

Article

Distributed Task Allocation and Trajectory Planning for Heterogeneous UAV Swarms in Multi-Constraint Environments

Bochang Yu ¹, Feng Gao ^{2,*}, Wen Wu ¹, Heng Chai ³, Qun Yao ⁴, Guidao Lin ⁵, Qi Chen ⁵ and Yanbin Liu ²

¹ School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing 210000, China; bochangyu@163.com (B.Y.); wuwen@mail.njust.edu.cn (W.W.)

² College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing 210000, China; liuyb@nuaa.edu.cn

³ College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210000, China; chaiheng8888@163.com

⁴ School of Computer Science, Nanjing University, Nanjing 210000, China; 632023330005@smail.nju.edu.cn

⁵ Yangzhou Marine Electronic Instrument Research Institute, Yangzhou 225000, China; ltfjysr@126.com (G.L.); nju.chenqi@163.com (Q.C.)

* Correspondence: bz2515506@nuaa.edu.cn

Abstract

Owing to the stringent spatio-temporal coupling and kinematic constraints, the task allocation problem for heterogeneous unmanned aerial vehicle (UAV) swarms is generally regarded as an NP-hard problem. To address this, this paper proposes the Sequentially Extended Consensus-Based Bundle Algorithm (SECBBA), a deadlock-free distributed scheduling framework. First, a multi-task allocation model is established by incorporating constraints associated with payload resources, task scheduling, and threat zone. Subsequently, the conventional Consensus-Based Bundle Algorithm (CBBA) is extended through the integration of a deadlock detection and resolution mechanism based on directed graph Depth-First Search (DFS), thereby guaranteeing conflict-free task allocation. Furthermore, a sequential hierarchical strategy is introduced to transform global temporal dependencies into tractable soft time-window constraints. Finally, to ensure physical feasibility, Dubins curves are tightly coupled with the allocation process, enabling nonholonomic path planning for fixed-wing UAVs. Simulation results demonstrate that SECBBA reduces global task costs by 13.3%, 22.7%, and 39.4% compared to the Consensus-Based Bundle Algorithm with Temporal Consistency Constraints (CBBA-TCC), Improved Genetic Algorithm (IGA) and Q-Learning baselines, respectively. It consistently maintains performance advantage of 9.8%, 23.2% and 19.0% under variable weights with high computational efficiency, significantly enhancing swarm timeliness in complex, coupled multi-task scenarios.

Keywords: heterogeneous UAV swarms; task allocation; path planning; resource scheduling; Sequentially Extended Consensus-Based Bundle Algorithm (SECBBA)

Academic Editor: Konstantinos Kontis

Received: 6 March 2026

Revised: 23 June 2026

Accepted: 27 June 2026

Published: 30 June 2026

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

1. Introduction

The rapid development of UAV swarms has fundamentally transformed the paradigm of multi-UAV task execution in complex environments [1,2]. Against the backdrop

of heterogeneous fixed-wing UAV swarms collaboratively executing complex, tightly coupled tasks, UAV swarms demonstrate superior fault tolerance and robustness compared to individual UAVs [3–6]. However, the high degree of complexity inherent in UAV swarm systems poses significant challenges to communication networking, control strategies, and task planning [7,8].

As a fundamental component of high-level task planning, task allocation determines the execution sequence of tasks assigned to UAVs. Existing studies on task allocation can be broadly categorized into centralized and distributed approaches. Among them, centralized task allocation methods typically seek global optimality through mixed-integer linear programming (MILP) formulations and heuristic algorithms. For instance, Xu et al. [9] proposed a Multi-Discrete Wolf Pack Algorithm (MDWPA) to address UAV task allocation in intricate environments. Similarly, Dong [10] integrated Roulette Wheel Selection (RWS) principles and Elitist Strategies (ES) into Ant Colony Optimization (ACO) to tackle the multi-objective, nonlinear, and non-convex challenges associated with UAV swarm task allocation and flight trajectory planning. However, as the swarm scale expands, the computational complexity of such centralized approaches escalates exponentially. Furthermore, in communication-constrained or adversarial scenarios, these architectures are highly susceptible to system-wide paralysis induced by single-point failures, thereby failing to satisfy the stringent requirements for high reliability and low latency inherent in modern missions.

Most distributed task allocation approaches are market-based and do not rely on a central node for computation, thereby offering enhanced system scalability and robustness [11]. In Ref. [12], the CBBA is proposed, which integrates the convergence properties of consensus algorithms with the robustness of auction-based mechanisms. It achieves conflict-free distributed task allocation through an iterative two-stage process consisting of bundle construction and conflict resolution. Ref. [13] addresses these limitations by introducing a task-insertion optimization strategy and a block-based information sharing mechanism, which effectively mitigate the tendency of conventional distributed consensus algorithms to converge to local optima and suffer from communication-induced conflicts. This approach enables collaborative task allocation and path optimization for heterogeneous UAV swarms. Ref. [14] proposes a reinforcement learning-based distributed task allocation algorithm. By employing a decentralized decision-making architecture, the algorithm enables UAVs to autonomously form collaborative teams without the need for a central scheduling authority.

Refs. [15,16] further integrate distributed task allocation with dynamic task reassignment strategies, enabling rapid replanning in response to emergent tasks. However, most existing distributed task allocation studies focus on independent tasks in simplified scenarios and neglect the temporal coupling between tasks. Moreover, research on distributed temporal task allocation has yet to address issues arising from simultaneous task execution by multiple UAVs, such as potential deadlocks. Research on distributed multi-constraint task allocation remains insufficiently explored. In many existing approaches, task allocation and route planning are conducted in an asynchronous manner, which may consequently degrade the reliability of the allocation outcomes.

To address the logical dependencies among heterogeneous tasks in distributed task allocation, this paper proposes a SECBBA with deadlock-correction capability. A hierarchical spatio-temporal coupling model is constructed, transforming the strong sequential dependency of heterogeneous task chains into elastic spatio-temporal constraints through a soft time-window mechanism, thereby significantly improving the solution success rate in complex scenarios. A deadlock-correction mechanism based on directed graphs and Depth-First Search (DFS) is designed to achieve autonomous identification and resolution

of swarm conflicts in a decentralized environment, ensuring the logical consistency of distributed decision-making. Dubins curve kinematic constraints are deeply integrated to ensure that allocation schemes a priori satisfy the non-holonomic dynamic limitations and Size, Weight, and Power (SWaP) constraints of fixed-wing UAVs. Finally, the effectiveness of the proposed method is verified through simulation experiments.

2. Swarm Task Allocation and Decision-Making Modeling Methods

2.1. Problem Description

This paper addresses the pre-mission planning phase prior to fixed-wing UAVs performing time-sensitive remote sensing missions. When UAVs perform search and surveillance tasks, they typically need to maintain a constant cruising altitude. Therefore, the three-dimensional task scheduling is simplified into two-dimensional path planning [9,17–20]. In such tasks, a heterogeneous UAV swarm N_v is deployed to designated areas to survey ground targets $T = \{T_1, T_2, \dots, T_{N_t}\}$ while avoiding threat zones. To circumvent the inherent limitations of single-UAV platforms regarding spatial coverage and resolution, the mission is architected as a closed-loop “Detection-Interaction-Verification” data chain. The swarm consists of three categories of heterogeneous UAVs, each equipped with distinct sensor payloads: Type-1 UAVs are equipped with Synthetic Aperture Radar (SAR). Leveraging the wide-area coverage and signal penetration capabilities of SAR, they execute the primary wide-area search phase, tasked with detecting potential targets under complex meteorological conditions. Type-2 UAVs carry hyperspectral imagers and are responsible for the secondary phase, which involves fine-grained feature extraction of the targets identified by Type-1 UAV. This phase mandates that the UAVs maintain push-broom scanning in close proximity to the target to guarantee a high signal-to-noise ratio. Type-3 UAVs are outfitted with high-resolution Electro-Optical/Infrared (EO/IR) cameras to execute the tertiary visual verification phase. Their objective is to acquire high-definition imagery of the targets post-interaction. Within this framework, the set of tasks executable by UAV i is defined as J_i , and the set of UAVs capable of performing task $j \in J$ is denoted by I_j .

To clarify the focus of this study and simplify the solution process, while ensuring the validity of the model, this paper makes the following assumptions regarding the model:

Assumption 1. *Joint Connectivity of Communication Topology.* The swarm communication network satisfies joint connectivity in the time domain. Specifically, UAV nodes do not need to maintain persistently stable end-to-end direct connection paths; they only need to rely on discrete and intermittently available point-to-point single-hop communication links. Through multi-hop relay forwarding, information can cover all nodes within a finite time, thereby ensuring distributed consensus.

Assumption 2. *Functional Decoupling of Hierarchical Architecture.* The system adopts a decoupled hierarchical architecture, where the task allocation layer is separated from the flight control layer. The lower layer is responsible for trajectory tracking and disturbance rejection, while the upper layer handles task allocation.

Assumption 3. *Target Localization Error.* The target positions supplied to SECBBA are derived from post-processed reconnaissance products generated through prior wide-area surveillance, intelligence fusion, and filtering procedures, rather than from raw sensor measurements. Residual

localization errors may still persist after data processing. Let e_p denote the target localization error and R_{wp} denote the waypoint acceptance radius. The planning model assumes:

$$e_p \leq R_{wp}$$

That is, the localization error is smaller than the mission waypoint radius. Under this condition, the localization error does not alter task priority relationships, resource requirements, or the bidding outcomes of the task allocation process.

2.2. Modeling Task Constraints and Optimization Objectives

In order to facilitate the description of various constraint conditions, we introduce decision variables:

$$x_{ij} = \begin{cases} 1, & \text{UAV } i \text{ executes task } j \\ 0, & \text{UAV } i \text{ doesn't execute task } j \end{cases} \quad (1)$$

2.2.1. Resource Type Constraints

Each category of subtask requires specific spectral or radar resources; therefore, the task resource constraint is:

$$\sum_{i=1}^{N_r} x_{ij} = \text{req}_j, \forall j \in J \quad (2)$$

where req_j denotes the payload resource demand vector of task j . Each dimension sequentially corresponds to the required number of Type-1, Type-2, and Type-3 UAVs. When a single-UAV detection task is performed, $\text{req}_j = [1, 0, 0]$; when two UAVs are required to collaboratively execute the detection task, $\text{req}_j = [2, 0, 0]$.

2.2.2. Temporal Constraints

Within cooperative operations, radar sensors are initially deployed for target localization, followed by spectral imaging, which imposes strict temporal ordering on task execution. The soft time-window constraint dictates that, for any given target T_j , the three subtasks must follow the chronological sequence $T_j^{\text{det}} \rightarrow T_j^{\text{int}} \rightarrow T_j^{\text{ver}}$, meaning that Type-2 UAVs can initiate interaction only after Type-1 UAVs have completed detection and generated precise coordinates, while Type-3 verification must strictly follow Type-2 completion. These sequential constraints can be expressed using Equation (3). In addition, a hard time-window constraint establishes the maximum expected completion time τ_{epc} for each task, requiring that the total mission completion time must satisfy Equation (4).

$$\begin{aligned} t_{\text{end}}(M_C^j) &\leq t_{\text{start}}(M_A^j) \\ t_{\text{end}}(M_A^j) &\leq t_{\text{start}}(M_V^j) \end{aligned} \quad (3)$$

$$t_{\text{max}} \leq \tau_{\text{epc}} \quad (4)$$

In the equation, $t_{\text{start}}(\cdot)$ and $t_{\text{end}}(\cdot)$ represent the start time and end time of the task, respectively, and t_{max} denotes the total completion time of the overall task.

