{cases}Q\cdot\exp\left(\dfrac{X_{\text{worst},j+}^{t}-X_{i,j}^{t}}{i^2}\right), & i>n/2,\\ X_{p,j}^{t+1}+\left|X_{i,j}^{t}-X_{p,j}^{t+1}\right|\cdot A^{+}\cdot L, & \text{otherwise},\end{cases} \tag{18}$$

where p is the index of the optimal position.

When i exceeds half of n, it indicates that those scroungers with lower fitness fail to find food for themselves, making it necessary for them to fly elsewhere for food sources. Those scroungers with higher fitness values move closer to the food source found by the producers.

2.1.3 Guards

$$X_{i,j}^{t+1}=\begin{cases}X_{\text{best}}^{t}+\beta\cdot\left|X_{i,j}^{t}-X_{\text{best},j}^{t}\right|, & f_i>f_g,\\ X_{i,j}^{t}+K\cdot\left(\dfrac{\left|X_{i,j}^{t}-X_{\text{worst},j}^{t}\right|}{(f_i-f_w+\epsilon)}\right), & f_i=f_g,\end{cases} \tag{19}$$

where K and ϵ are predefined constants; f_g is the global best fitness value.

In the population, about 10% to 20% individuals will be aware of the danger. In this phase, sparrow individuals positioned in the middle of the population will move toward the worst individuals, while individuals at the edge of the population will move toward other sparrows to minimize the likelihood of becoming prey.

2.2 Modified self-adaptive SSA

Although SSA exhibits fast convergence speed and simple parameter configuration, falling into local optimum easily is a major drawback of SSA. Additionally, according to Eq. (18), it can be seen that the scroungers with lower fitness are more inclined to fly to the position near the zero point, and the diversity of the population is somewhat insufficient. When facing the non-zero point optimization problem, the performance of SSA is significantly compromised. To address these problems, SSA will be improved in this work. This work presents a modified self-adaptive SSA that mainly enhances both the convergence and the population diversity of the algorithm. The details of the improvements include dynamic population, adaptive t-distribution perturbation operator and random learning mechanism for scroungers. The new algorithm is denoted as DRSSA.

2.2.1 Dynamic population

In basic SSA, the proportion of producers and scroungers is set in the initialization phase of the

algorithm and remains constant throughout the iteration. From the mathematical modeling of producers and scroungers, it can be seen that the producers are designed to improve the algorithm's exploration capabilities and the scroungers are designed to increase the exploitation of the algorithm. Under such circumstances, we could dynamically adjust the proportion of producers in the population so that more producers pursue the global optimal solution in the early stage of the algorithm, and later on it favors the scroungers to exploit near the global optimal solution. Adjustments made to the proportion of producers PD are:

$$PD = PD_{\min} + (PD_{\max} - PD_{\min}) \cdot \cos\left[\frac{\pi}{2} \cdot (iter/iter_{\max})\right]. \tag{20}$$

The percentage of producers PD decreases from $PD_{\max}$ at the beginning to $PD_{\min}$ in the end as the number of iterations $iter$ escalates.

2.2.2 Adaptive t-distribution perturbation operator

It can be seen from Eq. (17) that the next position of those producers who are in a safe position is related to the index of the individual i and the random number α. For a particular individual, its index number i is fixed, so that for the ith individual, its position at the next iteration is only related to the random number α. In addition, $\exp\left(\frac{-i}{\alpha \cdot iter_{\max}}\right)$ is a value between 0 and 1, resulting in a restricted search range for producers. This would appear to be a lack of randomness and exploration. An appropriate modification to this part is required, which can be done by replacing Eq. (17) with

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t} \cdot (1 + t(iter)), & R_2 < ST, \\ X_{i,j}^{t} + Q \cdot L, & R_2 \geqslant ST, \end{cases} \tag{21}$$

where $t(iter)$ denotes a random number that obeys a t-distribution with a degree of freedom $iter$. Based on the visual representation provided in Fig. 3, when the degree of freedom is a small number, the t-distribution curve is low in the middle and high on both sides, and vice versa. As a result, during the early phase of the algorithm, the value of $t(iter)$ is more inclined to fall on either side, which helps to enhance the exploration of the algorithm.

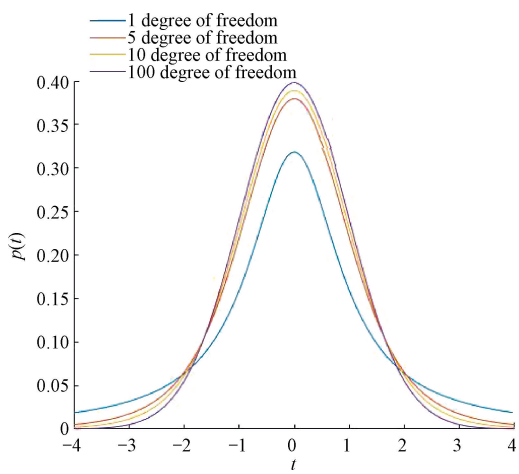

Fig. 3 Characteristics of t-distribution

2.2.3 Random learning mechanism for scroungers

In Eq. (18), the position of the scroungers with lower fitness in the next iteration is almost closer to zero, leading to a decrease in the population diversity. For a general optimization problem, its convergence and exploration are even out of the question. Consequently, changes need to be made to Eq. (18)

