Improved SCSO Algorithm Incorporating Disorienting Behavior and Dynamic Adaptive Adjustment Strategy into UAV Path Planning Optimization

Yong-Chen Li, Qiang Qu*

Abstract—**To improve the planning of paths for unmanned aerial vehicles (UAVs) in complex environments, we propose a hybrid improvement strategy based on the Sand Cat Swarm Optimization (SCSO) algorithm. This enhancement, termed Dynamic Sand Cat Swarm Optimization (DPSCSO), aims to optimize the global and local search capabilities of the original SCSO algorithm. Initially, we utilize the Piecewise Linear Chaotic Map (PWLCM) in conjunction with opposition-based learning for population initialization. This approach enhances the uniformity of the initial population distribution, thereby boosting the global search capability of DPSCSO. Additionally, we implement a dynamic adaptive adjustment mechanism during both the search and development phases to broaden the search range and enhance convergence accuracy. Furthermore, we integrate the nocturnal hunting behavior of sand cats into the selection phase of the algorithm to improve its practical search performance. Together, these enhancements form the novel DPSCSO path planning algorithm. To assess the efficacy of DPSCSO, we compare its performance with five other meta-heuristic algorithms—Sand Cat Swarm Optimization (SCSO), Dung Beetle Optimization (DBO), Sine Cosine Algorithm (SCA), Whale Optimization Algorithm (WOA), and Black-winged Kite Algorithm (BKA)—using the CEC2017 benchmark functions. Simulations in complex three-dimensional environments are conducted to evaluate their respective path planning capabilities. Our experimental results and comparative analysis indicate that the proposed DPSCSO algorithm outperforms the other algorithms in path planning tasks.**

Index Terms—**Unmanned Aerial Vehicle, Path Planning, Sand Cat Optimization Algorithm, Lost Behavior, Dynamic Population Strategy.**

I. Introduction

THE field of 3D trajectory planning plays a crucial role in the continuous advancement of UAV technology, especially in military applications. It enables UAVs to quickly find the optimal path to reach the intended target in complex and hazardous environments, significantly enhancing the operational effectiveness of UAVs. However, in UAV trajectory planning systems, numerous constraints, such as terrain characteristics,istics, obstacles, and no-fly zones, lead to the existence of a large number of infeasible paths. To effectively overcome these challenges and achieve efficient trajectory planning, researchers generally adopt strategies such as roadmap trajectory planning, the potential field method, and the population intelligence algorithm [1]. Although traditional UAV trajectory planning algorithms [2], such as Dijkstra's algorithm [3] and A* algorithm, are capable of accomplishing basic path searching tasks in simple environments, their computational effort increases dramatically when dealing with large-scale and complex environments, resulting in a significant decrease in efficiency. For the problem of optimizing the target trajectory under complex constraints [4], traditional population intelligence optimization methods are often prone to fall into local optimal solutions, making it difficult to explore the globally optimal flight path. Therefore, the development of more efficient and intelligent optimization algorithms to meet the complex needs of UAV trajectory planning has become a key issue in current research.

In order to address the challenges associated with trajectory planning in complex environments, international scholars have introduced a wide range of intelligent optimization algorithms, such as the Ant Colony Algorithm, Fireworks Algorithm, and others. Feature models derived from practical problems indicate that intelligent optimization algorithms can generally be categorized into four major types. The first category consists of algorithms that simulate biological reproduction behaviors, promoting population evolution based on the natural principle of survival of the fittest. Examples include the Genetic Algorithm (GA) [5] and the Differential Evolutionary Algorithm (DE) [6]. The second category is based on human behavior and mimics cognitive processes as well as human interactions with the external environment. Examples include the Brainstorming Optimization Algorithm (BSO) [7], Ideology Algorithm (IA) [8], Fireworks Algorithm (FWA) [9], and Tabu Search Algorithm (TS) [10]. The third group encompasses optimization algorithms grounded in mathematical, physical, and chemical principles, such as the Sine-Cosine Optimization Algorithm (SCA) [11], the Black Hole Algorithm (BHA) [12], and the Gravitational Search Algorithm (GSA) [13]. The fourth category is based on swarm intelligence, which solves optimization problems by simulating collective behaviors such as predation, reproduction, and migration in biological groups. Representative algorithms include the Ant Colony Algorithm (ACA) [14], Particle Swarm Optimization (PSO) [15], Artificial Bee Colony Algorithm (ABC) [16], Gray Wolf Optimizer (GWO) [17], Whale Optimization Algorithm (WOA) [18],

Manuscript received June 16, 2025; revised September 5, 2025.

This work was supported by the National Natural Science Foundation of China (71371092), and the Scientific Research Fund Project of Liaoning Provincial Department of Education (2020LNJC04).

Yong-Chen Li is a postgraduate student of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, China (email: 18341523917@163.com).

Qiang Qu is a professor of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, 114051, China (*corresponding author to provide email: quqiang@ustl.edu.cn).

Harris Hawks Optimization (HHO) [19], Sparrow Search Algorithm (SSA) [20], Snake Optimizer (SO) [21], Sand Cat Swarm Optimization (SCSO) [22], and Spider Wasp Optimizer (SWO) [23]. PSO is a population-based, gradient-free optimization algorithm proposed by Kennedy and Eberhart et al. [15]. It simulates the foraging behavior of bird flocks and achieves global search capabilities through adaptive learning and interactions among individuals. The Ant Colony Algorithm (ACA) is inspired by ants' behavior of releasing pheromones to identify optimal paths or solutions. The ABC algorithm simulates the honey collection behavior of honey bees. GWO performs optimization by simulating the hunting behavior of gray wolves. WOA mimics the spiral foraging strategy of humpback whales. HHO simulates the cooperative hunting behavior of Harris's hawks and incorporates Lévy Flight to address complex, high-dimensional problems. SSA, introduced in 2020, models the foraging and anti-predation behaviors of sparrows. SO and SCSO simulate the foraging and mating behaviors of snakes and sand cats, respectively. The SWO algorithm enhances performance through simulation of female wasps' hunting, nesting, and mating behaviors using a set of unique update strategies. In addition, numerous other algorithms have been developed in this field. Recently, several algorithms with superior performance have been proposed[24], [25], [26], [27], [28], [29], [30], [31].

As an emerging swarm intelligence algorithm, the Sand Cat Swarm Optimization (SCSO) algorithm features a unique affine principle and search mechanism. However, it exhibits certain limitations in practical applications, such as a tendency to fall into local optima, slow convergence speed, and limited effectiveness in solving complex high-dimensional problems.

Improving the sand cat swarm optimization algorithm and applying it to UAV trajectory planning will help further explore the algorithm's potential and expand the theoretical foundation of swarm intelligence algorithms. By adjusting algorithmic parameters and enhancing search strategies, we can achieve a deeper understanding of key performance indicators such as convergence and stability, thereby providing valuable references and insights for the development of other optimization algorithms. The core principle of population intelligence algorithms lies in performing optimization searches through continuous iterative processes until predefined stopping criteria are met. During each iteration, individuals update their states according to predefined rules, gradually converging toward the global optimal solution. This study addresses the problem of UAV 3D trajectory planning, with an in-depth investigation into UAV flight environment modeling and the representation of complex threat scenarios. Based on the Sand Cat Swarm Optimization (SCSO) algorithm, we propose an improvement scheme to address the inherent limitations of this swarm intelligence approach, resulting in a novel path planning algorithm—Dynamic Population-based Sand Cat Swarm Optimization (DPSCSO). This work is closely aligned with the practical requirements of UAV trajectory planning, leading to the development of a tailored solution specifically designed for 3D trajectory planning. To evaluate the effectiveness and efficiency of the proposed algorithms and framework, comprehensive tests were conducted across various complex terrain models. Experimental results confirm the algorithm's effectiveness, feasibility, and robustness in UAV trajectory planning applications. The remainder of this study is structured as follows: Section II describes the constraints modeling for UAVs, Section III analyzes the original SCSO algorithm and outlines the improvement strategy for Dynamic Sand cat Population Optimization (DPSCSO). Following this, Section IV tests the performance of the DPSCSO algorithm, while Section V simulates experiments on 3D maps. Finally, Section VI summarizes the results of this paper.

II. MODELING OF UAV CONSTRAINTS

When performing trajectory planning, UAVs must satisfy constraints in numerous dimensions. In order to ensure the successful completion of the flight mission, it is crucial to construct a scientific and reasonable cost assessment model. The accuracy and rationality of the model will directly affect the success or failure of the flight mission. The constraints faced by UAVs during flight are mainly categorized into two main groups: firstly, the constraints of the vehicle itself, which include the cost of path length, the limitations of turn angle, and the limitations of climb and dive angles; and secondly, the constraints of the environmental space, which involves the cost of flight altitude as well as the avoidance of dangers or obstacles. Next, we will focus on the various cost constraints mentioned above.

A. Cost of flight altitude

During UAV flight, its altitude must be limited to a specific range and always remain above the terrain. The specific flight altitude cost function is as follows:

$$h_i = \begin{cases} H_{U,i} - (H_{M,i} + H_{\max}), & H_{U,i} > H_{M,i} + H_{\max} \\ 0, & H_{M,i} + H_{\min} \le H_{U,i} \le H_{M,i} \\ \frac{1}{H_{U,i} - H_{M,i}}, & H_{M,i} + d \le H_{U,i} < H_{M,i} + H_{\min} \\ 100, & H_{U,i} < H_{M,i} + d \end{cases} \tag{1}$$

$$C_1 = \sum_{i=1}^{n+1} h_i \tag{2}$$

Where Hu,i,Hm,j,denote the height values of the UAV and terrain at the ith track point respectively, and the height of the UAV flight is obtained from the z coordinate value zi of the track point; Hmin is the minimum safe height between the UAV and the terrain, which is generally set to be the height position of 5 meters above the terrain; Hmax is the cost value of the UAV flight height, which is generally set to be the height position of 10 meters above the terrain; n denotes the number of waypoints, and the number of waypoints including the endpoints is n+1; the length, width and height of the UAV itself are set to d. The range of heights between Hmin and Hmax above the terrain is called the terrain safety margin. Safety margin refers to a certain amount of cushion or margin that is set aside during the design, operation, or evaluation process to ensure its safe and reliable operation in all expected and unanticipated situations. The function hi is essentially a penalty function and the following can be concluded from Eq.

