I.J. Wireless and Microwave Technologies, **2026, 4, 1-15**
Published Online on August 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijwmt.2026.04.01

A Multi-UAV Planning Framework for Task Allocation, Route Optimization and Trajectory Smoothing

Mykola Nikolaiev*
Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" Kyiv, 03056, Ukraine
E-mail: nickolay.dev@gmail.com
ORCID iD: https://orcid.org/0009-0003-0787-6726
*Corresponding Author

Mykhailo Novotarskyi
Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" Kyiv, 03056, Ukraine
E-mail: novotar@gmail.com
ORCID iD: https://orcid.org/0000-0002-5653-8518

Received: March 15, 2026; Revised: May 31, 2026; Accepted: July 20, 2026; Published: August 8, 2026

Abstract: Coordinated mission planning for multiple unmanned aerial vehicles in cluttered static three-dimensional environments requires consistent treatment of obstacle-aware motion, fleet-level task allocation, route sequencing, and executable trajectory generation. In many existing approaches, these elements are optimized separately, or fleet-level decisions are made using simplified geometric distances that do not accurately reflect UAV-specific motion feasibility in obstacle-constrained space. This paper presents a Multi-UAV Planning Framework for Task Allocation, Route Optimization and Trajectory Smoothing for static environments with known obstacle geometry. In the first stage, an offline single-UAV planner based on a hybrid Differential Evolution and Enhanced Whale Optimization Algorithm computes feasible raw paths for all relevant ordered node pairs and constructs a UAV-specific directed travel-cost matrix. In the second stage, these planner-derived matrices are used for feasibility-aware balanced task distribution and route optimization with exchange-based refinement under a composite total-cost–makespan objective. In the third stage, the raw paths corresponding to the final selected routes are reconstructed and transformed into executable trajectories by adaptive cubic B-spline smoothing. Experimental evaluation was conducted at the local-planning, fleet-planning, and smoothing levels in known static environments. The hybrid planner generated high-quality pairwise obstacle-avoiding paths and exhibited favorable convergence behavior relative to standard WOA, PSO, DE, SOS, and GWO in the tested scenarios. At the fleet level, the full framework reduced makespan by 2.1–3.4% and the composite objective by 0.9–1.4% relative to balanced partitioning without exchange refinement on benchmark instances. In the smoothing stage, the adaptive cubic B-spline reduced path length by 8.5% and maximum curvature by 41.5% relative to the unsmoothed polyline representation. These results demonstrate that the proposed hierarchical formulation is computationally effective, physically consistent, and well suited to multi-UAV mission planning.

Index Terms: UAV, task allocation, route optimization, path planning, trajectory smoothing, optimization, mission planning

1. Introduction

The increasing use of unmanned aerial vehicles (UAVs) in inspection, surveillance, mapping, environmental monitoring, precision logistics, and infrastructure servicing has intensified the need for robust mission-planning methods capable of coordinating multiple vehicles in obstacle-rich three-dimensional environments [1,2]. In many practical deployments, the workspace is largely known in advance: buildings, towers, protected volumes, terrain boundaries, and mission targets can be specified before execution, and the planner is expected to generate safe and efficient trajectories for a fleet operating from a common depot [3,4]. In such settings, the planning problem is not limited to finding a single collision-free path. Rather, it requires simultaneous treatment of task allocation across vehicles, route sequencing for each assigned task subset, and generation of smooth executable trajectories that respect vehicle-specific motion limits [5,6]. Achieving all three objectives within a single coherent framework remains a demanding problem in multi-UAV autonomy.

A central difficulty arises from the coexistence of two fundamentally different planning scales. At the local scale, a UAV must traverse cluttered three-dimensional space while avoiding static obstacles, controlling altitude changes, and maintaining feasible motion. This is a continuous optimization problem over geometrically constrained trajectory space. At the fleet scale, the planner must determine which vehicle should service which task and in what order, which is a discrete combinatorial problem with strong coupling through endurance, workload balance, and completion time [7,8]. In many existing formulations, these two scales are handled separately with only weak information exchange between them. Local path planning is often performed independently of task assignment, while fleet-level optimization frequently relies on Euclidean distances or other coarse surrogates that do not capture obstacle-induced detours, direction-dependent travel costs, or UAV-specific motion characteristics. As a result, task assignments that appear efficient under simplified distance models may be suboptimal or even infeasible when translated into actual obstacle-aware motion [9].

A related limitation concerns the representation of the final trajectory. Even when assignment and route order are well chosen, the resulting path is often represented as a polyline or as a sequence of locally planned arcs that may contain abrupt turns, irregular curvature, or unnecessary geometric complexity [10]. For real UAV platforms, such trajectories can degrade tracking accuracy, increase control effort, and consume additional energy. Parametric smoothing methods, particularly B-spline-based formulations, offer a mathematically attractive mechanism for improving trajectory quality because they provide local control, continuity of derivatives, and direct access to curvature and acceleration information. However, smoothing is frequently treated as a generic post-processing step detached from upstream mission planning, rather than as a stage integrated with the same obstacle-aware geometric information that informed the routing decisions [11]. This separation can lead to inconsistencies between the route selected by the mission planner and the trajectory ultimately delivered for execution.

The static-environment setting provides an opportunity to address these issues more systematically. When obstacle geometry is known and fixed, it becomes possible to compute a reusable database of pairwise feasible paths and corresponding motion-aware travel costs before fleet-level optimization begins. Such a database can serve as an interface between continuous path planning and discrete mission planning: expensive geometric reasoning is performed once offline, while higher-level task allocation and route optimization operate on a directed graph whose edge weights already reflect obstacle avoidance and UAV-specific traversal effort. If this graph-based planning stage is then followed by trajectory smoothing that reuses the stored pairwise paths selected in the final solution, the full planning pipeline becomes both computationally tractable and internally consistent.

Motivated by this observation, this paper proposes a Multi-UAV Planning Framework for Task Allocation, Route Optimization and Trajectory Smoothing. The framework is organized into three computational stages. In the first stage, a hybrid Differential Evolution (DE) and Enhanced Whale Optimization Algorithm (E-WOA) is used to solve a set of offline single-UAV pairwise path-planning problems. For each UAV and each relevant ordered node pair, this stage yields both a feasible raw path and a scalar planner-derived travel cost, from which a directed cost matrix is constructed. In the second stage, these matrices are used for feasibility-aware multi-UAV task distribution and route optimization. A balanced spatial partition guided by a surrogate workload model provides an initial assignment, after which depot-to-depot routes are optimized on the exact planner-derived matrices and refined through inter-route task exchanges under a combined total-cost–makespan objective. In the third stage, only the raw paths associated with the final selected routes are retrieved from the offline database and converted into smooth executable trajectories by adaptive cubic B-spline fitting and time scaling.

The methodological contribution of the paper is therefore not merely the introduction of an additional routing heuristic or smoothing routine, but the coupling of three planning layers that are often treated independently. First, obstacle-aware local motion is encoded into a reusable directed cost structure rather than approximated geometrically at the fleet level. Second, multi-UAV assignment and routing are formulated on that motion-consistent structure, enabling the planner to consider workload balance and mission completion time without repeated geometric replanning. Third, smoothing is applied only after the discrete mission plan has been fixed, ensuring that the final trajectories preserve the selected route structure while improving curvature behavior and execution quality. This design is particularly well suited to static environments, where the validity of the planner-derived pairwise path database is preserved throughout the optimization process.

Importantly, the manuscript does not claim that joint multi-UAV assignment, routing, and smoothing are entirely new problems. Rather, the contribution lies in the specific information interface between these layers: obstacle-aware pairwise paths are computed offline, converted into UAV-specific directed travel-cost matrices, reused during fleet-level optimization, and then mapped back to geometry for route-consistent smoothing. This explicit reuse of motion-aware pairwise information differentiates the proposed framework from studies that address these layers jointly but do not preserve a reusable path-based interface across them.

The main contributions of this work are threefold. First, a hybrid DE+E-WOA local planner is used to generate UAV-specific directed travel-cost matrices and raw feasible pairwise paths in known static obstacle fields. Second, a fleet-level planning module is introduced that combines feasibility masking, balanced task partitioning, exact route optimization, and exchange-based refinement under a unified total-cost–makespan objective. Third, an adaptive cubic B-spline smoothing stage is integrated with the selected route geometry so that the final mission output consists of continuous, collision-free, and dynamically feasible trajectories rather than merely discrete node sequences.