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t} + \omega \cdot r_1 \cdot (X_{\text{rand},j}^{t} - X_{i,j}^{t}) + \\ \quad (1 - \omega) \cdot r_2 \cdot (X_{\text{best},j}^{t} - X_{i,j}^{t}), & i > n/2, \\ X_{p,j}^{t+1} + |X_{i,j}^{t} - X_{p,j}^{t+1}| \cdot A^{+} \cdot L, & \text{otherwise}, \end{cases} \tag{22}$$

$$\omega = 1 - \frac{e^{iter/iter_{\max}} - 1}{e - 1}, \tag{23}$$

where $X_{\text{rand},j}^{t}$ denotes a random individual; $X_{\text{best},j}^{t}$ denotes the best-preformed individual; r_1 and r_2 are random numbers between 0 and 1; ω denotes the random learning coefficient. To prevent the algorithm from converging prematurely, a random learning mechanism is introduced. In Eq. (22) and Eq. (23), the random learning coefficient ω decreases nonlinearly with the increasing number of iterations to reach the effect of focusing on exploration in the early stage and on convergence in the later stage. Besides, this mechanism also promotes the diversity of populations, effectively decreasing the chances of getting trapped in a local optimum for the algorithm. A schematic of the random learning mechanism for scroungers is shown in Fig. 4. In Fig. 4, the scrounger individual converges towards the local optimum. By introducing an adaptive random learning mechanism, the scrounger individual learns from other random individuals to keep it from converging towards the local optimum.

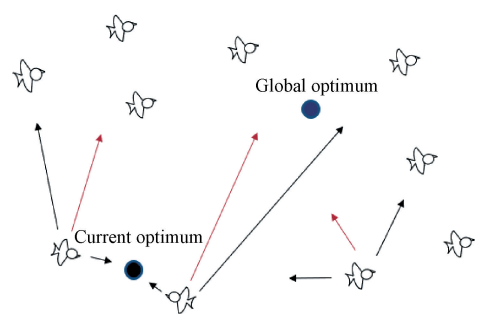

Fig. 4 Schematic of random learning mechanism

Based on the above analysis, the algorithmic representation of the DRSSA procedure can be outlined in Algorithm 1.

Algorithm 1 DRSSA

Input:

$iter_{\max}$: the maximum iterations

$PD_{\min}$: the minimum number of producers at the end

$PD_{\max}$: the maximum number of producers at the beginning

SD : the number of sparrows who perceive the danger

ST : the safety threshold

n: the number of sparrows

Output:

X_g , f_g , $Archi$.

Initialize a population of n sparrows in the search space.

while ($iter < iter_{\max}$)

Rank individuals according to their fitness values, find the current best individual X_{best} and the current worst individual X_{worst} .

$R_2 = rand\ (1)$

Calculate PD by Eq. (20)

for $i = 1: PD$

Update the sparrow's location by Eq. (21);

end for

for $i = PD + 1: n$

Update the sparrow's location by Eq. (22);

end for

for i in $randi\ (n, 1, SD)$

Update the sparrow's location by Eq. (21);

end for

if $iter > iter_{\max}/2$

Store the new locations of all sparrows in solution archive $Archi$;

end if

Update the location of the individual, which depends on whether the individual's new location is better than the previous one;

$iter = iter + 1$;

end while

return X_g, f_g, $Archi$

In Algorithm 1, $rand\ (1)$ denotes a random number; $randi\ (n, 1, SD)$ denotes SD numbers randomly drawn from 1 to n.

2.2.4 SO strategy

While the UAV operates along the planned path, fluctuations in its position should not pose safety risks or deteriorations in performance. Presently, the most of studies on SSA primarily concentrate on achieving the optimal solution, with limited consideration given to ensuring the robustness of the solution. Nevertheless, the real environments are often unstable and prone to disturbances. If the optimal solution is highly sensitive to such disturbance, the optimal solution becomes of no practical significance[19]. Robustness and optimality of a solution are always opposites, which means that it is necessary to achieve robustness at the expense of some optimality. Finding the right balance between robustness and optimality is a crucial concern.

The so-called robustness of the solution refers to the resistance of the perturbation solution. In other words, the objective function does not cause too much fluctuation in the face of perturbations in decision variables. The SO is one of the approaches to finding solutions with good robustness. The goal of SO is to minimize the expectation of the total cost function under uncertainty. In multi-UAV path planning, the objective function is redefined as follows.

$$\begin{aligned} &\min \mathrm{E}[F(X')], \\ &\boldsymbol{X}' = \boldsymbol{X} + \boldsymbol{\delta},\ \boldsymbol{X}' \in \boldsymbol{\Omega},\ \boldsymbol{X} \in \boldsymbol{\Omega}, \end{aligned} \tag{24}$$

where F is the UAV cost function $F_{\min}$ given in Eq. (16); $\boldsymbol{X}$ is a vector of decision variables representing the path; $\boldsymbol{\delta}$ is a perturbation vector that follows a random distribution.

After DRSSA is performed, the SO is conducted, which is denoted as DRSSA-SO. At the end of DRSSA, an optimal solution without considering uncertainty will be found, the solution will be marked as X_g . Meanwhile, all solutions found by the sparrows during the second half of DRSSA will be stored in $Archi$. These solutions are sub-optimal solutions during deterministic optimization, which are likely to have good robustness. During SO, the

perturbations are simulated by Latin hypercube sampling in the neighborhood of one solution, and *num* represents the number of samples.

