axioms

Article

# UAV Path Planning Based on an Improved Chimp Optimization Algorithm

**Qinglong Chen, Qing He * and Damin Zhang**

College of Big Data and Information Engineering, Guizhou University, Guiyang 550025, China; lqc327@gmail.com (Q.C.); dmzhang@gzu.edu.cn (D.Z.)
* Correspondence: qhe@gzu.edu.cn

**Abstract:** Path planning is one of the key issues in the research of unmanned aerial vehicle technology. Its purpose is to find the best path between the starting point and the destination. Although there are many research recommendations on UAV path planning in the literature, there is a lack of path optimization methods that consider both the complex flight environment and the performance constraints of the UAV itself. We propose an enhanced version of the Chimp Optimization Algorithm (TRS-ChOA) to solve the UAV path planning problem in a 3D environment. Firstly, we combine the differential mutation operator to enhance the search capability of the algorithm and prevent premature convergence. Secondly, we use improved reverse learning to expand the search range of the algorithm, effectively preventing the algorithm from missing high-quality solutions. Finally, we propose a similarity preference weight to prevent individuals from over-assimilation and enhance the algorithm's ability to escape local optima. Through testing on 13 benchmark functions and 29 CEC2017 complex functions, TRS-ChOA demonstrates superior optimization capability and robustness compared to other algorithms. We apply TRS-ChOA along with five well-known algorithms to solve path planning problems in three 3D environments. The experimental results reveal that TRS-ChOA reduces the average path length/fitness value by 23.4%/65.0%, 8.6%/81.0%, and 16.3%/41.7% compared to other algorithms in the three environments, respectively. This indicates that the flight paths planned by TRS-ChOA are more cost-effective, smoother, and safer.

**Keywords:** path planning; UAV; chimp optimization algorithm; 3D environment

check for updates

**Citation:** Chen, Q.; He, Q.; Zhang, D. UAV Path Planning Based on an Improved Chimp Optimization Algorithm. *Axioms* **2023**, *12*, 702. https://doi.org/10.3390/axioms12070702

Academic Editors: Jorge Mario Cruz-Duarte and Nelishia Pillay

Received: 18 May 2023
Revised: 13 July 2023
Accepted: 16 July 2023
Published: 19 July 2023

## 1. Introduction

With the continuous progress of technology, more and more automated machines appear in the daily lives of individuals, bringing many conveniences. Unmanned aerial vehicles (UAVs) are the products of a deep integration of aviation technology and information technology (IT) [1] and are also a representative of high automation equipment. Since the earliest UAV was developed and applied in the military field, people have gradually realized that this technology will have a significant impact on the economy and development of human society in the future. Today, UAVs equipped with various small intelligent sensors have been widely popularized and used [2]. They assist or replace people in completing various tasks and have become an important means for obtaining spatial data [3]. Path planning is always a key issue in the research of UAV technology. Its main goal is to generate a flight path that connects the initial state and the target state while satisfying the required constraints [4]. In reality, the environmental model containing a variety of landforms and other obstacles is very complex, which not only increases the diversity of UAV flight paths but also increases the computational cost of path planning. Obstacles may divide the flight environment into multiple spaces, and path planning under such conditions is a non-convex problem because the search space has a larger scope and there may be multiple extreme points, making it very difficult to find the optimal path [5].

In the past, dynamic planning [6] and geometric algorithms were commonly used to plan flight paths for UAVs, such as the Measurement Assisted Dynamic Planning (MAD-P) algorithm [7], which measures UAV throughput in real-time through the network and completes autonomous allocation of positions during flight. This method converts the path planning problem into a numerical cost minimization problem, which facilitates subsequent solutions and calculations. However, it ignores many constraints in the actual flight process of the UAV, such as terrain, altitude, etc., resulting in certain differences between the experimental results and the actual cost. Geometric algorithms rely heavily on cost graphs [8], which increase the time cost of the algorithm. When dealing with complex problems, it is difficult to meet requirements of efficiency. In addition, researchers have proposed methods such as the A-star planner, the Theta-star planner, the Dijkstra planner, and the Voronoi diagram for the path planning of autonomous vehicles [9–13]. However, these methods suffer from high computational loads and difficulty generating feasible paths in complex 3D environments. Kim [14] proposed the use of virtual proxies and virtual sensors to quickly generate three-dimensional paths to the target point, effectively reducing the computation load of the path planner. However, the obtained paths are generally only locally optimal. To address the shortcomings of previous methods, researchers have proposed the computational intelligence (CI) method [15]. CI uses numerical methods to simulate and implement human intelligence, which has good adaptive ability and strong global search ability. In recent years, more and more research has focused on CI methods for solving UAV path planning problems.

The swarm intelligence optimization algorithm is an important branch of CI that originated from the study of various insect or animal swarm behaviors. By summarizing the intelligent characteristics existing in their series of social behaviors, corresponding algorithm models are established. Common swarm intelligence optimization algorithms include particle swarm optimization (PSO) [16], gray wolf optimization (GWO) [17], cuckoo search (CS) [18], sparrow search algorithm (SSA) [19], whale optimization algorithm (WOA) [20], ant lion optimization (ALO) [21], and Tunicate Swarm Algorithm (TSA) [22]. Swarm intelligence algorithms not only have advantages such as simple structure, few parameters, and no need for gradient information, but also possess self-organizing patterns, which provide them with good robustness in solving path planning problems. This enables them to adapt well to dynamic environments, making them highly suitable for optimizing UAV systems [23]. Roberge et al. [24] used particle swarm optimization algorithms to calculate the optimal trajectory of UAVs in a three-dimensional environment. Wen et al. [25] proposed a three-dimensional UAV deployment scheme based on heuristic algorithms to ensure the connectivity of UAV networks in static and dynamic scenarios. Guan et al. [26] proposed an algorithm based on double-ant colonies to speed up path planning. Chai et al. [27] proposed a multi-strategy fusion differential evolution algorithm (MSFDE), which converts environmental constraints and UAV physical characteristics into targets through a linear weighted summation method. Zhang et al. [28] proposed an improved Harris-Hawkes UAV path planning algorithm (HHO) to expand the search space by introducing Cauchy mutation strategies during exploration. Ji et al. [29] proposed a learning particle swarm optimization algorithm based on dual-dynamic biogeography (DDBLPSO) for solving UAV flight paths in large-scale environments. Qu et al. [30] combined the simplified gray wolf optimizer (SGWO) with modified symbiotic organisms search (MSOS) to improve the capability of UAV path development. Shin et al. [5] proposed an improved particle swarm optimization (IPSO) for designing UAV paths in environments containing unknown threats. Qu et al. [31] proposed a Grey Wolf Optimization Algorithm based on reinforcement learning (RLGWO) to optimize and adjust UAV paths. Yu et al. [32] proposed an adaptive selection mutation-constrained differential evolution algorithm to solve the path of UAV rescue in disasters. Jiang et al. [33] proposed an algorithm based on the Partially Observable Markov Decision Process (POMDP) and Grey Wolf Optimizer (GWO), which enhances the anti-collision performance of UAVs during flight. Du et al. [34] improved the Chimp Optimization Algorithm by incorporating the somersault foraging strategy and applying it to path planning in a grid-based three-dimensional space. However, there are problems with these methods, such as insufficient consideration of terrain factors and computational complexity that exceeds the carrying capacity of the UAV. How to enhance the global optimization capability of UAVs, replace local path planning, and reduce the computational complexity of this process is still a topic worth discussing.

Khishe et al. proposed the Chimp Optimization Algorithm (ChOA) in 2020 [35], inspired by the individual intelligence and sexual motivation of chimps in their group hunting. ChOA divides individuals in the population into separate groups and assigns different update strategies to each group, enabling them to play different roles in the "hunting" process and enhancing the diversity of individuals in the algorithm. Therefore, ChOA exhibits stronger global search capabilities and robustness. In recent years, ChOA has been increasingly used by researchers, confirming its outstanding performance in solving nonlinear, multivariable, and multimodal function optimization problems. Kaur et al. [36] used the improved ChOA based on sine and cosine functions to solve engineering optimization problems. Hu et al. [37] used ChOA to stabilize extreme learning machines to improve the reliability of deep networks. Housein et al. [38] proposed an improved ChOA based on Lévy Flight, which is used for multi-level threshold segmentation of thermal imaging breast cancer images. In these studies, due to the different constraints and objective functions of the problem, the improvement strategies proposed by researchers are targeted, making the algorithm tendentious and no longer applicable to all complex optimization problems. Path planning for UAVs in a 3D environment is a complex, multi-constraint combinatorial optimization problem. When carrying out path planning, we should not only find the shortest path but also consider energy consumption, path safety, and other issues. Therefore, we must make a reasonable trade-off between multiple objectives. The unique strategy of ChOA endows it with stronger robustness and global search ability and has enormous potential for solving path planning problems. Therefore, we propose a ChOA (TRS-ChOA) that integrates multiple strategies for improvement and uses TRS-ChOA to solve path planning problems in the 3D environment we constructed.

In summary, the main contributions of this article are as follows:

(1) We established 3D environment models for UAV trajectory planning, covering different terrains or buildings such as plains, mountains, hills, and human engineering.
(2) The path length, flight altitude, and angle loss during the flight of UAVs were considered, which constituted the comprehensive evaluation index of path planning. The cubic spline interpolation method is used to smooth the trajectory of UAVs to solve the problem of low accuracy of interpolation points in B-spline curves [39].
(3) TRS-ChOA: To solve the 3D UAV path planning problem, we propose an enhanced version of the original ChOA based on differential evolution, improved reverse learning, and similarity preference weights.
(4) The optimization performance of TRS-ChOA is verified by the benchmark test function and the CEC2017 complex test function.
(5) Several well-known meta-heuristic methods are compared with the proposed TRS-ChOA in different 3D environments.

The rest of this article is organized as follows: In Section 2, we built the 3D environment required for the experiment. Section 3 introduces the basic ChOA. Section 4 puts forward three strategies to improve ChOA. In Section 5, the optimization performance of TRS-ChOA is verified by the test function. In Section 6, TRS-ChOA and a variety of algorithms are tested in a UAV trajectory planning model.

## 2. UAV Path Planning Problem Model

### *2.1. Background*

In recent years, UAVs have shown great potential for autonomous sensing [40]. UAV systems have been widely used to map, monitor, search, and track target individuals or areas. Compared to manned aircraft, UAVs can more efficiently perform repetitive and dangerous tasks, such as delivering supplies to disaster areas, exploring unknown areas, and so on. UAVs often face complex, three-dimensional environments when performing tasks. How to plan a UAV trajectory in three-dimensional space is a multi-constraint combinatorial optimization problem that needs to consider the threats brought by the environment and the performance limitations of the UAV. Therefore, it is crucial to construct an appropriate cost function around these problems to solve the optimal UAV flight trajectory. The usual modeling flow of the UAV path planning problem is shown in Figure 1. As can be seen from Figure 1, the UAV trajectory planning process can be divided into two major parts, namely, environmental modeling and track modeling. Finally, the smoothed trajectory data is uploaded to the UAV control system to realize the autonomous flight of the UAV. To simplify the modeling process, we make the following assumptions: (a) The UAV is regarded as a particle during the experiment and always flies at a uniform speed. (b) The terrain and dangerous areas of the target area can be detected and mapped by satellites.

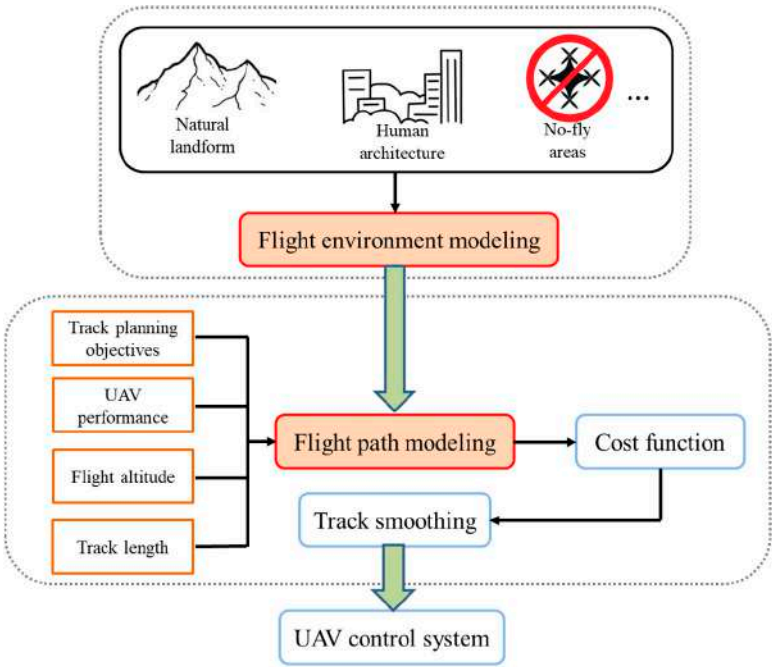

**Figure 1.** Flow chart of UAV flight path planning.

## 2.2. *Environmental Model*

Environmental modeling refers to the process of abstracting mathematical models from the actual environment that can be processed by planning algorithms [41]. Compared to environmental modeling for 2D path planning, more topographic factors need to be considered in 3D environments, such as mountains, hills, and plains. These environmental challenges require UAVs to have more coping strategies during flight. To restore the real geomorphic features as much as possible, this paper uses the method of function simulation to construct the plain landform. The plains are characterized by flat terrain without excessive fluctuations. Equation (1) shows the expression for the simulation function.

$$z(x,y) = \sin(y+a) + b\sin(x) + c\cos(d\sqrt{y^2+x^2}) + e\cos(y) + f\sin(f\sqrt{y^2+x^2}) + g\cos(y) \tag{1}$$

where $x$, $y$, and $z$ represent the coordinate values of a point in three-dimensional space. Generally, the plane composed of the $x$-axis and the $y$-axis is used as the ground, and $z$ represents the height. In Equation (1), $a$, $b$, $c$, $d$, $e$, $f$, and $g$ are all random numbers between [0, 3], which makes the constructed plain terrain more random. In order to increase the challenge and authenticity of the environment, we superimpose the mountain model on the plain terrain. Equation (2) shows the mathematical expression of the mountain model.

$$h(x,y) = \sum_i h_i \exp[-\frac{x-x_{oi}}{a_i^2} - \frac{y-y_{oi}}{b_i^2}] + h_o \tag{2}$$

where $h_o$ represents the height of the base terrain and $h_i$ represents the height of the $i$-th mountain. ($x_{oi}$, $y_{oi}$) represents the center coordinate of the $i$-th mountain. $a_i$ and $b_i$ are the slopes of the $i$-th mountain along the $x$-axis and $y$-axis directions, respectively. By setting the values of $h_i$, $a_i$, and $b_i$, we can get different mountains or hills, providing a more complex test environment for the UAV. Equation (3) is the expression of environmental height after superimposing the mountain model.

$$H(x, y) = \max[z(x, y), h(x, y)] \tag{3}$$

In reality, UAVs have to deal with natural obstacles as well as human engineering threats. These threats include high-rise buildings, no-fly zones, and the surveillance range of enemy radar or air defense missiles. If the UAV does not avoid these areas in its path, there is a risk of a crash. To simplify the model, a cylinder with a radius of $r$ is used to represent the area covered by the radar or missile. The threat coefficient at the center of the cylinder is the highest and gradually decreases outward. In addition, we use regular polygonal prisms to represent human buildings. These threatening human projects are also superimposed on environmental models to verify whether the algorithm has obstacle avoidance capabilities.

### *2.3. Track Model*

The simplified UAV path planning task is similar to the Traveling Salesman problem (TSP) [42]. In TSP, salespeople are expected to travel through all urban nodes with the shortest travel distance, and path planning tasks also require UAVs to consume the least amount of power when reaching their destinations. Ergezer et al. [43] designed a 3D path-planning method for multiple UAVs based on the genetic algorithm. They let UAVs fly along the desired path as much as possible to achieve the goal of reducing the flight distance. However, when faced with complex flight environments, global optimization is more effective. The flight path planning of UAVs should not only consider terrain factors but also be constrained by their performance. The main performance indexes include the flight path length, flight height, minimum step size, steering cost, maximum climb angle, etc. These limitations increase the complexity of the optimal path and make it contain many different features. In order to make the model more consistent with the actual situation, this paper uses a more complex trajectory evaluation function to judge the pros and cons of the generated path.

