aerospace

MDPI

Article

A Spatio-Temporal Collaborative Improved Multi-Strategy Dung Beetle Optimization Algorithm for 3D Path Planning of Multiple Unmanned Aerial Vehicles in Urban Environments

Yaowei Yu and Meilong Le *

College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China
* Correspondence: lemeilong901@163.com

Abstract

Collaborative 3D path planning for multiple unmanned aerial vehicles (UAVs) in dense urban airspace is difficult, which does not come from one factor alone. Buildings, flight restrictions, moving obstacles, and inter-UAV coupling all act together, and the search space grows quickly as the scene becomes more crowded. In such cases, a standard swarm optimizer may still find a path, but it often struggles with early feasibility, later-stage refinement, and local replanning after the environment changes. To deal with these issues, this paper develops a spatio-temporal collaborative improved multi-strategy dung beetle optimization algorithm, called STC-IMSDBO, for urban multi-UAV path planning. The framework combines five linked components: feasible-airspace population initialization, spatio-temporal variable-step search, multi-factor adaptive weighting, local game-based conflict handling, and rolling-horizon replanning. A normalized multi-objective cost is used to balance flight efficiency, smoothness, obstacle avoidance, airspace compliance, and cooperative safety. The method is tested in four simulated urban scenarios and compared with six representative methods. In the tested cases, the STC-IMSDBO generates shorter feasible routes, uses less energy, converges in fewer iterations, and maintains better cooperative safety than the comparison methods. These results suggest that the method is a useful planning option for dense urban missions such as logistics, inspection, and emergency response. That said, larger-swarm runtime tests and field validation are still needed.

Keywords: multiple UAVs; three-dimensional path planning; dung beetle optimization algorithm; cooperative obstacle avoidance; urban environment

Check for updates

Academic Editor: Miroslav Kelemen

Received: 14 March 2026
Revised: 6 May 2026
Accepted: 16 May 2026
Published: 29 May 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) license.

1. Introduction

Urban low-altitude operations are shifting from isolated flights to repeated task-oriented missions. Delivery, inspection, emergency response, and traffic surveillance all require routes that are safe, compliant, and efficient [1–3]. Compared with single-UAV operation, multi-UAV cooperation can divide tasks and execute them in parallel, which helps reduce mission time and enlarge the service area [4,5]. For this reason, multi-UAV cooperation has become an important technical route for urban low-altitude operations.

In urban scenes, multiple UAVs have to fly through an irregular three-dimensional space. Tall buildings block parts of the airspace, no-fly and restricted zones overlap, and low-altitude moving obstacles, such as birds or other aircraft, may appear during flight [6,7]. Path planning in this setting must handle single-UAV flight quality and fleet-level coordination at the same time. For a single UAV, the route should be short, smooth,

energy-aware, and free from static and dynamic obstacles. For the fleet, the planner must also consider inter-UAV separation, communication stability, airspace compliance, and coordinated arrival timing [8].

As the urban scene becomes denser and the fleet size increases, the planning problem becomes harder in two ways. First, the feasible airspace is fragmented by buildings, no-fly zones, and altitude restrictions. Second, the motion of one UAV affects the feasible choices of the others through separation, communication, and timing constraints. These factors make urban multi-UAV path planning a high-dimensional and strongly constrained problem, especially when online replanning is also required.

The path planning algorithms for UAVs can be broadly classified into conventional methods and meta-heuristic methods [9]. The conventional methods include A* [10], Dijkstra [11], and Rapidly exploring Random Trees (RRTs) [12] and their improved variants. These methods are useful in simple static single-UAV scenarios because their search logic is clear and easy to interpret. Their limitation becomes more visible in dense multi-UAV scenes, where the planner has to handle spatial obstacles, temporal coupling, and cooperative safety at the same time [10,13,14]. A* may suffer from rapidly increasing node expansion in cluttered maps [15], Dijkstra is often computationally expensive when the search graph grows [16], and RRTs can generate discontinuous or weakly smoothed paths, and it does not naturally encode spatio-temporal coordination among multiple UAVs [17]. Figure 1 shows a conceptual diagram of drone logistics delivery in urban areas.



Figure 1. Multi-UAV logistic delivery in urban area.

Swarm intelligence optimizers offer another route. They search the solution space through population update rules and usually do not require explicit graph expansion. The dung beetle optimization (DBO) algorithm, proposed in 2022, is a recently developed swarm intelligence optimizer [18]. It imitates several dung beetle behaviors, including ball rolling, dancing, reproduction, foraging, and stealing. Compared with some classical swarm optimizers, the DBO has a relatively simple structure and a small number of control parameters. These properties make it attractive for path planning problems where repeated fitness evaluation is required.

However, the standard DBO cannot be used directly for dense urban multi-UAV planning without modification. Its original update rules were designed for general optimization, not for constrained airspace, inter-UAV coupling, or dynamic local changes. Therefore, several mismatches appear when it is transferred to this task. This specifically manifests in five aspects:

1. The initial population is randomly generated without incorporating the constraints of the urban airspace. A large number of initial individuals fall into obstacles and

prohibited flight zones, resulting in insufficient effectiveness and diversity of the population, which directly weakens the global search ability of the algorithm.
2. The iteration step size is fixed and unable to balance the fine search for local obstacle avoidance by a single drone and the large-scale optimization of multi-UAV global collaboration. In the later stages of iteration, it is prone to fall into local optimum.
3. The position update of the stealing dung beetle adopts a fixed weight and does not introduce multi-UAV collaborative constraints to adjust to dynamic environments. In high-dimensional multi-UAV scenarios, the algorithm's stability is insufficient, and the collaborative planning effect is poor.
4. There is no dedicated multi-UAV collaborative obstacle avoidance mechanism. During the iteration process, there are often cases of path intersection and collision between UAVs, which cannot ensure the safety of the operation.
5. The static iteration framework cannot respond to the real-time changes in urban dynamic obstacles and does not have the ability to re-plan the path dynamically. It cannot adapt to low-altitude dynamic environments.

There are currently many studies on improvements to the DBO algorithm, mostly focusing on single-UAV path planning scenarios, and very few making targeted optimizations around the core constraints of urban multi-UAV collaboration. There are obvious shortcomings in scenario adaptation: most improvements do not optimize the population initialization by combining urban airspace rules, and still have the problem of low initial population effectiveness; no multi-UAV collaborative obstacle avoidance mechanism has been designed, which cannot solve the problem of inter-vehicle temporal and spatial collisions; nor has a dynamic replanning framework been constructed, making it difficult to adapt to the urban low-altitude dynamic environment and unable to support actual operational requirements.

The method proposed in this paper, termed STC-IMSDBO, is designed around these practical gaps. The point is not to add several isolated tricks to the original DBO. Each modification is tied to a specific weakness that appears in urban multi-UAV planning, and the five parts are connected through the same constraint and cost framework.

1. An airspace-constrained adaptive chaotic Latin hypercube strategy is used to generate the initial population directly in feasible urban airspace. This reduces invalid individuals at the beginning of the search;
2. The original tangential dance stage is replaced with a spatio-temporal coupled variable-step golden-sine update. The step size changes with iteration progress, individual fitness, and inter-UAV spacing, so the search does not rely on one fixed movement scale;
3. A multi-factor adaptive weighting mechanism is introduced into the stealing dung beetle stage. Instead of using a fixed coefficient, the update reflects iteration progress, diversity, fitness quality, and cooperative cost at the same time;
4. A local non-cooperative game update is inserted into the cooperative avoidance stage. In this way, inter-UAV separation is handled during path refinement rather than only after a candidate route has already been formed;
5. A rolling-horizon replanning module is added for dynamic scenes, so the planner can revise only the affected local segment after new obstacle information is sensed.

The structure of this paper is as follows: Section 2 reviews the related research on multi-UAV urban path planning and the improvement of the DBO algorithm; Section 3 builds a three-dimensional urban multi-UAV path planning model, including environmental modeling, constraint conditions, energy consumption model, and multi-objective cost function; Section 4 describes the STC-IMSDBO algorithm, including the original DBO, the five improvement strategies, implementation steps, complexity analysis, and convergence discussion; Section 5 conducts simulation experiments, including experimental

environment setup, comparative experiments, ablation experiments, statistical tests, and real-scenario testing, analyzing the experimental results; Section 6 summarizes the entire paper and looks forward to future research directions.

2. Related Work

2.1. Research on Urban Multi-UAV Path Planning

Urban multi-UAV path planning is a research hotspot at the intersection of unmanned aerial vehicle autonomous control and intelligent computing. The core lies in planning safe, efficient, and collaborative flight paths for multiple UAVs in complex urban environments. Existing algorithms are classified into three categories based on optimization principles: traditional search algorithms, swarm intelligence optimization algorithms, and deep reinforcement learning algorithms.

Traditional search algorithms are centered around graph/tree search, and representative algorithms include A* [19], Dijkstra [16], RRTs [20], and their improved versions. Reference [21] overcomes the limitations of the traditional A* algorithm in terms of path robustness and planning efficiency through three strategies: extended distance, bidirectional search, and smoothing. Reference [22] weights the heuristic function of the A* algorithm to improve computational efficiency. Reference [23] improves the A* algorithm by adopting bidirectional sector expansion, adding additional estimated costs, and variable step-size search strategies, ensuring the optimality of the generated path. Some studies have integrated risk networks and feedback mechanisms to improve the RRT* algorithm, enhancing the planning efficiency in complex environments. Reference [24] proposes a new framework that integrates RRTs and A*, with A* path optimization integrated as a post-processing module into the 3D-MIHE-RRT algorithm, constructing a dual-stage hybrid architecture, effectively addressing the problems of redundant and irregularly generated paths. However, when these improved algorithms are applied to multi-UAV collaborative scenarios, severe dimension explosion problems occur, with the computational complexity increasing randomly and sharply, unable to handle multi-UAV spatio-temporal collaborative constraints, and prone to path conflicts.

The swarm intelligence optimization algorithms simulate the collaborative behavior of biological groups and achieve global optimization through information exchange and iterative updates among individuals. Classic algorithms include GWO [25], WOA [26], etc. In recent years, new algorithms such as Parrot Optimization Algorithm (PO) [27], Anguilla Lizard Optimization Algorithm (ALOA) [28], and Artificial Lemming Algorithm (ALA) [29] have also emerged. In current research, [30] developed an Adaptive Multi-population Cooperative Whale Optimization Algorithm (AMCWOA), which balances exploration and exploitation by analyzing population distribution and improves the quality of solutions in challenging environments, but it did not cover multi-UAV collaborative scenarios. Reference [31] proposed a multi-UAV collaborative improved GWO algorithm, which uses chaotic mapping to optimize the initial solution and adopts a chaotic local search strategy to maintain the balance between exploration and exploitation, but the convergence speed is relatively slow in high-dimensional scenarios; there are also improvements to the DBO algorithm for single-UAV three-dimensional path planning, but they did not consider multi-UAV collaborative constraints [32]. In general, these algorithms perform better than traditional algorithms in single-UAV scenarios, but in urban multi-UAV collaborative scenarios, there are generally problems such as poor coordination, ease of falling into local optimum, and inability to handle spatial collisions.

Deep reinforcement learning algorithms have also been applied to multi-UAV path planning in recent years. Commonly used algorithms include Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [33] and Multi-Agent Proximal Policy Optimization

(MAPPO) [34], etc. Reference [33] proposed an improved MADDPG algorithm for path planning in complex buildings, enhancing the exploration ability; [34] designed an event-triggered MAPPO strategy to achieve multi-UAV cluster obstacle avoidance in restricted scenarios. These algorithms have certain adaptability to dynamic environments, but they have inherent defects such as low sample utilization rate, difficult training, and unstable convergence, making it difficult to ensure the global optimality of the path. They are prone to obstacle avoidance failure in urban environments with dense obstacles and cannot support actual operations [35]. Unlike MARL approaches that rely on stochastic policy exploration, STC-IMSDBO leverages deterministic geometric projections within the digital twin, making the constraint-handling process more explicit and easier to inspect in safety-critical planning tasks.

2.2. Research on the Improvement of the Dung Beetle Optimization Algorithm

Since the DBO algorithm was proposed in 2022, it has been widely applied in fields such as function optimization and path planning due to its simple structure and fast convergence speed. However, the inherent defects of the standard DBO algorithm have limited its application in complex optimization problems involving multiple UAVs in urban areas. Scholars at home and abroad have conducted various improvements from multiple dimensions. Zheng et al. proposed an improved DBO method for collaborative 3D UAV path planning [36]. Their study is closely related to the present work, but it mainly focuses on strengthening the DBO search process itself. In contrast, this paper further couples feasible-airspace initialization, local game-based separation handling, and rolling-horizon replanning for dense urban multi-UAV missions.

The current research on the improvement of the DBO algorithm mainly focuses on three aspects: population initialization, iteration mechanism, and weight strategy. In terms of population optimization, methods such as chaotic mapping [37], reverse learning [38], and good point set [39] are introduced to optimize the population distribution, which respectively improve the population diversity, uniformity of distribution, and selection of high-quality initial populations. However, none of these consider the specific scene constraints of urban airspace. A large number of initial individuals fall into obstacles or restricted flight zones and fail, resulting in low population effectiveness. Fuch mapping is used to generate initial solutions, and the reverse learning strategy is integrated to expand search space, improving the quality of the initial population [40]. In the iteration mechanism, evolutionary algorithms [41] and Levy flight are introduced to optimize the iteration mechanism, using evolutionary algorithms to optimize the iteration process to improve the convergence speed [41], and combining Levy flight to expand the search space and improve the optimization accuracy [42]. However, these improvements are all targeted at single-UAV scenarios and do not consider the constraints of multi-UAV collaboration, making them unable to meet the requirements of multi-UAV collaborative planning.

In terms of the weight strategy, the original DBO adopts a fixed weight for updating the position, which has poor adaptability to high-dimensional complex environments and affects the stability of the algorithm. Existing studies have proposed various adaptive weight strategies to optimize the update mechanism [43,44]. By using adaptive inertia weights, the stability of the algorithm is improved. However, only a single factor such as the number of iterations is considered, and the cost of multi-UAV collaboration is not integrated. The stability of the algorithm in multi-UAV scenarios is still insufficient.

2.3. Motivation

The review above suggests that the current gap is not simply that the DBO has not been improved enough. The deeper issue is that many existing improvements are still not built around the actual structure of dense urban multi-UAV missions.

One common problem appears at the beginning of the search. Population samples may be spread well in a mathematical sense, yet many of them are still useless in real urban airspace because they fall into blocked or restricted regions.

Another problem arises during optimization. Single-UAV path quality and fleet-level coordination are often treated in separate steps. That usually works in light scenes, but once the airspace becomes tight, the planner may keep a decent individual route while the fleet-level result becomes unstable.

A third limitation is the static setting used in many studies. Static experiments are still useful, but they do not fully reflect what happens when local obstacle information changes during flight.

The STC-IMSDBO is developed with these points in mind. The goal is a more task-aligned DBO framework for urban multi-UAV planning, rather than a general purpose optimizer meant to fit every routing problem.

3. Urban Multi-UAV 3D Path Planning Model

3.1. Urban 3D Low-Flying Environment Modeling

This paper constructs a digital twin 3D environment model of a city, which completely covers the static obstacles, airspace control areas, and dynamic obstacles in low-flying scenarios in a city. The model is established based on the Cartesian coordinate system, and the 3D space covered by the model is $X \in [X_{\min}, X_{\max}]$, $Y \in [Y_{\min}, Y_{\max}]$ and $Z \in [Z_{\min}, Z_{\max}]$.

3.1.1. Modeling of Static Obstacles

The main static obstacles in the city are the buildings within the city, which are usually modeled as cube-shaped models. The mathematical Equation (1) for the i-th building is:

$$\Omega_{b,i} = \left\{ (x,y,z) \middle| \begin{array}{l} x \in [x_{i,s}, x_{i,e}], y \in [y_{i,s}, y_{i,e}] \\ z \in [0, H_{b,i}] \end{array} \right\}, \tag{1}$$

where x_{is}, x_{ie} represents the starting and ending coordinates of the building along the x-axis, y_{is}, y_{ie} represents the starting and ending coordinates along the y-axis, and $H_{b,i}$ represents the height of the building.

3.1.2. Modeling of Airspace Control Areas

The urban environment airspace control areas are divided into no-fly zones and restricted-fly zones. The no-fly zone is a three-dimensional area where UAVs cannot enter at all, while the restricted-fly zone is an area where UAVs can enter but cannot exceed the designated altitude limit.

No-fly zone:

$$\Omega_{nf} = \left\{ (x,y,z) \middle| \begin{array}{l} x \in [x_{nf,s}, x_{nf,e}], y \in [y_{nf,s}, y_{nf,e}] \\ z \in [0, H_{nf,max}] \end{array} \right\}; \tag{2}$$

restricted flight zone:

$$\Omega_{rf} = \left\{ (x,y,z) \middle| \begin{array}{l} x \in [x_{rf,s}, x_{rf,e}], y \in [y_{rf,s}, y_{rf,e}] \\ z \in [H_{rf,min}, +\infty) \end{array} \right\}, \tag{3}$$

where $H_{nf,\max}$ represents the maximum height of the no-fly zone, and $H_{rf,\min}$ represents the minimum restricted height of the restricted-fly zone. For multiple no-fly and restricted flight zones, Ω_{nf} and Ω_{rf} denote the union of all corresponding airspace regions.