If the UAV's flight altitude is lower than the terrain altitude, its surrogate value is 100. When the UAV's flight altitude exceeds the terrain altitude and is lower than the minimum value of the terrain safety margin, the surrogate value will be determined based on the actual distance between the UAV and the terrain. This configuration can effectively avoid

a collision between the UAV and the terrain. If the UAV flies within the range of the safety margin of the terrain, the surrogate value is 0. If the UAV flies at a height higher than the maximum value of the terrain safety margin, the surrogate value of the corresponding penalties increases as the UAV flies at a higher height, and as the cost of flight increases. By applying the above method, the UAV can be maintained within a reasonable altitude range that is both safe and economical.

B. Path length consideration

Path length is a crucial metric in UAV trajectory planning tasks. We consider a path as the UAV flight distance from the origin to the destination, the path length. To achieve path planning, the path length is used as a cost function to make the UAV fly a shorter distance and consume less energy. By minimizing this cost function, a better flight path can be found. The expression is given below:

$$I_i = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 - (z_{i+1} - z_i)^2} \quad (3)$$

$$C_2 = \sum_{i=1}^{n+1} I_i \quad (4)$$

Where Ii denotes the length of the i-th segment of the track and (xi,yi,zi) denotes the coordinate value of the track point. Since the number of trajectory points containing start and end points is n+2, n+1 is used to represent the number of trajectory segments.

C. Hazard or obstacle restraint

In order to ensure that the UAV can effectively avoid the hazardous area or obstacle area during flight, in the study of this paper, the flight track points that are within the hazardous area or obstacle area are penalized so that the UAV can maintain flight within the safe area. The corresponding constraint function can be determined as follows:

$$r_i = \begin{cases} 100, & x_l \le x_i \le x_u,\ y_l \le y_i \le y_u,\ z_l \le z_i \le z_u \\ 0, & \text{else} \end{cases} \quad (5)$$

$$C_3 = \sum_{i=1}^{n} r_i \quad (6)$$

Where, xi , xu denote the lower and upper limit values of the hazardous area or obstacle area in the x coordinate, Y_{i}, Y_{u} denote the lower and upper limit values of the hazardous area or obstacle area in the y coordinate, Z_{i},Z_{u} denote the lower and upper limit values of the hazardous area or obstacle area in z coordinate, and n denotes the number of track points. Z_{i},Z_{u} denote the lower and upper limit values of the hazardous or obstacle area in the coordinate z, and n denotes the number of waypoints.

D. Turning angle constraint

Considering the physical performance characteristics of UAVs, their turning angle during steering is restricted and should not be too large. In fact, too large a turning angle is not practical. The limitation of the turning angle is to some extent equivalent to the limitation of the turning radius. Specifically, the smaller the steering angle, the larger the corresponding turning radius; a larger turning radius helps the UAV maintain higher stability during flight. Therefore, during flight, the maximum turning angle should be smaller than the safety preset angle. The current turning angle of the UAV flight is calculated by the following equation. The turn angle constraint function is as follows:

$$\theta_i = \arccos\left(\frac{\mathbf{v}_i \cdot \mathbf{v}_{i+1}}{\|\mathbf{v}_i\| \cdot \|\mathbf{v}_{i+1}\|}\right) \quad (7)$$

$$a_i = \begin{cases} \frac{1}{\beta - \theta_i}, & \theta_i < \beta, \\ 100, & \theta_i \ge \beta. \end{cases} \quad (8)$$

$$C_4 = \sum_{i=1}^{n} a_i \quad (9)$$

Where θ_i is the angle constituted by the current three neighboring track points, the angle constituted by the two neighboring track paths, and β is the safety preset angle. Since the number of trajectory points containing the start and end points is n+2, the number of corners constituted by all the trajectory points is n. As can be seen from the equation, the smaller the current turning angle is, the smaller the penalized generation value is, and such a setting can effectively make the UAV turn more smoothly.

E. Climb and dive angle constraint

During flight, the climb and dive angles of the UAV must be controlled within safe limits. If the climb and dive angles are too large, they may pose a threat to the flight safety of the UAV. The current climb or dive angle of the UAV can be calculated by the following formula: the climb or dive angle constraint function is as follows:

$$\varphi_i = \arctan\left(\frac{|z_{i+1} - z_i|}{\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}}\right) \quad (10)$$

$$u_i = \begin{cases} \frac{1}{\Phi - \varphi_i}, & \varphi_i < \Phi, \\ 100, & \varphi_i \ge \Phi. \end{cases} \quad (11)$$

$$C_5 = \sum_{i=1}^{n+1} u_i \quad (12)$$

Where φ_i is the climb angle constituted by the current two neighboring waypoints, and Φ is the safety preset angle. Since the number of trajectory points containing the start and end points is n+2, the number of all the trajectory points constituting the climb angle is n+1. From Eq. it can be seen that when the climb or dive angle of the UAV is small, the corresponding penalization surrogate value will be reduced, and such a setup will help to ensure that the UAV's climb or dive process is smoother.

F. Integrated cost constraints for trajectory planning

In this paper, the objective function of the integrated cost constraint for trajectory planning is chosen as:

$$F_C = \omega_1 C_1 + \omega_2 C_2 + \omega_3 C_3 + \omega_4 C_4 + \omega_5 C_5 \quad (13)$$

Where F_C denotes the integrated cost of the trajectory planning, $Omega_1$ are the weighting coefficients, and the sum is 1.

III. IMPROVEMENT OF SAND CAT SWARM OPTIMIZATION ALGORITHM

A. Overview of the original sand cat colony optimization algorithm

Sand Cat Swarm Optimization Algorithm is a swarm intelligence optimization algorithm proposed by Farzad Kiani and Amir Seyyesabbasi in 2022, which is mainly inspired by the way sand cats hunt. Sand cats are equipped with a unique hunting mechanism in which they utilize their acute sense of hearing to capture low-frequency noises in order to detect prey activity beneath the ground. Although the rim of the earwheel (outer ear) is similar to that of domestic cats, sand cats have longer middle ear apertures and larger middle ear cavities, which allow them to accurately differentiate between the arrival times of different sounds. Scientific studies have shown that sand cats have the ability to perceive frequencies below 2 kilohertz, and in this frequency band, their hearing sensitivity is 8 decibels higher than that of domestic cats. This superior hearing ability helps sand cats detect noise, track prey, and launch precise attacks based on the location of the prey. The sand cat swarm optimization algorithm mainly simulates two major behavioral patterns of sand cats - search and attack. Based on these characteristics of the sand cat, the optimization algorithm includes two phases: search and attack.

B. Mathematical modeling of optimization algorithms for sand cat swarms

When searching for prey, the sand cat relies mainly on its auditory perception of low-frequency noise. In the specific search for prey, the sensitivity r_{G} decreases linearly from 2 to 0. The mathematical expression is shown in equation (14).

$$\overrightarrow{r_G} = s_M - \left(\frac{2 \times s_M \times t}{T_{max}}\right) \tag{14}$$

Where t is the current iteration number; Tmax is the maximum iteration number; SM is a constant to simulate the auditory characteristics of sand cat. At the same time, the sand cat optimization algorithm introduces the parameter R for the transition control of the two phases of search and attack, and each sand cat updates its own position according to the optimal position and the current position and the sensitivity perception range, and the mathematical expressions are shown in Eqs. (15)-(17).

$$R = 2 \times \overrightarrow{r_G} \times rand(0,1) - \overrightarrow{r_G} \tag{15}$$

$$\vec{r} = \overrightarrow{r_G} \times rand(0,1) \tag{16}$$

$$\overline{Pos}(t+1) = \vec{r} \cdot \left(\overline{Pos}_b(t) - \mathrm{rand}(0,1) \cdot \overline{Pos}_c(t)\right) \tag{17}$$

Where Pos_{b} is the current global optimal position; Pos_{c} is the current position rand(0,1) represents the random number of [0,1]. r is used for the search and attack phases of the operation.

In the attack phase, we assume that the sensitivity perception range of the sand cat constitutes a circular area whose movement direction can be determined by a randomly selected angle (θ) on the circumference. Since the selected random angle ranges from 0 to 360 degrees, its normalized value will be between -1 and 1. In this way, each member of the population is able to move in the search space along a different circumferential direction. The SCSO algorithm uses a roulette selection mechanism to assign a random angle to each sand cat. In this way, the sand cats are able to approach the target location efficiently while avoiding falling into local optimal solutions. The mathematical expression of this stage is shown in Eqs. (18)-(19).

$$\overline{Pos_{md}} = \left| rand(0,1) \cdot \overline{Pos_b(t)} - \overline{Pos_c(t)} \right| \tag{18}$$

$$\overline{Pos\,(t+1)} = \overline{Pos_b\,(t)} - \vec{r} \cdot \overline{Pos_{md}} \cdot cos(\theta) \tag{19}$$

Where Pos_{rnd} denotes the randomized position and ensures that the sand cat involved can be close to the prey. By taking different values of R, the algorithm is made to explore and exploit these two different behaviors, avoiding local optimality to some extent. The position update formula of the final sand cat algorithm is shown in (20).

$$\vec{X}(t+1) = \begin{cases} \overline{Pos_b(t)} - \vec{r} \cdot \overline{Pos_{md}} \cdot \cos(\theta), & |R| \le 1, \\ \vec{r} \cdot \left(\overline{Pos_b(t)} - \mathrm{rand}(0,1) \cdot \overline{Pos_c(t)}\right), & |R| > 1. \end{cases} \tag{20}$$

C. Pseudo-code for the sand cat swarm optimization algorithm

Step	SCSO Algorithm
01	Initializing the population
02	Calculate the fitness value for each individual and determine the optimal individual X_{best}
03	Initialize the parameters r, r_G and R
04	**while** $(t < T)$ **do**
05	**for** each individual sand cat **do**
06	Take a random angle between 0-360 degrees
07	**if** $\lvert R \rvert > 1$ **then**
08	Renewal of sand cat individuals using the formulae
09	**else**
10	Renewal of individual sand cats using the formula
11	**end if**
12	Update the parameters r_2, r_G and R
13	**end for**
14	**end while**