The first consideration in evaluating a path is its length. Shorter flight paths can save more fuel and time for UAVs while reducing the probability of distress during flight. The flight path of the UAV refers to the distance it travels from its origin to its destination. Suppose a complete route contains $n$ nodes, where the distance between the $i$-th node and the ($i$ + 1)-th node is $D_i$, and the coordinates of the two nodes are ($x_i$, $y_i$, $z_i$) and ($x_{i+1}$, $y_{i+1}$, $z_{i+1}$), abbreviated as $N(i)$ and $N(i + 1)$. Equation (4) describes the length of the UAV's actual flight path, Dpath.

$$\begin{cases} D_i = \|N(i+1) - N(i)\|_2 \\ D_{path} = \sum\limits_{i=1}^{n-1} D_i \end{cases} \tag{4}$$

Natural landforms and human engineering on the flight path may cause UAVs to crash and fail to complete flight missions. Basada-Portas et al. [44] reduce the probability of distress during flight by sharing the optimal path between UAVs, but increase the computational cost of the system. Generally, the UAV will fly at a lower altitude to take advantage of the favorable terrain and avoid being exposed to enemy radar. However, a low flying altitude increases the chances of the UAV colliding with trees, mountains, and buildings. Frequent adjustments of the UAV's flight angle and altitude will also consume more fuel and increase the burden on the control system. Therefore, it is necessary to keep the UAV at a stable altitude as much as possible to avoid a bumpy flight path. Equation (5) shows the flight altitude model of the UAV.

$$\begin{cases} H_{UAV} = \sqrt{\frac{1}{n}\sum\limits_{i=0}^{n-1}(z(i)-\overline{z})^2} \\ \overline{z} = \frac{1}{n}\sum\limits_{i=0}^{n-1} z(i) \end{cases} \tag{5}$$

where $z(i)$ represents the flight altitude of the UAV at the $i$-th node. Steering is also involved in the flight of the UAV. Good steering performance makes the UAV more maneuverable, but too much steering may also cause instability in the UAV fuselage. Therefore, the steering angle of the UAV during flight should not be greater than the preset maximum value. We assume that the maximum steering Angle of the UAV is $\Phi$ and the current steering Angle is $\theta$. Equation (6) is the UAV's steering angle loss function.

$$\begin{cases} \cos\theta = \frac{s_i^T s_{i+1}}{|s_i||s_{i+1}|} \\ A_{loss} = \sum\limits_{i=1}^{n}(\cos(\Phi - \cos\theta)) \end{cases} \tag{6}$$

where $s_i$ represents the segment vector of the $i$-th segment trajectory and $|s_i|$ represents its length. Through separate discussions of the three aspects mentioned above, we obtain the loss function for UAV path planning as shown in Equation (7).

$$\begin{cases} C_{loss} = \alpha_1 D_{path} + \alpha_2 H_{UAV} + \alpha_3 A_{loss} \\ \sum\limits_{i=1}^{3} \alpha_1 = 1 \\ \alpha_i \geq 0 \end{cases} \tag{7}$$

$C_{loss}$ is the total loss function, and $\alpha_1$, $\alpha_2$, and $\alpha_3$ represent the weight. In this paper, we set $\alpha_1$ = 0.4, $\alpha_2$ = 0.4, and $\alpha_3$ = 0.2. After effective processing of $C_{loss}$, a flight path composed of line segments is obtained, but this trajectory is only theoretically feasible. We use cubic spline interpolation instead of the B-spline curve method to smooth the trajectory so as to improve the accuracy of interpolation points.

Considering the two-dimensional case, assume that there are $n$ + 1 nodes along the planned path, marked as ($x_0$, $y_0$), ($x_1$, $y_1$), ($x_2$, $y_2$), and ($x_n$, $y_n$). If linear fitting is used to connect these discrete points, the resulting function is not smooth enough. To smooth the line between two adjacent nodes, we use a cubic function to fit them, as shown in Equation (8).

$$y_i = a_i x^3 + b_i x^2 + c_i x + d_i,\ i = 0,\ 1,\ 2,\ \ldots, n-1 \tag{8}$$

where $a_i$, $b_i$, $c_i$, and $d_i$ represent the coefficient terms in the curve equation. When fitting a trajectory, it is required that the junction points be continuous and smooth, so the first and second derivatives of a cubic spline curve should also be continuous. For ease of illustration, suppose we need to fit three points $P_1$ ($x_1$, $y_1$), $P_2$ ($x_2$, $y_2$), and $P_3$ ($x_3$, $y_3$), marking the cubic function between $P_1$ and $P_2$ as $J$ and the cubic function between $P_2$ and $P_3$ as $K$, as shown in Equation (9).

$$\begin{cases} J = a_1 x^3 + b_1 x^2 + c_1 x + d_1 \\ K = a_2 x^3 + b_2 x^2 + c_2 x + d_2 \end{cases} \tag{9}$$

Points $P_1$ and $P_2$ are on the curve $J$, and points $P_2$ and $P_3$ are on the curve $K$. This relationship is shown in Equation (10).

$$\begin{cases} y_1 = a_1{x_1}^3 + b_1{x_1}^2 + c_1x_1 + d_1 \\ y_2 = a_1{x_2}^3 + b_1{x_2}^2 + c_1x_2 + d_1 = a_2{x_2}^3 + b_2{x_2}^2 + c_2x_2 + d_2 \\ y_3 = a_2{x_3}^3 + b_2{x_3}^2 + c_2x_3 + d_2 \end{cases} \tag{10}$$

Because the derivatives of the spline curve at the nodes are continuous, the first derivatives of the two cubic functions at point $P_2$ are equal. In addition, if the spline is a cubic spline with a free boundary, the second derivatives at the starting point and the ending point are also required to be continuous. Therefore, by calculating the second derivative of Equations (10) and (11), the following is obtained:

$$\begin{cases} 6a_1x_1 + 2b_1 = 0 \\ 6a_1x_2 + 2b_1 = 6a_2x_2 + 2b_2 \\ 6a_2x_3 + 2b_2 = 0 \end{cases} \tag{11}$$

By using Equations (10) and (11) and the coordinates of this set of points, the polynomial coefficients ($a_1$, $b_1$, $c_1$, $d_1$, $a_2$, $b_2$, $c_2$, $d_2$) of the two cubic splines can be determined through algebraic calculation. However, path-planning involves multiple segments, so matrix equations are often used to solve spline parameters. To calculate the coefficients ($a_\mathrm{i}$, $b_\mathrm{i}$, $c_\mathrm{i}$, $d_\mathrm{i}$) of each spline curve, first calculate the step length $h_i$ between two adjacent nodes, as shown in Equation (12).

$$h_i = x_{i+1} - x_i,\ i = 0,\ 1,\ 2,\ \ldots, n-1 \tag{12}$$

Substitute the path point and endpoint condition (if it is a free boundary, the endpoint condition in the cubic spline is ${y_i}'' = 0$) into Equation (13) to obtain the quadratic differential value $m_i$, thereby deriving the coefficients of the spline curve, as shown in Equation (14).

$$\begin{bmatrix} 1 & 0 & 0 & & \cdots & 0 \\ h_0 & 2(h_0+h_1) & h_1 & 0 & \cdots & 0 \\ 0 & h_1 & 2(h_1+h_2) & h_2 & 0 & 0 \\ 0 & 0 & h_2 & 2(h_2+h_3) & h_3 & \vdots \\ \vdots & & 0 & \ddots & 0 & \\ 0 & \cdots & 0 & \cdots & 2(h_{n-2}+h_{n-}) & h_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \\ \vdots \\ m_n \end{bmatrix} = 6 \begin{bmatrix} 0 \\ \frac{y_2-y_1}{h_1} - \frac{y_1-y_0}{h_0} \\ \frac{y_3-y_2}{h_2} - \frac{y_2-y_1}{h_1} \\ \cdots \\ \frac{y_n-y_{n-1}}{h_{n-1}} - \frac{y_{n-1}-y_{n-2}}{h_2} \\ 0 \end{bmatrix} \tag{13}$$

$$\begin{cases} a_i = y_i \\ b_i = \frac{y_{i+1}-y_i}{h_i} - \frac{h_i}{2}m_i - \frac{h_i}{6}(m_{i+1} - m_i) \\ c_i = \frac{m_i}{2} \\ d_i = \frac{m_{i+1}-m_i}{6h_i} \end{cases} \tag{14}$$

The corresponding spline function expression within each subinterval $x_\mathrm{i} \le x \le x_i + 1$ is shown in Equation (15).

$$f_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d(x - x_i)^3 \tag{15}$$

Considering that the derivation process involves numerical analysis and other foundations and is far from the topic of UAV path planning, this article omits the detailed derivation process. The complete derivation process of the cubic spline interpolation method can be found in Reference [45]. In 3D space, the equation of a curve is $z = f\ (x, y)$, and its parametric form is $x = x\ (t)$, $y = y\ (t)$, and $z = z\ (t)$. The original set of input points $\{(x_i, y_i, z_i)\}$ becomes $P_{o1} = \{(t_i, x_i)\}$, $P_{o2} = \{(t_i, y_i)\}$, and $P_{o3} = \{(t_i, z_i)\}$. $P_{o1}$, $P_{o2}$, and $P_{o3}$ satisfy the monotonicity of the spline. Set $t_i = 1, 2, \ldots, n$, do cubic spline interpolation on the three point sets, respectively, and calculate the corresponding $(x_i, y_i, z_i)$, which is the point set after uniform interpolation. By connecting these points, a smooth flight path can be obtained.

## 3. Chimp Optimization Algorithm (ChOA)

ChOA [35] is inspired by the predatory behavior of chimp populations. During hunting, chimps are divided into four groups according to their functions: attacker, driver, barrier, and chaser. The attacker is the overall leader of the hunt, assisted by the other three chimps. The hunting process of chimps can be divided into two steps, namely, Chasing prey and attacking prey, which correspond to the exploration and development stages of the algorithm. The mathematical model for these two steps is described below.

### *3.1. Driving and Chasing the Prey*

All chimpanzees participating in predation randomly change their position to approach the prey. Equations (16) and (17) describe this process.

$$D = \left| c \cdot x_{prey}(t) - m \cdot x_{chimp}(t) \right| \tag{16}$$

$$x_{chimp}(t+1) = x_{prey}(t) - a \cdot d \tag{17}$$

where, $D$ represents the distance between the chimp and the prey. $x_{chimp}$ is the chimp's position vector. $x_{prey}$ is the prey's position vector. $t$ represents the current iteration number. $a$, $m$, and $c$ are coefficient vectors. $a$, $m$, and $c$ are determined by Equations (18)–(20), respectively.

$$a = 2 \cdot f \cdot r_1 - f \tag{18}$$

$$c = 2 \cdot r_2 \tag{19}$$

$$m = Chaotic\_value \tag{20}$$

where the value of $f$ decreases from 2.5 to 0 during the iteration period. $r_1$ and $r_2$ are random vectors within the range of [0, 1]. $m$ is a chaotic vector calculated based on various chaotic maps, which represents the influence of sexual motivation on chimp during hunting.

### *3.2. Attacking Method*

When attacking prey, it is assumed that the first attacker (the best solution available), driver, barrier, and chaser are better informed about the location of potential prey, and other chimps are forced to update their positions according to the best chimps locations. Equations (21)–(23) describe this relationship.

$$\begin{cases} d_{Attacker} = |c_1 x_{Attacker} - m_1 x| \\ d_{Barrier} = |c_2 x_{Barrier} - m_2 x| \\ d_{Chaser} = |c_3 x_{Chaser} - m_3 x| \\ d_{Driver} = |c_4 x_{Driver} - m_4 x| \end{cases} \tag{21}$$

$$\begin{cases} V_1 = x_{Attacker} - a_1(d_{Attacker}) \\ V_2 = x_{Barrier} - a_2(d_{Barrier}) \\ V_3 = x_{Chaser} - a_3(d_{Chaser}) \\ V_4 = x_{Driver} - a_4(d_{Driver}) \end{cases} \tag{22}$$

$$x(t+1) = \frac{V_1 + V_2 + V_3 + V_4}{4} \tag{23}$$

where $d_{Attacker}$, $d_{Barrier}$, $d_{Chaser}$, and $d_{Driver}$ represent the distance between the four types of chimps and the prey in the current population. $x_{Attacker}$, $x_{Barrier}$, $x_{Chaser}$, and $x_{Driver}$ are their position vectors relative to the prey. $V_1$, $V_2$, $V_3$, and $V_4$ represent their position update vector. $x(t + 1)$ is the position of the $t + 1$ generation chimps. $a_1$~$a_4$, $m_1$~$m_4$, and $c_1$~$c_4$ are all coefficient vectors. After food satisfaction, the chimps would release hunting responsibilities, meaning they would no longer take on four hunting roles and instead scramble to get food. This chaotic behavior helps prevent the algorithm from falling below its local optimal value. Equation (24) is a mathematical model of this phenomenon.

$$x_{chimp}(t+1) = \begin{cases} x_{prey}(t) - a \cdot d, & if \mu < 0.5 \\ Chaotic\_value, & if \mu \geq 0.5 \end{cases} \tag{24}$$

where $\mu$ is a random number between [0, 1], and *Chaotic_value* represents the chaotic mapping.

## 4. Improved Chimp Optimization Algorithm (TRS-ChOA)

In this section, three improvements to ChOA were introduced. In order to improve ChOA's exploration ability and solve its premature convergence issue, this study integrated the mutation mechanism of differential evolution (DE) [46] into ChOA. To expand the search range of ChOA and enhance the diversity of individuals, an improved reverse learning mechanism was introduced. Finally, to prevent ChOA from getting stuck in local optima, a similarity preference weight was proposed to assist ChOA in position updates. The specific descriptions of these three improvement strategies are described below.

### *4.1. The Differential Evolution*

The basic idea of the differential evolution (DE) algorithm is to use vector differences to generate new individuals for optimizing the objective function. The differential operation is the core of DE, which generates new solution vectors (offspring) by mutating and crossing the current population. If the fitness value of an offspring is better than that of its corresponding parent, the offspring will replace its parent. The DE algorithm has the advantages of fast global search speed, simple implementation, and fewer algorithm parameters. Therefore, it has been successfully applied in optimizing various practical problems and has demonstrated good performance and robustness. Algorithm 1 shows the pseudocode of the DE algorithm.

In Algorithm 1, $x_i$ represents the individuals in the population, and $d$ is the dimension of the problem. $j_{\text{rand}}$ represents a random integer between [1, $d$]. $r_1$, $r_2$, and $r_3$ are individuals randomly selected from the population to participate in the mutation process. $CR$ is the crossover probability. Randreal [0, 1) generates a random real number between 0 and 1. $x^*$ represents the current best individual. $F$ is the mutation operator, which is generally taken between [0, 2]. $x_i(j)$ is the $j$-th variable of the $i$-th individual in the population, and $v_i$ is its offspring. From the pseudocode of the DE algorithm, it can be seen that DE has strong exploration ability but is not good at local exploitation.

**Algorithm 1** (The DE Algorithm)

```
1. Generate the initial population $x_i$ ($i$ = 1, 2, . . . , $N$)
2. Evaluate the fitness of each individual in $x_i$
3.   while ($t < T$)
4.      for $i$ = 1 to $N$ do
5.         Select uniform randomly $r_1 \neq r_2 \neq r_3 \neq i$
6.         $j_{\text{rand}}$ = randint(1, $n$)
7.         for $j$ = 1 to $d$ do
8.            if randreal$_j$ [0, 1) > $CR$ or $j == j_{rand}$ then
9.               $v_i(j) = x^*(j) + F \times (x_{r2}(j) - x_{r3}(j))$
10.           else
11.              $v_i(j) = x_i(j)$
12.           end if
13.        end for
14.     end for
15.     Evaluate the offspring $v_i$
16.     if $v_i$ is better than $X_i$ then
17.        Update individual $i$, $x_i = v_i$
18.        if $v_i$ is better than $x^*$ then
19.           Update best individual, $x^* = v_i$
20.        end if
21.     end if
22. end while
```

To improve the exploration ability of the ChOA algorithm, we hybridize the operators of ChOA with the mutation operator of DE to obtain a mixed operator. In addition, a nonlinear transformation convergence factor is introduced to control individuals choosing different operators and behaviors (exploration or exploitation) at different stages of iteration. The nonlinear convergence factor $\alpha$ is adjusted by Equation (25).

$$\alpha = 1 - (\frac{e^{t/T} - 1}{e - 1})^{\beta} \tag{25}$$

where $e$ represents the Euler constant. $T$ represents the maximum number of iterations; $\beta \in [1, 10]$ is used to control the decay rate of $\alpha$, and the larger $\beta$, the slower the convergence factor decays. In order to allow ChOA to explore the space sufficiently in the early stage of iteration and to perform well in local exploitation in the later stage, $\beta$ is set to 3.5 in this paper, which has been obtained as the optimal state after multiple experimental tests. However, it can be adjusted according to different optimization tasks. The mutation strategy of DE gives it excellent exploration performance [47]. After combining ChOA with DE, the development ability is similar to before, while the difference is that in the exploration phase, more elite individuals (good offspring replacing parents) are retained in the population. During the crossover and mutation processes, we still need to consider boundary constraints. If a dimension of an individual violates the constraint condition, it should be adjusted by Equation (26).

$$x_i(j) = \begin{cases} lb_j + randreal(0,1) \times (ub_j - lb_j) & if \quad x_i(j) < lb_j \\ ub_j - randreal(0,1) \times (ub_j - lb_j) & if \quad x_i(j) < ub_j \end{cases} \tag{26}$$

$lb_j$ and $ub_j$ represent the lower and upper bounds of the $j$-th dimension, respectively. $x_i(j)$ represents the $j$-th dimension of the $i$-th solution. $randreal(0, 1)$ represents a random number between 0 and 1.