The scope of the study is intentionally limited to static, common-depot environments; partially known maps and field deployment are outside the present validation scope. The remainder of the paper presents the proposed methodology, reports the experimental results at the local-planning, fleet-planning, and smoothing levels, and concludes with a discussion of the implications and limitations of the proposed framework.

2. Literature Review

Recent studies show that multi-UAV mission planning is increasingly treated as a coupled optimization problem involving task assignment, route generation, and path feasibility, rather than as a set of isolated subproblems. Contemporary review studies consistently emphasize four recurring challenges: heterogeneous vehicle capabilities, scalability with increasing fleet and task size, uncertainty in mission conditions, and the strong coupling between task allocation and motion planning. In parallel, the literature has moved away from purely geometric abstractions and toward more operationally realistic formulations in which timing, resource limits, communication structure, and motion feasibility are incorporated directly into the planning model [12,13].

A substantial body of recent research has focused on integrated multi-UAV mission planning. Recent formulations include joint sensor allocation and path planning for heterogeneous fleets [14], reconnaissance-task allocation with richer target representations such as point, line, and area targets [15], priority-aware joint task assignment and path planning [16], cooperative assignment under simultaneous-arrival and resource constraints [17], and heterogeneous mapping-task assignment with embedded path-planning considerations [18]. These studies clearly demonstrate the benefits of integrated modeling, especially when task value, coordination constraints, or vehicle heterogeneity must be represented explicitly. However, in most of these approaches, task assignment and routing are optimized in a tightly coupled fashion, and the fleet-level model still lacks a reusable obstacle-aware pairwise motion representation that could connect local path feasibility with high-level assignment decisions in a computationally efficient way [19,20].

Another active line of research addresses dynamic or uncertain environments. Recent studies have introduced dynamic time-slicing strategies, uncertainty-aware grey wolf optimization, adaptive sampling, and task-rationality review mechanisms to improve assignment quality when target information or mission conditions vary during execution [21,22]. These contributions are important because they extend mission planning beyond deterministic static settings and demonstrate how online adaptation can improve fleet responsiveness. Nevertheless, they are directed primarily at dynamic replanning rather than at exploiting the special structure of static environments, where obstacle geometry is fully known in advance and can therefore be processed offline. This distinction is central: in a static setting, it becomes feasible to compute a reusable database of pairwise obstacle-aware paths and costs before fleet-level optimization begins, which is methodologically different from the repeated online adjustment strategies emphasized in dynamic-allocation studies [23].

At the local-motion level, recent work on single-UAV trajectory planning continues to rely heavily on population-based metaheuristics for three-dimensional obstacle-constrained problems. Differential-evolution-based approaches have been shown to improve global exploration and convergence robustness in complex terrain and high-dimensional trajectory search spaces [24], while improved whale-optimization formulations have been reported to enhance optimization precision and reduce premature convergence in 3D path planning. At the same time, B-spline-based trajectory generation and smoothing has gained renewed attention because it offers local geometric control, derivative continuity, and direct curvature management. Recent studies have shown that adaptive B-spline trajectory generation can produce safe and feasible UAV motion [25], and that B-spline representations can be combined with obstacle-boundary modeling to improve three-dimensional collision-avoidance quality [26]. Despite these advances, the recent literature still tends to treat metaheuristic path planning, multi-UAV task distribution, and B-spline smoothing as separate methodological layers [25,27].

Taken together, the recent literature suggests a clear methodological gap. Strong progress has been made in integrated multi-UAV assignment and routing, in metaheuristic 3D path planning, and in spline-based trajectory generation, yet much less work connects these elements through a common reusable motion model. Accordingly, the contribution of the present study is not the first joint treatment of these components per se, but a hierarchical integration in which UAV-specific, obstacle-aware pairwise paths and travel-cost matrices serve as the interface between offline local planning, fleet-level optimization, and final trajectory smoothing in known static environments.

3. Methodology

The proposed framework addresses the multi-UAV mission-planning problem in a three-dimensional environment by decomposing it into three sequential stages. First, a single-UAV planner computes feasible point-to-point paths between all relevant ordered node pairs for each UAV and stores the corresponding planner-derived travel costs in a directed matrix $c_u(p,q)$. Second, these matrices are used for fleet-level task allocation and route optimization, including inter-route refinement. Third, only the paths that belong to the final multi-UAV solution are converted into smooth executable trajectories by adaptive cubic B-spline fitting. This separation is appropriate for static scenes because obstacle geometry is completely known in advance, allowing the expensive geometric planning stage to be executed once offline and reused during the combinatorial optimization stages. The overall workflow of the proposed methodology is illustrated in Fig. 1.

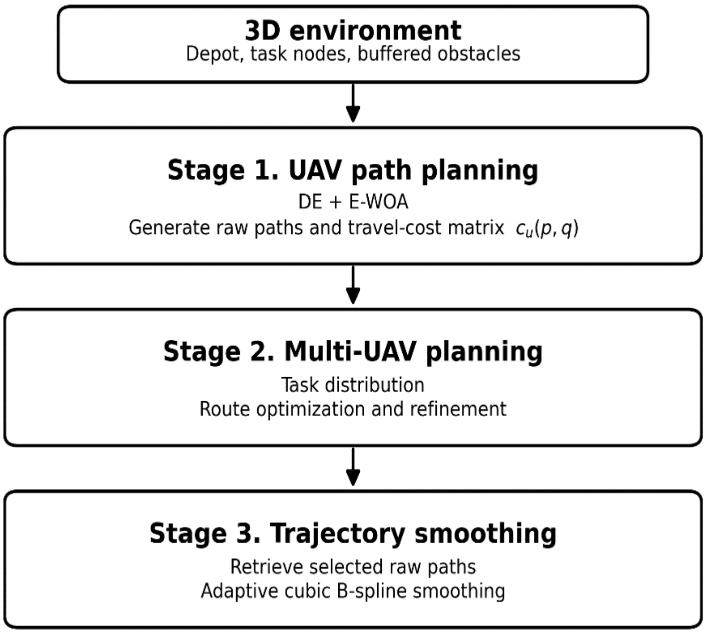

Fig. 1. Overall methodology workflow

3.1. Problem statement

This study considers a cooperative mission-planning problem for a fleet of M UAVs operating in a known static three-dimensional environment. The fleet must depart from a common depot, visit N task locations, perform the required service at each location, and return to the depot while avoiding obstacles and satisfying vehicle constraints.
Let $\mathcal{U} = \{1, \dots, M\}$ denote the set of UAVs and $\mathcal{N} = \{1, \dots, N\}$ the set of tasks. Each task $i \in \mathcal{N}$ is associated with a fixed position $x_i \in R^3$ and a service time s_i. The depot is indexed by node 0, so the full node set is $V = \{0,1, \dots, N\}$. The environment is represented by a bounded flight region $\Omega \subset R^3$ containing static obstacles $O = \{O_1, \dots, O_K\}$. Each obstacle is expanded by a safety margin, yielding buffered obstacles $\overline{O_\alpha}$. The admissible free space is therefore

$$\Omega_f = \Omega,\ \bigcup_{\alpha=1}^{K} \overline{O_\alpha} \tag{1}$$

Each UAV $u \in \mathcal{U}$ has its own operational limits, including maximum speed v_u^{max}, maximum acceleration a_u^{max}, maximum curvature κ_u^{max}, and endurance budget B_u. Since the fleet may be heterogeneous, the cost of traveling between the same two nodes may differ across UAVs.
The planning objective is to determine, for each UAV u, a task subset $S_u \subseteq \mathcal{N}$, a depot-to-depot visit order

$$\pi_u = \left(0, i_1, i_2, \dots, i_{|S_u|}, 0\right) \tag{2}$$

and a smooth trajectory $\Gamma_u(t)$. The task subsets must satisfy

$$\bigcup_{u=1}^{M} S_u = \mathrm{N}, \qquad S_u \cap S_v = \varnothing \quad \forall u \neq v \tag{3}$$

so that each task is assigned exactly once.

