

Article

A Butterfly Algorithm That Combines Chaos Mapping and Fused Particle Swarm Optimization for UAV Path Planning

Linlin Wang¹, Xin Zhang¹, Huilong Zheng^{2,*}, Chuanyun Wang¹, Qian Gao¹, Tong Zhang¹, Zhongyi Li¹ and Jing Shao¹

¹ College of Artificial Intelligence, Shenyang Aerospace University, Shenyang 110136, China; wanglinlin@sau.edu.cn (L.W.); zhangxin22@stu.sau.edu.cn (X.Z.); wangcy0301@sau.edu.cn (C.W.); gaoqian@buaa.edu.cn (Q.G.); zhangtong4@stu.sau.edu.cn (T.Z.); lizhongyi@stu.sau.edu.cn (Z.L.); shaojing@stu.sau.edu.cn (J.S.)

² School of Aeronautics and Astronautics, University of Chinese Academy of Sciences, Beijing 100049, China

* Correspondence: zhenghuilong@iet.cn

Abstract: Effective path planning is essential for autonomous drone flight to enhance task efficiency. Many researchers have applied swarm intelligence algorithms to drone path planning. For instance, the traditional Butterfly Optimization Algorithm (BOA) has been used for this purpose. However, traditional BOA faces challenges such as slow convergence and susceptibility to being trapped in local optima. An Improved Butterfly Optimization Algorithm (IBOA) has been developed to identify optimal routes to address these limitations. Initially, ICMIC mapping is utilized to establish the butterfly community, enhancing the initial population's diversity and preventing premature algorithm convergence. Following this, a population reset strategy is introduced, replacing weaker individuals over a specified number of iterations while maintaining a constant population size. This strategy enhances the algorithm's ability to avoid local optima and increases its robustness. Additionally, characteristics of the Particle Swarm Optimization (PSO) algorithm are integrated to enhance the butterfly's location update mechanism, accelerating the algorithm's convergence rate. To evaluate the performance of the IBOA algorithm, this study designed a CEC2020 function test experiment and compared it with several swarm intelligence algorithms. The results showed that IBOA achieved the best performance in 70% of the function tests, outperforming 75% of the other algorithms. In the path planning experiments within a simulated environment, IBOA quickly converged to the optimal path, and the paths it planned were the shortest and safest compared to those generated by other algorithms.

Keywords: path planning; unmanned aerial vehicle; butterfly algorithm; ICMIC Chaotic Mapping



Citation: Wang, L.; Zhang, X.; Zheng, H.; Wang, C.; Gao, Q.; Zhang, T.; Li, Z.; Shao, J. A Butterfly Algorithm That Combines Chaos Mapping and Fused Particle Swarm Optimization for UAV Path Planning. *Drones* **2024**, *8*, 576. <https://doi.org/10.3390/drones8100576>

Academic Editor: Abdessattar Abdelkefi

Received: 1 August 2024

Revised: 5 October 2024

Accepted: 9 October 2024

Published: 11 October 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Unmanned Aerial Vehicles (UAVs) are aircraft that autonomously execute flight missions using advanced electronics, sensors, navigation systems, and control mechanisms, eliminating the need for human intervention. In contrast to manned aircraft, UAVs offer numerous advantages, including cost-effectiveness, diverse task capabilities, enhanced flexibility, superior adaptability, and ease of operation. UAVs can operate in challenging or hazardous environments, thus improving safety levels by replacing pilots. Due to the rapid advancements in UAV technology, UAV applications continue to increase across various sectors. In the civil domain, UAVs find utility in inspecting power transmission lines for faults, executing agricultural tasks like spraying, watering, and fertilizing to enhance crop yield and quality, as well as conducting activities such as express delivery, aerial photography, aerial displays, and geographical mapping. Furthermore, in military settings, UAVs play integral roles in surveillance, offensive strikes, and logistical support. This underscores not only the swift evolution of UAVs but also their extensive applications in diverse fields.

The navigation and trajectory design of UAV systems is a pivotal aspect of executing flight operations, striving to determine the most efficient flight route from the origin to the destination while adhering to specific constraints. Standard planning methods include intelligent algorithms such as the ant colony algorithm, PSO algorithm, genetic algorithm, and classical algorithms such as the artificial potential field method and the A* algorithm. Over time, a wealth of research has been dedicated to UAV path planning algorithms, with a growing array of techniques being employed for this purpose. In these studies, the flight environments of UAVs vary greatly. They can be roughly categorized into several types: predictable environments, unpredictable environments, dynamic environments, static environments, indoor environments, outdoor environments, environments that are not explicitly specified, or a combination thereof. Furthermore, UAV path planning can be conducted in both three-dimensional and two-dimensional environments. Researchers and scholars utilize various swarm intelligence algorithms to tackle path-planning problems for robots and UAVs, yielding significant achievements. Junlin Li et al. [1] improved the A* algorithm by incorporating dynamic weighting to adjust the evaluation function and modifying the search neighborhood for adaptive search adjustments. Experimental results demonstrated the superior performance of the improved algorithm. Jinming Hu et al. [2] enhanced the rapid discovery random tree technique to tackle the problem of inadequate instant efficiency in mapping out routes for UAVs, and the workability of the method was confirmed through simulations. Duo Qi et al. [3] set out to solve the problems of sluggish convergence in the early phase and the inclination to get stuck in the local best solutions in the later stage by integrating adaptable modifying elements and refined condition alteration plans into the ant community technique, thereby proving the practicality of this approach. Chengzhi Qu et al. [4] enhanced the performance of the grey wolf algorithm by simplifying it and combining it with an improved symbiotic organism search algorithm. Simulation experiments demonstrated the effectiveness of this method. Guoqiang Hao et al. [5] improved the artificial potential field method, enabling the enhanced method to plan reasonable paths and reduce energy consumption during flight. Hao Liu et al. [6] optimized GA using POS to enhance its local search capability. They tested the path planning performance in complex environments through simulations. In addressing the obstacle of charting courses for UAVs, Chaoqun Zhang et al. [7] introduced a rescue-enhanced, heuristic exploration algorithm that expedites the formula's convergence rate. Yong He et al. [8] improved the sparrow search algorithm by introducing a nonlinear dynamic weighting factor and a dynamic boundary lens imaging reverse learning strategy. Experimental results demonstrated the algorithm's high convergence accuracy, confirming its superiority. Xiaobing Yu et al. [9] attained superior enhancement outcomes for UAV route arrangement by mingling the GWO algorithm and the Differential Evolution Formula, in contrast to the primary formula. Buqian Chen et al. [10] proposed an improved bat optimization algorithm based on spherical coordinates and a truncated mean stability strategy. Simulation experiments demonstrate that, compared to other algorithms, the improved bat algorithm generates more effective optimal solutions. AMALA SONNY et al. [11] refined the particle flock formula and accomplished UAV tridimensional route arrangement in intricate surroundings. Kai Meng et al. [12] introduced the Evolutionary State Estimation-based Multi-Strategy Jellyfish Search (ESE-MSJS) technique to scout optimal routes for coordinating flight paths for numerous UAVs, demonstrating its effectiveness in supporting intricate, collaborative UAV path planning scenarios. Keming Jiao et al. [13] put forward an approach for charting routes in three-dimensional settings specifically targeted at UAVs with the utilization of the Enhanced Gravitational Search Algorithm (EGSA), and empirical findings provide evidence of both the practicality and superiority of this technique.

Research on swarm intelligence algorithms in UAV path planning has matured, with a growing focus on the fusion of swarm intelligence algorithms to enhance aerial vehicle trajectory arrangement. One approach involves integrating partially improved swarm intelligence algorithms with other swarm intelligence algorithms for UAV path planning. Meanwhile, reinforcement learning has emerged as a promising method to address path-

mapping challenges in trajectory arrangement. Meng Xi et al. [14] proposed a lightweight reinforcement learning method for UAV path planning, optimizing the training process, network structure, and algorithm model. Comparative experiments demonstrated the superiority of this method. Yuting Cheng et al. [15] designed a novel algorithm combining hierarchical reinforcement learning and simulated annealing to address UAV path planning issues. Experimental results indicate that this algorithm converges faster, achieves better learning outcomes, and produces more optimal routes. Haotian Shi et al. [16] introduced an adaptive dimensionality reduction framework in reinforcement learning to simplify high-dimensional state spaces and reduce computational complexity. Experimental results demonstrated that this method significantly enhanced exploration capabilities, reduced computational complexity, and improved path-planning efficiency. Xuqiong Luo et al. [17] enhanced the TD3 algorithm to enable UAVs to autonomously generate path trajectories through iterative online learning and continual experimentation. This showcases its efficacy in guiding UAVs through complex obstacle-ridden environments toward designated targets.

Swarm intelligence algorithms, such as the BOA, have been identified as promising solutions for addressing UAV route planning issues due to their advantageous characteristics. Unlike traditional methods, these algorithms leverage the collaboration and information-sharing abilities of a group, allowing for more extensive exploration within the search space. Through interactions and exchange of information among individuals, swarm intelligence algorithms excel at discovering global optimal solutions, thereby overcoming the limitations associated with local optimal solutions. Additionally, their robustness and adaptability enable them to adjust and optimize based on variations in the environment and problem characteristics. Moreover, the design and implementation of swarm intelligence algorithms are highly flexible, allowing for easy adjustments and extensions based on specific problem characteristics, making them well-suited for a variety of path-planning challenges. These exceptional features render swarm intelligence algorithms particularly adaptable to complex path-planning problems. Despite the commendable performance of traditional algorithms like A*, artificial potential field, RRT*, and reinforcement learning methods, each has its own set of constraints. For instance, the computational complexity of A* increases exponentially with the search space, the artificial potential field approach may encounter difficulties in planning tasks, the RRT* algorithm may face challenges in approximating the optimal path, and reinforcement learning demands substantial training inputs and time commitments. In contrast, swarm intelligence algorithms stand out for their ability to offer efficient and effective solutions with fewer parameters, simpler structures, and greater ease of comprehension.