### 4.2. Improved Reverse Learning

In order to improve the population diversity and individual quality of ChOA, we used improved reverse learning to obtain the reverse solutions of individuals and then retained the individuals with high fitness values. Traditional reverse learning increases the diversity of particles, but obtaining new positions symmetrically makes it easy for the algorithm to fall into the local extremum. Therefore, we combine the refraction principle of light with reverse learning. As shown in Figure 2, the refraction angle is introduced when obtaining the reverse position of the current individual, thus expanding the search scope of the individual and improving the generalization ability of the algorithm.

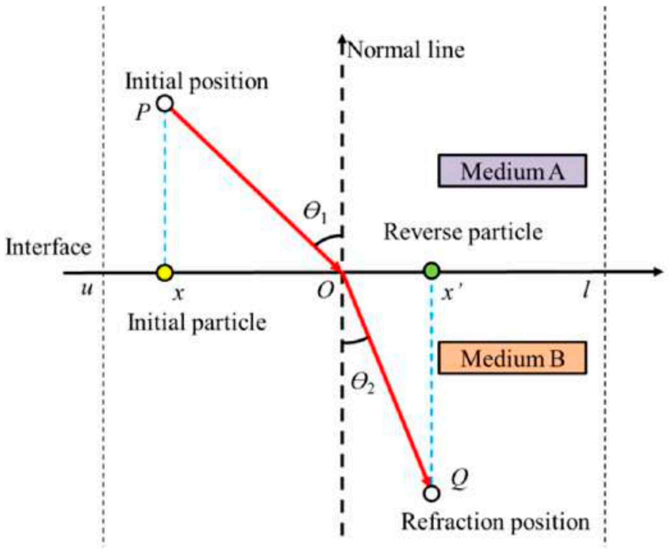

**Figure 2.** Schematic diagram of refraction and reverse learning.

In refraction reverse learning, the upper and lower sides of the horizontal axis are distributed with mediums A and B, which have different refractive indexes. The vertical axis is the normal line. $u$ and $l$ represent the upper and lower bounds of the search space, respectively, $x \in [u, l]$, and O is the midpoint of the interval $[u, l]$. Light enters the medium A along the $PO$ direction (Angle of incidence $\theta_1$), refracts at O, and exits along the $OQ$ direction (Angle of refraction $\theta_2$). Equations (27) and (28) can be obtained from the geometric relationship in Figure 2.

$$\begin{cases} \sin\theta_1 = ((u+l)/2 - x)/|PO| \\ \sin\theta_2 = (x\prime - (u+l)/2)/|OQ| \end{cases} \tag{27}$$

$$\eta = \frac{\sin\theta_1}{\sin\theta_2} \tag{28}$$

where $\eta$ represents the refractive index. Let $k = |PO|/|OQ|$, then the refraction reverse learning solution can be determined, as shown in Equation (29).

$$x\prime = \frac{u+l}{2} + \frac{u+l}{2k\eta} - \frac{x}{k\eta} \tag{29}$$

By extending Equation (29) to an $n$-dimensional space, the general form of the inverse solution can be obtained, as shown in Equation (30).

$$x\prime_i = \frac{u_i+l_i}{2} + \frac{u_i+l_i}{2k\eta} - \frac{x_i}{k\eta} \tag{30}$$

where $u_i$ and $l_i$ represent the $i$-th dimensional vector of the upper and lower bounds, respectively. At the later stage of the iteration, due to the concentrated distribution of particles near high-quality solutions, it may be difficult for the algorithm to find the global optimal solution. Therefore, hyper-parametric $\omega$ is introduced. It can adjust adaptively according to different iteration stages to increase the randomness of the solution, so as to improve the ability of the algorithm to escape from local optima. The improvement is shown in Equation (31).

$$\begin{cases} x\prime_i = \frac{u_i+l_i}{2} + \frac{u_i+l_i}{2\omega} - \frac{x_i}{\omega} \\ \omega = \frac{\sigma}{2} - \left(\frac{e^{t/T}-1}{e-1}\right)^{\sigma} \end{cases} \quad (31)$$

where $t$ represents the current number of iterations and $T$ represents the maximum number of iterations. $\sigma$ can control the attenuation rate of $\omega$; the larger the $\sigma$, the slower the $\omega$ decays. After obtaining the reverse position of each chimp, individuals with higher fitness values were retained through the greedy strategy, while individuals with lower fitness values were eliminated, as shown in Equation (32).

$$x_{update} = \max\_fitness(x_i, x\prime_i) \quad (32)$$

### 4.3. Similarity Preference Weight

In ChOA, the average positions of attacker, driver, barrier, and chaser determine the update trend of other chimps' positions. If these four types of chimps are located near the local optimal solution and other individuals converge towards them without distinction, it may result in the inability to search for a global optimal solution. Therefore, we propose to consider the similarity of each chimp's attacker, driver, barrier, and chaser separately. In an $n$-dimensional space, the similarity between two individuals is described by the sum of their absolute characteristic differences in each dimension. This idea fits with the Manhattan distance. The Manhattan definitions of $n$-dimensional vectors $a = (x_{11}, x_{12}, x_{13}, \ldots, x_{1n})$ and $b = (x_{21}, x_{22}, x_{23}, \ldots, x_{2n})$ are shown in Equation (33).

$$d(a,b) = \sum_{k}^{n} |x_{1k} - x_{2k}| \quad (33)$$

In Equation (33), the similarity between two chimps is inversely proportional to the value of $d$. In the process of location updating, we make chimps more inclined to approach leaders with lower similarity and stay away from leaders with higher similarity. This mechanism can effectively alleviate the serious assimilation phenomenon at the end of iteration and avoid the algorithm falling into local optima. Equation (34) shows an improved location update method based on Equation (23).

$$\begin{cases} x(t+1) = \lambda_1 V_1 + \lambda_2 V_2 + \lambda_3 V_3 + \lambda_4 V_4 \\ \lambda_i = \frac{d_i}{d_1+d_2+d_3+d_4}, i = 1,2,3,4 \end{cases} \quad (34)$$

where, $\lambda_1$, $\lambda_2$, $\lambda_3$, and $\lambda_4$ represent the current chimp's preference weights for attacker, driver, barrier, and chaser, respectively. $d$ represents the similarity value calculated from Equation (33).

### 4.4. TRS-ChOA Pseudocode

Algorithm 2 is the pseudocode for the improved ChOA algorithm, which combines three different strategies. The improved ChOA consists of two parts: the exploration part (lines 16–21 of Algorithm2) and the exploitation part (lines 22–27 of Algorithm 2).

**Algorithm 2** (TRS-ChOA Algorithm)

```
1.  Generate the initial population $x_i$ ($i = 1, 2, \ldots, N$)
2.  Initialize $f$, $m$, $a$ and $c$
3.  Divide chimps randomly into independent groups
4.  Calculate the fitness of each chimp
5.  $x_{Attacker}$ = the best search agent
6.  $x_{Chaser}$ = the second-best search agent
7.  $x_{Barrier}$ = the third-best search agent
8.  $x_{Driver}$ = the fourth-best search agent
9.   while ($t < T$)
10.      for $i = 1$ to $N$ do
11.         Extract the chimp's group
12.         Use its group strategy to update $f$, $m$, $c$, $a$ and $d$
13.         Select uniform randomly $r_1 \neq r_2 \neq i$
14.         Update $\alpha$ by the Equation (25), $j_{rand} = randint(1,n)$, $p$ = randreal(0,1), $\mu$ = randreal(0,1)
15.         for $j = 1$ to $d$ do
16.            if $p \leq \alpha$ then
17.               if randreal$_j$[0,1) $\leq$ CR or $j == j_{\text{rand}}$ then
18.                  $v_i(j)$ = randchoice{$x_{Attacker}(j)$, $x_{Chaser}(j)$, $x_{Barrier}(j)$, $x_{Driver}(j)$} + $F \times (x_{r1}(j) - x_{r2}(j))$
19.               else
20.                  $v_i(j) = Chaotic\_value$
21.               end if
22.            else if $p > \alpha$ then
23.               if randreal$_j$ [0,1) $\leq$ 0.5 then
24.                  $v_i(j)$= $x_{Attacker}(j) - a \times d$
25.               else
26.                  Update the position of the current search agent using the Equation (34)
27.               end if
28.            end if
29.         end for
30.      end for
31.   Calculate the reverse position of each chimp by Equation (31)
32.   Update high-quality individuals by Equation (32)
33.   Ranking chimp individuals by fitness value
34.   Update $x_{Attacker}$, $x_{Driver}$, $x_{Barrier}$, $x_{Chaser}$
35.   $t = t + 1$
36.  end while
37. return $x_{Attacker}$
```

### 4.5. Time Complexity Analysis of TRS-ChOA

The time complexity of the basic ChOA algorithm is $O$ ($N \times d \times T$). Where, $N$ is the population size, $d$ defines the dimension of a test function, and $T$ is the maximum number of iterations. The time complexity of TRS-ChOA is analyzed as follows:

(1) The time complexity after combining with differential evolution is represented as $O$ ($N \times d$), so the time complexity of the algorithm becomes $O$ ($N \times d \times T + N \times d$) = $O$ ($N \times d \times T$) after it is introduced;

(2) The time complexity of using improved reverse learning to update the position of the population is $O$ ($N \times d \times T$), However, this is a juxtaposed loop, so the time complexity of the algorithm is $O$ ($N \times d \times T + N \times d \times$ T) = $O$ ($N \times d \times T$).

(3) Assuming that the time required to introduce the similarity preference weight is $t$, then the time complexity of the algorithm is $O$ ($N \times d \times T + t$) = $O$ ($N \times d \times T$)

In summary, the time complexity of TRS-ChOA is $O$ ($N \times d \times T$). It can be seen that the time complexity of the TRS-ChOA proposed in this paper is consistent with the time complexity of ChOA.

## 5. TRS-ChOA Optimized Performance Test

To verify the optimization performance of TRS-ChOA, we selected part of the benchmark test function and the CEC2017 complex test function [48] to test it. The environment of the experiment was Windows 10, with a CPU of 3.4 GHz Intel Core i7-6500 U, 8 GB of RAM, and a 64-bit operating system. The simulation software is Matlab R2021. Table 1 records the internal parameters of the algorithms involved in the experiment and their references.

**Table 1.** Algorithm parameter setting.

| Algorithm | Parameters Setting | Reference |
|---|---|---|
| GWO | $r_1 \in [0, 1]$, $r_2 \in [0, 1]$ | [17] |
| SSA | proportion of discoverers: 20%<br>proportion of scouter: 10%<br>alert threshold: 0.7 | [19] |
| WOA | $b = 1$, $r_1 \in [0, 1]$, $r_2 \in [0, 1]$,<br>$l \in [-1, 1]$, $p \in [0, 1]$ | [20] |
| ALO | $w = 1$, $t \leq 0.1$ T<br>$w = 2$, $t > 0.1$ T<br>$w = 3$, $t > 0.5$ T<br>$w = 4$, $t > 0.75$ T<br>$w = 5$, $t > 0.9$ T<br>$w = 6$, $t > 0.95$ T | [21] |
| ChOA | $r_1 \in [0, 1]$, $r_2 \in [0, 1]$, $m$ = chaos (3,1,1) | [35] |
| TRS-ChOA | $F \in [0, 1]$, CR = 0.1, k $\in [0, 1]$, $\sigma = 2.5$ | Section 4 of this article |

### *5.1. Benchmark Function Test*

We selected 13 benchmark functions [49] as the content of the test, and their details are recorded in Table 2. Among the 13 functions, $f_1$–$f_6$ are single module functions, which are mainly used to study the convergence speed and accuracy of the algorithm. $f_7$–$f_{13}$ are multi-modal functions used to evaluate the algorithm's exploration ability and its ability to avoid local optima. In the experiment, the dimensions of the test functions are $d = 30/500/1000$ to verify the ability of the algorithm to handle low- and high-dimensional problems. To verify the effectiveness of the three improvement strategies proposed in Section 4, the ChOA that integrates the differential evolution algorithm, improves reverse learning, and similarity preference weights are recorded as TChOA, RChOA, and SchOA, respectively, and they are tested together with ChOA and TRS-ChOA. This is an ablation study. To ensure the fairness of the experiment, basic parameters were set uniformly during the experiment: population size $N = 30$, maximum number of iterations $T = 500$. After each algorithm runs 50 times independently, the mean value and standard deviation of the results are recorded. The specific experimental results are shown in Table 3.

**Table 2.** Introduction to benchmark functions.

| Fun No. | Name | Range | Dim | Optimal Value | Function Type |
|---|---|---|---|---|---|
| $f_1$ | Sphere Function | $[-100, 100]$ | 30, 500, 1000 | 0 | Single-modal |
| $f_2$ | Schwefel's problem 2.22 | $[-10, 10]$ | 30, 500, 1000 | 0 | Single-modal |
| $f_3$ | Schwefel's problem 1.2 | $[-100, 100]$ | 30, 500, 1000 | 0 | Single-modal |
| $f_4$ | Schwefel's problem 2.21 | $[-100, 100]$ | 30, 500, 1000 | 0 | Single-modal |
| $f_5$ | Generalized Rosenbrock's Function | $[-30, 30]$ | 30, 500, 1000 | 0 | Single-modal |
| $f_6$ | Step Function | $[-100, 100]$ | 30, 500, 1000 | 0 | Single-modal |
| $f_7$ | Quartic Function | $[-1.28, 1.28]$ | 30, 500, 1000 | 0 | Single-modal |
| $f_8$ | Generalized Schwefel's problem2.26 | $[-500, 500]$ | 30, 500, 1000 | 12,569.5 | Multi-modal |
| $f_9$ | Generalized Rastrigin's Function. | $[-5.12, 5.12]$ | 30, 500, 1000 | 0 | Multi-modal |
| $f_{10}$ | Ackley'sFunction | $[-32, 32]$ | 30, 500, 1000 | 0 | Multi-modal |
| $f_{11}$ | Generalized Criewank's Function | $[-600, 600]$ | 30, 500, 1000 | 0 | Multi-modal |
| $f_{12}$ | Generalized Penalized Function 1 | $[-50, 50]$ | 30, 500, 1000 | 0 | Fixed multi-modal |
| $f_{13}$ | Generalized Penalized Function 2 | $[-50, 50]$ | 30, 500, 1000 | 0 | Fixed multi-modal |