D. Flowchart of sand cat swarm optimization algorithm

E. Improvement of sand cat swarm optimization algorithm

The sand cat swarm optimization algorithm is known for its fewer parameters, simple structure and applicability. However, the algorithm uses roulette to execute the attack behavior, and the blindness of this attack may reduce the efficiency in the optimization search process. In the search behavior, the algorithm uses the overall fitness as the criterion for evaluating the individual's superiority or inferiority, which does not completely and accurately reflect the relationship between the individual's superiority or inferiority in spatial location. In addition, for coordinates that have converged to the optimal position in a certain dimension, the search behavior may cause them to oscillate near the optimal solution, thus reducing the convergence

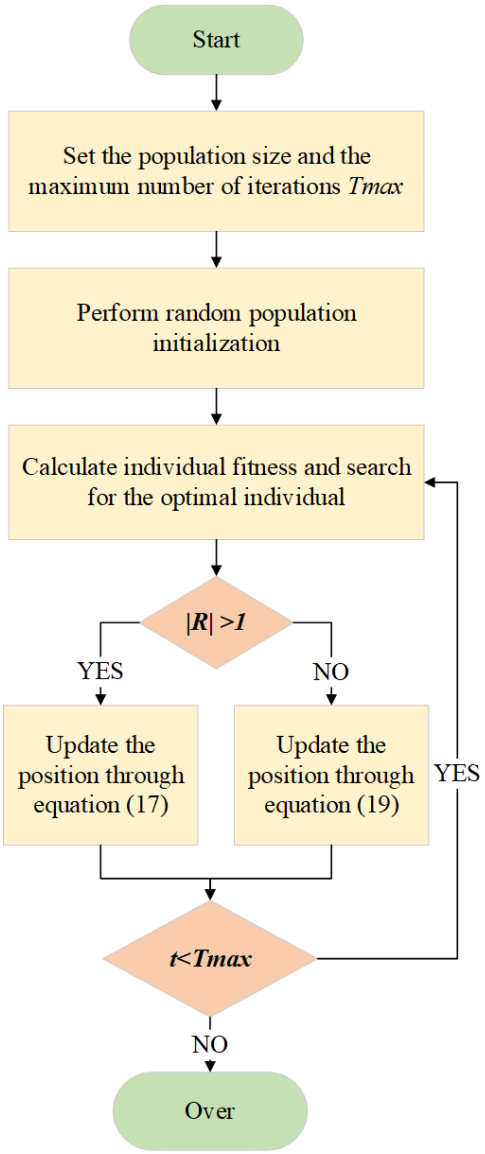


Fig. 1: Flowchart of sand cat swarm optimization algorithm

efficiency. Meanwhile, the original algorithm also suffers from the problems of relying on the initial solution and the decrease of population diversity in the late iteration. In this study, three improvement strategies are proposed with the aim of enhancing the optimization performance of the SCSO algorithm.

F. PWLCM mapping combined with reverse learning for population initialization strategy

In population intelligence algorithms, a high-quality initial population prompts the algorithm to converge faster and effectively improves the algorithm to search globally. However, in the original SCSO, a random approach is used for population initialization, resulting in the population not being evenly distributed in the solution space, which limits the search capability of SCSO. Chaotic mapping, with its own characteristics of randomness and traversal, has been widely used in population diversity improvement for population intelligence algorithms. Currently, mappings such as Tent, Logistic, and Sine are widely used in this field. Compared with other mappings, PWLCM mapping has better

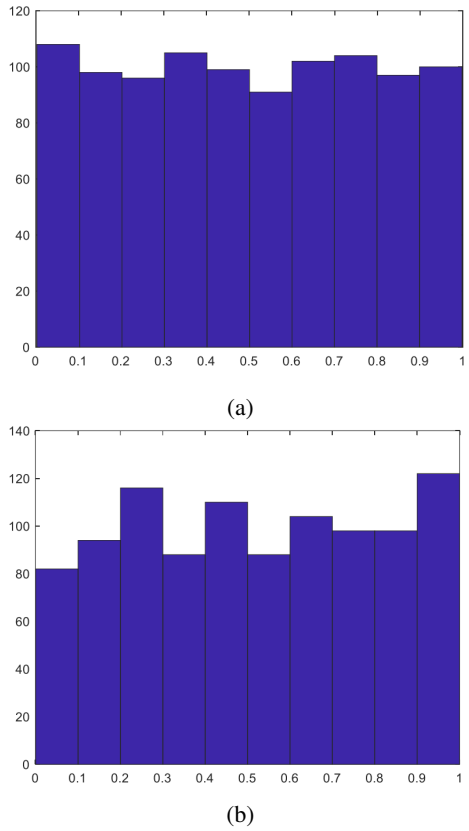


Fig. 2: Histogram of scatter frequency distribution

randomness and traversal due to its segmentation mechanism in the mapping space. Therefore, we select the PWLCM mapping for population initialization, which is defined by the following equation:

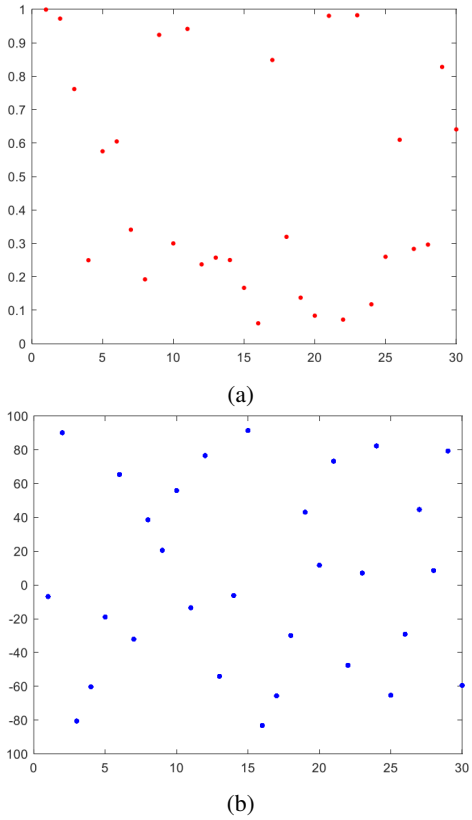
$$f(t+1)=\begin{cases}\frac{f(t)}{p}, & 0\le f(t)<p\\ \frac{f(t)-p}{0.5-p}, & p\le f(t)<0.5\\ \frac{1-p-f(t)}{0.5-p}, & 0.5\le f(t)<1-p\\ \frac{1-f(t)}{p}, & 1-p\le f(t)<1\end{cases} \tag{21}$$

Where p is the control parameter of this mapping, $p\in[0,1]$.f(t) is the value of the generated random iteration, $f(t)\in[0,1]$. Fig.2 illustrates the histograms of the frequency distributions of the 1000 point sets generated by the PWLCM mapping and the randomized method in a two-dimensional space. Observing Fig.2, it can be seen that the number of point sets distributed by PWLCM mapping in each interval is about 100, which is relatively uniformly distributed; in contrast, the distribution of point sets by the randomized method in different intervals varies significantly. This shows that the use of PWLCM mapping can significantly improve the quality of the initial population, which in turn enhances the global search ability of the algorithm. The generation method of the initial population is detailed in Equation (22).

Where x_i(k) is the kth dimension of the ith individual of the population,$k\in[1,dim]$. dim is the optimization problem dimension. lb,ub are the upper and lower bounds of the optimization problem. is the PWLCM mapping perturbation factor.

$$x_i(k)=lb(k)+(ub(k)-lb(k))\cdot f(i) \tag{22}$$

On the basis of utilizing PWLCM chaotic mapping to



(a)

(b)

Fig. 3: Distribution of initialized populations

generate initial solutions, we introduce a reverse learning strategy. This strategy further expands the search space of the sand cat algorithm and improves the quality of the initial population of sand cats. The specific expression of the reverse learning strategy is shown in Eq. (23):

$$X_{new_i} = k \cdot (X_{max} + X_{min}) - X_i \tag{23}$$

A reverse learning strategy is used to perturb the population twice. Fig. 2 shows 30 sand cat individuals generated in 2D space using PWLCM mapping, combining the reverse learning method and stochastic method.

As can be seen from Fig.3, the populations generated using PWLCM mapping combined with reverse learning are more evenly distributed in the solution space without aggregation than those generated in a randomized manner, which is more conducive to improving the quality of the initial populations.

G. Nocturnal disorientation behavior of sand cats

At night, the moon's trajectory exhibits stable regularity and repeatability, and its azimuth changes over time, providing reliable navigational coordinates for nocturnal sand cats. Sand cats are able to orient themselves according to the orientation of the moon and maintain a stable line of travel in the desert. Once lost, sand cats will search by circling around, using their keen sense of smell to capture the scent marks of their companions, the aroma of plants, and the traces of their prey; at the same time, they rely on their sense of hearing to perceive the sound of the wind, the sound of the winging of insects, and the sound of the flow of water sources, so as to achieve the reidentification and localization of the direction. Therefore, in this paper, the model of lost

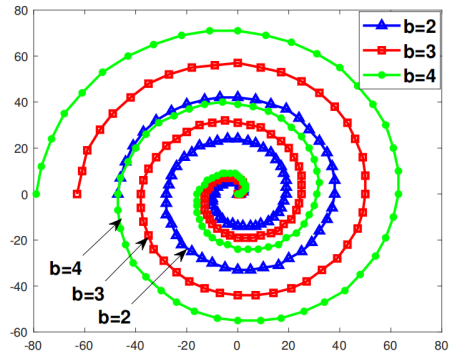


Fig. 4: Two-dimensional representation of spiral curves

behavior was introduced in the study of sand cat population, and its mathematical formula is described as follows:

$$\begin{aligned} &\overline{Pos}\,(t+1) = \overline{Pos}_b\,(t) \\ &\vec{r} \cdot \overline{Pos}_{md} \cdot \cos(\theta) \cdot a \cdot b \cdot l \cdot \cos(2\pi l), \mid R \mid \leq 1 \end{aligned} \tag{24}$$

$$\begin{aligned} &\overline{Pos}\,(t+1) = \overline{Pos}_b\,(t) - rand\,(0,1) \cdot \overline{Pos}_c\,(t) \\ &\left(\overline{Pos}_b\,(t) - 2 \cdot \alpha \cdot b \cdot l \cdot \cos\left(2\pi l\right)\right), \mid R \mid > 1 \end{aligned} \tag{25}$$

Where $b \cdot l \cdot \cos(l \cdot 2 \cdot \pi)$ represents the helix factor, which models the curved path of a sand cat. α is a natural coefficient that models the effects of wind and other natural factors and has a value of 1 or -1.

