

## Article

# Research on Simultaneous Arrival Route Planning for UAV Clusters Based on an Improved NSGA-III Algorithm

Duo Qi <sup>1</sup>, Xiaoyu Shi <sup>2</sup>, Hao Li <sup>3,\*</sup>, Xingyu He <sup>1</sup> and Xiaoyue Ren <sup>1</sup>

<sup>1</sup> Air Traffic Control Navigation College, Air Force Engineering University, Xi'an 710051, China; qi33song@sina.com (D.Q.); hxy\_19890708@163.com (X.H.); m15291810964@163.com (X.R.)

<sup>2</sup> Department of Air-Defense Early Warning Equipment, Air Force Early Warning Academy, Wuhan 430012, China; star-yu.1@163.com

<sup>3</sup> Department of Intelligence, Air Force Early Warning Academy, Wuhan 430012, China

\* Correspondence: afeu\_li@163.com; Tel.: +86-138-7116-1491

## Highlights

### What are the main findings?

- A time-of-arrival prediction model was constructed, achieving simultaneous arrival for the UAV swarm. The optimization objectives and constraints were refined, enhancing the feasibility of the flight paths.
- A method based on the Swarm-optimized NSGA III was developed, improving the efficiency of NSGA III in multi-objective optimization.

### What is the implication of the main finding?

- The improved NSGA III optimization strategy addresses the shortcomings of the NSGA III algorithm in handling complex constraints and multi-objective optimization problems, significantly enhancing search accuracy and convergence speed.
- The proposed method optimizes flight time, reduces resource consumption, and effectively avoids threats while ensuring simultaneous arrival of the UAV swarm.

## Abstract

This paper addresses the challenge of simultaneous arrival for UAV clusters and proposes a route planning method based on an enhanced Non-dominated Sorting Genetic Algorithm III (NSGA-III). Initially, the paper defines the simultaneous arrival problem and formulates the corresponding mathematical model, considering the complexity of multi-objective optimization in UAV clusters. A novel path generation framework is introduced, which incorporates multiple optimization objectives—such as time coordination, threat mitigation, and resource consumption—aimed at improving flight safety, efficiency, and resource management. To enhance the algorithm's search performance, a hybrid approach combining the Artificial Bee Colony (ABC) algorithm with NSGA-III is proposed. This improved NSGA-III strategy overcomes the limitations of the original algorithm in managing complex constraints and multi-objective optimization problems, resulting in significant improvements in search accuracy and convergence speed. Finally, the performance of the improved algorithm is evaluated through simulations and compared with traditional methods. The results show that the proposed approach optimizes flight time, reduces resource consumption, and effectively mitigates threats, all while ensuring the simultaneous arrival of UAV clusters.

Academic Editor: Eugene Levner

Received: 5 January 2026

Revised: 8 February 2026

Accepted: 13 February 2026

Published: 15 February 2026

**Copyright:** © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

**Keywords:** UAV cluster; simultaneous arrival; spatiotemporal synergy; NSGA-III

---

## 1. Introduction

With the rapid advancement of UAV technology, their ability to perform various tasks such as environmental monitoring, security operations, and express delivery has significantly improved [1–3]. However, as the demand for UAV applications becomes increasingly diverse and complex, issues such as limited efficiency, low flexibility, and reduced reliability in the task execution of a single UAV have become more pronounced. This highlights the importance and value of continued research aimed at enhancing UAV capabilities. From a cost-effectiveness perspective, multi-UAV cooperative operations offer a promising solution to address the limitations associated with single UAV operations. As a result, cooperative multi-UAV operations have emerged as a key focus in the research and application of unmanned systems.

To realize effective cooperative operations of multiple UAVs, the support of core space–time coordination technologies is essential [4,5]. In particular, synchronous arrival, a crucial component of space–time coordination, has found widespread applications in fields such as disaster relief, environmental monitoring, agricultural security, surveillance, and logistics [6–11]. Compared to single UAVs, multi-UAV systems demonstrate faster response times, as well as improved mission efficiency and stability when executing tasks. Additionally, by facilitating mutual backup, multi-UAV systems enhance the reliability and robustness of task execution, particularly in complex environments.

In an effort to optimize the space–time coordination of multi-UAV systems, extensive research on synchronous arrival technology has led to substantial advancements. These efforts can be categorized into three primary directions.

The first category of research focuses on ensuring the real-time stability of multi-UAV systems through the design of control strategies [12–17]. J. Pan et al. [12] explored time-varying formation tracking under external disturbances, providing both sufficient and necessary conditions for system stability based on Euler-Lagrange systems. K.-X. Yan et al. [13] proposed a novel formation control algorithm to address time-varying formation tracking using distributed adaptive methods, achieving minimal energy consumption. The convergence of this method was demonstrated through linear matrix inequality (LMI) techniques and Lyapunov theory. X. Zhang et al. [14] developed an adaptive control-based time-varying tracking protocol, which effectively addresses the challenges of distributed adaptive time-varying formation tracking. X. Zhang et al. [15] introduced an integrated control structure for a leader-follower formation to ensure the stability of the follower in maintaining the leader's formation. To improve control in nonlinear systems, H.-Q. Hou et al. [16] proposed an adaptive fuzzy fixed-time varying formation control (TVFC) method, utilizing fuzzy logic to address nonlinear system constraints, thus enhancing convergence speed and reducing dependence on initial conditions. Stability was proven using fixed-time stability theory. Finally, B.S. Park et al. [17] examined target tracking control for quadrotor UAVs in environments with moving obstacles, offering a control strategy and proving system convergence using Lyapunov stability theory.

The second category focuses on designing spatio-temporal coordination control methods for emergency situations [18–27]. In scenarios with non-autonomous leaders and adaptive followers, J. Li et al. [18] proposed a resilient scheme involving attack detection, isolation strategies, and a formation tracking protocol to ensure the maintenance of a desired time-varying formation. Stability was proven using the impulsive Lyapunov function. In situations where significant communication and control delays occur between system members, X.-J. Peng et al. [19] developed a robust control methodology to mitigate

the impact of external disturbances and time-varying delays on formation. Y. Zhao et al. [20] addressed cases where communication between leaders and followers is hindered by system interference, constructing a distributed consensus error dynamic system and introducing an event-triggered time-varying formation (BTVF) control protocol to reduce external interference and maintain system stability. J. Li et al. [21] also investigated time-varying formation control under network attacks and ineffective information transmission, proposing a resilient time-varying output formation (RTVOF) tracking scheme. Feasible conditions for the control system were provided, with stability proven through graph theory. X. Liu et al. [22] designed a distributed control protocol for robust  $H^\infty$  time-varying formation control in the presence of external disturbances and joint topology, with feasibility demonstrated through Laplacian matrix decomposition and the Cauchy convergence criterion. X. Wu et al. [23] introduced a fault-tolerant time-varying formation control protocol to address bias faults, loss of effectiveness faults, and unknown control inputs, along with an optimization scheme for controller parameters, proving its feasibility via Lyapunov theory. M. Abbasi et al. [24] further investigated system stability under internal communication time delays in multi-agent systems, proposing a method for each agent to estimate its state based on its own state and that of neighboring agents, thereby reducing communication within the system and alleviating the impact of communication time delays. Event-triggering parameters were designed to ensure system stability. J. He et al. [25] proposed an adaptive time-varying group formation tracking protocol, employing multiple asynchronous dynamic edge-event-triggered mechanisms to enhance control of complex systems and improve network communication efficiency. X. Wang et al. [26] developed a distributed fixed-time nonsingular terminal sliding mode (FxTNTSM) control scheme for autonomous surface vehicles (ASVs) to address environmental interference and uncertain system parameters, ensuring the system's approach to a stable state within a fixed time. J. Li et al. [27] further studied control strategies for multiple autonomous underwater vehicles (AUVs) in the presence of positional interference, communication delays, and time-varying disturbances.

The third category focuses on UAV route planning methods [28–31], which can be classified into three modes: distance planning, speed planning, and joint distance-speed planning. In distance planning, it is assumed that each UAV maintains a constant flight speed. The objective is to ensure that each UAV follows a path of equal length, thereby enabling all UAVs to arrive simultaneously. Ismail A et al. [28] introduced a path growth algorithm based on the Dubins method, calculating the longest Dubins path for each UAV and equalizing the path lengths through the adjustment of turning arc radii. Yao W.R. et al. [29] developed a path planning method for multi-UAV rendezvous based on Dubins paths, providing conditions for UAV circling and turning, as well as calculating the waiting time for the first-arriving UAV to ensure simultaneous arrival at the mission node. In speed planning, the flight speed of each UAV is adjusted to achieve simultaneous arrival, taking into account allowable speed ranges and safety constraints. McLain T.W. et al. [30] utilized Voronoi diagrams to generate UAV paths and designed a coordination function to optimize and adjust flight speeds. L. Zhai et al. [31] summarize the typical process of route planning from four aspects: battlefield environment modeling, constraint formula and objective function, intelligent optimization algorithm, and collaborative path optimization. The method of optimizing the flight path using the wolf swarm algorithm is given. Manathara J.G. et al. [32] enhanced speed regulation methods based on the principle of consistency, reducing the likelihood of speed saturation. Joint distance-speed planning optimizes both the path and speed to ensure simultaneous arrival. Choe R. et al. [33] employed pH curves to construct UAV tracks and established a timing law to describe the differential relationship between path and time, adjusting UAV speeds accordingly.

The aforementioned research provides valuable insights and serves as a foundation for advancing the development of multi-UAV cooperative control and simultaneous arrival technologies. However, there are several limitations from both technical and application perspectives. From a technical standpoint, there is a lack of integration between control methods and mission objectives. While existing control methods have primarily focused on maintaining stability, less attention has been given to path optimization and space–time coordination. From an application perspective, some studies prioritize achieving optimal trajectories—such as minimizing threat levels or energy consumption—yet pay insufficient attention to non-optimal objectives, such as time coordination.

To address these gaps, this paper reformulates simultaneous arrival and path optimization as a multi-objective optimization problem. We then enhance the Non-dominated Sorting Genetic Algorithm III (NSGA-III) using a bee colony algorithm to optimize both the Estimated Time of Arrival (ETA) and the UAV path.

The main contributions of this paper are as follows:

- (1) An arrival moment prediction model has been constructed, enabling the simultaneous arrival of drone swarms.
- (2) A new path generation framework has been designed, enabling the optimization of trajectories arriving simultaneously.
- (3) Improved the artificial bee colony algorithm and constructed a route planning process method based on bee colony optimization improved NSGA III, enhancing the efficiency of route planning.

The structure of the paper is organized as follows: Section 2 introduces the problem to be addressed and the basic model. Sections 3 and 4 present the new optimization framework and the objective functions to be optimized, respectively. In Section 5, we describe the multi-objective optimization method, in which NSGA-III is enhanced using the bee colony algorithm. Section 6 outlines the UAV cluster route planning method to achieve simultaneous arrival. In Section 7, we provide simulation verification and comparisons. Finally, the conclusions are presented in Section 8.

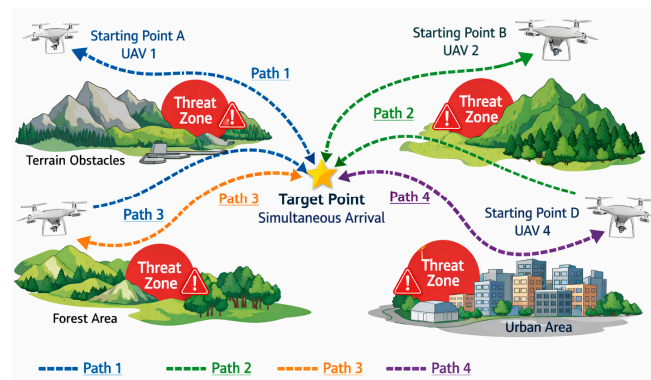
## 2. Simultaneous Arrival Problem Description and Modeling

### 2.1. Problem Description

The problem of cooperative path planning for a UAV cluster, where all UAVs are required to arrive at the target simultaneously, involves determining the flight paths of each UAV, each originating from different starting points, so that they converge at the target location at the same time, both spatially and temporally. Within the framework of collaborative operation, this problem is formulated as a nonlinear, constrained optimization problem, characterized by a set of optimization objectives and constraints. The primary goal of the optimization is to minimize the overall flight cost of the UAV cluster, subject to the condition that all UAVs arrive at the target simultaneously. Additionally, various constraints, including avoidance of external threats, prevention of intra-cluster collisions, and adherence to UAV-specific flight limitations, must be satisfied.