2.2.3. UAV Performance Constraints

To improve the flight performance of drones during Task execution, all drones must meet the following constraints:

$$\text{dis}(i) \leq L_{\text{max}}(i), \forall U_i \in U \quad (5)$$

$$v(i) \leq v_{\max}(i), \forall U_i \in U \quad (6)$$

$$r(i) \geq r_{\min}(i), \forall U_i \in U \quad (7)$$

$$|\gamma(i)| \leq \gamma_{\max}(i) \quad (8)$$

$$|\dot{\gamma}| = \dot{\gamma}_{\max} \quad (9)$$

$$\text{scan}(i) \leq \text{scan}_{\max}(i), \forall U_i \in U \quad (10)$$

$$d_{ij} > d_{\text{safe}} \quad (11)$$

$$\sum_{j=1}^{N_T} x_{ij} \leq L_i, \forall j \in J \quad (12)$$

Equation (5) specifies that the cumulative flight distance $\text{dis}(i)$ of the UAV i must be less than the maximum range $L_{\max}(i)$, U denotes the set of UAVs and $\forall U_i \in U$ represents the i -th UAV belonging to this set. Equations (6) and (7) represent the flight speed constraint and the turning radius constraint for fixed-wing UAVs, respectively; $v_{\max}(i)$ denotes the UAV maximum flight speed, and $r_{\min}(i)$ denotes the UAV minimum turning radius; together, these two factors determine the feasibility of generating a collision-free Dubin curve [21,22]; Equations (8) and (9) represent the maximum climb angle and maximum climb rate constraints for the UAV, respectively; in Equation (10), $\text{scan}_{\max}(i)$ represents the sensor maximum coverage range, which ensures the validity of the data collected by the sensor during task execution; Equation (11) defines the safety separation constraint between UAVs, computed using the Euclidean distance, which guarantees collision-free operations between UAVs during flight.

Finally, during data acquisition task, the volume of acquired raw data is exceptionally voluminous. Constrained by the stringent Size, Weight, and Power (SWaP) limitations inherent to small tactical UAVs, both onboard storage capacity and real-time downlink bandwidth constitute strictly finite resources. Consequently, to preclude buffer overflow or data loss, the cumulative data acquired by an individual UAV must not exceed its hardware capacity threshold. This constraint is formulated in Equation (12), where L_i denotes the maximum number of tasks executable by UAV i . To ensure load balancing across the swarm while concurrently enhancing task completion efficiency, L_i is formulated as follows:

$$L_i = \begin{cases} \text{ceil}(\sum_{j=1}^{N_T} \text{req}_j / N_V), \forall M_j \in M_C \text{ or } M_V \\ \min\left(\text{ceil}(\sum_{j=1}^{N_T} \text{req}_j / N_V), \text{load}_i\right), \forall M_j \in M_A \end{cases} \quad (13)$$

where $\text{ceil}(\cdot)$ denotes the ceiling operator, N_V represents the total number of UAVs designated for the current task type, and load_i indicates the amount of data stored by UAV i . For detection tasks M_C and verification tasks M_V , these two categories are typically collaboratively executed by swarms of a single UAV type, and their task durations are relatively fixed; L_i takes the ceiling value of the ratio between the total demand of such tasks and the number of available UAVs, thereby ensuring load balancing among UAVs of the same type. For interaction tasks M_A , due to their exceptionally voluminous

data and the stringent SWaP constraints inherent to tactical UAVs, L_i takes the minimum value between the aforementioned ceiling integer and the single-UAV data storage capacity load_i , thereby preventing data loss caused by onboard storage overflow.

It is worth noting that the defined maximum task quantity L_i represents a theoretical capacity upper bound; the actual executable quantity is determined by the maximum flight range constraint in Equation (5). During the task bundle construction phase, if the total flight distance of the task path generated according to L_i exceeds $L_{\max}(i)$, the UAV is automatically deemed capacity-saturated and stops.

2.2.4. Environmental Constraints

Within complex environments, UAV swarms are frequently required to navigate through airspaces fraught with high-risk obstacles, including topographical barriers and zones of intense electromagnetic interference. To ensure the survivability of the UAVs and the integrity of the acquired data, the flight trajectories must not intersect with these no-fly zones [23]. Consequently, the threat zone constraint for the flight path is mathematically formulated as follows:

$$x_{ij} = \begin{cases} 0, & \text{if } \text{path}(i, j) \cap W = \emptyset \\ 1, & \text{otherwise} \end{cases} \quad (14)$$

where $\text{path}(i, j)$ denotes the flight trajectory from task node i to task node j , and W represents the designated threat zones.

2.2.5. Objective Function

The ultimate goal of collaborative scheduling is to maximize the overall efficiency of data acquisition. Considering the time-sensitive nature of tasks, information value decays over time. Therefore, with the objective of minimizing the overall task completion time and the average execution time of tasks by each UAV, the global objective function is constructed as follows:

$$\text{score} = \alpha \max_{j=1}^{N_T} t_{ij}^f x_{ij} + \beta \frac{1}{N_V} \sum_{i=1}^{N_V} \sum_{j=1}^{N_T} t_{ij}^f x_{ij} \quad (15)$$

where α is the weight for task completion time, $\sum_{j=1}^{N_T} t_{ij}^f x_{ij}$ is the overall task completion time, β is the weight for task execution efficiency, $\frac{1}{N_V} \sum_{i=1}^{N_V} \sum_{j=1}^{N_T} t_{ij}^f x_{ij}$ is the average task execution time across UAVs, and both α and β are greater than 0, with specific values determined according to requirements. t_{ij}^f represents the total time required for UAV i to complete its assigned tasks, which is the sum of the flight time, waiting time, and task execution duration, calculated as follows:

$$t_{ij}^f = t_{ij}^{\text{dis}} + t_{ij}^{\text{wait}} + t_{ij}^{\text{dur}} \quad (16)$$

where t_{ij}^{dis} denotes the minimum flight time for UAV i from its preceding task node to task node j , t_{ij}^{wait} indicates the waiting time, and t_{ij}^{dur} represents the execution time required for task j .

2.3. Task Allocation Model and Optimization Algorithm

While the conventional CBBA [12] is proficient in independent task allocation, it exhibits limitations when applied to complex, coupled multi-task scenarios. To address this deficiency, the SECBBA is proposed. As illustrated in Figure 1, the SECBBA framework comprises four fundamental phases: bundle construction, conflict resolution, deadlock recovery, and hierarchical task sequencing.

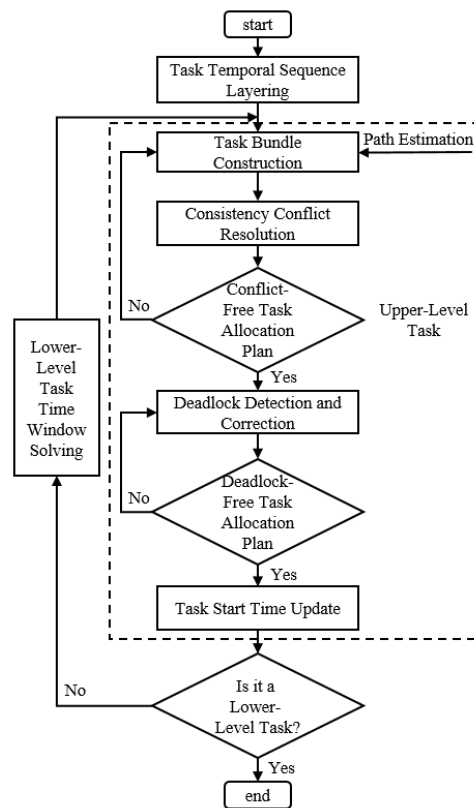


Figure 1. SECBBA Process.

2.3.1. Task Bundle Construction

According to Reference [24], during the bundle construction phase, each UAV concurrently and locally employs a Diminishing Marginal Gain (DMG) scoring mechanism to select tasks to add to its bundle b_i , and generates a task path list P_i , a winner matrix Z_i , and a winning bid value matrix Y_i . Specifically, tasks are selected based on the principle of maximizing their marginal reward gain. This process iterates continuously until the bundle capacity is reached or no eligible tasks remain available. In each selection round, the UAV first calculates the marginal gain for all tasks satisfying the constraints to serve as the bid value. The marginal gain c_{ij} for inserting task j into bundle b_i is calculated as follows:

$$c_{ij} = \max_{n \leq |P_i|} S_i^{P_i \oplus_n \{j\}} - S_i^{P_i}, \forall j \in J \setminus b_i \quad (17)$$

where $P_i \oplus_n \{j\}$ denotes the new path list generated by inserting task j into the n -th position of P_i , with the original n -th element and all subsequent elements shifted back-

ward while strictly maintaining their relative sequence. $J \setminus b_i$ represents the set of available tasks after excluding those already present in b_i . $S_i^{p_i}$ indicates the total reward obtained by UAV^{*i*} executing tasks along path p_i , which is calculated as:

$$S_i^{p_i} = \sum_{j \in p_i} (\text{rwd}_{ij} - \text{cst}_{ij}) \quad (18)$$

where rwd_{ij} is the reward for UAV^{*i*} executing task j . To guarantee the execution of all tasks, this paper formulates the reward by integrating a fixed reward with a time-discounted reward, expressed as:

$$\text{rwd}_{ij} = (R_{j0} + e^{-\lambda_j(t_{ij} - \tau_j^{\text{start}})}) R_j u(t_{ij}) \quad (19)$$

where R_{j0} represents the fixed reward for task execution, R_j is the base value of the time-discounted reward, and $0 < \lambda_j < 1$ denotes the time discount coefficient for task j , which dictates the decay rate of the base reward over time. $u(t_{ij})$ is a decision variable dependent on the execution time:

$$u(t_{ij}) = \begin{cases} 1, & \text{if } \tau_j^{\text{start}} \leq t_{ij} \leq \tau_j^{\text{end}} - \tau_j^\Delta \\ 0, & \text{otherwise} \end{cases} \quad (20)$$

where $[\tau_j^{\text{start}}, \tau_j^{\text{end}}]$ signifies the time window of task j , and τ_j^Δ denotes the time consumption of task j . Consistent with the established task models, the durations for interaction and verification tasks are set as constants, whereas the minimum time consumption for detection tasks is derived through calculation:

$$\tau_j^\Delta = \text{area}_j / (\text{req}_j \times \text{scan}_{\max} \times v_{\max}) \quad (21)$$

where area_j represents the area of the target region associated with task j . In Equation (18), cst_{ij} indicates the execution cost of the task, which is primarily evaluated based on the flight range cost:

$$\text{cst}_{ij} = \gamma_i d_{ij} \quad (22)$$

where γ_i denotes the flight range cost coefficient for UAV^{*i*}, and d_{ij} signifies the flight range cost incurred by UAV^{*i*} to execute task j . This distance is estimated utilizing a Dubins curve path equipped with obstacle-avoidance mechanisms, the specific methodology for which will be elaborated subsequently.

