Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

MultiAO: A Novel Optimizer Inspired by Collaborative Intelligence in Complex Systems for Delivery Path Planning in UAV Clusters

Weifei Gan

wfgan@sspu.edu.cn

Shanghai Polytechnic University

Xiliang Zhang

Shanghai Polytechnic University

Research Article

Keywords: Cluster Optimization, MultiAO Algorithm, Metaheuristic Algorithm, Multi-Agent System, Unmanned Aerial Vehicle Path Planning

Posted Date: March 19th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9010250/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

MultiAO: A Novel Optimizer Inspired by Collaborative Intelligence in Complex Systems for Delivery Path Planning in UAV Clusters

Weifei Gan [a, *] , Xiliang Zhang [b, *]

[a] School of Computer and Information Engineering, Institute for Artificial Intelligence,Shanghai Polytechnic University, Shanghai 201209, China

[b] School of Intelligent Manufacturing and Control Engineering, Shanghai Polytechnic University, Shanghai 201209, China

[*] These authors are corresponding author e-mail: wfgan@sspu.edu.cn; xlzhang@sspu.edu.cn

Abstract. Within the complex domain of cluster optimization, high-dimensional decision-making environments have persistently confronted formidable optimization bottlenecks. To address these challenges, this paper proposes a novel multi-agent optimization algorithm predicated upon a state-dependent dynamic evolution mechanism, designated as the Multi-Agent Optimizer (MultiAO). The primary contributions of this work are threefold. First, we introduce an innovative algorithmic framework characterized by three distinct evolutionary phases: gravitational accretion and centroid rebellion, orthogonal singularity decoupling, and a hysteresis effect coupled with a memory potential field, which accurately capture agent state changes across different search stages. Second, we provide rigorous empirical validation across two dimensions of the CEC2022 benchmark suite. MultiAO ranked first in comparative analyses against five highly-cited and five recently proposed algorithms, yielding average Friedman rankings of 2 and 2.25. Finally, we successfully deploy the algorithm to solve complex real-world engineering challenges, specifically unmanned aerial vehicle (UAV) path planning across urban, mountainous, and composite terrains. In both single-UAV and multi-UAV contexts, MultiAO consistently exhibited optimal convergence, achieving a minimum optimal value as low as 324.332 in multi-UAV scenarios. These contributions collectively demonstrate the superior cluster optimization capabilities of MultiAO, establishing it as a highly competitive metaheuristic algorithm and providing a significantly more robust application paradigm for the field.

Keywords: Cluster Optimization; MultiAO Algorithm; Metaheuristic Algorithm; Multi-Agent System; Unmanned Aerial Vehicle Path Planning

1 Introduction

Unmanned aerial vehicles (UAVs), a class of mechanical devices that are operated and piloted without a human onboard pilot, can be controlled by humans or by computer algorithms through their informational components, such as cameras, sensors, and communication equipment [1]. With the continuous advancement of UAV technology, their application scenarios have gradually expanded from early military uses to a wide range of industries related to disaster monitoring, emergency rescue, environmental perception, logistics transportation, and the low-altitude economy [2][3][4]. Especially in complex terrain and dynamic environments, UAVs can accomplish high-risk or high-density information acquisition tasks at relatively low cost, demonstrating advantages that are difficult for traditional manned platforms to replace. For this reason, the safety, autonomy, and mission execution efficiency of UAV systems have progressively become core issues in research and engineering practice; in particular, path planning, as a key component within the UAV autonomous decision-making and control architecture, directly affects mission completion quality and system reliability, and therefore urgently requires high-performance algorithms as support [5][6].

In essence, the UAV path planning problem belongs to a typical class of constrained optimization problems, whose objective is to seek an optimal or approximately optimal flight path under multiple constraints such as flight safety, energy consumption limitations, and dynamic obstacle avoidance. However, as application scenarios become more complex and environmental dimensionality continues to increase, path planning problems often exhibit highly nonlinear, non-convex, and multimodal characteristics, and may even evolve into NP-hard problems in three-dimensional space [7]. Against this backdrop, traditional deterministic optimization methods have gradually exposed limitations in computational efficiency and global search capability, making it difficult to obtain solutions with engineering feasibility within a limited time [8].

To remedy the deficiencies of deterministic methods, metaheuristic algorithms (Metaheuristic Algorithms) have been widely introduced into the solution process for complex optimization problems. Such algorithms typically employ stochastic search and population-based cooperative mechanisms to efficiently explore the solution space without requiring explicit gradient information [9]. Existing studies indicate that metaheuristic algorithms possess four notable advantages when dealing with complex nonlinear problems: a relatively simple algorithmic structure, strong adaptability and extensibility, no need for rigorous mathematical derivations, and the ability to escape local optima [10]. Accordingly, they have been successfully applied to UAV path planning, engineering design, feature selection, and scheduling optimization, among other domains [11-14].

Existing metaheuristic algorithms can, in broad terms, be divided into several categories, including swarm intelligence algorithms, physics-inspired algorithms, evolutionary algorithms, and human behavior-inspired algorithms [15]. To more clearly compare the differences among various types of metaheuristic algorithms in terms of modeling assumptions and search mechanisms, Table 1 summarizes the design inspirations, core mechanisms, and typical characteristics in UAV path planning of several representative optimization algorithms.

Table 1. Four Types of Metaheuristic Algorithms.

Algorithm Category	Representative Algorithms	Design Inspiration	Search Characteristics	Major Limitations
Swarm Intelligence Algorithms	Particle Swarm Optimization (PSO) [16] Grey Wolf Optimizer (GWO) [17] Whale Optimization Algorithm (WOA) [18] Harris Hawks Optimizer (HHO) [19] Dung Beetle Optimizer (DBO) [20]	Cooperative behavior and predation in biological populations	Strong inter-individual information sharing, exhibiting relatively strong global search capability and simplicity of implementation	Prone to premature convergence, with insufficient adaptability to complex constraints and multi-stage search
Physics-Inspired Algorithms	Rime Optimization Algorithm (RIME) [21] Snow Ablation Optimizer (SAO) [22] Electromagnetic Wave Propagation Algorithm (EWPA) [23] Wave Optics Optimizer (WOO) [24] Kirchhoff's Law Algorithm (KLA) [25]	Physical processes or laws of nature	Facilitates global exploration via probabilistic transitions or energy evolution, with diverse search trajectories	Parameter-sensitive, strongly dependent on physical models, and its generalization capability remains to be validated
Evolutionary Algorithms	Genetic Algorithm (GA) [26] Differential Evolution (DE) [27] Self-adaptive Differential Evolution (SaDE) [28] JADE [29] Alpha Evolution [30]	Biological evolution and natural selection	High population diversity, strong robustness, and suitability for complex nonlinear problems	Requires many iterations, with a relatively slow convergence speed
Human Behavior Algorithms	Traffic Jam Optimizer (TJO) [31] Market Game Optimization Algorithm (MGOA) [32] Basketball Team Optimization Algorithm (BTOA) [33] Tianji Horse Racing Optimization (THRO) [34] Dream Optimization Algorithm (DOA) [35]	Human behavior, games, or social activities	The search strategy is characterized by stage-wise progression and contextual adaptability, and is readily mapped to engineering problems	Theoretical analysis is insufficient, and algorithmic stability and interpretability still require further investigation

First, swarm intelligence algorithms realize the search process by simulating the mechanisms of coordination and information sharing among individuals in nature (such as bird flock foraging, wolf pack encirclement and predation, or whale pod bubble-net attacks), and they exhibit outstanding performance in continuous-space optimization and path-planning problems [16–20]. However, this class of algorithms typically relies on a single biological metaphor; its search strategy lacks sufficient dynamic adjustment capability across different stages, and it is prone to falling into premature convergence in complex environments [17,20]. Physics-inspired algorithms, by contrast, construct search mechanisms on the basis of physical processes or natural laws—such as energy diffusion, phase transition, or wave propagation—and enhance global exploration capability via probabilistic jumps or state evolution [21–25]. Although such methods possess certain advantages in escaping local optima, their performance is often sensitive to parameter settings and to the assumptions of the underlying physical model. Evolutionary algorithms take genetic variation and natural selection as their core; by maintaining population diversity they improve algorithmic robustness and have broad application in multi-constraint, nonlinear optimization problems [26–30], yet their computational overhead is relatively large and their convergence speed is comparatively slow. In recent years, human behavior-inspired heuristic algorithms have gradually emerged; such methods extract decision-making mechanisms from transportation, games, sports competition, or human cognitive activities, and construct search strategies with staged characteristics and contextual adaptability [31–35]. Although these algorithms demonstrate strong flexibility in engineering applications, their theoretical foundations and stability analyses still await further refinement. When an algorithm enters the late stage of convergence or faces high-dimensional space, problems such as search stagnation, imbalance between exploration and exploitation, and insufficient utilization of historical information are prone to occur. These shortcomings are particularly evident in three-dimensional UAV path-planning tasks.

It is worth noting that, according to the No Free Lunch (NFL) theorem, no single optimization algorithm can achieve optimal performance on all problems [32][36]. Therefore, designing novel algorithms with adaptive search mechanisms tailored to specific problem structures still carries important theoretical significance and practical value.

In the UAV domain, metaheuristic algorithms, by virtue of their strong global search and robust optimization capabilities, have become one of the mainstream approaches for solving complex path-planning problems. In recent years, scholars have continuously improved planning performance and convergence efficiency in three-dimensional dynamic environments by refining classical algorithms or devising novel hybrid strategies. For example, Fu et al. (2024) proposed a brand-new Red-billed Blue Magpie Optimizer that simulates the species' foraging and social behaviors, and verified its superior optimization capability and stability in 2D/3D UAV path planning and engineering design problems [8]. Meanwhile, in order to enhance the performance of "old-era" algorithms, Yu et al. (2023) combined the Grey Wolf Optimizer (GWO) with Differential Evolution (DE) and proposed a hybrid algorithm that strengthened local exploitation while maintaining population diversity, thereby significantly improving UAV path quality in complex terrain [37]. Sonny et al. (2023), oriented toward the application scenario of UAV-assisted wireless networks, improved the Particle Swarm Optimization (PSO) algorithm by dynamically adjusting the inertia weight and learning factors, achieving efficient trajectory generation under communication-coverage constraints [38]. Liu et al. (2021), on the basis of the Sparrow Search Algorithm (SSA), introduced adaptive weights and a mutation mechanism, effectively overcoming the original algorithm's tendency toward premature convergence and successfully applying it to three-dimensional route planning [39]. In addition, Zheng et al. (2025) proposed an improved Dung Beetle Optimization algorithm; by introducing a cooperative search mechanism and a dynamic boundary-handling strategy, it enabled cooperative path planning for multiple UAVs and exhibited strong performance in obstacle avoidance and formation maintenance [40]. Gu et al. (2024) further designed a multi-strategy enhanced Marine Predators Algorithm, integrating Lévy flights, opposition-based learning, and chaotic local search, among other strategies, and achieved better solution quality and convergence speed in global optimization and UAV swarm path planning [41]. These studies indicate that, through structural refinements, hybrid strategies, or innovations in bio-inspired mechanisms, metaheuristic algorithms can effectively address the high-dimensional, dynamic, and multi-constraint challenges in UAV path planning.

Motivated by the above considerations, this paper proposes a novel multi-agent optimization algorithm—Multi-Agent Optimizer (MultiAO)—and applies it to the three-dimensional UAV path-planning problem. The design inspiration of MultiAO is drawn from the "State-Dependent Dynamic Evolution" mechanism in complex systems, rather than from a single biological-behavior metaphor. By characterizing the system-state changes corresponding to agents across different search stages, the algorithm introduces three interrelated evolutionary phases: gravitational accretion and centroid rebellion, orthogonal singularity decoupling, and hysteresis effects with a memory potential field. These mechanisms respectively correspond to population convergence, adaptive fine-grained search, and an exploration–recall balance guided by historical information in the search process, enabling the algorithm to dynamically switch between global exploration and local exploitation.

In three-dimensional UAV path-planning tasks, MultiAO can effectively address challenges such as high-dimensional search spaces, complex constraints, and multi-objective conflicts. By introducing a population-centroid reference, component-wise search strategies, and a memory potential-field mechanism, the algorithm significantly improves solution diversity and stability while maintaining convergence efficiency.

The main contributions of this paper can be summarized as follows:

- Propose a multi-agent optimization algorithm, MultiAO, based on a state-dependent dynamic evolution mechanism;
- Construct a unified optimization framework integrating population accretion, orthogonal decoupling, and hysteretic memory to enhance search robustness;
- Apply MultiAO to the three-dimensional UAV path-planning problem and verify its effectiveness and feasibility in complex environments.

The remainder of this paper is organized as follows: Section 2 introduces in detail the design rationale and mathematical model of MultiAO; Section 3 introduces in detail the mathematical model of MultiAO; Section 4 presents the three-dimensional UAV path-planning model and experimental settings, and conducts a comparative analysis of algorithmic performance; Section 5 summarizes the paper and outlines future research directions.

2 Inspiration from Collaborative Intelligence in Multi-Agent Systems

In the fields of statistical physics and social dynamics, MAS, by virtue of its inherent learning and autonomous decision-making capabilities, exhibits greater flexibility: agents learn new contexts and actions through interactions with neighboring agents or with the environment. The design inspiration of MultiAO originates precisely from the State-Dependent Dynamic Evolution mechanism in complex systems. In an MAS setting, if environmental changes are induced solely by agent behaviors, the environment is static; if the environment undergoes spontaneous changes independent of agents' actions, it is dynamic. In a dynamic environment, an agent's decision-making process depends on information perceived from the environment, and meanwhile the agent must detect changes in the environmental state and update the perceived information [42][43]. Inspired by this, MultiAO no longer assumes that the solution space is stationary; instead, it simulates optimization by agents within a potentially dynamically changing environment, and describes it through three physical phase-transition stages as follows:

- Gravitational accretion and centroid rebellion

In a dynamic environment, the current optimal solution may become invalid at any time. Therefore, the early-stage evolution of the algorithm simulates a social force field. Individuals with low fitness tend toward gravitational accretion, namely, conforming to the current elite individuals to reduce survival risk, analogous to matter falling into a gravitational potential well. However, in order to cope with independent changes that the environment may undergo, high-fitness leader individuals must possess a centroid rebellion mechanism: they do not blindly follow a single leader, because that leader's advantage may vanish instantaneously due to environmental variation; instead, they take the geometric centroid of the population as a reference and initiate rebellious exploration

that deviates from the mainstream. This mechanism simulates the additional overhead that agents in dynamic systems must incur to update perceived information, thereby broadening the search boundary.

- Orthogonal singularity decoupling

When MAS approaches a convergent state, complex coupling among multidimensional variables can lead to lag in information perception. Physics indicates that, under high-pressure critical states, the degrees of freedom of a system undergo decoupling. MultiAO simulates this process: when macroscopic population motion becomes inefficient due to the curse of dimensionality, the algorithm switches from full-vector motion to component-based orthogonal fine-tuning motion. By adjusting only one dimension at a time on an orthogonal basis, agents can precisely capture tiny fluctuations in the environmental state; this strategy is analogous to a dimension-reduction strike near a singularity, achieving high search precision.

- Hysteresis effects and a memory potential field

The state evolution of MAS is hysteretic, meaning that current decisions depend not only on real-time perception but are also constrained by historical trajectories. In MultiAO, past optimal states do not dissipate in space, but instead form a persistent memory potential field. On the one hand, this potential field exerts a repulsive force on agents that get too close to already explored regions, preventing them from wasting computational resources on information that may already be outdated. On the other hand, it provides a recall path for agents that become lost in high-entropy regions. This mechanism effectively addresses the high overhead of information updating in dynamic environments, leveraging historical memory to assist real-time decision-making.

3 Mathematical Model of the MultiAO

This section elaborates the mathematical construction of MultiAO. The algorithm regards the optimization process as a system in which N agents evolve over time in a D-dimensional space. The entire optimization process will be viewed as the system-state evolution trajectory that unfolds over time as these agents, driven by interaction forces and environmental feedback, undergo state changes with the passage of time.