**Table 3.** Comparison of optimization results of different improvement strategies for ChOA.

| Fun No. | Dim | ChOA | | TChOA | | RChOA | | SChOA | | TRS-ChOA | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| $f_1$ | $d = 30$ | $1.69 \times 10^{-21}$ | $4.76 \times 10^{-21}$ | $1.70 \times 10^{-215}$ | $4.95 \times 10^{-214}$ | $1.46 \times 10^{-243}$ | $1.53 \times 10^{-240}$ | $1.76 \times 10^{-300}$ | $8.66 \times 10^{-298}$ | **0** | **0** |
| | $d = 500$ | $2.03 \times 10^{-22}$ | $3.61 \times 10^{-22}$ | $5.69 \times 10^{-168}$ | $6.42 \times 10^{-174}$ | $3.58 \times 10^{-233}$ | $1.76 \times 10^{-230}$ | $2.20 \times 10^{-301}$ | $6.33 \times 10^{-300}$ | **0** | **0** |
| | $d = 1000$ | $7.84 \times 10^{-16}$ | $4.38 \times 10^{-17}$ | $3.77 \times 10^{-164}$ | $4.29 \times 10^{-150}$ | $1.25 \times 10^{-200}$ | $9.23 \times 10^{-203}$ | $6.53 \times 10^{-276}$ | $4.85 \times 10^{-280}$ | **0** | **0** |
| $f_2$ | $d = 30$ | $2.13 \times 10^{-15}$ | $1.41 \times 10^{-15}$ | $1.13 \times 10^{-78}$ | $5.74 \times 10^{-77}$ | $3.38 \times 10^{-112}$ | $3.14 \times 10^{-103}$ | $4.49 \times 10^{-326}$ | $2.47 \times 10^{-341}$ | **0** | **0** |
| | $d = 500$ | $3.74 \times 10^{-15}$ | $1.89 \times 10^{-14}$ | $4.81 \times 10^{-82}$ | $5.11 \times 10^{-85}$ | $6.52 \times 10^{-105}$ | $4.32 \times 10^{-105}$ | $7.55 \times 10^{-317}$ | $4.35 \times 10^{-377}$ | **0** | **0** |
| | $d = 1000$ | $6.51 \times 10^{-15}$ | $7.03 \times 10^{--9}$ | $7.64 \times 10^{-77}$ | $3.58 \times 10^{-77}$ | $2.78 \times 10^{-67}$ | $7.53 \times 10^{-69}$ | $4.89 \times 10^{-300}$ | $3.14 \times 10^{-305}$ | **0** | **0** |
| $f_3$ | $d = 30$ | $5.15 \times 10^{1}$ | $2.90 \times 10^{1}$ | $3.50 \times 10^{-167}$ | $1.22 \times 10^{-165}$ | $4.33 \times 10^{-214}$ | $8.64 \times 10^{-213}$ | **0** | **0** | **0** | **0** |
| | $d = 500$ | $2.82 \times 10^{1}$ | $7.26 \times 10^{1}$ | $2.21 \times 10^{-154}$ | $1.97 \times 10^{-148}$ | $4.67 \times 10^{-212}$ | $6.38 \times 10^{-213}$ | $6.47 \times 10^{-303}$ | $4.54 \times 10^{-302}$ | **0** | **0** |
| | $d = 1000$ | $4.15 \times 10^{1}$ | $3.91 \times 10^{1}$ | $5.30 \times 10^{-144}$ | $1.79 \times 10^{-139}$ | $3.55 \times 10^{-210}$ | $7.24 \times 10^{-218}$ | $6.52 \times 10^{-170}$ | $6.09 \times 10^{-175}$ | **0** | **0** |
| $f_4$ | $d = 30$ | $4.92 \times 10^{-1}$ | $2.34 \times 10^{-1}$ | $1.36 \times 10^{-81}$ | $1.93 \times 10^{-80}$ | $3.72 \times 10^{-141}$ | $3.69 \times 10^{-151}$ | $3.36 \times 10^{-184}$ | $2.36 \times 10^{-185}$ | **0** | **0** |
| | $d = 500$ | $4.57 \times 10^{-1}$ | $1.99 \times 10^{-1}$ | $2.14 \times 10^{-56}$ | $4.31 \times 10^{-67}$ | $3.11 \times 10^{-110}$ | $2.76 \times 10^{-133}$ | $3.52 \times 10^{-217}$ | $8.12 \times 10^{-142}$ | **0** | **0** |
| | $d = 1000$ | $8.44 \times 10^{0}$ | $7.34 \times 10^{0}$ | $3.42 \times 10^{-40}$ | $2.66 \times 10^{-42}$ | $1.52 \times 10^{-86}$ | $3.74 \times 10^{-76}$ | $4.16 \times 10^{-182}$ | $3.81 \times 10^{-163}$ | **0** | **0** |
| $f_5$ | $d = 30$ | $2.90 \times 10^{1}$ | $4.25 \times 10^{1}$ | $1.33 \times 10^{1}$ | $1.04 \times 10^{1}$ | $2.82 \times 10^{-2}$ | $1.76 \times 10^{-1}$ | $2.88 \times 10^{1}$ | $1.29 \times 10^{1}$ | **$2.64 \times 10^{-4}$** | **$8.62 \times 10^{-4}$** |
| | $d = 500$ | $4.36 \times 10^{1}$ | $1.69 \times 10^{1}$ | $4.22 \times 10^{2}$ | $1.67 \times 10^{1}$ | $4.66 \times 10^{-1}$ | $1.80 \times 10^{-1}$ | $2.69 \times 10^{1}$ | $3.58 \times 10^{0}$ | **$2.15 \times 10^{-5}$** | **$8.33 \times 10^{-5}$** |
| | $d = 1000$ | $8.92 \times 10^{2}$ | $4.53 \times 10^{2}$ | $1.73 \times 10^{2}$ | $6.51 \times 10^{1}$ | $2.06 \times 10^{1}$ | $4.69 \times 10^{0}$ | $1.18 \times 10^{2}$ | $2.39 \times 10^{1}$ | **$3.05 \times 10^{-5}$** | **$6.69 \times 10^{-4}$** |
| $f_6$ | $d = 30$ | $3.53 \times 10^{1}$ | $3.05 \times 10^{0}$ | $8.24 \times 10^{-4}$ | $5.56 \times 10^{-4}$ | **$3.62 \times 10^{-5}$** | **$1.51 \times 10^{-5}$** | $1.33 \times 10^{0}$ | $7.39 \times 10^{-1}$ | $1.37 \times 10^{-4}$ | $6.12 \times 10^{-4}$ |
| | $d = 500$ | $4.31 \times 10^{0}$ | $4.81 \times 10^{0}$ | $7.96 \times 10^{-2}$ | $4.81 \times 10^{-2}$ | **$3.69 \times 10^{-5}$** | **$1.71 \times 10^{-4}$** | $1.57 \times 10^{0}$ | $6.63 \times 10^{-1}$ | $1.47 \times 10^{-2}$ | $5.77 \times 10^{-3}$ |
| | $d = 1000$ | $6.37 \times 10^{1}$ | $4.22 \times 10^{1}$ | $1.67 \times 10^{-3}$ | $4.10 \times 10^{-2}$ | **$9.04 \times 10^{-4}$** | **$2.58 \times 10^{-2}$** | $7.33 \times 10^{1}$ | $1.09 \times 10^{0}$ | $1.55 \times 10^{-2}$ | $7.04 \times 10^{-2}$ |
| $f_7$ | $d = 30$ | $1.82 \times 10^{-3}$ | $6.88 \times 10^{-4}$ | $2.55 \times 10^{-2}$ | $3.41 \times 10^{-2}$ | $2.13 \times 10^{-3}$ | $1.46 \times 10^{-3}$ | $5.38 \times 10^{-4}$ | $1.91 \times 10^{-4}$ | **$4.77 \times 10^{-7}$** | **$9.28 \times 10^{-8}$** |
| | $d = 500$ | $2.03 \times 10^{-3}$ | $5.16 \times 10^{-4}$ | $2.18 \times 10^{-2}$ | $3.63 \times 10^{-2}$ | $4.30 \times 10^{-4}$ | $3.87 \times 10^{-4}$ | $4.01 \times 10^{-3}$ | $3.19 \times 10^{-3}$ | **$4.35 \times 10^{-6}$** | **$7.81 \times 10^{-8}$** |
| | $d = 1000$ | $1.66 \times 10^{-2}$ | $6.09 \times 10^{-2}$ | $8.25 \times 10^{-1}$ | $4.52 \times 10^{-1}$ | $1.94 \times 10^{-3}$ | $8.33 \times 10^{-2}$ | $6.13 \times 10^{-3}$ | $5.46 \times 10^{-1}$ | **$5.72 \times 10^{-6}$** | **$3.20 \times 10^{-5}$** |
| $f_8$ | $d = 30$ | $-5734.36$ | $8.95 \times 10^{-9}$ | $-5498.63$ | $3.21 \times 10^{2}$ | $-10340.06$ | $2.21 \times 10^{3}$ | $-8334.31$ | $6.26 \times 10^{2}$ | **$-12{,}567.28$** | **$2.53 \times 10^{-10}$** |
| | $d = 500$ | $-5529.71$ | $6.34 \times 10^{-8}$ | $-5736.44$ | $2.73 \times 10^{2}$ | $-10649.63$ | $2.08 \times 10^{2}$ | $-8221.50$ | $5.90 \times 10^{2}$ | **$-12{,}496.63$** | **$4.10 \times 10^{-10}$** |
| | $d = 1000$ | $-6017.87$ | $3.65 \times 10^{-5}$ | $-4396.07$ | $8.60 \times 10^{3}$ | $-8774.59$ | $6.77 \times 10^{2}$ | $-8005.19$ | $2.53 \times 10^{3}$ | **$-12{,}195.10$** | **$2.74 \times 10^{-9}$** |
| $f_9$ | $d = 30$ | $1.37 \times 10^{1}$ | $6.11 \times 10^{2}$ | $8.07 \times 10^{1}$ | $2.88 \times 10^{0}$ | **0** | **0** | **0** | **0** | **0** | **0** |
| | $d = 500$ | $1.63 \times 10^{1}$ | $6.40 \times 10^{2}$ | $7.46 \times 10^{1}$ | $3.00 \times 10^{0}$ | **0** | **0** | **0** | **0** | **0** | **0** |
| | $d = 1000$ | $2.70 \times 10^{2}$ | $8.39 \times 10^{0}$ | $3.98 \times 10^{2}$ | $5.14 \times 10^{0}$ | $7.57 \times 10^{-279}$ | $9.41 \times 10^{-278}$ | $7.03 \times 10^{-131}$ | $3.72 \times 10^{-110}$ | **0** | **0** |
| $f_{10}$ | $d = 30$ | $2.00 \times 10^{1}$ | $9.03 \times 10^{-14}$ | $5.89 \times 10^{0}$ | $3.86 \times 10^{-1}$ | **$4.34 \times 10^{-15}$** | $3.97 \times 10^{-11}$ | $1.60 \times 10^{-12}$ | $7.11 \times 10^{-16}$ | $8.88 \times 10^{-14}$ | **0** |
| | $d = 500$ | $3.12 \times 10^{1}$ | $6.51 \times 10^{-13}$ | $5.53 \times 10^{0}$ | $4.02 \times 10^{-1}$ | $4.22 \times 10^{-12}$ | $2.67 \times 10^{-8}$ | $3.07 \times 10^{-11}$ | $2.97 \times 10^{-15}$ | **$6.93 \times 10^{-14}$** | **0** |
| | $d = 1000$ | $4.73 \times 10^{1}$ | $7.73 \times 10^{-4}$ | $8.14 \times 10^{1}$ | $7.50 \times 10^{-2}$ | $5.96 \times 10^{-13}$ | $7.51 \times 10^{-8}$ | $8.46 \times 10^{-7}$ | $5.29 \times 10^{-10}$ | **$5.00 \times 10^{-13}$** | **$1.37 \times 10^{-12}$** |
| $f_{11}$ | $d = 30$ | $4.42 \times 10^{-2}$ | $3.97 \times 10^{-12}$ | $4.45 \times 10^{-1}$ | $3.24 \times 10^{-2}$ | $1.61 \times 10^{-216}$ | $1.62 \times 10^{-223}$ | **0** | **0** | **0** | **0** |
| | $d = 500$ | $4.90 \times 10^{-3}$ | $3.62 \times 10^{-11}$ | $4.73 \times 10^{-2}$ | $1.88 \times 10^{-2}$ | **0** | **0** | **0** | **0** | **0** | **0** |
| | $d = 1000$ | $2.26 \times 10^{-3}$ | $4.74 \times 10^{-11}$ | $5.30 \times 10^{-2}$ | $4.09 \times 10^{-2}$ | $2.73 \times 10^{-10}$ | $2.11 \times 10^{-10}$ | **$6.16 \times 10^{-14}$** | **$8.40 \times 10^{-15}$** | $3.71 \times 10^{-13}$ | $6.11 \times 10^{-16}$ |
| $f_{12}$ | $d = 30$ | $4.68 \times 10^{-1}$ | $1.60 \times 10^{-11}$ | $1.41 \times 10^{1}$ | $3.17 \times 10^{0}$ | $9.29 \times 10^{-2}$ | $7.76 \times 10^{-2}$ | $9.02 \times 10^{-3}$ | $5.12 \times 10^{-2}$ | **$6.07 \times 10^{-6}$** | **$8.36 \times 10^{-13}$** |
| | $d = 500$ | $3.90 \times 10^{-1}$ | $4.25 \times 10^{-10}$ | $2.68 \times 10^{1}$ | $5.33 \times 10^{0}$ | $6.75 \times 10^{-2}$ | $5.77 \times 10^{-1}$ | $4.57 \times 10^{-2}$ | $8.16 \times 10^{-2}$ | **$9.12 \times 10^{-6}$** | **$6.31 \times 10^{-12}$** |
| | $d = 1000$ | $9.57 \times 10^{0}$ | $7.75 \times 10^{-8}$ | $4.69 \times 10^{2}$ | $8.01 \times 10^{1}$ | $5.10 \times 10^{-2}$ | $9.69 \times 10^{-2}$ | $3.88 \times 10^{-3}$ | $1.90 \times 10^{-2}$ | **$5.17 \times 10^{-6}$** | **$6.21 \times 10^{-12}$** |
| $f_{13}$ | $d = 30$ | $2.71 \times 10^{0}$ | $1.66 \times 10^{-13}$ | $2.25 \times 10^{1}$ | $3.37 \times 10^{1}$ | $5.48 \times 10^{-1}$ | $2.82 \times 10^{-6}$ | $2.85 \times 10^{-1}$ | $7.45 \times 10^{-9}$ | **$1.93 \times 10^{-4}$** | **$5.65 \times 10^{-17}$** |
| | $d = 500$ | $6.24 \times 10^{0}$ | $3.57 \times 10^{-12}$ | $3.06 \times 10^{1}$ | $5.65 \times 10^{1}$ | $3.98 \times 10^{-1}$ | $2.17 \times 10^{-5}$ | $1.74 \times 10^{0}$ | $8.61 \times 10^{-5}$ | **$2.01 \times 10^{-4}$** | **$4.85 \times 10^{-15}$** |
| | $d = 1000$ | $4.82 \times 10^{0}$ | $3.07 \times 10^{-10}$ | $6.44 \times 10^{1}$ | $6.00 \times 10^{1}$ | $4.32 \times 10^{-1}$ | $2.73 \times 10^{-1}$ | $9.38 \times 10^{0}$ | $1.00 \times 10^{-3}$ | **$2.13 \times 10^{-4}$** | **$9.32 \times 10^{-14}$** |

As can be seen from Table 3, the averages and standard deviations of TRS-ChOA are better than the other four algorithms in most cases, indicating that the three improved strategies effectively enhance the optimization accuracy and stability of the original ChOA. When dimensions are between 30 and 500, TRS-ChOA can search for optimal values on functions $f_1, f_2, f_3, f_4, f_9$, and $f_{11}$, while ChOA does not obtain the global optima on any of the 13 functions. On the multi-modal function $f_9$, RChOA and SChOA can find theoretical optima at 30 and 500 dimensions, and have higher accuracy than ChOA at 1000 dimensions, indicating that reverse learning strategies and similarity preference weights can help the algorithm escape local optima. TChOA performs better than ChOA on functions $f_1, f_2, f_3, f_4$, $f_5, f_6$, and $f_{10}$. Most of these functions are single-modal functions, which indicates that The differential evolution can improve the optimization ability of the algorithm on single-mode problems but is not effective in dealing with multi-mode problems. Although TRS-ChOA cannot find the theoretical optimal value for all functions, its optimization results can be close to the theoretical extreme value. For example, on function $f_8$, TRS-ChOA can converge to $-12{,}567.28$, which is the closest theoretical value among all algorithms. It can be concluded that TRS-ChOA exhibits stronger search performance than ChOA in both low-dimensional and high-dimensional functions and is an optimization algorithm with better stability and robustness.

The results in Table 3 reflect the advantages of TRS-ChOA in terms of optimization accuracy and stability. To show the dynamic convergence characteristics of TRS-ChOA more directly, we give the average convergence curves of TChOA, RChOA, SChOA, ChOA, and TRS-ChOA on the benchmark functions $f_1$–$f_{13}$. The dimensions of the functions are $d$ = 500, the population size is $N$ = 30, and the maximum iteration number is $T$ = 500.

The specific results are shown in Figure 3. From the convergence curves of functions $f_1$, $f_2$, $f_3$, $f_4$, $f_9$, $f_{10}$, and $f_{11}$, it can be seen that compared with the other four algorithms, TRS-ChOA has a faster convergence speed, while ChOA has the slowest optimization speed, indicating that the three improvement strategies enhance the performance of ChOA to varying degrees. From the convergence curves of functions $f_5$, $f_7$, $f_8$, and $f_{12}$, it can be found that TChOA, RChOA, SChOA, and TRS-ChOA have higher optimization accuracy than ChOA while guaranteeing the optimization speed, which indicates that the three improvement strategies increase the population diversity of ChOA and improve the ability of the algorithm to escape from local optima. According to the above analysis, TRS-ChOA has a faster convergence rate in the optimization process of benchmark functions and can escape in time to improve the accuracy of optimization when falling into local extremes. In short, it is a better algorithm for solving global optimization problems.

### 5.2. Wilcoxon Rank-Sum Test

When the data sample is not normally distributed, the Wilcoxon rank-sum test is often used to compare the difference between the distribution positions of two independent samples [50]. To comprehensively analyze the performance difference between the TRS-ChOA and other algorithms, we use the running results of ChOA, TChOA, RChOA, SChOA, and TRS-ChOA in thirteen benchmark test functions to conduct the Wilcoxon rank sum test. In the Wilcoxon rank-sum test, it is stipulated that when $p < 5\%$, the test results of the two algorithms are significantly different. When $p > 5\%$, there is no significant difference between the test results of the two algorithms. The results of the Wilcoxon rank-sum test are shown in Table 4. Where, "NaN" indicated that there were no data for comparison. +, =, and—indicate that the performance of TRS-ChOA is superior, approximate, and inferior to that of the compared algorithms, respectively.