In particular, the requirement for simultaneous arrival necessitates that the UAVs, each departing from different locations, traverse complex and potentially hazardous areas to reach their destination at the same time. The flight environment is often characterized by terrain obstacles and sources of threat, which require careful route planning to ensure that each UAV avoids these risks effectively. Moreover, the model must account for UAV performance characteristics, such as relative flight altitude, fuel constraints, and maximum turning angles, to ensure flight safety. The planned routes must guarantee the safety of the UAVs throughout the flight. To ensure safe operation, a collision avoidance mechanism is implemented to maintain a safe separation distance between any pair of UAVs. Additionally, the flight speed of each UAV is adjusted according to the planned path to

ensure synchronized arrival at the target. A route planning diagram illustrating the UAV cluster's simultaneous arrival is shown in Figure 1.



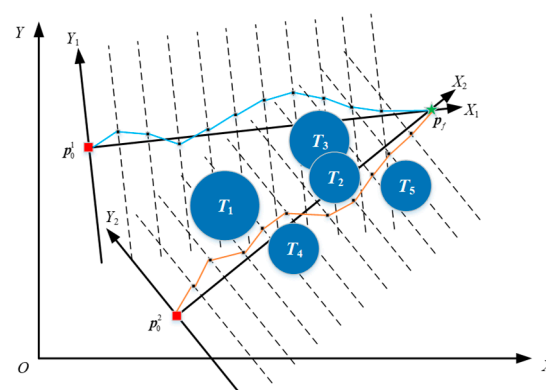
**Figure 1.** Schematic diagram of route planning for simultaneous arrival of UAV clusters.

The flight area consists of four UAVs, multiple threat zones, and terrain obstacles. Assuming a fixed flight area with static and known obstacles and threat zones, the objective of the route planning is to determine feasible paths for each UAV that enable all to reach the predetermined destination simultaneously. Each path must avoid entering threat zones and terrain obstacles while minimizing the overall flight cost. Consequently, the primary challenge lies in accurately modeling the flight paths and the interrelationships among the various UAV trajectories throughout the flight process.

In summary, the essence of the problem of multiple drones arriving simultaneously is an optimization problem under constraints. The objective function is to minimize the penalties imposed by terrain and threat sources, as well as the costs incurred by turning, fuel consumption, and climbing, during the flight of each drone. Additionally, constraints such as simultaneous arrival, collision avoidance between drones, and obstacle avoidance must be met. This optimizes a path that satisfies the constraints and minimizes the total cost of all drones. This is the core problem to be solved in drone swarm path optimization. The penalty and cost functions will be detailed in Section 4. Next, this article will introduce the basic model of path optimization.

## 2.2. Basic Model

The path planning problem of a UAV can be simplified to the path point generation problem, as shown in Figure 2.



**Figure 2.** Schematic diagram of track point generation.

Point  $P_0^i$  in Figure 2 represents the starting location of the  $i$ -th UAV, where  $i = 1, 2, \dots, N$ .  $p_f$  denotes the predetermined target point. In the simultaneous arrival problem, the target point for all UAVs is identical, meaning that:

$$\forall i \in N, \quad p_f^i = p_f \quad (1)$$

$T_j$  represents the central coordinates of the  $j$ -th threat area. As a UAV approaches the center of a threat area, its path is influenced by the proximity to the threat, with the impact increasing as the distance decreases. The objective of the route planning is to determine the optimal or suboptimal flight path for each UAV within the flight area, ensuring that all UAVs reach the target point  $p_f$  at the same time, while minimizing the overall flight cost, avoiding collisions, and evading threat zones.

### 3. New Track Generation Architecture

Route planning is typically modeled as a continuous-time optimization problem. In this context, after defining the objective function and constraints, the path is progressively generated towards the target point using an optimization-based recursive method. While this approach guarantees that the generated path is optimal, it also leads to a large search space. Furthermore, if a UAV's path becomes trapped within a threat area, the entire optimization process can experience significant delays.

To address this challenge, this paper proposes a route planning method based on a “benchmark” approach. As illustrated in Figure 2, each UAV has an ideal shortest flight path  $L_i = P_0^i p_f$ , represented by the direct line between the starting point and the target point. However, it is evident that if this direct path  $L_i$  is followed, the UAVs will likely traverse obstacle and threat areas, and the arrival moment of the UAVs will vary. This outcome is inconsistent with the objective of synchronized arrival for all UAVs.

Despite this, the shortest flight path provides valuable reference information. Regardless of the route planning method employed, if the shortest path or the shortest time is incorporated into the optimization objective function, the planned route will inherently resemble  $L_i$ . Consequently, this paper suggests using the shortest flight path as a “reference” for subsequent route planning. The purpose of introducing this “reference” is to guide the UAVs along the shortest path as much as possible. When encountering obstacles, the path can be adjusted by extrapolating to suitable points along the route. The degree of extrapolation will vary depending on the required arrival moment, ultimately ensuring that the UAV cluster reaches the destination simultaneously, while minimizing the overall flight cost.

The concept of extrapolation is illustrated in Figure 2. For the  $i$ -th UAV, along its shortest flight path  $L_i$ ,  $M$  straight lines are inserted perpendicular to  $L_i$  at uniform inter-

vals, and these lines are recorded as a set  $V^{(i)} = \{V_1^{(i)}, \dots, V_M^{(i)}\}$ .

Although the shortest flight paths  $L_i$  and  $L_j$  differ for any two UAVs, for the sake of simplicity in calculation and expression, this paper assumes that the number of segments,  $M$ , is the same for all paths. For any UAV, the corresponding flight point  $P_m^i$  related to  $V_m^{(i)}$  can be determined through planning. Specifically, this point represents the  $m$ -th way-point along the path of UAV $i$ . Accordingly, the path of the  $i$ -th UAV is denoted as  $P_i = \{P_0^{(i)}, P_1^{(i)}, \dots, P_M^{(i)}, p_f\}$ .

To improve the expression and optimize the computational speed of the algorithm, a coordinate transformation is applied in this paper. Let the body coordinate system of the  $i$ -th UAV be denoted as  $X_i O Y_i$ , where the origin  $O$  is the center of gravity of the UAV, the nose of the UAV points along the positive  $X$ -axis, and the direction perpendicular to

the plane of the fuselage defines the positive Z-axis. The Y-axis direction is determined by the right-hand rule.

After the transformation, for UAV<sub>i</sub>, the coordinate origin is shifted to  $p_0^i$ , and the direction connecting  $p_0^i$  to the target point  $p_f$  defines the positive X-axis. The Z-axis is aligned with the height direction in physical space, opposite to the direction of gravity. The Y-axis is determined according to the right-hand rule.

The angle  $\theta_i$  between the new coordinate system and the global coordinate system, specifically the angle between the path  $L_i$  and the OX, represents the rotation angle required for the transformation between the two coordinate systems. The geometric relationship between the global coordinate system  $X_T O Y_T$  and the body coordinate system  $X_i O Y_i$  for the same track point is described by Formula (2).

$$\begin{bmatrix} x_T^i \\ y_T^i \\ z_T^i \end{bmatrix} = \begin{bmatrix} \cos \theta_i & \sin \theta_i & 0 \\ -\sin \theta_i & \cos \theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^i - x_0^i \\ y^i - y_0^i \\ z^i - z_0^i \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ z_0^i \end{bmatrix} \quad (2)$$

Let  $(x_0^i, y_0^i, z_0^i)$  denote the starting point  $p_0^i$  of UAV<sub>i</sub> in the global coordinate system. The coordinates  $(x^i, y^i, z^i)$  and  $(x_T^i, y_T^i, z_T^i)$  represent the original and transformed coordinates, respectively, of the same track point in the two coordinate systems. The horizontal coordinate  $x_j^i$  of the  $j$ -th track point can then be expressed as:

$$x_j^i = \frac{j}{M+1} \times \|p_0^i p_f\| + x_{p_0^i} \quad (3)$$

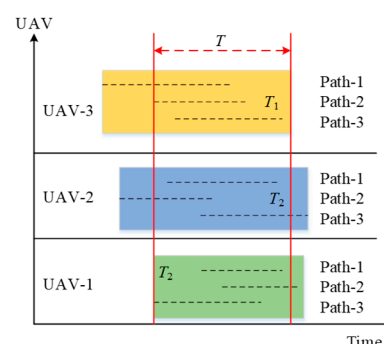
Here,  $\| \cdot \|$  represents the 2-norm. By using this simplified representation, one dimension can be reduced, thereby decreasing the dimensionality of the optimization space and conserving computational resources.

## 4. Objective Function of Track Optimization

To optimize the trajectories of multiple UAVs arriving simultaneously, it is essential to quantify the flight time of each UAV, along with other objective functions to be optimized. Therefore, this section introduces several factors that must be comprehensively considered in the route planning process from various perspectives.

### 4.1. The Method of Time Coordination

To ensure that each UAV reaches its designated location at the scheduled time, it is essential for each UAV to consume the same amount of time during its journey. At the same time, the total flight time of the UAVs should be minimized to reduce the accumulated risk in the flight area and conserve energy. Therefore, this section introduces the concept of the Ideal Feasible Arrival moment (IFAT) for UAVs, as illustrated in Figure 3.



**Figure 3.** Schematic diagram of IFAT.

For the  $i$ -th UAV, its speed is constrained within the range  $[v_{i,\min}, v_{i,\max}]$ . While an infinite number of paths exist for the UAV to reach its destination, considering factors such as path length, threat levels, fuel consumption, and others, a finite set of  $Q$  feasible paths is identified. Let the length of route  $q$  be  $D_q$ ; the UAV's travel time  $t_q^i$  on this route must satisfy the following conditions:

$$t_q^i \in \left[ \frac{D_q}{v_{i,\max}}, \frac{D_q}{v_{i,\min}} \right] \quad (4)$$

Therefore, for UAV $_i$  with  $Q$  feasible paths, its IFAT  $t_{IFAT}^i$  should be defined as the union of the arrival moments for all paths:

$$t_{IFAT}^i = \bigcup_{k \in K} t_k^i \quad (5)$$

In Figure 3, the rectangular box, colored in yellow, blue, or green, represents the possible times for the corresponding path, with the dotted line indicating the potential time intervals. After performing the intersection operation, the rectangular box represents the IFAT  $t_{IFAT}^i$  for the corresponding UAV.

Furthermore, as shown in Figure 3, for a cluster of  $n$  UAVs, the Simultaneous Arrival moment (SAT), denoted as  $T_{SAT}$ , is defined as the intersection of all UAVs' IFATs, as follows:

$$T_{SAT} = \bigcap_{i \in N} t_{IFAT}^i \quad (6)$$

The corresponding value is represented by the interval between the two red lines in Figure 3. It is evident that  $T_{SAT}$  represents the collection of arrival moments for each drone at the mission point. For ease of subsequent calculation and optimization,  $t_{SAT}$  is set as the specific simultaneous arrival moment to be optimized in this paper.