To support fleet-level planning, a single-UAV planner first computes a directed travel-cost matrix $c_u(p,q)$ for each UAV u and each ordered node pair (p,q). The value $c_u(p,q)$ represents the travel time of the best feasible path from node p to node q in the static obstacle field. Using this matrix, the route cost of UAV u is defined as

$$C_u\left(\pi_u\right)=\sum_{l=0}^{|S_u|} c_u\left(\pi_u\left(l\right),\pi_u\left(l+1\right)\right)+\sum_{i\in S_u} s_i \tag{4}$$

subject to the endurance constraint

$$C_u(\pi_u)\leq B_u \tag{5}$$

At the fleet level, the optimization seeks to minimize both total cost and mission completion time through the objective

$$F=\sum_{u=1}^{M} C_u\left(\pi_u\right)+\beta\max_{u\in \mathrm{U}} C_u\left(\pi_u\right), \qquad \beta\geq 0 \tag{6}$$

where β controls the trade-off between overall efficiency and makespan reduction.

After task allocation and route optimization, the selected paths are smoothed into executable trajectories. The final trajectory of each UAV must remain in free space and satisfy the vehicle limits on speed, acceleration, and curvature. Thus, the complete problem is to determine task assignments, route orders, and smoothed trajectories that satisfy all mission and feasibility constraints while minimizing the fleet-level objective.

3.2. UAV path planning

To support the subsequent multi-UAV task-allocation and route-optimization stages, a single-UAV path-planning procedure is first executed for each vehicle and for every relevant ordered node pair in the mission graph. Because the environment is static, this preprocessing step can be performed once before fleet-level optimization begins. Its purpose is to replace repeated low-level geometric planning with a reusable planner-derived representation of motion cost. Specifically, for each UAV u and each ordered pair of nodes (p,q), where $p,q\in V$ and $p\neq q$, the planner computes both a feasible raw path $\mathcal{P}_u(p,q)$ and a scalar travel cost $c_u(p,q)$. Collecting these values over all relevant ordered pairs yields a UAV-specific directed cost matrix $C_u=[c_u(p,q)]$.

This matrix constitutes the motion-aware interface between local path planning and higher-level multi-UAV optimization.

The use of a precomputed matrix is particularly important in the present framework for three reasons. First, the fleet may be heterogeneous, so two UAVs traveling between the same nodes may incur different costs because of different speed, maneuverability, or endurance characteristics. Second, the obstacle field may induce direction-dependent detours, making the motion cost asymmetric, so that in general

$$c_u(p,q)\neq c_u(q,p) \tag{7}$$

Third, task allocation and route optimization must evaluate a large number of candidate assignments and route modifications; invoking a full 3D path planner during each such evaluation would be computationally prohibitive. The offline matrix therefore compresses the geometric complexity of the environment into a form that can be reused efficiently in later stages without losing consistency with obstacle-aware motion feasibility.

For a fixed UAV u and node pair (p,q), the local planner searches for a collision-free path in the buffered free space Ω_f defined in the previous section. A candidate path is represented by a sequence of intermediate 3D waypoints connecting the start node x_p to the goal node x_q. Each candidate is evaluated by a multi-criteria objective that balances geometric efficiency, smoothness, safety, and energetic effort. In compact form, the local optimization problem is written as

$$\min_{\mathcal{P}_u(p,q)} J_{u,pq}=\alpha_L L+\alpha_S S+\alpha_R R+\alpha_E E+\alpha_H H \tag{8}$$

subject to collision avoidance and vehicle-feasibility constraints. Here, L denotes path length, S penalizes abrupt heading and climb changes, R quantifies proximity to buffered static obstacles, E is an energy-related effort term, and H regularizes the altitude profile. The coefficients $\alpha_L,\alpha_S,\alpha_R,\alpha_E,\alpha_H$ determine the relative priority of the individual criteria. This objective is designed so that the selected path is not merely short, but also operationally realistic: a slightly longer path may be preferred if it maintains safer obstacle clearance and avoids excessive turning or vertical maneuvering.

Among these terms, the obstacle-risk component is especially important because it enforces the static-environment geometry directly in the pairwise planning stage. Candidate paths intersecting buffered obstacles are treated as infeasible, while near-obstacle segments are penalized to discourage paths that pass too close to obstacle boundaries. This design ensures that the local planner searches for paths that are not only topologically feasible, but also conservative enough to support later smoothing without immediate loss of clearance. Likewise, the smoothness and energy terms reduce the likelihood that the local planner returns unnecessarily jagged or dynamically aggressive arcs that would be difficult to refine into executable trajectories in the final stage.

Once the best feasible path between p and q has been found for UAV u, its travel time is stored as the corresponding matrix entry,

$$c_u(p,q) = \tau^{\star}_{u,pq} \tag{9}$$

where $\tau^{\star}_{u,pq}$ is the predicted traversal time of the selected path under the UAV's nominal motion model. Travel time is used as the stored scalar cost because it is directly meaningful for endurance checking, route-duration evaluation, and makespan minimization in the multi-UAV stage. At the same time, the geometric path $\mathcal{P}_u(p,q)$ itself is retained, because it will later be needed to reconstruct the final selected routes before trajectory smoothing. If no feasible path exists for a particular ordered pair, the corresponding entry is assigned a prohibitive value, effectively marking that transition as forbidden in all subsequent assignment and routing operations.

The pairwise optimization is solved using a hybrid Differential Evolution and Enhanced Whale Optimization Algorithm. This combination is adopted because the 3D obstacle-avoidance problem induces a highly non-convex search space with multiple local minima, especially when several alternative corridors exist between the same node pair. In such settings, a single search strategy often struggles to provide both broad global exploration and sufficiently precise local refinement. The hybrid structure addresses this by delegating exploration and exploitation to complementary metaheuristics.

In the first phase, Differential Evolution performs global exploration of the waypoint search space. Each individual in the DE population encodes a candidate path via its intermediate waypoint coordinates. New candidate solutions are generated by mutation and crossover from existing population members. The essential mutation step is

$$v_i^{(g)} = x_{r_1}^{(g)} + F\left(x_{r_2}^{(g)} - x_{r_3}^{(g)}\right) \tag{10}$$

where $x_{r_1}^{(g)}, x_{r_2}^{(g)}, x_{r_3}^{(g)}$ are distinct candidate solutions at generation g, and F is the scaling factor. This mechanism perturbs candidate paths using population differences, allowing the search to probe new geometric regions of the feasible space rather than merely performing local perturbations around the current best solution. Trial candidates are then formed by crossover and evaluated under the objective $J_{u,pq}$, with greedy selection retaining the better solution. The role of DE here is to discover promising obstacle-avoidance corridors and to reduce the chance of premature convergence to an inferior route family.

Once the DE phase has identified a promising subset of candidate paths, the search transitions to an Enhanced Whale Optimization Algorithm for local exploitation. This second phase refines the geometry of the best DE-generated candidates and improves convergence near high-quality solutions. For a candidate $X(t)$ and the current best solution $X^*(t)$, the core encircling update is expressed as

$$D = |C \odot X^*(t) - X(t)|, X(t+1) = X^*(t) - A \odot D \tag{11}$$

where A and C are coefficient vectors that control contraction and displacement, and $\odot$ denotes elementwise multiplication. This mechanism drives candidates toward the most promising region already found, thereby improving exploitation. In the enhanced variant used here, three additional mechanisms are incorporated: boundary repair to keep waypoints inside the admissible search domain, quasi-oppositional learning to examine mirrored alternatives that may accelerate convergence, and adaptive diversification to reintroduce limited search variability when the algorithm stagnates. These enhancements are important in cluttered 3D environments, where naive local exploitation may otherwise settle too early into a path that is feasible but suboptimal.

The respective roles of the two phases can therefore be interpreted clearly. DE is responsible for identifying where a feasible and promising path corridor lies in the obstacle field, whereas E-WOA is responsible for improving how the path traverses that corridor. This division of labor is well aligned with the needs of static-environment UAV planning: robust global search is needed to avoid poor topological choices, while refined local search is needed to obtain reliable, low-cost arcs suitable for later mission-level optimization.