3.1 Adaptive Parameter Evolution

From the perspective of complex evolution in MAS, fixed control parameters run counter to the central claim in the No Free Lunch theorem regarding strategic diversity. MultiAO regards the adaptive adjustment of parameters as an evolutionary process of agents in a dynamic environment, assuming that the optimal control strategy is not a static value, but rather a probability distribution that shifts as the system evolves. We introduce into the algorithm a memory mechanism akin to Bayesian inference, letting the scaling factor F represent the kinetic energy of jumps; a fixed F would constrain the diversity of the search. To emulate the Lévy-flight effect in physics, we model F using a Cauchy distribution rather than a Gaussian distribution. The probability density function of the Cauchy distribution is defined as follows:

$$f(x;x_0,y)=\frac{1}{\pi\gamma\left[1+\left(\frac{x-x_0}{\gamma}\right)^2\right]}, \tag{1}$$

This heavy-tailed property allows an agent, with low probability, to obtain an extremely large instantaneous kinetic energy, thereby achieving physical escape when it is trapped at a gravitational singularity corresponding to a local optimum. By contrast, the crossover probability CR represents the frequency of information exchange and tends to fluctuate around the population's historical experiential mean; accordingly, adopting a Gaussian distribution is more consistent with the steady-state characteristics of social dynamics.

On this basis, the algorithm introduces a memory mechanism akin to Bayesian inference, storing historically successful evolutionary trajectories via a memory archive M. Therefore, the parameter derivation for the i-th agent at the t-th iteration is given as follows:

$$\begin{cases} F_i = C(\mu_{F,M}, 0.1) \\ CR_i = N(\mu_{CR,M}, 0.1) \end{cases}, \tag{2}$$

where, $C(\cdot)$ and $N(\cdot)$ respectively denoting the Cauchy and Gaussian (Normal) probability distributions. $\mu_{F,M}$ and $\mu_{CR,M}$ denote the mean values of F and CR in the memory archive, respectively, and 0.1 represents the scale parameter of the distributions. If a generated $F_i \le 0$, it is considered a failure of kinetic energy and triggers a resampling mechanism until effective actuation is achieved. Conversely, if CR_i falls outside the valid range [0, 1], it is physically truncated to a saturated state (i.e., clamped to the nearest bound).

3.2 Phase A: Gravitational Accretion and Centroid Rebellion

The core innovation of MultiAO lies in introducing state-dependent update rules that conform to the characteristics of complex systems. In the evolutionary process under nonequilibrium conditions, an agent's behavioral mode depends on its relative state within the population. To quantify this state, we define the relative potential-energy level of agent *i*, using the normalized fitness ratio ΔE_i as follows:

$$\Delta E_i = \frac{Fit_i(t) - Fit_{best}(t)}{\left|Fit_{avg}(t)\right| + \varepsilon}, \tag{3}$$

where, $Fit_i(t)$ is the fitness of the *i*-th agent at the *t*-th iteration.

When ΔE_i is high, the agent is located in a high-potential-energy region far from the optimal solution, confronting severe information scarcity. In a social force field, the rational strategy at this time is conformity, namely, leveraging the gravitational potential well generated by the elites to accelerate settling. To this end, we construct a vector field interwoven by two fundamental forces: the elite attractive force $\vec{v}_{att}$ of the current best solution x_{best}, and the random social differential perturbation force $\vec{v}_{diff}$, as given by the following formula:

$$\vec{v}_{att} = x_{best} - x_i(t), \tag{4}$$

$$\vec{v}_{diff} = x_{r_1} - x_{r_2}, \tag{5}$$

where, $x_i(t)$ is the current position of the agent; x_{r_1} and x_{r_2} are two distinct individuals randomly selected from the population; and $r_1, r_2 \neq i$. By combining these two forces, the agent's motion follows the directed-acceleration formula given below:

$$v_i(t) = x_i(t) + F_i(x_{best} - x_i(t)) + F_i(x_{r_1} - x_{r_2}), \tag{6}$$

where, $v_i(t)$ is the new position that the *i*-th agent plans to move to. This formula ensures that, while maintaining the necessary random oscillations, the agent spirals toward the elite individuals.

When ΔE_i is low, however, the agent has already become an elite within the population. At this point, merely following x_{best} would lead to stagnation. To break through the singularity of a local optimum, these leader individuals must initiate rebellion. Rather than blindly adhering to a single leader, they take the population's geometric centroid as a more robust macroscopic frame of reference, and define the centroid c_t as the geometric center of the top 10% elite agents:

$$c_t = \frac{1}{N_{top}} \sum_{k=1}^{N_{top}} x_{sorted,k}(t), N_{top} = \left[0.1N\right], \tag{7}$$

where, N_{top} denotes the set of elite agents, and $x_{sorted,k}(t)$ denotes the position vector of the *k*-th agent after all agents have been ranked by fitness at the *t*-th iteration. For elite agents, we impose an overshoot coefficient $F_{boost} = 1 + rand$, and *rand* is a random number between 0 and 1. The physical rationale is that, in order to surmount the current fitness "mountain" and seek a higher "crest," an agent must possess the momentum to leap over the average "plain" (the centroid). This process can be expressed as:

$$v_i(t) = c_t + F_{boost}(x_{r_1} - x_{r_2})\vec{M}_{sparse}, \tag{8}$$

where, $\vec{M}_{sparse}$ is a binary sparse matrix, ensuring that not all dimensions are perturbed simultaneously, thereby preserving the structural stability of elite solutions. The ultimately generated mutant vector $v_i(t)$ will be blended with the original position $x_i(t)$ via binomial crossover or exponential crossover, producing the trial vector x_{new}.

Phase A can be illustrated by Figure 1.

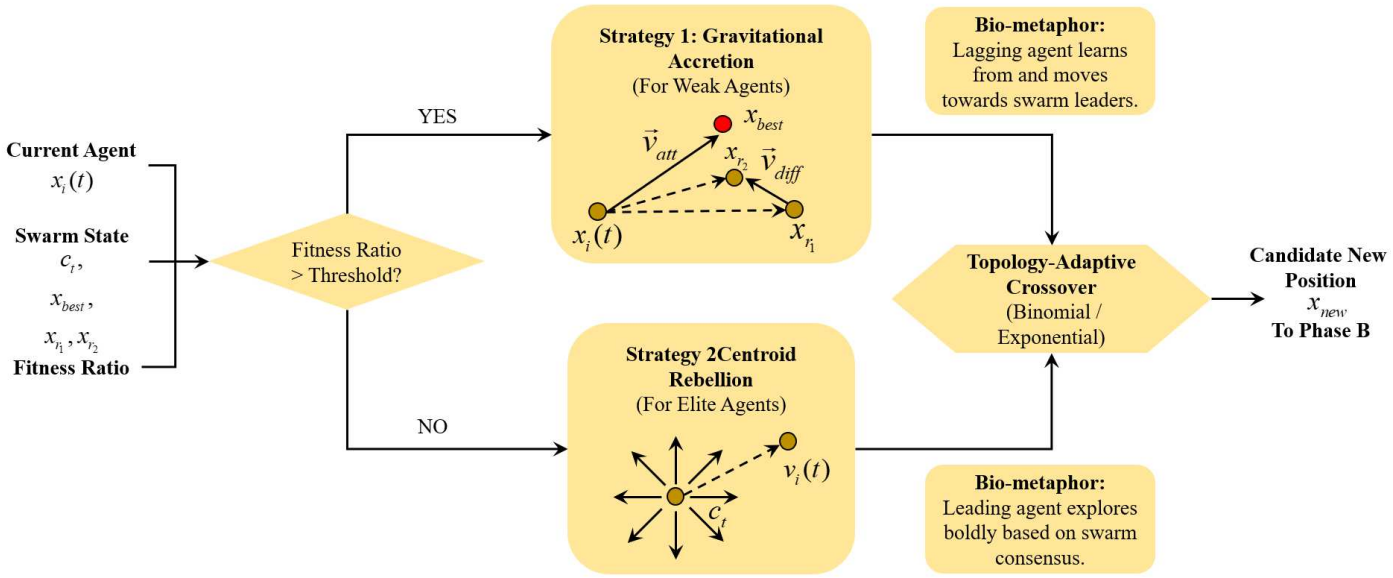


Figure 1. Phase A Gravitational Accretion and Centroid Rebellion.

3.3 Phase B: Orthogonal Singularity Decoupling

When the evolution proceeds into the middle-to-late stage, the population has completed macroscopic gravitational accretion, and the population tends toward convergence. In the high-dimensional space $D > 50$, attempting to obtain a better solution by perturbing all *D* dimensions simultaneously yields a joint probability *P* that decays exponentially with the dimensionality, which can be expressed as:

$$P(Fit_{new} < Fit_{best}) \approx 0.5^D \to 0. \tag{9}$$

where, 0.5^D indicates that, for a single dimension, the probability of obtaining a better solution is 50%. This implies that, within the deep valleys of complex terrain, any slight random full-vector movement has an extremely high probability of causing fitness degradation. To overcome this obstacle, MultiAO simulates the degree-of-freedom decoupling mechanism in physical systems. At this point, the algorithm freezes most degrees of freedom and applies an orthogonal decoupling strategy only to the current global best solution x_{best}. By reducing a complex multidimensional coupling problem into a sequence of independent one-dimensional line searches, we define the orthogonal basis vector $\vec{e}_d = [0,\cdots,1,\cdots,0]$. The step size $\lambda(t)$ decays quadratically over time as:

$$\lambda(t) = 0.05(UB - LB)\left(1 - \frac{t}{T}\right)^2, \tag{10}$$

where, UB and LB are the upper and lower bounds, and 0.05 is the scaling coefficient. Therefore, the position update formula for the orthogonal fine-tuning motion is:

$$x_{new} = x_{best} \pm \lambda(t)\vec{e}_d, \tag{11}$$

The algorithm introduces, via the symbol $\pm$, a symmetric mechanism of forward probing and backward probing. This is equivalent to bidirectional sensing along the ridge of a potential-energy surface, ensuring the algorithm's stability near the global optimum.

Phase B can be illustrated by Figure 2.

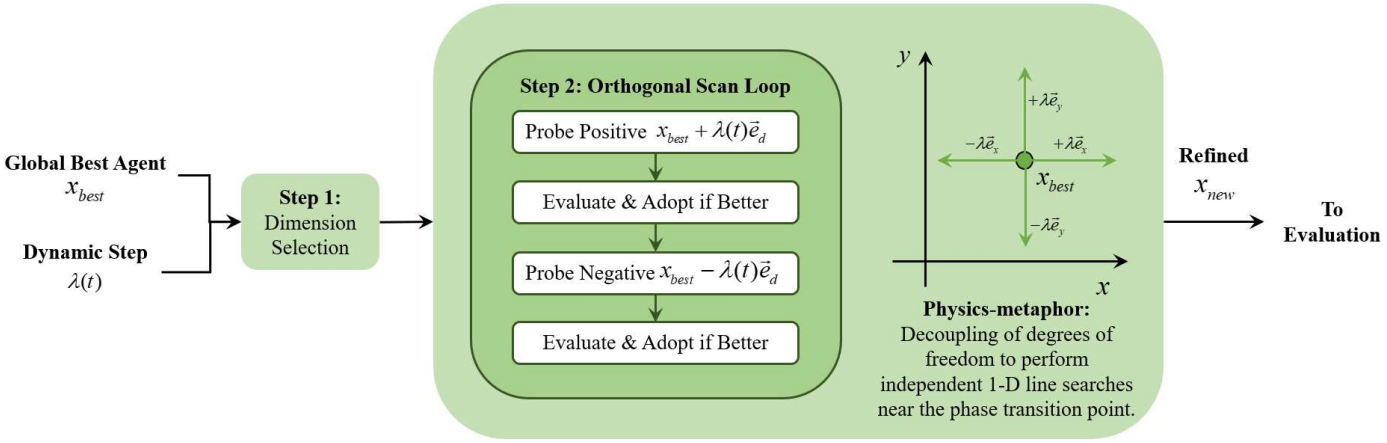


Figure 2. Phase B Orthogonal Singularity Decoupling.

3.4 Phase C: Hysteresis and Memory Potential Field

Physical systems often exhibit hysteresis in their evolution, meaning that a system's current state depends not only on the input but also on its historical trajectory. MultiAO incorporates this property into the optimization process by maintaining a holographic memory archive A that contains historical elite solutions, which we model as a potential field. According to the Euclidean distance d_{ij} between agent x_i and a historical memory point $x_{archive,j}$, two different types of interactions are triggered.

First, when d_{ij} is smaller than the repulsion threshold r_{th}, it indicates that the agent has mistakenly entered a low–information-gain region that has already been explored. To avoid redundant computational expenditure, the memory potential field activates a short-range repulsion mechanism. We introduce a normalized repulsive vector $\vec{R}$ to indicate the escape direction, and the repulsive-vector formula is as follows:

$$\vec{R} = \frac{x_i - x_{archive,j}}{d_{ij} + \varepsilon}, \tag{12}$$

Introducing the tangent term $\left|\tan(\square)\right|$ provides an infinite-impulse potential energy analogous to a Cauchy distribution, ensuring that the agent is forcibly ejected from redundant regions; introducing the sinusoidal chaotic perturbation term $\sin(AF \cdot x_i)$ prevents the ejection trajectory from becoming singular, where $AF = \sqrt{t/T}$. The integrated update formula is as follows:

$$x_{new} = x_i + \vec{R}\left|\tan(\pi(rand - 0.5))\right| 5r_{th} + \sin(AF \cdot x_i)0.005(UB - LB), \tag{13}$$

Second, for "lost" agents that have not been updated for a long time and have fallen into a local optimum, a purely random walk has already become inefficient. At this point, the algorithm activates a bridging recall mechanism, attempting to establish a bridge from a historical memory point $x_{archive,j}$ to the current global optimum x_{best}. The algorithm constructs in parallel two competing candidates, x_A and x_B: x_A adopts a vector-guidance strategy, establishing a linear channel pointing from the historical memory point toward the current optimal solution; x_B adopts a scalar-diffusion strategy, performing isotropic Gaussian reconstruction and chaotic oscillation centered at the position of the historical memory point. The two candidates can be expressed as:

$$x_A = x_{archive,j} + \sigma_{vec}(x_{archive,j} - x_i) + \sigma_{guide}(x_{best} - x_{archive,j}), \tag{14}$$

$$x_B = x_{archive,j} + \sigma_{scal}\mu(-0.5, 0.5) + \sin(AF \cdot x_{archive,j}), \tag{15}$$

where, σ_{vec} is a random scaling coefficient, σ_{guide} is a guidance-strength coefficient, σ_{scal} is a scalar diffusion coefficient, and $\mu(-0.5,0.5)$ is a uniform random distribution. Finally, the algorithm performs a forced update by competitively selecting the better position between x_A and x_B .

Phase C can be illustrated by Figure 3.