Upon calculating the bid values, the UAV determines whether to bid based on the task requirements and a comparative analysis with the winning bid values, subsequently adding the task yielding the highest marginal gain into its bundle. The bundle construction logic is detailed in Algorithm 1.

Algorithm 1 Task Package Construction

Input: $b_i(t-1), p_i(t-1), \mathbf{Z}^i(t-1), \mathbf{Y}^i(t-1)$

Output: $b_i(t), p_i(t), \mathbf{Z}^i(t), \mathbf{Y}^i(t)$

- 1: while $|b_i(t-1)| \leq L_i$ do
 - 2: $c_{ij} = \max_{n \leq |p_i|} S_i^{p_i \oplus \{j\}} - S_i^{p_i}, \forall j \in J \setminus b_i$;
 - 3: for $\forall j \in J$
 - 4: $h_{ij} = 0$;
-

```

5:   if  $\text{req}_j = 1$  then;
6:        $h_{ij} = \mathbf{I}(c_{ij} > \sum_{k=1}^{N_V} y_{kj}^i)$ ;
7:   else if  $\text{req}_j > 1$  then;
8:       if  $\text{req}_j > \sum_{k=1}^N z_{kj}^i$  then;
9:            $h_{ij} = 1$ ;
10:      else if  $c_{ij} > \min(y_{kj}^i)$  then;
11:           $h_{ij} = 1$ ;
12:      end if
13:  end if
14: end for
15:  $J_i = \arg \max_j (c_{ij} \times h_{ij})$ ;
16:  $n_{i,J_i} = \arg \max_n S_i^{p_i \oplus n \{J_i\}}$ ;
17:  $b_i(t) = b_i(t-1) \oplus_{\text{end}} \{J_i\}$ ;
18:  $p_i(t) = p_i(t-1) \oplus_{n_{i,J_i}} \{J_i\}$ ;
19:  $z_{i,J_i}^i(t) = 1, y_{i,J_i}^i(t) = c_{i,J_i}$ ;
20: end while

```

In Algorithm 1, $h_{ij} = 1$ implies that UAV^{*i*} is eligible to bid on task *j*. The value of the indicator function $\mathbf{I}(\bullet)$ is conditional, equating to 1 when the designated condition is met and 0 otherwise. The operator $\oplus$ denotes inserting a task into the current path sequence. Initially, the bid values of UAV^{*i*} for all eligible tasks are calculated according to Equation (17).

When $\text{req}_j = 1$, indicating that task *j* is a single-UAV execution type, there is at most one non-zero value in the *j*-th column of $\mathbf{Y}_i$. In this case, UAV^{*i*} only needs to compare its own bid value with the winning bid value of task *j* to determine whether to bid. If the bid values are identical, UAV^{*i*} proceeds with the bid provided it possesses a smaller identification index.

When $\text{req}_j > 1$, task *j* is a multi-UAV cooperative execution type, necessitating an evaluation of whether the number of bidding UAVs satisfies the requirements of task *j*.

If $\text{req}_j > \sum_{k=1}^{N_V} z_{kj}^i$, it indicates that UAV^{*i*} perceives task *j* as under-allocated; thus, it directly proceeds to bid. If $\text{req}_j = \sum_{k=1}^{N_V} z_{kj}^i$, indicating that UAV^{*i*} perceives task *j* as fully allocated, UAV^{*i*} must compare its bid value with the minimum winning bid value of task *j* to decide on bidding. Similarly, in the event of a tie, the UAV with the smaller index is prioritized.

Upon the completion of a bidding round, UAV^{*i*} selects the task with the maximum bid value from all bid-upon tasks to insert into its bundle b_i , and concurrently updates its path list and winning information. The UAV continuously executes this bundle construction process until its bundle reaches maximum capacity or no tasks satisfying the

constraints remain available. Subsequently, it transitions into the consensus negotiation phase for conflict resolution.

2.3.2. Consistency Conflict Resolution

Since the task bundle construction phase is completed independently by each UAV based on partial information, a globally conflict-free task allocation scheme cannot be directly obtained. Therefore, UAVs must exchange data with neighboring UAVs and update winning information and timestamps s_i according to the consensus rules of the CBBA in Table 1, thereby achieving an unambiguous task allocation scheme and realizing global consistency. The so-called global consistency refers to the state in which, after a finite number of information exchange rounds, the locally stored task allocation scheme of each UAV is identical to the global task allocation scheme.

Table 1. Consensus Rules of the CBBA.

Sender k 's z_{kj}	Receiver l 's z_{lj}	Receiver Behavior (Default: Depart)
k	i	If $y_{bj} > y_{ij}$, Update
	k	Update
	$m \notin \{i, k\}$	If $s_{km} > s_{im}$ or $y_{kj} > y_{ij}$, Update
	$\emptyset$	Update
i	i	Depart
	k	Reset
	$m \notin \{i, k\}$	If $s_{km} > s_{im}$, Reset
	$\emptyset$	Depart
$m \notin \{i, k\}$	i	If $s_{km} > s_{im}$ and $y_{bi} > y_{ij}$, Update
	k	If $s_{km} > s_{im}$, Update
		Others, Reset
	m	If $s_{km} > s_{im}$, Update
		If $s_{km} > s_{im}$ and $s_{kn} > s_{in}$, Update
	$n \notin \{i, k, m\}$	If $s_{km} > s_{im}$ and $y_{bj} > y_{ij}$, Reset
		If $s_{kn} > s_{in}$ and $s_{km} > s_{im}$, Reset
$\emptyset$	$\emptyset$	If $s_{km} > s_{im}$, Update
	i	Depart
	k	Update
	$m \notin \{i, k\}$	If $s_{km} > s_{im}$, Update
	$\emptyset$	Depart

The timestamp list is denoted as $s_i = \{s_{i1}, s_{i2}, \dots, s_{iN_r}\}$, which stores the latest time UAV i acquired information from neighboring UAVs. In the original CBBA, upon receiving the winning information and timestamp from a neighboring UAV, UAV i can execute one of the following three actions regarding task j :

$$\begin{cases} \text{Update: } z_{ij} = z_{kj}, y_{ij} = y_{kj} \\ \text{Reset: } z_{ij} = \emptyset, y_{ij} = 0 \\ \text{Depart: } z_{ij} = z_{ij}, y_{ij} = y_{ij} \end{cases} \quad (23)$$

where “Update” means the receiver adopts the sender’s winner information and bid information to overwrite its corresponding records; “Reset” means the receiver clears the local allocation of relevant tasks and releases subsequent associated tasks when conflicts

or information inconsistencies are detected; “Depart” means the receiver maintains its current state unchanged without adjusting existing allocations.

The specific consensus rules [12] are detailed in Table 1. The default action for the receiving UAV is “Depart”. If an “Update” or “Reset” action is triggered, the task bundle will be correspondingly updated upon the conclusion of the consensus phase.

This paper extends the consensus rules of the original CBBA to facilitate the conflict-free allocation of multi-UAV cooperative tasks. Assuming all UAVs employ a synchronous communication mechanism during the consensus phase [25], the consensus-based conflict resolution algorithm is formulated as Algorithm 2.

Algorithm 2 Consistency Conflict Resolution

Input: $\mathbf{Z}_i(t-1), \mathbf{Y}_i(t-1), s_i(t-1)$

Output: $\mathbf{Z}_i(t), \mathbf{Y}_i(t), s_i(t)$

```

1: UAV  $i$  receives  $\mathbf{Z}^k(t-1), \mathbf{Y}^k(t-1)$  and  $s_k(t-1)$  sent by UAV  $k$ ;
2: for  $\forall j \in J$ 
3:   if  $\text{req}_j = 1$  then
4:      $\mathbf{Z}_i \rightarrow z_i, \mathbf{Y}_i \rightarrow y_i$ ;
5:     Resolve conflicts according to the consensus rules in Table 1;
6:      $z_i \rightarrow \mathbf{Z}_i, y_i \rightarrow \mathbf{Y}_i$ ;
7:   else if  $\text{req}_j > \sum_{k=1}^{N_j} z_{kj}^i$  then;
8:     for  $\forall m \in I$  where  $z_{mj}^i = 1$ ;
9:       if  $k = m$  and  $s_{km} > s_{im}$  then;
10:         $z_{mj}^i = z_{mj}^k, y_{mj}^i = y_{mj}^k$ ;
11:      end if
12:    end for
13:    for  $\forall m \in I$  where  $z_{mj}^i = 1$ 
14:      if  $m \neq i$  and  $z_{mj}^i = 0$   $s_{km} > s_{im}$  then
15:        if  $\text{req}_j > \sum_{m=1}^N z_{mj}^i$  then
16:           $z_{mj}^i = z_{mj}^k, y_{mj}^i = y_{mj}^k$ 
17:        else if  $\min(y_{nj}^i \forall n) < y_{mj}^k$  then
18:           $z_{nj}^i = 0, z_{mj}^i = z_{mj}^k$ ;
19:           $y_{nj}^i = 0, y_{mj}^i = y_{mj}^k$ ;
20:        end if
21:      end if
22:    end for
23:  end if
24:   $s_{ik} = t$ ;
25: end for

```

In Algorithm 2, UAV i initially receives the winning information and timestamp from its neighbor UAV k , and subsequently updates its own winning records. For a given task j , if $\text{req}_j = 1$, it indicates a single-UAV execution type. Because this framework utilizes matrices to store winning information, it is requisite to revert the matrices $\mathbf{Z}_i$ and $\mathbf{Y}_i$ back to the list formats z_i and y_i . Following this, the original CBBA consensus rules

are applied for conflict resolution, and the lists are ultimately converted back into matrices for storage.

Conversely, if $\text{req}_j > 1$, task j is designated as a multi-UAV cooperative type, and the conflict resolution process is bifurcated into two stages. In the first stage, the receiver i compares its stored winning information with that of the sender k to verify that its local data is current. In the second stage, the receiver extracts new information from the sender and merges it with its own. If the sender k indicates that UAV $m (m \neq i)$ is assigned to execute task j , and receiver i lacks this specific record, receiver i must evaluate whether the allocation for task j is fully saturated. If it is under-allocated, an update is directly executed. If task j is fully allocated, receiver i must compare the minimum non-zero bid value in the j -th column of y_{mj}^k . If UAV m 's bid value is strictly greater, the UAV holding the minimum bid value is replaced by UAV m , and the winning information is updated accordingly. Similarly, in the event of identical bid values, the UAV possessing the smaller identification index is prioritized.

Upon the termination of the consensus negotiation phase, each UAV verifies whether its winning information conflicts with its bundle information. Should a conflict be detected, the UAV releases the conflicting task along with all subsequent tasks from its bundle, thereby initiating a new iteration of bundle construction and conflict resolution.

2.3.3. Deadlock Detection and Correction

Through the iterative execution of the bundle construction and conflict resolution phases, a conflict-free task allocation scheme satisfying resource requirements is derived. However, the concurrent execution of localized detection tasks by multiple UAVs may precipitate system deadlocks, resulting in indefinite waiting times and task execution failures. Consequently, it is imperative to subject the preliminary allocation results to rigorous deadlock detection and recovery protocols.