3.1.3. Dynamic Obstacle Modeling

The urban dynamic obstacles mainly include moving low-altitude aircraft, birds, etc. They are modeled using a moving sphere model. The position of the l-th dynamic obstacle at time t is $(x_{l,t}, y_{l,t}, z_{l,t})$, and the safety radius is $R_{d,l}$. The mathematical model is as shown in Equation (4):

$$\Omega_{d,l}(t) = \left\{ (x,y,z) \middle| \sqrt{(x-x_{l,t})^2 + (y-y_{l,t})^2 + (z-z_{l,t})^2} \leq R_{d,l} \right\}. \tag{4}$$

If the path points fall within the defined range $\Omega_{d,l}(t)$ at any time, the dynamic obstacle avoidance penalty cost will be triggered.

This study designs four different levels of complexity of urban three-dimensional scenarios, namely low, medium, high, and ultra-high complexity, corresponding to urban suburban industrial parks, ordinary residential areas, core business districts, and core airspace + dynamic obstacles scenarios. Specific parameters are detailed in the experimental section.

3.2. Constraints for Multi-UAV Cooperative Flight

To ensure the safe and compliant flight of multiple UAVs in the urban environment, the planned path must meet the following constraints.

3.2.1. Spatial Boundary Constraint

The flight paths of all UAVs must be within the designated three-dimensional simulation space. The coordinates of any point on the path must satisfy:

$$0 \leq x_{p,s} \leq X_{\max}, 0 \leq y_{p,s} \leq Y_{\max}, 0 \leq z_{p,s} \leq Z_{\max}, \tag{5}$$

where $(x_{p,s}, y_{p,s}, z_{p,s})$ represents the coordinate of the s-th path point on the path of the p-th UAV.

3.2.2. Single-UAV Obstacle Avoidance Constraints

During flight, each UAV must keep a minimum clearance from both static and dynamic obstacles. The minimum clearance threshold is denoted by R_{safe}, and its measurement unit is meters. In this study, $R_{safe} = 5$ m. For the i-th obstacle, $o_i = (x_i, y_i, z_i)$ denotes the reference point of the i-th obstacle used for distance evaluation. For a spherical dynamic obstacle, it is the obstacle center. For a building obstacle, it can be taken as the closest point on the obstacle boundary to the current path point. The obstacle avoidance constraint is written as Equation (6):

$$\| p_{p,s} - o_i \|_2 > R_{safe}, \forall p, \forall s, \forall i, \tag{6}$$

where $p_{p,s} = (x_{p,s}, y_{p,s}, z_{p,s})$ denotes the s-th sampled path point of the p-th UAV.

3.2.3. Airspace Compliance Constraints

The flight path must not enter the prohibited area, and when entering the restricted area, it must not exceed the specified altitude, as shown in Equation (7):

$$(x_{p,s}, y_{p,s}, z_{p,s}) \notin \Omega_{nf}, \forall s. \tag{7}$$

For restricted flight zones, the horizontal projection of the UAV may enter the zone only when its altitude does not exceed the prescribed altitude limit.

3.2.4. Multi-UAV Collision Prevention Constraints

At any given time, the spatial distance between any two UAVs must be greater than the multi-UAV safety distance $d_{sep} = 10$ m, as shown in Equation (8):

$$\| p_p(h) - p_q(h) \|_2 > d_{sep}, \forall p \neq q. \tag{8}$$

Here, d_{sep} denotes the minimum inter-UAV separation distance.

3.2.5. Communication Constraints

When multiple UAVs are operating collaboratively, the distance between any UAV and the cluster's central node must not exceed the maximum communication distance $R_{com} = 500$ m to ensure stable cluster communication:

$$\| p_p(h) - p_c(h) \|_2 \leq R_{com}. \tag{9}$$

Here, $p_c(h)$ is the location of the cluster center node.

3.2.6. Temporal Coordinated Constraints

For scenarios such as urban logistics distribution, the UAV must arrive at the target point within the specified time window, as shown in Equation (10):

$$t_p^{arr} \in [t_{start}, t_{end}], \forall p. \tag{10}$$

Here, t_p^{arr} represents the arrival time of the p-th UAV, and $[t_{start}, t_{end}]$ represents the specified time window.

3.3. Urban Multi-UAV Flight Energy Consumption Model

In urban environments, UAVs frequently accelerate, decelerate, turn, and adjust altitude. The flight energy consumption is directly related to the path length, smoothness, and the number of accelerations and decelerations. This paper constructs a multi-UAV energy consumption model, which is divided into level flight energy consumption, attitude adjustment energy consumption, and vertical takeoff and landing energy consumption.

3.3.1. Level Flight Energy Consumption

Level flight energy consumption is mainly affected by air resistance, as shown in Equation (11):

$$E_{fly} = (\frac{1}{2} C_D \rho S v^3 + \frac{mgv}{\eta}) t_{fly}, \tag{11}$$

where C_D represents the air resistance coefficient, S represents the frontal area, ρ represents the air density, v represents the level flight speed, m represents the mass of the UAV, g represents the gravitational acceleration, η represents the propeller efficiency, and t_{fly} represents the level flight time.

3.3.2. Attitude Adjustment Energy Consumption

Attitude adjustment energy consumption comes from the attitude changes during the turning, acceleration and deceleration processes of the UAV, and is positively correlated with path curvature and the number of acceleration and deceleration times. The calculation formula is shown in Equation (12):

$$E_{att} = k_{att} \sum_{s=1}^{M} |\kappa_s| + k_{acc} N_{acc}, \tag{12}$$

where k_{att} is the curvature energy consumption coefficient, κ_s is the curvature of the path at position s, k_{acc} is the acceleration and deceleration energy consumption coefficients, and N_{acc} is the number of acceleration and deceleration operations.

3.3.3. Vertical Takeoff and Landing Energy Consumption

Vertical takeoff and landing energy consumption is mainly affected by the drone's own weight and the height of takeoff and landing. The calculation formula is shown in Equation (13):

$$E_{vtol} = \frac{F_{vtol} H_{vtol}}{\eta_{vtol}}, \tag{13}$$

where T represents the vertical takeoff and landing thrust, H_{vtol} represents the total lift-off height, and η_{vtol} represents the vertical takeoff and landing efficiency.

The total flight energy consumption of the unmanned aircraft is shown in Equation (14):

$$E_{total} = E_{fly} + E_{att} + E_{vtol}. \tag{14}$$

3.4. Multi-UAV Cooperative Multi-Objective Cost Function

This paper constructs a combined centralized-distributed multi-objective cost function, which is divided into single-UAV cost and multi-UAV cooperative cost. The smaller the value of the objective function, the better the path planning effect. The cost function expression is as follows in Equation (15):

$$J = \frac{1}{N_u} \sum_{p=1}^{N_u} J_{single,p} + J_{coop}, \tag{15}$$

where N_u represents the number of UAVs, $J_{single,p}$ represents the individual cost of the p-th UAV, and J_{coop} represents the cost of multi-UAV collaboration.

3.4.1. Individual Cost Function

The individual cost function comprehensively considers the core requirements of drone flight, and its expression is as shown in Equation (16):

$$J_{single,p} = \omega_1 J_{energy,p} + \omega_2 J_{smooth,p} + \omega_3 J_{obs,p} + \omega_4 J_{air,p}, \tag{16}$$

where ω_1, ω_2, ω_3, ω_4 are the weight coefficients of energy consumption, path smoothness, obstacle avoidance risk, and airspace violation risk, respectively, and they satisfy $\omega_1 + \omega_2 + \omega_3 + \omega_4 = 1$. In this study, $\omega_1 = 0.3$, $\omega_2 = 0.2$, $\omega_3 = 0.3$, and $\omega_4 = 0.2$. This setting gives slightly higher priority to energy use and obstacle avoidance risk, while path smoothness and airspace violation penalty are also retained in the integrated cost. Airspace compliance is further enforced by the hard constraints in Section 3.2.

The calculation formulas for each sub-cost are as follows:

1. Flight energy consumption cost $J_{energy,p}$: Based on the normalization processing of the total flight energy consumption of the drone, the smaller the value, the lower the energy consumption. As shown in Equation (17):

$$J_{energy,p} = \frac{E_{total,p}}{E_{\max,p}}, \tag{17}$$

where $E_{total,p}$ represents the actual total energy consumption of the p-th UAV, and $E_{\max,p}$ represents the maximum theoretical energy consumption of this UAV.

2. Path smoothness cost $J_{smooth,p}$: Calculated based on the mean curvature of the path—the smaller the value, the higher the smoothness. As shown in Equation (18):

$$J_{smooth,p} = \frac{1}{M}\sum_{s=1}^{M} |\kappa_{s,p}|. \tag{18}$$

Here, M represents the number of path points, and $\kappa_{s,p}$ represents the curvature of the path of the p-th UAV at position s.

3. Obstacle Avoidance Risk Cost $J_{obs,p}$: This assesses the collision risk between the UAV and the obstacle. The smaller the value, the lower the risk, as shown in Equation (19):

$$\begin{aligned} J_{obs,p} &= \sum_{i=1}^{N_{obs}} c_r \exp(-\lambda_o d_{p,i}^{\min}) \\ d_{p,i}^{\min} &= \min \parallel p_{p,s} - o_i \parallel_2, \end{aligned} \tag{19}$$

where N_{obs} denotes the number of obstacles, $d_{p,i}^{\min}$ denotes the minimum distance between the path of the p-th UAV and the i-th obstacle, $\lambda_o > 0$ is the obstacle-risk decay coefficient, and $c_r > 0$ is a dimensionless risk-scaling coefficient. A smaller $d_{p,i}^{\min}$ leads to a larger obstacle-risk cost, while a larger distance makes the exponential penalty decay rapidly.

4. Airspace violation cost $J_{air,p}$: To assess the risk of UAVs violating airspace regulations, when entering prohibited or restricted flight zones, the cost will sharply increase, as shown in Equation (20):

$$J_{air,p} = k_{nf} N_{nf,p} + k_{rf} N_{rf,p}, \tag{20}$$

where k_{nf}, k_{rf} is the penalty coefficient for violations, $N_{nf,p}$ is the number of times an individual enters the restricted flight zone, and $N_{rf,p}$ is the number of violations in the restricted flight zone.

3.4.2. Multi-UAV Cooperative Cost Function

The multi-UAV cooperative cost function takes into account the collaborative constraints of multi-UAV flights, as shown in Equation (21):

$$J_{coop} = \gamma_1 J_{col} + \gamma_2 J_{comm} + \gamma_3 J_{time}, \tag{21}$$

where $\gamma_1, \gamma_2, \gamma_3$ are the weight coefficients of collision risk, communication interruption, and temporal coordination deviation, respectively, and they satisfy $\gamma_1 + \gamma_2 + \gamma_3 = 1$. In this paper, $\gamma_1 = 0.5$, $\gamma_2 = 0.3$, and $\gamma_3 = 0.2$ are set according to the priority of urban multi-UAV operation: collision avoidance is assigned the highest priority, communication maintenance is assigned the second priority, and arrival time coordination is treated as a soft coordination term.

1. Multi-UAV collision risk cost J_{col}: Evaluates the collision risk among multiple UAVs. When the distance is less than the safety distance, the cost rises sharply, as shown in Equation (22):

$$J_{col} = \sum_{h=1}^{N_t} \sum_{p=1}^{N_u-1} \sum_{q=p+1}^{N_u} c_{col} \exp(-\mu_{col} d_{pq}(h)), \tag{22}$$

where N_t denotes the number of discrete time steps, N_u denotes the number of UAVs, and $d_{pq}(h) = \parallel p_p(h) - p_q(h) \parallel_2$ is the Euclidean distance between the p-th UAV and the q-th UAV at the discrete flight step h. The coefficient $\mu > 0$ controls the decay rate of the collision-risk penalty, and $c_{col} > 0$ is the collision-risk scaling coefficient. This summation form evaluates each UAV pair only once at each time step.

2. Communication interruption cost J_{comm}: Evaluates the communication stability of the UAV cluster. A penalty cost is incurred when the communication distance is exceeded, as shown in Equation (23):

$$J_{comm} = k_{comm} N_{comm}, \tag{23}$$

where k_{comm} is the penalty coefficient for communication interruption, and N_{comm} is the number of communication interruptions.

3. Temporal coordination deviation cost J_{time}: Evaluates the coordination deviation of the arrival time of the UAV. The smaller the value, the higher the coordination accuracy. As shown in Equation (24):

$$J_{time} = \frac{1}{N_u} \sum_{p=1}^{N_u} \left| t_p^{arr} - t_{tar} \right|, \tag{24}$$

where t_{tar} represents the target arrival time.

3.4.3. Rationale for Additive Weighting and Weight Assignment

The present study uses an additive weighted form after normalization. This choice is practical rather than absolute.

One reason is readability. The cost terms describe different physical concerns, such as flight efficiency, safety margin, and coordination quality. After normalization, an additive form makes it easier to see which term is pushing the update at a given stage, which is helpful in debugging and in ablation analysis.

Another reason is stability. In trial calculations, multiplicative coupling made the overall score more sensitive to small changes in certain terms, especially when one term approached a very small value. That behavior was not helpful for the present path planning task.

The weights therefore express design preference, not universal optimality. In dense urban missions, a moderate detour can be accepted, but collisions, airspace violations, and communication breakdown cannot. For that reason, safety- and compliance-related terms are given higher priority than pure path-shortening terms.

A full Pareto analysis and a broader sensitivity study would be useful. They are not included here and remain part of the next step.

4. Spatio-Temporal Coordinated Improved Hybrid Multi-Strategy Dung Beetle Optimization Algorithm

4.1. Original Dung Beetle Optimization Algorithm

The DBO algorithm simulates four types of behaviors including rolling, dancing, reproduction, foraging, and stealing, corresponding to four types of individuals: dung beetle roller, breeding dung beetles, small dung beetles, and stealing dung beetles. The algorithm achieves global optimal search through the position update of different types. The core formula is as follows:

1. Dung beetle roller:

$$X_i(t+1) = X_i(t) + a \times k \times X_i(t) + r_1 \times (X_{worst} - X_i(t)), \tag{25}$$

where t represents the current iteration number, a is the natural coefficient (taking -1 or 1), k is the deflection coefficient ($0 < k \leq 0.2$), r_1 is a random number between 0 and 1, and X_{worst} is the current global worst position.

When encountering an obstacle, the dung beetle roller updates its position through a tangential dance:

$$X_i(t+1) = X_i(t) + \tan(\theta) \times |X_i(t) - X_i(t-1)|, \tag{26}$$

where θ represents the deflection angle.

2. Breeding dung beetles:

$$B_i(t+1) = X_{lbest} + r_1 \times (B_i(t) - Lb^*) + r_2 \times (B_i(t) - Ub^*), \tag{27}$$

where X_{lbest} represents the current local optimal position, r_1, r_2 represent two independent random vectors. UB, LB represent the upper and lower bounds of the search space, $Lb^* = \max(X_{lbest} \times (1-R), Lb)$, $Ub^* = \min(X_{lbest} \times (1+R), Ub)$, and $R = 1 - t/T_{\max}$, and $T_{\max}$ represents the maximum number of iterations.

3. Small dung beetles:

$$X_i(t+1) = X_i(t) + r_3 \times (X_i(t) - Lb^b) + r_4 \times (X_i(t) - Ub^b), \tag{28}$$

where $Ub^b = \min(X_{glbest} \times (1+R), Ub)$ and $Lb^b = \max(X_{gbest} \times (1-R), Lb)$, X_{gbest} represents the current best recorded solution. r_3 is a random vector following a normal distribution, r_4 is a random number between 0 and 1, and D represents the dimension of the optimization problem.

4. Stealing dung beetles:

$$X_i(t+1) = X_{gbest} + S \times randn(1, D) \times (\left| X_i(t) - X_{gbest} \right| + |X_i(t) - X_{lbest}|). \tag{29}$$

Here, S is a constant, X_{gbest} represents the current best recorded solution, and $randn(1, D)$ is a random vector following a normal distribution.

The original DBO algorithm has five core flaws in the urban multi-UAV scenario:

1. The initial population is randomly generated, without considering the constraints of the urban airspace. A large number of individuals fall within obstacles or restricted flight zones, resulting in low population effectiveness.
2. Fixed step-size iteration cannot balance the global collaboration of multiple UAVs and the local obstacle avoidance of individual UAVs. It is prone to getting stuck in local optima.
3. The fixed weight update of the dung beetle thief algorithm does not consider the constraints of multi-UAV collaboration. The algorithm stability is poor in high-dimensional scenarios.
4. There is no multi-UAV collaborative obstacle avoidance mechanism. During the iteration process, path collisions are prone to occur.
5. The static iteration mechanism cannot adapt to the dynamic obstacles in the city and cannot meet real-time planning requirements.

This paper addresses these flaws and proposes five core improvement strategies to construct the STC-IMSDBO algorithm.

4.2. Improvement Strategies

4.2.1. Airspace-Constrained Adaptive Chaotic Latin Hypercube Sampling

The random sampling of the original DBO algorithm cannot adapt to urban airspace constraints. Based on Latin hypercube sampling [45], this paper introduces logical chaotic mapping [46] and incorporates urban feasible spatial constraints to propose the spatial constraint adaptive chaotic Latin hypercube sampling (Airspace-Constrained Adaptive

Chaotic Latin Hypercube Sampling, AS-ACLHS) strategy, ensuring that the initial population is uniformly distributed within the feasible spatial domain and enhancing the effectiveness and diversity of the population.

