

# Robust and Efficient Trajectory Planning for Formation Flight in Dense Environments

Lun Quan\*, Longji Yin\*, Tingrui Zhang, Mingyang Wang,  
Ruilin Wang, Sheng Zhong, Xin Zhou, Yanjun Cao, Chao Xu, and Fei Gao

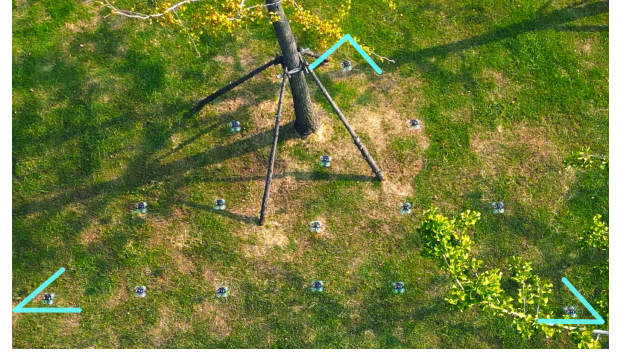
**Abstract**—Formation flight has a vast potential for aerial robot swarms in various applications. However, existing methods lack the capability to achieve fully autonomous large-scale formation flight in dense environments. To bridge the gap, we present a complete formation flight system that effectively integrates real-world constraints into aerial formation navigation. This paper proposes a differentiable graph-based metric to quantify the overall similarity error between formations. This metric is invariant to rotation, translation, and scaling, providing more freedom for formation coordination. We design a distributed trajectory optimization framework that considers formation similarity, obstacle avoidance, and dynamic feasibility. The optimization is decoupled to make large-scale formation flights computationally feasible. To improve the elasticity of formation navigation in highly constrained scenes, we present a swarm reorganization method that adaptively adjusts the formation parameters and task assignments by generating local navigation goals. A novel swarm agreement strategy called global-remap-local-replan and a formation-level path planner is proposed in this work to coordinate the global planning and local trajectory optimizations. To validate the proposed method, we design comprehensive benchmarks and simulations with other cutting-edge works in terms of adaptability, predictability, elasticity, resilience, and efficiency. Finally, integrated with palm-sized swarm platforms with onboard computers and sensors, the proposed method demonstrates its efficiency and robustness by achieving the largest scale formation flight in dense outdoor environments.

**Index Terms**—Aerial swarms, formation flight, obstacle avoidance, motion planning, distributed trajectory optimization.

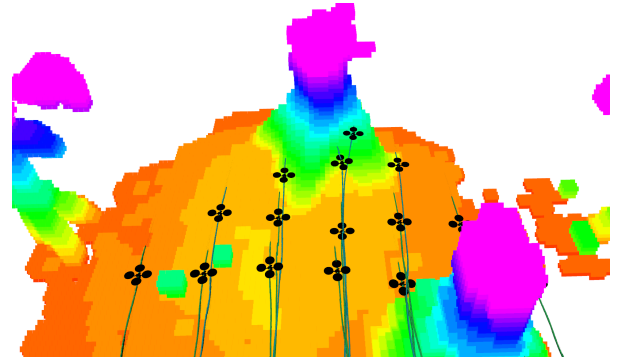
## I. INTRODUCTION

**F**ORMATION flight has become a fundamental capability for autonomous swarms to achieve coordinated aerial maneuvers. In cluttered wilds and complex urban areas, formation navigation has a wide potential in search and rescue [1], collaborative mapping [2], package delivery [3], and so on. However, effectively integrating real-world constraints into an aerial formation remains an unresolved problem. This article aims to empower aerial swarms to maintain cooperative formation behaviors in dense environments by proposing a complete formation flight system.

Inspired by natural swarm systems like bird flocks and fish schools, an ideal formation flight system should possess the capability to flexibly adapt and deform in dense environments. By striving to maintain the swarm in a “critical



(a) Snapshot of 16 quadrotors navigating in a triangular queue shape



(b) Rviz diagram of the formation flight

Fig. 1. Large-scale formation flight in the dense outdoor environment. (a) Snapshot of the moment the swarm robots prepare to fly through the woods. (b) Rviz diagram of the executed trajectories. The grid map is merged using log data offline. Please watch our attached videos for more information about the experiments at <https://www.youtube.com/watch?v=uEMyvPxYqmA>.

state”, swarm formation can dynamically balance the conflicts between maintaining formation and avoiding obstacles.