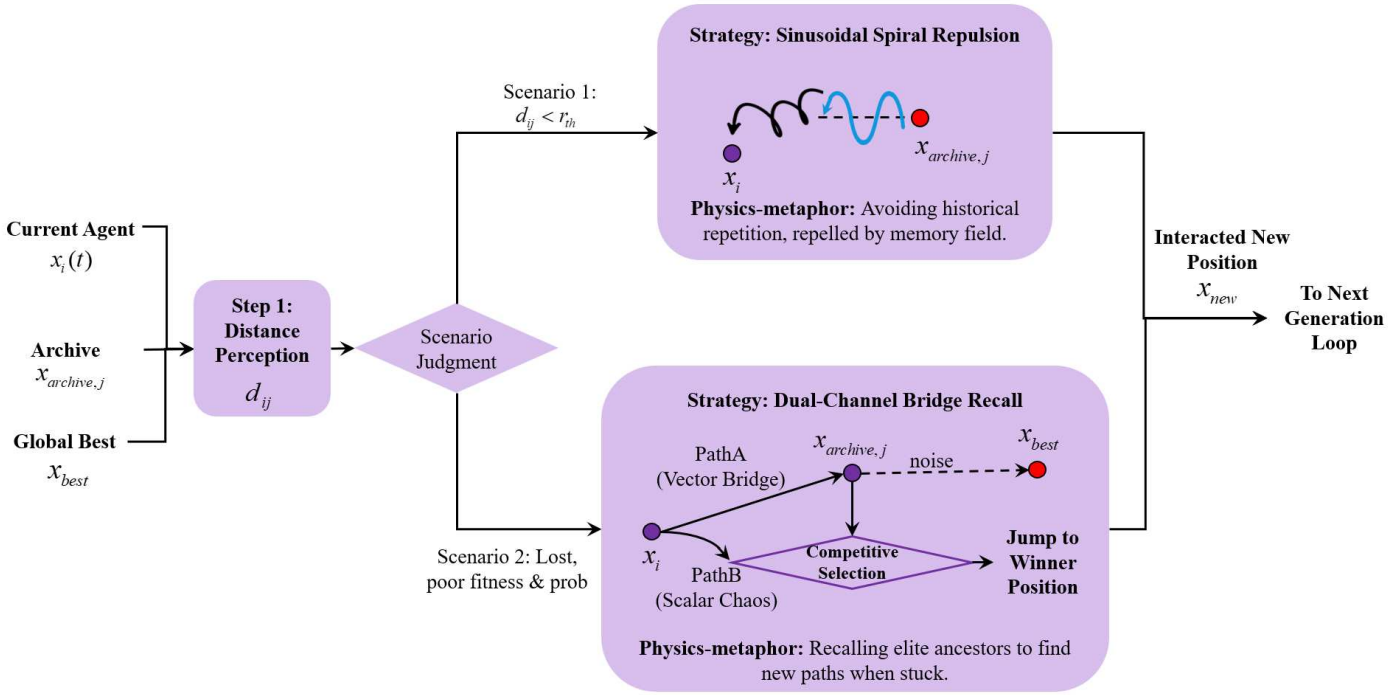


Figure 3. Phase C Hysteresis and Memory Potential Field.

3.5 Entropy Injection and Singularity Restart

To ensure the thermodynamic vitality of the system, MultiAO monitors in real time the population's radius of gyration $R_{pop} = \|std(X)\|$. When R_{pop} falls below the critical threshold and the global best solution has not yet reached the theoretical convergence precision, the system is judged to have entered a metastable state, namely, it is trapped in a spurious local-minimum potential well. At this point, microscopic perturbations are no longer sufficient for escape, and a singularity restart must be triggered. Leveraging the pole-divergence property of the tangent function, the algorithm injects instantaneous high-energy entropy pulses into these individuals, forcibly breaking the population's spatial symmetry. The definition of the restart position $x_{i,restart}$ is as follows:

$$x_{i,restart} = x_{best} + \tan\left(\pi\left(rand - 0.5\right)\right)R_{pop} \cdot 100, \tag{16}$$

By exploiting the heavy-tailed property of the Cauchy distribution, high-energy pulses are injected into non-elite individuals, forcibly breaking symmetry.

3.6 Pseudocode of the MultiAO Algorithm

The pseudocode of the algorithm is shown in Table 2, covering the integration of all mechanisms.

Table 2. Pseudo-code of MultiAO Algorithm.

Algorithm 1 MultiAO
Start SMNBKA-ICMIC
1.INPUT
Population size *N*, Max iterations *T*, Bounds [*LB*, *UB*], Dim *D*
2.OUTPUT
Global Best Position *x_best*
3.LOOP START
Initialize Population *x* randomly
Initialize Memory banks (*M_F*, *M_CR*) and Archive *A*

```
Calculate Fitness and Sort Population; Identify x_best
While t < T do
    Calculate Population Radius R_pop
    If R_pop < 1e-6 (Metastable State) then
        Trigger Singularity Restart (Entropy Injection) (Eq. 16)
    End If
    Update Population Size N_curr linearly

    --- Phase A: Gravitational Accretion and Centroid Rebellion ---
    Calculate Centroid C of top 10% agents (Eq. 7)
    For i = 1 to N_curr do
        Sample F and CR from Memory (Eq. 2)
        Calculate Fitness_Ratio (Eq. 3)
        If Fitness_Ratio > Threshold (Weak Agent) then
            v = Gravitational Accretion (Eq. 6)
        Else (Elite Agent) then
            v= Centroid Overshoot Adversarial (Eq. 8)
        End If
        Perform Crossover to generate x_new
        Selection: If Fit(x_new) < Fit(x_i), update x_i and Archive A
    End For

    --- Phase B: Orthogonal Singularity Decoupling ---
    Calculate Decay Step_Size lambda(t) (Eq. 10)
    Select active dimensions randomly (20%)
    For each dimension d in active_dims do
        Probe Positive and Negative directions on x_best (Eq. 11)
        If improvement found, Update x_best immediately
    End For

    --- Archive Maintenance ---
    Sort Archive A and keep top N agents

    --- Phase C: Hysteresis and Memory Potential Field ---
    For i = 1 to N_curr do
        Calculate distance to nearest Archive member
        If distance < Radius_Threshold then
            Apply Repulsive Force + Chaos (Eq. 13)
        Else If Agent is Lost (Low Fitness) then
            Generate Candidate A (Bridge) and Candidate B (Chaos) (Eq. 14 and Eq. 15)
            Select the better Candidate to replace x_i
        End If
    End For

    Update Memory M based on successful parameters
    t = t + 1
End While
End MultiAO
```

Through the above mathematical model, MultiAO successfully combines macroscopic gravitational dynamics with microscopic quantum randomness, achieving an adaptive balance between exploration and exploitation. The computational complexity is approximately $O(T \cdot \bar{N} \cdot D)$, comparable to that of traditional algorithms, but with a stronger capability to escape local optima.

4 Experiment and Analysis

To rigorously demonstrate the effectiveness of the proposed MultiAO in solving complex optimization tasks, we conduct a multi-dimensional performance comparison between MultiAO and ten mature metaheuristic algorithms on CEC2022. In addition, we apply MultiAO to the three-dimensional UAV path-planning problem to further verify its effectiveness and feasibility in complex environments.

4.1 Experimental Setup

To ensure fairness and reproducibility, all experiments in this paper are conducted under a unified environment; the experimental configuration is shown in Table 3.

Table 3. Experimental Setup.

Configuration	Description
CPU Model	Intel i7-11800H 16 GB
Operating System	Windows 11
GPU Model	NVIDIA RTX-3060 12 GB
Experimental Software	MATLAB 2024b

4.2 Comparative Results on CEC2022

4.2.1 Function Suite and Algorithm Parameters

The CEC2022 benchmark test suite[44] is specifically designed to rigorously examine an algorithm's global exploration capability in high-dimensional search spaces, its ability to handle complex constraints, its capability to traverse narrow feasible regions, and its competence in escaping local optima, thereby providing the latest high-standard benchmark for assessing the overall performance and scalability of modern optimization algorithms in solving large-scale engineering optimization problems. Table 4 presents the complete set of CEC2022 suites.

Table 4. CEC2022 Test Table.

	No.	Functions	F_i^*
Unimodal function	1	Shifted and full Rotated Zakharov Function	300
Basic functions	2	Shifted and full Rotated Rosenbrock's Function	400
	3	Shifted and full Rotated Expanded Schaffer's f_6 Function	600
	4	Shifted and full Rotated Non-Continuous Rastrigin's Function	800
	5	Shifted and full Rotated Levy Function	900
Hybrid functions	6	Hybrid Function 1 (N=3)	1800
	7	Hybrid Function 2 (N=6)	2000
	8	Hybrid Function 3 (N=5)	2200
Composition functions	9	Composition Function 1 (N=5)	2300
	10	Composition Function 2 (N=4)	2400
	11	Composition Function 3 (N=5)	2600
	12	Composition Function 4 (N=6)	2700
Search range: $[-100,100]^D$			

This study compares MultiAO with ten algorithms, including five highly cited algorithms: Particle Swarm Optimization (PSO)[45], Grey Wolf Optimizer (GWO)[46], Whale Optimization Algorithm (WOA)[47], Harris Hawks Optimization (HHO)[48], and Dung Beetle Optimizer (DBO)[49], as well as five mature algorithms released in 2025: Artificial Lemming Algorithm (ALA)[50], Hannibal Barca Optimizer (HBO)[51], StarFish Optimization Algorithm (SFOA)[52], Tornado Optimizer with Coriolis Force (TOC)[53], and Whale Migration Algorithm (WMA)[54], thereby validating the efficiency and state-of-the-art nature of the algorithm.

To ensure reproducibility, Table 5 presents the initial parameter configurations of these ten algorithms.

Table 5. Initial Algorithm Parameters.

Algorithms	Parameter
MultiAO	All parameters are either random numbers in the range of 0–1 or vary as a function of the number of iterations.
PSO	$V_{Max}=6, w_{Max}=0.9, w_{Min}=0.6, c_1=2, c_2=2$
GWO	All parameters are either random numbers in the range of 0–1 or vary as a function of the number of iterations.
WOA	$b=1$
HHO	$\beta=1.5$
DBO	$p_{percent}=0.2$
ALA	All parameters are either random numbers in the range of 0–1 or vary as a function of the number of iterations.
HBO	$Romanrate=0.63, c=0.94$
SFOA	$C=0.8, T=0.5$
TOC	$nto=4, nt=3, b_r=100000, chi=4.1$
WMA	All parameters are either random numbers in the range of 0–1 or vary as a function of the number of iterations.

To maintain fairness in the performance comparison, all algorithms are uniformly configured with 2,000 iterations and a population size of 50, and each algorithm is executed in 50 independent runs to record the resulting metrics.

4.2.2 Ten-Dimensional Results on CEC2022

This study first compares MultiAO with the other ten algorithms on CEC2022 (Dim = 10); the results are shown in Figure 4 and Table 6.

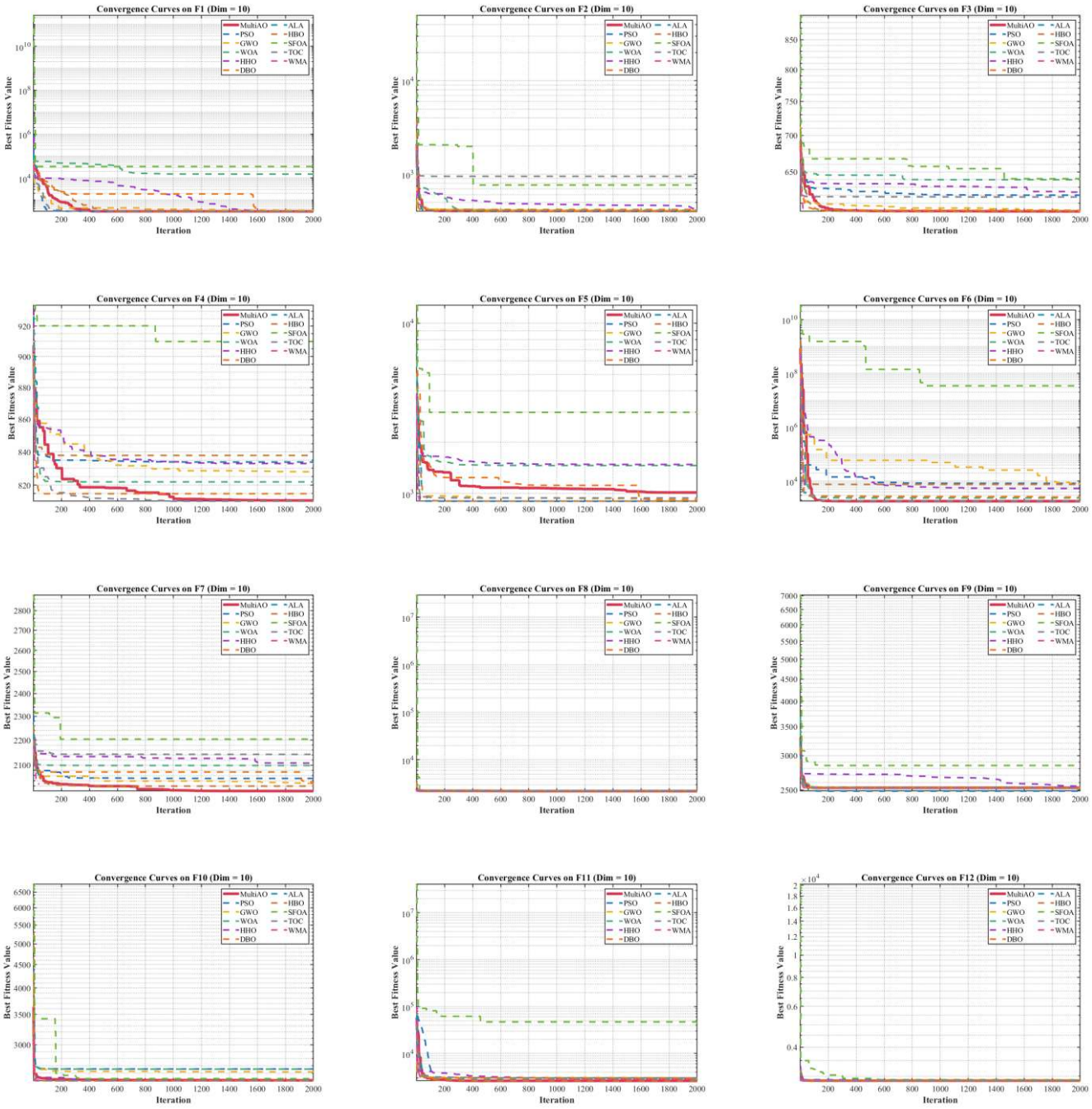


Figure 4. Convergence Curves of Various Algorithms on CEC2022 (Dim = 10).

Table 6. Mean Results of Each Algorithm on CEC2022 (Dim = 10).