The details of the SO are as follows. Firstly, *num* samples are randomly generated in the perturbed neighborhood of the optimal solution X_g. The expected fitness value f_{best}^{avg} and the worst fitness value f_{best}^{wor} of X_g can be calculated. Secondly, since solutions with better robustness are generally in the neighborhood of the optimal or sub-optimal solutions, the robustness of solutions in *Archi* will also be evaluated. Thirdly, for each solution in *Archi*, compare its worst fitness f_i^{wor} with f_{best}^{wor}. If the former is higher than the latter, this individual cannot be a robust optimal solution and can be filtered out. The remaining individuals are candidates for the robust optimal solution. Finally, the expected fitness value f_i^{avg} of each remaining solution in *Archi* is calculated. The smaller the f_i^{avg}, the better the robustness of the solution. The individual with the minimum expected fitness value f_{min}^{avg} is the robust optimal solution.

Algorithm 2 illustrates the pseudo-code of the entire SO.

Algorithm 2 SO

Input:
X_g : global optimal solution
Archi : sub-optimal solutions
$[-\delta_d, \delta_d]$: perturbation range
Output:
X_{robust}, f_{robust}
Calculate the expected fitness value f_{best}^{avg} and the worst fitness value f_{best}^{wor} of the global optimal solution X_g within the perturbation range;
while (X_i in *Archi*)
Calculate the expected fitness value f_i^{avg} and the worst fitness value f_i^{wor} of the sub-optimal solution X_i within the perturbation range;
if $f_i^{wor} > f_{best}^{wor}$
continue;
end if
if $f_i^{avg} < f_{min}^{avg}$
Update the minimum expected fitness $f_{min}^{avg} = f_i^{avg}$ and corresponding individual solution $X_{robust} = X_i$;
end if
end while
$f_{robust} = f(X_{robust})$
return X_{robust}, f_{robust}

3 Algorithm Validation and Comparison

In this part, to verify the performance of DRSSA, comparative experiments are conducted based on 29 classical functions from the CEC 2017 benchmark suite. In general, the multi-UAV path planning problem has a higher dimension. The dimensions D available in CEC 2017 benchmark suite are $D=50$ and $D=100$, which match the research of this paper, so CEC 2017 benchmark suite is selected. Details about the CEC 2017 benchmark suite[20] can be accessed. CEC 2017 benchmark suite encompasses a total of 30 functions, due to the instability of F_2 fuction, no performance testing experiment is conducted in this study. In this validation, grey wolf optimizer (GWO), whale optimization algorithm (WOA), SSA and chaos SSA (CSSA)[21] are selected for comparison and the parameter settings are shown in Table 1.

Table 1 Parameter setting

Algorithm	DRSSA	SSA	CSSA	GWO	WOA
Parameter	$ST = 0.8$ $PD_{max} = 0.5$ $PD_{min} = 0.2$ $SD = 0.2$	$ST = 0.8$ $PD = 0.2$ $SD = 0.2$	$ST = 0.8$ $PD = 0.2$ $SD = 0.2$	$b: 2 \rightarrow 0$	$b=1$

To uphold the principle of fairness, the experiments are conducted with a uniform population size N of 100 and the maximum number of iterations $iter_{max}$ set to 5 000. The dimensions of the selected decision variables are $D=50$ and $D=100$, and the search space ranges from $[-100, 100]$. Each algorithm undergoes an independent run of 20

times, and the evaluation indicators are recorded as the average performance across these 20 runs. If an algorithm achieves the best average performance in a function, it is scored 1 point. The scores obtained by each algorithm are summed up to calculate a total score. The results are shown in Tables 2 and 3.

Table 2 CEC 2017 benchmark suite experimental results in $D=50$ (in mean)