After all relevant ordered node pairs have been processed, the output for each UAV u consists of a complete directed matrix $C_u = [c_u(p,q)]$, a corresponding set of raw feasible paths $\mathcal{P}_u(p,q)$. Together, these encode which node transitions are admissible under the UAV's constraints. This offline database serves two distinct methodological functions in the remainder of the framework. First, the matrix C_u is used as the sole inter-node travel model during multi-UAV task distribution and route optimization; thus, all fleet-level decisions remain grounded in obstacle-aware, UAV-specific motion feasibility rather than in simple Euclidean approximations. Second, the stored raw paths are used later to reconstruct the geometry of the selected final routes before smoothing is applied.

The static-environment assumption is fundamental to the validity of this approach. Because obstacle locations do not change during planning or execution, the pairwise feasibility and cost structure remains fixed, and the offline path database remains valid throughout the optimization process. This makes the precomputed cost matrix not merely a computational convenience, but a structurally essential element of the methodology. It provides the link between low-level continuous path planning and high-level multi-UAV combinatorial optimization, enabling the overall framework to remain both computationally tractable and physically consistent.

3.3. Multi-UAV task distribution and route optimization

Once the pairwise path-planning stage has been completed, fleet-level planning is performed exclusively on the planner-derived travel-cost matrices $\{C_u\}_{u=1}^{M}$. At this stage, the geometric obstacle field is no longer queried directly; instead, all assignment and routing decisions are made using the precomputed travel times $c_u(p, q)$, which already encode static-obstacle avoidance and UAV-specific motion characteristics. This separation allows continuous geometric complexity to be handled offline while fleet-level decisions are made on a reusable motion-aware graph. In effect, the complexity of continuous 3D path planning is transferred to an offline preprocessing stage, allowing the multi-UAV problem to be solved as a structured combinatorial optimization problem over mission nodes.

The multi-UAV stage has two tightly coupled objectives. First, tasks must be distributed among the UAVs in a way that respects basic feasibility and avoids severe workload imbalance. Second, for each assigned task set, an efficient depot-to-depot visit order must be constructed and then refined so that the fleet-level mission cost is minimized. Let $\mathcal{U} = 1, \dots, M$ denote the UAV set and $\mathcal{N} = 1, \dots, N$ the task set. The output of this stage is a collection of disjoint task subsets $\{S_u\}_{u=1}^{M}$ and corresponding route orders $\{\pi_u\}_{u=1}^{M}$, such that each task is serviced exactly once by exactly one UAV.

A direct joint optimization of task assignment and route sequencing is computationally expensive, especially when the number of UAVs and tasks is large. The present framework therefore separates the problem into two linked steps. The first step produces a balanced and spatially coherent initial assignment using a surrogate workload model. The second step optimizes each route on the exact planner-derived matrices and then refines the assignment through cost-aware task exchanges. This decomposition retains computational tractability while allowing the final solution to be evaluated on the true motion-aware cost model.

The first requirement in task allocation is basic assignment feasibility. Although definitive endurance feasibility depends on the full route and cannot be established before sequencing, obviously impossible task-UAV pairings can be excluded early. For this purpose, a binary feasibility mask is defined as

$$A_{u,i} = \begin{cases} 1, & \text{if } c_u(0, i) + s_i + c_u(i, 0) \le B_u \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

where B_u is the endurance budget of UAV u, s_i is the service time at task i, and node 0 denotes the common depot. This criterion checks whether task i can at least be executed as an isolated sortie by UAV u. While conservative, it is inexpensive and removes assignments that are clearly infeasible before the routing stage begins.

Task distribution is then guided by a surrogate workload model. Using exact route cost during the assignment stage would require solving a routing problem for every tentative partition, which is computationally impractical. Instead, each possible task-UAV pairing is assigned a surrogate weight

$$w_{ui} = s_i + \lambda\big(c_u(0, i) + c_u(i, 0)\big) \tag{13}$$

where $\lambda > 0$ controls the influence of depot-based travel burden relative to service time. This weight is deliberately constructed from the same matrix $c_u(\cdot,\cdot)$ used later in route optimization, ensuring consistency between assignment and realized mission cost. The surrogate load of UAV u is then $L_u = \sum_{i \in S_u} w_{ui}$.

The corresponding mean load is

$$\bar{L} = \frac{1}{M} \sum\nolimits_{u=1}^{M} L_u \tag{14}$$

and load imbalance is measured by the variance-like objective

$$f_{\text{bal}} = \sum\nolimits_{u=1}^{M} \left(L_u - \bar{L}\right)^2 \tag{15}$$

Minimizing f_{bal} promotes an equitable distribution of workload while still allowing the later routing stage to optimize sequence-dependent efficiency.

To preserve geographical coherence during assignment, the task distribution is embedded in a weighted spatial partition. Let $y_i \in R^2$ denote the horizontal projection of task i. Each UAV u is associated with a region anchor q_u, a capability scaling factor $\varsigma_u > 0$, and an additive offset ω_u. Since the UAVs share a common depot, the anchors are not treated as physical start locations but as movable region representatives. The assignment score of task i to UAV u is defined as

$$D_u(y_i) = \frac{y_i - q_u{}^2}{\varsigma_u} + \omega_u \tag{16}$$

Task i is assigned to the feasible UAV with minimum score,

$$u^\star(i) \in arg \min_{u:,A_{u,i}=1} D_u(y * i), i \in S * u^\star(i) \tag{17}$$

This rule has two advantages. First, it encourages spatially compact task clusters, which tends to improve subsequent routing efficiency. Second, it offers a simple continuous mechanism for balancing loads through the offsets ω_u. If a UAV becomes overloaded, its offset can be increased, shrinking its effective region and encouraging neighboring UAVs to absorb nearby tasks.

The offsets are updated iteratively according to the current surrogate loads,

$$\omega_u^{(t+1)} = \omega_u^{(t)} + \eta_t \left(L_u^{(t)} - \overline{L^{(t)}}\right), \eta_t = \frac{\eta_0}{1 + \gamma t} \tag{18}$$

where $\eta_0 > 0$ is the initial step size and $\gamma \geq 0$ is a damping factor. In this way, overloaded UAVs gradually shrink their effective regions, whereas underloaded UAVs expand theirs and absorb nearby tasks. After each update, the offsets are recentered to remove the redundant common shift. The task assignment is then recomputed, and the region anchors are updated to the centroids of the current clusters,

$$q_u^{(t+1)} = \frac{1}{\left|S_u^{(t+1)}\right|} \sum_{i \in S_u^{(t+1)}} y_i \tag{19}$$

whenever $S_u^{(t+1)} \neq \emptyset$. The balancing procedure stops when the assignments stabilize, the imbalance falls below a prescribed threshold, or the maximum number of iterations is reached.

Once the task subsets $\{S_u\}_{u=1}^{M}$ have been obtained, a depot-to-depot route is constructed for each UAV using its planner-derived matrix $c_u(\cdot,\cdot)$. Initial routes are generated by a cheapest-insertion strategy and then improved by local search operations such as relocation, swap, and directed 2-opt. Because the initial partition is based on surrogate loads rather than exact sequence-dependent costs, a final refinement step is performed at the fleet level.

This refinement allows tasks to move between UAVs if doing so improves the global mission objective. Candidate relocations and swaps are first screened using marginal route-cost changes. For example, if task x lies between predecessor p and successor q on the current route of UAV a, the approximate gain from removing it is

$$\Delta_a^-(x) = c_a(p, x) + c_a(x, q) - c_a(p, q) \tag{20}$$

Similarly, if task x is inserted into the route of UAV b, the best approximate insertion penalty is

$$\Delta_b^+(x) = \min_{(i,j) \in \pi_b} [c_b(i, x) + c_b(x, j) - c_b(i, j)] \tag{21}$$

Only promising moves are then re-evaluated exactly after local route reoptimization, and a move is accepted only if it preserves feasibility and yields a strict improvement in the fleet objective. To maintain efficiency and spatial coherence, exchanges are restricted to boundary tasks and neighboring UAVs. The refinement process terminates when no further improving move exists. The output of this stage is therefore a set of balanced task subsets and optimized route orders that define the discrete mission plan for the subsequent trajectory-smoothing stage.