The spiral factor $b \cdot l \cdot \cos(l \cdot 2 \cdot \pi)$ describes an isosceles spiral, where b is a constant that controls the pitch of the spiral, and the larger the value of b, the larger the pitch of the spiral. l is a random number in the range $[-1, 1]$. When b takes the value of 3, 4 or 5, its image in two dimensions is shown in Fig. 4:

From Fig.4, it can be visualized that the value of the parameter b is directly proportional to the helix pitch of the spiral curve, the larger the value of b, the larger the helix pitch, which leads to a corresponding increase in the search range of the algorithm. However, too large a spiral pitch may cause the algorithm to ignore some potentially valuable regions of the solution space. After a series of tests, we finally decided to set the value of b to 4.

H. Dynamic adaptive adjustment mechanism strategy

In the realm of biodynamics, the sand cat has demonstrated extraordinary aerial attitude adjustment, the core mechanism of which is based on the law of conservation of angular momentum. When the sand cat is in free fall, it achieves attitude correction through asymmetric body movements. In the initial inverted state, the sand cat utilizes the high flexibility of its spine to break down its body into two rotational units, the front and the back. Through the reverse twisting of these two parts, the cat is able to correct the posture of the front half of the body while keeping the total angular momentum unchanged; then, through the stretching of the waist, it completes the synchronized adjustment of the back half of the body, and finally achieves a safe quadrupedal landing posture.

In addition, the cat adopts the dynamic adjustment strategy of "tuck-and-stretch" to optimize the moment of inertia. Specifically, the cat reduces the rotational inertia of the back by stretching the forelimbs and curling the hindlimbs to

improve the local rotational flexibility; then it changes the limb morphology by curling the forelimbs and stretching the hindlimbs to complete the fine-tuning of the stance. This process is consistent with the classical theory of rigid body moment of inertia regulation, just like figure skaters controlling rotational speed through limb expansion and contraction. At the same time, the tail of the sand cat acts as a "biological propeller", utilizing tail rotation to generate reverse angular momentum to assist in the dynamic balance of the overall posture.

At the algorithm simulation level, a nonlinear adaptive adjustment mechanism is introduced to model the attitude adjustment process of the sand cat. The sand cat swarm optimization algorithm (SCSO) regulates the dynamic balance between searching for prey and attacking prey through the balance parameter R, while the value of R is influenced by the auditory sensitivity parameter r_{G}. In the SCSO algorithm, r_{G} decreases linearly (from 2 to 0) as the number of iterations increases, but it is difficult to accurately simulate the complex dynamics of the sand cat's posture adjustment during the hunting process in this linear update approach. For this reason, the optimized r_{G} updating formula (26) is proposed to more accurately reflect the adaptive adjustment mechanism of sand cat during global posture adjustment (corresponding to the algorithm global search) and local posture correction (corresponding to the algorithm local attack) through the nonlinear mapping relationship, thus realizing the deep coupling of biomechanical process and intelligent algorithm. Where: a, b are non-zero constant coefficients.

$$r_G = 2 - 2\left(\frac{e'^{/t_{mex}} - 1}{e} - a\right)^b \tag{26}$$

Dung beetles are social insects that release pheromones upon discovering food sources to attract conspecifics, thereby enhancing the efficiency of resource collection and transportation back to their habitat. However, as food availability within the habitat increases, a "laziness phenomenon" emerges within the population, wherein an increasing number of individuals cease active foraging and instead resort to stealing resources from their peers. This behavioral pattern has been adopted in current research to model the behavior of sand cats, where certain individuals avoid the energetic costs of foraging and transporting resources by exploiting their nocturnal habits to covertly appropriate the efforts of others. Such behavior reflects a strategic trade-off between energy expenditure and potential gains at the individual level. From the perspective of optimization algorithms, this biological phenomenon serves as a compelling analogy for the balance between exploration (searching for new resources) and exploitation (utilizing known resources) in the search process. By mapping the global optimum to the most favorable food source in nature and simulating the "thieving dung beetle" mechanism around this optimum, researchers have developed a bio-inspired strategy for designing heuristic optimization algorithms. Nevertheless, traditional models often lack the dynamic adaptability required to effectively balance global exploration with local exploitation in complex and dynamic search environments, leading to issues such as premature convergence and reduced search efficiency.

Based on a comprehensive understanding of the ecological behavior of sandcats, this paper introduces a dynamic population strategy. By simulating the dynamic changes in the proportions of active foraging and resource transporting individuals as well as stealing individuals within the sandcat population, the algorithm categorizes the population into two groups: the rolling ball population and the stealing population. Specifically, the rolling ball population simulates the sandcats' active foraging and exploration of new resources, thereby performing extensive exploration within the search space to identify potentially optimal solutions. In contrast, the stealing population mimics the stealing behavior of sandcats, focusing on in-depth exploitation and refinement within the vicinity of the current best solution. The algorithm incorporates a dynamic adjustment mechanism that responds to the search status in real time. Analogous to how sandcats adjust their behavioral patterns based on food availability in their environment, this mechanism continuously evaluates the progress of the search throughout the iterative process and dynamically reallocates the proportion of individuals between the two populations. As a result, the algorithm achieves an adaptive balance between global exploration and local exploitation when addressing complex optimization problems, thereby significantly improving its optimization performance and robustness. The position update formula for this behavior is:

$$\begin{aligned}
&p_s = p_{TS} - p_T \\
&p_{TS} = pop \times 55\% \\
&p_T = Round(r_G) \\
&r_G = w_1 + c - (w_1 - w_2)/\left((2e^{t/t_{max}}/e) - 1 - a\right)^b \\
&\overline{Pos}\,(t+1) = \overline{Pos_b}\,(t) + S \times g \times \\
&(\left|\overline{Pos_c}\,(t) - \overline{Pos_{\mathrm{bc}}}\,(t)\right| + \left|\overline{Pos_c}\,(t) - \overline{Pos_b}\,(t)\right|)
\end{aligned} \tag{27}$$

Among them, PS represents the number of adaptive stealing sand cats, PTS is the sum of the number of rolling sand cats and stealing sand cats in the original SCSO, and PT is the number of adaptive rolling sand cats. r_{G} is a nonlinear adaptive factor, w_1 and w_2 are the initial and final values of the rolling sand cats, respectively. Through the grid search method, they are set to 14 and 3. t is the number of iterations, M is the maximum number of iterations, and c is an adjustment factor, mainly used to keep the convergence curve between w_1 and w_2, with a value of 1.4. a and b are non-zero constants. Pos_{bc} is the global optimal value, Pos_{b} is the current global optimal position, and Pos_{c} is the current position, which is a d-dimensional vector following a normal distribution, and is a constant, with a value of in the original text. To explain the dynamic population strategy more clearly, the dynamic change process of the sand cat population after 500 iterations is shown in Figure 5.

I. Description of the DPSCSO algorithm flow

The detailed steps of the improved DPSCSO (Desert Cat Population Search Optimization Algorithm) are as follows:

(1) Parameter initialization: set the number of desert cat population N, the search dimension D, the activity range of each dimension [lb, ub], the maximum number of iterations Tmax, and the selection of the fitness function;

(2) Initialization of sand cat location: segmented chaotic mapping initialization is used;

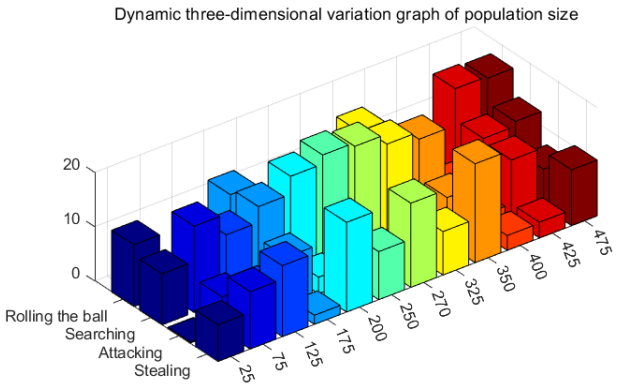


Fig. 5: Dynamic three-dimensional variation graph of population size

(3) Calculating individual fitness value: calculating individual fitness value based on the initial individual position in steps (1) and (2) and the updated individual position in steps (5) and (6) as the number of iterations increases;

(4) Updating parameters: calculate the value of the nonlinear factor r_G according to Eq. (26), and then update the values of parameters r, θ, and R;

(5) Conditional judgment: decide the next behavior of the desert cat colony based on the value of R. Judge the size of $|R|$ and 1 for the update of the desert cat individual position, and then judge the size of the current iteration number t and T. If $t < T$, then update the values of the parameters r, θ, and R according to step (4);

(6) Desert cat individual position update: when $|R| > 1$, the desert cat individual position is updated according to Equation (25), and when $|R| \leq 1$, the desert cat individual position is updated according to Equation (24).

(7) Algorithm iteration stopping judgment: if the current iteration number $t <$ maximum iteration number T, then return to step (3) and then skip step (4), (5), (6) in turn, until the algorithm reaches the maximum number of iterations and outputs the global optimal solution.

J. DPSCSO time complexity analysis

The time complexity of the SCSO algorithm is O(N×D×T), where N represents the population size, D represents the dimensionality, and T represents the number of iterations. DPSCSO adds four components compared to SCSO. The one-dimensional Piecewise chaotic mapping has O(N × D) time complexity. By introducing the dynamic adaptive adjustment mechanism strategy and the nighttime lost behavior, it aims to optimize the position update mechanism while keeping the time complexity of the original algorithm unchanged. Therefore, the time complexity of DPSCSO is calculated as O (N × D × T) + O (N × D), which remains O (N × D × T) after simplification. Compared with SCSO, DPSCSO maintains the same time complexity.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. Algorithm testing

To ensure objectivity and fairness, the experiments were conducted on 64-bit Windows 10 operating system, CPU model i7-9750H, graphics card model GTX1650, and simulation software MatlabR2024a.

B. Experimental design

In order to verify the effectiveness of the DPSCSO algorithm, we compared it with five other metaheuristic algorithms - DBO (Dung Beetle Optimization Algorithm), SCSO (Sand Cat Swarm Optimization Algorithm), SCA (Sine Cosine Optimization Algorithm), WOA (Whale Optimization Algorithm), and BKA (Black-Winged Kite Optimization Algorithm) – were compared. On the CEC2017 standard function set, we selected eight representative functions for testing, including the single-peak function F1, the multi-peak function F2, the hybrid functions F6, F7, and F8, as well as the composite functions F9, F11, and F12. The search interval for all the functions was set to be from -100 to 100. each experimental function was run independently for 50 runs, and the maximum number of iterations was set to be 500, and the population size is 30. evaluation criteria include mean (Ave), standard deviation (Std) and optimal value (Min).