This paper describes the deadlock problem using a directed graph $G = \langle V, E \rangle$, where the nodes V are restricted to task nodes with $\text{req}_j > 1$, and E is the set of edges in the directed graph; an edge (i, j) in E indicates that there is a task executed by a drone traveling from node i to node j . According to reference [26], cycles in G represent deadlocks, and a deadlock-free task sequence is equivalent to the directed graph G being a Directed Acyclic Graph (DAG). An existing deadlock in the directed graph is shown in Figure 2a, where nodes 2 and 3 form a deadlock cycle; the schematic diagram after correction without deadlock is shown in Figure 2b. Cycles in the directed graph can be detected using the Depth-First Search (DFS) algorithm; if a cycle exists, the nearest-neighbor principle is adopted to correct the task sequence to eliminate the deadlock. The deadlock detection and correction algorithm is presented as Algorithm 3.



Figure 2. Diagram of deadlock in a directed graph, (a) deadlock exists. (b) deadlock-free after the fix.

Algorithm 3 Deadlock Detection and Resolution

Input: Conflict-free task allocation result

Output: Deadlock-free task allocation result

```

1: Construct the task sequence directed graph  $G = \langle V, E \rangle$ ;
2: Use the DFS algorithm to detect if there is a cycle in  $G$ , and store the cycle information
   in 'loop';
3: while loop  $\neq \emptyset$  do
4:    $(i, j) = \arg \min_{i,j} (E_{ij}), \forall i, j \in \text{loop and } i > j$ ;
5:   for  $\forall k \in I$ 
6:     if  $(i, j) \in p_k$  then
7:       Swap the positions of  $i$  and  $j$  in  $p_k$ ;
8:     end if
9:   end for
10:  Update the directed graph  $G$ ;
11: Use the DFS algorithm to detect cycles and update the 'loop';
12: end while

```

In Algorithm 3, the directed graph G of multi-UAV execution task nodes is first constructed, and the DFS algorithm is used to detect if there are any cycles in G . If a cycle exists, the nearest-neighbor principle is applied to find the two nodes i and j with the minimum exchange cost. These nodes are sorted by their identifiers, and the UAV k that executes task i before task j is identified. The positions of nodes i and j are swapped in UAV k 's path list. The directed graph G is then updated, and cycle detection is performed again. If a cycle is still present, further corrections are made until the entire task allocation scheme is free from deadlock. Since the deadlock resolution process only involves adjusting the task execution order, the corrected scheme still satisfies the payload resource constraints.

2.3.4. Sequential Hierarchical Solution of Related Tasks

By transforming the inter-task temporal constraints into soft time-window constraints, this paper employs a hierarchical sequencing strategy for problem resolution. Tasks sharing identical priorities are processed under a parallel execution paradigm. Conversely, tasks with disparate priorities are sequentially resolved in strict accordance with their temporal constraints, utilizing the completion time of the preceding task as the initiation time for the subsequent task. Prior to formulating the soft time-window constraints, it is imperative to update the initiation times of the preceding tasks to guarantee the synchronized activation of multi-UAV cooperative tasks. The procedure for updating the initiation times of upper-tier tasks is detailed in Algorithm 4.

Algorithm 4 Deadlock Detection and Resolution

Input: Unupdated task start time times_old

Output: Updated task start time times_new

```

1: while flag=1 do
2:  Update the start time of each task in  $p_i$  and store it in ' $t\_latest_j$ ';
3:  Calculate the difference between the original start time of tasks in  $p_i$  and  $t\_latest_j$ ,
   and store it in  $t\_delta_{i,m}$ ;
4:  if  $\sum t\_delta_{im} > 0$  then
5:    flag=1;
6:     $t\_delta_{im} = \max(t\_delta_{i,m-1}, t\_delta_{im})$ ;
7:     $times\_new_{im} = times\_old_{im} + t\_delta_{im}$ ;
8:  else if  $\sum t\_delta_{im} = 0$  then

```

```

9:     flag=0;
10:  end if
11: end while

```

In Algorithm 4, the estimated arrival time of the UAV that reaches the latest for task j is first used as the reference to update the latest start time t_{latest_j} for the current task. The difference between the original start time and the latest start time for tasks in P_i is computed, and when all values in $t_{delta_{i,m}}$ are zero, it indicates that the algorithm has converged. After updating the start times of all upper-level detection tasks, the start times of the lower-level interaction tasks in each region are replaced with the end times of the upper-level tasks. A soft time window for the associated interaction tasks within the same region is then constructed to satisfy the temporal constraints between the detection and interaction tasks. The interaction tasks are then solved using the SECBBA, followed by deadlock detection and correction. Finally, the start times of the interaction tasks are updated according to the simultaneous interaction constraints, and a soft time window for the verification tasks is constructed. The verification tasks are then allocated without conflicts. Since the verification tasks are single-UAV tasks, deadlock detection and start time updates are not necessary.

2.3.5. Coupled Path Planning

The conventional CBBA utilizes the Euclidean distance between nodes as the metric for evaluating flight range cost. However, within operational environments containing threat zones, this metric exhibits significant deviations in estimating the actual flight trajectory, thereby degrading the reliability of the task allocation scheme. To address this deficiency, a Dubins curve-based path planning method [27] is incorporated to generate obstacle-avoidance flyable trajectories, aiming to enhance the reliability of the flight range cost estimation.

The Dubins curve is a classical approach for solving the shortest path problem under kinematic curvature constraints. Its path configurations containing straight segments can be categorized into four types, where 'L' denotes counterclockwise circular motion, 'R' denotes clockwise circular motion, and 'S' denotes straight-line motion. When utilizing Dubins curves for obstacle avoidance, it is initially necessary to determine whether the straight-line path from the origin to the destination intersects a threat zone. The criterion for identifying such intersection is whether the perpendicular distance from the center of the threat zone to the straight-line path is strictly less than the radius of the threat zone. For scenarios where the trajectory intersects a threat zone, a detour strategy must be employed, as illustrated in Figure 3. This involves identifying a proximal detour waypoint and subsequently constructing Dubins paths between the origin and the detour waypoint, and between the detour waypoint and the destination, respectively, to achieve effective obstacle avoidance.

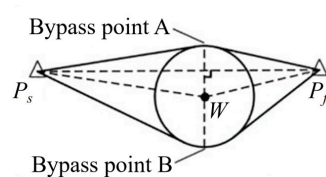


Figure 3. Path Affected by Threat Zone Illustration.

The original Dubins path begins and ends with the same circular radius; however, it is necessary to address scenarios with differing radii during obstacle avoidance. This paper leverages the Euclidean principle and employs the construction of a CLC (Curvature-Limited Circle) line [28] to determine the path length, as illustrated in Figure 4. It should be noted that the Dubins curves employed in this paper are not merely used for flight distance estimation, but rather serve directly as a flyable trajectory generation model between task nodes. Fixed-wing UAVs are nonholonomic constraint systems, whose heading changes must be achieved through continuous curvature trajectories and cannot execute the spot turns or instantaneous large-angle maneuvers characteristic of multirotor UAVs. Consequently, all arc segments in the Dubins paths satisfy Equation (7), thereby ensuring that the generated trajectories consistently satisfy the kinematic reachability requirements of fixed-wing UAVs.

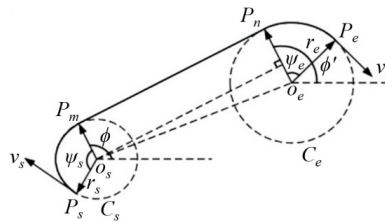


Figure 4. Dubins Path Schematic.

Given the starting attitude point $P_s(x_s, y_s, \phi_s)$ and the ending attitude point $P_e(x_e, y_e, \phi_e)$, the centers of the starting circle C_s and the ending circle C_e , denoted as $o_s(x_{cs}, y_{cs})$ and $o_e(x_{ce}, y_{ce})$, can be determined using the following equations:

$$\begin{aligned} x_{cs} &= x_s - r_s \cos(\phi_s \pm \pi/2) \\ y_{cs} &= y_s - r_s \sin(\phi_s \pm \pi/2) \\ x_{ce} &= x_e - r_e \cos(\phi_e \pm \pi/2) \\ y_{ce} &= y_e - r_e \sin(\phi_e \pm \pi/2) \end{aligned} \quad (24)$$

In the equation, the sign of $\pm$ depends on the direction of motion: clockwise is $+$, and counterclockwise is $-$. According to the rules in Table 1 of this paper, the parameters ϕ for the starting circle and ϕ' for the ending circle are determined. In Table 2, $\text{mod}(\cdot)$ represents the modulo operator, and $\text{sign}(\cdot)$ represents the sign function. From this, the angles of the arc segments, ψ_s and ψ_e , are calculated as:

$$\begin{aligned} \psi_s &= \text{mod}(2\pi + \frac{\pi}{2} \pm (\phi_s - \phi), 2\pi) \\ \psi_e &= \text{mod}(2\pi + \frac{3\pi}{2} \pm (\phi_e - \phi'), 2\pi) \end{aligned} \quad (25)$$

In the equation, the sign of $\pm$ depends on the direction of motion: clockwise is $+$, and counterclockwise is $-$. The positions of the tangency points P_m on circle C_s and P_n on circle C_e can be calculated using the following equations:

$$\begin{aligned} x_{P_m} &= x_{cs} + r_s \cos \phi \\ y_{P_m} &= y_{cs} + r_s \sin \phi \\ x_{P_n} &= x_{ce} + r_e \cos \phi' \\ y_{P_n} &= y_{ce} + r_e \sin \phi' \end{aligned} \quad (26)$$

The total length of the Dubins path is:

$$L_{\text{Dubins}} = r_s \psi_s + \sqrt{(x_{P_m} - x_{P_n})^2 + (y_{P_m} - y_{P_n})^2} + r_e \psi_e \quad (27)$$

Table 2. The Solution Rules of ϕ and ϕ' .

Curve	Starting Circle ϕ	Ending Circle ϕ'
RSR	$\text{mod}(\arcsin \frac{r_e - r_s}{\sqrt{(x_{cs} - x_{ce})^2 + (y_{cs} - y_{ce})^2}} + \arctan(\frac{y_{ce} - y_{cs}}{x_{ce} - y_{cs}}) + \text{sign}(x_{ce} - x_{cs}) \frac{\pi}{2} + 2\pi, 2\pi)$	ϕ
LSL	$\text{mod}(-\arcsin \frac{r_e - r_s}{\sqrt{(x_{cs} - x_{ce})^2 + (y_{cs} - y_{ce})^2}} + \arctan(\frac{y_{ce} - y_{cs}}{x_{ce} - y_{cs}}) + \text{sign}(x_{ce} - x_{cs}) \frac{\pi}{2} + 2\pi, 2\pi)$	ϕ
RSL	$\text{mod}(-\arcsin \frac{r_e + r_s}{\sqrt{(x_{cs} - x_{ce})^2 + (y_{cs} - y_{ce})^2}} + \arctan(\frac{y_{ce} - y_{cs}}{x_{ce} - y_{cs}}) + \text{sign}(x_{ce} - x_{cs}) \frac{\pi}{2} + 2\pi, 2\pi)$	$\text{mod}(\phi + \pi, 2\pi)$
LSR	$\text{mod}(\arcsin \frac{r_e + r_s}{\sqrt{(x_{cs} - x_{ce})^2 + (y_{cs} - y_{ce})^2}} + \arctan(\frac{y_{ce} - y_{cs}}{x_{ce} - y_{cs}}) + \text{sign}(x_{ce} - x_{cs}) \frac{\pi}{2} + 2\pi, 2\pi)$	$\text{mod}(\phi + \pi, 2\pi)$

The aforementioned Dubins curves primarily address the path planning problem during the cruise phase. However, fixed-wing UAVs must also satisfy kinematic feasibility requirements during the climb and descent phases. To this end, this paper adopts a decoupled three-dimensional trajectory generation strategy, in which the horizontal path during the cruise phase follows a Dubins curve that avoids threat zones, while the vertical profiles during the climb and landing phases are constrained by the maximum climb angle $\gamma_{\max}(i)$. Since the climb and landing phases follow similar procedures, this paper focuses solely on the analysis of the climb phase.