Presented in reference [18], the BOS is a novel swarm intelligence optimization method developed by Sankalap Arora. BOA has demonstrated notable advantages in solving unconstrained mathematical functions due to its simple structure and minimal parameters. However, the algorithm's reliance on interactions between pairs of butterfly individuals for exploration highlights a vulnerability to overlooking optimal solutions, hindering its exploratory capabilities. Moreover, the absence of a mutation process within BOA's internal population causes other individuals to be drawn towards the leading individual on the global stage, resulting in entrapment in local optima and impeding the algorithm's ability to identify the most favorable solution. To address these limitations, researchers have proposed various enhancement strategies, such as parameter-based adjustments, chaos optimization integration, learning strategy modifications, and amalgamating with other algorithms, to bolster the algorithm's search efficiency and minimize susceptibility to local optima. Sushmita Sharma et al. [19] enhanced the BOA by incorporating adaptive parameter settings, the Lagrange interpolation formula, and Levy flight search strategies. They also modified the scent generation scheme of BOA to improve its exploratory capabilities. Various experiments demonstrated the competitiveness of the proposed algorithm. Apu Kumar Saha et al. [20] proposed an improved strategy that combines the BOA with the sine-cosine algorithm to enhance exploration and search capabilities. When compared with other algorithms, the proposed method achieved superior results in 75% of cases,

demonstrating its excellent performance. Yu Li et al. [21] proposed an opposition-based BOA with adaptive elite mutation. The algorithm incorporates an opposition learning mechanism to enhance population diversity and uses a segmented adjustment factor to balance global and local search processes, thereby improving optimization accuracy. Additionally, the elite mutation is employed to prevent premature convergence. Benchmark function tests demonstrated that the proposed algorithm outperforms other algorithms. Sushmita Sharma et al. [22] proposed a novel hybrid BOA that integrates the symbiotic and parasitic phases of the Symbiotic Organisms Search (SOS) algorithm to enhance the search capabilities of the BOA. Experimental results confirmed the success of this improvement, demonstrating satisfactory convergence times.

Other scholars have made various improvements to the BOA. Arora, Sankalap et al. [23] introduced chaotic mapping into the BOA to enhance its performance in avoiding local optima and improving convergence speed. Experimental results indicate that chaotic mapping significantly boosts BOA's performance. Yanju Guo et al. [24] applied guided weights and population restart strategies to the BOA to enhance its performance. Experimental results indicate that the proposed algorithm improves convergence speed, accuracy, and the ability to escape local optima. Wenxin Gao et al. [25] proposed a BOA with Cauchy mutation and adaptive weight. Experimental results indicate that the proposed method achieves better accuracy, speed, and stability for most test functions. Inspired by their work, this paper introduces a series of enhancements to the BOA to improve its performance. The enhancements not only boost the algorithm's capacity for exploration and escaping local optima but also quicken its convergence rate. Moreover, the algorithm's calculation is utilized for UAV route planning. The primary enhancements, as delineated below, aim to improve the BOA:

- (1) By using disorderly mapping to start the butterfly group, the variety of the butterfly group is strengthened.
- (2) Combining the butterfly rule with the PSO rule leverages the qualities of the PSO rule to accelerate the convergence pace of the butterfly rule.
- (3) Integrating active search strategies modifies the equilibrium between global exploration and local exploitation in the BOA, thus enhancing the algorithm's search capacity.
- (4) Implementing a population restart mechanism involves removing inferior individuals from the population and replacing them with newly generated individuals.

The organization of this paper is structured as follows: Part 2 presents a comprehensive background of the BOA. Part 3 provides a concise overview of the PSO method. Part 4 thoroughly examines the enhancements made to the BOA. Part 5 conducts experimental analysis and comparisons, while Part 6 offers a summary of the entire document and discusses the implementation of the work.

2. Algorithm Description

This section provides an overview of traditional BOA and PSO, followed by a discussion of four key improvements of IBOA in UAV trajectory planning.

2.1. Basic Butterfly Optimization Algorithm

Butterflies use their senses, such as smell, taste, and touch, to locate food, mating partners, and avoid predators in nature. They rely on sensory receptors situated on their antennae, legs, and proboscis to perceive odors, discern between different scents, and gauge their intensity. The odor emitted by each butterfly is influenced by its fitness level. Consequently, any change in a butterfly's position will result in a corresponding alteration in its fitness value. When a butterfly can detect the scent of other butterflies, it will move toward the source of the odor, engaging in what is termed global exploration. On the other hand, if a butterfly is unable to sense the smell, its movement will lack direction and be random, referred to as local search.

When a butterfly emits more fragrance, the surrounding butterflies sense it and are attracted to it. The fragrance coefficient can be updated in (1).

$$f = cI^\alpha \quad (1)$$

The magnitude of scent awareness, indicating the force at which other butterflies notice the scent, is symbolised by f , and c signifies the sensorial style, while I denotes the stimulus strength, and α represents the power index connected to the mode, with α and c fluctuating between $[0, 1]$.

Drawing from the behavior of butterflies, the optimization function of the BOA can be broken down into several key characteristics. Firstly, butterflies emit specific fragrances to attract each other's attention, influencing their movement dynamics significantly. This behavior impacts the distribution of scent, as butterflies decide whether to move randomly or towards individuals with higher fitness levels. Additionally, the intensity of a butterfly's fragrance correlates with the optimization function, highlighting the connection between a butterfly's location and the optimization process. The BOA can be divided into three fundamental stages: the initial phase, the iterative phase, and the final stage. The initiation phase focuses on setting objectives, defining resolution parameters, and organizing the initial clusters of butterfly populations. Subsequently, the iterative phase involves butterflies relocating based on a specific formula, assessing their fitness levels, and repeating this process multiple times to refine their positions. Throughout these iterations, information is generated at each position following a predefined formula. During these repetitive cycles, the algorithm leverages two critical stages: a comprehensive search and localized exploration, which aid in identifying and advancing toward optimal solutions within the defined problem space. In the process of overall investigation, butterflies shift toward the optimum solution (g^*), which can be denoted as Equation (2):

$$\chi_t^{n+1} = \chi_t^n + (R^2 \times g^* - \chi_t^n) \times f_t \quad (2)$$

Following n rounds of iteration, χ_t^n embodies the answer array of the t th butterfly, χ_t . The g^* epitomises the optimal solution discovered from all available solutions in the ongoing round. The fragrance of the t th butterfly is denoted by f_t , while R denotes represents a random digit within the range of 0 to 1.

During the localized exploration stage, the ensuing location of a butterfly can be ascertained using (3).

$$\chi_t^{n+1} = \chi_t^n + (R^2 \times \chi_j^n - \chi_k^n) \times f_t \quad (3)$$

where χ_j^n and χ_k^n respectively represent the j th and k th solutions.

The entirety of the BOA comprises the steps mentioned earlier, where, upon reaching the maximum iteration count, the algorithm ceases iterating. It produces the optimal solution determined by its lowest fitness value. By mimicking the actions of butterflies, this method becomes capable of solving optimization issues. These idealized features and algorithm stages provide a robust framework for the BOA. It is capable of searching for global optimal solutions in complex solution spaces and possesses certain robustness and adaptability. Therefore, the BOA has the potential to be effective in dealing with various optimization problems and can serve as an efficient optimization method. The pseudocode for the algorithm is shown as Algorithm 1.

2.2. Particle Swarm Optimization

The working principle of the PSO [26] method is based on the concept of mimicking a flock of birds foraging in a multi-dimensional exploration environment. In this method, each particle is considered a unique entity that moves through the exploration space to find the optimal solution. The key aspects of a particle's state include its position and speed, where the position corresponds to its location in the exploration space, and the speed determines both the direction and rate of its movement. Particles in the PSO algorithm

continuously update their positions and velocities to improve their chances of reaching the optimal outcome through iterative refinement. During each iteration, particles adjust their positions and velocities by considering their personal best solution so far as well as the collective knowledge gained from the entire group, known as the global best solution. This collaborative approach enables PSOs to converge toward the best solution efficiently. The equations used for updating the position and speed of particles in the PSO algorithm are as follows:

$$v_t^{n+1} = \omega \times v_t^n + c_1 \times R_1 \times (p_{best} - \chi_t^n) + c_2 \times R_2 \times (g_{best} - \chi_t^n) \quad (4)$$

$$\chi_t^{n+1} = \chi_t^n + v_t^{n+1} \quad (5)$$

Algorithm 1 Butterfly Optimization Algorithm

```

1: Generation initial population of Butterfly
2: Define sensor modality  $c$ , power exponent  $\alpha$  and switch probability  $p$ 
3: For  $t = 1$ :the max iterations do
4:   For each butterfly do
5:     Calculate fragrance for butterfly using Equation (1)
6:   End for
7:   Find the best butterfly
8:   For each butterfly do
9:     Generate a random number  $r$  from  $[0, 1]$ 
10:    If  $rand < p$  then
11:      Move towards the best butterfly/solution using Equation (2)
12:    else
13:      Move randomly using Equation (3)
14:    End if
15:  End for
16:  Update the value of  $\alpha$ 
17: End for
18: Output the best solution found

```

At which v_t^n and v_t^{n+1} symbolises the speed of the i th particle at iteration n and $n + 1$. p_{best} signifies the private prime resolution, and g_{best} denotes the worldwide prime resolution. Customarily, $c_1 = c_2 = 2$. R_1 and R_2 are haphazard figures within $[0, 1]$. The w can be derived as:

$$\omega(t) = \omega_{\max} - \frac{(\omega_{\max} - \omega_{\min}) \cdot T_t}{T_{\max}} \quad (6)$$

where ω represents the inertia weight, which influences the algorithm's search capability. Here, ω_{\max} is the maximum inertia weight, and ω_{\min} is the minimum inertia weight. In this study, ω_{\max} is set to 0.9 and ω_{\min} to 0.1. T_{\max} stands for the upper limit of iterations, and T_t stands for the present number of iterations.