C. Comparative analysis of algorithm results

The performance of the algorithm is evaluated by the optimal value, mean value and standard deviation, as shown in Table I. The smaller the optimal value and mean value, the higher the convergence accuracy of the algorithm; the smaller the standard deviation, the more stable the algorithm's optimization process. From the analysis results in Table 1, it can be seen that the average performance of DPSCSO is better than other algorithms in the eight test functions. In addition, the standard deviation of DPSCSO is significantly lower than that of other algorithms in most of the test functions, which indicates that it has better optimization ability and stability.

The results for the single-peak function F2 show that DPSCSO exhibits good convergence speed. Meanwhile, DPSCSO shows the best performance in terms of both mean and standard deviation of these functions. In the tests of multimodal functions F6-F8, DPSCSO outperforms the other algorithms. Although DPSCSO slightly underperforms DBO on functions F1 and F12, it still outperforms the other algorithms in terms of overall performance. It is particularly noteworthy that DPSCSO shows a significant advantage on functions F9, F11, which emphasizes its effectiveness in solving composite problems.

D. Algorithm convergence analysis

Convergence curve comparison: the convergence curves of the DPSCSO, DBO, SCSO, SCA, WOA, and BKA algorithms for the CEC2017 test functions are shown in Figure 6. In the CEC2017 comparison, the DPSCSO algorithm demonstrates excellent performance in most of the functions and achieves the highest level of accuracy. It is worth noting that the convergence rate of DPSCSO shows an accelerated trend as the number of iterations increases. This phenomenon

TABLE I: CEC2017 Benchmark Function Results

Function	Metric	DPSCSO	DBO	SCSO	SCA	WOA	BKA
F1	Ave	**6.5448E+02**	1.0252E+03	2.1857E+03	3.2404E+03	7.3064E+03	1.5434E+03
	Std	**1.3179E+03**	1.1926E+03	2.4723E+03	1.4905E+03	7.2190E+03	1.7915E+03
	Min	3.0008E+02	**3.0000E+02**	3.8572E+02	1.4614E+02	8.8825E+02	3.0559E+02
F2	Ave	**8.2115E+03**	8.3154E+03	8.3102E+03	8.4907E+03	8.4357E+03	8.2594E+03
	Std	1.1020E+01	1.1144E+01	8.6586E+00	**7.5300E+00**	1.7911E+01	8.3093E+00
	Min	**8.0400E+03**	8.0994E+03	8.1892E+03	8.3053E+03	8.1385E+03	8.0895E+03
F6	Ave	**1.8559E+04**	2.4935E+06	2.0364E+06	2.3186E+07	6.8369E+06	6.9494E+05
	Std	**2.6588E+04**	5.2536E+06	2.6174E+06	1.5233E+07	7.4534E+06	1.1727E+06
	Min	**2.4196E+03**	2.6026E+03	1.1938E+04	4.9751E+06	7.8011E+04	5.8143E+03
F7	Ave	**2.6314E+03**	1.8099E+04	1.4020E+04	8.4412E+04	1.6161E+04	1.3626E+04
	Std	**2.1817E+03**	1.9468E+04	1.0230E+04	5.8856E+04	1.3717E+04	9.7851E+03
	Min	**1.4204E+03**	1.5664E+03	3.0934E+03	8.8066E+03	2.4839E+03	2.4259E+03
F8	Ave	**1.4798E+03**	1.9577E+03	2.9938E+03	2.1765E+03	2.9418E+03	2.2806E+03
	Std	**3.5314E+01**	6.5637E+02	1.7806E+03	9.9534E+02	1.5573E+03	1.5103E+03
	Min	**1.4285E+03**	1.4920E+03	1.4682E+03	1.4987E+03	1.5025E+03	1.4345E+03
F9	Ave	**1.7612E+03**	1.7913E+03	1.8596E+03	1.8385E+03	1.9587E+03	1.7976E+03
	Std	**9.2642E+01**	1.2866E+02	1.1804E+02	1.0051E+02	1.2776E+02	1.5912E+02
	Min	**1.6016E+03**	1.6135E+03	1.6275E+03	1.6844E+03	1.7016E+03	1.6017E+03
F11	Ave	**1.9491E+03**	9.7463E+03	8.7077E+03	9.3716E+03	8.6450E+04	6.4905E+03
	Std	**3.4087E+02**	1.6924E+04	7.6346E+03	8.0540E+03	1.5881E+05	5.6861E+03
	Min	**1.9056E+03**	2.0337E+03	1.9198E+03	1.9443E+03	2.1842E+03	1.9171E+03
F12	Ave	**2.1030E+03**	2.1051E+03	2.1343E+03	2.1119E+03	2.1927E+03	2.1184E+03
	Std	**4.7880E+01**	7.0460E+01	5.4695E+01	2.2862E+01	9.4515E+01	6.7060E+01
	Min	2.0406E+03	**2.0221E+03**	2.0433E+03	2.0658E+03	2.0602E+03	2.0175E+03

can be attributed to the use of chaotic mapping initialization in the DPSCSO algorithm, which helps to explore promising regions in the search space during the initial iteration phase. In addition, the dynamic adaptive regulation mechanism strategy employed by DPSCSO as well as the nighttime lost behavior strategy help to maintain the search capability in the later stages of the iteration. This superiority is particularly evident in functions F1, F6, and F9.

V. MODELING OF HAZARDOUS OR OBSTRUCTED AREAS

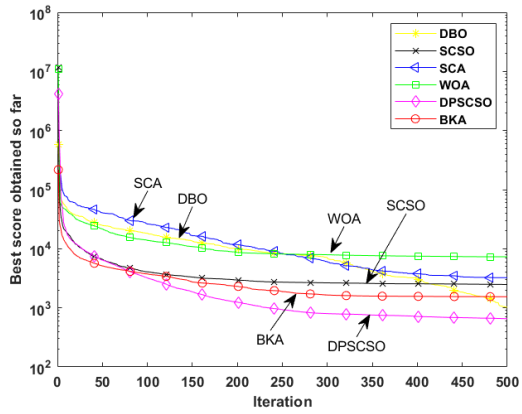
During UAV flight, its trajectory planning must take full account of various potential threat areas and obstacles. For example, threats such as birds and enemy attacks in the flight environment, and obstacles such as buildings. In order to facilitate the implementation of the trajectory planning algorithm, a cylinder model is used in this chapter to represent these hazardous regions or obstacles. The radius parameter of the cylinder determines the coverage of the threat area, and the larger the radius, the higher the danger level of the area or the wider the influence of the obstacles. In this study, a 100×150×2 discrete elevation data matrix is used to construct the environment model, and this modeling method can intuitively reflect the spatial distribution characteristics of the threat, provide effective obstacle avoidance constraints for UAV trajectory planning, and thus significantly enhance flight safety. Its environment simulation modeling graphic is shown in Fig 7.

A. Experimental analysis of trajectory planning

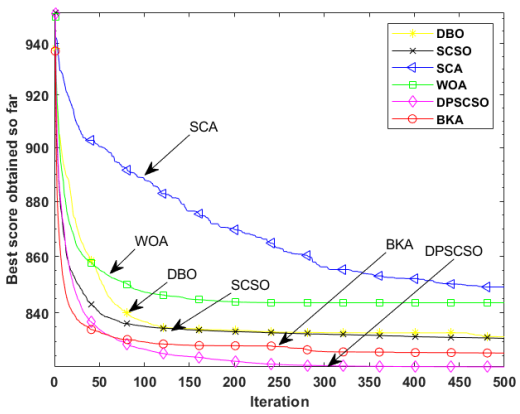
In order to verify the effectiveness and superiority of the Dynamic Sand Cat Swarm Optimization Algorithm (DPSCSO) in UAV 3D trajectory planning, this chapter conducts 20 simulation experiments of DPSCSO with the Sand Cat Swarm Optimization Algorithm (SCSO), the Dung Beetle Optimization Algorithm (DBO), the Snake Optimization Algorithm (SO), the Whale Optimization Algorithm (WOA), and the Black-winged Kite Optimization Algorithm (BKA), and the results are The results are analyzed in depth. The simulation platform chosen was Matlab2024a.During the simulation of the algorithms, the relevant parameters were set as follows: the planning area was a 100×150×2 3D map space, the population size was set to 30, and the maximum number of iterations of the algorithm was 200. In the comprehensive constraint objective function, the weight coefficients of each objective function are set as 1=0.2, 2=0.25, 3=0.25, 4=0.15, 5=0.15. The coordinates of the starting point S are (10,90,1.1), and the coordinates of the target point T are (130,10,1.4). The parameter settings of each related algorithm are detailed in Table II.

In order to comprehensively test the efficiency of the Dynamic Sand Cat Swarm Optimization Algorithm (DPSCSO) for UAV 3D trajectory planning in different environments, a variety of different map environments are constructed in this chapter for validation. Among them, the experimental data of trajectory planning in environment I is detailed in Table III.