The UAV climbs from an initial altitude z_s to the mission altitude z_m . Under the normal overload factor n_z constraint, the climb trajectory is decomposed into three consecutive phases: (i) an initial transition phase, during which the climb angle gradually increases from its initial value to the maximum climb angle $\gamma_{\max}(i)$; (ii) a steady climb phase, during which the maximum climb angle $\gamma_{\max}(i)$ is maintained and the UAV rapidly approaches the target altitude at a constant climb rate; and (iii) an altitude correction phase, during which the climb angle smoothly decays to zero, allowing the UAV to converge stably to the designated mission altitude z_m . The altitude evolution equation is given by:

$$z(t) = \begin{cases} z_s + \int_0^t v \cdot \sin \gamma(t) dt, & 0 \leq t \leq t_1 \\ z_1 + \int_{t_1}^t v \cdot \tan \gamma_{\max}(t) dt, & t_1 < t \leq t_2 \\ z_2 + \int_{t_2}^t v \cdot \tan \gamma(t) dt, & t_2 < t \leq t_3 \\ z_m, & t > t_3 \end{cases} \quad (28)$$

where t_1, t_2, t_3 denotes the switching instants between the three phases, and $\gamma(t)$ is the climb angle at the current time. The corresponding phase-switching conditions are determined by the following system of equations:

$$\begin{cases} t_1 = \frac{V}{g} \int_{\gamma_s}^{\gamma_{\max}} \frac{\cos \gamma}{\sin \gamma (n_z - \cos \gamma)} d\gamma, & z_1 = z_s + \int_0^{t_1} V \sin \gamma(t) dt \\ t_2 = t_1 + \frac{(z_2 - z_1) \tan \gamma_{\max}}{V}, & z_2 = z_m - \int_{t_2}^{t_3} V \tan \gamma(t) dt \\ t_3 = t_2 + \frac{V}{g} \int_{\gamma_{\max}}^0 \frac{\cos \gamma}{\sin \gamma (n_z - \cos \gamma)} d\gamma, & z_m = z_3 \end{cases} \quad (29)$$

Furthermore, since all UAVs considered in this study are fixed-wing platforms operating at the same mission altitude, the fixed task overhead incurred by climb and descent

is identical across different UAVs. As this overhead does not alter the relative ranking among candidate task allocation schemes, it is not explicitly treated as an optimization variable in the CBBA bidding process.

2.3.6. Dynamic Obstacle-Avoidance Strategy

During mission execution by fixed-wing UAVs, conflicts typically occur during the climb phase, cruise phase, and landing phase. Since fixed-wing UAVs are subject to minimum turning radius and maximum climb angle constraints and cannot employ hovering-and-waiting for collision avoidance, this paper adopts a conflict avoidance strategy of “altitude adjustment priority, supplemented by loiter-and-wait”.

When the predicted minimum separation between two UAVs falls below a predefined safety threshold, the system identifies a potential conflict. Subsequently, the system determines the collision avoidance strategy based on the current remaining fuel of each UAV. If a UAV's remaining fuel exceeds 50% of its total onboard capacity, it is assigned the loiter-and-wait strategy, allowing it to temporarily defer its landing time and vacate the conflicting airspace. Otherwise, the UAV executes an altitude adjustment maneuver. Under the altitude adjustment strategy, the yielding UAV gradually climbs to a temporary safe altitude layer through a transition segment, a steady climb phase, and an altitude correction segment. Once the inter-UAV distance again exceeds the minimum safety separation, the UAV executes a symmetric descent procedure and returns to its flight trajectory.

Since fixed-wing UAVs are subject to minimum turning radius constraints, the waiting trajectory cannot be directly entered through abrupt heading changes. To ensure trajectory continuity, this paper employs the Low-altitude Type trajectory generation method from three-dimensional Dubins paths to construct the waiting loiter trajectory [29].

Let the altitudes of the waiting trajectory entry point and exit point be denoted as z_s, z_e , respectively, and let the corresponding two-dimensional Dubins trajectory length be $L_{2D}(r_{\min})$; then the optimal flight path angle can be expressed as:

$$\gamma^* = \tan^{-1} \left(\frac{|z_e - z_s|}{L_{2D}(r_{\min})} \right) \quad (30)$$

The corresponding three-dimensional Dubins trajectory length is:

$$L_{3D}(r_{\min}, \gamma^*) = \frac{L_{2D}(r_{\min})}{\cos \gamma^*} \quad (31)$$

3. Simulation Experiments and Analysis

This section systematically evaluates the efficacy of the proposed SECBBA framework within a typical collaborative multi-task scenario. To rigorously validate its performance superiority, three representative algorithms are selected as comparative baselines: the distributed baseline algorithm (CBBA-TCC), the centralized improved genetic algorithm (IGA), and the distributed reinforcement learning algorithm (Q-Learning). The simulations predicate that the UAV swarm operates under a Fully Connected Topology to ensure optimal information exchange. All experimental simulations are implemented within the MATLAB R2019a environment, executed on a computational platform equipped with an Intel Core i5-6300HQ @ 2.30GHz processor and 8 GB of RAM.

3.1. Typical Scenario Task Allocation Results

To systematically validate the effectiveness of the SECBBA in resolving spatio-temporal coupled scheduling problems for heterogeneous UAVs, a representative data acquisition scenario is constructed. The simulation environment is defined within a $50\text{km} \times 50\text{km}$ operational airspace, deploying a swarm of 16 heterogeneous UAVs to monitor 10 randomly distributed targets. The task dictates that the swarm must perform collaborative observations following the strict “Detection-Interaction-Verification” temporal logic, while rigorously evading 4 known threat zones. The parameters related to UAVs, targets, and threat zones are provided in Table 3, Table 4, and Table 5, respectively. The UAV parameters are adopted from [3]. The task reward is set to a fixed value of 5, and the weighting coefficients are set as $\alpha = 0.55, \beta = 0.45$. The baseline rewards for the detection, interaction, and verification tasks are specified as 10, 5, and 5, with corresponding temporal discount factors of 0.05, 0.1, and 0.1, respectively. Furthermore, the expected task completion time is constrained to 1500 s, the maximum flight range for each UAV is 100 km, and the flight range cost coefficient is set to 0.1.

The maximum climb angle is $\gamma_{\max} = 30^\circ$ and the maximum climb angle rate is $\dot{\gamma}_{\max} = 1.5^\circ/\text{s}$. As shown in Figure 5, the UAV enters the initial transition phase after take-off. Constrained by the maximum climb angle rate, the climb angle increases to the maximum value of 30° at $t = 20$ s, after which the UAV enters the steady climb phase and continues climbing at the maximum climb angle. At $t = 124$ s, the UAV transitions to the altitude correction phase, during which the climb angle smoothly decays to zero, enabling the UAV to converge stably to the designated mission altitude.

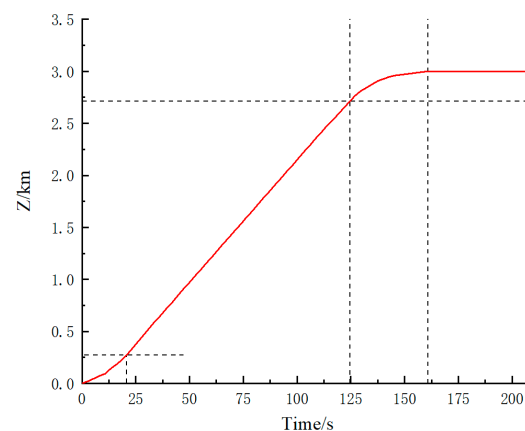


Figure 5. Fixed-wing UAV Climb Process.

The task allocation results, the timing diagram of task execution by each UAV, and the three-dimensional and two-dimensional diagrams of task allocation results are shown in Table 6, Figures 6–8, respectively. It should be noted that in Figures 6–8, blue indicates detection tasks, orange indicates interaction tasks, yellow indicates verification tasks, and the task execution sequence follows the precedence dependency relationship of “Detection-Interaction-Verification”. Taking Target 8 as an example, the process is specifically as follows: the resource demand vector of Target 8 is [3,2,1], which indicates that during the complete task cycle, this target requires 3 Type-1 UAVs equipped with SAR payloads to collaboratively execute search, 2 Type-2 UAVs carrying hyperspectral imagers to perform fine interaction, and 1 Type-3 UAV equipped with an EO/IR camera to complete final verification. Detection is a high-level task that must be completed before low-level tasks (Interaction and Verification) can be executed. The allocation results are shown in Table 6

and Figure 5; the SECBBA achieves precise scheduling within temporal and spatial ranges: UAVs 4, 5, and 6 are assigned to jointly execute the detection phase of Target 8, followed by UAVs 11 and 12 assigned to execute the interaction phase, and finally UAV 16 assigned to complete the verification task. Notably, the synchronous launch time of 679.2 s for UAVs 4, 5, and 6 in the Target 8 detection task in Table 6 is a design outcome of Algorithm 4: the algorithm uniformly updates the start time based on the longest estimated Dubins curves flight time among the three UAVs, and the time difference for UAVs arriving earlier is incorporated into the waiting time t_{ij}^{wait} in Equation (13), thereby compensating for spatial disparities through the “first-arrival waiting” strategy. Regarding communication overhead, the distributed negotiation prior to 679.2 s was completed during the ground preparation phase through the conflict resolution mechanism of Algorithm 2; during the task execution phase, the swarm merely follows the pre-generated unified temporal schedule without requiring real-time communication synchronization. This allocation result satisfies temporal constraints among reconnaissance, strike, and evaluation tasks for the same target area, multi-UAV execution tasks satisfy synchronization constraints, the task allocation scheme is deadlock-free, and the influence of threat zones is considered in task allocation with flyable paths satisfying curvature constraints planned synchronously.

Table 3. UAV Parameters.

UAV ID	Location/km	Payload Resources	Maximum Speed/(m/s)	Minimum Turning Radius/m
1	(2, 7)	[1, 0, 0]	50	200
2	(14, 5)	[1, 0, 0]	50	200
3	(22, 8)	[1, 0, 0]	50	200
4	(26, 7)	[1, 0, 0]	50	200
5	(30, 6)	[1, 0, 0]	50	200
6	(41, 6)	[1, 0, 0]	50	200
7	(6, 2)	[0, 3, 0]	45	220
8	(10, 1)	[0, 3, 0]	45	220
9	(19, 6)	[0, 3, 0]	45	220

Table 4. Target Area Parameters.