2.3. Improving the Butterfly Algorithm

2.3.1. Chaos Mapping for Initializing the Butterfly Population

Chaotic sequences possess various characteristics, such as ergodicity, randomness, sensitivity, and regularity. Mathematically, chaos is defined as seemingly random sequences generated by simple deterministic systems. Chaos is a widespread nonlinear phenomenon characterized by complex and semi-random behavior, making it a suitable choice for initializing the butterfly population to enhance the efficiency and acceleration of the method. Utilizing chaotic mapping to initiate the butterfly swarm ensures a more uniform and extensive distribution of the swarm, enabling broader coverage in the search space. The initialization of the butterfly population is crucial for the success and fast convergence of the BOA. Traditional random initialization may lead to the algorithm being stuck in local optima. By contrast, chaotic initialization can effectively steer the optimization algorithm toward more favorable paths, hastening the convergence rate. The document selects chaotic

cartography to start the butterfly population and avoid falling into local optima. Due to the regularity of chaotic sequences, they are generated by deterministic functions, providing a theoretical basis for their application in optimization algorithms. Chaos’s characteristics make chaotic variables more effective for optimization search than random variables. To achieve a more extensive and uniform distribution of the butterfly population, an iterated chaotic mapping with infinite folding is employed in this paper to initialize the butterfly population. This map is formally defined as the follows:

$$x_{n+1} = \sin\left(\frac{\alpha\pi}{x_n}\right), a \in (0,1), x_n \in (-1,1) \tag{7}$$

In the equation, chaos sensitivity is observed when α is greater than 0.6, where minor changes in initial conditions can lead to significant variations in output values. Chaotic sequences were generated with varying initial values x_0 at $\alpha = 0.65$. The paper presents scatter plots of 500 points showcasing the generated values on a two-dimensional plane, as illustrated in Figures 1–3 on the left side. Concurrently, the probability histogram of the generated values is displayed on the right side. The results exhibit varying outcomes obtained under different initial conditions, thereby confirming the inherent chaos sensitivity phenomenon.

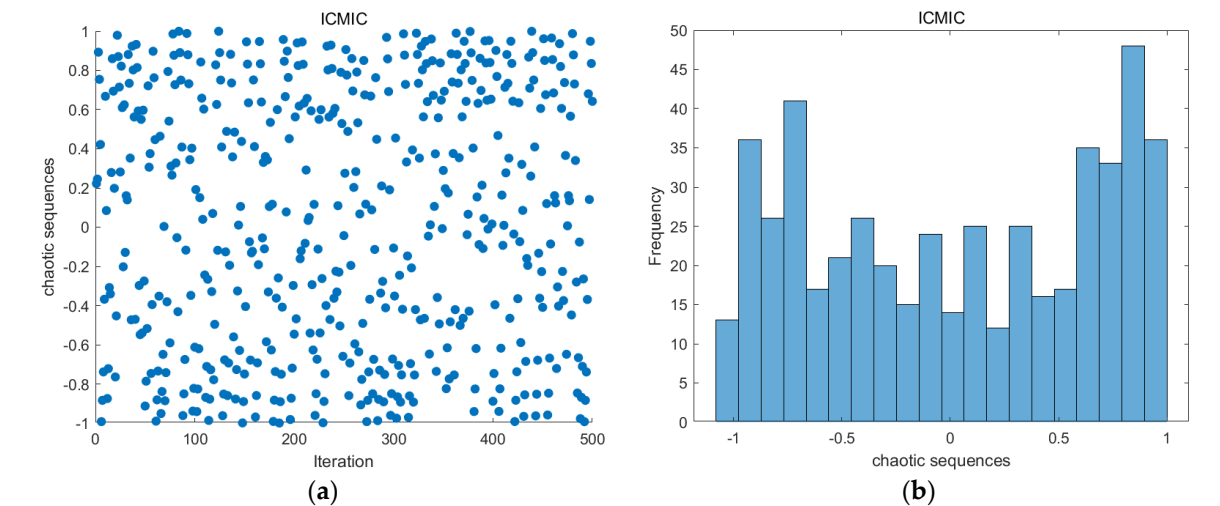


Figure 1. The initial value is 0.7. (a) Scatter plot; (b) Probability histogram.

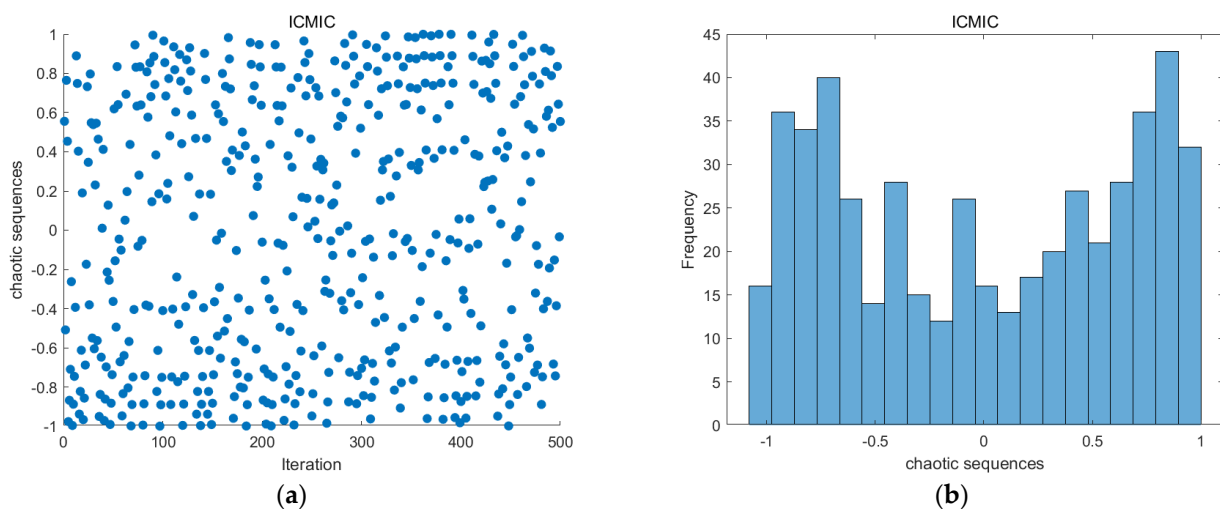


Figure 2. The initial value is 0.8. (a) Scatter plot; (b) Probability histogram.

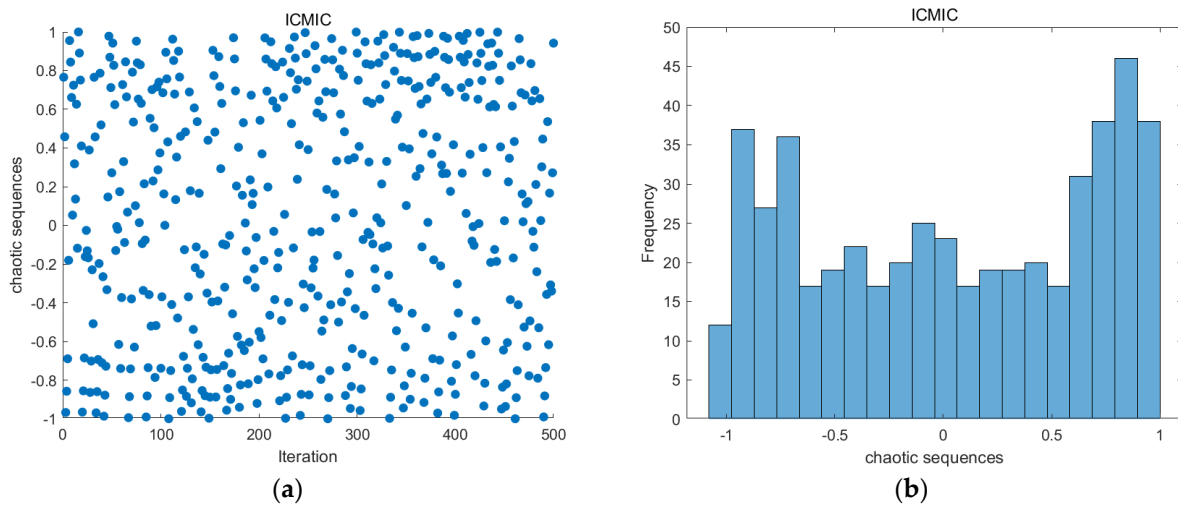


Figure 3. The initial value is 0.9. (a) Scatter plot; (b) Probability histogram.