Function	Mertri	MultiAO	PSO	GWO	WOA	HHO	DBO	ALA	HBO	SFOA	TOC	WMA
F1	Mean	309.7693	**300**	791.9678	9746.065	300.6731	**300**	10077.86	**300**	41569.64	2117.829	5426.352
	Std	28.59431	1.11E-08	1251.37	4327.986	0.245349	3.36E-06	6107.132	4.67E-05	25811.8	3709.695	1729.52
F2	Mean	**405.0099**	414.9168	415.461	418.8809	411.7172	407.3027	449.6339	417.1903	1710.851	416.1452	408.4641
	Std	3.089273	29.11146	13.85627	28.27018	28.23197	3.048358	39.89449	30.27583	1040.782	16.70794	1.719251
F3	Mean	**600.0111**	601.9542	600.2987	641.712	625.8058	604.9513	611.7536	600.0208	663.1389	619.3649	601.5414
	Std	0.015677	3.022338	0.371478	12.70288	13.10142	6.060809	4.715523	0.06577	23.56663	12.48828	1.0501
F4	Mean	**811.6382**	820.3966	812.8173	834.8858	825.2473	831.096	845.0771	824.1276	899.179	823.3519	844.8221
	Std	4.533288	9.453536	3.248565	9.800491	7.351676	10.25357	10.72117	10.19847	22.83338	11.73583	3.043462
F5	Mean	916.632	**900.0269**	908.8624	1201.211	1467.699	924.3977	1368.131	919.5789	3811.21	978.5477	900.6668
	Std	33.39205	0.043246	11.94965	146.5539	120.4168	35.12347	515.5782	46.53779	1666.019	135.8926	0.525261
F6	Mean	**1813.176**	3468.642	6674.422	4147.59	2718.641	5068.826	6052.774	4037.725	3.66E+08	2535.545	5816.643
	Std	18.47385	1785.806	2410.363	2422.322	1029.306	2844.195	1607.27	2576.901	3.71E+08	2006.01	2943.793
F7	Mean	**2008.141**	2029.929	2025.18	2055.731	2041.729	2030.139	2058.95	2018.668	2189.617	2052.048	2055.352
	Std	7.663163	12.44817	4.214017	19.65314	16.60254	13.30968	17.00327	6.360637	55.15169	21.44445	9.732063
F8	Mean	**2211.754**	2218.392	2223.352	2230.976	2238.844	2225.567	2232.701	2221.547	2316.784	2225.918	2233.226
	Std	7.951245	6.16669	2.519135	3.334503	14.88621	5.412536	6.348287	1.604629	101.226	4.287069	3.965312
F9	Mean	2529.284	**2485.502**	2561.118	2536.695	2550.581	2535.906	2537.948	2529.284	2772.913	2537.279	2544.082
	Std	3.85E-06	1.61E-07	39.27171	21.05951	44.88273	19.80066	14.18193	5.25E-13	107.6403	18.66326	46.44589
F10	Mean	**2492.716**	2632.755	2554.85	2540.931	2538.232	2527.183	2531.654	2571.503	3050.296	2750.846	2538.486
	Std	26.56265	176.232	57.50228	64.35325	60.17922	56.07292	64.36315	61.36552	730.2482	396.3413	61.51246
F11	Mean	2810.199	3006.953	2923.289	2867.262	2758.331	2795.137	3890.472	**2751.749**	51341.36	3036.01	2926.871
	Std	144.7637	453.5314	148.2669	106.9991	147.1138	123.4355	264.6611	160.0424	15300.19	99.1124	83.84291
F12	Mean	**2862.288**	2880.685	2863.91	2891.847	2897.17	2865.443	2864.392	2863.993	2884.679	2881.332	2865.744
	Std	1.653067	49.37254	1.096461	27.7743	36.46005	2.423264	0.954265	1.61615	10.17638	28.23636	1.861378

Figure 4 presents the convergence curves under the CEC2022 benchmark tests at 10 dimensions; the dark-red solid line denotes MultiAO. It can be observed that, except for F1, F4, and F5, MultiAO exhibits rapid convergence already in the early stage and is markedly superior to the other algorithms in the late-stage exploitation phase. In particular, F3, F6, and F7 show pronounced differences, highlighting its competitiveness among existing state-of-the-art methods. According to the results in Table 6, MultiAO achieves mean values superior to those of the other algorithms on functions F2, F3, F4, F6, F7, F8, F10, and F12; its outstanding convergence accuracy, robustness, and reliability fully validate its balanced capability in global exploration and local exploitation.

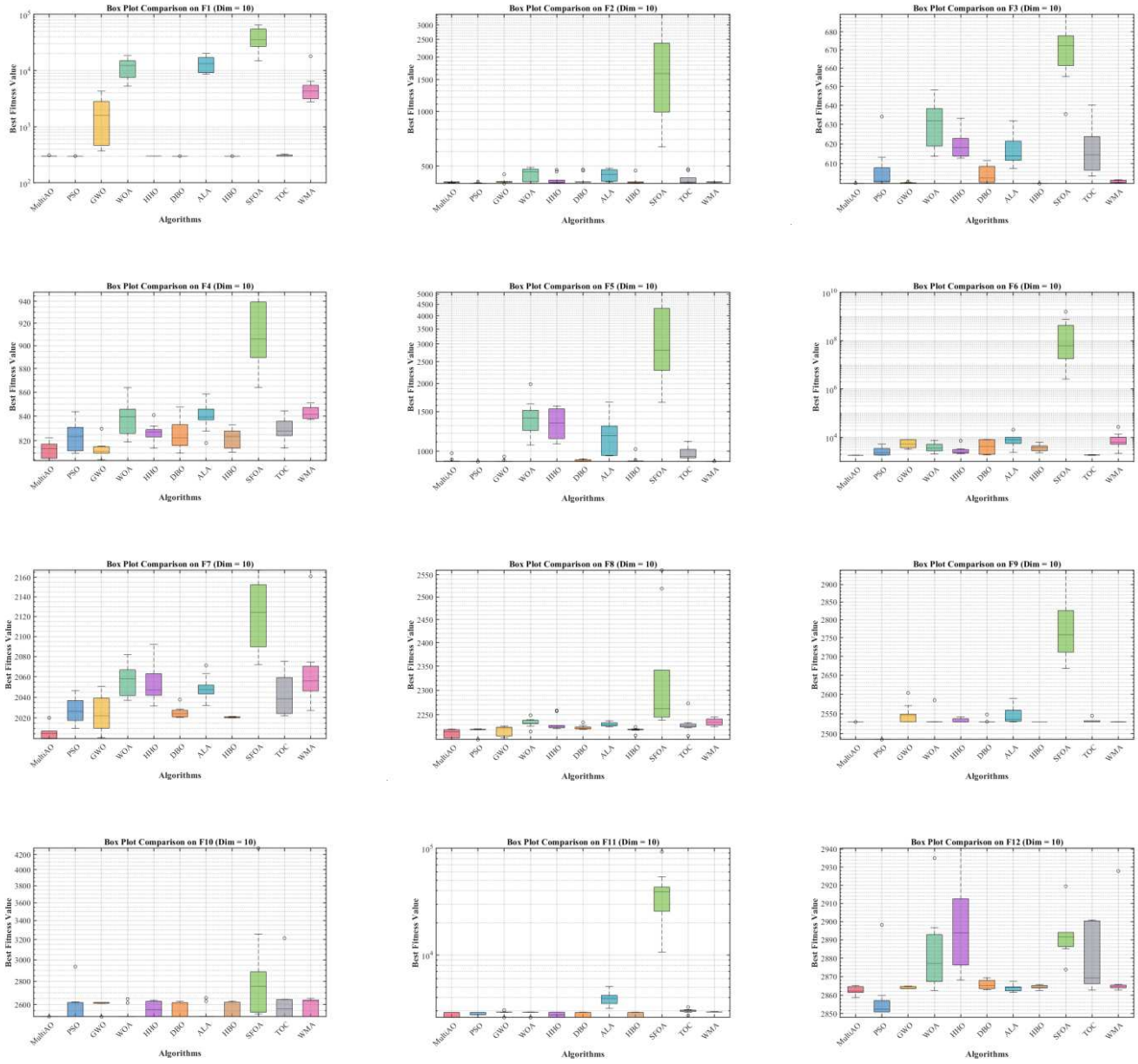


Figure 5. Boxplots of Various Algorithms on CEC2022 (Dim = 10).

As shown in Figure 5, MultiAO converges to the optimal value on F1, F2, F3, F5, F6, F7, F10, and F11; for the vast majority of functions, the fluctuation variance is extremely small, and it consistently maintains a low mean value throughout the entire benchmark test, indicating strong global optimization capability.

Table 7. Wilcoxon Test of Various Algorithms on CEC2022 (Dim = 10).

		vs PSO	vs GWO	vs WOA	vs HHO	vs DBO	vs ALA	vs HBO	vs SFOA	vs TOC	vs WMA
F1	p value	0.000246128	0.000768539	0.000182672	0.140465048	0.00042026	0.000182672	0.005795359	0.000182672	0.021133928	0.000182672
	sign	-	+	+	=	-	+	-	+	+	+
F2	p value	0.241321593	0.00058284	0.10410989	0.733729996	0.103589204	0.000182672	0.79133678	0.000182672	0.04515457	0.002827272
	sign	=	+	=	=	=	+	=	+	+	+
F3	p value	0.011329697	0.001007976	0.000182672	0.000182672	0.001007976	0.000182672	0.002827272	0.000182672	0.000182672	0.000182672
	sign	+	+	+	+	+	+	+	+	+	+
F4	p value	0.031209013	0.427355314	0.000246128	0.000768539	0.000182672	0.000182672	0.009108496	0.000182672	0.014019277	0.000182672
	sign	+	=	+	+	+	+	+	+	+	+
F5	p value	0.009108496	0.307489457	0.000329839	0.000182672	0.185876732	0.000439639	0.212293836	0.000182672	0.014019277	0.677584958
	sign	-	=	+	+	=	+	=	+	+	=
F6	p value	0.00058284	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.00058284	0.000182672	0.001314945	0.000182672
	sign	+	+	+	+	+	+	+	+	+	+
F7	p value	0.001007976	0.000439639	0.000182672	0.000182672	0.000246128	0.000182672	0.009108496	0.000182672	0.000182672	0.000182672
	sign	+	+	+	+	+	+	+	+	+	+
F8	p value	0.185876732	0.000439639	0.000182672	0.000182672	0.000246128	0.000439639	0.003610514	0.000182672	0.000439639	0.000182672
	sign	=	+	+	+	+	+	+	+	+	+
F9	p value	0.000149393	0.000149393	0.000149393	0.000149393	0.001730313	0.000149393	0.35167852	0.000149393	0.171706896	0.000149393
	sign	-	+	+	+	+	+	=	+	=	+
F10	p value	0.075661572	0.344704222	0.014019277	0.031209013	0.10410989	0.021133928	0.025748081	0.000182672	0.003610514	0.031209013
	sign	=	=	+	+	=	+	+	+	+	+
F11	p value	0.570750388	0.001706249	0.007284557	0.570750388	0.053361778	0.000182672	0.064022101	0.000182672	0.000182672	0.001314945
	sign	=	+	+	=	=	+	=	+	+	+
F12	p value	0.472675594	0.021133928	0.000329839	0.000182672	0.003610514	0.005795359	0.053902557	0.000177611	0.001706249	0.00058284
	sign	=	+	+	+	+	+	=	+	+	+
Total (+/=/-)		+:4, =:5, -:3	+:9, =:3, -:0	+:11, =:1, -:0	+:9, =:3, -:0	+:7, =:4, -:1	+:12, =:0, -:0	+:6, =:5, -:1	+:12, =:0, -:0	+:11, =:1, -:0	+:11, =:1, -:0

As shown in Table 7, at the 95% significance level ((\ )), the Wilcoxon rank-sum test is conducted on CEC2022 (Dim = 10). Here, "+" indicates that MultiAO is superior to the other algorithm, "-" indicates that the other algorithm is superior, and "=" indicates no significant difference. From the summary statistics in the last row, it can be seen that, when comparing MultiAO with ALA and SFOA, the results are "+:12, =:0, -:0" in both cases, demonstrating its extremely high stability. When comparing MultiAO with WOA, TOC, WMA, GWO, and HHO, the obtained results are "+:11, =:1, -:0", "+:11, =:1, -:0", "+:11, =:1, -:0", "+:9, =:3, -:0", and "+:9, =:3, -:0", respectively; these results are also dominated by "+", and MultiAO wins in more than half of the cases. When comparing MultiAO with DBO and HBO, the results are "+:7, =:4, -:1" and "+:6, =:5, -:1", with the number of wins exceeding the number of losses. When comparing MultiAO with PSO, the result is "+:4, =:5, -:3", which is essentially a draw. Examining the (p)-values for each function, the (p)-values of F3, F6, and F7 are all less than 0.05, indicating that the performance differences between MultiAO and the compared algorithms are significant. Overall, among the three outcomes, "+" occurs most frequently, suggesting that, on the CEC2022 test functions, MultiAO is typically superior to these optimization algorithms.

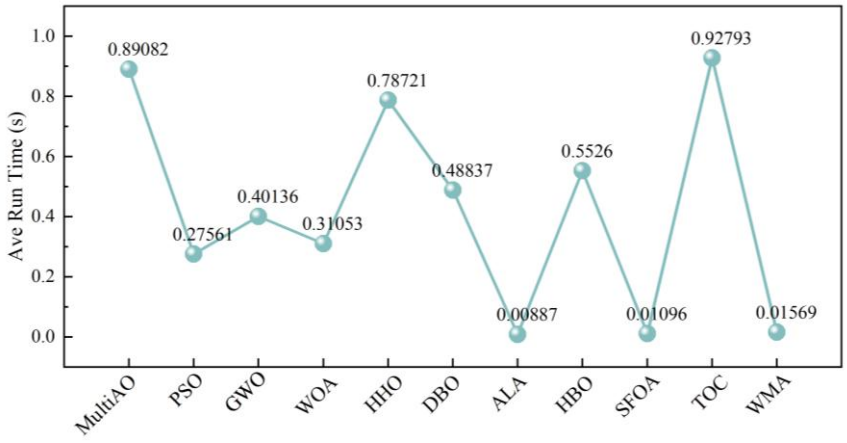
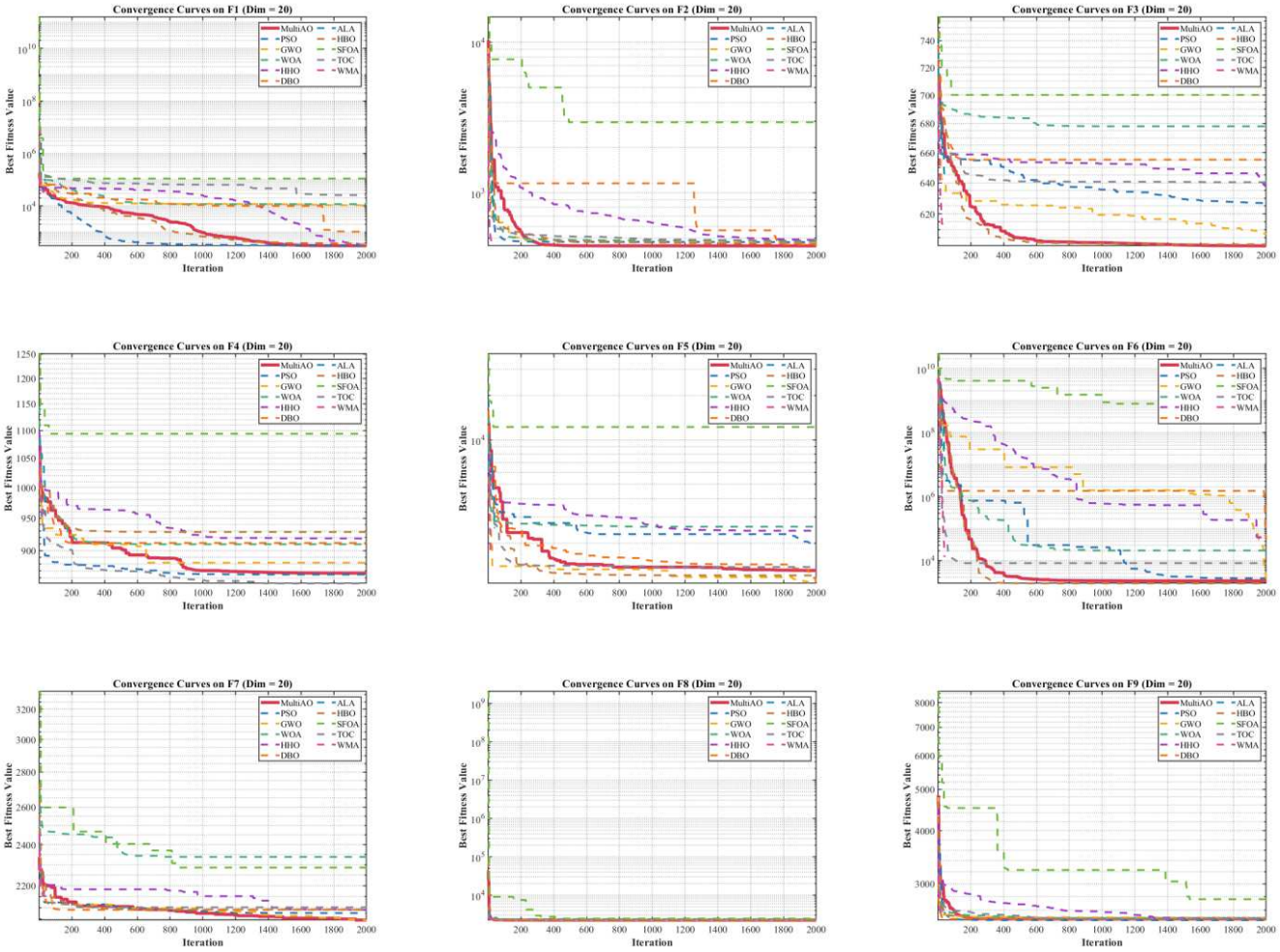


Figure 6. Average Computation Time of Various Algorithms on CEC2022 (Dim = 10).