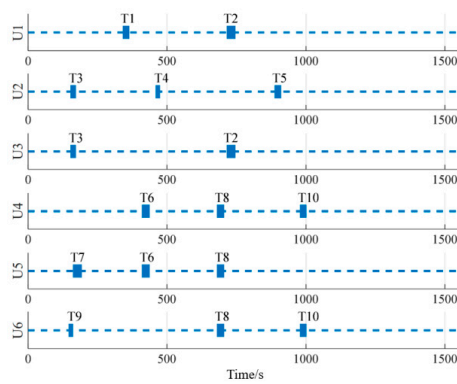
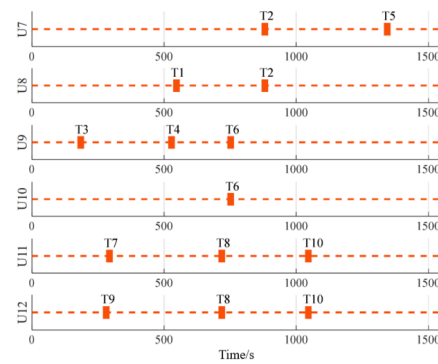
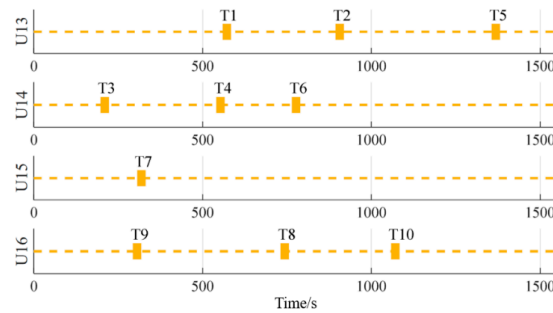
Target Area ID	Location/km	Area/km ²	Payload Requirements
1	(3,24)	0.3	[1,1,1]
2	(8,37)	0.8	[2,2,1]
3	(17,12)	0.5	[2,1,1]
4	(20,26)	0.2	[1,1,1]
5	(26,45)	0.3	[1,1,1]
9	(28,22)	0.7	[2,2,1]
7	(34,13)	0.4	[1,1,1]
8	(39,27)	1.0	[3,2,1]
9	(43,13)	0.2	[1,1,1]
10	(47,38)	0.6	[2,2,1]

Table 5. Threat Zone Parameters.

Threat Zone ID	Location/km	Radius/km
1	(11,20)	5
2	(21,35)	4
3	(27,14)	3
4	(36,38)	4

Table 6. Task Allocation Results.

UAV ID	Task Path	Task Start Time/s
1	12	340,6,718.0
2	3,4,5	152,3,458,7,885.0
3	32	152,3,718.0
4	6,8,10	409,6,679,2,977.9
5	7,6,8	161,2,409,6,679.2
6	9,8,10	145,6,679,2,977.9
7	25	868.8,1333.2
8	12	534,3,868.8
9	3,4,6	172,3,515,5,739.3
10	6	739.3
11	7,8,10	281,1,705,9,1033.2
12	9,8,10	268,5,705,9,1033.2
13	1,2,5	559,3,893,8,1356.5
14	3,4,6	243,3,549,7,764.3
15	7	306.1
16	9,8,10	293,5,730,9,1058.2

**(a)** Gantt Chart of Detection Tasks**(b)** Gantt Chart of Interaction Tasks**(c)** Gantt Chart of Verification**Figure 6.** Gantt Chart of Task Sequencing for Each UAV.

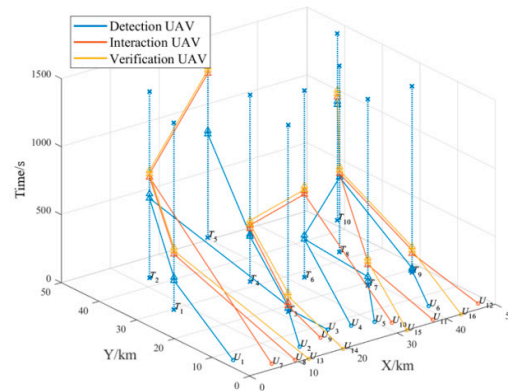


Figure 7. Spatiotemporal Illustration of Task Allocation Results.

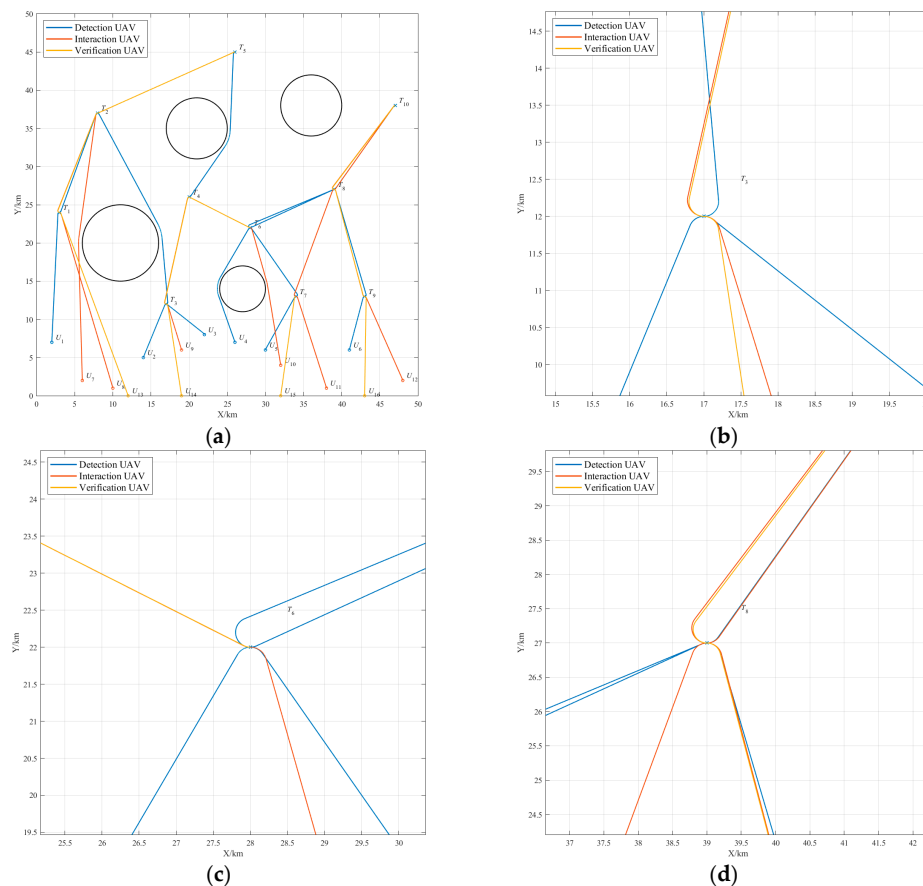


Figure 8. 2D Schematic of Task Allocation Results. (a) Global task planning diagram; (b) Close-up of Task 3; (c) Close-up of Task 6; (d) Close-up of Task 8.

3.2. UAV Obstacle-Avoidance Scenario

To validate the effectiveness of the proposed obstacle-avoidance strategy in three-dimensional dynamic conflict scenarios, a cooperative landing scenario involving three fixed-wing UAVs is constructed.

As shown in Figure 9, a trajectory conflict arises between UAV₁ and UAV₂ near the landing zone during the descent phase. Since the remaining fuel of UAV₂ exceeds 50% and UAV₁ possesses a higher landing priority, UAV₂ executes a loiter-and-wait maneuver in accordance with the proposed decision-making mechanism, thereby delaying its arrival at the landing point. Specifically, UAV₂ performs a spiral descent maneuver at 40% of its

original descent trajectory, during which the UAV maintains a constant velocity, and both the turning radius and climb angle satisfy the prescribed constraints.

Subsequently, a potential trajectory conflict arises between UAV₁ and UAV₃. Since the remaining fuel of UAV₃ falls below 50% of its total fuel capacity and its priority is lower than that of UAV₁, the altitude adjustment strategy is selected. Specifically, when the inter-UAV distance exceeds the safe separation threshold, UAV₃ first gradually increases its climb angle through the transition segment, then enters the steady climb phase to reach a temporary safe altitude layer, and completes a smooth trajectory transition through the altitude correction segment. Once the inter-UAV distance again exceeds the minimum safety separation, UAV₃ resumes the descent operation.

The simulation results demonstrate that the proposed method can effectively maintain safety separation between UAVs while satisfying fixed-wing kinematic constraints, achieving conflict resolution and safe landing.

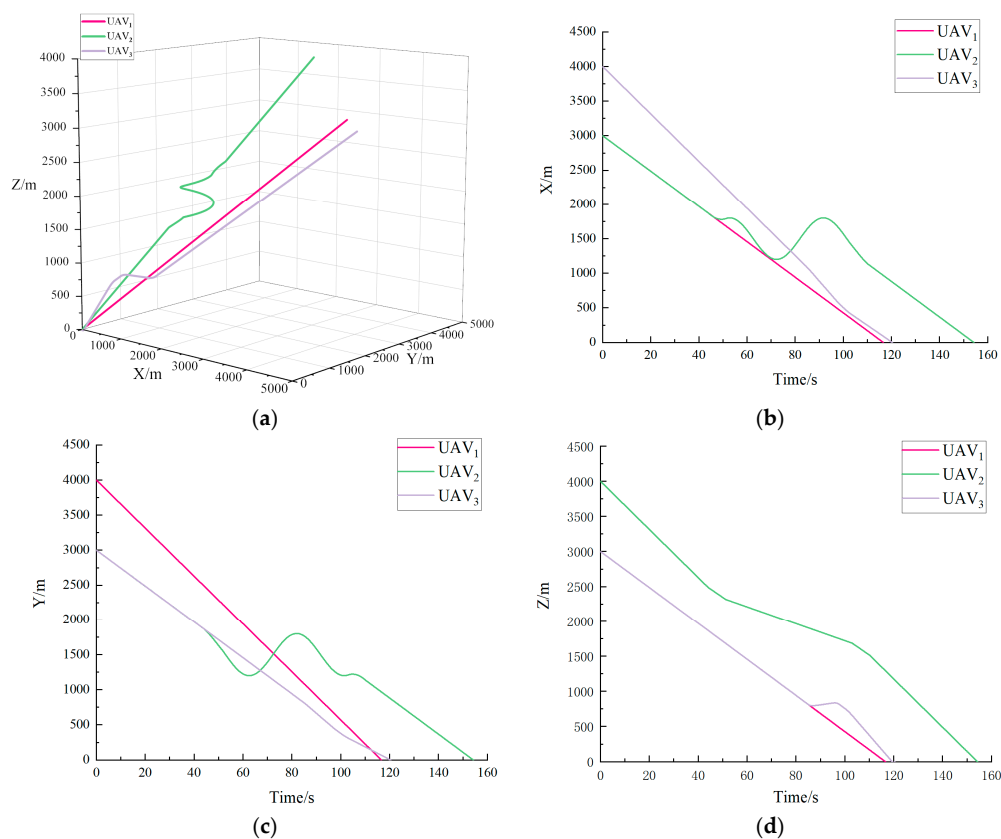


Figure 9. Schematic diagram of obstacle avoidance for a constant-speed fixed-wing UAV: (a) Overall schematic diagram; (b) X-T curve; (c) Y-T curve; (d) Z-T curve.