**Table 4.** Wilcoxon rank-sum test results.

| Fun No. | ChOA ($P_1$) | TChOA ($P_2$) | RChOA ($P_3$) | SChOA ($P_4$) |
|---|---|---|---|---|
| $f_1$ | $8.01 \times 10^{-14}$ | $8.01 \times 10^{-14}$ | $8.01 \times 10^{-14}$ | $8.01 \times 10^{-14}$ |
| $f_2$ | $1.83 \times 10^{-15}$ | $1.83 \times 10^{-15}$ | $1.83 \times 10^{-15}$ | $1.83 \times 10^{-15}$ |
| $f_2$ | $3.16 \times 10^{-13}$ | $3.16 \times 10^{-13}$ | $3.16 \times 10^{-13}$ | $3.16 \times 10^{-13}$ |
| $f_2$ | $3.02 \times 10^{-16}$ | $3.16 \times 10^{-13}$ | $3.16 \times 10^{-13}$ | $3.16 \times 10^{-13}$ |
| $f_3$ | $2.03 \times 10^{-7}$ | $4.73 \times 10^{-12}$ | $5.80 \times 10^{-15}$ | $1.01 \times 10^{-17}$ |
| $f_4$ | $1.86 \times 10^{-12}$ | $8.66 \times 10^{-14}$ | $2.64 \times 10^{-15}$ | $1.83 \times 10^{-17}$ |
| $f_5$ | $3.09 \times 10^{-9}$ | $1.71 \times 10^{-10}$ | $1.83 \times 10^{-11}$ | $1.83 \times 10^{-11}$ |
| $f_6$ | $7.77 \times 10^{-13}$ | NaN | $1.12 \times 10^{-13}$ | $1.83 \times 10^{-13}$ |
| $f_7$ | $1.64 \times 10^{-14}$ | $9.53 \times 10^{-17}$ | $7.07 \times 10^{-18}$ | $1.01 \times 10^{-17}$ |
| $f_8$ | $2.65 \times 10^{-18}$ | $7.08 \times 10^{-12}$ | $0.91 \times 10^{-6}$ | $9.54 \times 10^{-18}$ |
| $f_9$ | $3.31 \times 10^{-20}$ | $3.31 \times 10^{-20}$ | NaN | $8.97 \times 10^{-7}$ |
| $f_{10}$ | $3.31 \times 10^{-20}$ | $3.43 \times 10^{-13}$ | $6.45 \times 10^{-15}$ | NaN |
| $f_{11}$ | $3.31 \times 10^{-20}$ | $3.31 \times 10^{-20}$ | $3.27 \times 10^{-6}$ | NaN |
| $f_{12}$ | $7.10 \times 10^{-9}$ | $1.46 \times 10^{-12}$ | $7.06 \times 10^{-18}$ | $7.79 \times 10^{-12}$ |
| $f_{13}$ | $1.58 \times 10^{-12}$ | $1.62 \times 10^{-13}$ | $4.44 \times 10^{-15}$ | $2.38 \times 10^{-10}$ |
| +/=/— | 13/0/0 | 12/1/0 | 11/1/1 | 11/2/0 |

As can be seen from Table 4, in function $f_6$, the $p$ value of RChOA is $1.12 \times 10^{-13}$, indicating that there is a significant difference between RChOA and TRS-ChOA, and RChOA demonstrates stronger optimization performance on $f_6$ than TRS-ChOA. NaN indicates that the difference is not significant, i.e., the optimization performance of the two algorithms is equivalent. In most cases, the value of $p$ is less than 5%, indicating that the optimization performance of TRS-ChOA on benchmark functions is significantly superior to the other four algorithms.

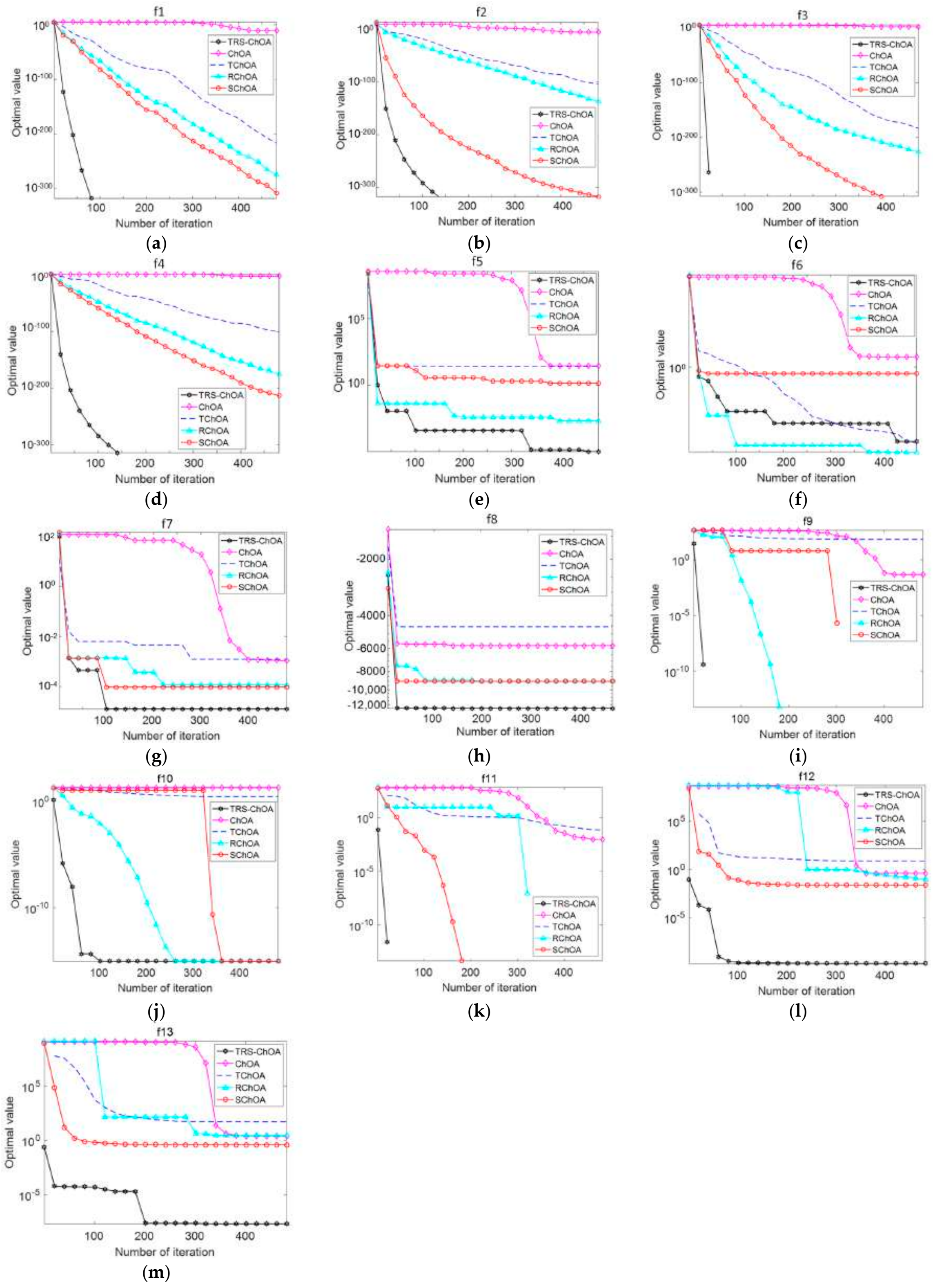

**Figure 3.** Convergence curve of TRS − ChOA and other algorithms on the benchmark test function. (**a**) Convergence curve of $f_1$. (**b**) Convergence curve of $f_2$. (**c**) Convergence curve of $f_3$. (**d**) Convergence curve of $f_4$. (**e**) Convergence curve of $f_5$. (**f**) Convergence curve of $f_6$. (**g**) Convergence curve of $f_7$. (**h**) Convergence curve of $f_8$. (**i**) Convergence curve of $f_9$. (**j**) Convergence curve of $f_{10}$. (**k**) Convergence curve of $f_{11}$. (**l**) Convergence curve of $f_{12}$. (**m**) Convergence curve of $f_{13}$.

### *5.3. CEC2017 Function Test*

The ablation experiments in Section 5.1 confirm that TRS-ChOA, which integrates the three improved strategies, has better performance than ChOA, but this result does not reflect the difference between TRS-ChOA and other classical swarm intelligence optimization algorithms. To further verify the performance of TRS-ChOA in dealing with complex functions, we selected TRS-ChOA, ChOA, SSA, WOA, and GWO for comparative testing on CEC2017 complex functions. Table 5 shows the specific information about the CEC2017 functions. These functions include complex functions composed of basic test functions and incorporate new problems such as rotated traps, posing greater challenges to the optimization ability of the algorithm. In Table 5, UF, SMF, HF, and CF indicate that the types of functions are Uni-modal Functions, Simple Multimodal Functions, Hybrid Functions, and Composition Functions, respectively. The basic parameters in the experiment are set as follows: population size $N$ = 100, dimension $d$ = 10/50, and maximum iteration number $T$ = 1000. After running each algorithm 50 times independently, calculate the mean value and standard deviation. The specific experimental results are shown in Table 6.

**Table 5.** CEC2017 test function information.

| **Fun No.** | **Dim** | **Function Type** | **Range** | **Optimal Value** |
|---|---|---|---|---|
| CEC01 | 10, 50 | UF (Uni-modal Function) | [−100, 100] | 100 |
| CEC02 | 10, 50 | UF | [−100, 100] | 200 |
| CEC03 | 10, 50 | SMF (Simple Multimodal Functions) | [−100, 100] | 300 |
| CEC04 | 10, 50 | SMF | [−100, 100] | 400 |
| CEC05 | 10, 50 | SMF | [−100, 100] | 500 |
| CEC06 | 10, 50 | SMF | [−100, 100] | 600 |
| CEC07 | 10, 50 | SMF | [−100, 100] | 700 |
| CEC08 | 10, 50 | SMF | [−100, 100] | 800 |
| CEC09 | 10, 50 | SMF | [−100, 100] | 900 |
| CEC10 | 10, 50 | HF (Hybrid Function) | [−100, 100] | 1000 |
| CEC11 | 10, 50 | HF | [−100, 100] | 1100 |
| CEC12 | 10, 50 | HF | [−100, 100] | 1200 |
| CEC13 | 10, 50 | HF | [−100, 100] | 1300 |
| CEC14 | 10, 50 | HF | [−100, 100] | 1400 |
| CEC15 | 10, 50 | HF | [−100, 100] | 1500 |
| CEC16 | 10, 50 | HF | [−100, 100] | 1600 |
| CEC17 | 10, 50 | HF | [−100, 100] | 1700 |
| CEC18 | 10, 50 | HF | [−100, 100] | 1800 |
| CEC19 | 10, 50 | HF | [−100, 100] | 1900 |
| CEC20 | 10, 50 | CF (Composition Function) | [−100, 100] | 2000 |
| CEC21 | 10, 50 | CF | [−100, 100] | 2100 |
| CEC22 | 10, 50 | CF | [−100, 100] | 2200 |
| CEC23 | 10, 50 | CF | [−100, 100] | 2300 |
| CEC24 | 10, 50 | CF | [−100, 100] | 2400 |
| CEC25 | 10, 50 | CF | [−100, 100] | 2500 |
| CEC26 | 10, 50 | CF | [−100, 100] | 2600 |
| CEC27 | 10, 50 | CF | [−100, 100] | 2700 |
| CEC28 | 10, 50 | CF | [−100, 100] | 2800 |
| CEC29 | 10, 50 | CF | [−100, 100] | 2900 |