3.4. Trajectory smoothing

After task allocation and route optimization have been completed, the discrete mission plan is converted into executable UAV trajectories. To preserve consistency with the routing stage, the duration of each smoothed segment is initialized from the corresponding matrix entry $c_u(p, q)$, and the final realized route duration is recomputed after smoothing to verify endurance feasibility. At this stage, smoothing is applied only to the node-to-node paths that belong to the final selected routes; no further changes are made to task assignments or visit order. This ordering is essential for methodological consistency: the combinatorial planning stage determines which arcs are flown, whereas the smoothing stage determines how those arcs are traversed.

For each UAV u, let $\pi_u = (0, i_1, \dots, i_{|S_u|}, 0)$ denote the final route obtained in the previous stage. The corresponding raw geometric route is reconstructed by concatenating the stored pairwise paths selected from the offline path database,

$$\mathcal{R}_u = \mathcal{P}_u(0, i_1) \oplus \mathcal{P}_u(i_1, i_2) \oplus \cdots \oplus \mathcal{P}_u(i_{|S_u|}, 0) \tag{22}$$

where $\oplus$denotes path concatenation with duplicate boundary nodes removed. Because each task location is a mandatory service point, smoothing is performed segment-wise between consecutive route nodes, which preserves exact task visitation and avoids distorting the mission sequence.

Each selected path segment is then approximated by a cubic B-spline,

$$r(\xi) = \sum_{j=0}^{m} \boldsymbol{d}_j N_{j,3}(\xi), \qquad 0 \le \xi \le 1 \tag{23}$$

where d_j are control points and $N_{j,3}$ are cubic basis functions defined on a nondecreasing knot vector. Cubic B-splines are used because they provide C^2 continuity, local shape control, and analytically tractable first and second derivatives, which are well suited to UAV motion. The initial control polygon is obtained from the raw path samples by chord-length parameterization and least-squares fitting.

The smoothed segment is computed by minimizing a compact objective that balances fidelity to the original path, geometric regularity, and feasibility,

$$J_{\text{sm}} = \eta_1 E_{\text{fit}} + \eta_2 E_{\kappa} + \eta_3 E_{\text{len}} + \eta_4 E_{\text{clr}} \tag{24}$$

where E_{fit} penalizes deviation from the raw segment, E_{κ} penalizes excessive curvature, E_{len} discourages unnecessary path elongation, and E_{clr} penalizes insufficient obstacle clearance. Curvature is evaluated from spline derivatives as

$$\kappa(\xi) = \frac{|r'(\xi) \times r''(\xi)|}{|r'(\xi)|^3} \tag{25}$$

This formulation keeps the smoothed trajectory close to the obstacle-safe corridor established by the original planner while removing sharp turns and local irregularities.

To ensure safety, every smoothed segment is checked against the buffered static obstacles used in the path-planning stage. If smoothing reduces clearance below the required margin, the corresponding control points are locally adjusted or the fitting weights are tightened until the segment remains inside the admissible corridor. In addition, the final trajectory must satisfy the UAV's speed, acceleration, and curvature limits; if necessary, a time-scaling step is applied after geometric smoothing so that the trajectory remains dynamically executable without altering the route geometry.

The final output of this stage is a set of continuous, collision-free, and dynamically feasible trajectories $\Gamma\ _{u=1}^{M}$ that preserve the route structure determined by the multi-UAV planner while improving smoothness and flight quality. In this way, trajectory smoothing acts as the final execution layer of the framework, transforming the discrete mission plan into physically realizable UAV motion.

4. Experiments and Results

To evaluate the proposed framework, experiments were organized according to its three computational stages: offline single-UAV pairwise path planning, fleet-level task distribution and route optimization, and post-route trajectory smoothing.

The pairwise path planner was tested in a static three-dimensional environment of size $1000 \times 1000 \times 1000$, populated by building-like prisms and cylindrical protected volumes. Candidate solutions were represented as waypoint sequences, and all compared algorithms optimized the same local cost function. The hybrid DE+E-WOA planner was compared against standard WOA, PSO, DE, SOS, and GWO. All methods used a population of 50 candidates and a maximum of 50 iterations.

The fleet-level planner was evaluated on heterogeneous common-depot benchmark instances with $(N, M) \in (60,6), (120,12), (240,24)$, where N is the number of tasks and M is the number of UAVs. Directed planner-derived travel-cost matrices were used throughout this stage. Four methods were compared: a cost-greedy assignment baseline, a load-balancing heuristic, a partition-only variant of the proposed method, and the full framework with exchange-based refinement. All reported runtimes for this stage exclude offline pairwise matrix generation, which is analyzed separately in the local-planning experiment.

The first question was whether the local planner could provide sufficiently high-quality pairwise arcs to justify the use of a precomputed travel-cost matrix in the fleet-level stage. Representative results are shown in Fig. 2. In these representative cases, the hybrid DE+E-WOA planner generated the most direct and visually smooth obstacle-avoiding paths, while maintaining a larger clearance from obstacle boundaries than the alternative methods. Competing metaheuristics tended either to produce longer detours or to pass more tightly around the obstacles, increasing either travel cost or risk exposure.

The convergence behavior of the same algorithms is illustrated in Fig. 3. The hybrid planner reduced the local objective sharply during the early iterations and reached a stable near-optimal regime well before exhausting the 50-iteration budget. By contrast, WOA, PSO, DE, SOS, and GWO either converged more slowly or stabilized at higher final cost levels. This behavior is consistent with the intended division of labor in the hybrid search strategy: DE provides broad exploration of alternative corridors in the obstacle field, while E-WOA refines the most promising candidates and improves local path geometry once a favorable basin has been located.

From the perspective of the full framework, these results are important for two reasons. First, better pairwise arc quality directly improves the realism of the stored cost $c_u(p,q)$, which is later used for task allocation, endurance screening, and route optimization. Second, since this stage is performed offline, the higher solution quality of the hybrid local planner is more important than marginal differences in online execution time. The results therefore support the use of DE+E-WOA as the local path-planning engine for matrix generation.

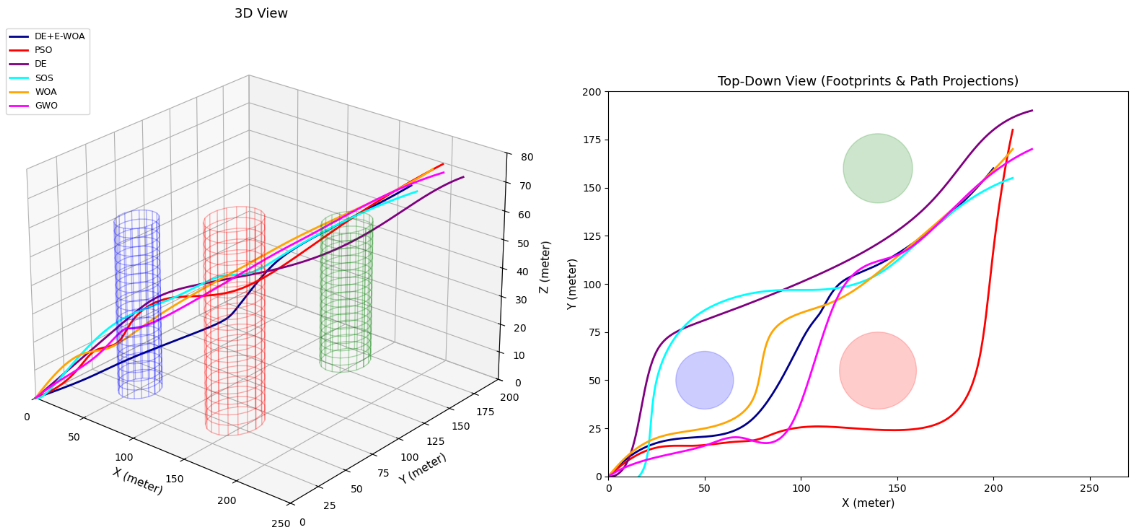

Fig. 2. Path Planning and Obstacle Avoidance Map

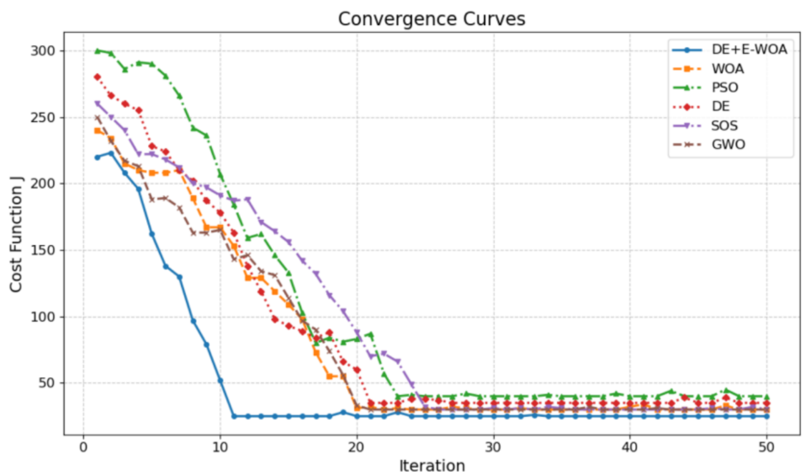

Fig. 3. Algorithms convergence curves

The second set of experiments examined how effectively the precomputed matrices support multi-UAV task allocation and route optimization. In Table 1, the compared methods are interpreted as follows: Cost-greedy assigns each task independently to the UAV with the smallest per-task surrogate transport cost; GreedyBalance emphasizes workload equality but ignores spatial structure; Partition-only denotes the balanced-partition method without inter-route exchange refinement; and Full framework denotes the complete assignment-and-routing pipeline.