Figure 6 presents the average computation time of MultiAO and the other ten optimization algorithms. It can be seen that MultiAO is slightly better than TOC, but it still lags behind the other algorithms; nevertheless, the running time of all algorithms is within 1 s, and all of them demonstrate excellent performance.

4.2.3 Twenty-Dimensional Results on CEC2022

Subsequently, this study continues testing at higher dimensionality by comparing MultiAO with the other ten algorithms on CEC2022 (Dim = 20); the results are shown in Figure 7 and Table 8.



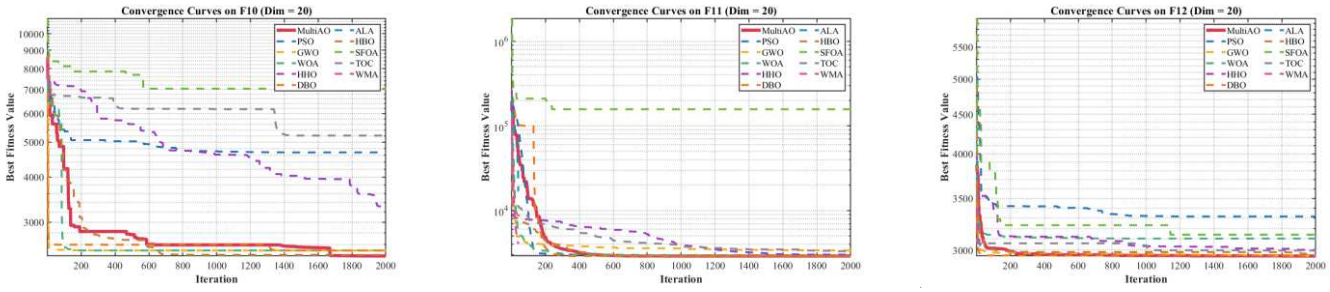


Figure 7. Convergence Curves of Various Algorithms on CEC2022 (Dim = 20).

Table 8. Mean Results of Various Algorithms on CEC2022 (Dim = 20).

Function	Mertri	MultiAO	PSO	GWO	WOA	HHO	DBO	ALA	HBO	SFOA	TOC	WMA
F1	Mean	1259.027	**301.0445**	8938.149	7664.105	335.8028	7385.554	55589.23	380.8495	112572.8	14517.49	35693.7
	Std	1530.765	0.77015	2286.773	2692.721	18.46496	6323.158	16116.01	56.46014	51549.26	20636.05	8393.206
F2	Mean	448.1712	**438.5197**	495.3911	513.3562	470.2691	478.7319	786.3943	455.6664	5208.513	506.4864	460.0234
	Std	17.09663	28.33523	41.47744	44.10901	9.932294	44.92441	171.0039	19.47372	3796.889	64.74446	16.66892
F3	Mean	**600.0247**	632.0296	602.505	659.9654	653.3409	620.7488	647.6614	600.4831	698.4767	639.0833	614.9118
	Std	0.025366	9.824781	1.573721	7.057595	7.648059	11.34305	13.90618	0.96658	17.13868	12.44949	8.714497
F4	Mean	852.9824	856.613	**851.5319**	916.0376	874.0689	903.7191	957.9489	874.8206	1058.734	888.6579	931.1292
	Std	21.75177	14.74599	25.2895	24.94675	16.5025	23.6352	29.67524	20.48088	27.92309	22.04917	16.61509
F5	Mean	1608.943	1490.884	**1095.015**	4089.03	2726.82	1826.305	3553.248	1952.405	10434.28	1432.629	1283.936
	Std	460.2207	512.0317	214.6048	1748.193	239.6278	456.5417	1319.153	449.8826	4462.491	454.5429	533.0213
F6	Mean	7107.144	**5234.229**	1661421	25037.51	72395.34	11049.23	11197564	3955.625	2.78E+09	11287.89	26869.77
	Std	4563.297	6212.116	4152257	19828.03	42644.29	9266.965	6263436	3171.14	1.65E+09	15264.37	13011.72
F7	Mean	2046.279	2091.262	2059.753	2201.457	2161.178	2105.97	2194.43	**2046.034**	2340.681	2156.596	2160.048
	Std	16.16399	43.49181	27.59742	94.79569	63.53778	53.98799	38.59938	17.08379	141.5257	78.01275	28.80693
F8	Mean	**2222.519**	2269.128	2252.931	2283.514	2239.571	2286.362	2335.714	2239.15	3041.734	2289.245	2293.014
	Std	1.569223	82.41689	49.62007	58.30913	8.734138	64.50076	53.39214	36.92486	335.9198	59.05412	55.68881
F9	Mean	2481.385	**2465.344**	2502.095	2518.924	2486.026	2487.979	2559.002	2481.166	3368.14	2525.857	2499.711
	Std	1.574	0.000392	18.85149	30.4282	3.614377	14.93982	53.40652	0.843258	460.5905	74.45327	31.64539
F10	Mean	**2518.377**	3256.201	3246.614	3218.12	3620.15	2627.237	5076.587	2684.399	7188.415	4867.187	4501.283
	Std	69.66422	771.7982	565.4901	1110.106	800.4239	332.1972	1395.606	280.1403	419.5471	1377.395	2136.177
F11	Mean	2902	**2900.023**	3467.35	2986.702	2951.3	2910	19905.79	2920.426	152228.5	3686.218	3883.375
	Std	2.885413	0.017807	285.8097	86.85514	150.63	31.62278	11326.76	42.18189	52328.81	459.7339	232.81
F12	Mean	**2942.38**	3145.95	2956.813	3041.488	3077.421	2989.908	2985.411	2954.528	3264.57	3022.02	2994.222
	Std	5.818619	275.136	12.52333	46.30159	56.92156	38.52078	26.92722	13.21364	120.4469	60.87237	57.1412

According to the results in Figure 7 and Table 8, in the 20-dimensional CEC2022 results, MultiAO exhibits comprehensive and superior performance: on functions F3, F8, F10, and F12, its mean values are better than those of the other algorithms, while its convergence speed is comparable. These results reflect its balanced capability in global exploration and local exploitation.

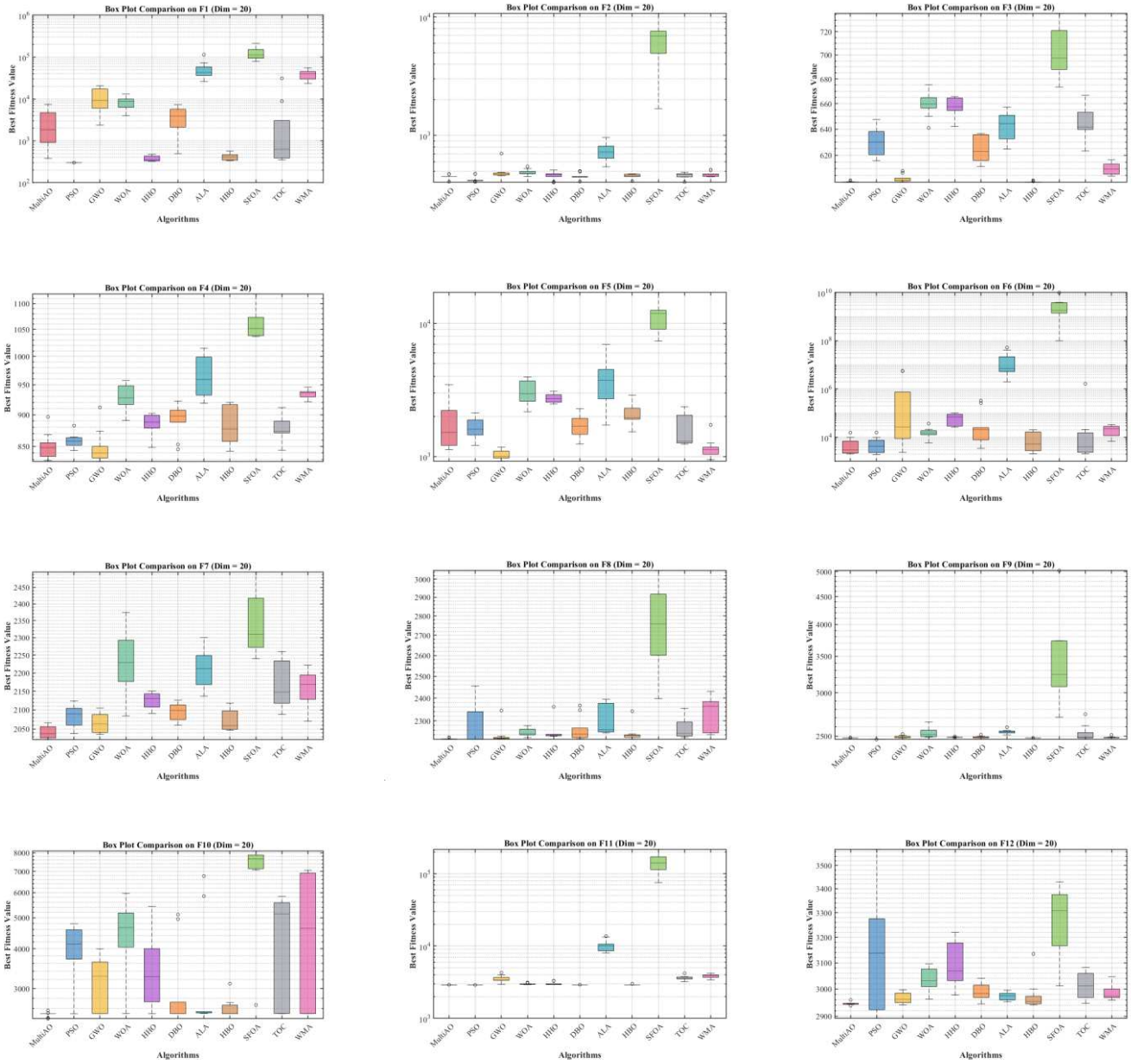


Figure 8. Boxplots of Various Algorithms on CEC2022 (Dim = 20).

As shown in Figure 8, MultiAO achieves the lowest mean values and standard deviations on F2, F3, F7, F8, F9, F10, F11, and F12. This advantage relative to mature, improved algorithms demonstrates that MultiAO possesses very strong capability in addressing challenges of robustness and sensitivity.

Table 9. CEC2022 Algorithm Evaluation Results.

		vs PSO	vs GWO	vs WOA	vs HHO	vs DBO	vs ALA	vs HBO	vs SFOA	vs TOC	vs WMA
F1	p value	0.000182672	0.000182672	0.000329839	0.004586392	0.001314945	0.000182672	0.007284557	0.000182672	0.021133928	0.000182672
	sign	-	+	+	-	+	+	-	+	+	+
F2	p value	0.570750388	0.002827272	0.000439639	0.001007976	0.053902557	0.000182672	0.520522883	0.000182672	0.002827272	0.031209013
	sign	=	+	+	+	=	+	=	+	+	+
F3	p value	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.000182672	0.005795359	0.000182672	0.000182672	0.000182672
	sign	+	+	+	+	+	+	+	+	+	+
F4	p value	0.185876732	0.520522883	0.00058284	0.014019277	0.001314945	0.000182672	0.021133928	0.000182672	0.004586392	0.000182672
	sign	=	=	+	+	+	+	+	+	+	+
F5	p value	0.677584958	0.007284557	0.000329839	0.000182672	0.344704222	0.000329839	0.088973012	0.000182672	0.427355314	0.088973012
	sign	=	-	+	+	=	+	=	+	=	=
F6	p value	0.307489457	0.27303634	0.025748081	0.000182672	0.307489457	0.000182672	0.427355314	0.000182672	0.000605486	0.000768539
	sign	=	=	+	+	=	+	=	+	=	+
F7	p value	0.001706249	0.427355314	0.000439639	0.000439639	0.009108496	0.000182672	0.909721889	0.000182672	0.000182672	0.000182672
	sign	+	=	+	+	+	+	=	+	+	+
F8	p value	0.570750388	0.000182672	0.000182672	0.000182672	0.000439639	0.000182672	0.04515457	0.000182672	0.000182672	0.000182672
	sign	=	+	+	+	+	+	+	+	+	+
F9	p value	0.000182672	0.001007976	0.000329839	0.000768539	0.088973012	0.000182672	0.79133678	0.000182672	0.001314945	0.001007976
	sign	-	+	+	+	=	+	=	+	+	+
F10	p value	0.021133928	0.031209013	0.307489457	0.000768539	0.850106739	0.00058284	0.27303634	0.000182672	0.002827272	0.064022101
	sign	+	+	=	+	=	+	=	+	+	=
F11	p value	0.121224503	0.000182672	0.000182672	0.002827272	0.002785945	0.000182672	0.79133678	0.000182672	0.000182672	0.000182672
	sign	=	+	+	+	+	+	=	+	+	+
F12	p value	0.472675594	0.007284557	0.000182672	0.000182672	0.000439639	0.000246128	0.021133928	0.000182672	0.000246128	0.000246128
	sign	=	+	+	+	+	+	+	+	+	+
Total (+/=/-)		+:2, =:8, -:2	+:10, =:2, -:0	+:11, =:1, -:0	+:11, =:0, -:1	+:7, =:5, -:0	+:12, =:0, -:0	+:5, =:6, -:1	+:12, =:0, -:0	+:10, =:1, -:0	+:10, =:2, -:0

Table 9 reports the Wilcoxon rank-sum test results for the 20-dimensional CEC2022 setting at the 95% confidence level. From the summary in the last row, when comparing MultiAO with ALA and SFOA, both comparisons yield "+:12, =:0, -:0," indicating that MultiAO outperforms these algorithms on every function. When comparing MultiAO with GWO, WOA, HHO, TOC, and WMA, the results are "+:10, =:2, -:0," "+:11, =:1, -:0," "+:11, =:0, -:1," "+:10, =:1, -:0," and "+:10, =:2, -:0," respectively, implying only one or two losses, with outcomes predominantly favorable. Compared with DBO, the result is "+:7, =:5, -:0," indicating that MultiAO only slightly outperforms DBO. However, for PSO and HBO, the results are "+:2, =:8, -:2" and "+:5, =:6, -:1," in which the number of wins is lower than the number of losses. Overall, unfavorable cases are far fewer than favorable outcomes, which also suggests that, on most functions, MultiAO is not inferior to the other algorithms and is often superior to them.

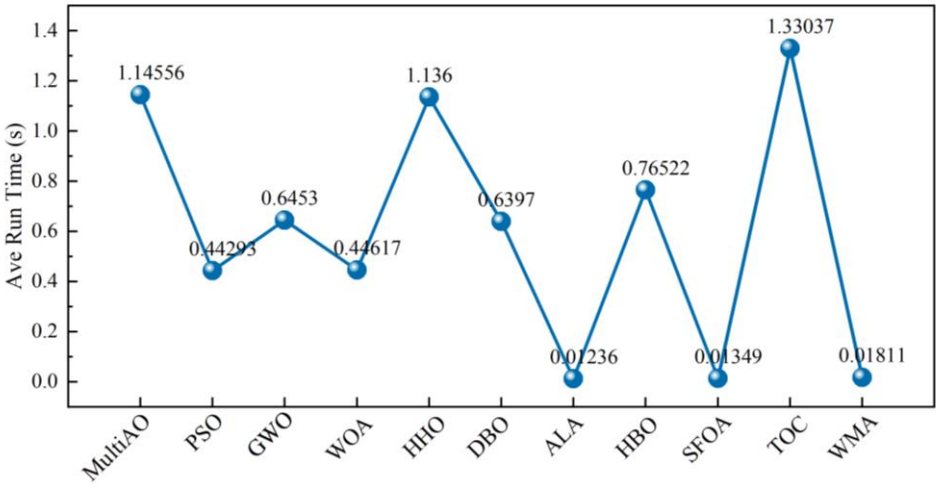


Figure 9. Average Computation Time of Various Algorithms on CEC2022 (Dim = 20).