2.3.2. Hybrid BOA with PSO

During the algorithm search process, PSO exhibits faster convergence and better global search capabilities but may get trapped in local optima near extrema. In contrast, the butterfly algorithm converges more slowly but can more effectively explore local regions. To address the limitations of these two algorithms, their strengths should be combined rather than used individually. By combining these strengths, the optimization process’s search efficiency and solution quality are enhanced. Meanwhile, the position and velocity update formulas of the PSO algorithm remain unchanged, as follows:

$$v_t^{n+1} = \omega \times v_t^n + c_1 \times R_1 \times (p_{best} - \chi_t^n) + c_2 \times R_2 \times (g_{best} - \chi_t^n) \tag{8}$$

$$x_t^{n+1} = x_t^n + v_t^{n+1} \tag{9}$$

At time n and $n + 1$ iteration, the velocity of the t th particle is represented by v_t^n and v_t^{n+1} . The individual learning factor, c_1 , increases a bird’s tendency to fly to locations where it previously found the most food. Conversely, the social learning factor, c_2 , increases a bird’s tendency to fly to locations where other birds have found the most food. $c_1 = 0.75$ and $c_2 = 0.25$. The variables R_1 and R_2 represent random numbers within the range of $[0, 1]$. ω can be obtained from Equation (6). The formulas for global and local searches in the BOA are altered by amalgamating the PSO with the BOA.

The new formula for global search is as follows:

$$\chi_t^{n+1} = \omega \times \chi_t^n + (R^2 \times g_{best} - \omega \times \chi_t^n) \times f_t \tag{10}$$

The new formula for local search is as follows:

$$\chi_t^{n+1} = \omega \times \chi_t^n + (R^2 \times \chi_j^n - \omega \cdot \chi_k^n) \times f_t \tag{11}$$

where χ_j^n and χ_k^n respectively represent the j th and k th solutions.

2.3.3. Population Restart

Swarm intelligence algorithms often struggle with getting stuck in local optimal solutions, hindering their ability to reach the global optimum. The solution to this problem is inspired by the natural law of survival of the fittest, where weaker individuals are eliminated from the population. Consequently, this paper implements a restart strategy known as the “individual elimination mechanism”. Through this approach, the population size remains constant, but as the population iterates, the probability of eliminating inferior individuals grows. At predetermined intervals, these inferior individuals are purged from

the population and replaced by newly generated individuals for continued search iterations. Implementing the individual elimination mechanism within the restart strategy helps tackle the challenge of local optima and elevates algorithm performance. The key strength of this approach lies in its continual elimination of inferior individuals, driving the population to refresh and evolve constantly. Consequently, this enhances the global search capability, enables the algorithm to escape local optima, and bolsters algorithm robustness. Below is the individual update formula for reference:

$$\chi_t^{n+1} = \text{ICMIC} \times (ub - lb) + lb \quad (12)$$

where χ_t^{n+1} is the solution vector of the t th butterfly at $t + 1$ th iteration; ICMIC is the sequence generated by the ICMIC chaotic mapping, generated according to the formula shown in Equation (7); ub and lb represents the upper and lower bounds of the solution vector. In this paper, it is chosen to update and replace inferior individuals every 25 iterations.

2.3.4. Dynamic Search Strategy

In the first edition of the BOA, an unfailing transition likelihood denoted as $p \in [0, 1]$ manages the determination involving worldwide exploration and regional investigation. A rational exploration process should involve conducting rapid, extensive global searches early in the algorithm to establish a general range for the optimal global solution. As the search progresses, the focus should gradually shift to enhancing local search capabilities for more precise exploration of local areas, thereby improving optimization accuracy. To achieve a balance between global and local exploration, a dynamic search strategy was designed. The formula for the switching probability is as follows:

$$p = 0.85 - 0.65 \times \frac{Iter - t}{Iter} \quad (13)$$

The total number of iterations is represented by $Iter$, while the current iteration count is denoted by t .

2.3.5. Improved Butterfly Optimization Algorithm

The manuscript introduces the IBOA, which combines disorderly mapping, populace rejuvenation tactics, PSO merges with BOA, and dynamic exploration strategy. Algorithm 2 presents the pseudocode for the algorithm.

Algorithm 2 Improved Butterfly Optimisation Algorithm

- 1: Butterfly population initialized using ICMIC chaotic mapping
 - 2: Initialize parameters $C1, C2, \omega_{\max}, \omega_{\min}$
 - 3: Define sensor modality c and power exponent α
 - 4: Calculate fitness for each butterfly and find the global best fitness
 - 5: **For** $t = 1$:the max iterations **do**
 - 6: **For** each butterfly **do**
 - 7: Calculate fragrance using Equation (1)
 - 8: Calculate the fitness of the current butterfly
 - 9: Calculate the switching probability p using Equation (13)
 - 10: Calculate ω using Equation (6)
 - 11: **If** $rand \times 1.1 > p$ **then**
 - 12: Move towards the best butterfly using Equation (10)
 - 13: **else**
 - 14: Move randomly using Equation (11)
 - 15: **End if**
 - 16: Calculate the fitness of the current butterfly
-

Algorithm 2 *Cont.*

```

17:   End for
18:   Find the best fitness
19:   For each butterfly do
20:     Update the velocity using Equation (8)
21:     If the global best fitness < fitness of the butterfly after search then
22:       Update the position of the butterfly using Equation (9)
23:     End if
24:     Perform collision detection and update the position of irrational butterflies
25:     Calculate the fitness of each butterfly, update the best fitness and best position
26:   End for
27:   If  $t == 20$  then
28:     Restart inferior butterfly individuals using Equation (12)
29: End if
30: Update the perceptual modality
31: End for
32: Return the best solution

```

3. Experiment and Result Analysis

This study validates the IBOA by comparing its performance with the original BOA, PSO, GWO, and other scholars' improved BOA, using ten benchmark functions from the CEC2020 numerical optimization competition. The study also involves planning the flight path of drones in a three-dimensional simulation environment. Upon analyzing the results obtained through these comparisons, it becomes apparent that the enhanced algorithm demonstrates superior optimization performance and provides substantial advantages in drone route planning.

The experiments were conducted on the same platform. MATLAB 2023b, installed on a Windows 11 (64-bit) system with an AMD Ryzen 7 6800H processor (3.20 GHz) and 16.0 GB RAM, was used to compare the results of all algorithms.

3.1. CEC2020 Experiments

The ten benchmark functions in the CEC2020 numerical optimization competition are challenging, as presented in Table 1. Function 1 has a single peak, Functions 2–4 are basic functions, Functions 5–7 are mixed functions, and Functions 8–10 are compound functions.

The trial will be conducted with Dim = 20, a group of 50, and 1000 turns. Figure 4 displays the trend line for CEC2020, while Table 2 presents the outcomes from 30 trials.

Table 1. The CEC2020 benchmark function.

	No.	Functions
Unimodal Function	1	Shifted and Rotated Bent Cigar Function
Basic Functions	2	Shifted and Rotated Schwefel's Function
	3	Shifted and Rotated Lunacek bi-Rastrigin Function
	4	Expanded Rosenbrock's plus Griewangk's Function
Hybrid Functions	5	Hybrid Function 1
	6	Hybrid Function 2
	7	Hybrid Function 3
Composition Functions	8	Composition Function 1
	9	Composition Function 2
	10	Composition Function 3

After 30 optimization runs, IBOA can achieve the minimal average value for unimodal function F1, showing significant advantages over other algorithms as revealed in Figure 4 and Table 2. To clearly compare the performance of various algorithms on the test functions,

this paper presents the minimum, maximum, and average values of each algorithm on the test functions in a line graph shown in Figure 5. The IBOA outperforms other algorithms on function F4, demonstrating comparable efficiency with GWO and PSO. However, challenges arise when dealing with functions F2–F3 and F6, with the IBOA tending to fall into local optima and underperforming, especially in F2, where a significant gap exists compared to other optimal values. The performance of IBOA is influenced by its initial parameters. In functions F2 and F3, the initial parameters may not be optimally set, resulting in slow convergence or insufficient accuracy. In function F6, the balance between global and local search in IBOA may be problematic, preventing it from finding better solutions within local optima. The performance of IBOA on function F9 is slightly inferior to other algorithms, though the difference is minimal. This may be due to IBOA’s slightly slower convergence rate on this function. In functions F5, F7, F8, and F10, IBOA achieved favorable results. Despite showing benefits in most cases in the performance testing of the CEC2020 function, the IBOA may face difficulties in escaping from local optimal solutions in specific functions. Moreover, IBOA may have limitations in scenarios requiring rapid convergence. IBOA exhibited outstanding performance in the tests of unimodal functions, hybrid functions, and composite functions, leading to the results. Despite not performing satisfactorily in the basic function tests of CEC2020, IBOA still ranked among the top three algorithms. On the other hand, the PSO demonstrated good performance in the basic function tests. While IBOA does not perform best in all test functions, it still shows excellent results in several, such as F1, F4, F5, F7, F8, and F10. This demonstrates that IBOA is reliable and effective in solving optimization problems.