**Table 6.** CEC2017 function optimization comparison.

| Fun No. | Dim | ChOA | | SSA | | GWO | | WOA | | TRS-ChOA | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Mean | Std | Mean | Std | Mean | Std | Mean | Std | Mean | Std |
| CEC1 | d = 10 | $2.32 \times 10^{2}$ | $7.39 \times 10^{1}$ | $8.47 \times 10^{2}$ | $2.15 \times 10^{2}$ | $1.36 \times 10^{2}$ | $3.13 \times 10^{1}$ | $5.78 \times 10^{2}$ | $2.62 \times 10^{0}$ | **$1.14 \times 10^{2}$** | **$1.59 \times 10^{0}$** |
| | d = 50 | $4.52 \times 10^{2}$ | $2.31 \times 10^{2}$ | $3.62 \times 10^{3}$ | $4.22 \times 10^{2}$ | $2.37 \times 10^{2}$ | $1.16 \times 10^{2}$ | $7.40 \times 10^{2}$ | $1.72 \times 10^{2}$ | **$1.32 \times 10^{2}$** | **$6.21 \times 10^{1}$** |
| CEC2 | d = 10 | $9.70 \times 10^{2}$ | $5.46 \times 10^{0}$ | $6.53 \times 10^{2}$ | $1.59 \times 10^{1}$ | **$3.35 \times 10^{2}$** | $4.61 \times 10^{1}$ | $4.41 \times 10^{2}$ | **$3.84 \times 10^{-1}$** | $3.51 \times 10^{2}$ | $4.32 \times 10^{0}$ |
| | d = 50 | $6.94 \times 10^{2}$ | $2.40 \times 10^{2}$ | $5.39 \times 10^{2}$ | $3.11 \times 10^{2}$ | $4.02 \times 10^{2}$ | $4.11 \times 10^{2}$ | $5.37 \times 10^{2}$ | $3.78 \times 10^{1}$ | **$3.77 \times 10^{2}$** | **$1.60 \times 10^{0}$** |
| CEC3 | d = 10 | $4.80 \times 10^{2}$ | $1.13 \times 10^{2}$ | $4.85 \times 10^{2}$ | $1.97 \times 10^{1}$ | $3.32 \times 10^{2}$ | $4.44 \times 10^{2}$ | $1.00 \times 10^{3}$ | $6.93 \times 10^{2}$ | **$3.28 \times 10^{2}$** | **$6.55 \times 10^{-1}$** |
| | d = 50 | $6.40 \times 10^{2}$ | $3.27 \times 10^{2}$ | $2.66 \times 10^{3}$ | $5.64 \times 10^{2}$ | $5.49 \times 10^{2}$ | $3.13 \times 10^{1}$ | $7.46 \times 10^{2}$ | $2.86 \times 10^{2}$ | **$4.09 \times 10^{2}$** | **$9.85 \times 10^{0}$** |
| CEC4 | d = 10 | $4.51 \times 10^{2}$ | $4.75 \times 10^{-2}$ | $4.22 \times 10^{2}$ | $9.53 \times 10^{-2}$ | $3.69 \times 10^{2}$ | $1.35 \times 10^{1}$ | $4.36 \times 10^{2}$ | $4.30 \times 10^{-2}$ | **$4.02 \times 10^{2}$** | $4.28 \times 10^{-2}$ |
| | d = 50 | $4.89 \times 10^{2}$ | $5.77 \times 10^{-1}$ | $6.35 \times 10^{2}$ | $4.91 \times 10^{-1}$ | $4.62 \times 10^{2}$ | $4.62 \times 10^{1}$ | $5.76 \times 10^{2}$ | $2.73 \times 10^{1}$ | **$4.55 \times 10^{2}$** | **$4.58 \times 10^{-1}$** |
| CEC5 | d = 10 | $6.44 \times 10^{2}$ | $1.14 \times 10^{0}$ | **$5.14 \times 10^{2}$** | $3.24 \times 10^{0}$ | $5.75 \times 10^{2}$ | $1.33 \times 10^{1}$ | $6.44 \times 10^{2}$ | $1.26 \times 10^{0}$ | $5.26 \times 10^{2}$ | **$6.82 \times 10^{-1}$** |
| | d = 50 | $6.97 \times 10^{2}$ | $4.03 \times 10^{0}$ | $5.96 \times 10^{2}$ | $1.95 \times 10^{1}$ | $5.20 \times 10^{2}$ | $4.67 \times 10^{-2}$ | $5.76 \times 10^{2}$ | $2.73 \times 10^{1}$ | **$5.19 \times 10^{2}$** | **$2.73 \times 10^{-2}$** |
| CEC6 | d = 10 | $6.37 \times 10^{2}$ | $2.99 \times 10^{1}$ | $6.47 \times 10^{2}$ | $6.21 \times 10^{1}$ | **$6.01 \times 10^{2}$** | $1.35 \times 10^{-1}$ | $6.72 \times 10^{2}$ | **$6.72 \times 10^{-2}$** | $6.22 \times 10^{2}$ | $1.24 \times 10^{0}$ |
| | d = 50 | $6.84 \times 10^{2}$ | $3.62 \times 10^{1}$ | $6.72 \times 10^{2}$ | $2.51 \times 10^{1}$ | $6.49 \times 10^{2}$ | **$5.23 \times 10^{-1}$** | $6.94 \times 10^{2}$ | $5.26 \times 10^{0}$ | **$6.43 \times 10^{2}$** | $7.30 \times 10^{0}$ |
| CEC7 | d = 10 | $7.66 \times 10^{2}$ | $5.39 \times 10^{1}$ | $8.06 \times 10^{2}$ | $3.67 \times 10^{0}$ | $7.15 \times 10^{2}$ | **$6.59 \times 10^{-3}$** | $8.21 \times 10^{2}$ | $3.00 \times 10^{1}$ | **$7.04 \times 10^{2}$** | $6.39 \times 10^{-1}$ |
| | d = 50 | $8.04 \times 10^{2}$ | $1.41 \times 10^{1}$ | $2.93 \times 10^{3}$ | $4.80 \times 10^{1}$ | $8.01 \times 10^{2}$ | $3.57 \times 10^{0}$ | $1.53 \times 10^{3}$ | $5.12 \times 10^{1}$ | **$7.87 \times 10^{2}$** | **0** |
| CEC8 | d = 10 | $2.41 \times 10^{3}$ | $9.41 \times 10^{1}$ | $1.19 \times 10^{3}$ | $9.13 \times 10^{1}$ | $9.36 \times 10^{2}$ | $5.43 \times 10^{1}$ | $9.77 \times 10^{2}$ | $1.84 \times 10^{2}$ | **$9.04 \times 10^{2}$** | **$4.32 \times 10^{1}$** |
| | d = 50 | $1.47 \times 10^{3}$ | $6.27 \times 10^{2}$ | $1.65 \times 10^{3}$ | $2.41 \times 10^{2}$ | $1.79 \times 10^{3}$ | $1.75 \times 10^{2}$ | **$1.01 \times 10^{3}$** | $3.17 \times 10^{2}$ | $1.23 \times 10^{3}$ | **$8.02 \times 10^{1}$** |
| CEC9 | d = 10 | $1.91 \times 10^{3}$ | $3.72 \times 10^{2}$ | $3.37 \times 10^{3}$ | $8.44 \times 10^{1}$ | $2.85 \times 10^{3}$ | $9.62 \times 10^{-1}$ | $1.38 \times 10^{3}$ | $3.16 \times 10^{2}$ | **$9.44 \times 10^{3}$** | **$7.06 \times 10^{-3}$** |
| | d = 50 | $3.34 \times 10^{3}$ | $8.00 \times 10^{2}$ | $4.17 \times 10^{3}$ | $1.94 \times 10^{2}$ | $1.45 \times 10^{3}$ | $1.16 \times 10^{1}$ | $3.93 \times 10^{3}$ | $2.40 \times 10^{2}$ | **$9.65 \times 10^{3}$** | **$1.00 \times 10^{1}$** |
| CEC10 | d = 10 | $5.34 \times 10^{2}$ | $7.01 \times 10^{2}$ | $2.63 \times 10^{3}$ | $2.61 \times 10^{2}$ | $9.10 \times 10^{2}$ | $8.21 \times 10^{0}$ | $5.20 \times 10^{3}$ | $7.46 \times 10^{0}$ | **$1.03 \times 10^{3}$** | **$3.13 \times 10^{0}$** |
| | d = 50 | $7.99 \times 10^{2}$ | $8.96 \times 10^{2}$ | $9.13 \times 10^{2}$ | $1.36 \times 10^{2}$ | $1.81 \times 10^{3}$ | $1.96 \times 10^{2}$ | $3.18 \times 10^{3}$ | $2.17 \times 10^{2}$ | **$1.08 \times 10^{3}$** | **$4.32 \times 10^{0}$** |
| CEC11 | d = 10 | $1.00 \times 10^{3}$ | $2.85 \times 10^{1}$ | $1.39 \times 10^{3}$ | $3.59 \times 10^{1}$ | $1.13 \times 10^{3}$ | $3.85 \times 10^{1}$ | $1.28 \times 10^{3}$ | $7.62 \times 10^{0}$ | **$1.11 \times 10^{3}$** | **$2.25 \times 10^{0}$** |
| | d = 50 | $1.41 \times 10^{4}$ | $4.28 \times 10^{2}$ | $2.77 \times 10^{3}$ | $2.99 \times 10^{1}$ | $4.31 \times 10^{3}$ | $3.38 \times 10^{2}$ | $7.19 \times 10^{3}$ | $2.85 \times 10^{2}$ | **$1.30 \times 10^{3}$** | **$7.36 \times 10^{0}$** |
| CEC12 | d = 10 | $3.45 \times 10^{3}$ | $2.09 \times 10^{1}$ | $2.07 \times 10^{3}$ | $8.00 \times 10^{0}$ | $2.96 \times 10^{3}$ | $4.45 \times 10^{1}$ | $1.84 \times 10^{3}$ | $4.51 \times 10^{2}$ | **$1.54 \times 10^{3}$** | **$7.31 \times 10^{-1}$** |
| | d = 50 | $4.06 \times 10^{3}$ | $5.23 \times 10^{1}$ | $2.62 \times 10^{3}$ | $1.36 \times 10^{2}$ | **$1.49 \times 10^{3}$** | $5.27 \times 10^{2}$ | $1.78 \times 10^{3}$ | $2.81 \times 10^{2}$ | $2.19 \times 10^{3}$ | **$4.85 \times 10^{0}$** |
| CEC13 | d = 10 | **$1.30 \times 10^{3}$** | $3.60 \times 10^{1}$ | $1.42 \times 10^{3}$ | $7.62 \times 10^{0}$ | $1.33 \times 10^{3}$ | $1.57 \times 10^{1}$ | $6.55 \times 10^{3}$ | $1.98 \times 10^{3}$ | $1.35 \times 10^{3}$ | **$1.43 \times 10^{1}$** |
| | d = 50 | $1.01 \times 10^{4}$ | $1.44 \times 10^{2}$ | $2.58 \times 10^{3}$ | $1.01 \times 10^{2}$ | $1.34 \times 10^{3}$ | $1.83 \times 10^{2}$ | $7.01 \times 10^{3}$ | $7.46 \times 10^{2}$ | **$1.28 \times 10^{3}$** | **$1.20 \times 10^{2}$** |
| CEC14 | d = 10 | $3.79 \times 10^{3}$ | **$2.16 \times 10^{0}$** | $1.71 \times 10^{3}$ | $1.56 \times 10^{2}$ | $1.64 \times 10^{3}$ | $2.27 \times 10^{2}$ | $8.84 \times 10^{3}$ | $2.48 \times 10^{2}$ | **$1.63 \times 10^{3}$** | $5.57 \times 10^{0}$ |
| | d = 50 | $2.95 \times 10^{3}$ | $7.02 \times 10^{1}$ | $3.26 \times 10^{3}$ | $3.71 \times 10^{2}$ | $1.89 \times 10^{3}$ | $1.23 \times 10^{2}$ | $1.07 \times 10^{4}$ | $5.05 \times 10^{2}$ | **$1.57 \times 10^{3}$** | **$8.23 \times 10^{0}$** |
| CEC15 | d = 10 | $2.44 \times 10^{3}$ | $8.15 \times 10^{2}$ | $2.25 \times 10^{3}$ | $5.61 \times 10^{2}$ | $5.49 \times 10^{3}$ | $7.61 \times 10^{1}$ | $1.74 \times 10^{3}$ | $3.46 \times 10^{1}$ | **$1.68 \times 10^{3}$** | **$2.17 \times 10^{1}$** |
| | d = 50 | $3.76 \times 10^{3}$ | $1.54 \times 10^{2}$ | $3.77 \times 10^{3}$ | $9.99 \times 10^{1}$ | $6.25 \times 10^{3}$ | $8.09 \times 10^{0}$ | $1.99 \times 10^{3}$ | $2.27 \times 10^{2}$ | **$1.39 \times 10^{3}$** | **$4.44 \times 10^{0}$** |
| CEC16 | d = 10 | $1.85 \times 10^{3}$ | $5.07 \times 10^{1}$ | $2.50 \times 10^{3}$ | $5.57 \times 10^{1}$ | $1.71 \times 10^{3}$ | $7.33 \times 10^{-1}$ | $1.77 \times 10^{3}$ | $1.65 \times 10^{1}$ | **$1.66 \times 10^{3}$** | **$6.59 \times 10^{-1}$** |
| | d = 50 | $1.90 \times 10^{3}$ | $4.39 \times 10^{1}$ | $3.98 \times 10^{3}$ | $4.35 \times 10^{1}$ | $1.61 \times 10^{3}$ | $4.82 \times 10^{0}$ | $5.28 \times 10^{3}$ | $6.92 \times 10^{2}$ | **$1.60 \times 10^{3}$** | **$4.78 \times 10^{0}$** |
| CEC17 | d = 10 | $2.02 \times 10^{3}$ | $3.18 \times 10^{1}$ | $2.03 \times 10^{3}$ | $5.70 \times 10^{1}$ | $1.73 \times 10^{3}$ | $1.17 \times 10^{0}$ | $1.79 \times 10^{3}$ | $6.83 \times 10^{0}$ | **$1.72 \times 10^{3}$** | **$1.08 \times 10^{0}$** |
| | d = 50 | $1.87 \times 10^{3}$ | **$4.67 \times 10^{1}$** | $3.77 \times 10^{3}$ | $9.92 \times 10^{1}$ | $3.13 \times 10^{3}$ | $5.60 \times 10^{2}$ | $4.46 \times 10^{3}$ | $4.30 \times 10^{2}$ | **$1.83 \times 10^{3}$** | $6.85 \times 10^{1}$ |
| CEC18 | d = 10 | $3.23 \times 10^{3}$ | $1.78 \times 10^{1}$ | $1.90 \times 10^{3}$ | $1.06 \times 10^{2}$ | **$1.86 \times 10^{3}$** | $3.27 \times 10^{2}$ | $1.54 \times 10^{3}$ | $2.12 \times 10^{2}$ | $1.88 \times 10^{3}$ | **$1.25 \times 10^{1}$** |
| | d = 50 | $1.45 \times 10^{3}$ | $5.14 \times 10^{2}$ | $3.26 \times 10^{3}$ | $2.29 \times 10^{1}$ | $2.39 \times 10^{3}$ | $4.38 \times 10^{2}$ | $2.36 \times 10^{3}$ | $7.03 \times 10^{2}$ | **$2.06 \times 10^{3}$** | **$1.73 \times 10^{1}$** |
| CEC19 | d = 10 | $1.46 \times 10^{3}$ | $3.69 \times 10^{2}$ | $4.24 \times 10^{3}$ | $4.50 \times 10^{2}$ | $3.12 \times 10^{3}$ | $1.05 \times 10^{2}$ | $3.18 \times 10^{3}$ | $2.37 \times 10^{2}$ | **$2.27 \times 10^{3}$** | **$1.01 \times 10^{2}$** |
| | d = 50 | $3.36 \times 10^{3}$ | $4.05 \times 10^{2}$ | $5.38 \times 10^{3}$ | $7.82 \times 10^{1}$ | $2.76 \times 10^{3}$ | $9.47 \times 10^{0}$ | $3.37 \times 10^{3}$ | $9.15 \times 10^{1}$ | **$2.69 \times 10^{3}$** | **$5.05 \times 10^{0}$** |
| CEC20 | d = 10 | $4.06 \times 10^{3}$ | $2.70 \times 10^{0}$ | $2.41 \times 10^{3}$ | $2.55 \times 10^{2}$ | $2.63 \times 10^{3}$ | $2.94 \times 10^{1}$ | $1.53 \times 10^{4}$ | $2.40 \times 10^{3}$ | **$2.28 \times 10^{3}$** | **$2.31 \times 10^{0}$** |
| | d = 50 | $3.00 \times 10^{3}$ | $7.18 \times 10^{1}$ | $5.75 \times 10^{3}$ | $9.57 \times 10^{0}$ | $4.79 \times 10^{3}$ | $2.70 \times 10^{2}$ | $5.18 \times 10^{3}$ | $2.17 \times 10^{2}$ | **$2.89 \times 10^{3}$** | **$8.86 \times 10^{0}$** |
| CEC21 | d = 10 | $2.33 \times 10^{3}$ | $1.28 \times 10^{2}$ | $2.37 \times 10^{3}$ | $4.09 \times 10^{2}$ | **$2.10 \times 10^{3}$** | **$3.12 \times 10^{-1}$** | $2.18 \times 10^{3}$ | $9.06 \times 10^{0}$ | $2.03 \times 10^{3}$ | $5.31 \times 10^{0}$ |
| | d = 50 | $2.94 \times 10^{3}$ | $5.05 \times 10^{1}$ | $2.63 \times 10^{3}$ | $5.33 \times 10^{2}$ | $2.64 \times 10^{3}$ | $1.50 \times 10^{2}$ | $2.88 \times 10^{3}$ | $3.54 \times 10^{1}$ | **$2.15 \times 10^{3}$** | **$3.51 \times 10^{1}$** |
| CEC22 | d = 10 | $2.41 \times 10^{3}$ | $6.22 \times 10^{1}$ | $2.88 \times 10^{3}$ | $2.68 \times 10^{1}$ | $2.29 \times 10^{3}$ | $2.84 \times 10^{1}$ | $2.32 \times 10^{3}$ | $2.49 \times 10^{1}$ | **$2.18 \times 10^{3}$** | **$1.97 \times 10^{1}$** |
| | d = 50 | $8.29 \times 10^{3}$ | $8.13 \times 10^{2}$ | $1.09 \times 10^{4}$ | $3.87 \times 10^{2}$ | $2.31 \times 10^{3}$ | $7.52 \times 10^{1}$ | $1.60 \times 10^{4}$ | $3.83 \times 10^{2}$ | **$2.28 \times 10^{3}$** | **$5.36 \times 10^{1}$** |
| CEC23 | d = 10 | $2.67 \times 10^{3}$ | $4.28 \times 10^{1}$ | $2.59 \times 10^{3}$ | $3.06 \times 10^{2}$ | $2.63 \times 10^{3}$ | **0** | $2.69 \times 10^{3}$ | $7.35 \times 10^{0}$ | **$2.37 \times 10^{3}$** | $4.28 \times 10^{1}$ |
| | d = 50 | $4.13 \times 10^{3}$ | $1.59 \times 10^{2}$ | $2.90 \times 10^{3}$ | $5.17 \times 10^{2}$ | $2.64 \times 10^{3}$ | **$4.48 \times 10^{-1}$** | $3.64 \times 10^{3}$ | $1.25 \times 10^{2}$ | **$2.29 \times 10^{3}$** | $4.11 \times 10^{1}$ |
| CEC24 | d = 10 | $2.64 \times 10^{3}$ | $3.22 \times 10^{1}$ | $2.73 \times 10^{3}$ | $6.85 \times 10^{1}$ | $2.29 \times 10^{3}$ | $2.38 \times 10^{0}$ | $2.73 \times 10^{3}$ | $8.76 \times 10^{0}$ | **$2.46 \times 10^{3}$** | **$2.02 \times 10^{0}$** |
| | d = 50 | $3.77 \times 10^{3}$ | $2.65 \times 10^{1}$ | $2.99 \times 10^{3}$ | $8.36 \times 10^{1}$ | $2.68 \times 10^{3}$ | $6.69 \times 10^{-1}$ | $3.50 \times 10^{3}$ | $3.29 \times 10^{2}$ | **$2.43 \times 10^{3}$** | **$1.99 \times 10^{-1}$** |
| CEC25 | d = 10 | $3.36 \times 10^{3}$ | $2.45 \times 10^{2}$ | $1.03 \times 10^{4}$ | $4.42 \times 10^{2}$ | $2.90 \times 10^{3}$ | $1.53 \times 10^{2}$ | $4.45 \times 10^{3}$ | $5.55 \times 10^{2}$ | **$2.64 \times 10^{3}$** | **$1.45 \times 10^{2}$** |
| | d = 50 | $6.63 \times 10^{3}$ | $2.52 \times 10^{1}$ | $9.13 \times 10^{3}$ | $3.57 \times 10^{2}$ | $3.18 \times 10^{3}$ | $3.41 \times 10^{2}$ | $6.29 \times 10^{3}$ | $8.12 \times 10^{1}$ | **$3.02 \times 10^{3}$** | **$2.21 \times 10^{1}$** |
| CEC26 | d = 10 | $4.69 \times 10^{3}$ | $1.58 \times 10^{3}$ | $3.89 \times 10^{3}$ | $2.29 \times 10^{2}$ | $4.45 \times 10^{3}$ | $1.91 \times 10^{2}$ | $4.84 \times 10^{3}$ | $6.35 \times 10^{0}$ | **$3.50 \times 10^{3}$** | **$4.75 \times 10^{0}$** |
| | d = 50 | $3.08 \times 10^{3}$ | $9.14 \times 10^{2}$ | $4.03 \times 10^{3}$ | $1.57 \times 10^{2}$ | $4.62 \times 10^{3}$ | $2.16 \times 10^{2}$ | $3.98 \times 10^{3}$ | $9.50 \times 10^{2}$ | **$2.94 \times 10^{3}$** | **$1.38 \times 10^{2}$** |
| CEC27 | d = 10 | $4.30 \times 10^{3}$ | $3.28 \times 10^{2}$ | **$2.29 \times 10^{3}$** | $1.11 \times 10^{2}$ | $3.18 \times 10^{3}$ | **$4.14 \times 10^{1}$** | $5.52 \times 10^{3}$ | $3.08 \times 10^{2}$ | $3.18 \times 10^{3}$ | $2.20 \times 10^{2}$ |
| | d = 50 | $3.01 \times 10^{3}$ | $5.24 \times 10^{1}$ | $3.53 \times 10^{3}$ | $3.34 \times 10^{2}$ | **$2.89 \times 10^{3}$** | $1.54 \times 10^{2}$ | $7.85 \times 10^{3}$ | $1.25 \times 10^{2}$ | $3.04 \times 10^{3}$ | **$4.00 \times 10^{1}$** |
| CEC28 | d = 10 | $3.44 \times 10^{3}$ | $4.37 \times 10^{2}$ | $4.88 \times 10^{3}$ | $1.20 \times 10^{2}$ | $3.19 \times 10^{3}$ | $2.40 \times 10^{2}$ | $6.07 \times 10^{3}$ | $1.37 \times 10^{2}$ | **$2.99 \times 10^{3}$** | **$7.49 \times 10^{1}$** |
| | d = 50 | $5.96 \times 10^{3}$ | $2.27 \times 10^{0}$ | $6.79 \times 10^{3}$ | $6.17 \times 10^{1}$ | $3.91 \times 10^{3}$ | $2.87 \times 10^{2}$ | $9.38 \times 10^{3}$ | $3.08 \times 10^{2}$ | **$3.37 \times 10^{3}$** | **$1.56 \times 10^{0}$** |
| CEC29 | d = 10 | **$2.82 \times 10^{3}$** | $8.13 \times 10^{2}$ | $3.11 \times 10^{3}$ | $3.18 \times 10^{2}$ | $2.27 \times 10^{3}$ | $1.50 \times 10^{2}$ | $4.58 \times 10^{3}$ | $9.71 \times 10^{1}$ | $3.10 \times 10^{3}$ | **$7.58 \times 10^{1}$** |
| | d = 50 | $2.76 \times 10^{3}$ | $3.03 \times 10^{2}$ | $4.35 \times 10^{3}$ | $6.07 \times 10^{2}$ | $1.14 \times 10^{4}$ | $3.62 \times 10^{3}$ | $6.61 \times 10^{3}$ | $7.49 \times 10^{2}$ | **$3.18 \times 10^{3}$** | **$2.74 \times 10^{2}$** |