Figure 9 presents the average computation time of each algorithm. It can be seen that MultiAO is slightly lower than TOC, and the computational costs of ALA, SFOA, and WMA are noticeably lower; however, overall, the runtime of all algorithms is controlled within 1.4 s, indicating high efficiency.

Table 10. Mean Friedman Ranks of All Algorithms on CEC2022.

	MultiAO	PSO	GWO	WOA	HHO	DBO	ALA	HBO	SFOA	TOC	WMA
Dim 10	**2**	4.166667	5.166667	7.666667	6.5	4.5	8.083333	3.916667	10.83333	6.75	6.416667
Dim 20	**2.25**	3.666667	4.75	7.583333	6	5	9.166667	2.916667	11	7	6.666667

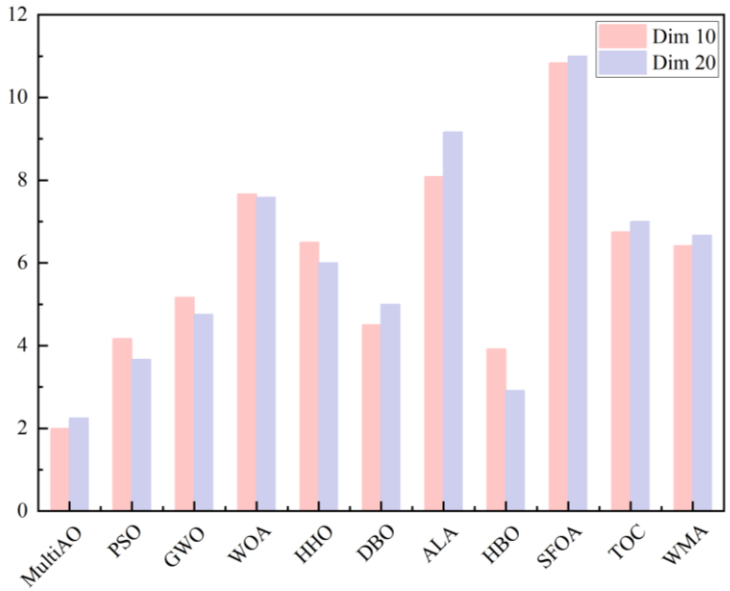


Figure 10. Performance of Mean Friedman Ranks Across Each Dimension on CEC2022.

Table 10 and Figure 10 present the mean Friedman ranks of the algorithms under Dim = 10 and 20, highlighting their relative performance across different settings. MultiAO ranks first on average in both 10 dimensions and 20 dimensions, demonstrating its superior performance and stability: it can achieve a competitive ranking while maintaining a relatively low computation time. By contrast, SFOA and ALA exhibit poorer rankings. These results indicate that MultiAO is overall balanced and efficient, and performs excellently across different dimensionalities.

4.3 Ablation Study

Next, this study conducts an ablation study to gain deeper insight into the contribution of each phase of the algorithm. By systematically removing Phases A, B, and C, we evaluate the impact of each phase, thereby revealing the most critical phase that influences the results. Table 11 reports the mean results after removing different phases.

Table 11. Mean Results After Removing Different Phases.

Function	Mertri	MultiAO	MultiAO (Removing Phase A)	MultiAO (Removing Phase B)	MultiAO (Removing Phase C)
F1	Mean	**1259.027**	38934.82	2994.21	8453.823
	Std	1530.765	10467.18	4101.341	7717.237
F2	Mean	467.4272	455.4421	458.0937	**453.7363**
	Std	31.01407	12.34377	10.37414	13.68303
F3	Mean	**600.0407**	654.8651	601.1939	600.0973
	Std	0.036268	18.90148	1.406858	0.120491
F4	Mean	**853.8523**	938.8413	858.1996	856.7036
	Std	18.88533	31.7254	19.35935	12.82423
F5	Mean	**1335.761**	4773.922	1667.264	1488.172
	Std	263.5585	1165.496	417.5809	529.5302
F6	Mean	6549.565	**6161.478**	7604.273	9074.913
	Std	4456.907	4608.454	4561.787	4764.438
F7	Mean	2040.776	2305.765	2044.279	**2039.98**
	Std	14.71109	135.2335	10.71551	13.83136
F8	Mean	**2222.488**	2493.967	2225.03	2225.424
	Std	2.051704	158.9582	1.977601	5.497301
F9	Mean	2481.8	2484.919	**2480.991**	2481.258
	Std	1.672835	2.56592	0.560322	0.512956
F10	Mean	2522.857	4275.281	**2521.574**	2485.702
	Std	71.04693	800.1662	16.11989	58.32546
F11	Mean	2904.85	**2902.462**	2935.453	2909.938
	Std	7.557174	6.71756	58.94132	25.82414
F12	Mean	2948.969	3156.518	**2944.635**	2950.039
	Std	6.927832	156.1106	4.7037	9.312254
Friedamn Ranking		**1.916667**	3.333333	2.5	2.25

According to the results in Table 11, MultiAO ranks first on F1, F3, F4, F5, and F8, and its mean Friedman rank is also first, indicating that every phase is indispensable. Comparing the three ablated phases, in terms of the mean rank, removing Phase A yields the worst ranking, removing Phase B is next, and removing Phase C results in the best ranking among the ablations; this suggests that Phase A contributes the most among the three, followed by Phase B and Phase C, confirming that Phase A plays an important role in algorithmic performance.

Table 12. Average Computation Time After Removing Different Phases.

Function	MultiAO	MultiAO (Removing Phase A)	MultiAO (Removing Phase B)	MultiAO (Removing Phase C)
Ave Run Time (s)	1.17196	0.554114	0.91046	0.74512

Table 12 reports the average computation time after removing different phases. It can be observed that, after removing Phase A, the time decreases substantially to 0.554114 s; removing Phase C yields the next largest decrease, whereas removing Phase B produces the smallest change. This indicates that Phase A exerts a significant influence on increasing the algorithm's computation time, which enables the algorithm to run efficiently.

4.4 UAV Path-Planning Problem

We apply MultiAO to the three-dimensional UAV path-planning problem to verify its effectiveness and feasibility in complex environments. The UAV path-planning model is given below [3].

4.4.1 Travel Distance Model

The travel distance model is used to compute the UAV's total flight distance. The start point is $A(x_A, y_A, z_A)$, the end point is $B(x_B, y_B, z_B)$, and assuming the UAV $P_i(x_i, y_i, z_i)$ travels along this route, the travel distance can be expressed as:

$$D = \sum_{i=1}^{n-1} \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2}, \tag{17}$$

where, n is the number of points contained in the path, which is determined by the number of algorithm iterations.

4.4.2 Altitude Variation Model

The altitude variation model describes how the UAV adaptively flies according to terrain elevation changes under predefined requirements. Let the height function be $z(x, y)$; the altitude variation can be expressed as:

$$\Delta z = \sum_{i=1}^{n-1} | z_{i+1} - z_i | . \tag{18}$$

4.4.3 Turning-Angle Variation Model

The turning-angle variation model requires the UAV to turn safely during flight so as to avoid obstacles and hazards while reducing energy consumption. The turning angle is divided into the horizontal turning angle θ_i and the vertical turning angle ϕ_i, which can be expressed by the following formulas, respectively:

$$\cos(\theta_i) = \frac{(x_i - x_{i-1})(x_{i+1} - x_i) + (y_i - y_{i-1})(y_{i+1} - y_i)}{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}}, \tag{19}$$

$$\Delta\theta = \sum_{i=2}^{n-1} | \theta_i - \theta_{i-1} |, \tag{20}$$

$$\cos(\phi_i) = \frac{(z_i - z_{i-1})(z_{i+1} - z_i)}{\sqrt{(z_i - z_{i-1})^2}\sqrt{(z_{i+1} - z_i)^2}}, \tag{21}$$

$$\Delta\phi = \sum_{i=2}^{n-1} | \phi_i - \phi_{i-1} | . \tag{22}$$

4.4.4 Objective Function

The objective function jointly considers travel distance, altitude variation, and turning-angle variation, forming a multi-objective optimization problem. The objective function is:

$$\min\left(D + \Delta z + \Delta\theta + \Delta\phi\right). \tag{23}$$

4.4.5 Constraints

First, it is necessary to ensure that the distance d_i between the UAV and an obstacle point $O_j(x_j, y_j, z_j)$ is greater than the safe distance d_{safe}, which can be expressed as:

$$d_i = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}, \tag{24}$$

Using the penalty-function method, a penalty term P_{safe} is introduced:

$$P_{safe} = \begin{cases} 0, & if\ d_i > d_{safe} \\ \infty, & if\ d_i < d_{safe} \\ d_{safe} - d_i, & otherwise \end{cases}, \tag{25}$$

Second, cubic spline interpolation is used to ensure path smoothness; for the UAV path points $\{P_0, P_1, \ldots, P_n\}$, the distance between two adjacent points of the cubic spline interpolation function $S_i(x)$ is:

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \tag{26}$$

The conditions are as follows:

$$S_i(x_i) = P_i, \tag{27}$$

$$S_i(x_{i+1}) = P_{i+1}, \tag{28}$$

$$S_i^{'}(x_{i+1}) = S_{i+1}^{'}(x_{i+1}), \tag{29}$$

$$S_i^{''}(x_{i+1}) = S_{i+1}^{''}(x_{i+1}). \tag{30}$$

In addition, it is also necessary to ensure that the UAV's flight altitude is higher than the terrain but lower than the maximum flight altitude:

$$z_{terrain}(x_i, y_i) \le z_i \le z_{max}, \tag{31}$$

$$\Delta z \le \Delta z_{\max}, \tag{32}$$

Using the penalty-function method, a penalty term P_{alt} is introduced:

$$P_{alt} = \begin{cases} 0, & if\ z_i > z_{max} \\ \infty, & if\ z_i < z_{terrain}(x_i, y_i). \\ z_{max} - z_{terrain}(x_i, y_i), & otherwise \end{cases} \tag{33}$$

Finally, it is necessary to ensure that the turning angle is within an acceptable range:

$$0 < \theta_i \le \theta_{\max}, \tag{34}$$

$$0 < \phi_i \le \phi_{\max}, \tag{35}$$

Using the penalty-function method, a penalty term P_{angle} is introduced:

$$P_{angle} = \begin{cases} 0, & if\ \theta_i > \theta_{\max}\ and\ \phi_i > \phi_{\max} \\ \infty, & if\ \theta_i \le 0\ and\ \phi_i \le 0 \\ \theta_i - \theta_{\max} + \phi_i - \phi_{\max}, & otherwise \end{cases}. \tag{36}$$

In summary, the complete objective function and constraint conditions are as follows:

$$\min\left(D + \Delta z + \Delta\theta + \Delta\phi + P_{safe} + P_{alt} + P_{angle}\right).$$
$$s.t. \begin{cases} S_i(x_i) = P_i \\ S_i(x_{i+1}) = P_{i+1} \\ S_i^{'}(x_{i+1}) = S_{i+1}^{'}(x_{i+1}) \\ S_i^{''}(x_{i+1}) = S_{i+1}^{''}(x_{i+1}) \\ z_{terrain}(x_i, y_i) \le z_i \le z_{max} \\ \Delta z \le \Delta z_{\max} \\ 0 < \theta_i \le \theta_{\max} \\ 0 < \phi_i \le \phi_{\max} \\ \Delta\theta \le \Delta\theta_{\max} \\ \Delta\phi \le \Delta\phi_{\max} \end{cases}. \tag{37}$$

4.4.6 Parameter Settings

To more realistically simulate the environment, this paper constructs three types of 3D maps and adds multiple obstacles to increase environmental complexity, and sets up two scenarios—single-UAV and multi-UAV—to validate the performance of MultiAO. The urban terrain, mountainous terrain, composite-terrain obstacles, and initial parameters in this paper are referenced from the study of Liu et al. [3], and the UAV's initial parameters are shown in Table 13.

Table 13. Initial UAV Parameters.

Parameter	Value
$z_{terrain}$	Calculate based on terrain
z_{max}	50m
$\theta_{\max}$	45°
$\phi_{\max}$	45°

4.4.7 Comparative Results Across Three Terrains (Single UAV)

To conduct a simulation-based comparison for the UAV path-planning problem, this part selects the top three algorithms from the benchmark-function results for comparison, namely MultiAO (ours), PSO, and HBO. In the single-UAV scenario, we repeat 50 runs in three complex environments—urban terrain, mountainous terrain, and composite terrain—and record the results for comparison.

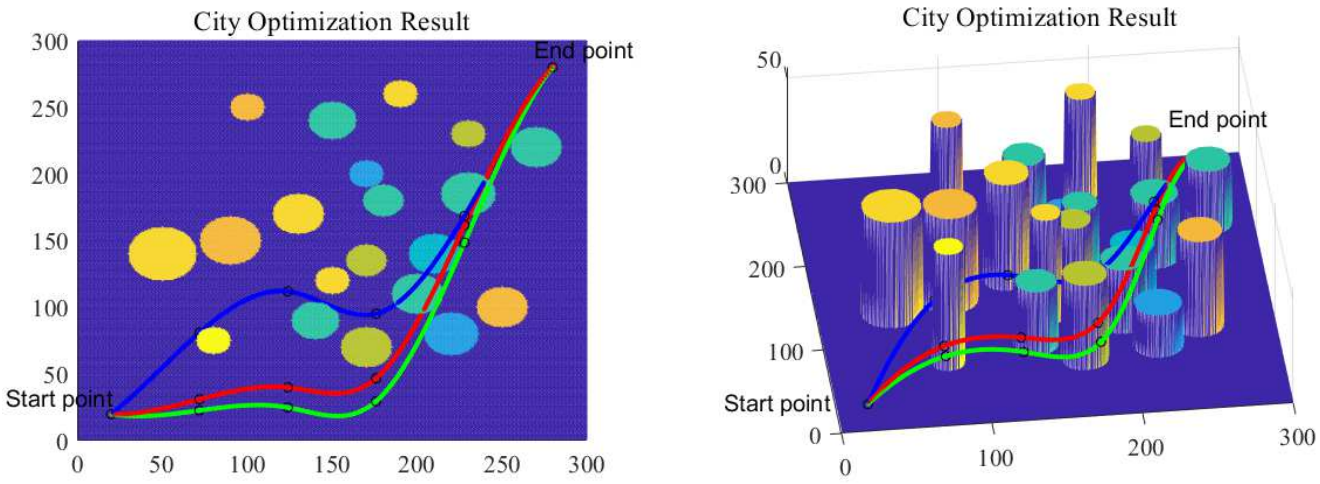


Figure 11. Path Comparison in Urban Terrain (Single UAV).

Figure 11 presents the path comparison of the three algorithms in urban terrain: the left panel is a 2D planar map, and the right panel is a 3D view. Here, red denotes MultiAO, blue denotes PSO, and green denotes HBO. It can be observed that MultiAO yields a smoother trajectory, a lower flight altitude, lower energy consumption, and a shorter path, whereas the other algorithms—especially PSO—have not yet found the optimal path.

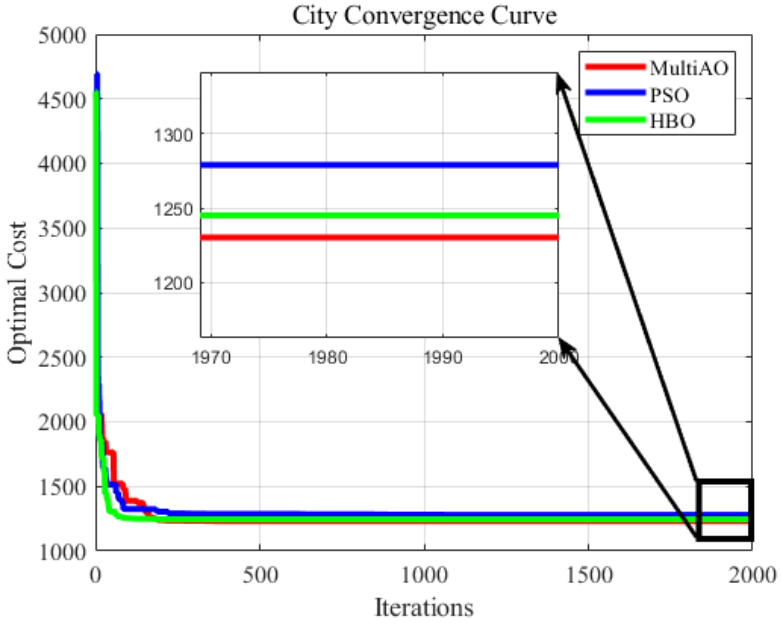


Figure 12. Comparison of Convergence Curves in Urban Terrain (Single UAV).