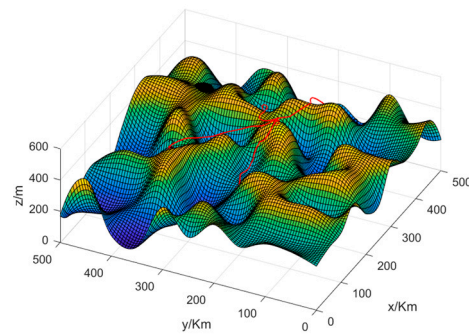
From the perspectives of efficiency, cost-effectiveness, and safety, a shorter simultaneous arrival moment for the UAVs is preferred. To this end, we define the first objective function as follows:

$$J_1 = \min(t_{SAT}) \quad (7)$$

Although it can be seen from Figure 3 and Formula (6) that the constraints on simultaneous arrival moment in this paper are relatively loose, there is still a possibility that the result of Formula (6) is an empty set. That is, it is impossible for the drones to arrive simultaneously.

At this point, it is necessary to adjust the selection of the objective function in Formula (7). That is, the time for the last drone to arrive should be as short as possible. In this way, the drones that arrive earlier can coordinate with the drones that arrive later by hovering and waiting, ultimately arriving at the task area simultaneously.

However, the above analysis is based on a global planning approach from the perspective of a single UAV, where the time required for each UAV's journey is calculated independently, without considering the influence of other UAVs. A typical scenario is illustrated in Figure 4.



**Figure 4.** Schematic diagram of route planning without mutual interference.

In Figure 4, if each UAV takes off from positions that are nearly completely separated, and their flight paths are sufficiently distant from one another, the overall cluster path planning can be treated as the independent planning of each UAV, step by step. However, when the flight paths of the UAVs are in close proximity or when there is mutual interference between the paths, it becomes necessary to adjust the time requirements to ensure the feasibility of the planning.

#### 4.2. Penalty Function for Threat Penalty

##### 4.2.1. Penalty Function for Terrain Obstacles

The feasible path of the UAV must avoid passing through mountains or no-fly zone obstacles [34], and it should prevent collisions with terrain features. In path planning, it is assumed that all obstacle information is known. Let  $h()$  represent the obstacle function. For a given coordinate point  $(x_{m,n}, y_{m,n})$ , the corresponding elevation value  $h(x_{m,n}, y_{m,n})$  at that point can be obtained from the obstacle function. Geographic elevation data is typically provided in matrix form, which prevents the direct retrieval of elevation values for any arbitrary point in continuous space. To address this, cubic spline interpolation can be used to approximate the elevation values.

Subsequently, the relative distance between the  $k$ -th path point  $z_k^i(x_{m,n}, y_{m,n})$  of the  $i$ -th UAV and the corresponding elevation value will be determined. If the elevation does not meet the obstacle avoidance requirements, this point will be penalized in the objective function. The corresponding cost function is then formulated as:

$$A_k^i = \begin{cases} 1 & \text{if } z_k^i(x_{m,n}, y_{m,n}) \leq h(x_{m,n}, y_{m,n}) \\ 0 & \text{others} \end{cases} \quad (8)$$

For no-fly zones, it is sufficient to assess whether the UAV's path point lies within the boundaries of the no-fly zone  $\Omega$ . The corresponding penalty function is expressed as:

$$C_k^i = \begin{cases} 1 & \text{if } z_k^i(x_{m,n}, y_{m,n}) \in \Omega \\ 0 & \text{others} \end{cases} \quad (9)$$

For a UAV with  $K$  path points, the penalty function for terrain obstacles can be expressed as:

$$J_2 = \min \sum_{i=1}^N \sum_{k=1}^K (A_k^i + C_k^i) \quad (10)$$

##### 4.2.2. Penalty Function for Threat Sources

During flight, UAVs must not only avoid physical obstacles, such as mountains, but also mitigate risks from other threats, including cloud clusters, interference fields, and

areas with weak signal coverage that can compromise flight safety. Unlike mountainous regions, where drones face immediate hazards like collisions, the threats posed by these areas do not result in direct collisions but can still present significant risks. For example, drones may become disoriented when navigating through cloud clusters or may lose communication in areas with strong electromagnetic interference or weak signals. Given the inherent probability of these events occurring and considering the time-sensitive nature of certain missions, drones may need to traverse these threat areas. For instance, during earthquake rescue operations, drones may pass through cloud clusters or areas with damaged signal infrastructure to ensure a timely response, despite the potential risks.

In this context, the coverage area of a threat source is modeled as a cylinder, with the  $p$ -th threat source represented by the vector  $T_p = [x_{Tp}, y_{Tp}, h_{Tp}, r_{Tp}, L_{Tp}]^T$ , where  $(x_{Tp}, y_{Tp})$  denotes the coordinate position of the threat source in the global coordinate system;  $h_{Tp}$  represents the coverage height range of the threat area  $T_j$ ;  $r_{Tp}$  is the coverage radius of the threat area  $T_j$ ; and  $L_{Tp}$  indicates the threat coefficient of the area, with larger values of  $L_{Tp}$  corresponding to a higher threat level and increased likelihood of drone failure.

Assuming that the  $k$ -th path point  $(x_k^i, y_k^i, z_k^i)$  of UAV $_i$  falls within threat area  $T_j$  the corresponding penalty function can be expressed as:

$$R_k^i = \begin{cases} \frac{L_{Tj}}{(D_{kp}^i)^2} & \text{if } z_k^i \leq h_{Tp}, D_{kp}^i \leq r_{Tp} \\ 0 & \text{others} \end{cases} \quad (11)$$

Among them,  $D_{kp}^i = \sqrt{(x_k^i - x_{Tp})^2 + (y_k^i - y_{Tp})^2}$  represents the three-dimensional distance from the path point to the center of the threat source.  $L_{Tj}$  is a parameter set by path planners based on the specific conditions and will not be deployed in this instance.

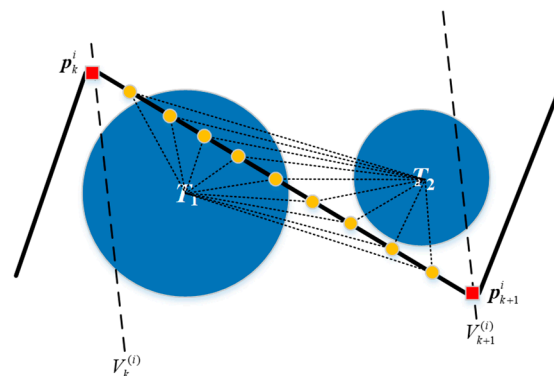
For a UAV cluster, the penalty function for threat sources is expressed as:

$$J_3 = \min \sum_{i=1}^N \sum_{k=1}^K (R_k^i) \quad (12)$$

#### 4.2.3. Improved Penalty Function

By employing the aforementioned constraint penalty method, it is possible to ensure that all path points avoid threats. However, certain special situations may arise.

Specifically, since path planning generates discrete points, in cases where the task area is large or time constraints are tight, the spatial distribution between the planned path points may become sparse, as illustrated in Figure 5.



**Figure 5.** Schematic diagram for calculating threat cost.

As shown in Figure 5, the planned path points  $P_k^i$  and  $P_{k+1}^i$  are both located outside the threat area. However, the line connecting these points, representing the UAV's route from  $P_k^i$  to  $P_{k+1}^i$ , passes through the threat area. This will result in two situations:

The first scenario involves the connection line traversing through a terrain obstacle zone, meaning neither of the two points encounters any obstacles, yet the drone encounters a mountain collision. In this case, the shortest distance  $d_{pmin}$  between the obstacle and the  $P_k^i P_{k+1}^i$  connection line can be calculated. Subsequently, a comparison can be made between this distance and the radius  $r_{mon}$  of the mountain. If the linear distance is less than the radius of the mountain, it indicates that the drone will collide with the mountain. Therefore, this path is infeasible. Otherwise, there is no issue.

The second scenario involves the drone traversing through a threat source area. To address this, this section proposes using a uniform sampling method to further assess and refine the threat level along the path.

As illustrated in Figure 5, if  $L$  yellow sampling points are used, the segment  $P_k^i P_{k+1}^i$  is divided into  $L + 1$  smaller segments. Let the  $l$ -th sampling point be denoted as  $P_{l/L}$ , where  $l = 1, 2, \dots, L$ . The sampling points are set to satisfy the following criteria:

$$2|P_k^i P_{1/L}| = |P_{1/L} P_{2/L}| = \dots = |P_{(L-1)/L} P_{L/L}| = 2|P_{L/L} P_{k+1}^i| \quad (13)$$

Once the sampling points are determined, the threat cost for the  $i$ -th UAV in the threat area between the  $k$ -th and  $(k + 1)$ -th path points can be calculated using the following formula:

$$J_k^i(k, k+1) = \frac{\|P_k^i P_{k+1}^i\|}{L} \sum_{j=1}^J \sum_{m=1}^{L+1} L_{Tj} \left( \frac{1}{d_{1/L,j}} + \dots + \frac{1}{d_{L/L,j}} \right) \quad (14)$$

In this context,  $\|P_k^i P_{k+1}^i\|$  represents the length of the flight segment  $P_k^i P_{k+1}^i$ ;  $L$  is the number of sampling points along the flight segment  $P_k^i P_{k+1}^i$ ;  $L_{Tj}$  is the threat coefficient of the threat area  $T_j$ ;  $d_{l/L,j}$  denotes the distance between the  $l$ -th sampling point and the center of the threat area; and  $J$  is the number of potential threat sources within the flight segment  $P_k^i P_{k+1}^i$ . By using Equations (13) and (14), the threat costs for all points along the path segment can be summed, and the total threat cost for the  $i$ -th UAV in the threat area is calculated as follows:

$$J_4^i = \sum_{k=0}^{K-1} J_k^i(k, k+1) \quad (15)$$

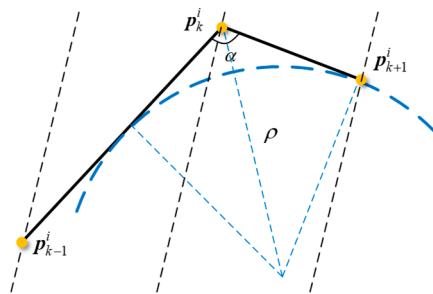
The path cost for the UAV swarm can be expressed as:

$$J_4 = \min \sum_{i=1}^N J_4^i \quad (16)$$

### 4.3. Cost Function for Resource Consumption

#### 4.3.1. Cost Function for Turning

The maneuverability of UAVs is constrained by their maximum turning angle. Due to these limitations, paths with large turning angles may not be feasible for the UAVs. In theory, the smaller the turning angle, the smoother the path, which in turn enhances the UAV's flight performance. Therefore, to minimize both the number and severity of turns, this section proposes an optimization objective that satisfies the planned turning constraints while minimizing the turning angle. The turning process is illustrated in Figure 6.



**Figure 6.** Schematic diagram of the turning process.

As shown in Figure 6, for the  $i$ -th UAV, the turning process occurs between way-points  $p_{k-1}^i p_k^i p_{k+1}^i$ . These three path points, as depicted in Figure 6, define a turning process. The turning angle  $\alpha$  is defined as the residual angle between adjacent flight segments. The turning angle  $\alpha$  can then be calculated using the law of cosines, as follows:

$$\alpha(p_{k-1}^i, p_k^i, p_{k+1}^i) = \arccos \frac{\|p_{k-1}^i p_k^i\|^2 + \|p_k^i p_{k+1}^i\|^2 - \|p_{k-1}^i p_{k+1}^i\|^2}{2 \|p_{k-1}^i p_k^i\| \cdot \|p_k^i p_{k+1}^i\|} \quad (17)$$

To ensure that the UAV can complete the turn smoothly while adhering to the turning constraints, the radius of the maximum inscribed circle formed by adjacent segments must be smaller than the fixed turning radius constraint  $R_t$ . The radius of the maximum inscribed circle, denoted as  $\rho$ , is given by:

$$\rho(p_{k-1}^i, p_k^i, p_{k+1}^i) = \tan \frac{\alpha(p_{k-1}^i, p_k^i, p_{k+1}^i)}{2} \times \min \{\|p_{k-1}^i p_k^i\|, \|p_k^i p_{k+1}^i\|\} \quad (18)$$

When the turning constraint for the UAV is satisfied, i.e.,  $\rho < R_t$ , the turning angles  $\pi - \alpha$  can be accumulated to achieve the smallest possible turning angle. When  $\rho \geq R_t$ , the turning angle  $\pi - \alpha$  is multiplied by a penalty coefficient. The final optimization objective function for the turning constraints of the UAV is expressed as follows:

$$J_5 = \min \sum_{i=1}^N \sum_{k=0}^{K-1} \mu [\pi - \alpha(p_{k-1}^i, p_k^i, p_{k+1}^i)] \quad (19)$$

where

$$\mu = \begin{cases} 1 & \text{if } \rho(p_{k-1}^i, p_k^i, p_{k+1}^i) < R_t \\ \rho(p_{k-1}^i, p_k^i, p_{k+1}^i) - R_t & \text{others} \end{cases} \quad (20)$$

The advantage of using  $\rho - R_t$  as the adaptive penalty coefficient is that the penalty cost is directly proportional to the degree of deviation from the optimal turning angle. Specifically, the larger the turning radius of the path, the higher the penalty cost, which reduces the likelihood of selecting such a path in the optimization process. This approach ensures that the path with the fewest and smallest turns is prioritized and optimized to the greatest extent possible.