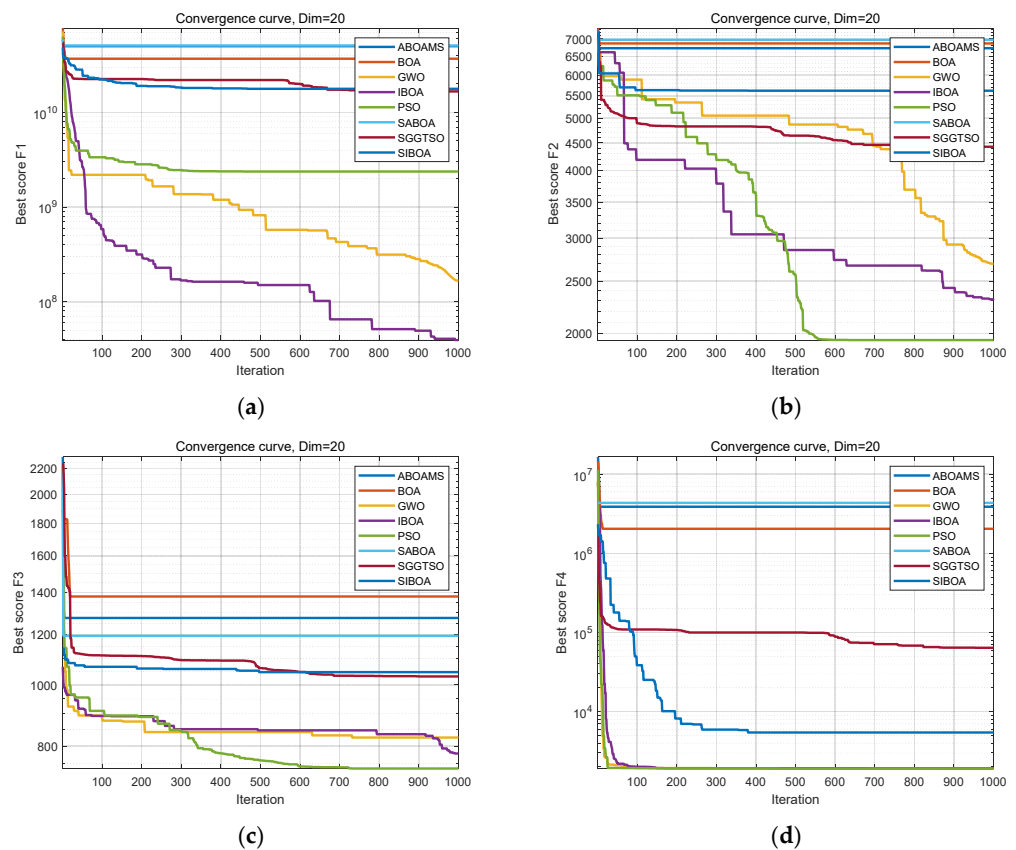


Figure 4. Cont.

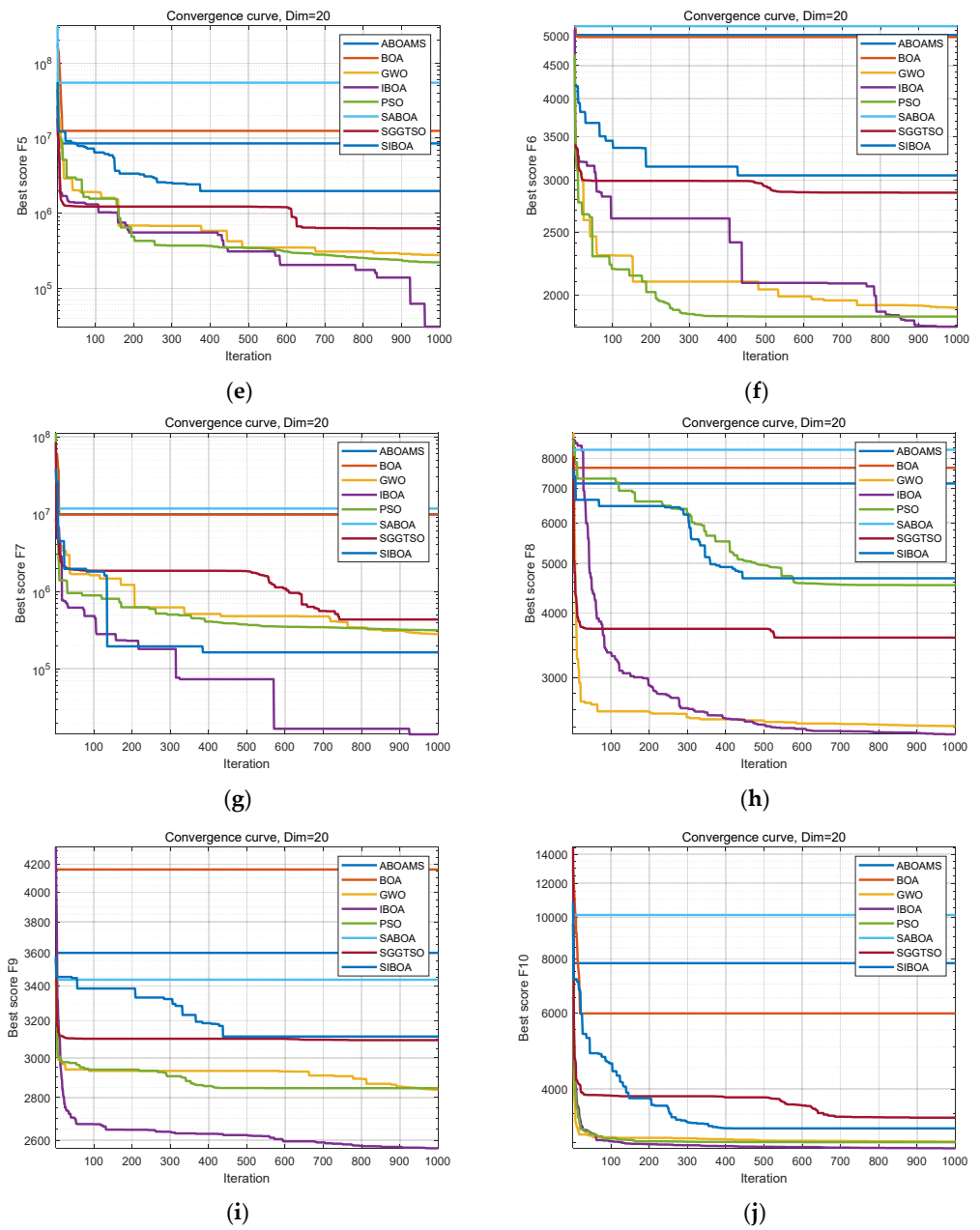


Figure 4. Comparison of convergence curves between ABOAMS, BOA, GWO, IBOA, PSO, SABOA, SGGTSSO, and SIBOA. (a–j) shows F1–F10.

Table 2. Outcome of the benchmark function evaluation.

Functions		ABOAMS	BOA	SABOA	SGGTSSO	SIBOA	GWO	PSO	IBOA
F1	Min	3.02×10^{10}	2.89×10^{10}	3.87×10^{10}	4.66×10^9	3.36×10^9	3.81×10^4	1.05×10^2	3.94×10^7
	Max	6.86×10^{10}	4.44×10^{10}	5.11×10^{10}	1.91×10^{10}	1.79×10^{10}	2.36×10^9	5.57×10^9	9.89×10^8
	Avg	4.21×10^{10}	3.61×10^{10}	4.97×10^{10}	1.20×10^{10}	1.04×10^{10}	3.03×10^8	9.16×10^8	1.69×10^8
F2	Min	6.21×10^3	5.34×10^3	5.30×10^3	3.01×10^3	4.79×10^3	1.60×10^3	1.38×10^3	2.31×10^3
	Max	7.34×10^3	7.39×10^3	7.62×10^3	5.12×10^3	5.93×10^3	4.53×10^3	2.59×10^3	4.78×10^3
	Avg	6.86×10^3	6.73×10^3	6.92×10^3	4.07×10^3	5.35×10^3	2.63×10^3	1.98×10^3	3.35×10^3

Table 2. Cont.

Functions		ABOAMS	BOA	SABOA	SGTSSO	SIBOA	GWO	PSO	IBOA
F3	Min	1.03×10^3	1.14×10^3	1.17×10^3	9.11×10^2	9.36×10^2	7.47×10^2	7.28×10^2	7.79×10^2
	Max	2.45×10^3	1.65×10^3	1.20×10^3	1.06×10^3	1.05×10^3	8.25×10^2	7.64×10^2	9.18×10^2
	Avg	1.24×10^3	1.38×10^3	1.19×10^3	9.87×10^2	9.97×10^2	7.76×10^2	7.45×10^2	8.32×10^2
F4	Min	3.17×10^5	2.91×10^5	5.48×10^5	3.66×10^3	2.97×10^3	1.90×10^3	1.90×10^3	1.90×10^3
	Max	1.52×10^7	3.86×10^6	2.67×10^7	2.73×10^5	2.54×10^5	2.41×10^3	1.27×10^4	1.99×10^3
	Avg	3.36×10^6	1.85×10^6	8.85×10^6	5.75×10^4	3.68×10^4	1.95×10^3	2.28×10^3	1.93×10^3
F5	Min	8.52×10^6	2.30×10^6	1.66×10^7	1.24×10^5	2.16×10^5	4.00×10^4	1.44×10^4	3.07×10^4
	Max	1.53×10^8	2.70×10^7	5.57×10^7	1.09×10^7	5.91×10^6	2.27×10^6	5.25×10^5	4.38×10^5
	Avg	4.18×10^7	1.28×10^7	4.96×10^7	2.48×10^6	1.46×10^6	6.41×10^5	1.91×10^5	1.52×10^5
F6	Min	3.30×10^3	3.48×10^3	3.72×10^3	2.20×10^3	2.54×10^3	1.64×10^3	1.66×10^3	1.79×10^3
	Max	5.30×10^3	5.55×10^3	6.38×10^3	3.23×10^3	3.26×10^3	2.29×10^3	2.07×10^3	2.59×10^3
	Avg	4.41×10^3	4.39×10^3	4.61×10^3	2.70×10^3	2.87×10^3	1.88×10^3	1.86×10^3	2.21×10^3
F7	Min	7.09×10^6	2.03×10^6	5.82×10^6	4.42×10^4	7.91×10^4	7.38×10^3	3.10×10^3	1.42×10^4
	Max	1.04×10^8	3.94×10^7	2.09×10^8	5.72×10^6	2.43×10^6	8.93×10^5	1.34×10^6	2.05×10^5
	Avg	3.84×10^7	1.45×10^7	6.72×10^7	9.59×10^5	5.13×10^5	2.02×10^5	1.49×10^5	6.16×10^4
F8	Min	6.13×10^3	6.04×10^3	6.64×10^3	2.82×10^3	2.90×10^3	2.26×10^3	2.30×10^3	2.33×10^3
	Max	8.93×10^3	8.73×10^3	9.69×10^3	6.18×10^3	4.68×10^3	6.63×10^3	4.90×10^3	2.42×10^3
	Avg	7.36×10^3	7.66×10^3	8.39×10^3	3.85×10^3	3.38×10^3	3.26×10^3	2.96×10^3	2.36×10^3
F9	Min	3.21×10^3	3.50×10^3	3.42×10^3	3.04×10^3	3.05×10^3	2.81×10^3	2.84×10^3	2.56×10^3
	Max	4.18×10^3	4.16×10^3	4.27×10^3	3.36×10^3	3.21×10^3	2.93×10^3	3.03×10^3	3.11×10^3
	Avg	3.71×10^3	3.79×10^3	3.73×10^3	3.15×10^3	3.13×10^3	2.86×10^3	2.91×10^3	2.96×10^3
F10	Min	4.82×10^3	5.78×10^3	9.38×10^3	3.20×10^3	3.24×10^3	2.91×10^3	2.91×10^3	2.81×10^3
	Max	1.60×10^4	1.01×10^4	1.14×10^4	4.54×10^3	4.14×10^3	3.05×10^3	3.05×10^3	2.97×10^3
	Avg	8.80×10^3	7.50×10^3	1.11×10^4	3.75×10^3	3.60×10^3	2.98×10^3	2.93×10^3	2.89×10^3