Function	Mean value				
	DRSSA	CSSA	SSA	GWO	WOA
*F*1	**1. 22E+03**	3. 96E+03	2. 93E+03	4. 98E+09	6. 82E+06
*F*3	1. 73E+04	1. 05E+04	**3. 30E+03**	7. 49E+04	7. 01E+04
*F*4	**4. 98E+02**	5. 18E+02	5. 08E+02	8. 44E+02	6. 85E+02
*F*5	**6. 55E+02**	8. 65E+02	8. 78E+02	6. 91E+02	9. 22E+02
*F*6	**6. 08E+02**	6. 53E+02	6. 64E+02	6. 11E+02	6. 77E+02
*F*7	**9. 77E+02**	1. 73E+03	1. 78E+03	1. 00E+03	1. 70E+03
*F*8	**9. 45E+02**	1. 18E+03	1. 19E+03	9. 93E+02	1. 23E+03
*F*9	**2. 81E+03**	1. 20E+04	1. 37E+04	3. 35E+03	2. 02E+04
*F*10	6. 81E+03	7. 63E+03	8. 97E+03	**6. 29E+03**	1. 00E+04
*F*11	**1. 24E+03**	1. 32E+03	1. 31E+03	2. 61E+03	1. 57E+03
*F*12	**6. 26E+05**	2. 49E+06	1. 31E+06	3. 60E+08	1. 69E+08
*F*13	**8. 40E+03**	1. 95E+04	1. 66E+04	9. 34E+07	3. 38E+05
*F*14	**3. 81E+04**	5. 17E+04	4. 66E+04	4. 54E+05	6. 57E+05
*F*15	**8. 46E+03**	1. 54E+04	1. 85E+04	3. 30E+06	8. 35E+04
*F*16	**2. 74E+03**	3. 70E+03	3. 92E+03	2. 81E+03	4. 75E+03
*F*17	2. 67E+03	3. 45E+03	3. 61E+03	**2. 52E+03**	4. 16E+03
*F*18	**1. 12E+05**	2. 04E+05	2. 05E+05	3. 10E+06	5. 11E+06
*F*19	**1. 98E+04**	2. 32E+04	2. 48E+04	3. 91E+06	2. 58E+06
*F*20	**2. 74E+03**	3. 21E+03	3. 75E+03	2. 88E+03	3. 61E+03
*F*21	**2. 43E+03**	2. 61E+03	2. 77E+03	2. 48E+03	2. 88E+03
*F*22	**3. 61E+03**	9. 82E+03	1. 07E+04	7. 97E+03	1. 18E+04
*F*23	2. 92E+03	3. 11E+03	3. 37E+03	**2. 91E+03**	3. 57E+03
*F*24	**3. 08E+03**	3. 27E+03	3. 58E+03	3. 11E+03	3. 71E+03
*F*25	3. 08E+03	**3. 05E+03**	3. 06E+03	3. 35E+03	3. 14E+03
*F*26	6. 52E+03	6. 29E+03	8. 90E+03	**5. 66E+03**	1. 31E+04
*F*27	3. 54E+03	**3. 48E+03**	3. 82E+03	3. 49E+03	4. 12E+03
*F*28	**3. 30E+03**	3. 30E+03	3. 31E+03	3. 96E+03	3. 45E+03
*F*29	**4. 15E+03**	4. 79E+03	5. 31E+03	4. 19E+03	7. 36E+03
*F*30	**9. 16E+05**	1. 10E+06	1. 23E+06	7. 09E+07	8. 37E+07
Score	22	2	1	4	0

Note: E+x represents $\times 10^x$, i. e., 1. 22E+03 represents 1.22×10^3.

Table 3 CEC 2017 benchmark suite experimental results in $D=100$ (in mean)

Function	Mean value				
	DRSSA	CSSA	SSA	GWO	WOA
*F*1	**4. 18E+03**	1. 32E+07	6. 21E+03	2. 76E+10	2. 84E+08
*F*3	2. 38E+05	2. 23E+05	**1. 99E+05**	2. 11E+05	6. 81E+05
*F*4	**6. 45E+02**	7. 45E+02	6. 56E+02	2. 85E+03	1. 20E+03
*F*5	**9. 49E+02**	1. 35E+03	1. 36E+03	1. 03E+03	1. 49E+03
*F*6	**6. 29E+02**	6. 62E+02	6. 64E+02	6. 30E+02	6. 84E+02
*F*7	**1. 78E+03**	3. 21E+03	3. 24E+03	1. 78E+03	3. 30E+03
*F*8	**1. 28E+03**	1. 78E+03	1. 84E+03	1. 35E+03	1. 95E+03
*F*9	**1. 17E+04**	2. 31E+04	2. 32E+04	2. 26E+04	4. 18E+04
*F*10	1. 48E+04	1. 52E+04	1. 69E+04	**1. 45E+04**	2. 04E+04
*F*11	**2. 13E+03**	2. 51E+03	2. 53E+03	4. 03E+04	1. 71E+04
*F*12	**3. 38E+06**	3. 98E+07	1. 10E+07	4. 13E+09	9. 84E+08
*F*13	**7. 71E+03**	2. 47E+04	1. 28E+04	2. 61E+08	7. 34E+05
*F*14	3. 12E+05	3. 10E+05	**2. 59E+05**	4. 30E+06	2. 76E+06
*F*15	**3. 39E+03**	1. 01E+04	1. 10E+04	4. 67E+07	2. 77E+05
*F*16	**4. 99E+03**	5. 87E+03	6. 62E+03	5. 57E+03	1. 10E+04
*F*17	**4. 44E+03**	5. 20E+03	5. 87E+03	4. 47E+03	7. 39E+03
*F*18	**2. 86E+05**	4. 29E+05	3. 88E+05	4. 11E+06	2. 90E+06
*F*19	**4. 23E+03**	5. 14E+03	7. 51E+03	1. 14E+08	2. 33E+07
*F*20	**4. 45E+03**	5. 33E+03	6. 05E+03	4. 37E+03	6. 31E+03
*F*21	**2. 71E+03**	3. 18E+03	3. 75E+03	2. 87E+03	3. 89E+03
*F*22	**1. 74E+04**	1. 88E+04	1. 96E+04	1. 75E+04	2. 51E+04
*F*23	**3. 37E+03**	3. 71E+03	4. 35E+03	3. 41E+03	4. 68E+03
*F*24	4. 03E+03	4. 23E+03	5. 33E+03	**3. 97E+03**	5. 98E+03
*F*25	3. 34E+03	3. 36E+03	**3. 28E+03**	5. 50E+03	3. 80E+03
*F*26	1. 43E+04	1. 44E+04	2. 26E+04	**1. 28E+04**	3. 12E+04
*F*27	3. 81E+03	**3. 58E+03**	4. 07E+03	3. 83E+03	5. 42E+03
*F*28	3. 41E+03	3. 47E+03	**3. 37E+03**	6. 47E+03	4. 00E+03
*F*29	**6. 78E+03**	7. 06E+03	7. 99E+03	7. 41E+03	1. 43E+04
*F*30	**1. 14E+04**	2. 42E+05	2. 59E+04	4. 23E+08	3. 44E+08
Score	21	1	4	3	0