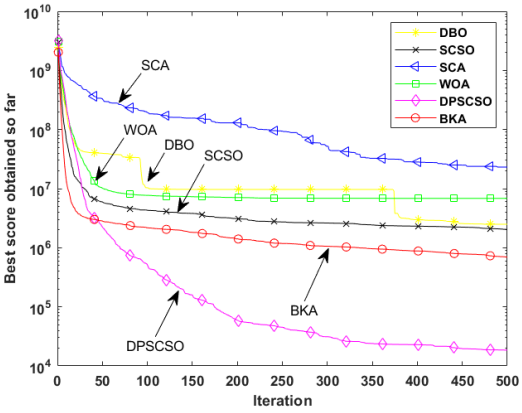
Table III shows the performance data statistics of each algorithm obtained through 20 experimental simulations. Observing the data, it can be found that DPSCSO outperforms the other algorithms in terms of the mean, standard deviation, maximum and minimum values of the objective function, except for BKA, which is slightly better in terms of the minimum value of the objective function. The difference between the maximum value and the minimum value of DPSCSO is small, which indicates that the results are more centralized, and the algorithm is more robust and the optimization is stable, avoiding the occurrence of the phenomenon of extreme deviation. . In terms of the maximum value of the objective



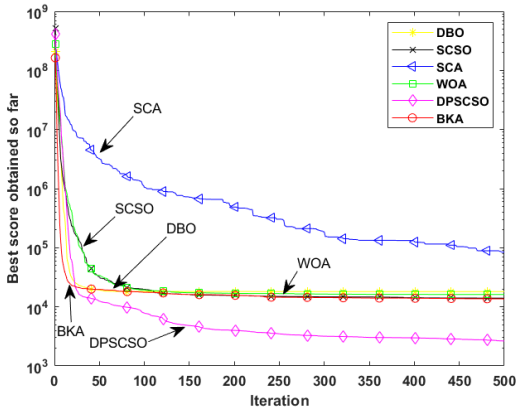
(a) F1



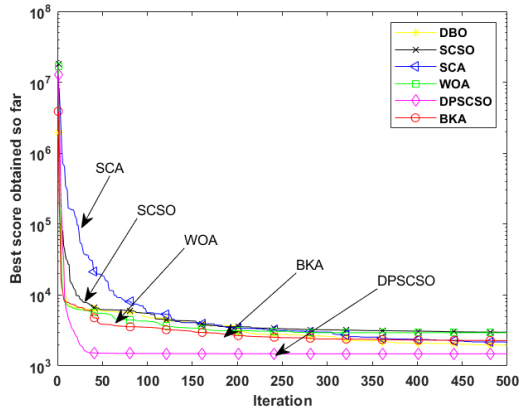
(b) F2



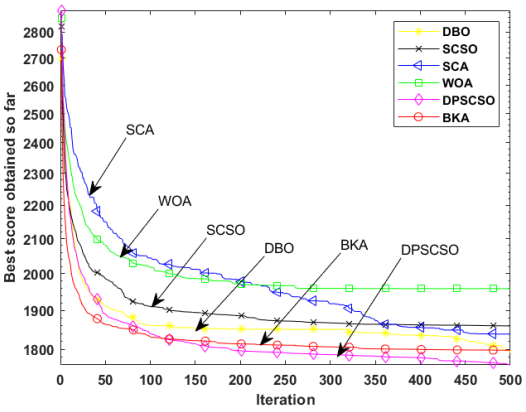
(c) F6



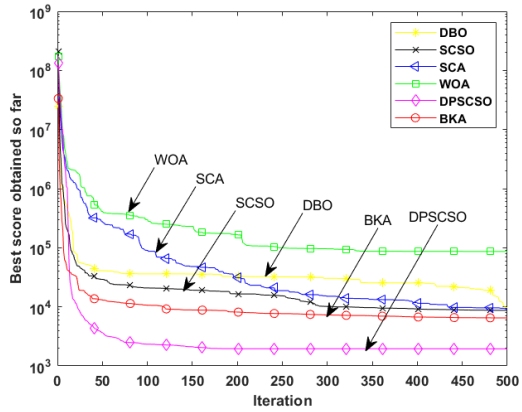
(d) F7



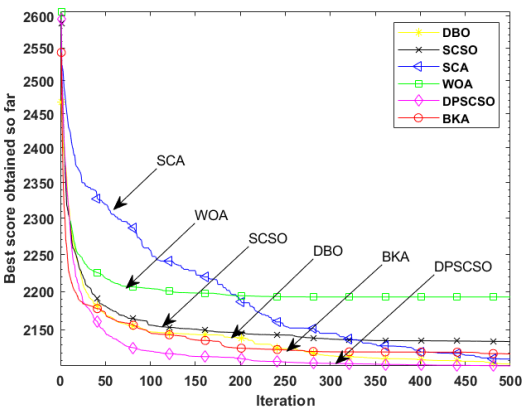
(e) F8



(f) F9



(g) F11



(h) F12

Fig. 6: Comparison of convergence curves for CEC2017 benchmark functions

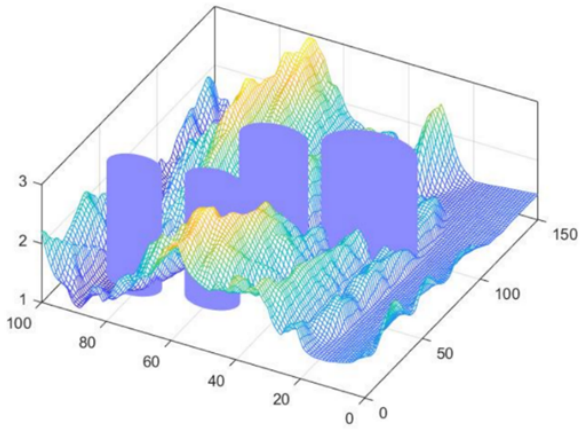


Fig. 7: 3D map modeling of hazardous or obstacle areas

function, the results obtained by DPSCSO optimization are smaller, which indicates that DPSCSO is able to consistently maintain better results during multiple optimization searches.

TABLE II: ALGORITHM PARAMETER SETTINGS

Algorithm	Population Size	Iterations	Parameters
DPSCSO	30	500	$r_G = 2 - 2\left(\frac{e^{t/T_{\max}}-1}{e-a}\right)^b$
SCSO	30	500	$\overline{r_G} = s_M \cdot \left(\frac{2\times s_M \times t}{T_{\max}}\right)$
DBO	30	500	rdb = 6; edb = 6; fdb = 7; sb = 11
SO	30	500	N = 30; T = 1000; c1 = 0.5; c2 = 0.05; c3 = 2
WOA	30	500	$a = 2 \times (1 - t/\max)$; $k = 1$
BKA	30	500	$p = 0.9$

TABLE III: TRAJECTORY PLANNING DATA IN 3D MAPS

Algorithm	Objective Function Mean	Standard Deviation	Maximum Value	Minimum Value
DPSCSO	73.2694	0.2022	73.6193	72.9834
SCSO	106.8034	16.9712	114.8828	73.5261
DBO	98.3795	16.3121	117.7606	75.0867
SO	105.8926	9.7267	114.9397	88.2019
WOA	76.8821	4.0359	85.9478	73.8753
BKA	75.0980	3.7928	84.1270	73.0241

B. Actual three-dimensional navigation route of the drone

Fig.8 presents the actual navigation routes of the UAV in 3D trajectory planning optimized by each algorithm. As can be seen from the 3D figure, DPSCSO plans a shorter path, and its route connects the starting point and the end point more directly, reducing unnecessary detours, thus completing the voyage with a shorter distance and saving resources. DPSCSO chooses a lower route, reducing the distance of upward flight. In addition, DPSCSO plans a smoother route with no obvious twists and turns, making the navigation process relatively smooth. For UAVs, this helps to reduce energy loss and mechanical wear due to frequent turns, thus improving navigation efficiency and extending equipment life. In contrast, the trajectories of other algorithms such as WOA and BKA have some unreasonable fluctuations in altitude changes.

C. Comparison of algorithm iteration curves

From the iteration curves of each algorithm in Fig.9, it can be seen that DPSCSO shows a significant improvement in terms of optimization speed and finding ability compared to other algorithms, demonstrating its excellent performance.

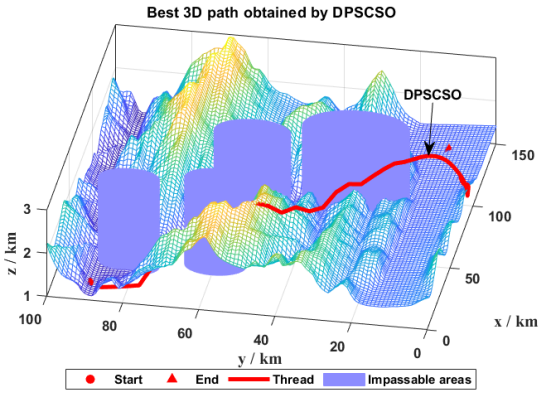
D. Experimental analysis of track planning in complex three-dimensional maps

To thoroughly evaluate the performance of the algorithm across diverse scenarios and mitigate the potential bias caused by random outcomes from testing on a single map, this study extends the experimental framework to include more complex three-dimensional maps. This enhancement provides a stronger theoretical foundation for the algorithm's practical application and broader promotion. In this chapter, seven recently proposed algorithms—LWSCSO, ISCSO, SCSO, HHO, WOA, GA, and SCA—are selected for comparative analysis. A total of twenty simulation experiments are conducted for each algorithm, and the results are systematically analyzed. To ensure experimental fairness, all tests are carried out on the MATLAB (2024a) platform under a consistent Windows 10 environment. The simulation parameters are set as follows: the planning area is defined as a three-dimensional map space of $100 \times 150 \times 2$; the population size is set to $m = 30$; and the maximum number of iterations is set to 500. In the comprehensive constrained objective function, the weight coefficients for each objective are assigned as follows: $\alpha_1 = 0.2$, $\alpha_2 = 0.25$, $\alpha_3 = 0.25$, $\alpha_4 = 0.15$, and $\alpha_5 = 0.15$. The S-coordinate of the starting point and the t-coordinate of the target point are specified accordingly. The parameter settings for the compared algorithms are summarized in Table IV.

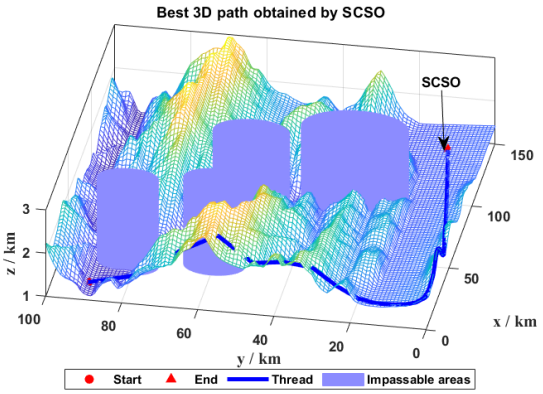
In order to comprehensively test the efficiency of the Dynamic Sand Cat Swarm Optimization Algorithm (DPSCSO) for UAV 3D trajectory planning in different environments, a variety of different map environments are constructed in this chapter for validation. Among them, the experimental data of trajectory planning in environment I is detailed in Table V.