Table 1. Performance summary (mean ± std over 5 instances; runtime is median seconds)

Instance (N,M)	Method	Makespan C_{max}	Total cost $\sum C_u$	Objective F	Imbalance CV	Runtime (s), median
(60,6)	Cost-greedy	48.33 ± 12.55	69.29 ± 1.19	98.29 ± 8.48	2.055 ± 1.256	0.05
(60,6)	GreedyBalance	18.14 ± 0.84	91.06 ± 2.65	101.95 ± 3.14	0.050 ± 0.020	0.05
(60,6)	Partition-only	12.95 ± 0.41	70.64 ± 2.35	78.41 ± 2.43	0.069 ± 0.023	0.22
(60,6)	Full framework	12.59 ± 0.49	69.73 ± 2.36	77.29 ± 2.61	0.099 ± 0.052	0.32
(120,12)	Cost-greedy	33.63 ± 6.21	128.72 ± 5.43	148.90 ± 9.15	1.531 ± 0.570	0.05
(120,12)	GreedyBalance	14.83 ± 0.65	163.10 ± 6.81	171.99 ± 7.20	0.046 ± 0.016	0.05
(120,12)	Partition-only	12.84 ± 0.41	128.60 ± 5.45	136.31 ± 5.92	0.071 ± 0.025	0.53
(120,12)	Full framework	12.41 ± 0.40	127.31 ± 5.34	134.76 ± 5.77	0.104 ± 0.056	0.93
(240,24)	Cost-greedy	19.32 ± 2.34	260.78 ± 8.81	272.37 ± 9.97	1.273 ± 0.312	0.05
(240,24)	GreedyBalance	12.47 ± 0.32	334.73 ± 6.42	342.21 ± 6.59	0.050 ± 0.012	0.05
(240,24)	Partition-only	11.76 ± 0.30	259.62 ± 8.60	266.68 ± 8.81	0.073 ± 0.023	1.56
(240,24)	Full framework	11.51 ± 0.31	257.44 ± 8.17	264.34 ± 8.38	0.099 ± 0.044	3.49

The results show a clear and consistent pattern across all three scales. The cost-greedy baseline achieved low total cost by construction, but at the price of severe imbalance and unacceptably large makespan. At $(60,6)$, its makespan reached 48.33, compared with 12.59 for the full framework. Even at the largest scale, the cost-greedy solution remained strongly unbalanced, with an imbalance coefficient of variation above 1.27.

GreedyBalance behaved oppositely: it equalized loads very effectively, but did so by ignoring geometry and vehicle-specific travel structure, which resulted in a large increase in total cost. At $(240,24)$, for example, total cost rose to 334.73, substantially above the 257.44 obtained by the full framework.

The proposed partition-only stage already delivered a strong balance-efficiency trade-off. Relative to the cost-greedy baseline, it reduced makespan dramatically while keeping total cost nearly unchanged. The exchange-refinement stage then yielded systematic additional gains. Across the three scales, the full framework reduced makespan by approximately 2.1–3.4% relative to partition-only and reduced the composite objective by approximately 0.9–1.4%, while increasing runtime only moderately. It should be noted that the compared baselines represent decomposition-style planners rather than fully integrated joint-optimization methods; stronger end-to-end baselines, such as those that simultaneously co-optimize assignment and routing under obstacle-aware costs, may narrow these margins and would constitute a valuable point of comparison in future work. These gains were achieved without global re-optimization of the full multi-route solution; instead, they arose from localized relocations and swaps evaluated directly on the planner-derived matrices.

This improvement is also evident qualitatively in Fig. 4, which illustrates the behavior of the two-stage routing framework on a representative instance. The initial balanced partition produces spatially coherent task clusters centered around the common depot, thereby yielding an interpretable and operationally sensible starting solution. After exchange-based refinement, only a limited number of boundary tasks are reassigned, yet the resulting tours become more compact and better aligned with the fleet-level objective. Importantly, the refinement does not destroy the structure established by the partitioning stage; instead, it introduces only targeted adjustments in regions where exact matrix-based evaluation reveals a measurable gain in routing efficiency or makespan reduction. This behavior confirms that the proposed refinement stage acts as a corrective mechanism for residual surrogate mismatch rather than as a wholesale reassignment procedure.

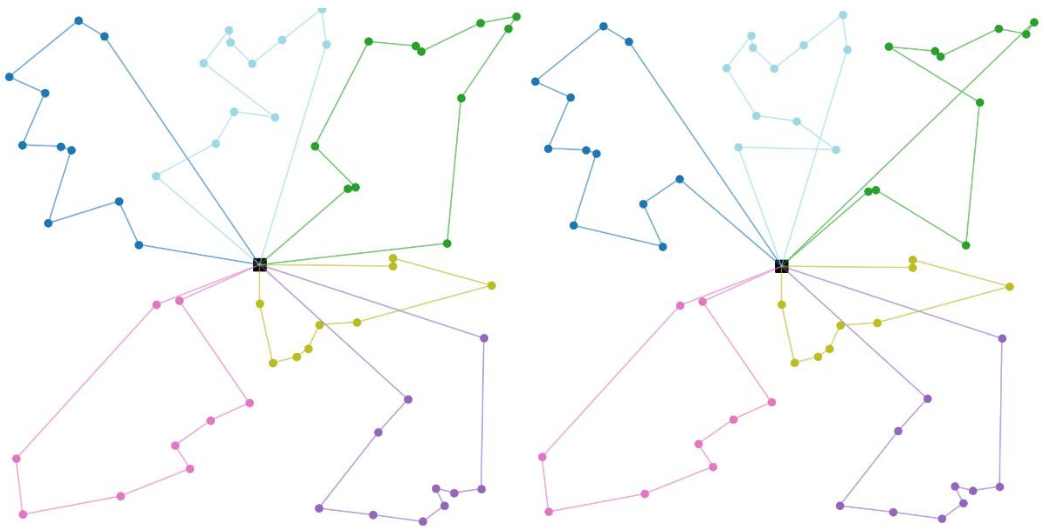

Fig. 4. Qualitative behavior on a representative instance

A complementary view is provided by the schematic representation in Fig. 5, which clarifies the task-allocation and depot-based tour abstraction underlying the experiments. Tasks distributed in three-dimensional space are assigned to specific UAVs and then connected through depot-to-depot tours, making explicit how the planner transforms a spatial mission into a set of coordinated vehicle routes. When considered together with the quantitative results in Table 1, the figure highlights the principal advantage of the proposed framework: it preserves the operational simplicity of route-based mission decomposition while incorporating UAV-specific travel costs derived from obstacle-aware local planning. Consequently, the method achieves a favorable balance between computational tractability, route interpretability, and mission-level performance, which is particularly important as the problem scale increases from small to large fleet-task instances.