As detailed in Tables 2 and 3, whether the dimension of the decision variable is 50 or 100, DRSSA performs best among all five algorithms. In $D=50$ and $D=100$, DRSSA scores 22 and 21, respectively, much higher than other compared algorithms, ranking the first. GWO and SSA have their advantages in different dimensions. CSSA is slightly better than SSA in many cases, but in general, the optimal performance is still inferior. WOA performs poorly in the CEC 2017 benchmark suite compared to other selected algorithms. By this comparison experiment, it is initially demonstrated that DRSSA is significantly different from other algorithms and performs better than other algorithms, which shows the superiority of DRSSA compared to other algorithms. Experimental results also show that DRSSA performs better than other algorithms in solving problems with higher dimensions.

4 Multi-UAV Path Planning Simulation

The environment for this path planning simulation experiment is a 3D spatial topographic map (200 km × 200 km × 10 km) generated by the function, and the range of the topographic map is also the range of the decision variable. Here, x and y represent the horizontal coordinates, while $Z(x,y)$ denotes the elevation at the

corresponding point (x, y). The path is smoothed using a B-spline curve.

Firstly, a basic terrain is generated with a function:

$$Z_1(x, y) = \sin(y + \lambda_1) + \lambda_2 \sin x + \lambda_3 \cos(\lambda_4 \sqrt{x^2 + y^2}) + \lambda_5 \cos x + \lambda_6 \sin(\lambda_7 \sqrt{x^2 + y^2}) + \lambda_8 \cos y, \quad (25)$$

where λ_1, λ_2, $\cdots$, λ_7, λ_8 are constants that determine the topographies. Changing these constants can obtain different basic scenarios.

Secondly, a peak model is stacked on the basic terrain to simulate mountain obstacles in flight. Similarly, a peak model is modeled with a function, and the specific function can be expressed as

$$Z_2(x,y) = \sum_i h_i \exp\left[- \frac{(x - x_i)^2}{a_i^2} - \frac{(y - y_i)^2}{b_i^2} \right], \quad (26)$$

where h_i denotes the maximum height of the ith peak; (x_i, y_i) is the coordinate position of the peak center; a_i and b_i determine the span of the ith peak in the x- and y-axis directions, respectively.

Thus the flight environment used for simulation consists of basic terrain and peak models. Finally, from Eqs. (25) and (26), the expression for the simulation environment is get:

$$Z(x, y) = \max[Z_1(x, y), Z_2(x, y)]. \quad (27)$$

Based on the modeling of the environment, combined with the constraints and evaluation functions described in Section 2, a complete model for this simulation is provided. The objective of this experiment is to find a suitable path planning scheme that is both optimal and robust as far as possible while satisfying all the constraints.

The simulation experiments were conducted in Matlab 2021b on a 3.40 GHz Intel i7 processor and 8 GB RAM computer. To highlight the convergence of DRSSA and the robustness of the SO strategy, a comparative analysis was first conducted between DRSSA and SSA. Subsequently, further enhancements to the results obtained from DRSSA were made through SO improvements, culminating in a qualitative analysis of the outcomes. In DRSSA and SSA proposed in this paper, the population size n is configured as 100, the maximum number of iterations $iter_{max}$ is 1 000. The safety threshold is defined as $ST = 0.8$, and the proportion of producers is $PD = 0.2$ in SSA. In DRSSA, the proportion of producers gradually decreases from 0.5 to 0.2, which is $PD_{max} = 0.5$ and $PD_{min} = 0.2$, and sparrows aware of the danger makeup 20% of the population.

Two sets of experiments have been conducted, each with different locations of terrain, threat areas and coefficients in the cost function, but the parameters of the algorithm are the same, as described above. The segment equalization factor k is set to 7 in Eqs. (13) and (15), and the minimum collision safety distance d_{min} in Eq. (9) is 3 km. Assume that the minimum velocity of the UAV v_{min} is 100 km/h and the maximum velocity of the UAV v_{max} is 200 km/h. In the SO stage, the perturbation range of the decision variables for each waypoint is $[-2.5, 2.5]$, $[-2.5, 2.5]$ and $[-0.25, 0.25]$ in the x-, y- and z-axis directions, respectively. The perturbations all obey a uniform distribution. The number of solutions within the perturbation range of a solution num is 1 000.