3.3. Algorithm Comparison

To evaluate the optimality, timeliness, and robustness of the proposed SECBBa in heterogeneous swarm cooperation, this paper adopts an improved genetic algorithm (IGA) under a centralized architecture and a representative distributed algorithm with temporal constraint handling capability (CBBA-TCC), and a distributed reinforcement learning algorithm (Q-Learning) as baseline methods for comparative analysis. The motivation for selecting IGA is to use it as a reference benchmark for global search optimization capability, thereby highlighting the differences in the distributed framework in terms of time delay and efficiency in large-scale problem solving. Second, the inclusion of CBBA-

TCC aims to provide a baseline within the same distributed consensus paradigm, allowing the proposed deadlock-correction mechanism and hierarchical coupling strategy to demonstrate their superior convergence robustness when handling tasks with strong logical dependencies. Finally, Q-Learning is introduced to compare the convergence and reliability of a data-driven decision strategy against a market-based auction consensus framework under strong logical dependencies and physical constraints. For IGA, the population size is 200, the maximum number of iterations is 600, the crossover probability is 0.8, the mutation probability is 0.2, and two elite individuals are retained per generation. For Q-learning, a Q-table with uniform dimensions of $N_V \times N_T$ is adopted. Action selection follows an ϵ -greedy policy, with an initial exploration rate of 0.5 that adaptively decays to 0.02 over 80% of the training episodes. The learning rate is set to 0.15, and the number of training $Episodes = 800 \times N_V \times N_T$. Algorithm results are evaluated using the global objective function with weights of 0.55 and 0.45.

First, to assess the scalability of the algorithms across varying task scales, a test suite comprising 8 UAVs and 1 to 12 randomly distributed targets was established. As depicted in Figure 10, SECBBA, CBBA-TCC, IGA, and Q-Learning are 1084.8, 1251.6, 1403.4, and 1791.0, respectively. Quantitative analysis reveals that SECBBA achieves cost reductions of 13.3%, 22.7%, and 39.4% compared to CBBA-TCC, IGA, and Q-Learning, respectively.

This performance disparity is primarily attributed to the inherent “Curse of Dimensionality” characterizing tightly coupled multi-stage heterogeneous UAV tasks. The strict spatio-temporal coupling of the multi-stage task chain, superimposed with Dubins kinematic constraints and multi-threat zone environments, results in physically feasible solutions being extremely sparse and exhibiting exponential growth within the discrete search space. The trial-and-error exploration strategy of Q-Learning cannot reliably capture the optimal, hard-constraint-satisfying action sequence within limited training episodes. Furthermore, its individual-completion-time-based credit assignment mechanism non-linearly maps to the global cost objective, hindering accurate assessment of a specific action’s contribution to reducing the global cost. Although IGA theoretically possesses global search potential, within a finite computational iteration window, it struggles to efficiently construct valid chromosomes that simultaneously satisfy all precedence, swarm capacity, and physical flight constraints. In contrast, SECBBA leverages a “valid-by-design” bundle construction mechanism, successfully decoupling high-dimensional global constraints into tractable local subproblems, thereby achieving rapid convergence towards high-quality feasible solutions.

Furthermore, to evaluate the timeliness of the proposed algorithm, a comparative analysis of execution time under varying problem scales is conducted, as shown in Figure 11 and algorithm. Regarding real-time performance evaluation, Figure 11 illustrates the comparison of algorithm execution times as the problem scale varies. As the task scale expands, the computation time of IGA increases significantly (Table 7). Specifically, in medium-scale scenarios, SECBBA requires less than 0.5 s to generate a feasible scheduling scheme, whereas both Q-Learning and IGA require over 10 s to converge to a stable solution. Such low computational complexity implies that SECBBA is highly suitable for deployment on embedded mission computers with constrained processing capabilities. The algorithm’s capacity to generate feasible scheduling schemes with negligible latency enables the UAV swarm to perform dynamic replanning in response to emergent targets or threats, thereby fulfilling the strict timeliness requirements of data acquisition missions.

Table 7. Computation Time Comparison of Algorithms(s).

Number of Tasks	SECBBA	CBBA-TCC	IGA	Q-Learning
1	0.14	0.45	11.87	10.30
2	0.12	0.21	13.13	10.81
3	0.12	0.11	15.06	11.18
4	0.15	0.16	17.16	9.07
5	0.14	0.22	18.58	11.23
6	0.21	0.32	20.19	10.23
7	0.16	0.22	23.76	12.68
8	0.18	0.15	26.37	11.92
9	0.34	0.46	28.25	13.12
10	0.19	0.25	27.87	11.17
11	0.20	0.18	29.48	14.91
12	0.27	0.44	34.61	17.32

Then, to further evaluate the optimality and timeliness of the algorithm, the numbers of targets and UAVs are fixed, with 8 UAVs assigned to execute tasks on 10 targets. A total of 50 Monte Carlo simulation experiments are conducted, and the algorithm's performance is evaluated based on the results of these simulations [30]. The scoring mechanism is as follows:

$$Q = \sum_{i=1}^{N_m} \left(a \times \frac{\text{score}_{\text{SECBBA}}^{\text{ave}}}{\text{score}_i} + b \times \frac{t_{\text{SECBBA}}^{\text{ave}}}{t_i} \right), a+b=1 \quad (32)$$

In the equation, N_m represent the number of Monte Carlo simulations, while $\text{score}_{\text{SECBBA}}^{\text{ave}}$ and $t_{\text{SECBBA}}^{\text{ave}}$ denote the average objective function value and average computation time of the SECBBA, respectively. The variables score_i and t_i correspond to the objective function value and computation time of the i -th simulation for each algorithm. This scoring scheme is normalized against SECBBA as the baseline. When a compared algorithm exhibits poorer solution quality or longer computation time, its corresponding sub-score ratio decreases, and a lower Q-value indicates more inferior comprehensive performance relative to SECBBA. The weighting coefficients a are assigned values ranging from 0.3 to 0.7, with an interval of 0.1, to evaluate the optimality and efficiency of the algorithms.

Figures 12–14 present the statistical performance metrics derived from the Monte Carlo simulation experiments across the evaluated algorithms. The empirical data demonstrate that the SECBBA exhibits a pronounced superiority in solution quality. Its average objective function value converges to 1324.7, ultimately yielding substantial performance gains of 9.8%, 23.2%, and 19.0% over the baseline algorithms, respectively. These outcomes indicate that, under equivalent task scale constraints, SECBBA possesses superior global optimization capabilities and enhanced solution stability.

Finally, to evaluate the robustness of the algorithm, simulations were conducted under five different sets of weighting coefficients, and the results demonstrate that the overall performance under varying coefficients is superior to that of the other three algorithms. Therefore, the proposed SECBBA is able to achieve superior objective function values while maintaining comparable computational efficiency.

In summary, SECBBA is capable of maintaining both high-quality solutions and high computational efficiency across a range of task scales. Based on an analysis of the algorithm implementation logic and complexity, SECBBA can handle up to 100 fixed-wing UAV operations simultaneously.

Although simulations have verified the effectiveness of the SECBBA in handling complex logical constraints, limitations remain in practical applications. First, regarding

application feasibility, the current simulation environment incorporates a certain degree of idealized assumptions in modeling UAV SWaP limitations and onboard computing capabilities; future embedded hardware-in-the-loop testing is required to further verify the real-time computational efficiency of the algorithm. Second, concerning communication delay and perceptual uncertainty, although the distributed consensus mechanism of SEC-BBA theoretically possesses logical convergence resilience under jointly connected network conditions, bandwidth limitations and signal attenuation on non-ideal communication links may lead to non-deterministic fluctuations in distributed consensus cycles, while measurement noise at the perception level can also cause deviations in path cost prediction. Furthermore, this paper has not yet conducted an in-depth discussion on swarm performance degradation caused by random node failures; such sudden failure scenarios impose more stringent real-time requirements on the algorithm's dynamic reallocation and fault tolerance mechanisms.

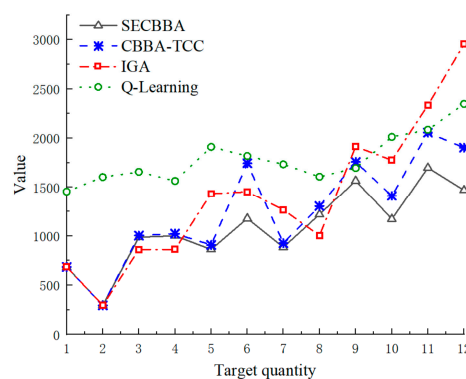


Figure 10. Comparison of Objective Function Values for Task Allocation Algorithms.

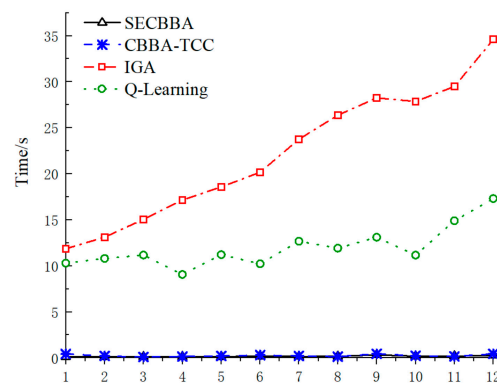


Figure 11. Comparison of Time Consumption in Solving Task Allocation Algorithms.

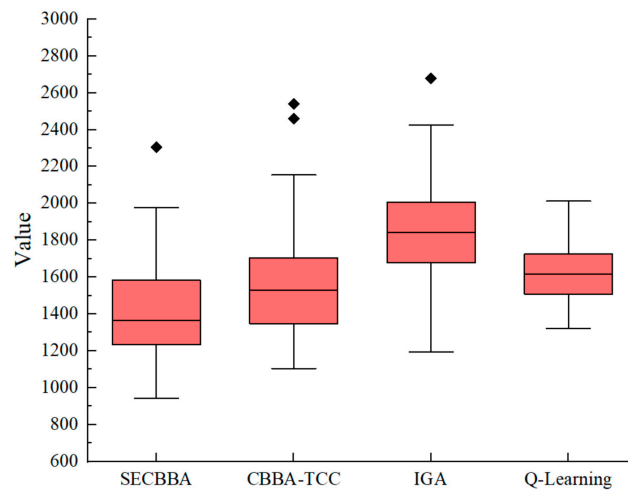


Figure 12. Comparison of Algorithm Objective Function Values.

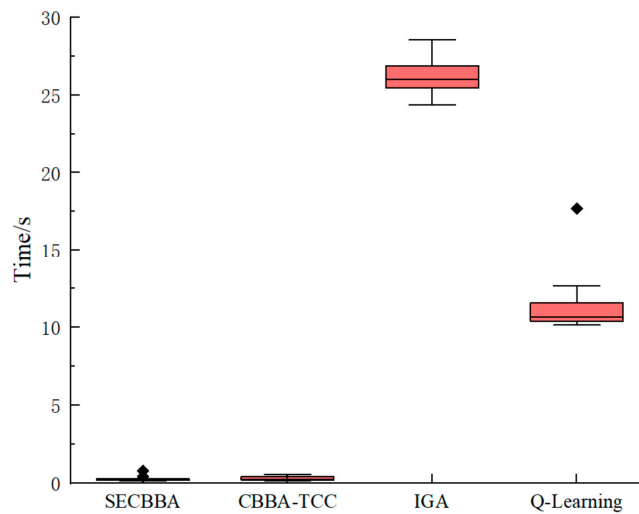


Figure 13. Comparison of Algorithm Computation Times.