As can be seen from Table 6, when dealing with complex CEC2017 test functions, the performance of all algorithms is degraded, and it becomes rare to directly obtain theoretical values. On functions CEC1, CEC4, CEC7, CEC10, CEC11, CEC16, CEC17, CEC22, CEC23, and CEC24, the results of TRS-ChOA are very close to the theoretical optimal value, and the accuracy is higher than other algorithms. When the dimension $d = 10$, the mean value of GWO on CEC2, CEC6, CEC18, and CEC21 is closer to the optimal value than that of TRS-ChOA, but

TRS-ChOA performs better when the dimension increases. In addition, the standard deviation of TRS-ChOA is generally superior to the other four algorithms, indicating that TRS-ChOA has stronger stability. In conclusion, compared with the other four algorithms, the proposed TRS-ChOA has more prominent advantages in the CEC2017 test functions, which indicates that TRS-CHOA has better robustness and can adapt to more complex problems.

By conducting experiments on 13 benchmark functions and 29 CEC2017 complex functions, it was proven that the three proposed improvement strategies effectively enhanced the performance of the original ChOA. TRS-ChOA demonstrates faster search speed and higher convergence accuracy compared to ChOA in both low- and high-dimensional benchmark functions. When solving CEC2017 complex functions, TRS-ChOA exhibits superior optimization performance over the other four algorithms and demonstrates good stability. In summary, TRS-ChOA has better optimization performance and robustness; it can not only handle simple single-mode problems but also solve high-dimensional multi-constraint optimization problems.

## 6. UAV Path Planning Test

In Section 2, we establish a mathematical model for the UAV path planning problem. Based on this model, we design three different 3D environments and utilize the proposed algorithms to plan flight paths for UAVs in these environments, aiming to mitigate the randomness of experimental results. In these models, there are challenges such as mountains, buildings (modeled as regular prisms), and areas with enemy radar scanning (modeled as cylinders), which pose threats to UAV flight. In order to verify the effectiveness of TRS-ChOA in solving UAV path planning, WOA, ALO, SSA, ChOA, and GWO were selected and compared with TRS-ChOA in different 3D environment models.

### *6.1. Parameter Settings*

To ensure the fairness of the experiment, the basic parameters of each algorithm are set uniformly: population size $N = 30$, maximum number of iterations $T = 50$. The internal parameters of WOA, ALO, GWO, SSA, ChOA, and TRS-ChOA are shown in Table 1. All environmental models have an area of $220 \times 220$ without limiting the height of the models. The basic terrain is constructed by Equation (1), and the mountains and human engineering are then superimposed onto the model to make it more realistic. The starting and ending point coordinates of the UAV are (20, 20, 20) and (200, 200, 20), respectively. The altitudes of the starting point and the ending point are 20 to avoid the problem of excessive climbing angle or descending angle of the UAV during takeoff or landing, which may lead to the crash of the UAV. Setting the height to 20 also tests whether the algorithm can search down in 3D space. In addition, since the height of obstacles in the model is all greater than 20, as long as the UAV flies over the obstacles, there will be a change in height, which will generate losses in Equation (5). This is fair to all algorithms participating in the simulation. The specific parameters of the mountain areas, buildings, and radar scanning areas in the model are shown in Table 7. Figure 4 shows a 3D environment built from the data in Table 7.

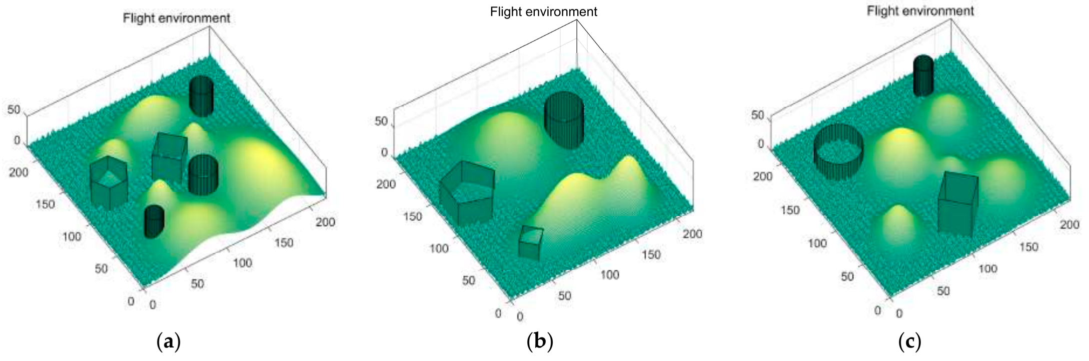

**Figure 4.** UAV trajectory planning environment model. (**a**) Environment 1. (**b**) Environment 2. (**c**) Environment 3.

**Table 7.** Parameters of the 3D environment model.

| 3D Environment | Threat Areas | Parameters | Value | | | | | |
|---|---|---|---|---|---|---|---|---|
| Model 1 | mountains | Central coordinate | [80,25] | [70,80] | [175,45] | [140,125] | [60,150] | [120,175] |
| | | height | 40 | 40 | 50 | 40 | 40 | 45 |
| | | Slope in the X direction | 40 | 15 | 35 | 15 | 20 | 35 |
| | | Slope in the Y direction | 40 | 15 | 60 | 15 | 20 | 20 |
| | buildings | Central coordinate | | [35,120] | | | [105,115] | |
| | | height | | 40 | | | 40 | |
| | | Apothem | | 15 | | | 15 | |
| | | Side length | | 30 | | | 15 | |
| | Radar areas | Central coordinate | | [45,50] | | [120,75] | [175,165] | |
| | | height | | 40 | | 40 | 40 | |
| | | radius | | 10 | | 15 | 12 | |
| Model 2 | mountains | Central coordinate<br>height | | [100,160]<br>60 | | [170,40]<br>70 | [105,50]<br>80 | |
| | | Slope in the X direction | | 40 | | 20 | 45 | |
| | | Slope in the Y direction | | 40 | | 20 | 20 | |
| | buildings | Central coordinate | | [30,125] | | | [50,40] | |
| | | height | | 40 | | | 30 | |
| | | Apothem | | 24 | | | 10 | |
| | | Side length | | 34.87 | | | 20 | |
| | Radar areas | Central coordinate | | | | [160,150] | | |
| | | height | | | | 50 | | |
| | | radius | | | | 20 | | |
| Model 3 | mountains | Central coordinate<br>height | [50,60]<br>40 | [130,100]<br>40 | | [135,90]<br>50 | [170,150]<br>40 | [170,50]<br>40 |
| | | Slope in the X direction | 40 | 15 | | 60 | 15 | 20 |
| | | Slope in the Y direction | 40 | 15 | | 35 | 15 | 20 |
| | buildings | Central coordinate | | | | [110,40] | | |
| | | height | | | | 60 | | |
| | | Apothem | | | | 17.3 | | |
| | | Side length | | | | 34.6 | | |
| | Radar areas | Central coordinate | | [50,170] | | | [170,195] | |
| | | height | | 30 | | | 50 | |
| | | radius | | 25 | | | 10 | |

### *6.2. Simulation Experiment and Results*

Figure 5 shows the optimization results of the six algorithms in three simulation environments, and Figure 6 shows their corresponding vertical views. As can be seen from Figure 5, all six algorithms can find the flight path from the starting point to the destination. However, in environment 1, to bypass human engineering at the center of the map, ALO, WOA, and GWO choose to increase the flight altitude of the UAV and fly along a route close to the mountain. This increases the distance of the flight and the likelihood of encountering danger. In environment 2, ALO, WOA, and SSA also chose to fly over the mountain on the left, resulting in the UAV wasting more resources. In environment 3, we place threatening human engineering on both sides of the map, but WOA, ALO, SSA, ChOA, and GWO still can't find the best path at the center of the map. In contrast, TRS-ChOA finds a better path in all three scenarios, not only avoiding the threat area but also always staying at a low flight altitude (flying over mountains or human engineering as little as possible), greatly shortening the UAV's flight distance. In addition, the path planned by TRS-ChOA in Figure 6 is smoother than other algorithms and maintains a safe distance from the threat area on the map. In environment 1, the steering angle of the paths planned by SSA and WOA is too large, which makes it easy to cause a crash accident. The routes of other algorithms in environment 2 and environment 3 fluctuate greatly and are close to mountains, which also increases the risk factor. Therefore, the UAV trajectory planned by TRS-ChOA is more secure and stable.

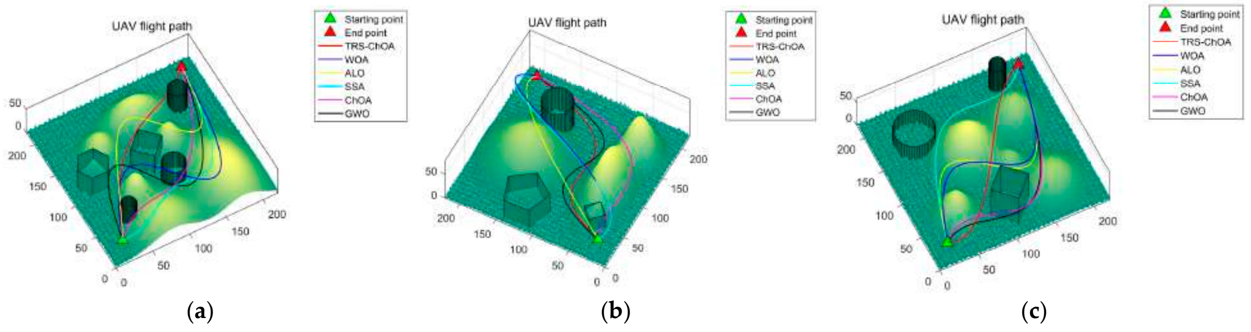

**Figure 5.** UAV-simulated flight path. (**a**) Environment 1 route, (**b**) environment 2 route, (**c**) environment 3 route.

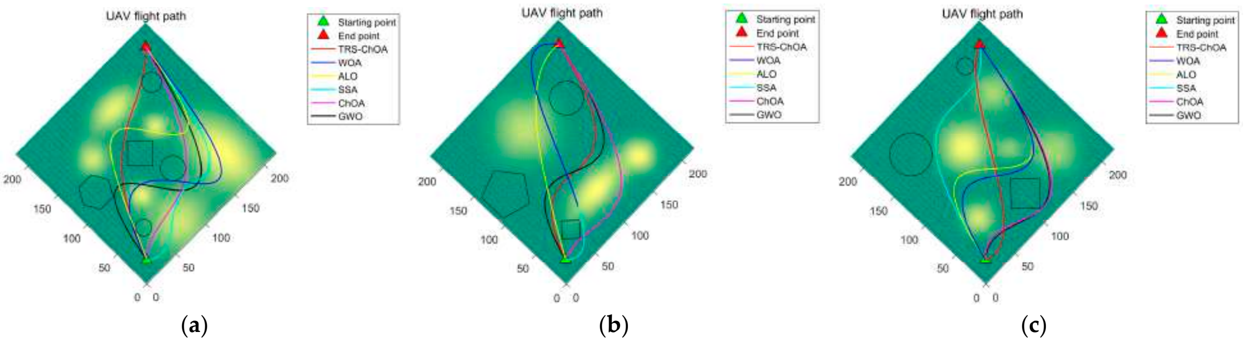

**Figure 6.** UAV simulated flight path (vertical view). (**a**) Environment 1 route (vertical view), (**b**) environment 2 route (vertical view), (**c**) environment 3 route (vertical view).

To more scientifically and objectively compare the performance of TRS-ChOA with five other algorithms in trajectory planning, we repeated the tests 30 times in all three environments. Table 8 records the length of the trajectories obtained by the six algorithms. Figure 7 shows the average convergence curves of the six algorithms in the path planning experiment. Table 9 records the loss function values of the trajectories obtained by each algorithm.

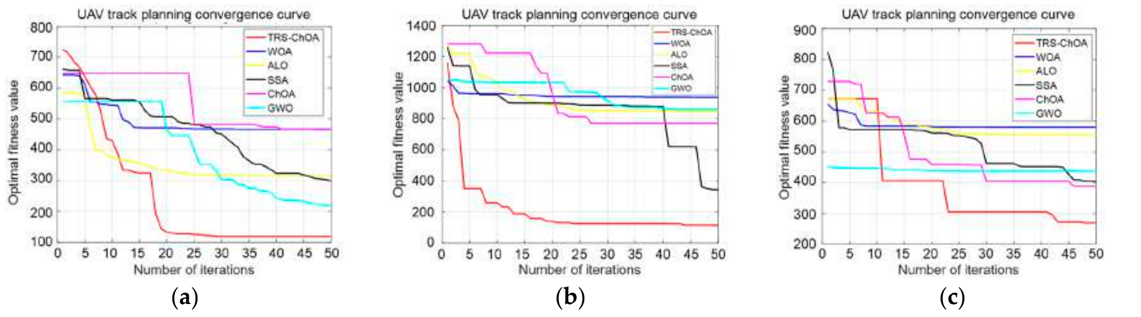

**Figure 7.** Convergence curve of the trajectory loss function. (**a**) Convergence curve of the loss function in environment 1. (**b**) Convergence curve of the loss function in environment 2. (**c**) Convergence curve of the loss function in environment 3.