In the first set of experiments, the coefficients of the total cost function in Eq. (16) were set as: $\omega_1 = 0.25$; $\omega_2 = 0.3$; $\omega_3 = 0.15$; $\omega_4 = 0.3$; $\eta = 10^8$. In the experiment, the number of UAVs is 3. The three UAVs start at (50, 180, 4), (20, 70, 3) and (30, 30, 3), respectively, and end at (180, 80, 6) (in km). In the second set, the coefficients of the total cost function F_{min} were set as: $\omega_1 = 0.25$; $\omega_2 = 0.3$; $\omega_3 = 0.15$; $\omega_4 = 0.3$; $\eta = 10^8$. The number of UAVs is also 3. The start points are (105, 15, 4), (25, 80, 3) and (15, 180, 3), respectively. The goal point is (180, 145, 6) (in km). More detailed information on the terrain environment and threat models in each set of experiment is available in Tables 4 and 5.

Table 4 Parameter setting of basic terrain

Set No.	Peak No.	$[\lambda_1, \lambda_2, \cdots, \lambda_7, \lambda_8]$	Peak center	Peak parameter		
				a	b	c
1	1	[1.00, 0.10, 0.07, 2.00, 0.15, 0.09, 2.00, 0.12]	(70, 50)	30	15	6
	2		(62, 125)	30	30	10
	3		(170, 170)	25	25	9
2	1	[1.00, 0.10, 0.07, 2.00, 0.15, 0.09, 2.00, 0.12]	(120, 55)	20	10	6
	2		(60, 55)	25	25	8
	3		(55, 130)	25	25	5
	4		(170, 55)	20	20	4

Table 5 Parameter setting of threat area and no-fly zone

Set No.	Type	Center	Radius	Height
1	Threat area	(135,110)	25	10
	No-fly zone	(140,45)	25	10
2	Threat area	(105,160)	25	10
	No-fly zone	(145,110)	20	10

In the first set, three UAV paths optimized by DRSSA are shown in Fig. 5. These paths navigate around obstacles, minimizing the cost function's value. The comparison results of SSA and DRSSA are plotted in Fig. 6, where the x-axis represents the optimization iterations, and the y-axis represents the convergence performance. A lower value on the y-axis indicates better performance, as it signifies a quicker or more efficient convergence to the optimal solution. It can be seen that DRSSA has better performance compared to SSA. However, as shown in Fig. 5, the path calculated by DRSSA is still too close to the edge of the penalty area. In the case when the decision variables are slightly perturbed, the cost function increases substantially, and further SO is required to obtain robust solutions. After SO, the fitness value of the robust solution obtained by DRSSA is 101.145, which is closely comparable to the fitness value of the non-robust solution obtained by SSA, measured at 100.686. Although the robust solution obtained by DRSSA sacrifices some convergence performance, it offers significantly better robustness. Therefore, the method presented in this paper can be characterized as a balance between optimal performance and robustness, enabling safe navigation under uncertainties.

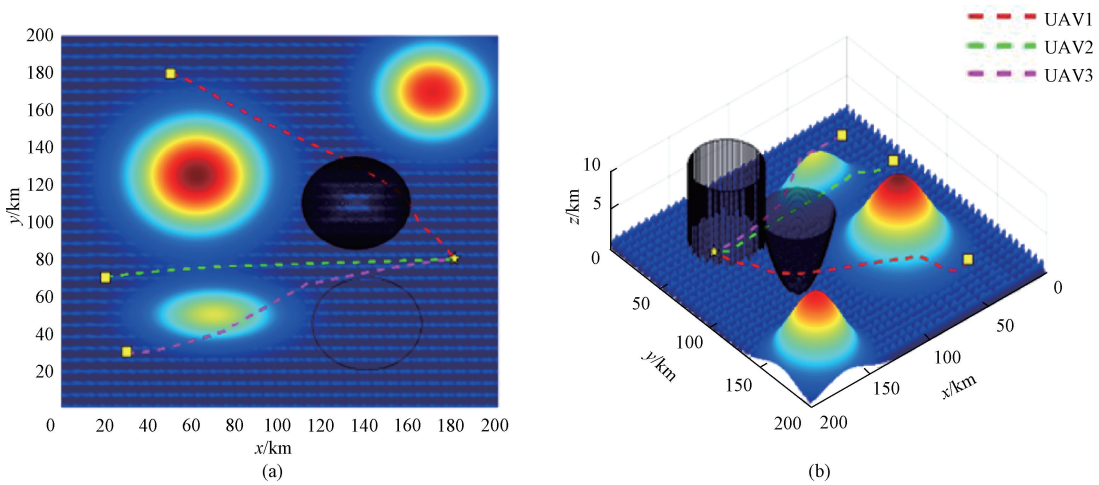

Fig. 5 Simulation result of DRSSA in the first set: (a) 2D top-down view; (b) 3D view

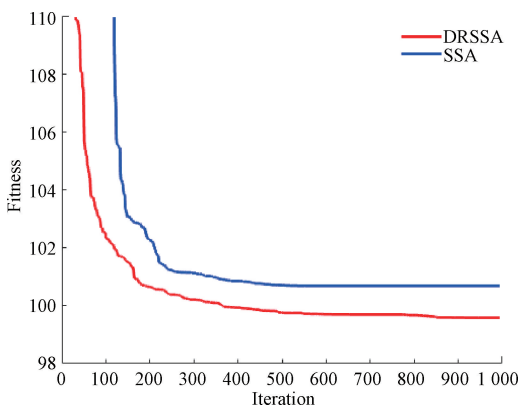

Fig. 6 Comparison of SSA and DRSSA in the first set

In the second set, Fig. 7 shows the results of three UAV path planning by improved DRSSA. Figure 8 shows the comparison of the convergence curves between DRSSA and SSA, which proves the algorithm's improvement for convergence after modification. Similarly, after SO, the fitness value of the robust optimal solution is 138.568, whereas the fitness value of the non-robust solution obtained by SSA is 138.5, demonstrating equivalent convergence performance. Therefore, owing to DRSSA's strong convergence capability, it can maintain comparable convergence performance even after considering robustness, providing safer and more efficient paths for UAVs.