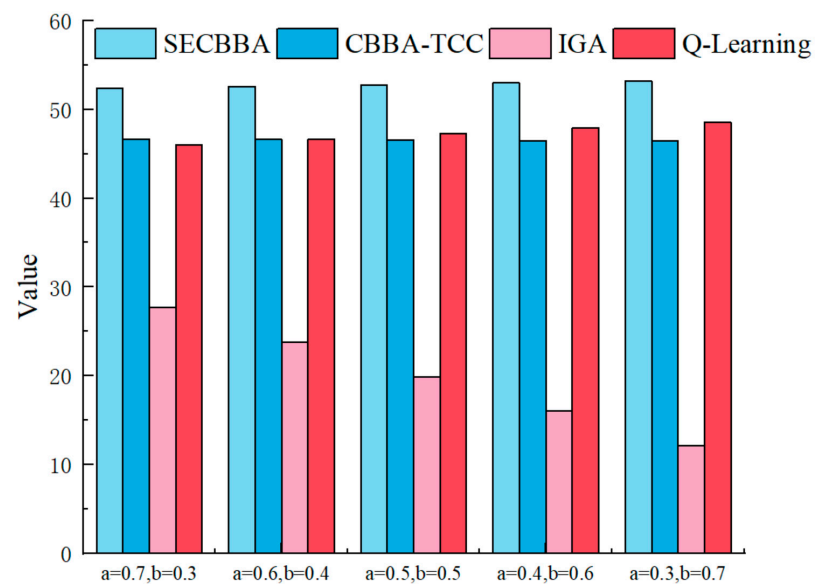


Figure 14. Comparison of Overall Algorithm Performance.

4. Conclusions

To address the NP-hard combinatorial optimization challenge of task allocation faced by heterogeneous drone swarms during coupled multi-stage tasks, this paper proposes SECBBA framework. By integrating task scheduling with UAV kinematics, it effectively solves the distributed task allocation problem under multiple constraints.

Specifically, the introduction of a mechanism based on directed graph DFS fundamentally avoids deadlock in drone allocation, ensuring robust and conflict-free assignment schemes. Furthermore, converting global temporal constraints into manageable soft time-window constraints significantly reduces computational complexity. By coupling Dubins curve path planning with the allocation process, SECBBA guarantees that generated drone flight trajectories satisfy constraints, achieving efficient global coordination.

In quantitative terms, simulation results demonstrate that SECBBA significantly outperforms both the CBBA-TCC, IGA, and Q-Learning algorithms baselines in terms of solution quality. Specifically, across scenarios with varying target scales, SECBBA achieves a notable reduction in the average objective function cost by 13.3%, 22.7%, and 39.4% CBBA-TCC, IGA, and Q-Learning, respectively, while sustaining superior computational timeliness. Furthermore, Monte Carlo simulations corroborate that the proposed algorithm yields substantial performance enhancements of 9.8%, 23.2%, and 19.0% under diverse weighting configurations, verifying its robust global optimization capabilities and decision stability.

Building upon the validated capabilities demonstrated above, future research will focus on extending the SECBBA framework to dynamic environments. This encompasses addressing UAV allocation challenges under fault-tolerant conditions, specifically UAV attrition, mobile task points, emergent threat zones, and newly added task points, thereby further enhancing the autonomy and robustness of heterogeneous UAV swarms.

Author Contributions: B.Y. and F.G. conceived and designed the experiments; W.W. and H.C. performed the experiments; Y.L., Q.C. and B.Y. analyzed the data; Q.Y. wrote the paper; B.Y. and F.G. contributed to data analysis revision; G.L. and W.W. contributed to English language corrections. All authors of article provided substantive comments. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by the National Natural Science Foundation of China (Grant Nos. 52272369 and U2570206).

Data Availability Statement: The original contributions presented in this study are included in the article.

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

Abbreviations

The following abbreviations are used in this manuscript:

SECBBA	Sequentially Extended Consensus-Based Bundle Algorithm
CBBA	Consensus-Based Bundle Algorithm
CBBA-TCC	Consensus-Based Bundle Algorithm with Temporal Consistency Constraints
SAR	Synthetic Aperture Radar
UAV	Unmanned Aerial Vehicle
IGA	Improved Genetic Algorithm
MILP	Mixed-Integer Linear Programming
DFS	Depth-First Search
GA	Genetic Algorithm
WPA	Wolf Pack Algorithm
MDWPA	Multi-Discrete Wolf Pack Algorithm

RWS	Roulette Wheel Selection
ES	Elitist Strategies
ACO	Ant Colony Optimization
SWaP	Size, Weight, and Power
CLC	Curvature-Limited Circle

References

- Nex, F.; Remondino, F. UAV for 3D mapping applications: A review. *Appl. Geomat.* **2014**, *6*, 1–15.
- Al-Kaff, A.; Martín, D.; García, F. Survey of computer vision algorithms and applications for unmanned aerial vehicles. *Expert Syst. Appl.* **2018**, *92*, 347–363.
- Cui, W.; Li, R.; Feng, Y.; Yang, Y. Distributed Task Allocation for a Multi-UAV System with Time Window Constraints. *Drones* **2022**, *6*, 226.
- Liu, H.; Shao, Z.; Zhou, Q.; Tu, J.; Zhu, S. Task Allocation Algorithm for Heterogeneous UAV Swarm with Temporal Task Chains. *Drones* **2025**, *9*, 574.
- Song, J.; Zhao, K.; Liu, Y. Survey on Mission Planning of Multiple Unmanned Aerial Vehicles. *Aerospace* **2023**, *10*, 208.
- Erdelj, M.; Natalizio, E.; Chowdhury, K.R. Help from the Sky: Leveraging UAVs for Disaster Management. *IEEE Pervasive Comput.* **2017**, *16*, 24–32.
- Wu, J.; Yu, Y.; Ma, J.; Wu, J.; Han, G.; Shi, J.; Gao, L. Autonomous Cooperative Flocking for Heterogeneous Unmanned Aerial Vehicle Group. *IEEE Trans. Veh. Technol.* **2021**, *70*, 12477–12490.
- Beard, R.W.; McLain, T.W.; Goodrich, M.A. Coordinated target assignment and intercept for unmanned air vehicles. *IEEE Trans. Robot. Autom.* **2003**, *18*, 911–922.
- Xu, S.; Li, L.; Zhou, Z.; Mao, Y.; Huang, J. A Task Allocation Strategy of the UAV Swarm Based on Multi-Discrete Wolf Pack Algorithm. *Appl. Sci.* **2022**, *12*, 1331.
- Dong, X.; Shi, C.; Wen, W.; Zhou, J. Multi-Mission Oriented Joint Optimization of Task Assignment and Flight Path Planning for Heterogeneous UAV Cluster. *Remote Sens.* **2023**, *15*, 5315.
- Chen, P.; Yan, F.; Liu, Z.; Cheng, G. Communication-Constrained Task Allocation of Heterogeneous UAVs. *Acta Aeronaut. Astronaut. Sin.* **2021**, *42*, 313–326.
- Choi, H.L.; Brunet, L.; How, J.P. Consensus-Based Decentralized Auctions for Robust Task Allocation. *IEEE Trans. Robot.* **2009**, *25*, 912–926.
- Yue, W.; Zhang, X.; Liu, Z. Distributed cooperative task allocation for heterogeneous UAV swarms under complex constraints. *Comput. Commun.* **2025**, *231*, 108043.
- Sun, P.; Yang, G.; Xu, X.; Zhang, J.; Deng, X.; Zhang, Y.; Cui, J. Distributed Task Allocation Algorithm for Heterogeneous UAVs Based on Reinforcement Learning. *Drones* **2026**, *10*, 220.
- Li, X.M.; Tang, J.Y.; Dai, J.J.; Bo, N. Dynamic coalition task allocation of heterogeneous multiple agents. *J. Northwest. Polytech. Univ.* **2020**, *38*, 1094–1104.
- Chen, J.; Qing, X.G.; Ye, F.; Xiao, K.; You, K.; Sun, Q. Consensus-based bundle algorithm with local replanning for heterogeneous multi-UAV system in the time-sensitive and dynamic environment. *J. Supercomput.* **2022**, *78*, 1712–1740.
- Zhang, Y.; Xu, W.; Ye, H.; Shi, Z. A Two-Stage Optimization Framework for UAV Fleet Sizing and Task Allocation in Emergency Logistics Using the GWO and CBBA. *Drones* **2025**, *9*, 501.
- Qi, G.; Niu, Y.; Gao, Y. Task allocation and path planning for multi-UAV search and rescue in complex environments. *Simul. Model. Pract. Theory* **2026**, *149*, 103283.
- Alqefari, S.; Menai, M.E.B. Multi-UAV Task Assignment in Dynamic Environments: Current Trends and Future Directions. *Drones* **2025**, *9*, 75.
- Chen, Y.; Chen, R.; Huang, Y.; Xiong, Z.; Li, J. Distributed Task Allocation for Multiple UAVs Based on Swarm Benefit Optimization. *Drones* **2024**, *8*, 766.
- Tan, Y.; Zhou, C.; Qian, F. Cooperative task allocation method for multi-unmanned aerial vehicles based on the modified genetic algorithm. *IET Intell. Transp. Syst.* **2024**, *18*, 1164–1173.
- Chen, J.; Xiao, K.; You, K.; Qing, X.; Ye, F.; Sun, Q. Hierarchical task assignment strategy for heterogeneous multi-UAV system in large-scale search and rescue scenarios. *Int. J. Aerosp. Eng.* **2021**, *2021*, 7353697.
- Zeng, R.; Luo, R.; Liu, B. UAV Path Planning for Forest Firefighting Using Optimized Multi-Objective Jellyfish Search Algorithm. *Mathematics* **2025**, *13*, 2745.

24. Chandrawanshi, V.S.; Tripathi, R.K.; Pachauri, R. An intelligent energy efficient clustering technique for multiple base stations positioning in a wireless sensor network. *J. Intell. Fuzzy Syst.* **2019**, *37*, 3601–3613.
25. Li, M.; Sun, P.F.; Jeon, S.W.; Zhao, X.; Jin, H. Multi-UAV Path Planning with Genetic Algorithm. In Proceedings of the 2023 14th ICTC, Jeju Island, Republic of Korea, 11–13 October 2023; pp. 918–920.
26. Yuan, Y.; Shen, X.; Pan, W. Constrained likelihood for reconstructing a directed acyclic Gaussian graph. *Biometrika* **2019**, *106*, 109–125.
27. Shanmugavel, M.; Tsourdos, A.; White, B. Cooperative path planning of multiple UAVs using Dubins paths with clothoid arcs. *Control Eng. Pract.* **2010**, *18*, 1084–1092.
28. Shanmugavel, M.; Tsourdos, A.; Zbikowski, R. Path Planning of Multiple UAVs Using Dubins Sets. In Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO, USA, 21–24 August 2006; p. 6142.
29. Owen, M.; Beard, R.W.; McLain, T.W. Implementing Dubins Airplane Paths on Fixed-Wing UAVs. In *Handbook of Unmanned Aerial Vehicles*; Valavanis, K.P., Vachtsevanos, G.J., Eds.; Springer: Dordrecht, The Netherlands, 2014; pp. 1677–1701.
30. Wu, W.; Chen, N.G.; Guo, J.F. Distributed task assignment method based on local information consensus and target estimation. *Control Theory Appl.* **2018**, *35*, 468–477.

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.