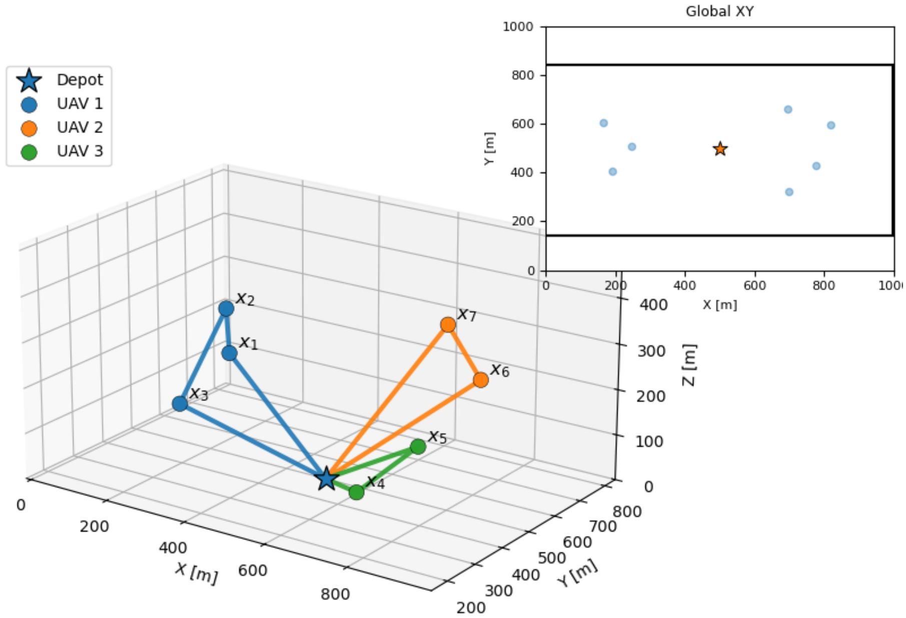

Fig. 5. Schematic illustration of UAV task allocation and depot-to-depot tour abstraction

The final set of experiments evaluated the post-route smoothing module. Since the route planner operates on raw pairwise paths extracted from the offline database, smoothing is the final step that converts these piecewise paths into executable trajectories with improved curvature behavior and flight efficiency. Four alternatives were compared: the unsmoothed raw polyline, a uniform cubic B-spline, a fixed-knot cubic B-spline, and the proposed adaptive cubic B-spline. Results are reported in Table 2.

Table 2. Performance of Path-Smoothing Methods.

Method	Path Length (m)	Max Curvature (m^{-1})	Computation Time (s)	Straightness Ratio
Raw polyline (No Smoothing)	2984.6	0.65	0	1.107
Uniform cubic B-spline	2840.8	0.56	0.25	1.053
Fixed-knot cubic	2803.2	0.49	0.31	1.039
Adaptive B-spline	2731.4	0.38	0.42	1.013

The smoothing results show that post-route refinement is not merely cosmetic. Relative to the raw polyline, the adaptive cubic B-spline reduced path length from 2984.6 m to 2731.4 m, a reduction of approximately 8.5%, while reducing maximum curvature from 0.65 to 0.38 m^{-1}, a reduction of approximately 41.5%. The straightness ratio improved from 1.107 to 1.013, indicating that the adaptive spline remained much closer to the direct start-to-goal line while still respecting obstacle constraints. These gains were obtained with a modest computation time of 0.42 s. Compared with the fixed-knot cubic B-spline, the adaptive spline also provided a meaningful improvement: path length decreased by approximately 2.6% and maximum curvature by approximately 22.4%. This indicates that adaptive knot placement contributes materially to local geometric quality even when the spline degree is held constant.

Taken together, the experiments validate the three components of the proposed framework and, more importantly, the logic of their integration. The local DE+E-WOA planner produces high-quality obstacle-aware arcs and a reliable travel-time matrix for higher-level planning. The fleet-level assignment and route optimizer exploits these matrices to obtain near-balanced multi-UAV solutions with low makespan and near-minimal total cost. The post-route adaptive B-spline smoother then transforms the selected raw arcs into dynamically preferable trajectories without altering the discrete mission plan.

5. Conclusion

This paper presented a hierarchical framework for multi-UAV mission planning in static three-dimensional environments that integrates obstacle-aware local path generation, fleet-level task allocation and route optimization, and post-route trajectory smoothing within a single consistent pipeline. The framework was motivated by a persistent methodological gap in the literature: low-level path feasibility is often treated independently from high-level mission optimization, and final smoothing is commonly applied without explicit linkage to the geometric information that guided task allocation and routing. By constructing planner-derived directed travel-cost matrices from single-UAV path planning and using those matrices as the sole motion model in the fleet-level stage, the proposed approach establishes a direct connection between continuous obstacle-aware motion and discrete multi-UAV mission planning.

The results support the effectiveness of this design. At the local level, the hybrid DE+E-WOA planner produced high-quality pairwise paths and showed stronger convergence behavior than the compared standalone metaheuristics, thereby justifying its use as the engine for offline cost-matrix generation. At the fleet level, the balanced partitioning stage already provided a strong compromise between workload equity and travel efficiency, and the exchange-refinement stage produced consistent, though moderate, additional gains. Relative to the partition-only solution, the full framework reduced makespan by 2.1–3.4% and the composite objective by 0.9–1.4% across benchmark instances ranging from $(60,6)$ to $(240,24)$ while preserving modest online runtimes. In the final smoothing stage, adaptive cubic B-splines substantially improved trajectory quality: compared with the raw polyline representation, path length was reduced by 8.5%, maximum curvature by 41.5%, and straightness improved to near-direct-route behavior. These improvements confirm that smoothing is not simply a cosmetic operation, but a meaningful execution-level enhancement of the mission plan.

The principal significance of the framework lies in its separation of roles. The local planner handles geometric complexity and obstacle avoidance offline; the fleet-level optimizer reasons over a reusable motion-aware graph; and the smoothing layer transforms the selected route structure into executable UAV motion. This decomposition makes the problem more tractable without sacrificing physical consistency within the scope of static, fully known environments evaluated here. At the same time, the study has clear boundaries that must be emphasised. The current formulation assumes a fully static environment, complete and accurate prior map knowledge, and a common-depot mission structure — conditions that may not hold in many operational deployments. All validation was performed on benchmark instances rather than physical hardware, and no real-flight or hardware-in-the-loop experiments were conducted. In addition, the pairwise path database is generated offline, which is appropriate for known environments but incompatible with highly dynamic or partially unknown settings without modification.

Future work should therefore extend the framework in several directions. The most important is relaxation of the static-environment assumption through incremental matrix updates or hybrid online replanning strategies. Additional avenues include explicit treatment of state-estimation uncertainty, tighter aerodynamic or energy models, heterogeneous depots or launch points, and hardware-in-the-loop or field validation on real UAV platforms. Even within its present scope, however, the proposed framework demonstrates that multi-UAV planning in known static environments can be addressed effectively by combining motion-aware pairwise path planning, structured fleet-level optimization, and adaptive post-route trajectory smoothing in a unified methodological architecture.

All the Declarations and Statements

Author Contributions Statement
Both authors made substantial contributions.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement
The authors declare no conflict of interest

Funding Declaration
This research received no external funding.

Data Availability Statement
N/A

Ethical Declaration
N/A

Acknowledgments

N/A

Declaration of generative AI in scholarly writing

Generative AI tools were used to assist in improving the clarity, grammar, and readability of the manuscript. The authors take full responsibility for the content of the work.

Abbreviations

The following abbreviations are used in this manuscript:

DE - Differential Evolution
E-WOA - Enhanced Whale Optimization Algorithm
GWO - Grey Wolf Optimization
PSO - Particle Swarm Optimization
SOS - Symbiotic Organisms Search
UAV - Unmanned Aerial Vehicle

Appendix A/B/C…, with appendix title

N/A

References

[1] D. Debnath, F. Vanegas, J. Sandino, A. F. Hawary, and F. Gonzalez, “A Review of UAV Path-Planning Algorithms and Obstacle Avoidance Methods for Remote Sensing Applications,” Remote Sensing, vol. 16, no. 21, p. 4019, Oct. 2024, doi: 10.3390/rs16214019.

[2] Y. Yang, X. Xiong, and Y. Yan, “UAV Formation Trajectory Planning Algorithms: A Review,” Drones, vol. 7, no. 1, p. 62, Jan. 2023, doi: 10.3390/drones7010062.

[3] S. Das and P. Roy, “Path planning of unmanned aerial systems in a static 3D environment using firefly algorithm F1-3 with B-spline,” Transactions of the Institute of Measurement and Control, Jul. 2025, doi: 10.1177/01423312251355291.