The implementation steps of the AS-ACLHS strategy are as follows:

1. Determine the feasible spatial domain Ω of the city, removing all obstacles, no-fly zones, and restricted flight zones.
2. Divide each dimension of the D-dimensional feasible spatial domain into N equal probability sub-intervals, where N is the population size.
3. Generate logical chaotic sequences within each sub-interval to ensure the randomness and ergodicity of the sampling points:

$$z_{n+1} = \mu_c z_n (1 - z_n), \tag{30}$$

where μ_c is the Logistic chaotic coefficient.

4. Map the chaotic sequence to the feasible sub-interval of the corresponding region, and generate the sampling points for each dimension:

$$x_{ij} = L_{b,j} + \frac{z_{ij} + \pi_j(i) - 1}{N} \times (U_{b,j} - L_{b,j}), X_i \in \Omega_{\text{fea}}, \tag{31}$$

where x_{ij} represents the j-th coordinate of the i-th sampling point, and $U_{b,j}$, $L_{b,j}$ represent the upper and lower bounds of the j-th dimension, $\pi_j(i)$ denotes the random permutation index on the j-th dimension. $X_i \in \Omega_{\text{fea}}$ means that the generated individual is located in the feasible airspace after constraint projection.

5. For each sampling point of each dimension, random permutation is performed to form the initial population, ensuring that there is only one sampling point in each row and each column of the hypercube.

The AS-ACLHS strategy not only ensures that the initial population is uniformly distributed within the feasible space, but also enhances the randomness and diversity of the population through chaotic mapping, avoiding the generation of invalid individuals from the source and laying a good foundation for the global search of the algorithm.

4.2.2. Spatio-Temporal Coupled Variable Step Size Golden Sine Iteration

The fixed step size iteration of the original DBO algorithm cannot balance the global collaboration of multiple UAVs and the local obstacle avoidance of a single aircraft. Based on the golden sine algorithm [46], this paper introduces the spatial-temporal distance constraint of multiple UAVs and proposes the Spatio-Temporal Coupled Variable Step Size Golden Sine Iteration (ST-VSSGSI) strategy to replace the tangential dance behavior of the dung beetle in the original DBO algorithm. The search step size is dynamically adjusted according to the iteration stage, the fitness of a single aircraft, and the spatial-temporal distance between multiple UAVs to achieve an adaptive balance between global exploration and local exploitation.

The position update formula of the ST-VSSGSI strategy is shown in Equation (32):

$$X_i(t+1) = X_i(t) \times \left|\sin(R_1)\right| + \alpha_{\text{st}}(t) \times \sin(R_2) \times \left|\tau_1 X_{global-best}(t) - \tau_2 X_i(t)\right|, \tag{32}$$

where $R_1 \in [0, 2\pi]$ and $R_2 \in [0, \pi]$ are the golden sine angles, τ_1, τ_2 is the golden ratio coefficient, and $\alpha_{\text{st}}(t)$ is the time-space coupling variable step size.

1. Golden ratio coefficient:

$$\tau_1 = -\pi + \pi \times \frac{1+\sqrt{5}}{2}, \tau_2 = -\pi \times \frac{1+\sqrt{5}}{2}. \tag{33}$$

The golden ratio coefficient makes the search direction more regular, improving the convergence speed and accuracy of the algorithm.

2. Spatio-Temporal Coupling Variable Step Size:

The step size is dynamically adjusted based on the number of iterations, the fitness value of a single UAV, and the spatio-temporal distance between multiple unmanned aircraft. The calculation formula is shown in Equation (34):

$$\alpha_{\text{st}}(t) = \alpha_{\max} - (\alpha_{\max} - \alpha_{\min}) \times \frac{t}{T} \times \frac{f_{best}(t)}{f_i(t) + \varepsilon} \times \frac{1}{1 + \exp(-\beta \bar{d}_i(t))}. \tag{34}$$

In the formula, $\alpha_{\max}$, $\alpha_{\min}$ represents the maximum step size, $f_i(t)$ represents the fitness value of the i-th individual, $f_{best}(t)$ represents the optimal fitness value of the current population, $\bar{d}_i(t)$ represents the mean spatio-temporal distance between the i-th individual's drone and other UAVs, β represents the adjustment coefficient, and ε is a very small positive number, used to avoid division by zero.

In the early stage of iteration, the step size is larger, and the algorithm focuses on the global collaborative exploration of multiple UAVs, quickly narrowing the search range; in the later stage of iteration, the step size is smaller, and the algorithm focuses on the local obstacle avoidance development of a single drone, improving the convergence accuracy. At the same time, individuals with better fitness values have smaller step sizes, conducting fine searches near the optimal position; individuals with distances too close to other UAVs automatically increase the step size to expand the search space and avoid path collisions.

4.2.3. Multi-Factor Coupled Synergistic Adaptive Weight

In the original DBO algorithm, the fixed weight update of the thief dung beetle does not consider the collaborative constraints of multiple UAVs and has poor stability in high-dimensional scenarios. This paper proposes the Multi-Factor Coupled Synergistic Adaptive Weight (MFCSAW) strategy, which takes into account the number of iterations, fitness value, population diversity, and the cost of multi-UAV collaboration, optimizing the weight update mechanism of the thief dung beetle and improving the stability of the algorithm in high-dimensional multi-UAV scenarios.

The updated position formula of the stealing dung beetle after the optimization of the MFCSAW strategy is as follows:

$$X_i(t+1) = X_{gbest}(t) + w(t) \times randn(1, D) \times \left| X_i(t) - X_{glest}(t) \right|, \tag{35}$$

where $w(t)$ represents the multi-factor coupled adaptive weight, and the calculation formula is:

$$w(t) = w_{\max} - (w_{\max} - w_{\min}) \times [\lambda_1 \frac{t}{T_{\max}} + \lambda_2 \frac{f_{avg}(t)}{f_i(t)} + \lambda_3 \frac{1}{\mathcal{D}(t)} + \lambda_4 \frac{J_{coop}(t)}{J_{coop,\max}}], \tag{36}$$

where $w_{\max}$, $w_{\min}$ represent the maximum and minimum weight, $f_{avg}(t)$ represents the mean fitness value of the current population, $\mathcal{D}(t)$ represents the diversity of the current population, $J_{coop}(t)$ represents the cost of multi-UAV collaboration, $J_{coop,\max}$ represents the maximum collaboration cost, and $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ represent the weight coefficient of each factor, which satisfies $\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1$.

The calculation formula for population diversity $\mathcal{D}(t)$ is:

$$\mathcal{D}(t) = \frac{1}{N \times D} \sum_{j=1}^{D} \sum_{i=1}^{N} (X_{ij}(t) - \overline{X_j(t)})^2, \tag{37}$$

where $\overline{X_j(t)}$ represents the mean value of the j-th dimension of the population.

The MFCSAW strategy dynamically adjusts the weight through four factors:

1. As the number of iterations increases, the weight gradually decreases, enhancing the local development ability of the algorithm.
2. Individuals with better fitness values have smaller weights, conducting fine searches near the optimal position.
3. When the population diversity is low, the weight automatically increases, expanding the search space and avoiding getting stuck in local optima.
4. When the cost of multi-UAV collaboration is high, the weights automatically increase, optimizing the multi-UAV collaboration path and reducing the risk of collision.

The weight adjustment based on multiple factors enables the algorithm to have better adaptability and stability in the multi-UAV high-dimensional scenarios of urban environments.

4.2.4. Distributed Cooperative Obstacle Avoidance Non-Cooperative Game Position Update

To reduce path conflicts among multiple UAVs during iterative search, this paper introduces a non-cooperative game-based position-update mechanism for distributed cooperative obstacle avoidance, referred to as NCGPU. The purpose of this mechanism is practical. Instead of checking inter-UAV conflicts only after a complete candidate path has been generated, the coordination step is inserted into the iterative refinement process, so that separation can be considered while the path is still being updated [47].

It should be emphasized that the game formulation used here is local rather than global. In dense urban environments, the full multi-UAV trajectory space is usually non-convex because of buildings, no-fly zones, altitude restrictions, and inter-UAV separation constraints. For that reason, the equilibrium statement in this section is made for locally projected feasible updates around the current path control points, rather than for the full global trajectory space. This treatment is closer to the actual implementation of the algorithm and avoids overstating the theoretical scope of the result.

Multi-UAV Obstacle Avoidance Non-Cooperative Game Model

The local cooperative obstacle avoidance process is modeled as a non-cooperative game:

$$G = \{P, S, U\},$$

where P is the player set, $S = \{S_p\}_{p=1}^{N_u}$ is the local strategy set family, and $U = \{U_p\}_{p=1}^{N_u}$ is the utility function family.

1. Players

The UAVs participating in the cooperative mission are treated as the players in the game. The player set is written as:

$$P = \{1, 2, \ldots, N_u\},$$

where N_u denotes the number of UAVs.

2. Strategy Space

For the p-th UAV, the strategy variable is denoted by s_p and the corresponding strategy set is denoted by S_p. Here, s_p represents a feasible local path update action after projection

onto the current admissible neighborhood of the path control points. In other words, the strategy is not the full global trajectory itself, but a bounded local adjustment around the current path.

The projected strategy set S_p must satisfy the current geometric bounds, obstacle avoidance requirements, airspace constraints, and inter-UAV separation conditions.

3. Utility function

The utility of the p-th UAV is defined as the negative of its local total cost, namely

$$U_p(s_p, s_{-p}) = -J_p(s_p, s_{-p}), \tag{38}$$

where s_{-p} denotes the current strategy combination of all UAVs except the p-th UAV, and J_p contains both the individual path-related cost and the cooperative cost associated with interaction with other UAVs. Under this definition, maximizing the utility is equivalent to minimizing the local total cost while keeping the update feasible.

This formulation allows each UAV to revise its local path update with awareness of the current actions of other UAVs, while still preserving a distributed update structure.

Existence of a Local Nash Equilibrium

The equilibrium discussion in this paper is made under a local projected setting. To keep the scope of the statement clear, the assumptions are listed explicitly.

Definition 1. *A strategy combination*

$$s^* = (s_1^*, s_2^*, \ldots, s_{N_u}^*)$$

is called a Nash equilibrium of the local projected game if, for any player (p) and any feasible local strategy $s_p \in S_p$,

$$U_p(s_p^*, s_{-p}^*) \geq U_p(s_p, s_{-p}^*),$$

where s_{-p}^* *represents the equilibrium strategies of all UAVs other than the p-th UAV.*

Theorem 1. *If the projected strategy set* S_p *of each UAV is nonempty, compact, and convex, and if the local utility function* U_p *is continuous and quasi-concave with respect to the player's own strategy* s_p*, then the local projected multi-UAV obstacle avoidance game admits at least one Nash equilibrium* [47].

Proof. The proof follows the standard fixed-point argument for non-cooperative games.

1. The player set P is finite because the number of UAVs in the cooperative mission is finite.
2. For each UAV, the strategy set S_p is constructed from a locally projected feasible neighborhood. Under the projection used in the present algorithm, this neighborhood is treated as nonempty, bounded, and closed, and is further approximated as convex during the local update step. Therefore, S_p can be regarded as a nonempty compact convex subset of a Euclidean space.
3. The utility function U_p is defined as the negative of the local total cost J_p. If J_p is continuous and convex with respect to the player's own projected update, then U_p is continuous and quasi-concave with respect to s_p.

Under these assumptions, the Debreu-Glicksberg-Fan fixed-point theorem [48,49] supports the existence of at least one Nash equilibrium for the local projected game. The proof is thus completed in the local projected sense. □

This result should be interpreted in a limited way. It supports the existence of an equilibrium in the local projected update problem under the stated assumptions. It does not imply that the full nonconvex multi-UAV trajectory problem always admits a globally optimal collision-free equilibrium.

Approximate Equilibrium Position Update Mechanism

Based on the above local game model, a bounded distributed iterative process is used to obtain an approximate equilibrium update for path refinement. The role of this process is practical. It is used to reduce inter-UAV conflicts during iterative search, rather than to provide a complete global solution theory.

The implementation procedure is as follows:

1. At the current outer iteration, each UAV takes the present local strategies of the other UAVs as given and computes its own best local response within the projected feasible neighborhood.
2. All UAVs revise their local path updates and then re-check the geometric, airspace, and inter-UAV separation constraints.
3. The local game iteration stops when either the update change becomes sufficiently small or the preset inner-loop limit is reached.

The result obtained in this way is an approximate local equilibrium used for cooperative path refinement. In the present framework, its function is to improve inter-UAV separation and reduce path conflicts during iterative search. It should not be interpreted as a proof that all multi-UAV conflicts are theoretically eliminated under every urban condition.

4.2.5. Dynamic Environment Rolling Horizon Replanning Strategy

To meet the real-time planning requirements for urban dynamic obstacles, this paper proposes the Rolling Horizon Replanning for Dynamic Environment (RHRDE) strategy. This strategy divides the entire flight process of the UAV into multiple equal-length receding horizon windows. In each window, combined with the real-time sensed dynamic obstacle information, the STC-IMSDBO algorithm is used for local path replanning to achieve real-time path updates in the dynamic environment [50].

The implementation steps of the RHRDE strategy are as follows:

1. Initialize the length of the receding horizon window T_w, the prediction horizon T_p, and the control time-domain T_c, ensuring that $T_c \leq T_w \leq T_p$.
2. At the initial time, use the STC-IMSDBO algorithm to plan the global initial path.
3. At the start time of each rolling window, use the onboard sensors to perceive the real-time position and speed information of the dynamic obstacles, and update the environment model.
4. Starting from the current position of the UAV, using the global path points within the prediction time-domain as sub-goals, perform local path replanning within the control time-domain using the STC-IMSDBO algorithm to obtain the optimal local path.
5. Execute the optimal path within the control time-domain, enter the next rolling window, and repeat steps 3–5 until the UAV reaches the destination.

The rolling horizon module is used to preserve local feasibility when the environment changes. It should be interpreted as a practical replanning device rather than a proof of global closed-loop optimality. In each window, only the local path segment affected by updated obstacle information is replanned, while the global route is retained as a reference. Its real-time performance still depends on sensing delay, onboard computation, and the size of the local planning window.

4.3. Pseudocode of the STC-IMSDBO

To make the workflow easier to follow, this section provides two pieces of pseudocode. Algorithm 1 describes the main global planning loop of the STC-IMSDBO. Algorithm 2 shows how the planner is called again in the rolling-horizon stage when local replanning is needed in a dynamic environment.

Algorithm 1. Global path planning procedure of STC-IMSDBO

Input: urban three-dimensional environment model Ω, number of UAVs N_u, set of origin and destination coordinates, population size N, maximum number of iterations T, optimization dimension D, upper and lower bounds of search space U_b, L_b, cost function weight coefficients ω and γ, core parameters of the improvement strategy (τ, λ, β, etc.), objective function $J(\cdot)$.
Output: Set of best recorded collaborative paths for multiple UAVs X_{best}, optimal fitness value $Best_{FF}$.
Set the parameters related to the STC-IMSDBO algorithm (such as drone flight constraints, maximum number of game iterations, etc.).
1: Initialize the algorithm parameters, environmental constraints, and drone flight constraints.
2: Eliminate obstacles and no-fly zones from the environment Ω and delineate the feasible airspace.
3: Adopt the adaptive chaotic Latin hypercube sampling (AS-ACLHS) strategy based on the airspace constraints, generating the initial population $Pop(0)$ through Equations (30) to (31), with each individual corresponding to the path control point of a drone.
4: Use cubic B-spline interpolation to smooth the initial path of each drone and calculate the fitness values of all individuals through the objective function.
5: Set $Best_{FF} = \min$ (fitness value set), and the optimal solution corresponds to X_{best}, with the iteration number t = 0.
6: while $t \leq T$ do
7: t = t + 1.
8: Divide the individual groups of dung-rolling dung beetles, breeding dung beetles, small dung beetles, and opportunistic dung beetles
9: //The position of the dung beetle is updated using the time-space coupling variable-step-size golden sine iterative (ST-VSSGSI) strategy.
10: for i = 1 to Number of rolling dung beetles, do
11: The time-space coupling variable step size α_{st}(t) is calculated through Equations (33) to (34).
12: Update the individual's new position Xi_{new} through Equation (32).
13: Verify whether Xi_{new} meets all the constraints. If not, update it again.
14: end for
15: //Update the positions of breeding dung beetles and small dung beetles.
16: for i = 1 to total number of breeding dung beetles and small dung beetles do
17: Update the individual's new position Xi_{new} by Equations (27) to (28).
18: Verify whether Xi_{new} meets all the constraints. If not, update it again.
19: end for

Algorithm 1. *Cont.*

20: //update the position of the thief dung beetle using the multi-factor coupled synergistic adaptive weight (MFCSAW) strategy.
21: The current population's mean fitness, population diversity $\mathcal{D}(t)$, and multi-UAV collaboration cost can be calculated through Equations (36) to (37).
22: for i = 1 to Number of stealing dung beetle do
23: Calculate the multi-factor coupled adaptive weights $w(t)$ through Equation (36)
24: Update the individual's new position Xi_{new} through Equation (35).
25: Verify whether Xi_{new} meets all the constraints. If not, update it again.
26: end for
27: //Complete the optimization of multi-computer cooperative game with distributed Cooperative Obstacle Avoidance Non-cooperative Game Update (NCGPU) mechanism
28: Initialize the number of game iterations k = 0 and set the maximum number $K_{\max}$ of game iterations
29: while $k < K_{\max}$ do
30: for p = 1 to N_u do
31: Solve optimal response strategy for the p-th UAV based on the current positions of other UAVs
32: Update path position of the p-th UAV, and minimize the total cost function
33: end for
34: if the strategies of all UAVs are unchanged and the local update change is below the preset threshold then
35: end if
36: k = k + 1
37: end while
38: //Boundary process, fitness evaluation and update optimal solution
39: for i = 1 to N do
40: To perform boundary process on Xi_{new}, calculate the new individual fitness value by the objective function $objfun$
41: if a better solution is found then synchronize X_{best} and $Best_{FF}$
42: end for
43: end while
44: smooth X_{best} with Cubic B-spline interpolation
45: Return X_{best} and $Best_{FF}$.