From Figure 12, it can be seen that although PSO and HBO can converge faster in the early stage, MultiAO finds the optimal value. While maintaining early-stage search intensity, it also ensures late-stage convergence capability; therefore, by comparison, MultiAO is more stable and efficient in complex urban terrain, highlighting its robustness.

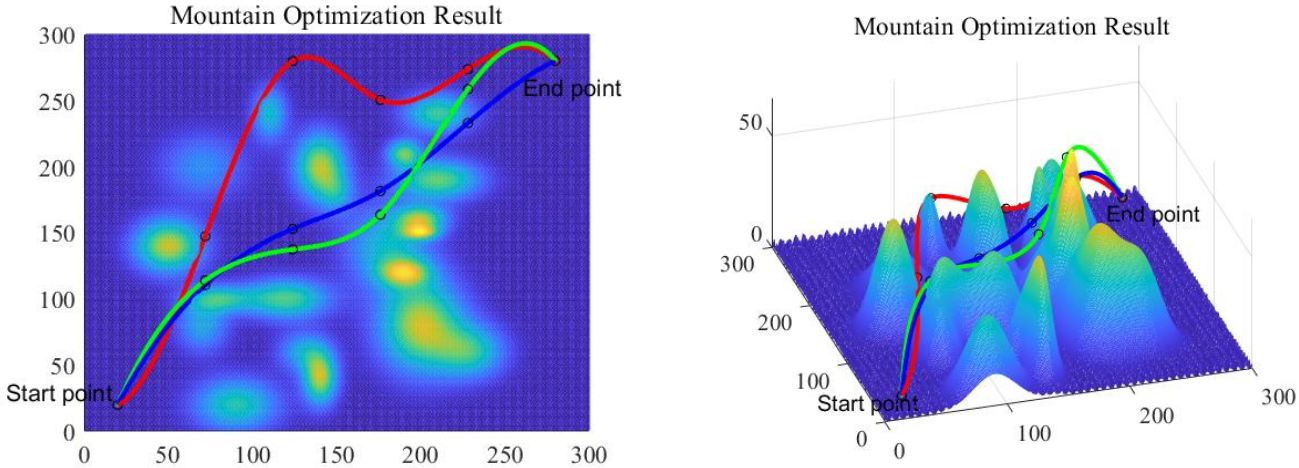


Figure 13. Path Comparison in Mountainous Terrain (Single UAV).

Figure 13 presents the 2D planar map and 3D view in mountainous terrain. From the path comparison, it can be found that the trajectory produced by MultiAO (red) traverses relatively simpler terrain, reducing high energy consumption in complex terrain, whereas PSO (blue) and HBO (green), although having a slightly shorter travel distance, neglect the high energy-consumption issues induced by turning-angle and altitude-variation changes.

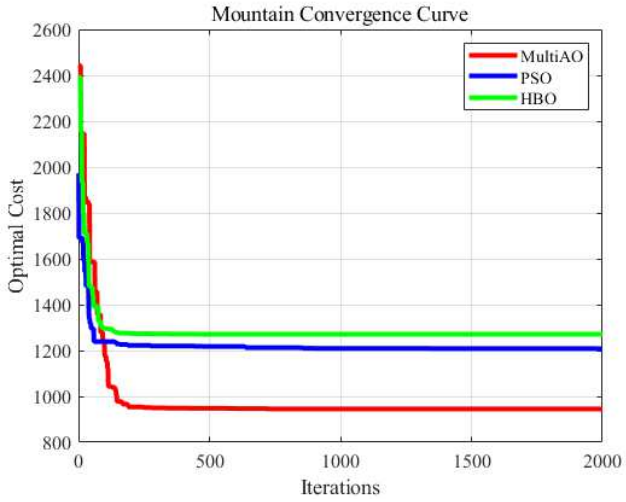


Figure 14. Comparison of Convergence Curves in Mountainous Terrain (Single UAV).

From Figure 14, it can be seen that the other two algorithms fall into local optima in the early stage, and the costs they obtain are far higher than that of MultiAO. This further confirms MultiAO's comprehensive search capability and efficient convergence ability, enabling it to rapidly find the optimal path and maintain optimal convergence even in more complex mountainous terrain.

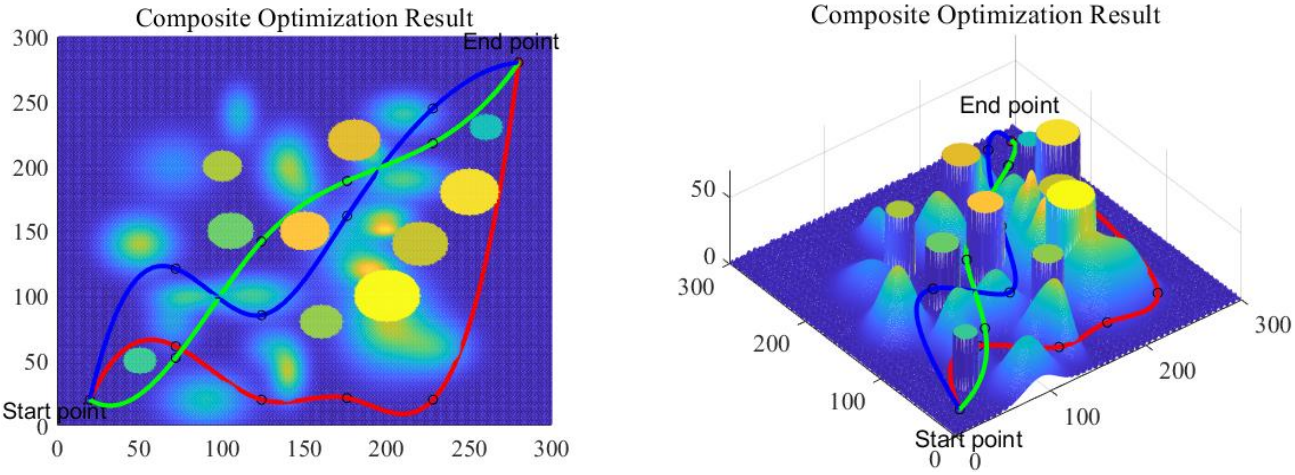


Figure 15. Path Comparison in Composite Terrain (Single UAV).

Figure 15 presents the path comparison in composite terrain. Composite terrain combines urban terrain and mountainous terrain and is more complex and challenging. MultiAO (red) still jointly considers travel distance, altitude variation, and turning-angle variation, achieving travel-distance minimization while keeping altitude variation and turning-angle variation minimal, whereas the other algorithms exhibit inadequate consideration, neglect the losses introduced by altitude and turning changes, overemphasize travel distance, and fail to find the minimum-cost path.

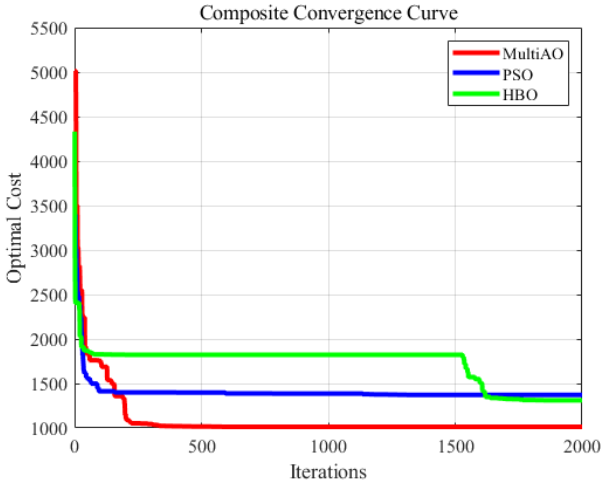


Figure 16. Comparison of Convergence Curves in Composite Terrain (Single UAV).

From Figure 16, it can be observed that PSO falls into a local optimum early on, and although HBO re-explores in the later stage, it still fails to find the global optimum. MultiAO, by contrast, maintains rapid convergence and finds the global optimum; moreover, in composite terrain, the gap between its best solution and those of the other two algorithms approaches 500, far larger than in urban and mountainous terrain, further validating its effectiveness in complex environments.

Table 14. Performance Comparison of Algorithms Across Three Terrains (Single UAV).

		MultiAO	PSO	HBO
Urban terrain	Best	366.2826	**366.274**	366.294
	Worst	946.9183	**1331.47**	1230.32
	Ave	**531.86**	993.4482	777.5766
	Std	**238.213**	379.2753	328.2359
	Time (s)	68.25	74.3	**67.17**
Mountainous terrain	Best	**749.057**	749.4312	749.0847
	Worst	990.0877	**1166.37**	1146.23
	Ave	**863.458**	874.1038	886.7219
	Std	**108.971**	185.9406	191.9926
	Time (s)	80.93	**80.46**	80.98
Composite terrain	Best	**820.894**	823.6718	824.8736
	Worst	**1148.42**	1014.184	1148.314
	Ave	981.4032	**951.504**	957.2217
	Std	117.9604	**77.412**	133.832
	Time (s)	**72.53**	86.48	82.36

Table 14 reports the performance comparison across the three terrains, where bold black font denotes the best case and bold red font denotes the worst case. It can be seen that, in terms of best-value comparison, MultiAO ranks first in both mountainous terrain and composite terrain. In terms of worst-value comparison, MultiAO reaches the worst case only in composite terrain. In terms of mean-value comparison, MultiAO achieves the best performance in urban terrain and mountainous terrain, and it also attains the lowest standard deviation. Regarding runtime, although it wins only in composite terrain, in the other terrains the time differences are not large. In summary, MultiAO attains more wins than PSO and HBO, demonstrating its excellent performance in solving UAV path-planning problems in complex environments.

4.4.8 Comparative Results Across Three Terrains (Multi-UAV)

In this section, the single-UAV setting is extended to a multi-UAV setting. We apply MultiAO to the path-planning problem of three UAVs, and we again repeat 50 runs in urban terrain, mountainous terrain, and composite terrain to record results for comparison.

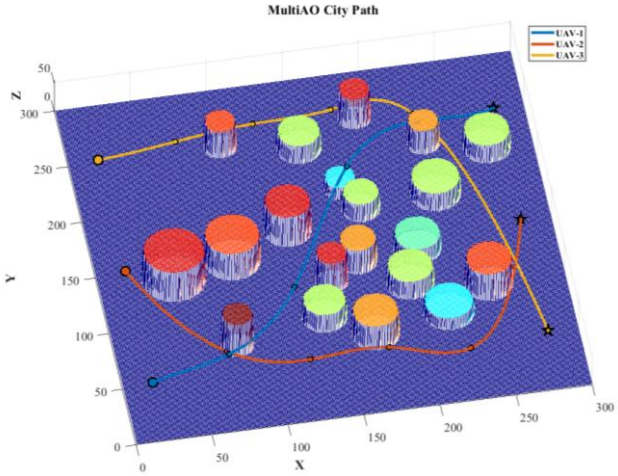


Figure 17. Path Comparison in Urban Terrain (Multi-UAV).

From Figure 17, it can be seen that, when addressing the multi-UAV path-planning problem, MultiAO still jointly considers travel distance, altitude variation, and turning-angle variation. Although the number of UAVs increases and the start and end points differ, MultiAO adaptively identifies more low-cost paths, yielding trajectories that are smooth and energy-efficient.

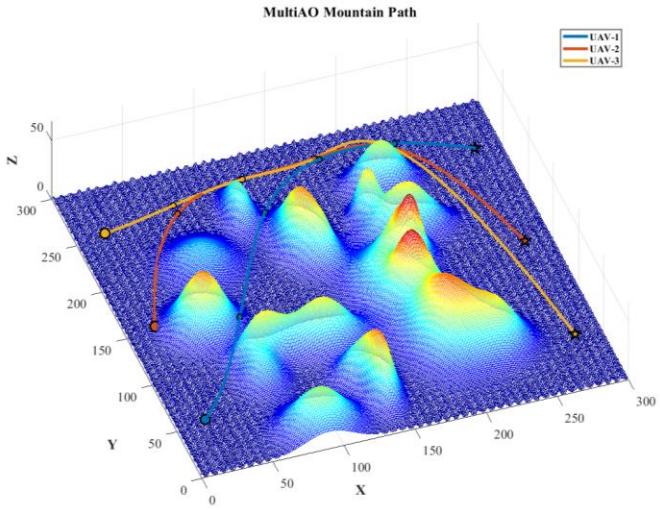


Figure 18. Path Comparison in Mountainous Terrain (Multi-UAV).

From Figure 18, it can be seen that MultiAO does not choose to pass through the middle in mountainous terrain to seek the shortest path; instead, it rapidly finds the global optimal path, and all three paths perform excellently. This further confirms MultiAO's stability and efficiency in solving complex problems.

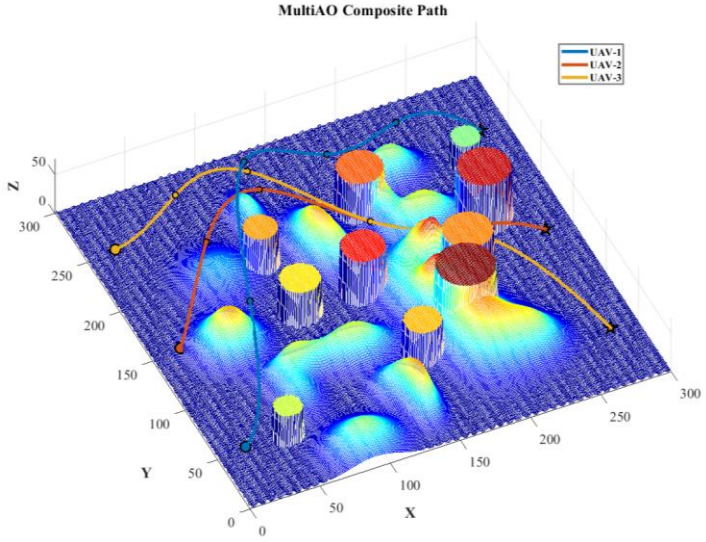


Figure 19. Path Comparison in Composite Terrain (Multi-UAV).

In Figure 19, the terrain is more complex; MultiAO also perfectly avoids complex obstacles and finds the optimal solution to this multi-objective optimization problem. The flight paths are not only flatter, but they also avoid sharp turns that could endanger safety and increase losses, demonstrating its effectiveness and feasibility in complex environments.

Table 15. Performance Comparison of MultiAO Across Three Terrains (Multi-UAV).