3.2. Dynamic Search Strategy Experiment

The proposed dynamic search strategy's effectiveness is verified by analyzing the balance between global search and local search in the IBOA. The line graph below illustrates the number of butterflies entering global and local searches throughout 500 iterations. Observing the trend of the line graph reveals that, throughout these iterations, the participation in global search gradually decreases from over twenty to single digits, indicating an overall downward trend. In contrast, participation in local searches displays an opposite trend, progressively increasing from single digits to over twenty times. Figure 6 demonstrates that the dynamic search strategy proposed can effectively facilitate intense global search in the initial stages of the algorithm and prioritize local search in the later stages. This observation serves as evidence for the efficacy of the approach.

3.3. Path Planning Experiment

This study utilizes a simulated environment featuring a hill with a peak elevation of 340 m for UAV path planning experiments. The environment is idealized, assuming the absence of tall trees and the presence of only low shrubs. Additionally, the UAV operates under clear weather conditions, eliminating the influence of weather and other external factors on its flight.

Figure 7 presents the comparison results with the PSO algorithm, GWO algorithm, improved tuna swarm optimization algorithm, and the BOA, assuming an iteration count of 500 and a population size of 30, with distance as the fitness calculation function. Additionally, Figure 8 illustrates the comparison results with other improved BOA, namely SABOA [27], SIBOA [28], and ABOAMS [29].

The diagram in Figure 7 compares the fitness levels achieved by different optimization algorithms. The IBOA stands out with the lowest fitness value of 1000, while other methods,

such as the GWO, optimized tuna swarm method, and original BOA, show fitness values around 2000, and the PSO exhibits the highest fitness value. The results indicate the superiority of the IBOA. This algorithm continuously updates its optimal value, leading to a consistent decrease in fitness value, demonstrating its ability to avoid local optima and identify the best solution. The experiments in this study involved route-design trials in a three-dimensional environment. Figure 7 illustrates the comparison of routes generated by various collective intelligence strategies. Among these routes, the IBOA stands out by producing the most optimal path. The terrain includes hilly obstacles between the start and endpoints. Other algorithms either struggle to navigate around the hills or take longer routes to avoid the barriers, failing to identify the most efficient path. In contrast, the IBOA proposed in this research effectively maneuvers through the mountainous terrain and determines the best path across the low-lying areas. This validation underscores the effectiveness and robustness of the proposed methodology.

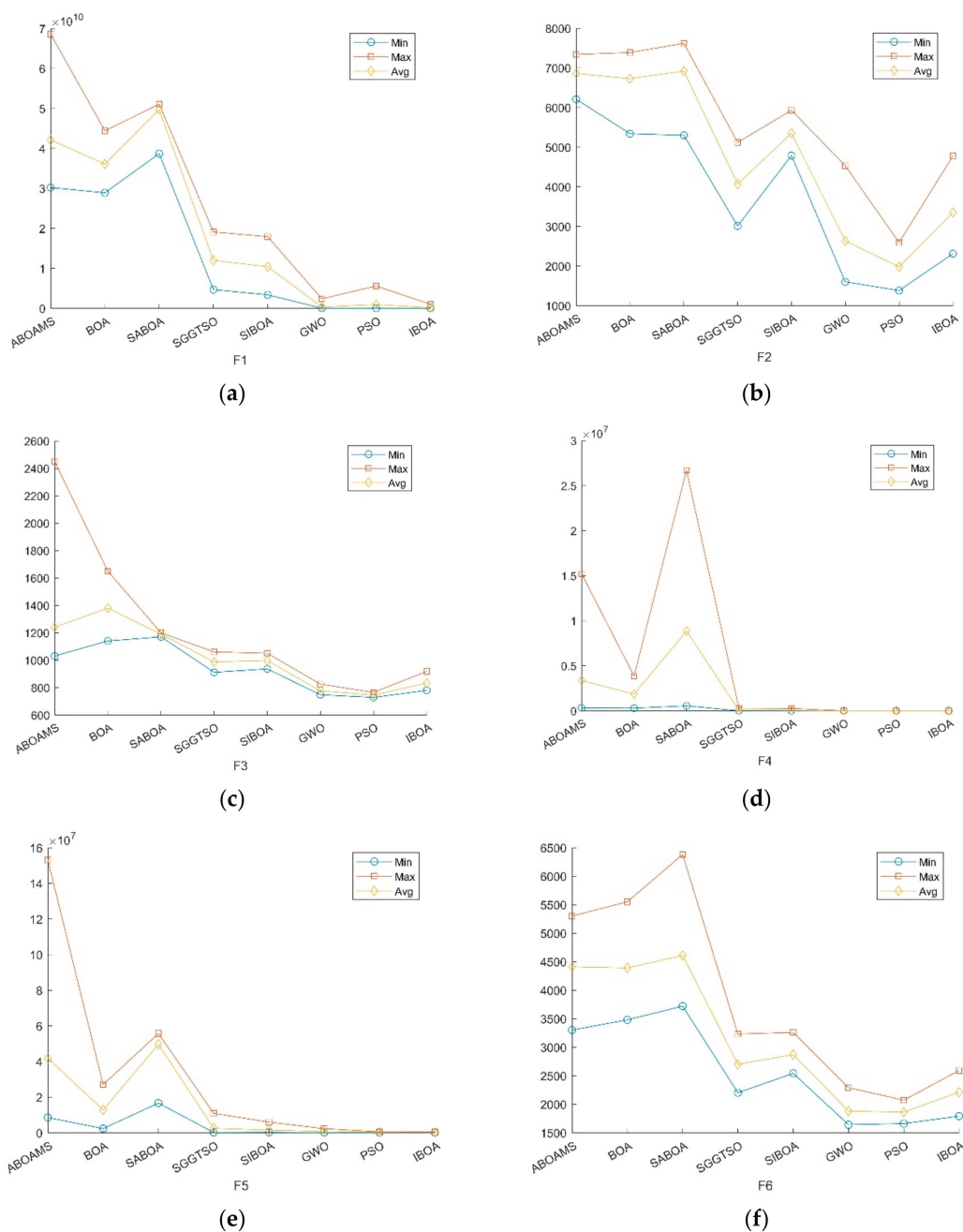


Figure 5. Cont.

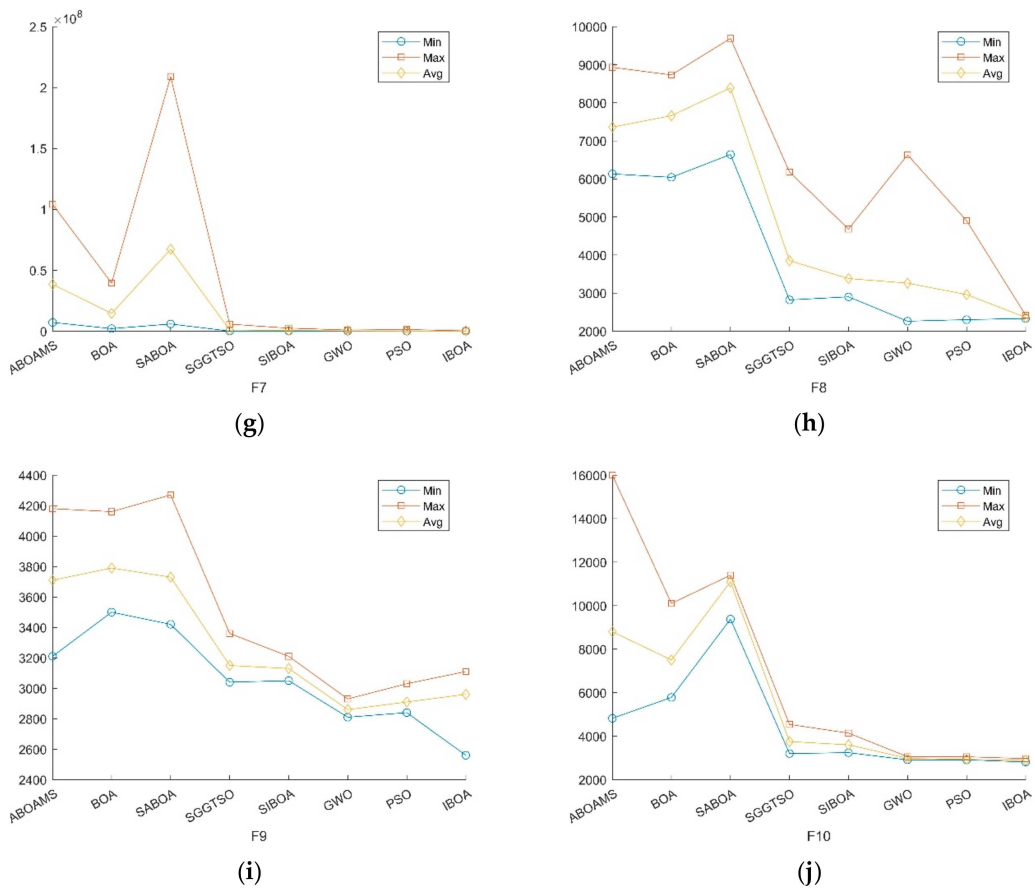


Figure 5. Figures (a–j) compare the maximum, minimum, and average values of ABOAMS, BOA, GWO, IBOA, PSO, SABOA, SGGTSSO, and SIBOA across test functions F1–F10.