#### 4.3.2. Cost Function for Flight Fuel and Path Consumption

To ensure both redundancy and cost-effectiveness in completing the task, the UAV should minimize fuel consumption during its flight. Fuel consumption is positively correlated with the length of the flight path. Therefore, this section utilizes two parameters to represent fuel consumption: the total path length and the distance from the current path point to the destination.

Under identical conditions, the shorter the path, the lower the fuel consumption and the shorter the UAV's flight time. By minimizing the UAV's exposure time in flight, the probability of detection by unknown threats in complex environments can also be reduced. Consequently, the cost function  $J_{dis}$  for the  $i$ -th UAV flying along the designated route can be expressed as:

$$J_{dis}(i) = \frac{\sum_{k=1}^{K-1} \sqrt{(x_{k+1}^i - x_k^i)^2 + (y_{k+1}^i - y_k^i)^2 + (z_{k+1}^i - z_k^i)^2}}{\sqrt{(x_0^i - x_f)^2 + (y_0^i - y_f)^2 + (z_0^i - z_f)^2}} \quad (21)$$

where  $(x_0^i, y_0^i, z_0^i)$  and  $(x_f, y_f, z_f)$  represent the start and end points of the flight path, respectively.

Additionally, this paper calculates the distance from each path point to the destination to assess the UAV's current position. For any UAV, multiple feasible path points exist along each vertical line. During the optimization process, the distance to the destination plays a crucial role. Therefore, the search direction in the path planning algorithm can be determined, significantly reducing computation time and minimizing fuel consumption. This approach ensures that the UAV prioritizes path points closer to the destination.

Let the distance from the  $k$ -th path point of the  $i$ -th UAV to the destination be denoted as  $d_i(k)$ . The average distance of all feasible path points along the path is then calculated, resulting in the cost function  $J_{Tof}$  for the path, which is expressed as:

$$J_{Tof}(i) = \frac{\sum_{k=1}^{K-1} \min[d_i(k)]}{K} \quad (22)$$

To summarize, the fuel consumption cost  $J_6$  for the UAV can be expressed as:

$$J_6 = \min \sum_{i=1}^N (J_{dis}(i) + J_{Tof}(i)) \quad (23)$$

#### 4.3.3. Cost Function for Inter-Aircraft Collision

When planning the flight paths for a UAV cluster, it is essential to verify whether any dangerous approach occurs between two UAVs and to ensure that there is no conflict in both spatial and temporal dimensions.

For any two UAVs, UAV<sub>*i*</sub> and UAV<sub>*j*</sub>, the path points of both flight paths must be checked pairwise. If, within a given time interval, the distance  $d_{i,j}(n,m)$  between the  $m$ -th path point of UAV<sub>*i*</sub> and the  $n$ -th path point of UAV<sub>*j*</sub> is less than the allowable safe distance  $d_{safe}$ , it can be concluded that a collision risk exists between the two UAVs. Thus, both distance and time are critical parameters in determining the collision risk:

$$\begin{cases} d_{i,j}(m,n) = \sqrt{(x_i(m) - x_j(n))^2 + (y_i(m) - y_j(n))^2 + (z_i(m) - z_j(n))^2} \\ t_{i,j}(m,n) = |t_i(m) - t_j(n)| \end{cases} \quad (24)$$

In practice, two parameters, the safety distance  $d_{safe}$  and the time interval  $t_{safe}$ , are artificially defined as safety margins. If the distance or time between the two UAVs falls below these preset values, the corresponding feasible paths of the UAVs will incur a penalty, as follows:

$$J_{coll}(i,j) = \begin{cases} \sum_m \sum_n \left( \frac{1}{d_{i,j}(m,n)} + \frac{1}{t_{i,j}(m,n)} \right), & d_{i,j}(m,n) < d_{safe} \quad \text{and} \quad t_{safe} < t_{i,j}(m,n) \\ 0, & \text{others} \end{cases} \quad (25)$$

The collision cost function  $J_7$  can then be expressed as:

$$J_7 = \min \sum_i \sum_{j \neq i} J_{coll}(i, j) \quad (26)$$

#### 4.3.4. Cost Function for Flight Altitude

While flying at higher altitudes can reduce fuel consumption, it introduces several challenges. As the flight altitude increases, the wind speed affecting the UAV's flight stability also rises, and the ambient temperature decreases, both of which pose greater risks to the UAV's flight control stability. Additionally, higher altitudes result in an extended signal transmission range, increasing the likelihood of the UAV losing contact. Therefore, this paper argues that the UAV's flight altitude should be kept relatively low.

However, due to fluctuations in ground elevation, the UAV requires sufficient altitude to adjust its motion state in advance. As a result, the flight altitude must exceed the minimum safe flight altitude  $H_{safe}$ .

In summary, for the  $k$ -th path point of the  $i$ -th UAV, the cost function  $JH_i(k)$  is expressed as:

$$JH_i(k) = \begin{cases} z_i(k) - f(x_i(k), y_i(k)) - H_{safe}, & \text{if } z_i(k) \geq f(x_i(k), y_i(k)) + H_{safe} \\ JH_{max}, & \text{others} \end{cases} \quad (27)$$

where  $f(x_i(k), y_i(k))$  represents the geographic elevation information corresponding to the  $k$ -th point of the UAV's path, and  $JH_{max}$  denotes the penalty value. Therefore, the objective function  $J_8$  is expressed as:

$$J_8 = \min \sum_{i=1}^N \sum_{k=1}^K JH_i(k) \quad (28)$$

#### 4.4. Objective Function to Be Optimized

The above is the objective function to be optimized, which is:

$$J = \begin{cases} \min J_1 \\ \min J_2 \\ \dots \\ \min J_8 \end{cases} \quad (29)$$

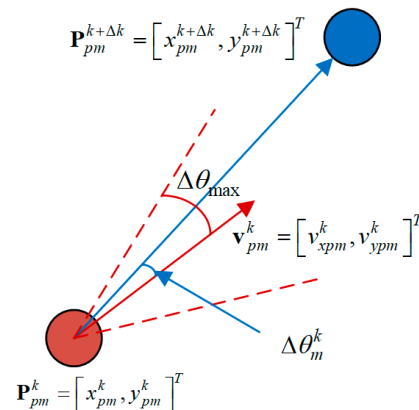
This paper employs the optimization strategy outlined in the previous section to optimize Equation (29). Specifically, the optimal cluster flight path is determined by optimizing the longitudinal deviation and flight altitude of each UAV along the vertical line in Figure 2.

As indicated by Equation (29), this represents a typical multi-objective optimization problem. A common approach for solving such problems is weighted summation; however, this method is highly dependent on the selection of appropriate weights. To overcome this limitation, this paper utilizes the improved Non-dominated Sorting Genetic Algorithm III (NSGA-III) to simultaneously optimize multiple objective functions.

#### 4.5. Constraints

In addition to the constraints mentioned above that are converted into objective functions, the dynamic motion constraints and communication constraints of the swarm also need to be considered during the operation of the UAV.

It was assumed that the motion state of the UAV is at  $k$  and the next moment, that is, the motion state at the moment  $k + \Delta k$ , as shown in Figure 7.



**Figure 7.** Schematic diagram of motion constraints.

The position and speed of  $m$ -th UAV at time  $k$  are  $\mathbf{P}_m^k = [x_m^k, y_m^k]$  and  $\mathbf{v}_m^k = [v_{xm}^k, v_{ym}^k]$ . Taking it as the initial condition, it was optimized to obtain the position and velocity in the next moment as  $\mathbf{P}_m^{k+\Delta k} = [x_m^{k+\Delta k}, y_m^{k+\Delta k}]$  and  $\mathbf{v}_m^{k+\Delta k} = [v_{xm}^{k+\Delta k}, v_{ym}^{k+\Delta k}]$ , respectively. The corresponding relationship is shown in Figure 7. Then, the motion constraints should be satisfied, namely:

$$\begin{cases} \mathbf{P}_m^{k+\Delta k} = \mathbf{P}_m^k + \mathbf{v}_m^k \Delta k \\ \|\mathbf{P}_m^{k+\Delta k} - \mathbf{P}_m^k\|_2 \leq \|\mathbf{v}_m^k\|_2 \Delta k \end{cases} \quad (30)$$

where  $\|\cdot\|_2$  means to take the 2-norm. The relationship between the speeds is:

$$\mathbf{v}_m^{k+\Delta k} = \mathbf{v}_m^k + \Delta \mathbf{v}_m^k \quad (31)$$

where  $\Delta \mathbf{v}_m^k$  is the value of the velocity change, which should satisfy:

$$\begin{cases} \|\Delta \mathbf{v}_m^k\|_2 \leq \Delta v_{\max} \\ v_{\min} \leq \|\mathbf{v}_m^{k+\Delta k}\|_2 \leq v_{\max} \end{cases} \quad (32)$$

That is, the speed and the amount of speed change cannot exceed their allowable limit.

Similarly, the change amount  $\Delta \theta_m^k$  of the UAV direction can be calculated according to the velocity vector at two moments, which should satisfy:

$$|\Delta \theta_m^k| \leq \Delta \theta_{\max} \quad (33)$$

where  $|\cdot|$  represents the absolute value.

The above are the motion constraints that the UAV should meet.

The communication constraints of the cluster require that for any drone, there must be at least one drone within a set communication distance.

This constraint is relatively simple, and will not be elaborated on in this article.

## 5. Improved NSGA-III Based on ABC Algorithm

### 5.1. Basic Principle of NSGA-III

NSGA-III is a multi-objective optimization algorithm designed to address problems involving multiple objective functions. Based on the genetic algorithm, NSGA-III balances the diversity and convergence of solutions through non-dominated sorting and a reference point mechanism. The algorithm employs uniformly distributed reference points in

the objective space to guide the optimization process, ensuring that the solutions are well-distributed across multiple objectives.

Similarly to NSGA-II, NSGA-III uses non-dominated sorting to rank solutions. Solutions are categorized into different fronts based on dominance relationships, with the first front containing the solutions with the best dominance relationships, the second front containing solutions dominated by the first front, and so on. However, unlike NSGA-II, NSGA-III incorporates a reference point mechanism, ensuring the uniform distribution of solutions in the multi-dimensional objective space. This mechanism reduces the likelihood that the search process becomes overly concentrated in any particular region of the objective space.

Let  $n$  be the number of objective functions to be optimized, denoted as  $f_n(x)$ . The reference point selection formula  $r$  is then given by:

$$r = \left( \frac{f_1(x)}{\|f_1\|}, \frac{f_2(x)}{\|f_2\|}, \dots, \frac{f_N(x)}{\|f_N\|} \right) \quad (34)$$

Here,  $\|f\|$  represents the normalized reference point in the objective space, which is used to assist in selecting the solutions. The algorithmic flow of NSGA-III is well-established, and therefore, this paper will not provide a detailed repetition of the procedure.