Algorithm 2. Rolling-horizon replanning procedure of STC-IMSDBO in dynamic environments

Input: global reference path set χ_{g}^{*} current UAV-state set $X_{cur}(t)$, rolling-horizon parameters (rolling horizon T_r, prediction horizon T_p, control horizon T_c), replanning iteration number $T_{\max}^{loc}$ dynamic obstacle set $O_{dyn}(t)$ perceived by onboard sensors in real time, urban environment model Ω, core parameters of STC-IMSDBO algorithm (the same as Algorithm 1), objective function $J(\cdot)$.
Output: updated local path set χ_{loc}^{*}, real-time UAV control command.

Algorithm 2. *Cont.*

1: Initialize rolling-horizon parameters and set t = 0, set $arrive_{flag} = false$.
2: while $arrive_{flag} = false$ do
3: Obtain the current dynamic obstacle information $O_{dyn}(t)$ from onboard sensors.
4: Update the urban environment model Ω(t) and delimit the current feasible airspace.
5: Set the local planning start state as $X_{cur}(t)$.
6: Extract local sub-goals from χ_g^* within the prediction horizon T_p.
7: Call Algorithm 1 to solve the local replanning problem within the control horizon T_c, using $T_{\max}^{loc}$ as the local iteration limit.
8: Obtain the updated local path set χ_{loc}^*.
9: Execute the first control segment of χ_{loc}^* within T_c.
10: Update the UAV-state set $X_{cur}(t)$.
11: t = t + T_c
12: if all UAVs reach their assigned terminal states then
13: $arrive_{flag} = true$
14: end if
15: end while
16: Return χ_{loc}^* and real-time UAV control instructions.

4.4. STC-IMSDBO Algorithm Implement Steps

The STC-IMSDBO algorithm is designed for 3D path planning of multiple UAVs in constrained urban airspace. The algorithm flowchart is shown in Figure 2. The implementation steps are as follows:

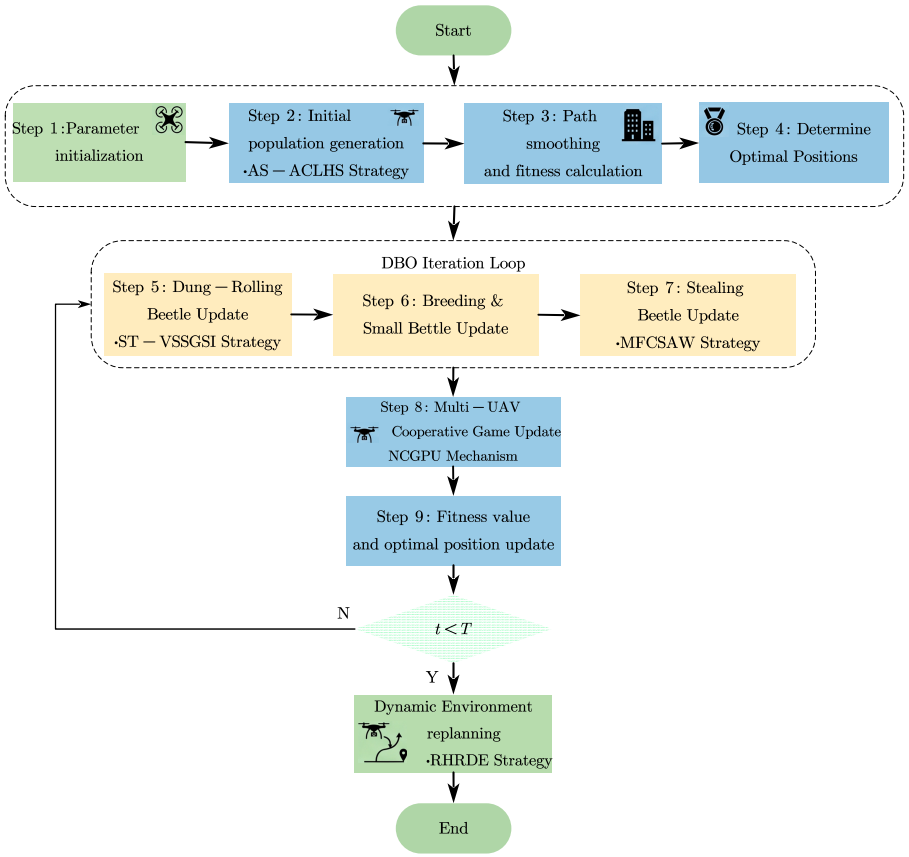


Figure 2. Flowchart of the STC-IMSDBO algorithm.

Step 1: Parameter initialization.

Set the population size N, the maximum number of iterations T, the optimization dimension D, search bounds, cost-function weights, improvement-strategy parameters, the number of UAVs N_u, and urban environment parameters.

Step 2: Initial population generation.

Use the AS-ACLHS strategy to generate the initial population in the feasible urban airspace. Each individual represents a candidate set of path control points for the UAV fleet.

Step 3: Path smoothing and fitness evaluation.

Apply cubic B-spline interpolation to smooth each candidate path and calculate its fitness using the normalized multi-objective cost function.

Step 4: Optimal position determination.

Rank the population according to fitness values and record the current best-so-far solution, local best solution, and worst solution.

Step 5: Position update of rolling dung beetle.

Update the position of dung beetle roller through ST-VSSGSI strategy, judge whether all constraints are met; if not, update again.

Step 6: Position update of breeding and small dung beetles.

Update the breeding and small dung beetle individuals according to the original DBO update rules, followed by the same feasibility check.

Step 7: Position update of stealing dung beetles.

Update the position of stealing dung beetles through MFCSAW strategy to determine whether the constraints are met; update it again if the constraints are not met.

Step 8: Multi-UAV cooperative game update.

Apply the NCGPU mechanism in the projected feasible neighborhood to obtain an approximate local equilibrium update. This step is used to reduce inter-UAV conflict risk during iterative path refinement.

Step 9: Fitness value and optimal position update.

Recalculate the fitness values of all individuals and update the best recorded solution, local optimal position and global worst position.

Step 10: Convergence judgment.

Determine whether the number of iterations reaches the maximum number of iterations T, if so, output the multi-UAV cooperative path corresponding to the best recorded solution; if not, return to Step 5 and continue the iteration.

Step 11: Dynamic environment replanning.

During the actual flight, local path replanning is carried out in each rolling horizon through the RHRDE strategy to adapt to the urban dynamic obstacles.

4.5. Computational Complexity

The computational cost of the STC-IMSDBO mainly comes from four parts: population initialization, fitness evaluation, iterative position update, and the bounded inner cooperative game.

In the initialization stage, N individuals are generated in a D-dimensional feasible space, so the basic cost is O(ND).

Fitness evaluation depends on path smoothing and constraint checking over sampled path points. If M denotes the number of path samples used in one fitness calculation, then the population-level evaluation cost can be written as O(NM).

The position-update stage includes the four DBO subgroups and the local cooperative update. If the inner game loop is bounded by K_g, and the number of UAVs in the cooperative mission is denoted by N_u, the pairwise interaction checking in the local game can be upper-bounded by $O\left(K_g N_u^2\right)$. Updating the best-so-far solution requires O(N). Here, the $K_g N_u^2$ term assumes that the local cooperative game is applied only to the current

collaborative path update in each outer iteration. If the same game update is applied to every population individual, this term becomes $O\left(NK_gN_u^2\right)$.

Therefore, for T outer iterations, the computational complexity of the static global planning stage can be written in compact form as

$$O\left(T\left(ND + NM + K_gN_u^2\right)\right).$$

Under a fixed scenario scale, where D, M, K_g and N_u are treated as bounded design quantities, the dominant order with respect to the population size N remains linear. In this limited asymptotic sense, the static global planning stage is in the same complexity class as the original DBO. This statement should not be interpreted as meaning that the practical runtime is identical.

In actual computation, the cooperative game, separation checking, and rolling-horizon replanning increase the constant factor. This effect becomes more visible as the fleet size grows. Therefore, the present analysis is used to clarify the order of growth of the planner, not to replace direct wall-clock benchmarking for larger UAV swarms.

4.6. Convergence Discussion Under Idealized Search Assumptions

Since the STC-IMSDBO is implemented with a finite iteration budget, a probability-one global convergence result would be too strong for the present planner. For this reason, the discussion in this section is limited to an asymptotic interpretation of the stochastic search dynamics under idealized assumptions.

4.6.1. Markov Interpretation of the Stochastic Search

Because the population at iteration t + 1 depends on the current population together with the random variables generated in that iteration, the outer search loop can be interpreted as a stochastic process in the population state space. A Markov interpretation is therefore useful for describing the search evolution [51]; however, its role here is still limited. It is mainly used to explain why the algorithm can continue visiting different feasible regions while retaining the best solution found so far.

4.6.2. Asymptotic Improvement Interpretation

If the randomized initialization and update operators preserve a nonzero probability of entering feasible regions, and if the best-so-far solution is retained monotonically, then the recorded best fitness forms a non-increasing sequence over iterations. In this sense, the search can be understood as having an asymptotic improvement tendency.

This statement does not mean that every finite run reaches the global optimum. It only suggests that the search can continue improving without discarding better feasible solutions once they have been found.

Accordingly, no rigorous finite-time global-optimality claim is made for the present planner in this paper.

5. Simulation Experiments and Result Analysis

5.1. Experimental Environment Setup

The simulation experiments were carried out in MATLAB R2023a on a workstation equipped with an Intel Core i9-12900H CPU, 64 GB RAM, and the Windows 10 operating system. To evaluate the method under different levels of urban complexity, four three-dimensional low-altitude scenarios were constructed, ranging from low-complexity suburban-like space to dense urban airspace with dynamic obstacles. The corresponding parameters are listed in Table 1.

Table 1. Parameter Settings for Four Simulation Scenarios.

Simulation Area (X,Y,Z)	Number of UAVs	Start Points	Target Points	Number of Buildings	Number of No-Fly Zones	Number of Dynamic Obstacles	Complexity
100 × 100 × 200	2	(10,10,30), (20,20,30)	(80,90,80), (90,80,80)	5	2	0	Low
200 × 200 × 300	4	(10,10,30), (20,20,30), (10,20,30), (20,10,30)	(180,190,100), (190,180,100), (180,180,100), (190,190,100)	10	4	0	Medium
300 × 300 × 500	8	Uniformly distributed in the lower left corner of the area	Uniformly distributed in the upper right corner of the area	20	8	0	High
500 × 500 × 600	16	Uniformly distributed in the lower left corner of the area	Uniformly distributed in the upper right corner of the area	30	12	10	Ultra-high

The four scenarios represent low-, medium-, high-, and ultra-high-complexity urban environments, respectively. They are used to test the planner under increasing levels of obstacle density, airspace restriction, fleet size, and dynamic interference. In this way, the experiments do not only compare average performance in a single map, but also examine how the method behaves as the planning difficulty grows.

To evaluate the effectiveness of the STC-IMSDBO, six representative path planning methods were selected for comparison: multi-UAV Adaptive Chaotic Gray Wolf Optimization (MACGWO) [31], MDBO [52], multi-UAV Whale Optimization (MWOA) [9], multi-agent proximal strategy Optimization (MAPPO) [53], Multi-Agent Deep Deterministic Policy Gradient (MADDPG) [54], and Multi-UAV Hyper-heuristic Whale Optimization (MHHWOA) [55]. These methods were chosen because they cover three different planning paradigms, including improved swarm intelligence methods, the original DBO baseline, and representative multi-agent reinforcement learning approaches. This combination makes the comparison less dependent on one algorithm family alone.

The reinforcement learning baselines were evaluated using their trained policies under the same scenario settings. Their role in the comparison is to provide a representative learning-based reference, rather than to imply that training cost is directly comparable to search-based runtime in one planning episode.

For fairness, all stochastic planners were independently executed 50 times in each scenario, and the reported results were computed from these repeated runs. The population size and the maximum iteration budget were kept identical across the population-based methods. For each comparison algorithm, the remaining control parameters followed the settings recommended in the corresponding source paper. No additional scenario-specific manual tuning was introduced after feasibility checks.

To obtain a more complete view of the planning quality, nine indicators were used in the comparison: mean path length, convergence iteration count, path smoothness, mean fitness, mean energy consumption, multi-UAV collision rate, airspace violation rate, temporal coordination error, and dynamic obstacle avoidance success rate. These indicators were selected to cover not only path efficiency, but also cooperative safety, regulatory compliance, and mission coordination quality.

In addition to the mean value comparison, a nonparametric statistical test is further reported in Section 5.4 to check whether the main performance differences are likely to be random.

5.2. Comparative Experiments

5.2.1. Comparison of Path Length and Energy Consumption

The mean path length results of all methods in the four scenarios are listed in Table 2. Path length is used here as a direct indicator of route efficiency, but it should be read together with feasibility, smoothness, and safety-related metrics rather than in isolation.

Table 2. Comparison of mean path lengths of each algorithm (unit: m).

Algorithms	Low Complexity	Medium Complexity	High Complexity	Ultra-High Complexity	Mean Shortening Ratio
STC-IMSDBO	126.37	289.52	512.46	986.37	—
MACGWO	165.29	386.41	702.35	1325.64	23.71%
MDBO	138.95	318.76	564.82	1086.93	9.26%
MWOA	160.38	375.29	735.61	1545.72	36.24%
MAPPO	140.26	322.48	576.39	1125.84	10.12%
MHHWOA	134.25	307.63	544.27	1061.25	6.12%
MADDPG	144.37	331.54	585.46	1126.93	12.47%

As shown in Table 2, the STC-IMSDBO provides the shortest mean feasible path in all four tested scenarios, with values of 126.37, 289.52, 512.46, and 986.37 m from the low-complexity case to the ultra-high-complexity case. The advantage is relatively modest in the simpler environment, but becomes more visible as the scene grows denser and the number of UAVs increases. For example, in the low-complexity scenario, the mean path length is 21.21% shorter than that of the MWOA, while in the ultra-high-complexity scenario, the reduction reaches 36.24%. This pattern suggests that the proposed planner is better able to control detours caused by congestion, constraint interaction, and cooperative separation in harder scenes.

The result should still be read with some care. A shorter mean path does not by itself imply that the planner is better in every respect. Its value is clearer when it appears together with acceptable safety, smoothness, and energy performance, which is why the following indicators are reported separately.

Figure 3 gives a visual comparison of the mean path lengths in the four scenarios. The purpose of the figure is descriptive. It helps show how the gap between methods changes as the environment becomes more complex.

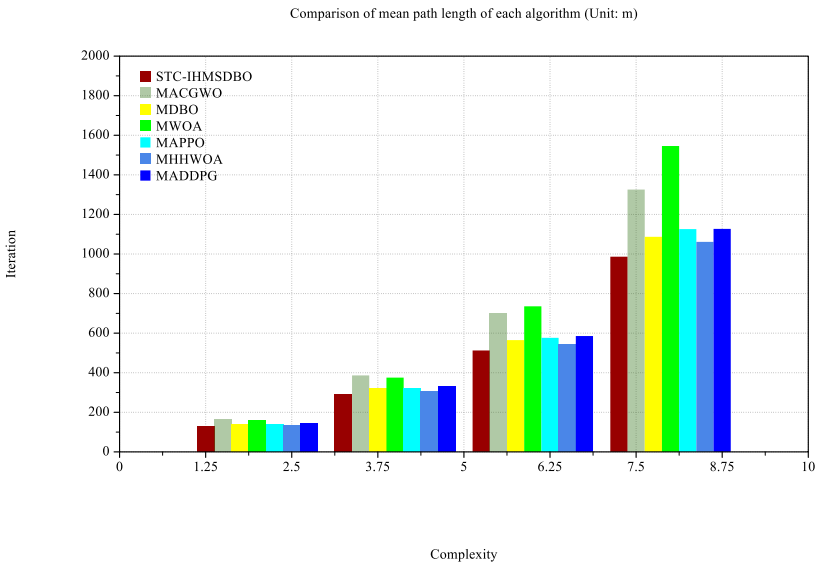


Figure 3. Bar chart comparing mean path lengths.

Table 3 reports the mean flight energy consumption of the compared methods. Unlike path length, the energy metric reflects not only travel distance, but also trajectory smoothness and maneuvering demand.

Table 3. Comparison of mean flight energy consumption of each algorithm (unit: kJ).