[4] K. Wu, J. Lan, S. Lu, C. Wu, B. Liu, and Z. Lu, “Integrative Path Planning for Multi-Rotor Logistics UAVs Considering UAV Dynamics, Energy Efficiency, and Obstacle Avoidance,” Drones, vol. 9, no. 2, p. 93, Jan. 2025, doi: 10.3390/drones9020093.

[5] G. M. Skaltsis, H.-S. Shin, and A. Tsourdos, “A Review of Task Allocation Methods for UAVs,” J. Intell. Robot. Syst., vol. 109, no. 4, Nov. 2023, doi: 10.1007/s10846-023-02011-0.

[6] K. A. R. Grøntved, U. P. S. Lundquist, and A. L. Christensen, “Decentralized Multi-UAV Trajectory Task Allocation in Search and Rescue Applications,” 2023 21st International Conference on Advanced Robotics (ICAR). IEEE, pp. 35–41, Dec. 5, 2023. doi: 10.1109/icar58858.2023.10406912.

[7] X. Zhan, Y. Chen, X. Chen, and W. Zhang, “Balanced Multi-UAV path planning for persistent monitoring,” Robotica, vol. 43, no. 1, pp. 332–349, Nov. 2024, doi: 10.1017/s0263574724001899.

[8] B.-M. Jeong, Y.-S. Oh, D.-S. Jang, N.-E. Hwang, J.-W. Kim, and H.-L. Choi, “Makespan-Minimizing Heterogeneous Task Allocation under Temporal Constraints,” Aerospace, vol. 10, no. 12, p. 1032, Dec. 2023, doi: 10.3390/aerospace10121032.

[9] H. Li, L. Liao, X. Dai, Y. Feng, R. Feng, and S. Tang, “Balancing Efficiency and Fairness: An Iterative Exchange Framework for Multi-UAV Cooperative Path Planning,” 2025 International Conference on New Trends in Computational Intelligence (NTCI). IEEE, pp. 49–53, Oct. 17, 2025. doi: 10.1109/ntci67886.2025.11308568.

[10] M. Peng and W. Meng, “Cooperative Obstacle Avoidance for Multiple UAVs Using Spline_VO Method,” Sensors, vol. 22, no. 5, p. 1947, Mar. 2022, doi: 10.3390/s22051947.

[11] W. Sun, P. Sun, W. Ding, J. Zhao, and Y. Li, “Gradient-based autonomous obstacle avoidance trajectory planning for B-spline UAVs,” Sci. Rep., vol. 14, no. 1, Jun. 2024, doi: 10.1038/s41598-024-65463-w.

[12] S. Poudel and S. Moh, “Task assignment algorithms for unmanned aerial vehicle networks: A comprehensive survey,” Vehicular Communications, vol. 35, p. 100469, Jun. 2022, doi: 10.1016/j.vehcom.2022.100469.

[13] J. Song, K. Zhao, and Y. Liu, “Survey on Mission Planning of Multiple Unmanned Aerial Vehicles,” Aerospace, vol. 10, no. 3, p. 208, Feb. 2023, doi: 10.3390/aerospace10030208.

[14] H. Zheng and J. Yuan, “An Integrated Mission Planning Framework for Sensor Allocation and Path Planning of Heterogeneous Multi-UAV Systems,” Sensors, vol. 21, no. 10, p. 3557, May 2021, doi: 10.3390/s21103557.

[15] S. Gao, J. Wu, and J. Ai, “Multi-UAV reconnaissance task allocation for heterogeneous targets using grouping ant colony optimization algorithm,” Soft Comput., vol. 25, no. 10, pp. 7155–7167, Feb. 2021, doi: 10.1007/s00500-021-05675-8.

[16] S. Poudel and S. Moh, “Priority-aware task assignment and path planning for efficient and load-balanced multi-UAV operation,” Vehicular Communications, vol. 42, p. 100633, Aug. 2023, doi: 10.1016/j.vehcom.2023.100633.

[17] F. Yan, J. Chu, J. Hu, and X. Zhu, “Cooperative task allocation with simultaneous arrival and resource constraint for multi-UAV using a genetic algorithm,” Expert Systems with Applications, vol. 245, p. 123023, Jul. 2024, doi: 10.1016/j.eswa.2023.123023.

[18] J. Li, X. Yang, Y. Yang, and X. Liu, “Cooperative mapping task assignment of heterogeneous multi-UAV using an improved genetic algorithm,” Knowledge-Based Systems, vol. 296, p. 111830, Jul. 2024, doi: 10.1016/j.knosys.2024.111830.

[19] T. Ma, P. Lu, F. Deng, and K. Geng, “Air–Ground Collaborative Multi-Target Detection Task Assignment and Path Planning Optimization,” Drones, vol. 8, no. 3, p. 110, Mar. 2024, doi: 10.3390/drones8030110.

[20] B. Zhang, K. Huang, Y. Chen, and D. Yang, “Task Allocation and Trajectory Optimization for Multi-UAV Cargo Systems with Cellular-Connected Constraints,” IET Communications, vol. 19, no. 1, Jan. 2025, doi: 10.1049/cmu2.70106.
[21] H. Huang, Z. Jiang, T. Yan, and Y. Bai, “Dynamic Task Allocation for Heterogeneous Multi-UAVs in Uncertain Environments Based on 4DI-GWO Algorithm,” Drones, vol. 8, no. 6, p. 236, Jun. 2024, doi: 10.3390/drones8060236.
[22] C. Sun, Y. Yao, and E. Zheng, “Enhancing Unmanned Aerial Vehicle Task Assignment with the Adaptive Sampling-Based Task Rationality Review Algorithm,” Drones, vol. 8, no. 9, p. 422, Aug. 2024, doi: 10.3390/drones8090422.
[23] Z. Han and W. Guo, “Dynamic UAV Task Allocation and Path Planning with Energy Management Using Adaptive PSO in Rolling Horizon Framework,” Applied Sciences, vol. 15, no. 8, p. 4220, Apr. 2025, doi: 10.3390/app15084220.
[24] M. Zhang, Y. Han, S. Chen, M. Liu, Z. He, and N. Pan, “A Multi-Strategy Improved Differential Evolution algorithm for UAV 3D trajectory planning in complex mountainous environments,” Engineering Applications of Artificial Intelligence, vol. 125, p. 106672, Oct. 2023, doi: 10.1016/j.engappai.2023.106672.
[25] K. Wang, Z. Meng, Z. Wang, and Z. Wu, “A Trajectory Generation Method for Multi-Rotor UAV Based on Adaptive Adjustment Strategy,” Applied Sciences, vol. 13, no. 6, p. 3435, Mar. 2023, doi: 10.3390/app13063435.
[26] M. Liu, H. Zhang, J. Yang, T. Zhang, C. Zhang, and L. Bo, “A path planning algorithm for three-dimensional collision avoidance based on potential field and B-spline boundary curve,” Aerospace Science and Technology, vol. 144, p. 108763, Jan. 2024, doi: 10.1016/j.ast.2023.108763.
[27] H. Wang, Z. Hao, and Y. Zhang, “Research on three-dimensional path planning of unmanned aerial vehicle based on improved Whale Optimization Algorithm,” PLoS ONE, vol. 20, no. 2, p. e0316836, Feb. 2025, doi: 10.1371/journal.pone.0316836.

Author’s Profile

Mykola Nikolaiev: PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute” Kyiv, 03056, Ukraine.
Areas of scientific interests: Distributed systems, optimization methods, algorithms, machine learning.

Mykhailo Novotarskyi: Doctor of Sciences, Professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute” Kyiv, 03056, Ukraine.
Areas of scientific interests: Mathematical modeling of complex physical processes, formal description of mathematical and simulation models, parallel locally asynchronous numerical methods, machine learning and data mining, reinforcement learning methods.

How to cite this paper

Mykola Nikolaiev, Mykhailo Novotarskyi, "A Multi-UAV Planning Framework for Task Allocation, Route Optimization and Trajectory Smoothing", International Journal of Wireless and Microwave Technologies(IJWMT), Vol.16, No.4, pp. 1-15, 2026. DOI:10.5815/ijwmt.2026.04.01