### 5.2. Shortcomings of NSGA-III

Although NSGA-III retains the multi-objective optimization advantages of NSGA-II, it is particularly well-suited for optimizing objective functions with more than two objectives. However, the method still has certain limitations, primarily reflected in four aspects:

① **High Computational Complexity:** Particularly when the number of objectives is large, the computational overhead associated with non-dominated sorting, congestion calculation, and reference point allocation in NSGA-III becomes substantial. As a result, it may not perform as efficiently as simpler multi-objective algorithms in high-dimensional objective spaces.

② **Reference Point Management:** In high-dimensional spaces, effectively selecting and managing reference points remains a significant challenge. The number of reference points tends to increase sharply with the number of objectives, leading to greater complexity in both calculation and management.

③ **Insufficient Convergence:** While NSGA-III provides a good distribution of solutions in multi-objective optimization, in some high-dimensional problems, the algorithm may struggle to find the global Pareto optimal solution. This can result in certain solutions deviating from the true Pareto front.

④ **Risk of Convergence to Local Optima:** In certain cases, particularly for complex multi-objective problems, NSGA-III may converge to a local optimum and fail to adequately explore the entire search space.

### 5.3. Artificial Bee Colony Algorithm

To address the limitations of NSGA-III, this section employs an improved Artificial Bee Colony (ABC) algorithm to enhance the NSGA-III method.

The ABC algorithm is an optimization algorithm that simulates the foraging behavior of bees. Compared to traditional optimization algorithms such as Particle Swarm Optimization (PSO) and Genetic Algorithm (GA), the ABC algorithm has two significant advantages, in addition to being simple and easy to implement with no complex parameter adjustments required. Firstly, the ABC algorithm boasts strong global search capabilities, effectively avoiding the issue of early convergence. Secondly, the division of labor among bees is well-organized, which can effectively prevent the algorithm from getting stuck in

local optima and enhance the search efficiency for global optimal solutions. The algorithmic flow of the ABC can be described as follows:

Step 1: Population Initialization. The first-generation population of size  $BN \times D$  is generated, where  $BN$  represents the number of individuals in the population, and  $D$  denotes the dimension of the search space.

$$\begin{bmatrix} x_{1,1}^1 & x_{1,2}^1 & \cdots & x_{1,D}^1 \\ x_{2,1}^1 & x_{2,2}^1 & \cdots & x_{2,D}^1 \\ \vdots & \vdots & \ddots & \vdots \\ x_{BN,1}^1 & x_{BN,2}^1 & \cdots & x_{BN,D}^1 \end{bmatrix} \quad (35)$$

For each initial parameter  $x_{i,j}$  in the  $i$ -th solution  $\mathbf{X}_i^1 = [x_{i,1}^1 \ x_{i,2}^1 \ \cdots \ x_{i,D}^1]$ , it can be generated using Equation (36).

$$x_{i,j}^1 = x_{\min,j}^1 + r(x_{\max,j}^1 - x_{\min,j}^1) \quad (36)$$

where  $i = 1, 2, \dots, BN, j = 1, 2, \dots, D$ , and  $x_{\max,j}^1$  and  $x_{\min,j}^1$  represent the upper and lower bounds of the  $j$ -th parameter, respectively.  $r$  is a random number uniformly distributed between 0 and 1.

Step 2: Employed Bees Search. The candidate solution  $v_{i,j}^{n+1}$  for iteration  $n + 1$  is obtained using Equation (37).

$$v_{i,j}^{n+1} = x_{i,j}^n + \phi(x_{i,j}^n - x_{k,j}^n) \quad (37)$$

where  $n$  and  $n + 1$  denote the number of iterations, with  $n = 1, 2, \dots, N$ .  $k$  is a positive integer with a uniformly random value in the range  $[1, BN]$ , and  $\phi$  is a random number uniformly distributed between  $(-1, 1)$ .  $\mathbf{V}_i^{n+1} = [v_{i,1}^{n+1} \ v_{i,2}^{n+1} \ \cdots \ v_{i,D}^{n+1}]$  and  $\mathbf{X}_i^n$  are used to

calculate the corresponding fitness functions  $F(\mathbf{V}_i^{n+1})$  and  $F(\mathbf{X}_i^n)$ , respectively. The better solution and its corresponding parameters are retained for the next iteration.

Step 3: The Guard Bee Chooses the Honey Source. The guard bee employs a roulette wheel selection method to choose the honey source for further exploration. The roulette wheel selection is performed as follows:

$$P_i^n = \frac{F(\mathbf{X}_i^n)}{\sum_{i=1}^{BN} F(\mathbf{X}_i^n)} \quad (38)$$

where  $P_i^n$  represents the probability that the guard bee will choose  $\mathbf{X}_i^n$  after the  $n$ -th search. Clearly, the better the fitness function, the higher the corresponding probability, which facilitates the concentration of resources around the optimal point, leading to improved optimization results.

#### 5.4. Improved NSGA-III Optimization Strategy Based on ABC Algorithm

To address the shortcomings of NSGA-III, this paper enhances the ABC optimization process to mitigate the issues associated with NSGA-III.

To overcome ① the high computational complexity, this section modifies Equation (37) and adjusts it as follows:

$$v_{i,j}^{n+1} = \max \{ x_{i,j}^n + \phi(x_{i,j}^n - x_{k,j}^n), x_{i,j}^n - \phi(x_{i,j}^n - x_{k,j}^n) \} \quad (39)$$

Equation (37) plays a crucial role in directly influencing the optimization efficiency of the algorithm. Since the value of  $\phi$  is random, the ABC algorithm faces challenges in optimizing towards a better direction, which limits its performance. The improved Equation (39) enhances the optimization of the fitness function in a more effective direction. Although the computational cost per iteration increases, it significantly reduces the number of iterations compared to the original algorithm, thereby reducing the overall computational overhead.

To address the issue of ② reference point management, this section modifies Equation (36), which defines the determination function for the initial position of the bees, as follows:

$$x_{i,j}^1 = x_{best,j-1} + r(x_{\max,j}^1 - x_{\min,j}^1) \quad (40)$$

where  $x_{best,j-1}$  represents the best result from the previous round of optimization. This is based on the fact that, for a UAV, although the optimization of the  $k$ -th and  $(k+1)$ -th path points is nearly independent, the state parameters of each UAV do not change significantly between iterations. In other words, the optimization result for the  $(k+1)$ -th path point is expected to be similar to that of the  $k$ -th path point. Therefore, using the optimal value from the previous iteration as a reference point provides greater relevance for subsequent optimization.

The ABC algorithm is more effective in addressing the issue of ③ insufficient convergence, as it inherently exhibits better convergence properties compared to other intelligent algorithms. This is because the ABC algorithm simulates the process of bee foraging, which involves different types of bees conducting global and local searches in the search for food sources. Worker bees refine their solutions through local search, while scout bees are responsible for exploring new areas, thus avoiding local optima and enhancing global search capabilities and convergence. Additionally, due to the presence of scout bees in the algorithm, they can randomly explore new areas. If the current local solution no longer shows improvement, scout bees will avoid falling into local optima by exploring new solution spaces. This mechanism enhances the algorithm's global exploration capabilities, thereby improving convergence.

To tackle the issue of ④ susceptibility to local optimization, this paper further modifies Equation (40), as follows:

$$x_{i,j}^1 = x_{best,j-1} + r_p(x_{\max,j}^1 - x_{\min,j}^1) \quad (41)$$

The recurrence formula of  $r_p$  is:

$$r_{p+1} = \begin{cases} 2r_p & , 0 \leq r_p \leq \frac{1}{2} \\ 2(1-r_p), \frac{1}{2} < r_p \leq 1 \end{cases} \quad (42)$$

Equation (42) essentially represents a tent map sequence. The advantage of a chaotic sequence over a random number is that chaos is ergodic, meaning that, in theory, chaotic sequences can explore all possible values. This property helps the algorithm avoid being trapped in local optima. The core of using chaotic sequences to improve the search strategy of bees lies in the good ergodicity of chaotic sequences. Due to the Lyapunov exponent and positive stability, the path of the system continuously “expands” rather than contracts over time. This implies that the system's orbit does not linger in a specific subset or region, but rather spreads throughout the entire phase space. As time progresses, chaotic orbits cover various regions of the phase space, achieving ergodicity. Therefore, the ergodicity

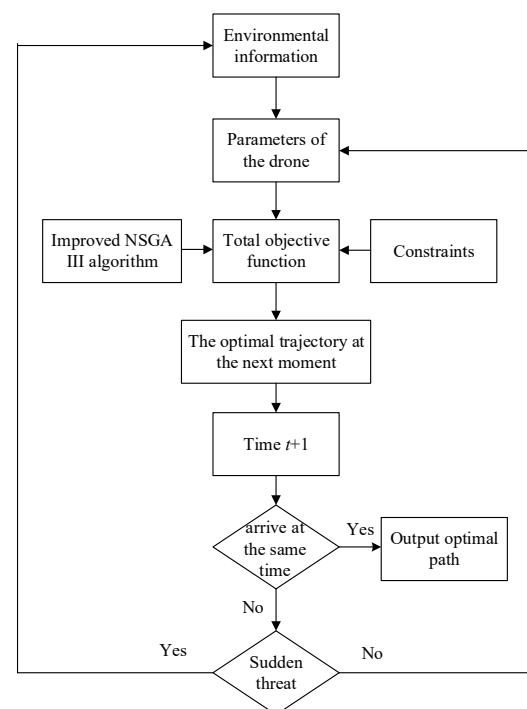
of the chaotic sequence is ensured. Consequently, the improved ABC algorithm is more likely to achieve global search.

In summary, this section addresses the four issues present in NSGA-III and enhances the optimization efficiency by improving the ABC algorithm.

Subsequently, this paper will apply this improved method to optimize the objective function and determine the path of the UAV cluster.

## 6. Path Planning Process Based on Improved NSGA III

Based on the objective function, constraints, and corresponding methods constructed in the previous section, this section develops a route optimization method, the process of which is shown in Figure 8.



**Figure 8.** Path planning flowchart.

To enable drone swarms to reach target points simultaneously, this section constructs the flowchart shown in Figure 8, and its steps can be detailed as follows:

Step 1: Construct the threat function in Section 4.2 based on environmental information.

Step 2: Obtain the current position and state information of each drone within the drone swarm, and construct the cost function mentioned in Section 4.3.

Step 3: Combine the threat function and cost function, and then incorporate the constraint conditions to construct the objective function to be optimized.

Step 4: Utilize the improved NSGA III algorithm to optimize the objective function, obtaining the position and state information of each drone for the next moment. Then increment the time  $t$  by 1.

Step 5: Determine whether the drones have arrived at the task point simultaneously. If not, update the drone location information and return to Step 2; otherwise, output the flight paths of all drones, and the algorithm ends.

Step 6: Determine whether there is an unexpected threat in the environment. If there is an unexpected threat, return to Step 1; if not, return to Step 2.

Through the aforementioned process, the flight paths for all drones to arrive simultaneously can be planned. Through the aforementioned process, the flight paths for all

drones to arrive simultaneously can be planned. In this paper, a centralized approach is adopted for drone decision-making. That is, each drone transmits its own state information to the decision-making drone, which performs unified optimization and calculation. This approach enables a good grasp of global information and avoids collisions between drones. In the event of a failure in one of the non-decision-making drones, the aforementioned Step 6 can be executed for re-planning. If the decision-making drone fails, the backup decision-making drone set according to the previous sequence will take over the subsequent planning.

A brief introduction to the threats in Step 6. When optimizing the drone's flight path, in addition to conventional static threats, the drone will also face challenges from sudden threats, dynamic threats, and unknown threats.

When drones face sudden threats, they can follow the strategy outlined in Figure 8. Specifically, the current situation is treated as a new initial condition, and the subsequent flight paths of each drone are re-planned. This is also an advantage of the method proposed in this paper. Because it does not require drones to take off from the same location, it relaxes the requirements for flight path planning methods. This ensures that drones can start re-planning from any location.