Algorithms	Low Complexity	Medium Complexity	High Complexity	Ultra-High Complexity	Mean Energy Consumption Reduction Rate
STC-IMSDBO	826.54	1758.32	3246.75	6584.29	—
MACGWO	1025.78	2246.59	4325.68	8765.34	24.88%
MDBO	934.26	1987.45	3654.21	7458.62	11.72%
MWOA	1086.35	2358.74	4587.93	9845.76	33.12%
MAPPO	942.58	2015.36	3726.54	7685.43	14.33%
MHHWOA	896.47	1887.62	3487.25	7158.36	8.02%
MADDPG	965.32	2058.47	3845.69	7854.21	16.17%

Table 3 shows the same trend in all four scenarios. The STC-IMSDBO always gives the lowest mean energy consumption, with values of 826.54, 1758.32, 3246.75, and 6584.29 kJ from the low-complexity case to the ultra-high-complexity case. Compared with the six reference methods, the reduction ranges from 8.02% to 33.12%. This result is close to the path length comparison, but it is not a simple repetition of it. In the present tests, the lower energy use appears to be related to shorter feasible routes and fewer unnecessary local corrections during cooperative updates.

Taken together, Tables 2 and 3 suggest that the STC-IMSDBO improves route efficiency without achieving that gain through unstable or excessively costly maneuvers. The next subsection examines whether a similar pattern also appears in convergence behavior. Figure 4 provides a visual comparison of the mean flight energy consumption across the four scenarios.

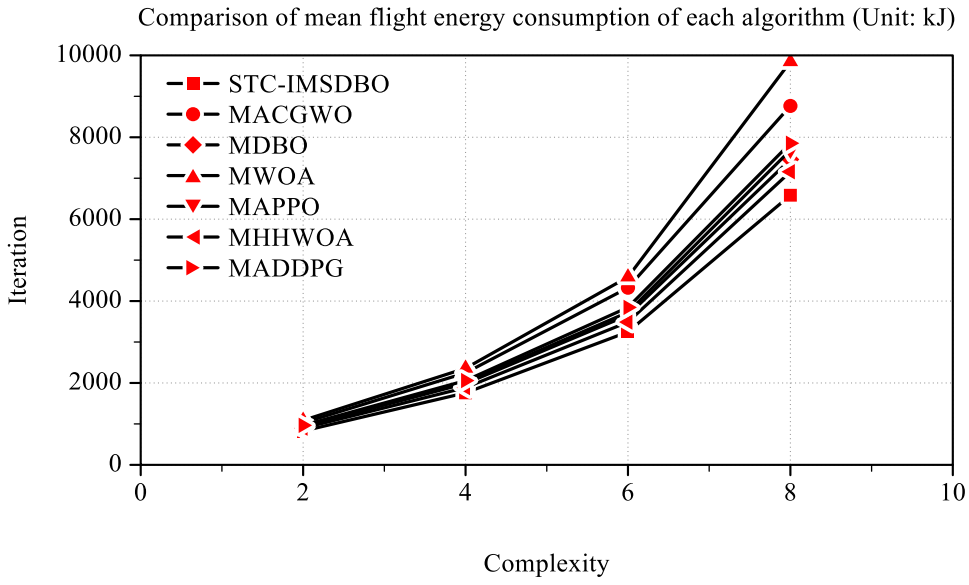


Figure 4. Line chart comparing mean flight energy consumption.

5.2.2. Convergence Speed Comparison

The convergence iteration results of all methods are listed in Table 4. Here, convergence speed is described by the number of iterations required to reach the best recorded solution under the preset stopping condition. A smaller iteration count indicates that the search enters a stable high-quality region earlier.

Table 4. Comparison of convergence iteration number for each algorithm.

Algorithms	Low Complexity	Medium Complexity	High Complexity	Ultra-High Complexity	Mean Number of Convergence
STC-IMSDBO	38	59	82	108	71.75
MACGWO	58	85	124	162	107.25
MDBO	52	78	112	145	96.75
MWOA	65	96	142	185	122
MAPPO	55	82	118	152	101.75
MHHWOA	46	68	102	132	87
MADDPG	57	84	121	158	105

As reported in Table 4, the STC-IMSDBO requires the fewest iterations in all four tested scenarios, with convergence counts of 38, 59, 82, and 108, and a mean value of 71.75. Compared with the six reference methods, the reduction in the mean convergence count ranges from 14.10% to 39.47%. In the low-complexity scenario, the STC-IMSDBO converges in 38 iterations, which is 41.54% fewer than the MWOA. In the ultra-high-complexity scenario, it converges in 108 iterations, still 41.62% fewer than MWOA.

This result suggests that the proposed update mechanism helps the search reach a useful feasible region earlier and reduces unnecessary late-stage wandering. The effect becomes clearer as the environment becomes more constrained, where cooperative interaction and obstacle structure make inefficient search more costly.

The result should not be read as a full runtime benchmark. Fewer iterations usually help practical efficiency, but wall-clock time is also influenced by fitness evaluation, inner cooperative updates, and implementation details. For this reason, the convergence count comparison is used here as an algorithmic efficiency indicator rather than a direct substitute for runtime profiling.

Figure 5 gives a visual view of the convergence iteration comparison. It is used to show how the gap between methods changes across the four scenarios, especially when the planning problem shifts from relatively open space to dense urban coordination.

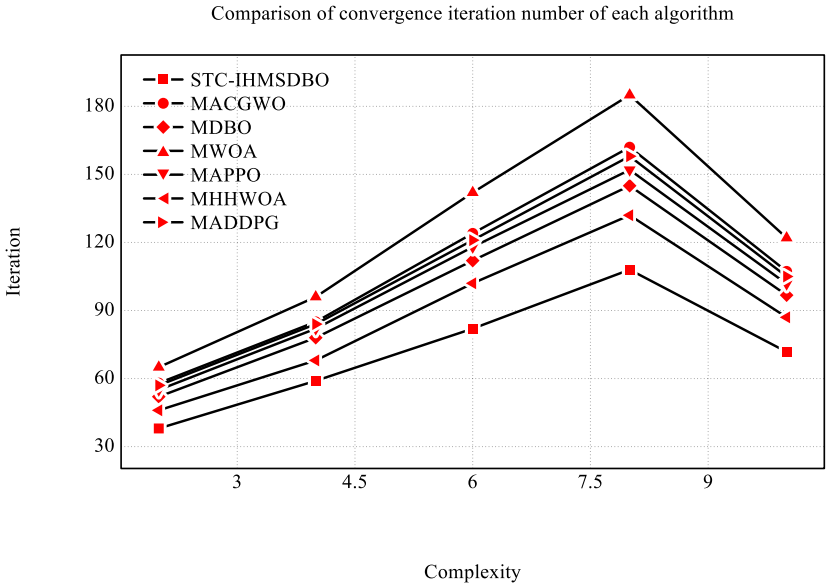


Figure 5. Line chart comparing convergence iteration number.

5.2.3. Comparison of Path Smoothness and Comprehensive Fitness

The comparison results of path smoothness and mean fitness are listed in Table 5. In this study, path smoothness is evaluated by the mean curvature of the UAV trajectories,

while mean fitness reflects the overall optimization result under the normalized multi-objective cost formulation.

Table 5. Comparison of path smoothness and mean fitness of various algorithms.

Algorithm	Mean Curvature (1/m)	Curvature Reduction Rate	Mean Fitness Value	Fitness Reduction Rate
STC-IMSDBO	0.0135	—	0.218	—
MACGWO	0.0204	33.82%	0.306	28.76%
MDBO	0.0176	23.30%	0.264	17.42%
MWOA	0.0227	40.53%	0.335	34.93%
MAPPO	0.0182	25.82%	0.276	21.01%
MHHWOA	0.0158	14.56%	0.243	10.29%
MADDPG	0.0189	28.57%	0.282	22.70%

As shown in Table 5, the STC-IMSDBO gives the lowest mean curvature among all compared methods, with a value of 0.0135. Relative to the six reference methods, the curvature reduction ranges from 14.56% to 40.53%. This indicates that the generated routes are smoother on average, especially when compared with the MWOA, MACGWO, and MADDPG.

The same table also shows that the STC-IMSDBO obtains the lowest mean fitness value, namely 0.218. Compared with the other six methods, the reduction in mean fitness ranges from 10.29% to 34.93%. Since the fitness value in this paper combines route efficiency, smoothness, obstacle avoidance, airspace compliance, and cooperative cost, this result suggests that the advantage of the STC-IMSDBO is not limited to one isolated metric.

These two indicators should also be read together rather than separately. A planner may reduce path length while still producing routes with larger curvature or higher integrated cost. In the present experiments, the STC-IMSDBO performs better on both metrics at the same time, which is consistent with the earlier results on path length and energy consumption.

Figure 6 gives a visual comparison of mean curvature and mean fitness. The figure is used to show the relative gap among methods more directly, while the detailed numerical comparison is still given by Table 5.

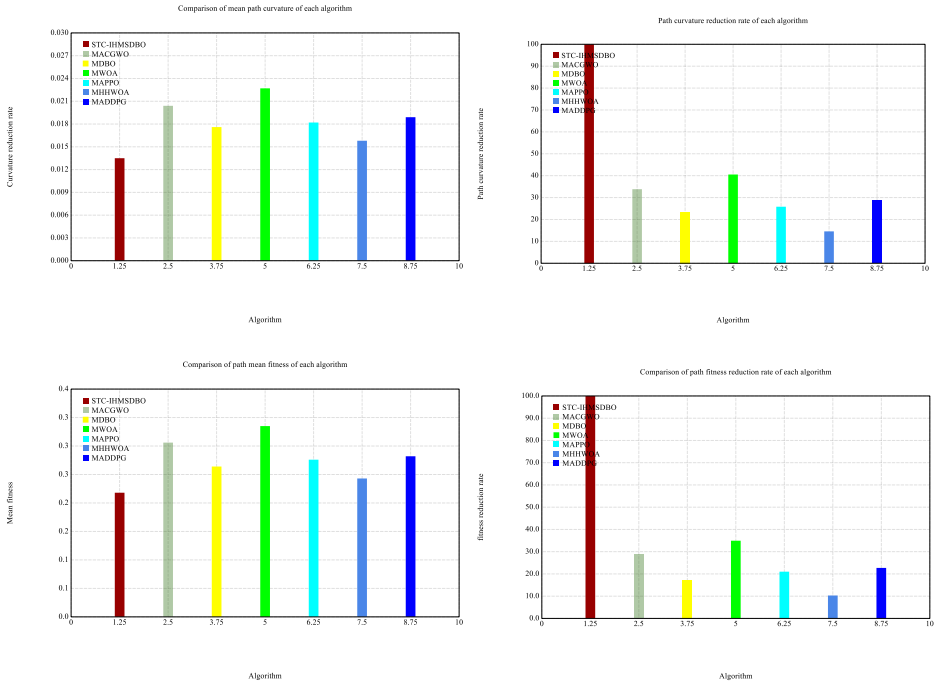


Figure 6. Bar chart comparing path smoothness and mean fitness.

5.2.4. Comparison of Multi-UAV Cooperative Performance

The comparison results of multi-UAV cooperative performance are listed in Table 6. Three indicators are considered here: multi-UAV collision rate, airspace violation rate, and temporal coordination error. Together, they reflect whether the planner can maintain separation, satisfy regulatory constraints, and support coordinated arrival behavior at the same time.

As reported in Table 6, the STC-IMSDBO records a collision rate of 0%, an airspace violation rate of 0%, and a temporal coordination error of 0.62 s. In contrast, the six reference methods show collision rates ranging from 2.2% to 12.6%, airspace violation rates from 3.4% to 15.2%, and temporal coordination errors from 1.08 to 2.84 s. Among them, MHHWOA is the closest to the STC-IMSDBO in this comparison, with values of 2.2%, 3.4%, and 1.08 s, while MWOA shows the largest deviation, with 12.6%, 15.2%, and 2.84 s.

These results indicate that the STC-IMSDBO performs more consistently against the cooperative indicators in the reported simulations. The zero recorded collision and zero recorded violation rates are encouraging, but they should be read as empirical results in the present test set rather than as a universal theoretical guarantee. The relatively small temporal coordination error also suggests that the proposed planner handles fleet-level timing more effectively under the current cost and constraint setting.

The result is better understood when the three indicators are viewed together. A planner may reduce collision events but still show larger timing deviation, or it may preserve timing coordination while violating airspace rules more often. In the present experiments, the STC-IMSDBO improves all three indicators at the same time, which is why its cooperative performance appears more balanced than that of the comparison methods.

Table 6 therefore supports the practical value of the proposed cooperative update design under the current simulation settings, especially in terms of maintaining separation and regulatory feasibility while preserving acceptable coordination accuracy. Figure 7 provides a visual comparison of the multi-UAV cooperative performance indicators.

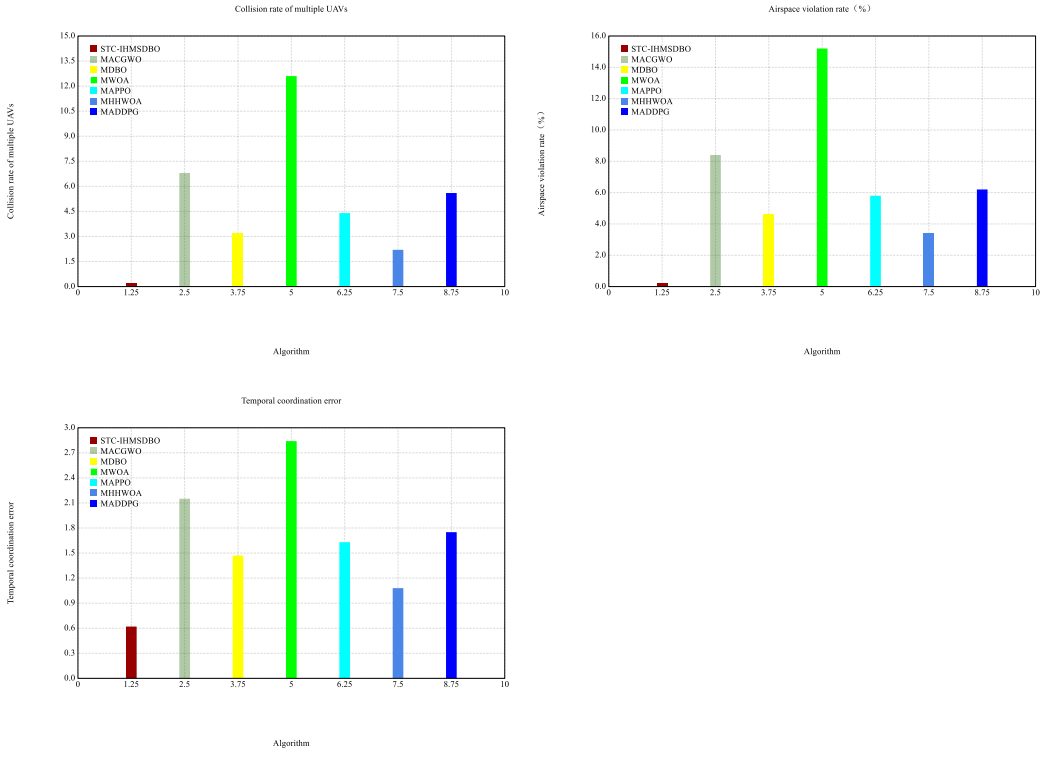


Figure 7. Bar chart comparing multi-UAV cooperative performance.

Table 6. Comparison of multi-UAV cooperative performance.

Algorithm	Collision Rate of Multiple UAVs (%)	Airspace Violation Rate (%)	Temporal Coordination Error (s)
STC-IMSDBO	0	0	0.62
MACGWO	6.8	8.4	2.15
MDBO	3.2	4.6	1.47
MWOA	12.6	15.2	2.84
MAPPO	4.4	5.8	1.63
MHHWOA	2.2	3.4	1.08
MADDPG	5.6	6.2	1.75

5.2.5. Comparison of Obstacle Avoidance Performance in Dynamic Environments

The comparison results of dynamic obstacle avoidance performance in the ultra-high-complexity scenario are listed in Table 7. Three indicators are reported here: success rate of dynamic obstacle avoidance, mean path length, and mean flight energy consumption. Together, they are used to check whether the planner can avoid moving obstacles without paying an excessive price in route efficiency and flight cost.

Table 7. Comparison of dynamic obstacle avoidance performance.

Algorithm	Success Rate of Dynamic Obstacle Avoidance (%)	Mean Path Length (m)	Mean Flight Energy Consumption (kJ)
STC-IMSDBO	100	986.37	6584.29
MACGWO	86.4	1325.64	8765.34
MDBO	92.6	1086.93	7458.62
MWOA	82.2	1545.72	9845.76
MAPPO	94.8	1125.84	7685.43
MHHWOA	96.2	1061.25	7158.36
MADDPG	93.4	1126.93	7854.21

As shown in Table 7, the STC-IMSDBO achieves a dynamic obstacle avoidance success rate of 100%, together with a mean path length of 986.37 m and a mean flight energy consumption of 6584.29 kJ. In contrast, the success rates of the six reference methods range from 82.2% to 96.2%, the mean path lengths range from 1061.25 to 1545.72 m, and the mean flight energy consumption ranges from 7158.36 to 9845.76 kJ. Among them, MHHWOA is the closest to the STC-IMSDBO in this scenario, with a success rate of 96.2%, a mean path length of 1061.25 m, and a mean energy consumption of 7158.36 kJ, whereas MWOA shows the weakest result, with 82.2%, 1545.72 m, and 9845.76 kJ.