While extensive research works focus on navigation in formation, few achieve robust formation flights in obstacle-rich areas. Three core challenges limit practical formation applications: (a) The inherent conflict between formation maintenance and obstacle avoidance is inevitable and difficult to mitigate. (b) Predefined formations lack elastic adaptability in response to constrained environments. (c) The swarm system cannot rapidly recover from disordered states caused by unknown obstacles or sudden changes in the desired formation shape.

Based on the above challenges, we conclude that an ideal formation flight system should have the ability to maintain formation while avoiding obstacles, adjust swarm formation distributions according to constrained environments, and reorganize the formation quickly after emergencies. These characteristics are summarized as the *PAPER* criteria:

- **Portability:** Aerial robotic swarms should comprise

\*Indicates equal contribution.

Corresponding author: Fei Gao, Chao Xu, and Yanjun Cao.

This work was supported by the National Natural Science Foundation of China under grant no. 62003299 and 62088101. All authors are from the State Key Laboratory of Industrial Control Technology, Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, China, and the Huzhou Institute, Zhejiang University, Huzhou 313000, China {lunquan, fgaoaa, cxu, yanjunhi}@zju.edu.cn, ljiyin6038@163.com.

lightweight platforms with scalable systems and distributed architecture. A scalable system means the main components, such as estimation, decision, planning, and control modules, are all the same on each robot. A distributed architecture is inherently robust against individual hardware failures. These are the basis for large-scale formation flight.

- **Adaptability:** When facing obstacles, robots should locally adapt their trajectories to avoid collision in a way that does the least damage to the overall formation performance. This ability mitigates the conflict between formation maintenance and obstacle avoidance.
- **Predictability:** Reactive local feedback methods are short-sighted and can not consider the constraints in advance. Robots should optimize the motions over a prediction horizon so that the formation can respond smoothly to the future environmental changes in its vicinity, which is necessary for dense areas.
- **Elasticity:** A feasible and safe trajectory for a fixed formation shape may not exist in constrained environments, such as narrow corridors or holes. Therefore, swarm robots need to have elastic and flexible deformation capabilities by adjusting formation distributions (such as the scale of shape or task assignments) while keeping the full maneuverability of the formation.
- **Resilience:** Formation flight could encounter many unfavorable situations caused by unknown obstacles or sudden changes of desired formation shape. The navigation system should be able to resiliently reorganize and guide the whole formation so that the flight can recover from disordered states timely.

A complete formation flight system should meet the above *PAPER* criteria and ensure that the conditions for each criterion are compatible with the others.

Our previous work [4] only partially met the first three terms of *PAPER* criteria. We tackled formation flight as a coupled collaborative trajectory optimization problem, suitable primarily for small-scale formation scenarios. However, resolving cooperative constraints of the formation using the graph-based similarity metric was computationally heavy, resulting in increased overhead during each optimization iteration. Moreover, integrating dynamic inter-robot relationships within the coupled trajectory optimization problem considerably affected the efficiency of the optimization process, rendering it less appropriate for larger formations or more complex scenarios.

In this paper, we present a complete formation flight system that satisfies all *PAPER* criteria. To address the challenges in [4], we introduce a decoupled formation optimization method to significantly improve computational efficiency. This method consists of two components. Firstly, an optimal formation position sequence is pre-computed, avoiding repetitive metric calculation during the optimization process. Secondly, a fixed time interval sampling method is used to convert dynamic inter-robot relationships into static constraints, greatly reducing the complexity of the optimization problem. These improvements make our method suitable for large-scale swarms. Besides, the previous method lacks the capability of reorganizing the swarm formation, which may lead to disordered

formation flights under adverse conditions, especially when the initial positions or task assignments are inappropriate. To address this, we propose a swarm reorganization method that can elastically adjust formation distributions by optimizing formation parameters and task assignments in response to external constraints. Subsequently, we develop a swarm agreement strategy called global-remap-local-replan, which enables rapid implementation of the swarm reorganization results to achieve consensus among swarm agents. Additionally, a formation-level global path-finding method, which treats the swarm formation as one entity, is also designed to guide the swarm out of the obstacle deadlocks. Finally, we integrate the estimation, mapping, decision, planning, and control modules into palm-sized swarm platforms [5] with onboard computers and sensors, enabling large-scale formation flight in dense environments. Detailed contributions are as follows.