E. Actual three-dimensional navigation route of the drone

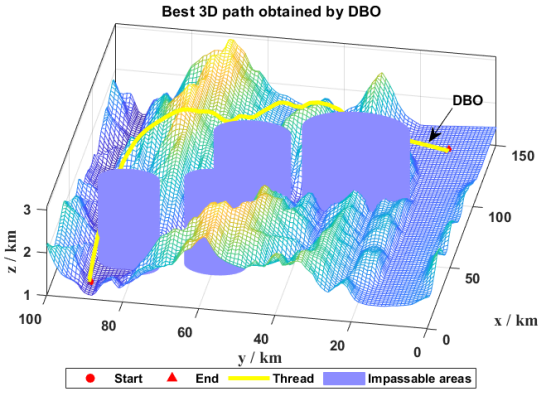
To comprehensively validate the effectiveness of DPSCSO in three-dimensional UAV trajectory planning under diverse environmental conditions, this chapter conducts simulations across multiple map scenarios for verification purposes. The trajectory planning experimental data for Environment 2 are summarized in Table V, which presents statistical results obtained from 20 independent simulation runs for each algorithm. As shown, the mean value of the objective function for DPSCSO is notably lower, indicating that, on average, DPSCSO demonstrates superior performance in optimizing



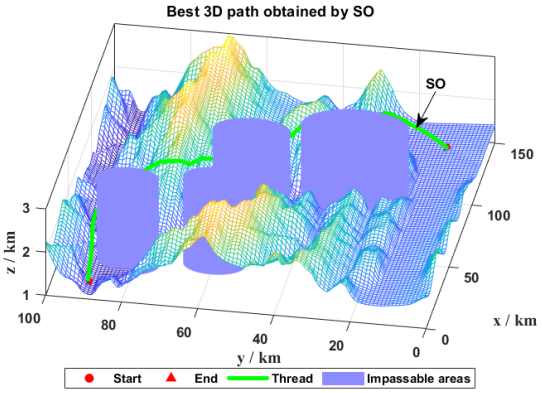
(a) DPSCSO



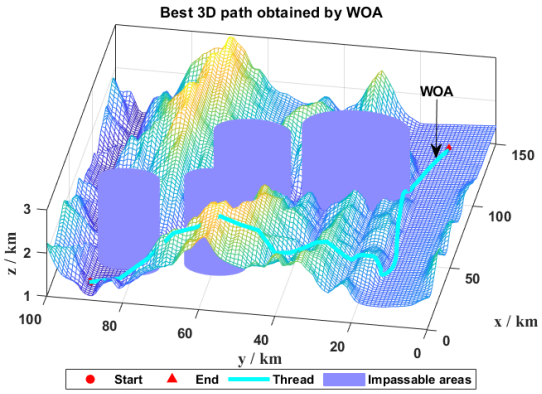
(b) SCSO



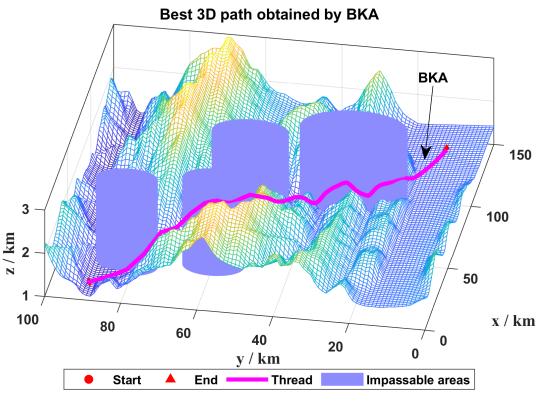
(c) DBO



(d) SO



(e) WOA



(f) BKA

Fig. 8: 3D map trajectory planning simulation

the objective function, thereby achieving better trajectory planning outcomes for drones. In comparison, the mean values of SCSO and SCA are significantly higher, suggesting that these original algorithms are less effective in optimization when compared to DPSCSO. Additionally, DPSCSO exhibits a relatively small standard deviation, reflecting its strong stability and robustness with minimal variation across multiple runs. On the contrary, SCSO and GA display larger standard deviations, indicating less consistent performance. The small gap between the maximum and minimum values of DPSCSO further confirms its consistent behavior. In contrast, algorithms such as SCSO show a considerable difference between their maximum and minimum values, highlighting greater variability in their results. Figure10 illustrates the three-dimensional trajectory planning of drones under various algorithm optimizations. It can be observed that the flight path generated by DPSCSO is lower, shorter, and avoids unnecessary detours. Moreover, the trajectory is relatively smooth with no abrupt turns, indicating a stable flight process and reducing the distance wasted on frequent maneuvers.

F. Comparison of algorithm iteration curves

Figure11 presents the iterative convergence curves of each algorithm in Environment 2. As shown in the figure, DPSCSO and LWSCSO converge rapidly in the early

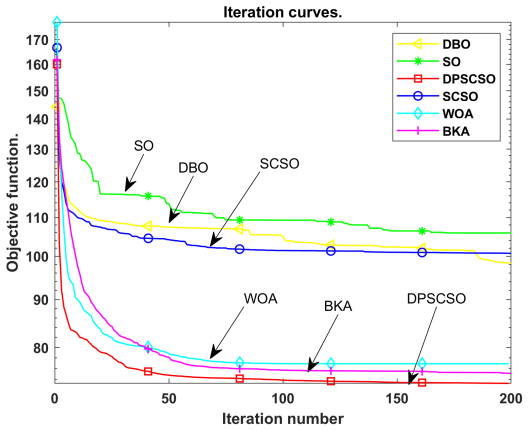


Fig. 9: Comparison of convergence speed in 3D maps

TABLE IV: ALGORITHM PARAMETER SETTINGS

Algorithm	Population Size	Number of Iterations	Parameters
DPSCSO	30	500	$a = [-1, 1]$; $b = 4$; $P = 0.7$; $W_1 = 14$; $W_2 = 3$; $c = 1.4$
LWSCSO	30	500	$W_1 = 0.9$; $W_2 = 0.7$; $\beta = 1.5$
ISCSO	30	500	$r_G = 2 - (\exp(((-2 * t)/T_{\max}) * \cos(2 * \pi * t)) + 1)$
SCSO	30	500	$r_G = S_M \cdot (2 * S_M * t)/T_{\max}$
HHO	30	500	$E_0 = [-1, 1]$
WOA	30	500	$a =$ 2 to 0
SCA	30	500	$a = 2$
GA	30	500	Cross $= 0.2$

stages of iteration, with their curves declining quickly and stabilizing early, which suggests that these algorithms can efficiently locate optimal solutions. In contrast, SCSO and GA exhibit slower convergence rates, with GA maintaining a relatively high objective function value over a prolonged period, leading to suboptimal performance. Similarly, WOA and HHO converge more slowly than DPSCSO and result in higher final objective function values. Overall, DPSCSO demonstrates superior optimization effectiveness and stability compared to the original algorithm and other metaheuristic algorithms such as HHO, WOA, GA, and SCA in UAV trajectory planning, indicating its promising application potential in this domain.

TABLE V: TRAJECTORY PLANNING DATA IN 3D MAPS

Algorithm	Objective Function Mean	Standard Deviation	Maximum Value	Minimum Value
DPSCSO	**73.0457**	**0.1115**	**73.1731**	**72.8976**
LWSCSO	73.6326	1.0864	76.7011	73.0681
ISCSO	75.2722	4.8485	88.2539	73.2114
SCSO	98.3568	19.1662	114.6881	73.9426
HHO	81.6374	6.8512	92.8741	73.2263
WOA	83.5943	17.8919	117.4369	87.9247
SCA	105.8267	9.7404	115.5224	92.3108
GA	94.0275	17.3316	118.2403	73.5557

Based on the above experiments, for the problem of drone trajectory planning in three-dimensional space, DPSCSO demonstrates superior path optimization capabilities in environments with low obstacle density and exhibits enhanced obstacle avoidance performance in areas with high obstacle density. In summary, the proposed swarm intelligence algorithm achieves notable improvements in performance across various trajectory planning tasks. This offers strong support for safe and efficient drone flight and indicates promising potential for application in complex operational environments. Future research will further investigate its applicability across a broader range of scenarios.

VI. CONCLUSION

This study proposes three enhancements to the Sand Cat Swarm Optimization (SCSO) algorithm. Firstly, the introduction of the Piecewise Linear Chaotic Mapping (PWLCM) strategy during population initialization improves the uniform distribution of the initial population, reducing the risk of local optima entrapment. Secondly, departing from conventional SCSO approaches, this research incorporates the nocturnal disorientation behavior of sand cats to develop a novel hybrid search strategy that integrates linear and spiral trajectories. This innovative approach not only diversifies the search patterns of SCSO but also boosts its capacity to detect global optima and hastens convergence. Lastly, a dynamic adaptive adjustment mechanism, grounded in nonlinear factors, is proposed to enable flexible positioning of sand cats. Initially, emphasizing nonlinear factors enhances global exploration, while in later stages, it effectively balances global exploration with local exploitation.

To assess the efficacy of the enhanced Sandcat Swarm Optimization (DPSCSO) algorithm, we conducted extensive experiments utilizing the CEC2017 benchmark test functions. Our findings from Experiment 1 reveal that DPSCSO exhibits superior global search performance and resilience when compared to meta-heuristic algorithms including SCSO, DBO, SO, WOA, and BKA. Experiment 2 results demonstrate that DPSCSO displays enhanced path avoidance capabilities and safety in regions with high-density obstacles, outperforming LWSCSO, ISCSO, SCSO, HHO, WOA, GA, and SCA. These thorough experiments confirm the efficacy of the DPSCSO algorithm in improving the efficiency of path planning for unmanned aerial vehicles.

REFERENCES

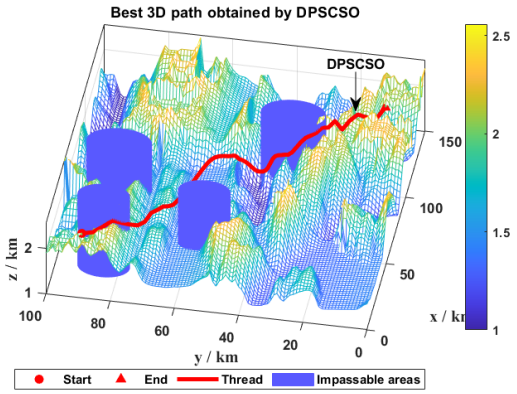
[1] S. Uluskan, "Noncausal trajectory optimization for real-time range-only target localization by multiple uavs," *Aerospace Science and Technology*, vol. 99, p. 105558, 2020.