These results suggest that the STC-IMSDBO is better able to maintain feasible local replanning when dynamic obstacles appear in a dense urban scene. The advantage is not reflected in the success rate alone. In the present test, the method also preserves a shorter path and lower energy use after replanning, which means that the obstacle avoidance gain is not obtained by simply accepting large detours or overly expensive maneuvers.

The result should still be interpreted as an empirical outcome under the current simulation setting. The 100% success rate is encouraging, but it does not by itself imply a universal guarantee for all dynamic environments. Within the present setup, the result is consistent with the role of the rolling-horizon replanning strategy, which allows local path correction after new obstacle information becomes available.

Figure 8 provides a visual comparison of dynamic obstacle avoidance performance. Its role is descriptive. The detailed numerical differences are given in Table 7.

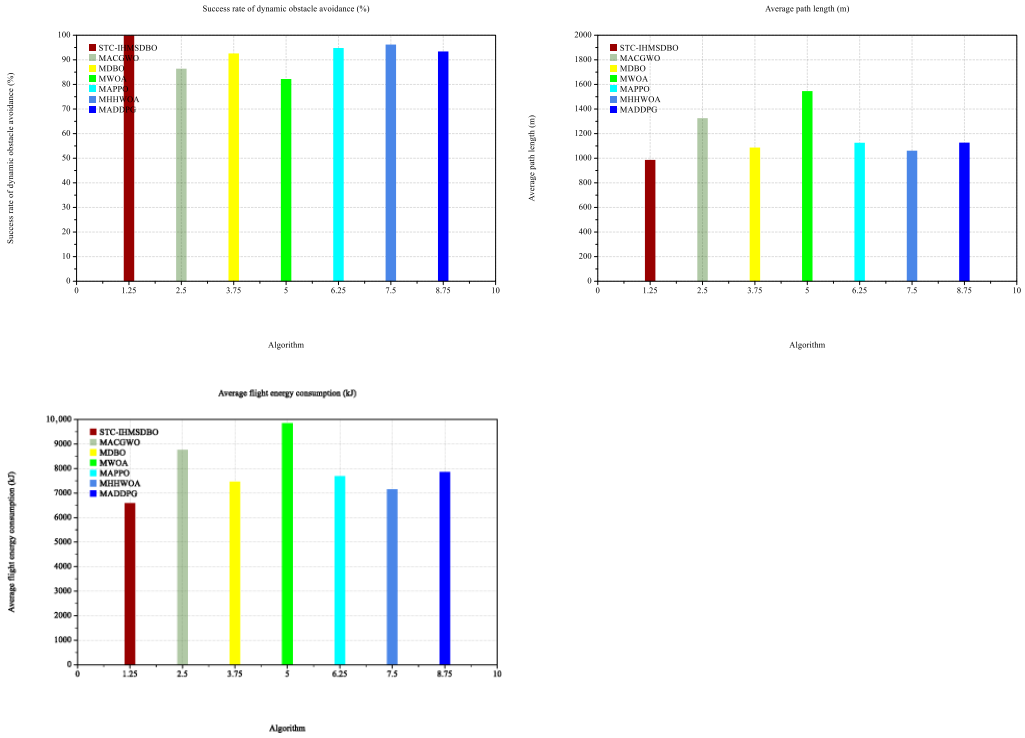


Figure 8. Bar chart comparing dynamic obstacle avoidance performance.

5.3. Ablation Experiment

To examine the contribution of each improvement module, an ablation experiment was conducted in the high-complexity scenario. The original MDBO was used as the baseline. Five single-module variants, one two-module variant, and the full STC-IMSDBO model were then compared under the same evaluation indicators, including path length, convergence iteration count, mean curvature, fitness value, and multi-UAV collision rate. The results are reported in Table 8.

Table 8. Comparison of ablation experiment results.

Algorithms	Path Length (m)	Number of Convergence	Mean Curvature (1/m)	Fitness Value	Collision Rate (%)
Original MDBO	564.82	112	0.0176	0.264	3.2
MDBO+AS-ACLHS	552.37	106	0.0172	0.256	2.4
MDBO+ST-VSSGSI	541.26	94	0.0158	0.248	2.2
MDBO+MFCSAW	556.48	102	0.0169	0.252	2.6
MDBO+NCGPU	548.35	108	0.0173	0.251	0
MDBO+RHRDE	560.27	110	0.0174	0.258	3.0
MDBO+AS-ACLHS+ST-VSSGSI	528.46	88	0.0152	0.237	1.8
STC-IMSDBO (Overall strategy)	512.46	82	0.0135	0.218	0

Table 8 shows that each single-module variant improves at least part of the original MDBO result. Compared with original MDBO, whose path length is 564.82 m, convergence count is 112, mean curvature is 0.0176, fitness value is 0.264, and collision rate is 3.2%, the single-module variants reduce one or more of these indicators. This suggests that the five modules are not merely decorative additions, although their effects are not identical.

The AS-ACLHS variant mainly improves early search quality. Its path length decreases from 564.82 to 552.37 m, and the collision rate decreases from 3.2% to 2.4%. This is consistent with its role in generating more feasible initial individuals in constrained urban airspace.

The ST-VSSGSI variant gives the clearest improvement in convergence speed and smoothness among the single-module variants. The convergence count decreases from 112 to 94, while the mean curvature decreases from 0.0176 to 0.0158. These changes correspond to reductions of 16.07% and 10.23%, respectively. This result is consistent with the function of the variable-step update, which adjusts the search scale during different iteration stages.

The MFCSAW provides gives a moderate but stable improvement. Its path length is 556.48 m, the convergence count is 102, and the fitness value decreases from 0.264 to 0.252. This indicates that adaptive weighting mainly helps stabilize the search and improve the integrated cost, rather than acting on one single indicator alone.

The NCGPU variant has the most direct effect on cooperative safety. Its collision rate decreases from 3.2% to 0%, while the path length and fitness value are also reduced to 548.35 m and 0.251, respectively. This result should be interpreted as an empirical outcome in the current ablation setting, not as a universal proof that all possible multi-UAV conflicts are eliminated.

The RHRDE variant shows a smaller effect in this static high-complexity ablation scene, with a path length of 560.27 m, a convergence count of 110, and a collision rate of 3.0%. This is expected, because the main role of RHRDE is dynamic replanning rather than static-map optimization. Its effect is more visible in the dynamic obstacle experiment reported in Section 5.2.5.

The two-module variant, MDBO+AS-ACLHS+ST-VSSGSI, performs better than all single-module variants. Its path length is 528.46 m, convergence count is 88, mean curvature is 0.0152, fitness value is 0.237, and collision rate is 1.8%. This result indicates that feasible initialization and variable-step search can support each other: a better initial population reduces invalid exploration, while the variable-step update improves later refinement.

The full STC-IMSDBO model gives the best overall result in this ablation group. Compared with original MDBO, the path length decreases from 564.82 to 512.46 m, the convergence count decreases from 112 to 82, the mean curvature decreases from 0.0176 to 0.0135, and the fitness value decreases from 0.264 to 0.218. The corresponding reductions are 9.27%, 26.79%, 23.30%, and 17.42%, respectively. The collision rate also decreases from 3.2% to 0%.

These results suggest that the five modules act on different bottlenecks of the planning process. AS-ACLHS mainly affects initialization quality, ST-VSSGSI affects search speed and smoothness, MFCSAW improves integrated cost stability, NCGPU affects cooperative safety, and RHRDE is mainly useful when dynamic replanning is required. The full model benefits from combining these effects, but the result should still be understood within the tested high-complexity scenario.

5.4. Statistical Test

To further examine whether the observed performance differences are likely to be random, a Wilcoxon rank-sum test was applied in the high-complexity scenario. Four indicators were tested: path length, convergence iteration count, mean curvature, and fitness value. The significance level was set to $\alpha = 0.05$. If the p-value is smaller than 0.05, the difference between the STC-IMSDBO and the corresponding comparison method is considered statistically significant under the current test setting. The results are listed in Table 9.

As shown in Table 9, all p-values are below 0.05. For MACGWO, the p-values are 0.011, 0.007, 0.013, and 0.008 for path length, convergence count, mean curvature, and fitness value, respectively. For MDBO, the corresponding values are 0.019, 0.015, 0.022, and 0.017. For MWOA, the p-values are smaller, namely 0.004, 0.002, 0.005, and 0.003.

Table 9. Wilcoxon rank-sum test results ($\alpha = 0.05$).

Comparison Algorithms	Path Length (p Value)	Number of Convergence (p Value)	Mean Curvature (p Value)	Fitness Value (p Value)	Overall Differences
MACGWO	0.011 (+)	0.007 (+)	0.013 (+)	0.008 (+)	Significantly superior
MDBO	0.019 (+)	0.015 (+)	0.022 (+)	0.017 (+)	Significantly superior
MWOA	0.004 (+)	0.002 (+)	0.005 (+)	0.003 (+)	Significantly superior
MAPPO	0.016 (+)	0.012 (+)	0.019 (+)	0.014 (+)	Significantly superior
MHHWOA	0.031 (+)	0.027 (+)	0.034 (+)	0.029 (+)	Significantly superior
MADDPG	0.014 (+)	0.011 (+)	0.017 (+)	0.013 (+)	Significantly superior

Note: (+) indicates that STC-IMSDBO is statistically significantly superior to the corresponding comparison algorithm at the significance level of $\alpha = 0.05$.

The same pattern is also observed for the learning-based and hybrid comparison methods. The p-values for MAPPO are 0.016, 0.012, 0.019, and 0.014, while those for MADDPG are 0.014, 0.011, 0.017, and 0.013. MHHWOA is the closest competitor in this test, with p-values of 0.031, 0.027, 0.034, and 0.029, but these values are still below the 0.05 threshold.

These results support the statistical significance of the observed differences in the selected high-complexity scenario. The difference between the STC-IMSDBO and MWOA is the most pronounced among the reported comparisons, while the difference between the STC-IMSDBO and MHHWOA is smaller but still statistically significant. This test reduces the likelihood that the reported performance gap is caused only by random variation in repeated runs. It should not be interpreted as a substitute for broader runtime or scaling validation across larger UAV swarms.

5.5. Digital Twin Simulation Test for Real City Scenarios

To further examine the method in a more realistic urban layout, a digital twin simulation scenario was constructed using topographic data from the Lujiazui core area in Shanghai. The test space covers 1000 m $\times$ 1000 m $\times$ 600 m and contains 42 buildings, eight no-fly zones, six restricted flight zones, and eight UAVs. The task setting follows an urban logistics delivery scenario. The UAV parameters are set with reference to the DJI M300 RTK platform.

The STC-IMSDBO was compared with MDBO, MWOA, and MHHWOA in this digital twin scenario. The tested indicators include total path length, obstacle avoidance success rate, runtime, mean flight energy consumption, and minimum safe distance. The numerical results are listed in Table 10.

As shown in Table 10, STC-IMSDBO obtains the shortest total path length, with a value of 1288.7 m. The corresponding values of MDBO, MWOA, and MHHWOA are 1584.2, 1512.6, and 1425.8 m, respectively. This suggests that the proposed planner reduces detours in the dense Lujiazui map, especially when compared with the original DBO baseline.

The runtime of the STC-IMSDBO is 8.45 s, while MDBO, MWOA, and MHHWOA require 18.52, 14.35, and 12.18 s, respectively. Relative to MDBO, the reduction in runtime is about 54.4%. This result indicates that the improved search strategy can reduce planning time in the reported digital twin test. It should still be interpreted as a simulation-level runtime result, not as a complete onboard real-time validation.

In terms of obstacle avoidance and safety margin, the STC-IMSDBO records an obstacle avoidance success rate of 99.8% and a minimum safe distance of 20.6 m. The corresponding success rates of MDBO, MWOA, and MHHWOA are 72.4%, 81.5%, and 91.2%, while their minimum safe distances are -3.2 m, 6.4 m, and 12.8 m, respectively. The negative value for MDBO indicates model penetration in this test case. These results suggest that proposed constraint-handling and cooperative update design can better maintain geometric feasibility in the dense urban map.

Table 10. Digital twin simulation results in the Lujiazui urban scenario.

Performance Indicator	MDBO	MWOA	MHHWOA	STC-IMSDBO
Total path length (m)	1584.2	1512.6	1425.8	1288.7
Obstacle avoidance success rate (%)	72.4%	81.5%	91.2%	99.8%
Runtime (s)	18.52	14.35	12.18	8.45
Mean flight energy consumption (kJ)	142.5	133.8	115.4	96.2
Minimum safe distance (m)	−3.2 (Severe model penetration)	6.4	12.8	20.6

The energy result follows the same general trend. The STC-IMSDBO gives a mean flight energy consumption of 96.2 kJ, lower than MDBO, MWOA, and MHHWOA, whose values are 142.5, 133.8, and 115.4 kJ, respectively. This improvement is consistent with the shorter path and smoother cooperative updates observed in the previous experiments.

Overall, the Lujiazui digital twin experiment supports the practical relevance of the STC-IMSDBO in a dense urban layout. At the same time, this test should not be viewed as a substitute for real flight validation. Sensor delay, wind disturbance, GNSS degradation, communication loss, and onboard computing limits are not fully represented in the current simulation. For this reason, the digital twin result is best understood as an intermediate validation step between synthetic benchmark scenarios and future hardware-in-the-loop or flight experiments.

Figure 9 is a radar chart comparison of the main indicators in the Lujiazui digital twin scenario. The chart summarizes the results in terms of path length, runtime, obstacle avoidance, energy consumption, and safety margin. It is used as a visual summary of Table 10, rather than as independent evidence of real-world deployment capability.

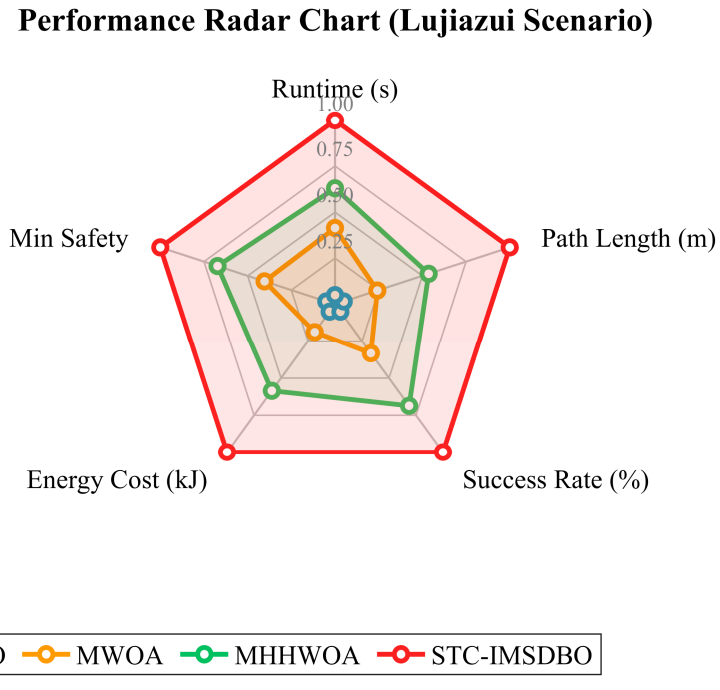
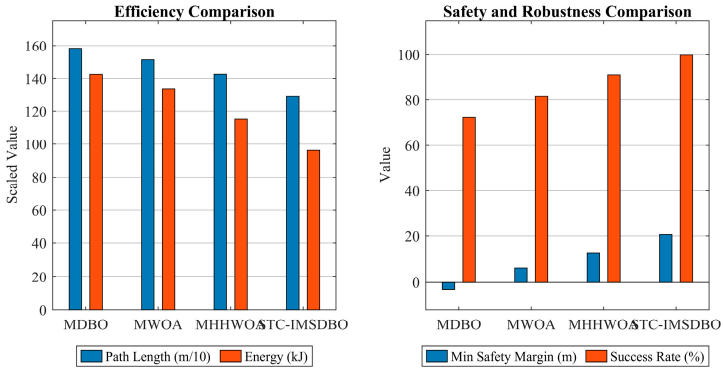
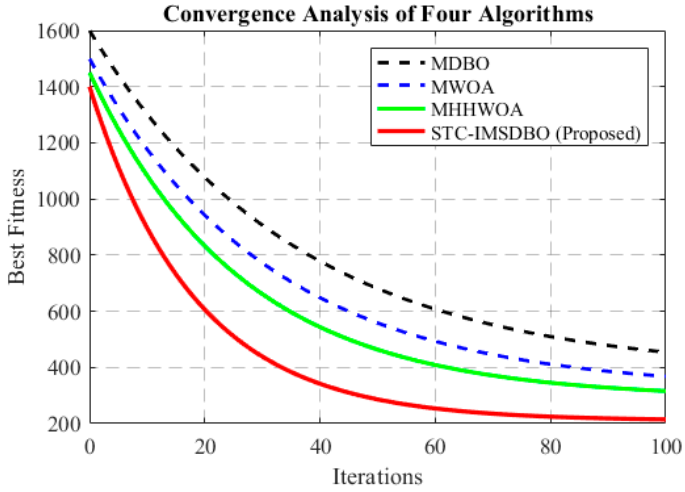


Figure 9. Radar chart.