- 1) We introduce an optimal formation position sequence, pre-computed using the differentiable graph-based metric [4]. This sequence represents the optimal position with the lowest similarity error, reducing the need for repetitive computation during the optimization process.
- 2) We design a decoupled spatial-temporal trajectory optimization framework that effectively handles dynamic inter-robot relationships, obstacle avoidance, and dynamic feasibility. Compared to our prior work [4], we achieve higher computational efficiency for large-scale swarms.
- 3) We present a swarm reorganization method to achieve elastic deformation of swarm distributions, which simultaneously solves optimal formation alignment and task assignment problems (ALAS for short). This method improves the elasticity of swarm formation against constrained environments. It relieves the dependence on the appropriate formation alignments and task assignments.
- 4) We design a global-remap-local-replan strategy (GRLR for short) that leverages the advantages of centralized formation parameter remapping and decentralized local trajectory replanning. With this strategy, the distributed asynchronous swarm is able to quickly recover from disordered states and return to formation flight quickly.
- 5) We integrate all these modules into a hierarchical formation flight system. Extensive benchmarks and simulations are conducted to validate the *PAPER* criteria of our method. A series of real-world experiments are designed to demonstrate the outstanding performance of the proposed distributed autonomous formation flight system.

## II. RELATED WORKS

### A. Distributed Swarm Trajectory Planning

Extensive works exist for trajectory planning of distributed swarms. The concept of velocity obstacle (VO) is leveraged and generalized by Van Den Berg et al. [6]–[8] to accomplish reciprocal collision avoidance for multiple robots. However, the smoothness of the resulting trajectories cannot be guaranteed by VO-based approaches, which significantly impairs the usability of the actual robot systems.

In order to produce high-quality collision-free trajectories, optimization-based methods are widely introduced in the literature on distributed multicopter swarms [9]–[11]. Zhou et

al. [12] incorporate Voronoi cell tessellation into a receding horizon QP scheme to prevent collision among the robots while planning. In [13], Chen et al. employ SCP to address the multiagent planning problem in non-convex space by incrementally tightening the collision constraints. Baca et al. [14] combine MPC with a conflict resolution strategy to ensure mutual collision avoidance for outdoor swarm operations. Nevertheless, the computational load of the above optimization-based methods is large, which could hamper the applicability of the planners in highly dense scenarios.

Recently, Zhou et al. [5] present a distributed autonomous quadrotor swarm system using spatial-temporal trajectory optimization, which generates collision-free motions in dense environments merely in milliseconds. Our distributed formation trajectory optimization is based on this work.

#### B. Formation Flight in Free Space

Various techniques have been proposed to achieve multi-robot navigation in formation, which include virtual structures [15], leader-follower [16], navigation functions [17], reactive behaviors [18], consensus-based local control laws [19], and barycentric-coordinate-based control [20]. However, most of the existing methods only consider obstacle-free cases.

Weinstein et al. [21] present a VIO-swarm system that performs all modules onboard and can execute formation flight without inter-robot collisions in free space. Parker et al. [22] present a distributed formation control method and relax the dependency of the common reference frame.

As the scale of swarms increases, researchers begin to notice that it is difficult to maintain the formation only by trajectory planning, especially when there are deadlocks between robots. Turpin et al. [23] consider the problem of concurrent assignment and collision-free trajectory generation. Turpin gives centralized and decentralized solutions to this problem, allowing flight formation on a large scale. Morgan et al. [24] also use model predictive control to solve task assignment and trajectory generation simultaneously when given the desired formation shape. In addition to considering task assignments, Agarwal and Akella [25] consider formation alignment problems to optimize the formation parameters such as scale and location. This method reduces the cost of forming formation and speeds up convergence. However, these methods ignore the influence of constrained environments, in which formation should elastically deform to navigate.