**Table 8.** Path length of each algorithm.

| 3D Model | Path Length / Algorithm | Longest | Shortest | Mean |
|---|---|---|---|---|
| Environment 1 | WOA | 410.32 | 284.73 | 390.27 |
| | ALO | 399.76 | 363.45 | 374.18 |
| | SSA | 341.25 | 295.07 | 318.08 |
| | ChOA | 292.08 | 259.62 | 283.24 |
| | GWO | 406.91 | 372.94 | 381.06 |
| | TRS-ChOA | 286.40 | 251.79 | **263.29** |
| Environment 2 | WOA | 354.61 | 293.55 | 318.46 |
| | ALO | 317.42 | 261.00 | 284.72 |
| | SSA | 320.08 | 294.18 | 315.65 |
| | ChOA | 327.10 | 281.38 | 304.01 |
| | GWO | 321.94 | 300.25 | 309.19 |
| | TRS-ChOA | 297.36 | 274.82 | **279.56** |
| Environment 3 | WOA | 352.47 | 312.29 | 338.97 |
| | ALO | 354.12 | 347.53 | 350.68 |
| | SSA | 297.74 | 263.80 | 285.88 |
| | ChOA | 333.26 | 304.91 | 317.93 |
| | GWO | 324.59 | 299.47 | 313.93 |
| | TRS-ChOA | 284.06 | 262.56 | **267.71** |

**Table 9.** Loss function value of each algorithm.

| 3D Model | Fitness Value / Algorithm | Optimal | Worst | Mean |
|---|---|---|---|---|
| Environment 1 | WOA | 464.13 | 487.51 | 477.19 |
| | ALO | 300.00 | 364.74 | 311.42 |
| | SSA | 287.85 | 329.99 | 300.61 |
| | ChOA | 463.11 | 490.07 | 476.08 |
| | GWO | 198.65 | 347.61 | 224.93 |
| | TRS-ChOA | 113.60 | 152.79 | **115.37** |
| Environment 2 | WOA | 392.64 | 985.06 | 937.40 |
| | ALO | 435.29 | 908.31 | 850.31 |
| | SSA | 224.57 | 678.34 | 343.06 |
| | ChOA | 339.15 | 852.38 | 771.56 |
| | GWO | 455.74 | 890.00 | 860.67 |
| | TRS-ChOA | 119.48 | 481.90 | **125.04** |
| Environment 3 | WOA | 314.60 | 869.75 | 573.51 |
| | ALO | 428.00 | 589.91 | 560.29 |
| | SSA | 365.77 | 714.70 | 401.57 |
| | ChOA | 316.49 | 462.39 | 385.48 |
| | GWO | 293.68 | 613.44 | 442.83 |
| | TRS-ChOA | 250.84 | 419.56 | **268.16** |

According to Table 8, in environment 1, the average path length of TRS-ChOA is shortened by 32.5%, 29.6%, 17.2%, 7.0%, and 30.9% compared to WOA, ALO, SSA, ChOA, and GWO, respectively, with an average reduction of 23.4%. In environment 2, the average path length of TRS-ChOA is shortened by 12.2%, 1.8%, 11.4%, 8.0%, and 9.6% compared to WOA, ALO, SSA, ChOA, and GWO, respectively, with an average reduction of 8.6%. In environment 3, the average path length of TRS-ChOA is shortened by 21.0%, 23.7%, 6.4%, 15.8%, and 14.7% compared to WOA, ALO, SSA, ChOA, and GWO, respectively, with an average reduction of 16.3%. These data indicate that TRS-ChOA can find a shorter path connecting the starting point and ending point in a complex 3D environment, saving more energy and flight time for UAVs.

As can be seen from Figure 7, the fitness values of TRS-ChOA in the three models are lower than those of the other five algorithms. In Figure 7a,c, although TRS-ChOA is not the algorithm with the fastest convergence speed in the early stage of iteration, it can jump out of the local optima and find the global optima in the later stage. In Figure 7b, TRS-ChOA converges quickly at the beginning of the iteration and finds the optimal value. Therefore, TRS-ChOA has a better global optimization capability than the other five algorithms and can minimize the loss of UAV path planning.

According to Table 9, in environment 1, the average fitness value of TRS-ChOA is decreased by 75.8%, 63.0%, 61.6%, 75.8%, and 48.7% compared to WOA, ALO, SSA, ChOA, and GWO, respectively, with an average reduction of 65.0%. In environment 2, the average fitness value of TRS-ChOA has decreased by 86.7%, 85.3%, 63.6%, 83.8%, and 85.5% compared to WOA, ALO, SSA, ChOA, and GWO, respectively, with an average reduction of 81.0%. In environment 3, the average fitness value of TRS-ChOA has decreased by 53.2%, 52.1%, 33.2%, 30.4%, and 39.4% compared to WOA, ALO, SSA, ChOA, and GWO, respectively, with an average reduction of 41.7%. In addition, the difference between the optimal and worst fitness values of TRS-ChOA is small, indicating that TRS-ChOA has excellent stability. In summary, TRS-ChOA has a stronger performance in solving multi-constraint combinatorial optimization problems. In the process of planning the flight path for UAVs, it can better balance the flight distance, flight altitude, and steering angle, saving more energy and time while ensuring the safety of UAVs.

## 7. Conclusions

In this paper, we propose an improved version of the Chimp Optimization Algorithm (ChOA), called TRS-ChOA, for solving the path planning problem of UAVs in 3D environments. We construct a challenging simulated 3D environment that includes various terrain and buildings that may pose threats to UAV flights. Additionally, we consider three performance constraints that UAVs need to consider during the path planning process: flight distance, climbing height, and turning angle, and integrate them into a comprehensive loss function for path planning. We make the following improvements to ChOA: firstly, we integrate the differential evolution mutation operator into ChOA to enhance the quality of individuals in the population. Secondly, we utilize improved reverse learning to expand the search space effectively, avoiding the possibility of missing high-quality solutions. Finally, we introduce similarity preference weights to prevent search agents from excessively assimilating, thereby enhancing the algorithm's ability to escape from local optima. In benchmark and CEC2017 complex function optimization tests, TRS-ChOA outperforms ChOA and other algorithms in both low-dimensional and high-dimensional scenarios. This demonstrates that these three improvement strategies effectively enhance the convergence speed, convergence accuracy, stability, and robustness of ChOA.

We conducted simulation experiments on path planning using TRS-ChOA and five well-known algorithms in three different 3D environments. The experimental results show that, compared to the other five algorithms, TRS-ChOA can obtain shorter, smoother, and safer flight paths in all three scenarios. It effectively reduces the flying time and distance of the UAV while avoiding threat areas on the map, ensuring safety during the UAV's flight. Additionally, the paths planned by TRS-ChOA try to avoid directly crossing obstacles in the environment. This eliminates the safety hazards caused by large-scale climbing or descending the UAV, as well as reducing energy consumption. We repeated the experiments 30 times in each of the three scenarios, and the results demonstrate that the average length of paths planned by TRS-ChOA is the shortest among all the algorithms. Compared to the other algorithms, it reduces the average length by 23.4%, 8.6%, and 16.3% in the three scenarios, respectively. Meanwhile, the fitness value of the paths generated by TRS-ChOA is reduced by an average of 65.0%, 81.0%, and 41.7% in the three scenarios, respectively. It can be seen from these data that TRS-ChOA exhibits good reliability and effectiveness in UAV path planning in 3D environments, providing a new strategy for solving such multi-constraint combinatorial optimization problems. However, in our research, we simplified the UAV as a uniformly moving point to simplify the flight model and ignored many dynamic issues that need to be considered in reality. Moreover, we constructed a static environment where the positions and sizes of obstacles are fixed. In the future, we will introduce more factors that may affect UAV flight, construct more realistic flight models and dynamic 3D environments, and further improve TRS-ChOA to plan flight paths for UAVs in real-time changing environments.

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

**Funding:** The research was funded by the National Natural Science Foundation of China "Research on the Evidence Chain Construction from the Analysis of the Investigation Documents (62166006)", supported by Guizhou Provincial Science and Technology Support Projects "Key Core Technologies for Spatial Optimisation and Allocation of Distributed Science and Technology Resources (Qiankehe Support [2023] General 093)", supported by Guizhou Provincial Science and Technology Projects "Research on The Key Technology of Constructing Evidence Relationship Diagram for Smart Court (Guizhou Science Foundation-ZK [2021] General 335)".

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Fan, B.; Li, Y.; Zhang, R.; Fu, Q. Review on the technological development and application of UAV systems. *Chin. J. Electron.* **2020**, *29*, 199–207. [CrossRef]
2. Aggarwal, S.; Kumar, N. Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges. *Comput. Commun.* **2020**, *149*, 270–299. [CrossRef]
3. Ren, H.; Zhao, Y.; Xiao, W.; Hu, Z. A review of UAV monitoring in mining areas: Current status and future perspectives. *Int. J. Coal Sci. Technol.* **2019**, *6*, 320–333. [CrossRef]
4. Zhao, Y.; Zheng, Z.; Liu, Y. Survey on computational-intelligence-based UAV path planning. *Knowl.-Based Syst.* **2018**, *158*, 54–64. [CrossRef]
5. Shin, J.-J.; Bang, H. UAV path planning under dynamic threats using an improved PSO algorithm. *Int. J. Aerosp. Eng.* **2020**, *2020*, 8820284. [CrossRef]
6. Liu, W.; Zheng, Z.; Cai, K.-Y. Bi-level programming based real-time path planning for unmanned aerial vehicles. *Knowl.-Based Syst.* **2013**, *44*, 34–47. [CrossRef]
7. Lu, N.; Zhou, Y.; Shi, C.; Cheng, N.; Cai, L.; Li, B. Planning while flying: A measurement-aided dynamic planning of drone small cells. *IEEE Internet Things J.* **2018**, *6*, 2693–2705. [CrossRef]
8. Lluvia, I.; Lazkano, E.; Ansuategi, A. Active mapping and robot exploration: A survey. *Sensors* **2021**, *21*, 2445. [CrossRef]
9. Munoz, P.; Rodriguez-Moreno, M. Improving efficiency in any-angle path-planning algorithms. In Proceedings of the 2012 6th IEEE International Conference Intelligent Systems, Sofia, Bulgaria, 6–8 September 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 213–218.
10. Choset, H.; Lynch, K.M.; Hutchinson, S.; Kantor, G.A.; Burgard, W. *Principles of Robot Motion: Theory, Algorithms, and Implementations*; MIT Press: Cambridge, MA, USA, 2005.
11. Kim, J.; Kim, S.; Choo, Y. Stealth path planning for a high speed torpedo-shaped autonomous underwater vehicle to approach a target ship. *Cyber-Phys. Syst.* **2018**, *4*, 1–16. [CrossRef]
12. Pettie, S. A new approach to all-pairs shortest paths on real-weighted graphs. *Theor. Comput. Sci.* **2004**, *312*, 47–74. [CrossRef]
13. Nash, A.; Daniel, K.; Koenig, S.; Felner, A. Theta*: Any-angle path planning on grids. *AAAI* **2007**, *7*, 1177–1183.
14. Kim, J. Fast Path Planning of Autonomous Vehicles in 3D Environments. *Appl. Sci.* **2022**, *12*, 4014. [CrossRef]
15. Fulcher, J. Computational intelligence: An introduction. In *Computational Intelligence: A Compendium*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 3–78.
16. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN′95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; IEEE: Piscataway, NJ, USA, 1995; pp. 1942–1948.
17. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
18. Yang, X.-S.; Deb, S. Cuckoo search via Lévy flights. In Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, India, 9–11 December 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 210–214.
19. Xue, J.; Shen, B. A novel swarm intelligence optimization approach: Sparrow search algorithm. *Syst. Sci. Control Eng.* **2020**, *8*, 22–34. [CrossRef]
20. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]

21. Mirjalili, S. The ant lion optimizer. *Adv. Eng. Softw.* **2015**, *83*, 80–98. [CrossRef]
22. Kaur, S.; Awasthi, L.K.; Sangal, A.; Dhiman, G. Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Eng. Appl. Artif. Intell.* **2020**, *90*, 103541. [CrossRef]
23. Poudel, S.; Arafat, M.Y.; Moh, S. Bio-Inspired Optimization-Based Path Planning Algorithms in Unmanned Aerial Vehicles: A Survey. *Sensors* **2023**, *23*, 3051. [CrossRef]
24. Roberge, V.; Tarbouchi, M.; Labonté, G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *IEEE Trans. Ind. Inform.* **2012**, *9*, 132–141. [CrossRef]
25. Wen, X.; Ruan, Y.; Li, Y.; Xia, H.; Zhang, R.; Wang, C.; Liu, W.; Jiang, X. Improved genetic algorithm based 3-D deployment of UAVs. *J. Commun. Netw.* **2022**, *24*, 223–231. [CrossRef]
26. Guan, Y.; Gao, M.; Bai, Y. Double-ant colony based UAV path planning algorithm. In Proceedings of the 2019 11th International Conference on Machine Learning and Computing, Zhuhai, China, 22–24 February 2019; pp. 258–262.
27. Chai, X.; Zheng, Z.; Xiao, J.; Yan, L.; Qu, B.; Wen, P.; Wang, H.; Zhou, Y.; Sun, H. Multi-strategy fusion differential evolution algorithm for UAV path planning in complex environment. *Aerosp. Sci. Technol.* **2022**, *121*, 107287. [CrossRef]
28. Zhang, R.; Li, S.; Ding, Y.; Qin, X.; Xia, Q. UAV Path Planning Algorithm Based on Improved Harris Hawks Optimization. *Sensors* **2022**, *22*, 5232. [CrossRef] [PubMed]
29. Ji, Y.; Zhao, X.; Hao, J. A novel UAV path planning algorithm based on double-dynamic biogeography-based learning particle swarm optimization. *Mob. Inf. Syst.* **2022**, *2022*, 8519708. [CrossRef]
30. Qu, C.; Gai, W.; Zhang, J.; Zhong, M. A novel hybrid grey wolf optimizer algorithm for unmanned aerial vehicle (UAV) path planning. *Knowl.-Based Syst.* **2020**, *194*, 105530. [CrossRef]
31. Qu, C.; Gai, W.; Zhong, M.; Zhang, J. A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles (UAVs) path planning. *Appl. Soft Comput.* **2020**, *89*, 106099. [CrossRef]
32. Yu, X.; Li, C.; Zhou, J. A constrained differential evolution algorithm to solve UAV path planning in disaster scenarios. *Knowl.-Based Syst.* **2020**, *204*, 106209. [CrossRef]
33. Jiang, W.; Lyu, Y.; Li, Y.; Guo, Y.; Zhang, W. UAV path planning and collision avoidance in 3D environments based on POMPD and improved grey wolf optimizer. *Aerosp. Sci. Technol.* **2022**, *121*, 107314. [CrossRef]
34. Du, N.; Zhou, Y.; Deng, W.; Luo, Q. Improved chimp optimization algorithm for three-dimensional path planning problem. *Multimed. Tools Appl.* **2022**, *81*, 27397–27422. [CrossRef]
35. Khishe, M.; Mosavi, M.R. Chimp optimization algorithm. *Expert Syst. Appl.* **2020**, *149*, 113338. [CrossRef]
36. Kaur, M.; Kaur, R.; Singh, N.; Dhiman, G. Schoa: A newly fusion of sine and cosine with chimp optimization algorithm for hls of datapaths in digital filters and engineering applications. *Eng. Comput.* **2021**, *38*, 975–1003. [CrossRef]
37. Hu, T.; Khishe, M.; Mohammadi, M.; Parvizi, G.-R.; Karim, S.H.T.; Rashid, T.A. Real-time COVID-19 diagnosis from X-Ray images using deep CNN and extreme learning machines stabilized by chimp optimization algorithm. *Biomed. Signal Process Control* **2021**, *68*, 102764. [CrossRef] [PubMed]
38. Houssein, E.H.; Emam, M.M.; Ali, A.A. An efficient multilevel thresholding segmentation method for thermography breast cancer imaging based on improved chimp optimization algorithm. *Expert Syst. Appl.* **2021**, *185*, 115651. [CrossRef]
39. Thompson, S.E.; Patel, R.V. Formulation of joint trajectories for industrial robots using B-splines. *IEEE Trans. Ind. Electron.* **1987**, *IE-34*, 192–199. [CrossRef]
40. Tisdale, J.; Kim, Z.; Hedrick, J.K. Autonomous UAV path planning and estimation. *IEEE Robot. Autom. Mag.* **2009**, *16*, 35–42. [CrossRef]
41. Lv, Z.; Yang, L.; He, Y.; Liu, Z.; Han, Z. 3D environment modeling with height dimension reduction and path planning for UAV. In Proceedings of the 2017 9th International Conference on Modelling, Identification and Control (ICMIC), Kunming, China, 10–12 July 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 734–739.
42. Hussain, A.; Muhammad, Y.S.; Nauman Sajid, M.; Hussain, I.; Mohamd Shoukry, A.; Gani, S. Genetic algorithm for traveling salesman problem with modified cycle crossover operator. *Comput. Intell. Neurosci.* **2017**, *2017*, 7430125. [CrossRef]
43. Ergezer, H.; Leblebicioğlu, K. 3D path planning for multiple UAVs for maximum information collection. *J. Intell. Robot. Syst.* **2014**, *73*, 737–762. [CrossRef]
44. Besada-Portas, E.; de la Torre, L.; de la Cruz, J.M.; de Andrés-Toro, B. Evolutionary trajectory planner for multiple UAVs in realistic scenarios. *IEEE Trans. Robot.* **2010**, *26*, 619–634. [CrossRef]
45. McKinley, S.; Levine, M. Cubic spline interpolation. *Coll. Redw.* **1998**, *45*, 1049–1060.
46. Brest, J.; Zumer, V.; Maucec, M.S. Self-adaptive differential evolution algorithm in constrained real-parameter optimization. In Proceedings of the 2006 IEEE international conference on evolutionary computation, Vancouver, BC, Canada, 16–21 July 2006; IEEE: Piscataway, NJ, USA, 2006; pp. 215–222.
47. Gong, W.; Cai, Z.; Ling, C.X. DE/BBO: A hybrid differential evolution with biogeography-based optimization for global numerical optimization. *Soft Comput.* **2010**, *15*, 645–665. [CrossRef]
48. Wu, G.; Mallipeddi, R.; Suganthan, P.N. *Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization*; Technical Report; National University of Defense Technology: Changsha, China; Kyungpook National University: Daegu, Republic of Korea; Nanyang Technological University: Singapore, 2017.

49. Suganthan, P.N.; Hansen, N.; Liang, J.J.; Deb, K.; Auger, A. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. *KanGAL Rep.* **2005**, *2005005*, 2005.
50. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18. [CrossRef]