When a drone faces dynamic threats, the threat zone of these threats can also be regarded as a static threat at a certain moment. The prediction of its dynamic threat zone can be based on previous research [2], to obtain the subsequent location of the threat area. After obtaining the subsequent location of the moving threat, the environmental parameters can be updated in real-time using the method shown in Figure 8, so as to plan the drone's subsequent flight path.

When there is an unpredictable threat in the environment, there are two ways to deal with it. The first is when the threat suddenly appears, in which case it can be handled as an emergency threat. The other is when the unpredictable threat exists in the form of probability, in which case it can be handled as a threat source in Section 4.2.2. This enables us to deal with unpredictable threats.

## 7. Simulation Verification

### 7.1. Verification of the Algorithm

To verify the validity of the proposed spatio-temporal Cooperative Route Planning Model and assess the effectiveness of the algorithm, simulation experiments are conducted in this paper. Ten threat sources are placed within the flight area, and the corresponding detailed parameter settings are provided in Table 1.

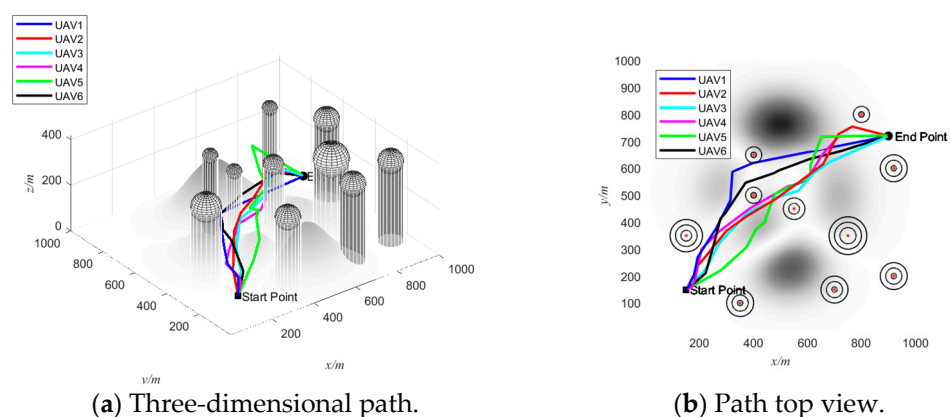
**Table 1.** Parameters of threat source.

| No. | Location (km) | Threat Radius (km) | Threat Coefficient |
|-----|---------------|--------------------|--------------------|
| 1   | (150,350)     | 60                 | 1.3                |
| 2   | (350,100)     | 50                 | 1.0                |
| 3   | (400,500)     | 30                 | 1.2                |
| 4   | (550,450)     | 40                 | 1.5                |
| 5   | (400,650)     | 30                 | 1.5                |
| 6   | (700,150)     | 50                 | 1.6                |
| 7   | (400,650)     | 30                 | 1.5                |
| 8   | (920,200)     | 50                 | 1.4                |
| 9   | (920,600)     | 50                 | 1.3                |
| 10  | (800,800)     | 30                 | 1.6                |

The starting and ending points of the six UAVs are set at coordinates (150, 150, 50) and (900, 720, 50), respectively. The population size of the bee colony used for optimization is 100. Considering that drone path optimization is a high-dimensional and complex problem, it is recommended to set the number of employed bees, denoted as  $SN$ , to a value between 20 and 50 based on ABC experience. Therefore, with the number of employed bees set to 40, the number of guard bees is set to 60. The particle dimension, denoted as  $D$ , is 18, representing the three-dimensional coordinates of 6 drones. Similarly, based on the empirical formula for the abandonment threshold  $Th_{li}$ , which is:

$$Th_{li} = D \times SN \quad (43)$$

The maximum iteration count for the ABC algorithm is recommended to be between 500 and 2000, and in this paper, it is set to 1000. Using the method proposed in this paper, the UAV path is obtained, as shown in Figure 9.

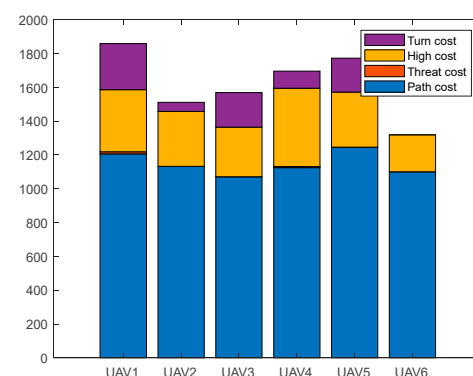


**Figure 9.** Optimization results of UAV track.

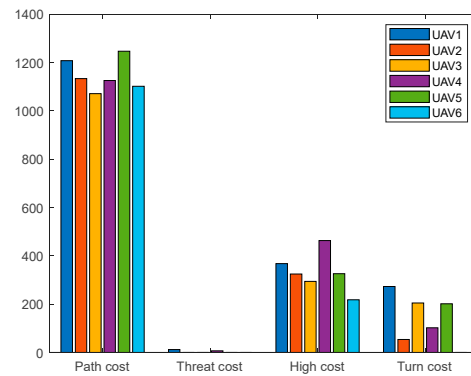
As shown in Figure 9, the six UAVs effectively avoid threats, with their flight paths remaining relatively smooth, ensuring smooth turns. The flight altitude is kept low while still adhering to the high-speed constraints.

The six UAVs maintain an equal number of path points from start to finish, ensuring that they arrive at the destination simultaneously. Based on the optimization results, this approach also demonstrates a reduction in flight time.

To further quantify the threats and costs encountered by the UAVs during the flight process, the corresponding costs are calculated. The results are presented in Figures 10 and 11.



**Figure 10.** The cost of each UAV.



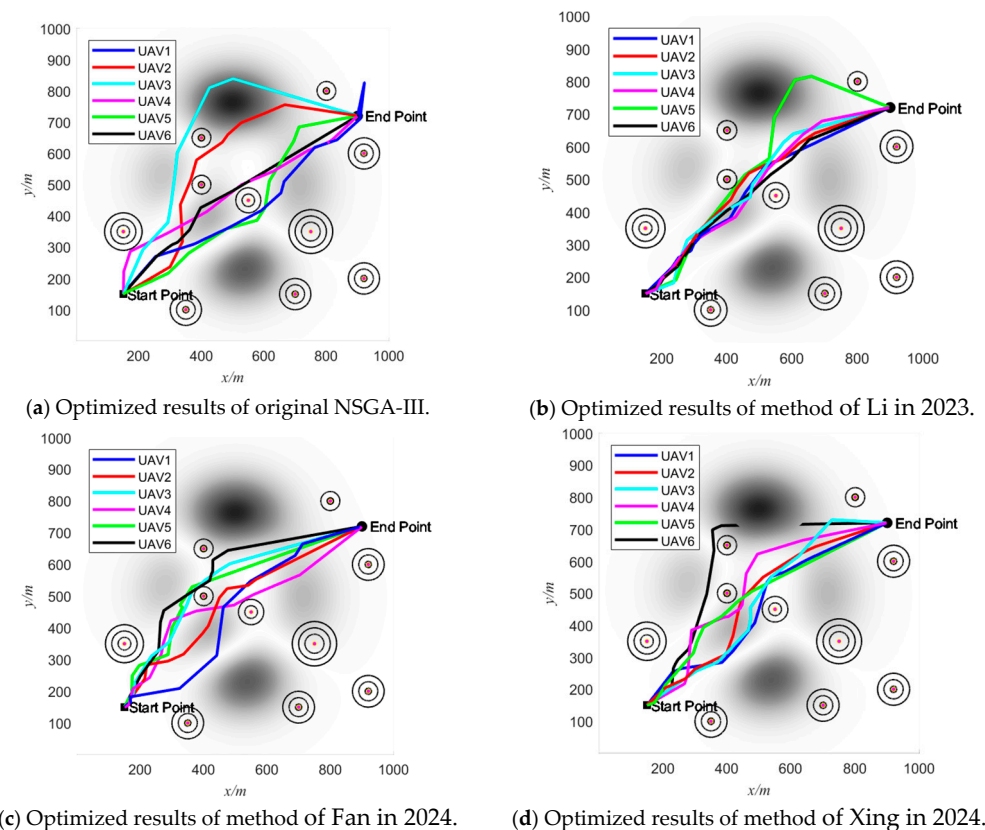
**Figure 11.** Result chart of each cost.

As shown in Figures 10 and 11, the path cost represents the largest component of the UAV's total cost, as the UAV incurs a cost for each segment of its flight path. The threat cost is the smallest component, emphasizing the importance of minimizing threat exposure to enhance UAV safety. The costs associated with altitude and turning are relatively low, indicating that the UAV performs fewer, gentler turns at a lower flight altitude, which further ensures its flight stability and safety.

From Figures 9–11, it can be concluded that the method proposed in this paper effectively plans the UAV's flight path, minimizing both flight cost and threat exposure while ensuring that the UAV cluster arrives at the destination simultaneously.

### 7.2. Comparison of Different Methods

To further evaluate the performance of the proposed algorithm, it is compared with the original NSGA-III algorithm, as well as the algorithms presented in References [1, 2, 35]. The simulation environment and conditions are kept consistent across all comparisons, and the resulting track optimization outcomes are presented in Figure 12.



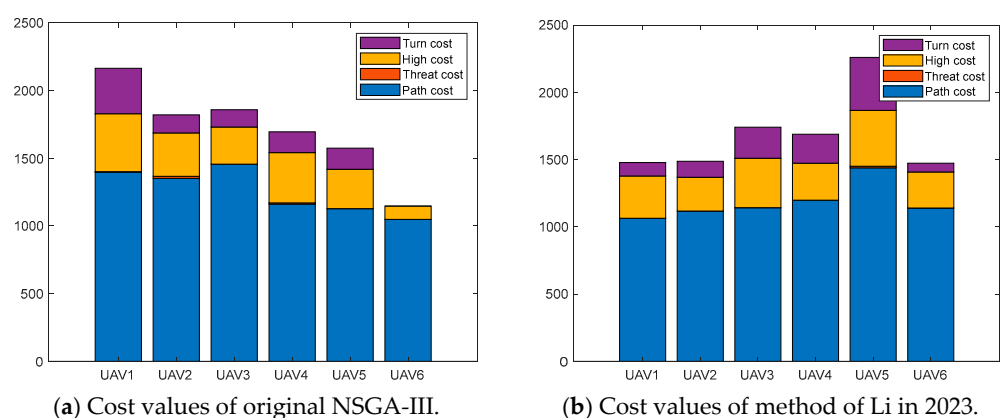
**Figure 12.** Optimization results of different algorithms.

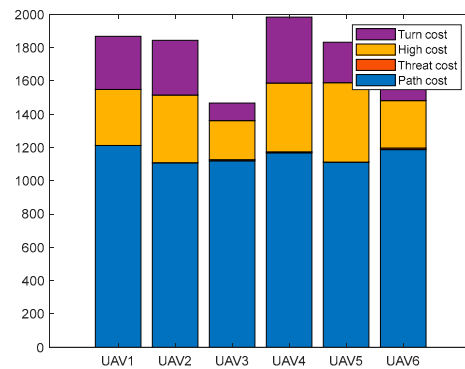
By comparing Figure 12 with Figure 9, it is evident that Figure 12a,b produce significantly longer flight paths. This is attributed to the particle swarm optimization algorithm used in the original NSGA-III and the method in [1], which both fall into local optima. As a result, in such local points, the upward migration becomes the best available solution at that time. However, this leads to a “butterfly effect,” causing noticeable path deviations and longer routes, as shown in Figure 12. This illustrates the impact of local optimization, where the incorrect selection of key steps can accumulate errors, significantly increasing future costs and risks. The reason for this situation is that the method in reference [1] employs group communication to escape from local optima. However, the optimization point falls at a time when group communication has not yet occurred, meaning that the point gets trapped in a local optimum, thereby exerting a significant impact on subsequent steps.