#### C. Formation Flight in constrained Environments

In constrained environments, where various obstacles and limitations exist, formation flight can be a challenging task that requires constant adjustments to maintain the swarm structure. An immediate solution is to design composite control laws that combine formation flight and collision avoidance by using multiple layered potential fields [26], which are prone to deadlock. A better solution is to allow the formation shape to deform while maintaining the overall swarm structure. Han et al. [27] propose a complex-valued graph Laplacian-based formation controller that regulates the scaling of formation shape during swarm maneuvering like passing through corridors. In [28], Zhao proposes a leader-follower control law enabling the affine transformation of formation in response to environmental changes. And the bearing-based local controller [29],

[30] exhibits translational, scaling, and rotational invariance of formation flight. However, these methods rely on leaders or predefined trajectories and struggle with complex obstacles or sudden potential collisions.

Compared to the local feedback methods, predictive optimization-based methods proactively plan the future motion of swarm robots, striking a balance between formation flight and obstacle avoidance. Alonso-Mora et al. [31] control swarm robots by optimally rearranging the desired formation and planning local trajectories for each drone. However, since there is no inter-vehicle coordination in the distributed planners, formation maintenance is not conducted during local planning. Peng et al. [32] propose a method to improve flight safety by enabling the affine transformation of formation shape and treating it as a soft constraint during B-spline optimization. However, this approach requires optimizing the trajectories of all robots simultaneously and cannot be applied to large-scale swarms. To tackle formation preservation, Parys et al. [33] propose a distributed model predictive formation controller. This framework imposes relative position constraints on the swarm and coordinates the agents to break passively once obstacles violate positional constraints. Overall, these approaches offer unique solutions for trajectory planning in swarm robotics, but they each have limitations when dealing with different scenarios and scales of robotic systems.

To address these drawbacks, we formulate the overall formation requirement with a differentiable metric in trajectory optimization. This allows us to fully utilize the collaboration ability of the swarm, effectively avoid deadlock, and foresee obstacle avoidance. Besides, we adopt a distributed and decoupled optimization method to ensure dynamic real-time performance. This approach can be applied to large-scale swarms while still maintaining efficient trajectory planning.

### III. SYSTEM OVERVIEW

This paper aims to optimize the autonomy of swarm robots in real-world environments by coordinating their movements to form a desired formation shape. To accomplish this, we adopt a distributed swarm aerial robot system and propose a spatial-temporal trajectory optimization for formation flight. To enhance the system's robustness, we also address the case of swarm disorder by incorporating an adaptive swarm reorganization method and an efficient swarm agreement strategy.

#### A. Swarm Aerial Robot System

The swarm aerial robot system is composed of palm-sized quadrotor platforms [5] with depth stereo camera<sup>1</sup> for imagery and depth sensing, as shown in Fig. 2. The software modules, including state estimation, environment perception, decision-making, trajectory planning, and flight control, run in real-time on an onboard computer<sup>2</sup>. This lightweight and scalable platform suit dense environments.

We use visual-inertial odometry (VIO) [34] to estimate each robot's pose with respect to its start frame, and we recover the transformations related to the start frames with

<sup>1</sup><https://www.intelrealsense.com/depth-camera-d435/>

<sup>2</sup><https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-xavier-nx/>