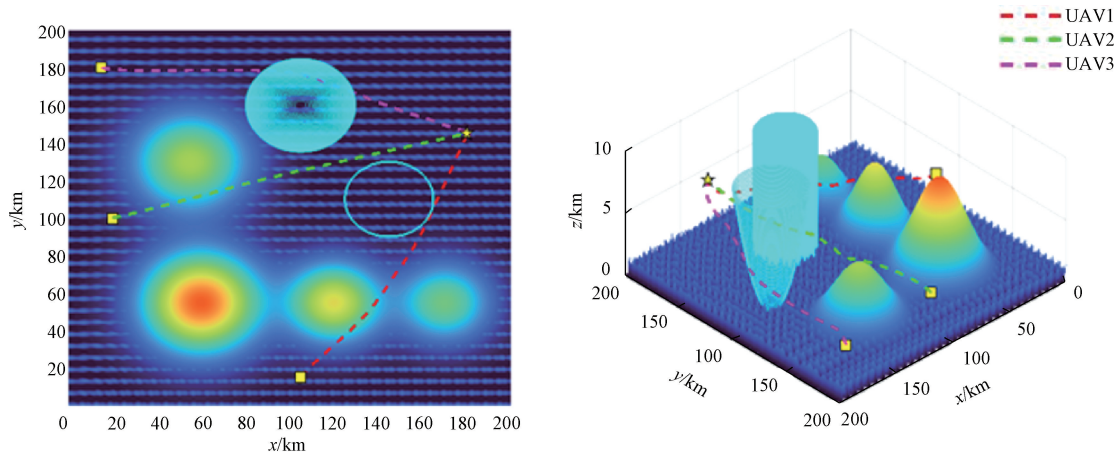

Fig. 7 Simulation result of DRSSA in the second set: (a) 2D top-down view; (b) 3D view

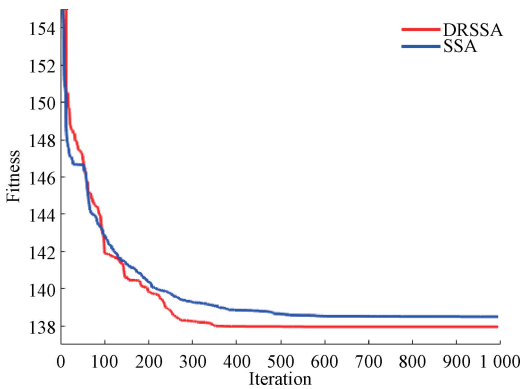

Fig. 8 Comparison of SSA and DRSSA in the second set

Table 6 shows the robustness of solutions before (DRSSA) and after SO (DRSSA-SO) in both sets of experiments. The robustness of a solution is measured by the expected fitness value of 1 000 solutions around it. The expected fitness value of the 1 000 solutions within the perturbation range of the robust optimal solution $f_{\min}^{\text{avg}}$ is much better than the expected fitness value of the 1 000 solutions around the solution obtained by DRSSA $f_{\text{best}}^{\text{avg}}$. It is clear that neither of the solutions around the robust optimal solution violates the constraints.

Since the SO strategy necessarily sacrifices the convergence of the algorithm to improve the robustness of the solution, through the improvement of the convergence of the algorithm, the convergence sacrificed by the SO is compensated, thus achieving the effect of both optimality and robustness. We can also visualize the superiority of the algorithm proposed in this paper through the specific analysis of the two sets of experiments. The results prove that the SO strategy is capable of planning cost-effective and safe paths for multiple UAVs in a 3D environment while maintaining convergence and robustness.

Table 6 Robustness of DRSSA and DRSSA-SO

Set No.	Robustness	
	DRSSA	DRSSA-SO
1	3.61E+08	102.15
2	5.68E+09	141.62

5 Conclusions

This paper presents the proposal of DRSSA, aimed at enhancing exploration and mitigating performance degradation in non-original optimization. We introduce the *t*-distribution perturbation coefficient to producers and a dynamic population strategy to enhance the search capability of the algorithm. A random learning strategy is introduced in the scroungers to achieve strong search capability in the early stage and high convergence in the later stage. DRSSA outperforms the other four algorithms on the 29 test functions of the CEC 2017 benchmark suite, proving a strong optimization search capability. Further more, an SO strategy is proposed for the problem of a large increase in the cost function due to the perturbation of decision variables. Through two sets of UAV simulation experiments, the algorithm proposed in this paper is shown to have a good balance between convergence and robustness in the face of uncertainties.

References

[1] PUENTE-CASTRO A, RIVERO D, PAZOS A, et al. A review of artificial intelligence applied to path planning in UAV swarms [J]. *Neural Computing and Applications*, 2022, 34(1): 153-170.

[2] QIU H X, DUAN H B. A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles [J].