In contrast, Figure 12c demonstrates that the ABC algorithm employed in [2] is more effective at escaping local optima. As a result, the issue of long paths observed in Figures 12a and 12b does not occur in Figure 12c. As shown in Figure 12c, the path is shorter, leading to a reduced corresponding cost. However, the route appears overly dense, which increases the risk of UAV collisions. This is due to the fact that the method in [2], which integrates NSGA-III, faces challenges when handling multiple objective functions, resulting in the dense path configuration depicted in Figure 12c.

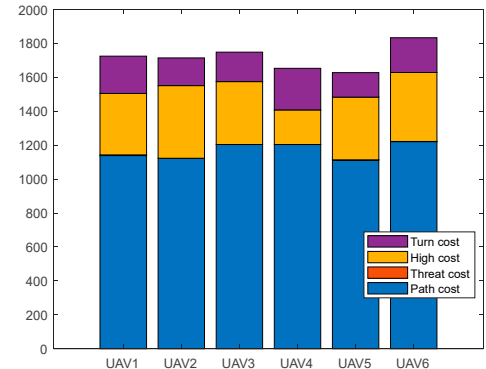
As evident from Figure 12d, the reinforcement learning-based approach adopts the strategy of traversing through threat sources to ensure a lower overall cost. Traversing through threat sources is a high-risk but also high-reward strategy. That is, if there are no sudden threats during the traversal process, the route is short, and the penalty is small. However, if there are threats, it becomes an extremely costly solution. This is also a characteristic of reinforcement learning, which can generate new solutions that deviate from conventional cognition.

To further compare the performance of the algorithms, the corresponding generation values for the three methods are presented in Figure 13. Additionally, 50 Monte Carlo simulation experiments were conducted for each algorithm. Using the improved algorithm proposed in this paper as a reference, the cost ratio and algorithm runtime are summarized in Table 2.





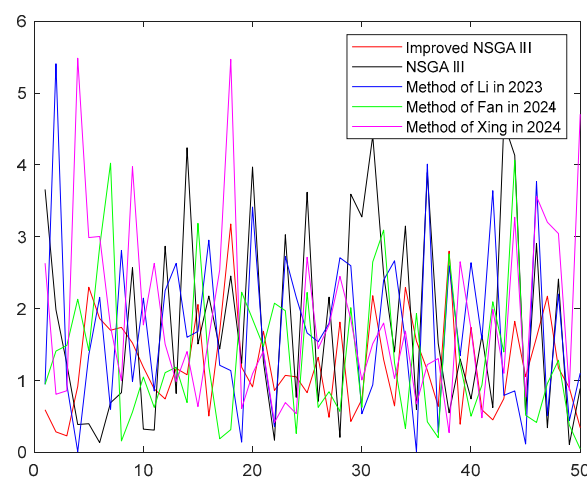
(c) Cost values of method of Fan in 2024.



(d) Cost values of method of Xing in 2024..

**Figure 13.** Cost values of different algorithms.

Meanwhile, 50 Monte Carlo simulation experiments were conducted on different algorithms, with the improved algorithm presented in this paper serving as the reference. The resulting data distribution is shown in Figure 14. Statistics were then compiled from the data in Figure 14, yielding the algorithm performance comparison presented in Table 2.

**Figure 14.** Comparison of results from 50 Monte Carlo experiments.**Table 2.** Cost comparison table.

| Algorithm         | Confidence Level of Normal Distribution | Ratio of Average Cost | Ratio of Cost Variance | Average Time/s |
|-------------------|-----------------------------------------|-----------------------|------------------------|----------------|
| Improved NSGA-III | 96%                                     | 1                     | 1                      | 75.71          |
| NSGA-III          | 90%                                     | 1.152                 | 2.450                  | 92.70          |
| Method in [1]     | 92%                                     | 1.136                 | 2.374                  | 72.62          |
| Method in [2]     | 94%                                     | 1.185                 | 1.891                  | 94.89          |
| Method in [35]    | 90%                                     | 1.083                 | 2.623                  | 34.50          |

The average cost performance ratio refers to the ratio of the cost incurred by each aircraft to the cost planned by this method for the corresponding aircraft, with the average of these values being taken. The variance of the cost ratio is calculated based on these ratios. The average time represents the time taken from the first point to the last point, reflecting the global optimization. Finally, after performing 50 Monte Carlo simulations and averaging all values, the results are presented in Table 2. Meanwhile, as can be seen

from Table 2, after conducting a normal distribution test on 50 sets of results, the confidence level of all data following a normal distribution is not lower than 90%. Therefore, relevant analysis methods for normally distributed data can be utilized to compare and analyze the data in Table 2.

By comparing Figure 13 with Figure 10 and referring to Table 2, it is evident that the improvements proposed in this paper enhance the performance of the NSGA-III method. The proposed method reduces the likelihood of NSGA-III and the particle swarm optimization algorithm, as seen in Reference [1], falling into local optima, thereby mitigating the issue of cost accumulation.

Furthermore, the new particle generation strategy in the ABC algorithm significantly improves the speed of the NSGA-III method. In fact, in terms of speed, this approach approaches that of the particle swarm optimization algorithm, which is known for its fast processing. This improvement arises because, when dealing with many objective functions, the speed of the particle swarm optimization algorithm tends to decrease. More importantly, the algorithm places the initial position of the new bee near the optimal result from the previous round, providing better initial conditions for optimization compared to randomly generated particles, thereby accelerating the optimization process.

In comparison with other methods, it is also apparent that the cost function in the original NSGA-III method and the method in Reference [1] is only slightly higher, but with a larger variance. This is due to the long path problem observed in Figures 12a and 12b, which corresponds to the peak cost values shown in Figure 13a,b. However, the method in reference [1] is an improvement based on the particle swarm optimization algorithm. From the comparison results, it can be seen that the optimization speed of reference [1] is significantly better than that of this paper. This is also an inherent deficiency of this paper, that is, the optimization speed of the ABC algorithm is significantly weaker than that of PSO. Therefore, when facing problems that require a rapid solution, the applicability of this paper is limited.

Conversely, the method in Reference [2] demonstrates the opposite behavior. As shown in Figure 13c, although the generation value in Reference [2] is higher than that of the other methods, the path cost remains relatively stable. From this comparison, it can be concluded that some methods struggle to balance optimality with stability, whereas the proposed method effectively balances both, achieving better optimization speed. The advantage of this paper over the work in reference [2] lies in the fact that it further improves the ABC algorithm based on the foundation laid in reference [2], thus enhancing its performance. However, this paper does not mention the motion threat discussed in reference [2]. Therefore, when facing motion threats, it is necessary to combine the method proposed in this paper with the author's previous work on reference [2] to achieve avoidance of motion threats.

Meanwhile, the method based on reinforcement learning in the literature [35], although having a lower mean, exhibits a higher variance, indicating that the algorithm is not stable enough. This is mainly due to three reasons. First and foremost, the problem addressed in this paper involves multiple variable functions. The performance of reinforcement learning decreases as the number and complexity of reward functions increase. Especially for tasks with clear objective functions and constraints, the effectiveness of reinforcement learning may not be as good as that of swarm intelligence methods. The second reason is that, despite reinforcement learning's powerful global learning ability, it tends to fall into local optima when solving multi-objective optimization problems. The third reason also stems from the inherent limitations of reinforcement learning. The selection of network parameters in reinforcement learning requires continuous exploration and trial and error, making it difficult to ensure stable algorithm operation. Although reinforcement learning is fast in practical applications, as can be seen from Table 2, the

method in the literature [35] is significantly superior to other methods. This is because it consumes a large amount of computational resources during training, resulting in a fast algorithm speed in practical use. However, the problem function studied in this paper has clear functions, simple constraints, and requires the algorithm to converge quickly. It is more suitable for the ABC algorithm, which can better balance the contradiction between global search and local search. Therefore, the ABC algorithm performs better. If this paper were to address a dynamically complex model or require agents to accumulate experience through long-term interaction, the effectiveness of reinforcement learning would inevitably be superior.

Through the qualitative and quantitative comparisons, as well as the effect analysis above, it can be concluded that the proposed method significantly improves the convergence of the original NSGA-III algorithm. Furthermore, compared to mainstream optimization methods, the efficiency of the combination of the ABC algorithm and NSGA-III is also enhanced.

### 7.3. Ablation Experiment

As can be seen from Section 7.2, the method proposed in this paper has shown significant improvement compared to the original ABC algorithm. However, this improvement is achieved through improvements in three aspects mentioned in Section 5.4. The overall performance improvement does not necessarily mean that each individual improvement has a positive effect. There may be cases where the performance improvement of one aspect is significant enough to offset the negative effects of other aspects. Therefore, ablation experiments are conducted in this section to quantitatively verify the role and function of each improvement.

The three improvement strategies in Section 5.4 are named as Improved Search Direction, Improved Initial Distribution, and Improved Iterative Strategy. Hereinafter, they are abbreviated as Strategies 1–3, respectively. All three methods underwent 50 Monte Carlo experiments, and the results are presented in Table 3.

**Table 3.** Cost comparison table of three strategies.

| Algorithm         | Confidence Level of Normal Distribution | Ratio of Average Cost | Ratio of Cost Variance | Average Time/s |
|-------------------|-----------------------------------------|-----------------------|------------------------|----------------|
| Improved NSGA-III | 96%                                     | 1                     | 1                      | 75.71          |
| NSGA-III          | 90%                                     | 1.152                 | 2.450                  | 92.70          |
| Strategy 1        | 94%                                     | 1.132                 | 2.033                  | 78.34          |
| Strategy 2        | 92%                                     | 1.147                 | 2.016                  | 85.11          |
| Strategy 3        | 96%                                     | 1.061                 | 1.483                  | 80.63          |

As can be seen from Table 3, the results obtained after conducting 50 Monte Carlo experiments for each method approximately follow a normal distribution. Therefore, data analysis can be performed on them.

The comparison between the algorithm presented in this paper and the original ABC algorithm has been elaborated in Section 7.2, and will not be repeated in this section.

Comparing Strategy 1 with the original algorithm, it can be seen that after improving the search direction, the algorithm's time consumption is significantly reduced, but there is no obvious improvement in the cost-to-variance ratio. This is because this method sacrifices the comprehensiveness of the search to enhance the optimization speed. Therefore, this improvement cannot enhance the overall performance of the algorithm.

Similarly, comparing Strategy 2 with the original algorithm reveals that this improved strategy does not enhance the cost-to-variance ratio, but it reduces the algorithm's time consumption. Strategy 2 sacrifices a small amount of diversity in the search space,

but in return, it enhances the algorithm's performance. This improvement comes at a low cost but yields high benefits.

Strategy 3 is a further improvement based on Strategy 1. The results show that this method enhances the overall performance of the algorithm. This is because Strategy 3 utilizes chaotic search to ensure the ergodicity of the search. At the same time, based on the improved search direction, it shortens the search time. Therefore, this method can more comprehensively balance diversity and search efficiency, thus ensuring the improvement of algorithm performance.

Meanwhile, Strategy 2 and Strategy 3 represent improvements made to the algorithm in two different dimensions. Therefore, the effectiveness of these two methods can be combined, further enhancing the search efficiency. Ultimately, the efficiency of the algorithm presented in this paper is achieved.

## 8. Conclusions

This paper presents a path planning method for UAV cluster simultaneous arrival based on an improved NSGA-III algorithm. By incorporating the strengths of the bee colony algorithm, the method effectively enhances the ability to solve multi-objective optimization problems. Through the introduction of multiple optimization objectives, such as time coordination, threat mitigation, and resource consumption, this study not only addresses the core challenge of ensuring simultaneous arrival for UAV clusters but also makes significant advancements in flight safety and resource optimization. Simulation results validate the effectiveness of the proposed method in UAV cluster path planning. Compared to other route planning algorithms, the improved NSGA-III algorithm shows significant improvements in both the accuracy and efficiency of the solutions, making it more suitable for addressing route planning challenges in complex environments. Additionally, the improved algorithm effectively avoids UAV collisions, optimizes flight paths, and enhances task completion efficiency.