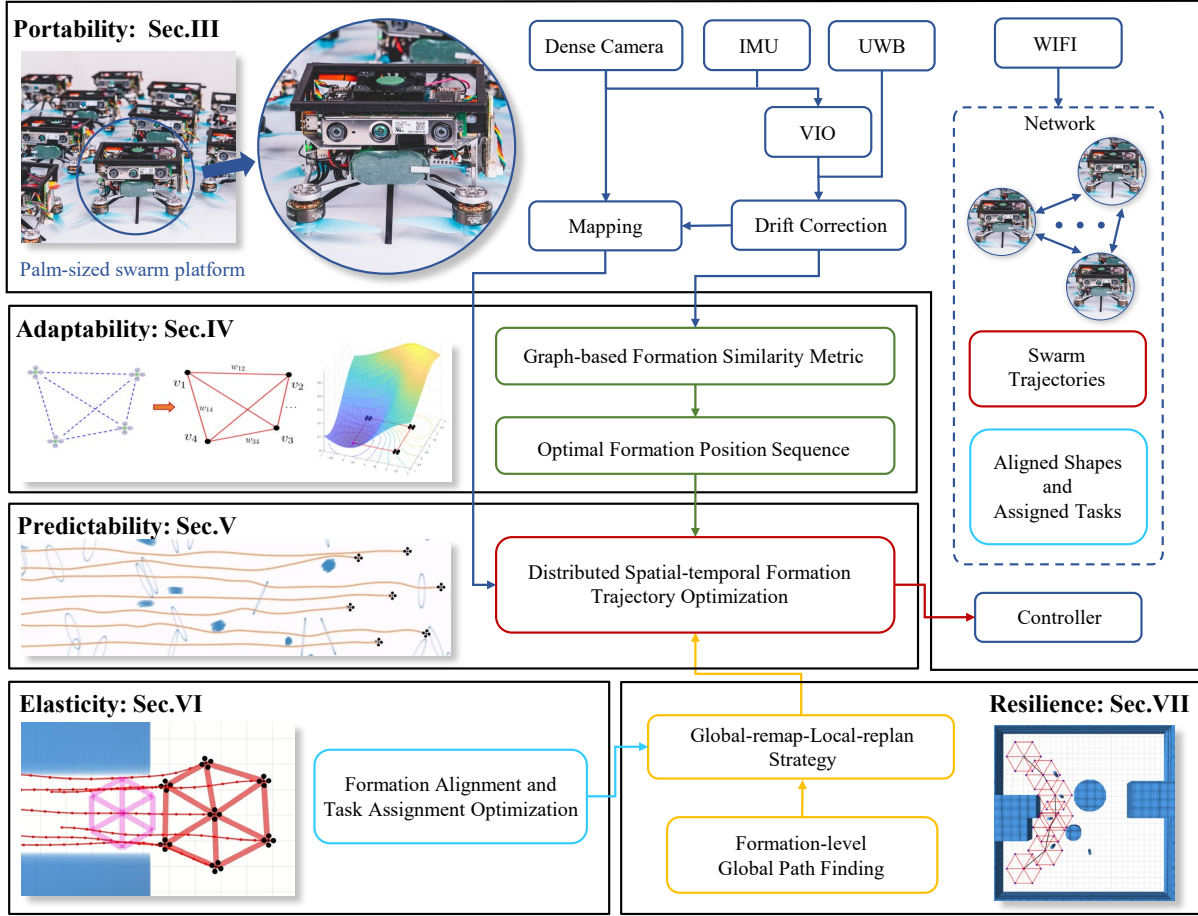


Fig. 2. Illustration of the system architecture. In order to facilitate understanding, we divide the various modules of the formation flight system according to the *PAPER* criteria mentioned in Sec.I. The main challenge in integrating *PAPER* criteria into a swarm formation flight system is to ensure that the conditions for each criterion are compatible with the others. Therefore we build a hierarchical formation flight system. We use different colors to represent different levels in the system. Moreover, the information broadcast through the network is from the same color module.

only anonymous bearing measurements in our previous work [35]. For simplicity, the transformations are known in our experiments by requiring robots to take off from pre-defined locations. To correct the localization drift between swarm robots, we use a drift correction method [5] with onboard ultra-wideband (UWB).

The distributed system architecture enables each robot to fully utilize its computing resources to process more information, relieving the pressure of network communication. The robots share only important information, such as trajectories, for high-fidelity wireless communication, and there is no ground station to send control inputs.