Information Sciences, 2020, 509: 515-529.
[3] AGGARWAL S, KUMAR N. Path planning techniques for unmanned aerial vehicles: a review, solutions, and challenges[J]. *Computer Communications*, 2020, 149: 270-299.
[4] TANG J, DUAN H B, LAO S Y. Swarm intelligence algorithms for multiple unmanned aerial vehicles collaboration: a comprehensive review[J]. *Artificial Intelligence Review*, 2023, 56(5): 4295-4327.
[5] TAN Y, DING K. A survey on GPU-based implementation of swarm intelligence algorithms [J]. *IEEE Transactions on Cybernetics*, 2016, 46(9): 2028-2041.
[6] PENG J L, SUN X X, ZHU F, et al. 3D path planning with multi-constrains based on genetic algorithm [C]//27th Chinese Control Conference. NewYork: IEEE, 2008: 94-97.
[7] SHAO S K, PENG Y, HE C L, et al. Efficient path planning for UAV formation via comprehensively improved particle swarm optimization[J]. *ISA Transactions*, 2020, 97: 415-430.
[8] DEWANGAN R K, SHUKLA A, GODFREY W W. Three dimensional path planning using grey wolf optimizer for UAVs [J]. *Applied Intelligence*, 2019, 49(6): 2201-2217.
[9] XU C, XU M, YIN C J. Optimized multi-UAV cooperative path planning under the complex confrontation environment [J]. *Computer Communications*, 2020, 162: 196-203.
[10] ZHANG D F, DUAN H B. Social-class pigeon-inspired optimization and time stamp segmentation for multi-UAV cooperative path planning [J]. *Neurocomputing*, 2018, 313: 229-246.
[11] COLOMI A, DORIGO M, MANIEZZO V. Distributed optimization by ant colonies [C]. Proceedings of the First European Conference on Artificial Life. Cambridge: MIT Press, 1991: 134-142.
[12] XU H Q, XING H X, LIU Y. Path planning of UAV by combing improved ant colony system and dynamic window algorithm [J]. *Journal of Donghua University (English Edition)*, 2023, 40(6): 676-683.
[13] KENNEDY J, EBERHART R. Particle swarm optimization [C]//Proceedings of ICNN'95 International Conference on Neural Networks. New York: IEEE, 2002: 1942-1948.
[14] XUE J K, SHEN B. A novel swarm intelligence optimization approach: sparrow search algorithm [J]. *Systems Science & Control Engineering*, 2020, 8(1): 22-34.
[15] AWADALLAH M A, AL-BETAR M A, ABU DOUSH I, et al. Recent versions and applications of sparrow search algorithm [J]. *Archives of Computational Methods in Engineering*, 2023, 30(5): 2831-2858.
[16] KAEDI M, AHN C W. Robust optimization using Bayesian optimization algorithm: early detection of non-robust solutions [J]. *Applied Soft Computing*, 2017, 61: 1125-1138.
[17] YAN S Q, LIU W D, YANG P, et al. Optimization of UAV cooperative path planning mathematical model based on personalized multigroup sparrow search algorithm in complex environment[J]. *Journal of Function Spaces*, 2022, 2022: 2521737.
[18] HAN T, TANG A D, ZHOU H, et al. Multiple UAV cooperative path planning based on LASSA method [J]. *Systems Engineering and Electronics*. 2022, 44(1): 233-241. (in Chinese)
[19] HO S L, YANG S Y. A fast robust optimization methodology based on polynomial chaos and evolutionary algorithm for inverse problems[J]. *IEEE Transactions on Magnetics*, 2012, 48(2): 259-262.
[20] AWAD N H, ALI M Z, LIANG J J, et al. Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective bound constrained real-parameter numerical optimization [D]. Singapore: Nanyang Technological University, 2016.
[21] LYU X, MU X D, ZHANG J, et al. Chaos sparrow search optimization algorithm [J]. *Journal of Beijing University of Aeronautics and Astronautics*, 2021, 47(8): 1712-1720. (in Chinese)

一种面向鲁棒多无人机路径规划的自适应麻雀搜索算法

孙至远[1]，沈　波[1,2*]，潘安琪[1]，薛建凯[1]，马宇航[1]
1. 东华大学 信息科学与技术学院，上海 201620
2. 东华大学 数字化纺织服装技术教育部工程研究中心，上海 201620

摘　要：随着技术的进步，多无人机协作已成为军事和民用领域的普遍趋势。路径规划是多无人机执行任务的关键步骤，它是一个带约束的非线性问题。传统的优化算法很难找到在各种约束条件下成本函数最小化的最优解。同时，为确保无人机可靠、安全地运行，还需要考虑鲁棒性。该文提出了一种自适应麻雀搜索算法。在优化过程中，采用动态种群策略来分配搜索，在探索性和开发性之间取得平衡；提出了一种 t 分布扰动系数来自适应调整搜索范围；采用随机优化策略来帮助算法，以避免陷入原点和局部最优附近。自适应麻雀搜索算法的收敛性通过 CEC 基准测试集中的 29 个测试函数进行测试。在算法中进一步引入随机优化策略，通过考虑潜在的扰动来提高路径的安全性。两组关于三维环境中多无人机路径规划的仿真实验表明，该算法在处理不确定情况时表现出很强的优化能力和鲁棒性。

关键词：多无人机；路径规划；麻雀搜索算法；随机优化