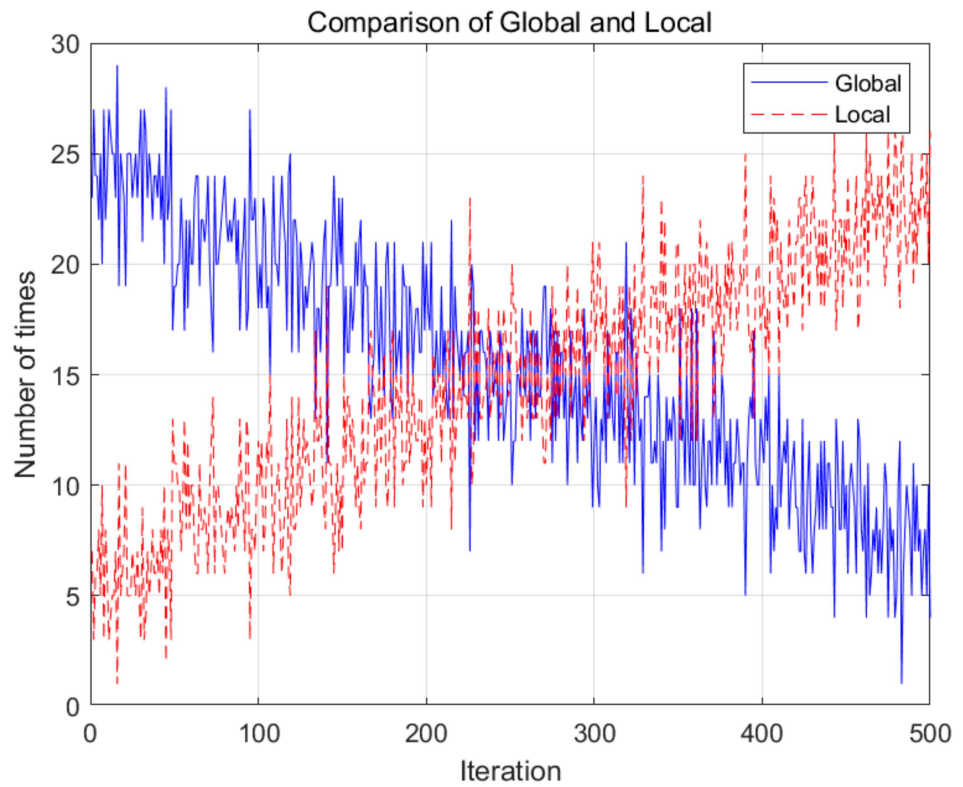


Figure 6. Comparison of Global and Local Search Frequencies.

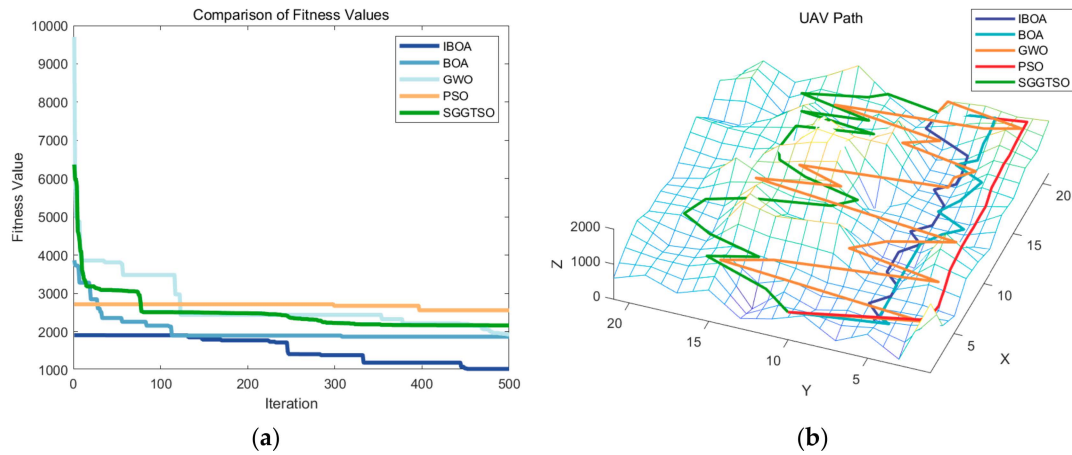


Figure 7. Comparison result 1. (a) Fitness value; (b) Path.

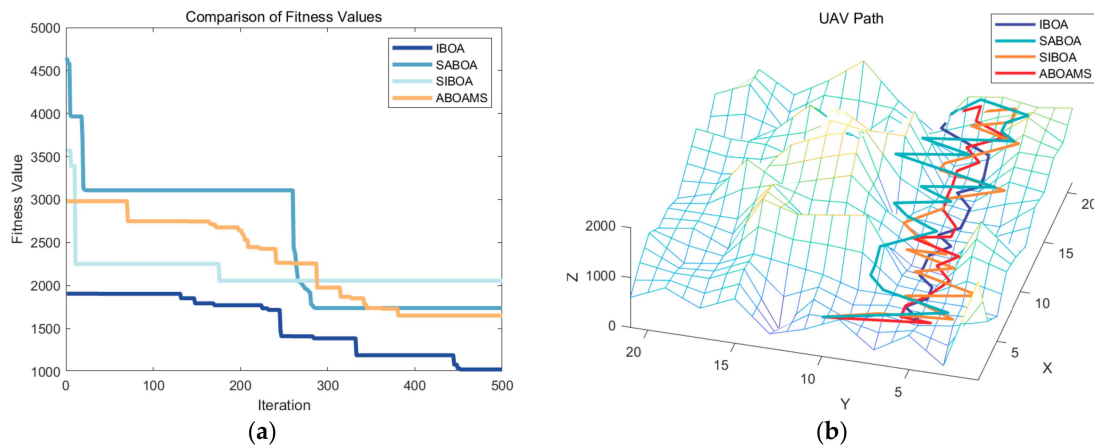


Figure 8. Comparison result 1. (a) Fitness value; (b) Path.

The manuscript conducted comparative trials between the IBOA and other upgraded BOA, with the results illustrated in Figure 8. As depicted in Figure 8, the comparison of fitness values demonstrates the superior performance of the IBOA over other enhanced BOAs in the context of UAV route design. Observing the path comparison chart in Figure 8, it becomes apparent that the IBOA can efficiently converge the UAV’s flight path into a valley, ensuring a secure trajectory for the UAV. In contrast, the alternative algorithms fail to converge to the optimal value, leading to the UAV flying at higher altitudes and increasing the risk of detection. In contrast, the IBOA proposed in this paper enables the UAV to navigate at lower altitudes, consequently enhancing the safety of the UAV throughout its flight.

3.4. Time Complexity

Since the algorithm in this study does not require large models and a huge number of parameters, its computational cost is relatively low and the time it consumes is also minimal, primarily depending on the number of iterations. Table 3 shows the time taken for each algorithm to plan paths after 500 iterations.

Table 3. Comparison of time complexity.

ABOAMS	BOA	SABOA	SGGTSO	SIBOA	GWO	PSO	IBOA
6.57 s	6.34 s	6.78 s	1.34 s	6.48 s	1.44 s	6.4 s	6.48 s

Although the GWO and SGGTSO algorithms exhibit shorter run times than the algorithms mentioned in the study based on the results shown in Figures 7 and 8 they fail to identify the optimal path in the simulation environment. In contrast, the algorithms mentioned in the study demonstrate a time advantage by being able to find the optimal path in the simulation environment. Therefore, it can be argued that our algorithm possesses a time efficiency advantage, allowing for rapid convergence to the optimal solution.

4. Conclusions

This manuscript addresses the concern of UAV route design by utilizing the BOA, a crucial component of UAV self-governance. The customary BOA encounters issues with tardy convergence velocity and vulnerability to regional optimal solutions, which present certain shortcomings in addressing UAV route design complications. To address this, IBO is proposed to enhance its performance and solve UAV path planning problems. To ensure the population exhibits good diversity and to avoid premature algorithm convergence, the initial butterfly population is initialized using ICMIC mapping to ensure it starts with a favorable initial state. Moreover, a population restart strategy is implemented to remove less capable individuals and improve the algorithm's capacity to break free from local optima while keeping the population size constant. In addition, drawing inspiration from the traits of particle flock enhancement, the butterfly's position update mechanism is enhanced to expedite the algorithm's convergence speed. The BOA integrates a dynamic exploration approach, enabling it to engage in wider-ranging initial exploration and shift towards more localized exploration as the search progresses with the ultimate goal of pinpointing the most optimal global solution. The IBOA is compared against the PSO algorithm, GWO algorithm, and various enhanced BOA. The findings indicate that the IBOA exhibits superior efficacy in resolving UAV path planning issues, as the optimal path can be rapidly achieved to solve UAV path planning problems effectively. Conversely, various algorithms struggle to locate the most efficient route due to particular deficiencies, such as the inability to avoid terrain obstacles and maintain suboptimal flight altitudes. Therefore, the IBOA is capable of selecting an optimal path for UAVs.