#### B. Distributed Local Formation Trajectory Optimization

The local formation trajectory optimization is distributed and asynchronous, which enables each robot to generate its trajectory only depending on local information and does not require the same start timestamp or same time duration of trajectory. Each robot evaluates the formation state by calculating the formation similarity error and generates its optimal formation position sequence (Sec.IV) to maintain the desired formation shape. Then a subsequent trajectory optimization module generates spatial-temporal formation trajectories (Sec.V) for real-time navigation. The above process cycles periodically within the receding horizon. The distributed local

formation trajectory optimization can always maintain the overall formation when navigating in a complex environment.

#### C. Swarm Reorganization and Agreement Methods

Only relying on the distributed local trajectory optimization may lead to poor formation flight quality when encountering narrow corridors or instant transformation of formation shapes. Therefore, we propose swarm reorganization (Sec.VI) and agreement methods (Sec.VII) to achieve the swarm consensus quickly. Inspired by the flock flying behavior of birds, we design a global-remap-local-replan (GRLR for short) strategy, which only centrally remaps crucial parameters of swarm formation and makes the swarm formation converge quickly through distributed replanning local trajectory. Firstly, when the stable state of the swarm formation is destroyed or about to be destroyed, the swarm robots designate a leader (drone 0 in this paper). The crucial parameters of swarm formation are calculated separately by formation alignment and task assignment optimization (ALAS for short) and formation-level global pathfinding method. From the perspective of swarm reorganization and agreement, the conflict between swarm formation and obstacle is alleviated by optimizing formation alignment, and the speed of formation convergence is greatly accelerated by optimizing task assignment. GRLR strategy is simple but very effective by combining distributed methods' efficiency and centralized ones' optimality.



#### IV. ADAPTIVE DESCRIPTION OF SWARM FORMATION

##### A. Graph-based Formation Definition

In this paper, a swarm formation of  $N$  robots is modeled by an undirected graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V} := \{1, 2, \dots, N\}$  is the set of vertices, and  $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$  is the set of edges. In graph  $\mathcal{G}$ , the vertex  $i$  represents the  $i^{th}$  robot with position vector  $\mathbf{p}_i = [x_i, y_i, z_i] \in \mathbb{R}^3$ . An edge  $e_{ij} \in \mathcal{E}$  that connects vertex  $i \in \mathcal{V}$  and vertex  $j \in \mathcal{V}$  means that robot  $i$  and  $j$  can measure the geometric distance between each other. In our work, each robot can obtain the positions of all robots  $\{\mathbf{p}_1, \dots, \mathbf{p}_i, \dots, \mathbf{p}_N\}$ , thus the graph  $\mathcal{G}$  is complete. Then we determine the adjacency matrix  $\mathbf{A} \in \mathbb{R}^{N \times N}$  and degree matrix  $\mathbf{D} \in \mathbb{R}^{N \times N}$  of the formation graph  $\mathcal{G}$  by

$$A_{ij} = w_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\|^2, \quad (1)$$

$$D_{ij} = \begin{cases} \sum_{j=1}^N A_{ij}, & \text{if } i = j, \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

where the non-negative edge weight  $w_{ij}$  is the squared distance between the  $i^{th}$  and  $j^{th}$  robots, and  $\|\cdot\|$  denotes the Euclidean norm. Thus, the corresponding Laplacian matrix is

$$\mathbf{L} = \mathbf{D} - \mathbf{A}. \quad (3)$$

With the above matrices, the symmetric normalized Laplacian matrix of graph  $\mathcal{G}$  is defined as

$$\hat{\mathbf{L}} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}, \quad (4)$$

where  $\mathbf{I} \in \mathbb{R}^{N \times N}$  is the identity matrix.  $\hat{\mathbf{L}}$  contains the information that is invariant to scale, translation, and rotation.

Finally, we use graph theory to describe various desired formation shapes, such as squares, hexagons, and pyramids. By specifying the positions  $\mathbf{p}_i^d = [x_i^d, y_i^d, z_i^d] \in \mathbb{R}^3$ ,  $i = 1, \dots, N$ , computing  $\hat{\mathbf{L}}_{des}$  is simple. It's important to note that the desired formation shape is independent of the coordinate system as long as the relative positions are provided.