[2] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

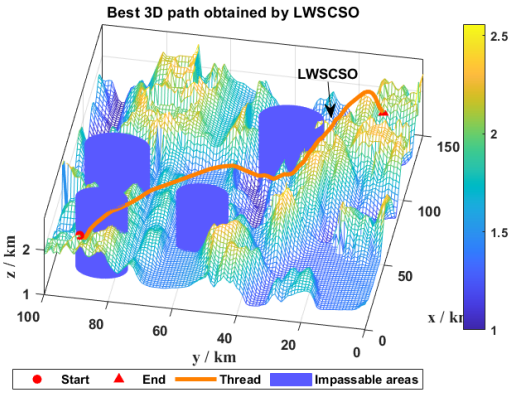
[3] G. G. Wang and S. Deb, "Brain storm optimization algorithm," *Knowledge-Based Systems*, vol. 48, pp. 1–12, 2013.

[4] M. Pant, H. Zaheer, L. Garcia-Hernandez *et al.*, "Differential evolution: A review of more than two decades of research," *Engineering Applications of Artificial Intelligence*, vol. 90, p. 103479, 2020.

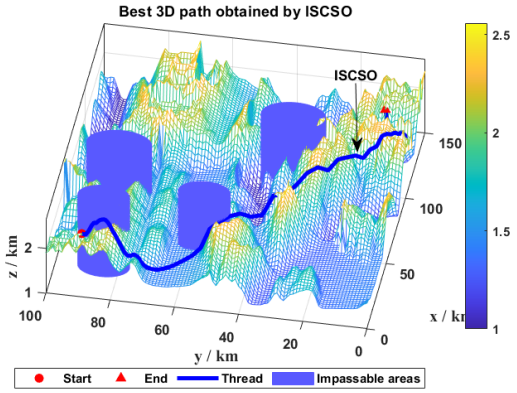
[5] X. Hu, S. Yang, Z. Cai *et al.*, "An ideology algorithm for global optimization," *Applied Soft Computing*, vol. 12, no. 11, pp. 3494–3504, 2012.



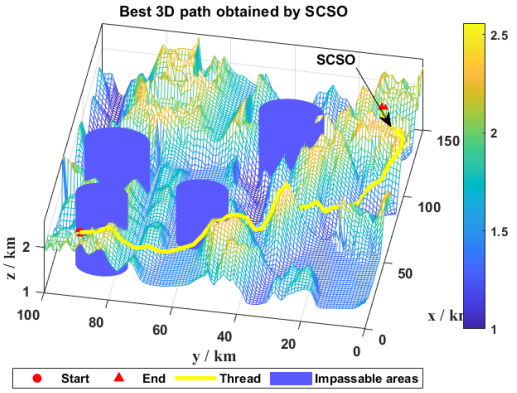
(a) DPSCSO



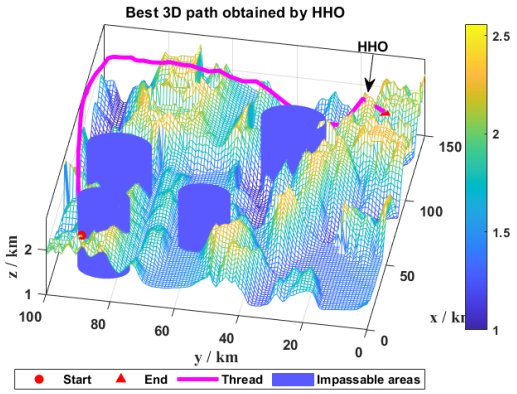
(b) LWSCSO



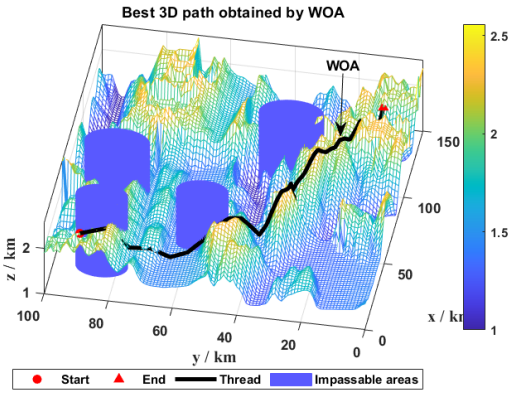
(c) ISCSO



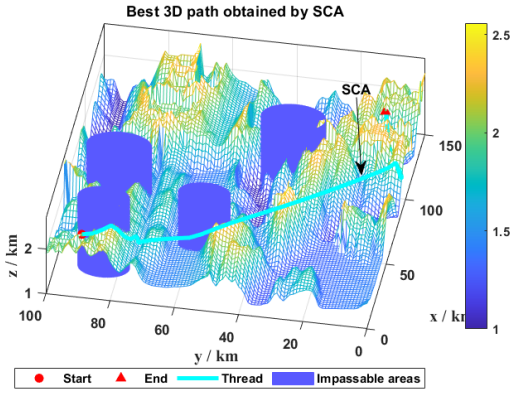
(d) SCSO



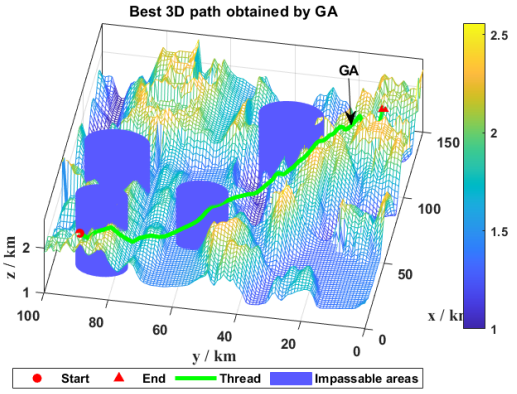
(e) HHO



(f) WOA



(g) SCA



(h) GA

Fig. 10: 3D map trajectory planning simulation

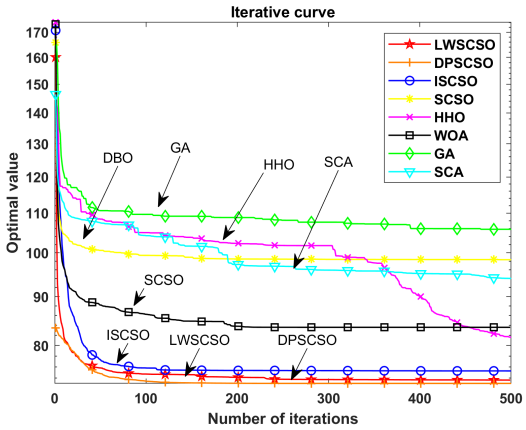


Fig. 11: Comparison of convergence speed in 3D maps

[6] S. Mirjalili, "Sine cosine algorithm: a novel optimization algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.

[7] A. Hatamlou, "Black hole: a new heuristic optimization approach for data clustering," *Information Sciences*, vol. 222, pp. 175–184, 2013.

[8] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "Gsa: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[9] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[10] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *Journal of Global Optimization*, vol. 39, pp. 459–471, 2007.

[11] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[12] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[13] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.

[14] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm," *Systems Science & Control Engineering*, vol. 8, no. 1, pp. 22–34, 2020.

[15] F. A. Hashim and A. G. Hussien, "Snake optimizer: A novel meta-heuristic optimization algorithm," *Knowledge-Based Systems*, vol. 242, p. 108320, 2022.

[16] A. Seyyedabbasi and F. Kiani, "Sand cat swarm optimization: A nature-inspired algorithm to solve global optimization problems," *Engineering with Computers*, vol. 39, no. 4, pp. 2627–2651, 2023.

[17] M. Abdel-Basset, R. Mohamed, M. Jameel, and M. Abouhawwash, "Spider wasp optimizer: a novel meta-heuristic optimization algorithm," *Artificial Intelligence Review*, vol. 56, no. 10, pp. 11 675–11 738, 2023.

[18] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowledge-Based Systems*, vol. 89, pp. 228–249, 2015.

[19] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm," *Computers & Structures*, vol. 169, pp. 1–12, 2016.

[20] B. Abdollahzadeh, F. S. Gharehchopogh, and S. Mirjalili, "African vultures optimization algorithm: A new nature-inspired metaheuristic algorithm for global optimization problems," *Computers & Industrial Engineering*, vol. 158, p. 107408, 2021.

[21] S. Mirjalili, "Sca: a sine cosine algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.

[22] P. D. Kusuma, "Best couple algorithm: A new metaheuristic with two types of equal size swarm splits," *IAENG International Journal of Applied Mathematics*, vol. 54, no. 8, pp. 1615–1623, 2024.

[23] P. D. Kusuma, "Group better-worse algorithm: A superior swarm-based metaheuristic embedded with jump search," *IAENG International Journal of Applied Mathematics*, vol. 54, no. 4, pp. 614–622, 2024.

[24] M. U. Sheikh, M. Riaz, F. Jameel *et al.*, "Quality-aware trajectory planning of cellular connected uavs," in *Proceedings of the 2nd ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond*, 2020, pp. 79–85.

[25] H. Wang, Y. Yu, and Q. Yuan, "Application of dijkstra algorithm in robot path-planning," in *2011 Second International Conference on Mechanical Automation and Control Engineering*. IEEE, 2011, pp. 1067–1069.

[26] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm-a literature review," in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*. IEEE, 2019, pp. 380–384.

[27] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *Advances in Swarm Intelligence: First International Conference, ICSI 2010, Beijing, China, June 12-15, 2010, Proceedings, Part I*, vol. 1. Springer Berlin Heidelberg, 2010, pp. 355–364.

[28] V. K. Prajapati, M. Jain, and L. Chouhan, "Tabu search algorithm (tsa): A comprehensive survey," in *2020 3rd International Conference on Emerging Technologies in Computer Engineering: Machine Learning and Internet of Things (ICETCE)*. IEEE, 2020, pp. 1–8.

[29] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the ICNN'95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[30] G. Chen, X. Wang, S. Mo, J. Zhang, W. Xiong, H. Long, and M. Zou, "Multi-objective power flow optimization based on improved hybrid crow search algorithm: A novel approach," *Engineering Letters*, vol. 30, no. 4, pp. 1417–1435, 2022.

[31] R. Gong, M. Chu, L. Liu, and L. Liu, "Twin margin hyperplanes distribution machine with equality constraints," *Engineering Letters*, vol. 33, no. 8, pp. 2949–2960, 2025.