In the real world, UAVs often need to perform tasks in dynamic environments and collaborate with other UAVs. Therefore, future research will concentrate on addressing these more challenging problems and further optimizing the algorithms. It is essential to acknowledge that the present suggested approach primarily pertains to the path arrangement issue of a lone UAV within an unchanging surrounding.

First and foremost, in the path planning problem of UAVs in dynamically changing environments, the consideration of moving obstacles is crucial. These obstacles may include other aircraft, vehicles, and buildings, among others. In such contexts, navigation algorithms are required to detect and avoid dynamic challenges to ensure the safety and efficient maneuvering of the UAV. Moreover, concerning multi-UAV collaborative path planning, the focus shifts to enabling multiple UAVs to collaborate and coordinate effectively in completing complex tasks. This involves establishing communication channels and sharing information among UAVs, as well as devising optimal collaboration strategies and path-planning schemes. Through the investigation and enhancement of the collective intelligence and coordination mechanisms within UAV networks, the efficiency and performance of collaborative UAV systems can be enhanced. Furthermore, the ongoing optimization of algorithms will be guided by practical application demands and specific scenarios. Special attention will be given to improving the real-time capabilities, stability, and scalability of algorithms to meet the swift decision-making requirements of UAVs as they navigate intricate environments. The effectiveness of these algorithms will be empirically assessed and validated in practical situations to affirm their applicability and reliability.

In conclusion, to enhance the effectiveness of UAV applications in various domains and complex tasks, future research should focus on dynamic environment-based UAV path planning, multi-UAV cooperative path planning, and algorithm optimization aligned with practical considerations. By addressing these key areas, researchers can develop

more viable and efficient solutions that cater to a broader range of applications and tasks involving UAVs.

Author Contributions: Conceptualization, L.W. and X.Z.; methodology, L.W.; software, X.Z.; validation, L.W., X.Z. and H.Z.; formal analysis, L.W.; investigation, C.W.; resources, L.W.; data curation, Q.G.; writing—original draft preparation, L.W.; writing—review and editing, T.Z.; visualization, Z.L.; supervision, L.W.; project administration, J.S.; funding acquisition, L.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (62376165) and the Liaoning Provincial Department of Education Project (longitudinal 20240188).

Data Availability Statement: The data used in this study are not public but available upon request.

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

References

- Li, J.; Xiong, X.; Yang, Y. A Method of UAV Navigation Planning Based on ROS and Improved A-star Algorithm. In Proceedings of the 2023 CAA Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS), Yibin, China, 22–24 September 2023; IEEE: New York, NY, USA, 2023; pp. 1–5.
- Hu, J.; Xie, K. Path Planning Algorithm for UAV Based on Smooth Rapidly Exploring Random Tree. In Proceedings of the 2022 International Conference on Human Machine Interaction, Beijing, China, 6–8 May 2022; pp. 79–83.
- Qi, D.; Zhang, Z.; Zhang, Q. Path planning of multirotor UAV based on the improved ant colony algorithm. *J. Robot.* **2022**, *2022*, 2168964. [[CrossRef](#)]
- Li, H.; Lv, T.; Shui, Y.; Zhang, J.; Zhang, H.; Zhao, H.; Ma, S. An Improved grey wolf optimizer with weighting functions and its application to Unmanned Aerial Vehicles path planning. *Comput. Electr. Eng.* **2023**, *111*, 108893. [[CrossRef](#)]
- Hao, G.; Lv, Q.; Huang, Z.; Zhao, H.; Chen, W. UAV Path Planning Based on Improved Artificial Potential Field Method. *Aerospace* **2023**, *10*, 562. [[CrossRef](#)]
- Liu, H. A novel path planning method for aerial UAV based on improved genetic algorithm. In Proceedings of the 2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2–4 February 2023; IEEE: New York, NY, USA, 2023; pp. 1126–1130.
- Zhang, C.; Zhou, W.; Qin, W.; Tang, W. A novel UAV path planning approach: Heuristic crossing search and rescue optimization algorithm. *Expert Syst. Appl.* **2023**, *215*, 119243. [[CrossRef](#)]
- He, Y.; Wang, M. An improved chaos sparrow search algorithm for UAV path planning. *Sci. Rep.* **2024**, *14*, 366. [[CrossRef](#)]
- Yu, X.; Jiang, N.; Wang, X.; Li, M. A hybrid algorithm based on grey wolf optimizer and differential evolution for UAV path planning. *Expert Syst. Appl.* **2023**, *215*, 119327. [[CrossRef](#)]
- Chen, B.; Yang, J.; Zhang, H.; Yang, M. An improved spherical vector and truncated mean stabilization based bat algorithm for uav path planning. *IEEE Access* **2023**, *11*, 2396–2409. [[CrossRef](#)]
- Sonny, A.; Yeduri, S.R.; Cenkeramaddi, L.R. Autonomous UAV path planning using modified PSO for UAV-assisted wireless networks. *IEEE Access* **2023**, *11*, 70353–70367. [[CrossRef](#)]
- Meng, K.; Chen, C.; Wu, T.; Xin, B.; Liang, M.; Deng, F. Evolutionary State Estimation-Based Multi-Strategy Jellyfish Search Algorithm for Multi-UAV Cooperative Path Planning. *IEEE Trans. Intell. Veh.* **2024**, *early access*. [[CrossRef](#)]
- Jiao, K.; Chen, J.; Xin, B.; Li, L.; Zheng, Y.; Zhao, Z. Three-dimensional path planning with enhanced gravitational search algorithm for unmanned aerial vehicle. *Robotica* **2024**, *42*, 2453–2487. [[CrossRef](#)]
- Cheng, Y.; Li, D.; Wong, W.E.; Zhao, M.; Mo, D. Multi-UAV collaborative path planning using hierarchical reinforcement learning and simulated annealing. *Int. J. Perform. Eng.* **2022**, *18*, 463.
- Shi, H.; Zhao, Z.; Chen, J.; Zhou, M.; Liu, Y. Enhancing UAV Path Planning in Multi-Agent Reinforcement Learning through Adaptive Dimensionality Reduction. *Preprints* **2024**. [[CrossRef](#)]
- Xi, M.; Dai, H.; He, J.; Li, W.; Wen, J.; Xiao, S.; Yang, J. A lightweight reinforcement learning-based real-time path planning method for unmanned aerial vehicles. *IEEE Internet Things J.* **2024**, *11*, 21061–21071. [[CrossRef](#)]
- Luo, X.; Wang, Q.; Gong, H.; Tang, C. UAV path planning based on the average TD3 algorithm with prioritized experience replay. *IEEE Access* **2024**, *12*, 38017–38029. [[CrossRef](#)]
- Arora, S.; Singh, S. Butterfly optimization algorithm: A novel approach for global optimization. *Soft Comput.* **2019**, *23*, 715–734. [[CrossRef](#)]
- Sharma, S.; Chakraborty, S.; Saha, A.K.; Nama, S.; Sahoo, S.K. mLBOA: A modified butterfly optimization algorithm with lagrange interpolation for global optimization. *J. Bionic Eng.* **2022**, *19*, 1161–1176. [[CrossRef](#)]
- Sharma, S.; Saha, A.K.; Roy, S.; Mirjalili, S.; Nama, S. A mixed sine cosine butterfly optimization algorithm for global optimization and its application. *Clust. Comput.* **2022**, *25*, 4573–4600. [[CrossRef](#)]
- Li, Y.; Yu, X.; Liu, J. An opposition-based butterfly optimization algorithm with adaptive elite mutation in solving complex high-dimensional optimization problems. *Math. Comput. Simul.* **2023**, *204*, 498–528. [[CrossRef](#)]

22. Sharma, S.; Saha, A.K.; Majumder, A.; Nama, S. MPBOA-A novel hybrid butterfly optimization algorithm with symbiosis organisms search for global optimization and image segmentation. *Multimed. Tools Appl.* **2021**, *80*, 12035–12076. [[CrossRef](#)]
23. Arora, S.; Singh, S. An improved butterfly optimization algorithm with chaos. *J. Intell. Fuzzy Syst.* **2017**, *32*, 1079–1088. [[CrossRef](#)]
24. Guo, Y.; Liu, X.; Chen, L. Improved butterfly optimisation algorithm based on guiding weight and population restart. *J. Exp. Theor. Artif. Intell.* **2021**, *33*, 127–145. [[CrossRef](#)]
25. Gao, W.X.; Liu, S.; Xiao, Z.Y.; Yu, J.F. Butterfly optimization algorithm based on Cauchy variation and adaptive weight. *Comput. Eng. Appl.* **2020**, *56*, 43–50.
26. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995.
27. Wang, Y.R.; Zhang, D.M. Butterfly optimization algorithm combining sine cosine and iterative chaotic map with infinite collapses. *Pattern Recognit. Artif. Intell.* **2020**, *33*, 660–669.
28. Fan, Y.; Shao, J.; Sun, G.; Shao, X. A self-adaption butterfly optimization algorithm for numerical optimization problems. *IEEE Access* **2020**, *8*, 88026–88041. [[CrossRef](#)]
29. Liu, K.; Dai, Y. Adaptive butterfly optimization algorithm based on mutation strategies. *Appl. Res. Comput./Jisuanji Yingyong Yanjiu* **2022**, *39*. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.