		MultiAO				PSO				HBO			
		UAV 1	UAV2	UAV 3	Total	UAV 1	UAV2	UAV 3	Total	UAV 1	UAV2	UAV 3	Total
Urban terrain	Best	**324.332**	294.2091	**351.359**	**969.8995**	324.3318	293.8651	424.1886	1042.3855	409.8906	**283.375**	363.5276	1056.7936
	Worst	1210.163	**379.168**	424.1897	2013.5213	**1210.77**	378.6925	532.9287	2122.3919	915.1745	378.8111	**1184.16**	**2478.15**
	Ave	837.5066	361.8909	**396.867**	1596.2641	786.5469	361.7238	451.732	1600.0027	**612.051**	**342.689**	564.2439	**1518.985**
	Std	449.9781	**37.8358**	**37.5493**	525.3632	398.0553	37.9341	47.0926	**483.082**	**276.71**	49.4971	347.5511	673.7585
Mountainous terrain	Best	699.8665	423.9954	**455.441**	1579.3027	699.5746	423.1428	455.6518	1578.3692	**698.454**	**423.112**	455.6249	**1577.191**
	Worst	802.274	**481.879**	456.0351	1740.1881	923.738	425.5236	456.2969	1805.5585	**925.417**	425.6376	**456.302**	**1807.357**
	Ave	**723.891**	444.885	**455.706**	**1624.4823**	777.4413	424.114	455.9105	1657.4658	821.2499	**423.869**	455.8562	1700.9752
	Std	**44.1791**	24.2803	**0.2354**	**68.6948**	106.6083	1.1627	0.2543	108.0253	114.1489	**1.1081**	0.2661	115.5231
Composite terrain	Best	**894.307**	**490.108**	906.1561	**2290.5708**	922.5855	1121.987	912.5655	2957.1383	885.8728	1113.847	**901.717**	2901.4367
	Worst	1067.437	**1367.75**	1318.516	3753.7041	**1263.56**	1189.59	1337.964	3791.1144	1018.212	1154.008	**1708.48**	**3880.698**
	Ave	949.7363	**1047.99**	**1223.91**	**3221.6405**	1053.724	1149.043	1242.269	3445.0357	**938.85**	1131.01	1308.164	3378.0236
	Std	70.5526	329.0696	**178.207**	577.8289	143.3031	27.1356	184.6678	355.1065	**49.1365**	**19.6458**	285.4597	**354.242**

In Table 15, bold black font denotes the best case, and bold red font denotes the worst case. In the "Total" column, MultiAO wins 5 times, with its best value reaching as low as 969.8995; PSO wins once; HBO wins 3 times and loses 3 times. This pronounced difference confirms that MultiAO's overall performance in complex terrain is superior to that of the other algorithms. In the best-value comparison, MultiAO wins 5 times, reaching as low as 324.332; PSO has no wins; HBO wins 4 times. In the mean-value comparison, MultiAO wins 4 times and attains the smallest standard deviation in all cases; PSO has no wins; HBO wins 4 times, but its standard deviation is not always the smallest. In the worst-value comparison, MultiAO loses 3 times, PSO loses 2 times, and HBO loses 7 times. Overall, therefore, MultiAO is relatively stable and demonstrates excellent performance for complex multi-UAV path-planning problems across diverse complex environments.

In summary, MultiAO exhibits significant advantages across various terrains and different UAV path-planning settings. In terrain-path comparisons, MultiAO's trajectories jointly consider multiple factors, remain smooth and energy-efficient, and achieve globally optimal paths, reflecting robustness. In convergence curves, MultiAO not only maintains early-stage global search but also preserves rapid late-stage convergence, without falling into local optima. Moreover, across multiple performance comparisons, MultiAO consistently demonstrates high performance and stability, further validating its effectiveness and feasibility in complex environments.

5 Conclusions

This paper proposes a multi-agent optimization algorithm based on a state-dependent dynamic evolution mechanism—MultiAO—aimed at addressing delivery path-planning problems for UAV swarms in complex environments. Inspired by the "State-Dependent Dynamic Evolution" mechanism in multi-agent complex systems, MultiAO characterizes the system-state changes corresponding to agents across different search stages and introduces three interrelated evolutionary phases—gravitational accretion and centroid rebellion, orthogonal singularity decoupling, and hysteresis effects with a memory potential field—thereby enabling dynamic switching between global exploration and local exploitation. Comparisons on the CEC2022 benchmark suite against other widely cited state-of-the-art metaheuristic algorithms show that MultiAO achieves the lowest mean values on 8 out of 12 test functions spanning 10 and 20 dimensions, with relatively low standard deviations; it also attains the highest number of wins in the Wilcoxon rank-sum tests, and its mean Friedman rank is consistently first. In the ablation study, this paper finds that all three phases of MultiAO make positive contributions to algorithmic performance, with Phase A contributing the most. In addition, this paper applies MultiAO to the UAV path-planning problem, and the results show that, across urban terrain, mountainous terrain, and composite terrain, under both single-UAV and multi-UAV scenarios, MultiAO achieves optimal convergence. Notably, in the multi-UAV scenario performance comparison, MultiAO wins five times, and its best value reaches as low as 324.332. Overall, while maintaining convergence efficiency, MultiAO improves solution diversity and stability, and demonstrates outstanding effectiveness and feasibility in complex environments.

MultiAO demonstrates exceptional capability in solving UAV path-planning problems. Future research may focus on further improving the adaptability and scalability of MultiAO, including developing new strategic mechanisms and conducting rigorous validation in application domains that demand higher requirements and allow less exploration, thereby addressing a broader range of diverse and complex real-world problems. In addition, MultiAO can be integrated with real-time environments to continuously enhance its efficiency and create new possibilities, providing solutions for more fields.

Acknowledgments. We would like to show our greatest appreciation to anonymous reviewers, editor and those who have helped to contribute to this paper writing.

Author Contributions Statement. Weifei Gan: Writing – original draft, Validation, Software, Methodology, Conceptualization. Xiliang Zhang: Writing – review & editing, Supervision, Funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding. No external funding provided.

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

Data Availability. The datasets used and analyzed during the current study available from the corresponding author on reasonable request.

References

1. Valavanis, K. P., & Vachtsevanos, G. J. (2014). Handbook of unmanned aerial vehicles. Springer Publishing Company, Incorporated.
2. Zheng, X., Bao, C., & He, Z. (2018, August). Design of simulation test platform for UAV flight control system. In Journal of Physics: Conference Series (Vol. 1069, No. 1, p. 012022). IOP Publishing.
3. Liu, S., Jin, Z., Lin, H., & Lu, H. (2024). An improve crested porcupine algorithm for UAV delivery path planning in challenging environments. Scientific Reports, 14(1), 20445.
4. Wu, Y., Nie, M., Ma, X., Guo, Y., & Liu, X. (2023). Co-evolutionary algorithm-based multi-unmanned aerial vehicle cooperative path planning. Drones, 7(10), 606.
5. Gugan, G., & Haque, A. (2023). Path planning for autonomous drones: Challenges and future directions. Drones, 7(3), 169.
6. Yahia, H. S., & Mohammed, A. S. (2023). Path planning optimization in unmanned aerial vehicles using meta-heuristic algorithms: A systematic review. Environmental Monitoring and Assessment, 195(1), 30.
7. Hooshyar, M., & Huang, Y. M. (2023). Meta-heuristic algorithms in UAV path planning optimization: A systematic review (2018–2022). Drones, 7(12), 687.
8. Fu, S., Li, K., Huang, H., Ma, C., Fan, Q., & Zhu, Y. (2024). Red-billed blue magpie optimizer: a novel metaheuristic algorithm for 2D/3D UAV path planning and engineering design problems. Artificial Intelligence Review, 57(6), 134.
9. Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic algorithms: A comprehensive review. Computational intelligence for multimedia big data on the cloud with engineering applications, 185-231.
10. Parouha, R. P., & Verma, P. (2021). Design and applications of an advanced hybrid meta-heuristic algorithm for optimization problems. Artificial Intelligence Review, 54(8), 5931-6010.
11. Xiong, T., Li, H., Ding, K., Liu, H., & Li, Q. (2023). A hybrid improved symbiotic organisms search and Sine–cosine particle swarm optimization method for drone 3D path planning. Drones, 7(10), 633.
12. Chen, Y., Tian, Z., Zhang, K., Zhao, F., & Zhao, A. (2025). An Improved Greater Cane Rat Algorithm with Adaptive and Global-Guided Mechanisms for Solving Real-World Engineering Problems. Biomimetics, 10(9), 612.
13. Kabir, M. M., Shahjahan, M., & Murase, K. (2012). A new hybrid ant colony optimization algorithm for feature selection. Expert Systems with Applications, 39(3), 3747-3763.

14. Liu, S., & Wan, G. (2025). Adaptive Customization of Electronic Commerce Packaging for Sustainable Business Development. Journal of Modern Social Sciences, 2(1), 56-64.
15. Akinola, O. O., Ezugwu, A. E., Agushaka, J. O., Zitar, R. A., & Abualigah, L. (2022). Multiclass feature selection with metaheuristic optimization algorithms: a review. Neural Computing and Applications, 34(22), 19751-19790.
16. Marini, F., & Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. Chemometrics and intelligent laboratory systems, 149, 153-165.
17. Hatta, N. M., Zain, A. M., Sallehuddin, R., Shayfull, Z., & Yusoff, Y. (2019). Recent studies on optimisation method of Grey Wolf Optimiser (GWO): a review (2014–2017). Artificial intelligence review, 52(4), 2651-2683.
18. Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. Advances in engineering software, 95, 51-67.
19. Alabool, H. M., Alarabiat, D., Abualigah, L., & Heidari, A. A. (2021). Harris hawks optimization: a comprehensive review of recent variants and applications. Neural computing and applications, 33(15), 8939-8980.
20. Xue, J., & Shen, B. (2023). Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. The journal of Supercomputing, 79(7), 7305-7336.
21. Su, H., Zhao, D., Heidari, A. A., Liu, L., Zhang, X., Mafarja, M., & Chen, H. (2023). RIME: A physics-based optimization. Neurocomputing, 532, 183-214.
22. Deng, L., & Liu, S. (2023). Snow ablation optimizer: A novel metaheuristic technique for numerical optimization and engineering design. Expert Systems with Applications, 225, 120069.
23. Abdullah, M., Alamgeer, K. S., Hafeez, G., Alghamdi, B., Alsafran, A. S., & Babar, M. Z. (2025). Electromagnetic wave propagation algorithm: A novel electromagnetic wave propagation-inspired optimizer for engineering applications. Ain Shams Engineering Journal, 16(11), 103615.
24. Peng, Y., Gu, S., Liang, Y., Ouyang, K., Li, Y., Wang, K., ... & Fan, C. (2025). Wave Optics Optimizer: A novel meta-heuristic algorithm for engineering optimization. Communications in Nonlinear Science and Numerical Simulation, 109337.
25. Ghasemi, M., Khodadadi, N., Trojovský, P., Li, L., Mansor, Z., Abualigah, L., ... & El-Kenawy, E. S. M. (2025). Kirchhoff's law algorithm (KLA): A novel physics-inspired non-parametric metaheuristic algorithm for optimization problems. Artificial Intelligence Review, 58(10), 325.
26. Weile, D. S., & Michielssen, E. (2002). Genetic algorithm optimization applied to electromagnetics: A review. IEEE Transactions on Antennas and Propagation, 45(3), 343-353.
27. Price, K. V. (2013). Differential evolution. In Handbook of optimization: From classical to modern approach (pp. 187-214). Berlin, Heidelberg: Springer Berlin Heidelberg.
28. Omran, M. G., Salman, A., & Engelbrecht, A. P. (2005, December). Self-adaptive differential evolution. In International conference on computational and information science (pp. 192-199). Berlin, Heidelberg: Springer Berlin Heidelberg.
29. Zhang, J., & Sanderson, A. C. (2009). JADE: adaptive differential evolution with optional external archive. IEEE Transactions on evolutionary computation, 13(5), 945-958.
30. Gao, H., & Zhang, Q. (2024). Alpha evolution: An efficient evolutionary algorithm with evolution path adaptation and matrix generation. Engineering Applications of Artificial Intelligence, 137, 109202.
31. Wang, J., & Shang, Z. (2025). Traffic jam optimizer: A novel swarm-based metaheuristic algorithm for solving global optimization problems. Applied Mathematical Modelling, 116410.
32. Liu, S., Xiang, Y., Guo, X., Zhao, F., Zhao, A., & Wu, W. (2025). Market Game Optimization Algorithm: A Metaheuristic Inspired by Symmetric Competitive Behavior of Merchants and Consumers. Symmetry, 17(12), 2118.
33. Chen, Y., Wang, G., Yin, B., Ma, C., Wu, Z., & Gao, M. (2025). Basketball team optimization algorithm (BTOA): a novel sport-inspired meta-heuristic optimizer for engineering applications. Scientific Reports, 15(1), 21629.
34. Wang, L., Du, H., Zhang, Z., Hu, G., Mirjalili, S., Khodadadi, N., ... & Zhao, W. (2025). Tianji's horse racing optimization (THRO): a new metaheuristic inspired by ancient wisdom and its engineering optimization applications. Artificial Intelligence Review, 58(9), 282.
35. Lang, Y., & Gao, Y. (2025). Dream Optimization Algorithm (DOA): A novel metaheuristic optimization algorithm inspired by human dreams and its applications to real-world engineering problems. Computer Methods in Applied Mechanics and Engineering, 436, 117718.
36. Wolpert, D. H., & Macready, W. G. (2002). No free lunch theorems for optimization. IEEE transactions on evolutionary computation, 1(1), 67-82.
37. Yu, X., Jiang, N., Wang, X., & Li, M. (2023). A hybrid algorithm based on grey wolf optimizer and differential evolution for UAV path planning. Expert Systems with Applications, 215, 119327.
38. Sonny, A., Yeduri, S. R., & Cenkeramaddi, L. R. (2023). Autonomous UAV path planning using modified PSO for UAV-assisted wireless networks. IEEE Access, 11, 70353-70367.
39. Liu, G., Shu, C., Liang, Z., Peng, B., & Cheng, L. (2021). A modified sparrow search algorithm with application in 3d route planning for UAV. Sensors, 21(4), 1224.
40. Zheng, X., Liu, R., & Li, S. (2025). A novel improved dung beetle optimization algorithm for collaborative 3D path planning of UAVs. Biomimetics, 10(7), 420.
41. Gu, G., Li, H., & Zhao, C. (2024). A Multi-Strategy Enhanced Marine Predator Algorithm for Global Optimization and UAV Swarm Path Planning (April 2024). IEEE Access.
42. Dorri, A., Kanhere, S. S., & Jurdak, R. (2018). Multi-agent systems: A survey. Ieee Access, 6, 28573-28593.
43. Wooldridge, M. (2009). An introduction to multiagent systems. John wiley & sons.
44. Bujok, P., & Kolenovsky, P. (2022, July). Eigen crossover in cooperative model of evolutionary algorithms applied to CEC 2022 single objective numerical optimisation. In 2022 IEEE congress on evolutionary computation (CEC) (pp. 1-8). IEEE.
45. Marini, F., & Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. Chemometrics and intelligent laboratory systems, 149, 153-165.
46. Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in engineering software, 69, 46-61.
47. Nasiri, J., & Khiyabani, F. M. (2018). A whale optimization algorithm (WOA) approach for clustering. Cogent Mathematics & Statistics, 5(1), 1483565.
48. Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. Future generation computer systems, 97, 849-872.
49. Xue, J., & Shen, B. (2023). Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. The journal of Supercomputing, 79(7), 7305-7336.
50. Xiao, Y., Cui, H., Khurma, R. A., & Castillo, P. A. (2025). Artificial lemming algorithm: a novel bionic meta-heuristic technique for solving real-world engineering optimization problems. Artificial Intelligence Review, 58(3), 84.
51. Ouertani, M. W., Manita, G., & Korbaa, O. (2025). Hannibal Barca optimizer: the power of the pincer movement for global optimization and multilevel image thresholding. Cluster Computing, 28(7), 482.

52. Zhong, C., Li, G., Meng, Z., Li, H., Yildiz, A. R., & Mirjalili, S. (2025). Starfish optimization algorithm (SFOA): a bio-inspired metaheuristic algorithm for global optimization compared with 100 optimizers. Neural Computing and Applications, 37(5), 3641-3683.
53. Braik, M., Al-Hiary, H., Alzoubi, H., Hammouri, A., Azmi Al-Betar, M., & Awadallah, M. A. (2025). Tornado optimizer with Coriolis force: a novel bio-inspired meta-heuristic algorithm for solving engineering problems. Artificial Intelligence Review, 58(4), 123.
54. Ghasemi, M., Deriche, M., Trojovský, P., Mansor, Z., Zare, M., Trojovská, E., ... & kadkhoda Mohammadi, S. (2025). An efficient bio-inspired algorithm based on humpback whale migration for constrained engineering optimization. Results in Engineering, 25, 104215.