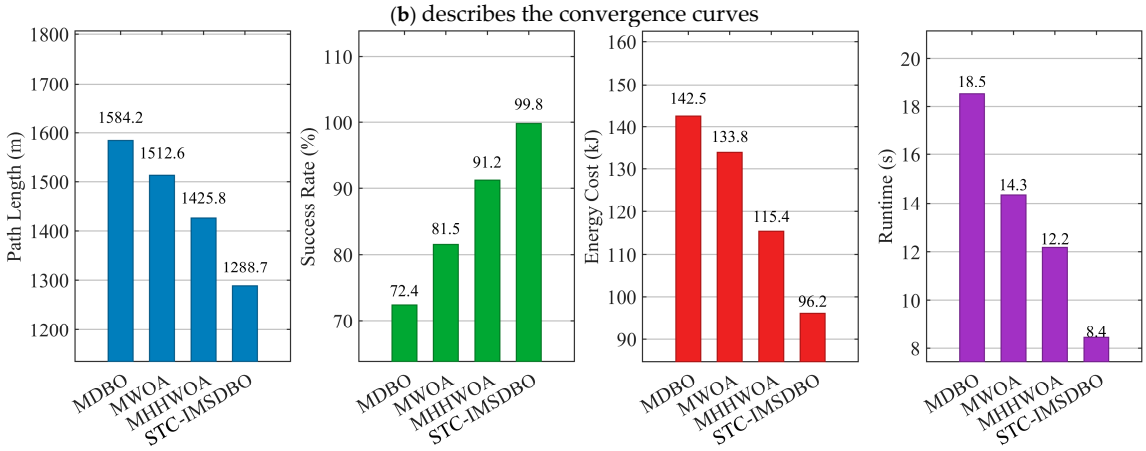
Figure 10 provides an additional visual comparison of the performance indicators. It helps show the relative differences among the four methods, while the main quantitative evidence is still provided by Table 10. Therefore, this figure should be interpreted as a supporting visualization.



(**a**) describes the bar chart comparison of efficiency and safety metrics



(**b**) describes the convergence curves



(**c**) describes the multi-indicator bar chart comparison

Figure 10. Performance comparison chart.

The CEC2017 results in Section 5.6 provide supplementary evidence for the numerical search stability of the STC-IMSDBO. They should not be used as a direct explanation of the Lujiazui runtime or obstacle avoidance performance. The results in Table 10 remain the main evidence for the digital twin scenario, while the CEC2017 benchmark only supports the general optimization behavior of the proposed search mechanism.

Figure 11 shows the multi-UAV paths generated by the four algorithms in the Lujiazui digital twin map. Compared with the reference methods, the STC-IMSDBO produces routes with fewer visible detours and better separation from dense building regions. This visual result is consistent with the shorter path length and larger minimum safe distance reported in Table 10.

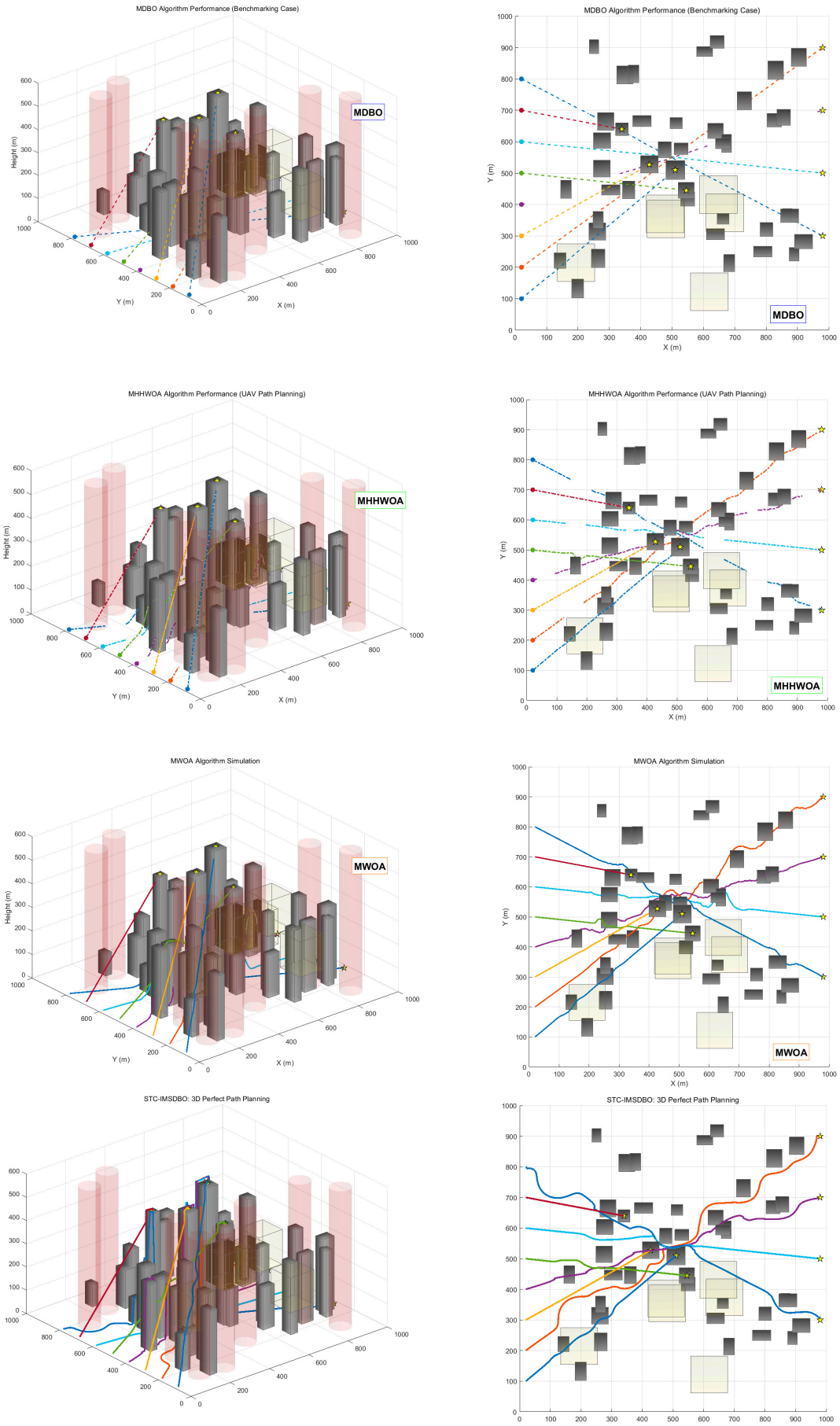


Figure 11. Multi-UAV path planning by four algorithms.

5.6. CEC 2017 Benchmark

The CEC2017 benchmark was used as a supplementary test to examine the general numerical search behavior of the STC-IMSDBO. This test is different from the urban multi-UAV path planning experiments in Sections 5.2–5.5. It does not evaluate flight safety or dynamic obstacle avoidance directly. Its role is to check whether the improved search mechanism still maintains stable optimization performance on standard nonlinear benchmark functions.

To keep the main text focused on the UAV planning problem, only representative CEC2017 results are reported in this section. The selected functions cover unimodal, multimodal, and composition cases. The full benchmark definitions follow the standard CEC2017 setting.

5.6.1. Experimental Setup

The benchmark experiments were conducted in MATLAB R2023a on the same workstation used in the previous simulations. The optimization dimension was set to ($D = 30$), and the search range followed the standard CEC2017 configuration. Seven algorithms were compared: MAPPO [53], MADDPG [54], MWOA [9], MDBO [52], MACGWO [31], MHHWOA [55], and STC-IMSDBO.

Each selected benchmark function was independently run 30 times. The mean value and standard deviation of the final fitness values were recorded. A smaller mean value indicates better optimization accuracy, while a smaller standard deviation indicates better stability across repeated runs.

5.6.2. Test Results and Analysis

The representative CEC2017 results are listed in Table 11. The STC-IMSDBO obtains lower mean values and smaller standard deviations than the comparison methods on the selected functions. For example, on the unimodal function F1, the STC-IMSDBO obtains $9.59 \times 10^1/1.34 \times 10^1$, while MHHWOA, the closest reference method, gives $7.68 \times 10^2/6.89 \times 10^1$. On F3, the corresponding values are $2.79 \times 10^2/1.53 \times 10^1$ for the STC-IMSDBO and $2.27 \times 10^3/3.33 \times 10^2$ for MHHWOA.

Table 11. Representative CEC2017 benchmark results (D = 30, Mean/Std).

Function	Type	MAPPO	MADDPG	MWOA	MDBO	MACGWO	MHHWOA	STC-IMSDBO
F1	Unimodal	$1.13 \times 10^5/1.03 \times 10^4$	$8.76 \times 10^4/5.16 \times 10^3$	$1.49 \times 10^4/2.22 \times 10^3$	$5.64 \times 10^3/7.96 \times 10^2$	$2.53 \times 10^3/3.25 \times 10^2$	$7.68 \times 10^2/6.89 \times 10$	$9.59 \times 10/1.34 \times 10$
F3	Unimodal	$3.27 \times 10^5/3.09 \times 10^4$	$2.43 \times 10^5/2.10 \times 10^4$	$4.68 \times 10^4/6.65 \times 10^3$	$1.63 \times 10^4/1.01 \times 10^3$	$7.10 \times 10^3/8.11 \times 10^2$	$2.27 \times 10^3/3.33 \times 10^2$	$2.79 \times 10^2/1.53 \times 10$
F4	Multimodal	$4.77 \times 10^5/5.64 \times 10^4$	$3.44 \times 10^5/4.07 \times 10^4$	$6.02 \times 10^4/5.83 \times 10^3$	$2.28 \times 10^4/2.30 \times 10^3$	$1.01 \times 10^4/1.00 \times 10^3$	$3.01 \times 10^3/2.86 \times 10^2$	$4.35 \times 10^2/3.53 \times 10$
F6	Multimodal	$7.91 \times 10^5/7.62 \times 10^4$	$4.68 \times 10^5/5.43 \times 10^4$	$9.31 \times 10^4/1.02 \times 10^4$	$3.66 \times 10^4/2.19 \times 10^3$	$1.65 \times 10^4/1.88 \times 10^3$	$4.55 \times 10^3/6.41 \times 10^2$	$5.60 \times 10^2/4.71 \times 10$
F30	Composition	$3.36 \times 10^6/3.52 \times 10^5$	$2.45 \times 10^6/1.55 \times 10^5$	$4.50 \times 10^5/3.65 \times 10^4$	$1.94 \times 10^5/2.78 \times 10^4$	$8.18 \times 10^4/9.05 \times 10^3$	$2.54 \times 10^4/3.64 \times 10^3$	$3.08 \times 10^3/1.66 \times 10^2$

Similar behavior appears in the multimodal functions. On F4, the STC-IMSDBO records $4.35 \times 10^2/3.53 \times 10^1$, compared with $3.01 \times 10^3/2.86 \times 10^2$ for MHHWOA. On F6, the STC-IMSDBO gives $5.60 \times 10^2/4.71 \times 10^1$, while MHHWOA gives $4.55 \times 10^3/6.41 \times 10^2$. These results suggest that the proposed search operators can maintain useful exploration and refinement ability in nonconvex landscapes.

The composition functions are more difficult because several shifted and rotated landscapes are combined. In the retained representative result F30, the STC-IMSDBO obtains $3.08 \times 10^3/1.66 \times 10^2$, while MHHWOA gives $2.54 \times 10^4/3.64 \times 10^3$, MACGWO

gives $8.18 \times 10^4 / 9.05 \times 10^3$, and MDBO gives $1.94 \times 10^5 / 2.78 \times 10^4$. The smaller standard deviation of the STC-IMSDBO in this case indicates more stable repeated run behavior under the selected benchmark setting.

However, these benchmark functions only test numerical optimization behavior. They should not be interpreted as direct evidence of urban flight safety, real-time deployability, or dynamic obstacle avoidance capability. The main evidence for those aspects is still provided by the urban multi-UAV simulations in Sections 5.2–5.5.

Figure 12 shows representative convergence curves on selected CEC2017 functions rather than the full benchmark suite. The selected functions cover different landscape types, including unimodal, multimodal, and composition cases. This reduced figure is easier to read and avoids compressing too many small plots into one panel.

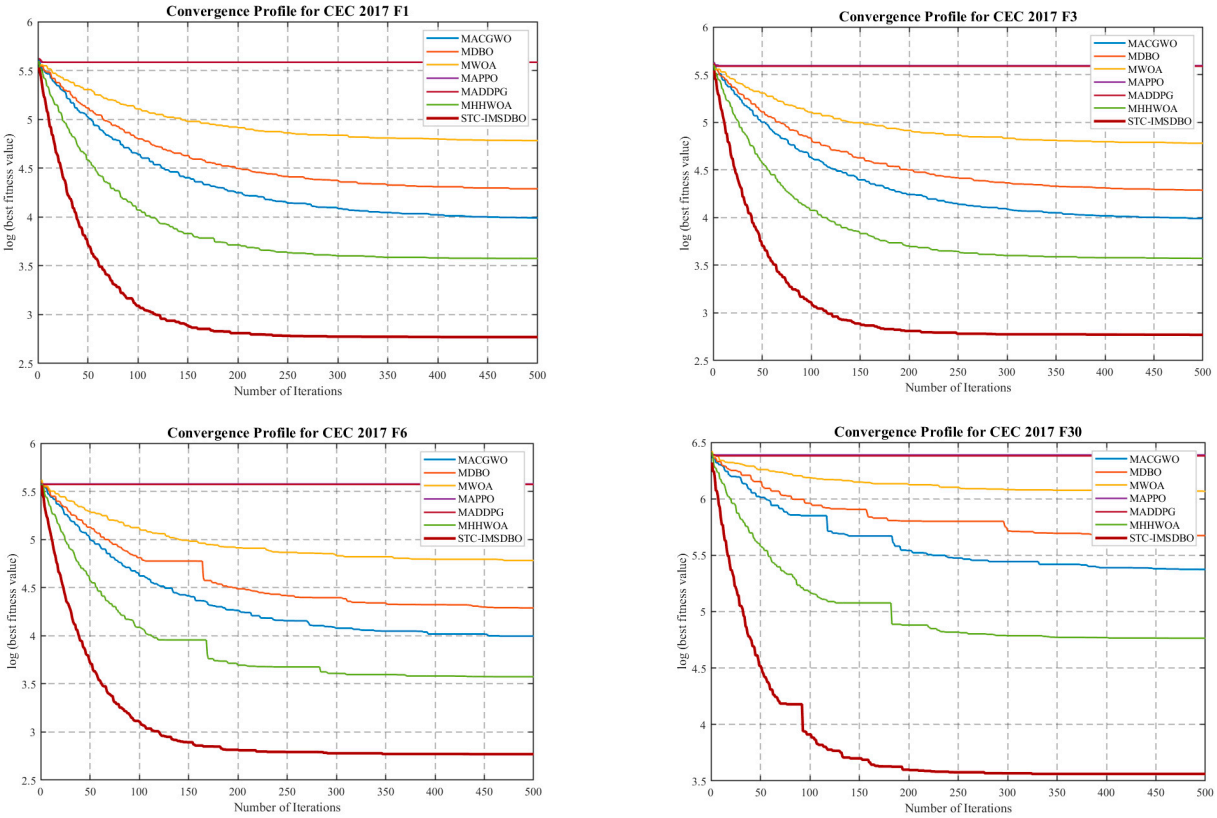


Figure 12. Representative convergence curves on selected CEC2017 functions (F1, F3, F6, and F30).

5.7. Summary

The simulation results give several observations about the behavior of the STC-IMSDBO.

First, the reduced CEC2017 benchmark shows that the proposed search mechanism maintains stable numerical optimization ability on representative test functions. On the retained functions, the STC-IMSDBO obtains lower mean/std values than the comparison methods. For example, on F1 and F30, the results are $9.59 \times 10^1 / 1.34 \times 10^1$ and $3.08 \times 10^3 / 1.66 \times 10^2$, respectively. These results support the general search stability of the algorithm, although they do not directly prove urban flight safety or real-time deployability.

Second, in the four urban multi-UAV planning scenarios, the STC-IMSDBO gives shorter feasible paths and lower energy consumption than the comparison methods. The mean path length reduction ranges from 6.12% to 36.24%, and the mean energy reduction ranges from 8.02% to 33.12%. This suggests that the proposed planner can reduce unnecessary detours while keeping the flight cost lower in the tested scenarios.

Third, the method also shows faster convergence and smoother trajectories. The mean convergence count is 71.75, which is 14.10–39.47% lower than that of the comparison

methods. The mean curvature is reduced by 14.56–40.53%, and the mean fitness value is reduced by 10.29–34.93%. These results indicate that the gain is not limited to path shortening alone.

Fourth, the cooperative indicators are improved in the reported tests. The STC-IMSDBO records a 0% collision rate, 0% airspace violation rate, and a temporal coordination error of 0.62 s. In the dynamic obstacle scenario, the dynamic obstacle avoidance success rate reaches 100%, with a mean path length of 986.37 m and mean energy consumption of 6584.29 kJ. These results should be read as empirical outcomes under the current simulation settings, not as universal guarantees for all urban flight conditions.

Fifth, the ablation experiment shows that the five improvement modules affect different parts of the planning process. AS-ACLHS mainly improves initialization quality, ST-VSSGSI improves convergence and smoothness, MFCSAW improves integrated cost stability, NCGPU improves cooperative safety, and RHRDE is mainly useful for local replanning in dynamic scenes. The full STC-IMSDBO model gives the best overall result in the ablation group, with path length, convergence count, mean curvature, and fitness value reduced by 9.27%, 26.79%, 23.30%, and 17.42%, respectively, compared with original MDBO.