However, this paper is an exploratory study on the simultaneous arrival route planning method. When facing complex geographical environments, numerous movements, and unknown threats, the model of the method presented in this paper still needs further improvement and refinement. Meanwhile, further research and demonstration are needed to enhance the performance of the ABC algorithm.

**Author Contributions:** Conceptualization, D.Q. and H.L.; Validation, X.S., H.L. and X.R.; Formal analysis, X.R.; Investigation, H.L.; Resources, D.Q.; Writing—original draft, D.Q. and X.H.; Writing—review and editing, X.S.; Visualization, X.H.; Supervision, X.R.; Funding acquisition, X.H. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Natural Science Basic Research Program of Shaanxi, grant number 2024JC-YBMS-466.

**Data Availability Statement:** The raw data supporting the conclusions of this article will be made available by the authors on request.

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

## References

1. Hao, L.; Xiangyu, F.; Manhong, S. Research on the Cooperative Passive Location of Moving Targets Based on Improved Particle Swarm Optimization. *Drones* **2023**, *7*, 264. <https://doi.org/10.3390/drones7040264>.
2. Fan, X.; Li, H.; Chen, Y.; Dong, D. A Path-Planning Method for UAV Swarm under Multiple Environmental Threats. *Drones* **2024**, *8*, 171. <https://doi.org/10.3390/drones8050171>.

3. Fan, X.; Bai, P.; Wang, H.; Zhang, J.; Li, H. Joint Optimization Method of Airborne MIMO Radar Track and Radiated Power Based on Mutual Information. *IEEE Access* **2020**, *8*, 136278–136295. <https://doi.org/10.1109/ACCESS.2020.3011421>.
4. Cui, S.; Li, H.; Fan, X.; Ni, L.; Hou, J. A Multi-UAV Distributed Collaborative Search Algorithm Based on Maximum Entropy Mechanism. *Drones* **2025**, *9*, 592. <https://doi.org/10.3390/drones9080592>.
5. Liu, M.; Huang, Y.; Zhang, Z.; Meng, Y. Multi-UAV Path Planning for Plural Data Collection: Distributed Multiagent Deep Reinforcement Learning Algorithms. *IEEE Internet Things J.* **2025**, *12*, 45175–45192. <https://doi.org/10.1109/JIOT.2025.3600404>.
6. Fan, X.; Li, H.; Chen, Y.; Dong, D. UAV Swarm Search Path Planning Method Based on Probability of Containment. *Drones* **2024**, *8*, 132. <https://doi.org/10.3390/drones8040132>.
7. Jiao, Z.; Wang, M.; Qiao, S.; Zhang, Y.; Huang, Z. Transformer-Based Object Detection in Low-Altitude Maritime UAV Remote Sensing Images. *IEEE Trans. Geosci. Remote Sens.* **2025**, *63*, 4210413. <https://doi.org/10.1109/TGRS.2025.3594085>.
8. Lyu, Y.; Wang, P.; Bai, X.; Tansey, K.; Ning, P.; Yang, L.; Zhang, T.; Hong, Y.; Zhang, J. Optimizing Soil Properties Mapping: A Comparative Study of UAV and Satellite Imagery with Ensemble Learning Framework. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2025**, *18*, 29366–29378. <https://doi.org/10.1109/JSTARS.2025.3631151>.
9. Chen, M.; Li, Y.; Dai, Z.; Zhang, T.; Zhou, Y.; Wang, H. A Robust Multi-Domain Adaptive Anti-Jamming Communication System for a UAV Swarm in Urban ITS Traffic Monitoring via Multi-Agent Deep Deterministic Policy Gradient. *IEEE Trans. Intell. Transp. Syst.* **2025**, early access. <https://doi.org/10.1109/TITS.2025.3584216>.
10. Deng, P.; Sun, B.; Jiang, Q.; Fan, Q. A simplified task-assisted constrained multi objective genetic algorithm for USV-UAV collaborative maritime search and rescue. *IEEE Trans. Veh. Technol.* **2025**, early access. <https://doi.org/10.1109/TVT.2025.3628067>.
11. Roberge, V.; Tarbouchi, M.; Labonte, G. Comparison of Parallel Genetic Algorithm and Particle Swarm Optimization for Real-Time UAV Path Planning. *IEEE Trans. Ind. Inform.* **2013**, *9*, 132–141. <https://doi.org/10.1109/TII.2012.2198665>.
12. Pan, J.; Han, T.; Xiao, B.; Zhan, X.-S.; Yan, H. Predefined-Time Bipartite Time-Varying Formation Tracking of Multiple Euler–Lagrange Systems via Estimator-Based Hierarchical Control Algorithm. *IEEE Trans. Circuits Syst. II Express Briefs* **2023**, *70*, 3444–3448. <https://doi.org/10.1109/TCSII.2023.3262963>.
13. Yan, K.-X.; Han, T.; Xiao, B.; Yan, H. Energy-Constraint Adaptive Time-Varying Formation Tracking for Multi-Agent Systems With Multiple Leaders and Bounded Unknown Inputs. *IEEE Trans. Circuits Syst. II Express Briefs* **2024**, *71*, 802–806. <https://doi.org/10.1109/TCSII.2023.3312854>.
14. Zhang, X.; Zhan, X.; Wu, J.; Han, T.; Yan, H. Distributed Adaptive Time-Varying Formation Tracking for Input Saturated Multi-Agent Systems With Multiple Leaders. *IEEE Trans. Circuits Syst. II Express Briefs* **2024**, *71*, 2084–2088. <https://doi.org/10.1109/TCSII.2023.3331844>.
15. Zhang, X.; Yang, Q.; Zeng, X.; Fang, H.; Chen, J. Cooperative Shape–Translation Estimation and Control for Time-Varying Linear Formation. *IEEE Trans. Autom. Control* **2025**, *70*, 5206–5221. <https://doi.org/10.1109/TAC.2025.3540570>.
16. Hou, H.-Q.; Liu, Y.-J.; Lan, J.; Liu, L. Adaptive Fuzzy Fixed Time Time-Varying Formation Control for Heterogeneous Multiagent Systems With Full State Constraints. *IEEE Trans. Fuzzy Syst.* **2023**, *31*, 1152–1162. <https://doi.org/10.1109/TFUZZ.2022.3195609>.
17. Park, B.S.; Yoo, S.J. Time-Varying Formation Control With Moving Obstacle Avoidance for Input-Saturated Quadrotors With External Disturbances. *IEEE Trans. Syst. Man Cybern. Syst.* **2024**, *54*, 3270–3282. <https://doi.org/10.1109/TSMC.2024.3358345>.
18. Li, J.; Hua, Y.; Yu, J.; Dong, X.; Feng, Z.; Ren, Z. Resilient Time-Varying Formation Tracking Control for General Linear Multiagent Systems With a Nonautonomous Leader and Adversarial Followers. *IEEE Trans. Cybern.* **2024**, *54*, 6203–6216. <https://doi.org/10.1109/TCYB.2024.3411092>.
19. Peng, X.-J.; He, Y.; Li, H.; Tian, S. Robust Time-Varying Formation Control of One-Sided Lipschitz Nonlinear Multiagent System With Delays via Optimization Algorithm. *IEEE Trans. Cybern.* **2025**, *55*, 1801–1813. <https://doi.org/10.1109/TCYB.2025.3535958>.
20. Zhao, Y.; Zhu, F.; Xu, D. Event-Triggered Bipartite Time-Varying Formation Control for Multiagent Systems With Unknown Inputs. *IEEE Trans. Cybern.* **2023**, *53*, 5904–5917. <https://doi.org/10.1109/TCYB.2022.3208228>.
21. Li, J.; Dong, X.; Yu, J.; Li, Q.; Ren, Z. Resilient Time-Varying Formation Tracking for Heterogeneous High-Order Multiagent Systems With Multiple Dynamic Leaders. *IEEE Trans. Control Netw. Syst.* **2024**, *11*, 89–100. <https://doi.org/10.1109/TCNS.2023.3272295>.
22. Liu, X.; Wu, X.; Xie, Y.; Guo, B. Time-Varying Formation Control With Transient Performance for Multiagent Systems Under Jointly Connected Topologies. *IEEE Syst. J.* **2024**, *18*, 497–500. <https://doi.org/10.1109/JSYST.2023.3306346>.
23. Wu, X.; Guo, Z.; Liu, X.; Xie, T. Fault-Tolerant Time-Varying Formation Tracking Control for Multi-Agent Systems With Actuator Faults and Switching Topologies. *IEEE Access* **2023**, *11*, 131140–131151. <https://doi.org/10.1109/ACCESS.2023.3333660>.

24. Abbasi, M.; Marquez, H.J. Dynamic Event-Triggered Formation Control of Multi-Agent Systems With Non-Uniform Time-Varying Communication Delays. *IEEE Trans. Autom. Sci. Eng.* **2025**, *22*, 8988–9000. <https://doi.org/10.1109/TASE.2024.3494658>.
25. He, J.; Yang, D.; Sun, J.; Zhang, J.; Li, C. Group Time-Varying Formation Tracking Control for Multiagent Systems Using Multiple Dynamic Edge-Event-Triggered Mechanisms. *IEEE Trans. Cybern.* **2025**, *55*, 4208–4218. <https://doi.org/10.1109/TCYB.2025.3581345>.
26. Wang, X.; Wang, Z.; Li, Z.; Bi, Y. Distributed Fixed-Time Nonsingular Terminal Sliding Mode Time-Varying Formation Control of ASVs Based on Hierarchical Control Mechanism. *IEEE Access* **2025**, *13*, 68920–68929. <https://doi.org/10.1109/ACCESS.2025.3561925>.
27. Li, J.; Du, J.; Li, Y.; Xu, G. Distributed Robust Prescribed Performance 3-D Time-Varying Formation Control of Underactuated AUVs Under Input Saturations and Communication Delays. *IEEE J. Ocean. Eng.* **2023**, *48*, 649–662. <https://doi.org/10.1109/JOE.2023.3245671>.
28. Ismail, A.; Bagula, B.A.; Tuyishimire, E. Internet-Of-Things in Motion: A UAV Coalition Model for Remote Sensing in Smart Cities. *Sensors* **2018**, *18*, 2184.
29. Yao, W.R.; Qi, N.M.; Liu, Y.F. Online Trajectory Generation with Rendezvous for UAVs Using Multistage Path Prediction. *J. Aerosp. Eng.* **2017**, *30*, 04016092.
30. McClain, T.W.; Beard, R.W. Coordination variables, coordination functions, and cooperative timing missions. *J. Guid. Control Dyn.* **2005**, *28*, 150–161.
31. Zhai, L.; Wu, H.; Lai, L.; Gao, Z. Intelligent Optimization Algorithms for Multi-UAV Path Planning: A Comprehensive Review. *IEEE Access* **2025**, *13*, 101106–101130. <https://doi.org/10.1109/ACCESS.2025.3577598>.
32. Manathara, J.G.; Ghose, D. Rendezvous of multiple UAVs with collision avoidance using consensus. *J. Aerosp. Eng.* **2012**, *25*, 480–489.
33. Choe, R.; Puig Navarro, J.; Cichella, V.; Xargay, E.; Hovakimyan, N. Cooperative Path Generation Using Pythagorean Hodograph Bézier Curves. *J. Guid. Control Dyn.* **2016**, *39*, 1744–1763.
34. Doğançay, K.; Hmam, H.; Xu, S. Distributed pseudolinear estimation and UAV path optimization for 3D AOA target tracking. *Signal Process.* **2017**, *133*, 64–78.
35. Xing, X.; Zhou, Z.; Li, Y.; Xiao, B.; Xun, Y. Multi-UAV Adaptive Cooperative Formation Trajectory Planning Based on an Improved MATD3 Algorithm of Deep Reinforcement Learning. *IEEE Trans. Veh. Technol.* **2024**, *73*, 12484–12499. <https://doi.org/10.1109/TVT.2024.3389555>.

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