Sixth, the Wilcoxon rank-sum test supports the statistical significance of the main differences in the high-complexity scenario. All reported p-values are below 0.05. This reduces the likelihood that the observed performance gaps are caused only by random variation in repeated runs.

Finally, the Lujiazui digital twin test provides an additional urban-case check. In this scenario, the STC-IMSDBO obtains a total path length of 1288.7 m, an obstacle avoidance success rate of 99.8%, a runtime of 8.45 s, a mean energy consumption of 96.2 kJ, and a minimum safe distance of 20.6 m. These results are encouraging, but they are still simulation-level evidence. Hardware-in-the-loop tests and real-flight validation are still needed before practical deployment claims can be made.

Overall, the experiments show that the STC-IMSDBO improves path efficiency, convergence behavior, smoothness, cooperative safety, and dynamic replanning performance in the reported simulation settings. The current evidence supports the effectiveness of the proposed framework, while larger-swarm runtime tests and field experiments remain necessary follow-up work.

6. Conclusions

This paper studied urban multi-UAV path planning from a practical viewpoint. The main concern was not only how to shorten routes, but also how to keep them feasible, cooperative, and adjustable when the local environment changes.

To this end, a spatio-temporal collaborative improved multi-strategy dung beetle optimization framework, the STC-IMSDBO, was developed. The method combines feasible-airspace initialization, variable-step search, adaptive weighting, local game-based separation handling, and rolling-horizon replanning within one planning loop.

In the reported simulations, the method produced shorter feasible paths, lower energy use, smoother trajectories, and better cooperative safety than the comparison methods across the tested urban scenarios. The ablation results also suggest that the performance gain does not come from one isolated modification. The five modules act on different bottlenecks and work better together than alone.

The current results still have a clear scope. The equilibrium discussion is local and assumption dependent. The convergence discussion is asymptotic rather than a finite-time proof. Larger-swarm runtime evaluation and outdoor validation are also still missing.

These limitations do not invalidate the reported results, but they define the next stage of validation. At the current stage, the STC-IMSDBO can be viewed as a useful planning framework for dense urban missions that require several UAVs to share constrained 3D airspace. The next step should move toward hardware-in-the-loop tests, broader weight sensitivity analysis, and real-flight validation under sensing and communication uncertainty.

Author Contributions: Conceptualization, Y.Y.; methodology, Y.Y.; software, Y.Y.; validation, Y.Y.; formal analysis, Y.Y.; investigation, Y.Y.; resources, Y.Y.; data curation, Y.Y.; writing—original draft preparation, Y.Y.; writing—review and editing, Y.Y.; visualization, Y.Y.; supervision, Y.Y.; project administration, M.L.; funding acquisition, M.L. M.L. is responsible for research direction guidance, overall review and revision of the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Shandong Provincial Natural Science Foundation (ZR2022QF091).

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

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

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned aerial vehicle
STC-IMSDBO	Spatio-Temporal Collaborative Improved Multi-Strategy Dung Beetle Optimization
RRTs	Rapidly exploring Random Trees
GWO	Gray Wolf Optimizer
WOA	Whale Optimization Algorithm
PO	Parrot Optimization
ALOA	Anguilla Lizard Optimization Algorithm
ALA	Artificial Lemming Algorithm
AMCWOA	Adaptive Multi-population Cooperative Whale Optimization
MADDPG	Multi-Agent Deep Deterministic Policy Gradient
MAPPO	Multi-Agent Proximal Policy Optimization
MARL	Multi-Agent Reinforcement Learning
3D	3-Dimensional
UB	Upper bounds
LB	Lower bounds
AS-ACLHS	Airspace-Constrained Adaptive Chaotic Latin Hypercube Sampling
ST-VSSGSI	Spatio-Temporal Coupled Variable Step Size Golden Sine Iteration
MFCSAW	Multi-Factor Coupled Synergistic Adaptive Weight
RHRDE	Rolling Horizon Replanning for Dynamic Environment
NCGPU	Non-cooperative Game Position Update
MWOA	Multi-UAV Whale Optimization
MDBO	Multi-UAV Dung Beetle Optimization
MACGWO	Multi-UAV Adaptive Chaotic Gray Wolf Optimization
MHHWOA	Multi-UAV Hyper-heuristic Whale Optimization

References

1. Javed, S.; Hassan, A.; Ahmad, R.; Ahmed, W.; Ahmed, R.; Saadat, A.; Guizani, M. State-of-the-Art and Future Research Challenges in UAV Swarms. *IEEE Internet Things J.* **2024**, *11*, 19023–19045. [CrossRef]
2. Yahia, H.S.; Mohammed, A.S. Path planning optimization in unmanned aerial vehicles using meta-heuristic algorithms: A systematic review. *Environ. Monit. Assess.* **2023**, *195*, 30. [CrossRef]

3. Zeng, X.; Yu, Y.; Yang, S.; Lv, Y.; Sarker, M.N.I. Urban Resilience for Urban Sustainability: Concepts, Dimensions, and Perspectives. *Sustainability* **2022**, *14*, 2481. [CrossRef]
4. Meng, K.; Wu, B.; Xin, B.; Deng, F.; Chen, C. Multiobjective multi-UAV path planning via evolutionary multitasking optimization with adaptive operator selection and knowledge fusion. *Swarm Evol. Comput.* **2025**, *99*, 102145. [CrossRef]
5. Jiaqi, S.; Li, T.; Hongtao, Z.; Xiaofeng, L.; Tianying, X. Adaptive multi-UAV path planning method based on improved gray wolf algorithm. *Comput. Electr. Eng.* **2022**, *104*, 108377. [CrossRef]
6. Causa, F.; Fasano, G. Multiple UAVs trajectory generation and waypoint assignment in urban environment based on DOP maps. *Aerosp. Sci. Technol.* **2021**, *110*, 106507. [CrossRef]
7. Levin, J.M.; Paranjape, A.; Nahon, M. Agile Fixed-Wing UAV Motion Planning with Knife-Edge Maneuvers. In Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS), Miami, FL, USA, 13–16 June 2017; pp. 114–123.
8. Chen, J.; Ye, F.; Li, Y. Travelling Salesman Problem for UAV Path Planning with Two Parallel Optimization Algorithms. In Proceedings of the Progress in Electromagnetics Research Symposium-Fall (PIERS-FALL), Singapore, 19–22 November 2017; pp. 832–837.
9. Yu, B.; Fan, S.; Cui, W.; Xia, K.; Wang, L. A Multi-UAV cooperative mission planning method based on SA-WOA algorithm for three-dimensional space atmospheric environment detection. *Robotica* **2024**, *42*, 2243–2280. [CrossRef]
10. Muslimov, T.Z.; Munasypov, R.A. Consensus-based cooperative control of parallel fixed-wing UAV formations via adaptive backstepping. *Aerosp. Sci. Technol.* **2021**, *109*, 106416. [CrossRef]
11. He, Z.; Zhao, L. The comparison of four UAV path planning algorithms based on geometry search algorithm. In Proceedings of the 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 26–27 August 2017; pp. 33–36.
12. Chengren, Y.; Guifeng, L.; Wenqun, Z.; Xinglong, P. An efficient RRT cache method in dynamic environments for path planning. *Robot. Auton. Syst.* **2020**, *12*, 103595.
13. Wheeb, A.H.; Nordin, R.; Samah, A.A.; Alsharif, M.H.; Khan, M.A. Topology-Based Routing Protocols and Mobility Models for Flying Ad hoc Networks: A Contemporary Review and Future Research Directions. *Drones* **2022**, *6*, 9. [CrossRef]
14. Hu, D.; Ba, Y.; Cao, W.; Lin, C.; Wang, Z. An Improved A-Star Algorithm for Path Planning of Outdoor Distribution Robots. In *EPCE 2022: 2022 Asia Conference on Electrical, Power and Computer Engineering (EPCE 2022)*; Association for Computing Machinery: New York, NY, USA, 2022; volume 66, pp. 1–6.
15. Zhang, B.; Cai, X.; Li, G.; Li, X.; Peng, M.; Wang, H. Path planning in radioactive environment of nuclear facilities based on modified A-star algorithm and search neighborhood optimization. *Nucl. Eng. Technol.* **2025**, *57*, 103711. [CrossRef]
16. Dijkstra, E.W. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. [CrossRef]
17. Shan, E.; Dai, B.; Song, J.; Sun, Z. A dynamic RRT path planning algorithm based on B-spline. In *Proceedings of the 2009 Second International Symposium on Computational Intelligence and Design*; IEEE Computer Society: Washington, DC, USA, 2009; pp. 25–29.
18. Xue, J.; Shen, B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. *J. Supercomput.* **2023**, *79*, 7305–7336. [CrossRef]
19. Hart, P.E.; Nilsson, N.J.; Raphael, B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
20. LaValle, S.M.; Kuffner, J.J. Rapidly-exploring Random Trees: Progress and prospects. In Proceedings of the 4th International Workshop on the Algorithmic Foundations of Robotics (WAFR), Dartmouth Coll, Hanover, NH, USA, 16–18 March 2000; pp. 293–308.
21. Wang, H.; Lou, S.; Jing, J.; Wang, Y.; Liu, W.; Liu, T. The EBS-A* algorithm: An improved A* algorithm for path planning. *PLoS ONE* **2022**, *17*, e0263841. [CrossRef]
22. Wang, X.; Zhang, H.; Liu, S.; Wang, J.; Wang, Y.; Shangguan, D. Path planning of scenic spots based on improved A* algorithm. *Sci. Rep.* **2022**, *12*, 1320. [CrossRef]
23. Zhang, Z.; Jiang, J.; Wu, J.; Zhu, X. Efficient and optimal penetration path planning for stealth unmanned aerial vehicle using minimal radar cross-section tactics and modified A-Star algorithm. *ISA Trans.* **2023**, *134*, 42–57. [CrossRef] [PubMed]
24. Wu, W.; Kong, C.; Xiao, Z.; Huang, Q. 3D-MIHE-RRT-A*: Multi-indicator heuristic evaluation hybrid path planning algorithm for UAV navigation in complex environments. *Measurement* **2026**, *260*, 119838. [CrossRef]
25. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
26. Mirjalili, S.; Lewis, A. The whale optimization algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
27. Lian, J.; Hui, G.; Ma, L.; Zhu, T.; Wu, X.; Heidari, A.A.; Chen, Y.; Chen, H. Parrot optimizer: Algorithm and applications to medical problems. *Comput. Biol. Med.* **2024**, *172*, 108064. [CrossRef]
28. Rezk, H.; Bouaouda, A.; Hashim, F.A. A novel improved horned lizard optimization algorithm to identify optimal parameters of adaptive fuzzy logic MPPT for performance boosting of PEM fuel cell. *Intell. Syst. Appl.* **2025**, *25*, 200478. [CrossRef]
29. Xiao, Y.; Cui, H.; Khurma, R.A.; Castillo, P.A. Artificial lemming algorithm: A novel bionic meta-heuristic technique for solving real-world engineering optimization problems. *Artif. Intell. Rev.* **2025**, *58*, 84. [CrossRef]

30. Chen, Y.; Zhang, D.; Jing, H.; Huang, Y.; Li, X. An Adaptive Multi-Population Cooperative Whale Optimization Algorithm for global optimization and 3D UAV path planning. *Adv. Eng. Inform.* **2026**, *69*, 104003. [CrossRef]
31. Zhang, S.; Wang, L. Multi-UAV cooperative path planning based on chaotic grey wolf optimiser. *Aeronaut. J.* **2025**, *130*, 741–761. [CrossRef]
32. Xu, T.; Chen, C. DBO-AWOA: An Adaptive Whale Optimization Algorithm for Global Optimization and UAV 3D Path Planning. *Sensors* **2025**, *25*, 2336. [CrossRef]
33. Zhang, C.; Liu, H.; Li, W. An exploration-driven framework for path planning in complex buildings using improved MADDPG. *J. Build. Eng.* **2025**, *107*, 112626. [CrossRef]
34. Liang, C.; Liu, L.; Li, X.; Yan, D.; Li, L. Flocking collision avoidance for multi-UAVs in restricted scenarios: An event-triggered MAPPO strategy based on task-hierarchical curriculum. *Expert Syst. Appl.* **2026**, *306*, 130842. [CrossRef]
35. Yang, W.; Li, B. Connectivity-aware UAV mobility in cellular networks: DRL path planning and predictive handover. *Ad Hoc Netw.* **2025**, *179*, 103999. [CrossRef]
36. Zheng, X.; Liu, R.; Li, S. A Novel Improved Dung Beetle Optimization Algorithm for Collaborative 3D Path Planning of UAVs. *Biomimetics* **2025**, *10*, 420. [CrossRef]
37. Zhang, Z.; Na, Z.; Lin, Y.; Lin, Y.; Nallanathan, A. Multi-UAV path planning based on an improved wolf pack algorithm in complex 3D terrains. *Chin. J. Aeronaut.* **2025**, *12*, 104012. [CrossRef]
38. Tu, B.; Wang, F.; Han, X. 3D path planning for UAV based on A hybrid algorithm of marine predators algorithm with quasi-oppositional learning and differential evolution. *Egypt. Inform. J.* **2024**, *28*, 100556. [CrossRef]
39. Chixin, X.; Zixing, C.; Yong, W. A good nodes set evolution strategy for constrained optimization. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation, Singapore, 25–28 September 2007; pp. 943–950.
40. Li, Y.; Sun, K.; Yao, Q.; Wang, L. A dual-optimization wind speed forecasting model based on deep learning and improved dung beetle optimization algorithm. *Energy* **2024**, *286*, 129604. [CrossRef]
41. Zhang, G.; Fu, S.; Li, K.; Huang, H. Differential evolution with multi-strategies for UAV trajectory planning and point cloud registration. *Appl. Soft Comput.* **2024**, *167*, 112466. [CrossRef]
42. Fan, X.; Wang, H.; Zhuo, Z.; Bei, S.; Li, Y.; Liu, Z. A novel dung beetle optimization algorithm based on Lévy flight and triangle walk. *Future Gener. Comput. Syst.* **2026**, *174*, 108006. [CrossRef]
43. Wu, X.J.; Xu, L.; Zhen, R.; Wu, X.L. Global and Local Moth-flame Optimization Algorithm for UAV Formation Path Planning under Multi-constraints. *Int. J. Control. Autom. Syst.* **2023**, *21*, 1032–1047. [CrossRef]
44. Wang, X.; Zhang, Y.; Zheng, C.; Feng, S.; Yu, H.; Hu, B.; Biomimetics, Z.X.J. An Adaptive Spiral Strategy Dung Beetle Optimization Algorithm: Research and Applications. *Biomimetics* **2024**, *9*, 519. [CrossRef] [PubMed]
45. Fei, H.; Du, Z.; Ma, P.; Liu, R.; Liu, F.; Wang, M.; Liu, X.; Zhou, J. A Multi-Strategy Particle Swarm Optimization Algorithm for Three-Dimensional Path Planning of Amphibious Unmanned Aerial Vehicles. *Eng. Appl. Artif. Intell.* **2026**, *167*, 113606. [CrossRef]
46. Shao, S.; Peng, Y.; He, C.; Du, Y. Efficient path planning for UAV formation via comprehensively improved particle swarm optimization. *ISA Trans.* **2020**, *97*, 415–430. [CrossRef]
47. Ershen, W.A.; Fan, L.I.; Chen, H.O.; Jing, G.U.; Lin, Z.H.; Jian, X.U.; Ning, H.E. MADRL-based UAV Swarm Non-Cooperative Game under Incomplete Information. *Chin. J. Aeronaut.* **2024**, *37*, 293–306. [CrossRef]
48. Glicksberg, I.L. A Further Generalization of the Kakutani Fixed Point Theorem, with Application to Nash Equilibrium Points. *Proc. Am. Math. Soc.* **1952**, *3*, 170–174. [CrossRef]
49. Fan, K. Fixed-point and Minimax Theorems in Locally Convex Topological Linear Spaces*. *Proc. Natl. Acad. Sci. USA* **1952**, *38*, 121–126. [CrossRef]
50. Ma, X.; Zang, S.; Li, X.; Ma, J. Receding Horizon Control with Extended Solution for UAV Path Planning. *Int. J. Aerosp. Eng.* **2022**, *2022*, 3588542. [CrossRef]
51. Dai, X.; Xiao, Z.; Jiang, H.; Lui, J.C. UAV-Assisted Task Offloading in Vehicular Edge Computing Networks. *IEEE Trans. Mob. Comput.* **2024**, *23*, 2520–2534. [CrossRef]
52. Tan, W.; Wu, Q.; Jiang, L.; Tong, T.; Su, Y. Dung Beetle Optimization Algorithm Based on Bounded Reflection Optimization and Multi-Strategy Fusion for Multi-UAV Trajectory Planning. *Comput. Mater. Contin.* **2025**, *85*, 3621–3652. [CrossRef]
53. Song, Y.; Zeng, Q.; Chen, G.; Yuan, G.; Duan, H. Multi-Agent Proximal Policy Optimization based efficient user association and resource allocation in UAV-assisted Heterogeneous Cellular Networks. *Comput. Commun.* **2025**, *238*, 108172. [CrossRef]
54. Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; Mordatch, I.J.A. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
55. Su, Y.; Dai, Y.; Liu, Y. A hybrid hyper-heuristic whale optimization algorithm for reusable launch vehicle reentry trajectory optimization. *Aerosp. Sci. Technol.* **2021**, *119*, 107200. [CrossRef]

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