##### B. Differentiable Formation Similarity Error Metric

To assess the deviation from the desired formation, we propose a differentiable formation similarity error metric as

$$f_s = f_s(\mathbf{p}_1, \dots, \mathbf{p}_i, \dots, \mathbf{p}_N) = f_s(\mathbf{A}, \mathbf{D}) = f_s(\hat{\mathbf{L}}, \hat{\mathbf{L}}_{des}) \\ = \|\hat{\mathbf{L}} - \hat{\mathbf{L}}_{des}\|_F^2 = \text{tr}\{(\hat{\mathbf{L}} - \hat{\mathbf{L}}_{des})^T (\hat{\mathbf{L}} - \hat{\mathbf{L}}_{des})\}, \quad (5)$$

where  $\text{tr}\{\cdot\}$  denotes the trace of a matrix,  $\hat{\mathbf{L}}$  is the symmetric normalized Laplacian of the current swarm formation,  $\hat{\mathbf{L}}_{des}$  is the counterpart of the desired formation. Frobenius norm  $\|\cdot\|_F$  is used in our distance metric. As a graph representation matrix,  $\hat{\mathbf{L}}$  contains information about the graph structure [36]. This allows  $f_s$  to consider only the geometric shape of the formation, and not be influenced by scaling, translation, or rotation. Additionally,  $f_s$  is a dimensionless value that solely reflects the error in formation shape similarity.

In particular, under the distributed framework, each robot can only change its positions to reduce the overall formation similarity error. Therefore, the only variable for robot  $i$  in (5) is  $\mathbf{p}_i$ , and  $f_s(\mathbf{p}_1, \dots, \mathbf{p}_i, \dots, \mathbf{p}_N)$  can be simplified as  $f_s(\mathbf{p}_i)$ .

Our metric is analytically differentiable with respect to the position of each robot. For robot  $i$ , we use the weights of its

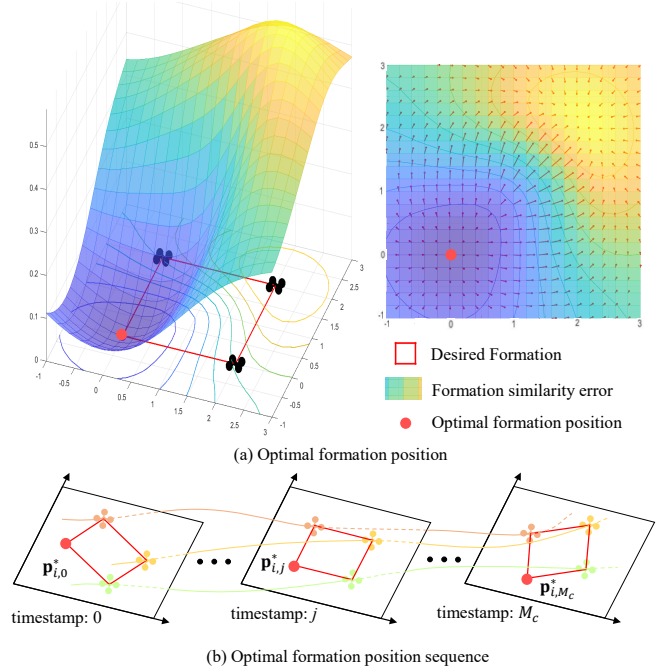


Fig. 3. Illustration of optimal formation position sequence using a 2D formation. (a) The surface shows the profile of the similarity metric when one UAV moves in the plane and the other three remain still. The minimum suggests the optimal formation position to form the desired shape. (b) The sequence of optimal formation positions corresponds to the timestamps.

$N$  adjacent edges  $\{e_{i1}, e_{i2}, \dots, e_{iN}\}$  to form a weight vector  $\mathbf{w}_i = [w_{i1}, w_{i2}, \dots, w_{iN}]^T$ . By the chain rule, the gradient of  $f_s$  with respect to  $\mathbf{p}_i$  can be written as

$$\frac{\partial f_s}{\partial \mathbf{p}_i} = \frac{\partial f_s}{\partial \mathbf{w}_i^T} \frac{\partial \mathbf{w}_i}{\partial \mathbf{p}_i}. \quad (6)$$

According to our metric (5), the gradient of  $f_s$  with respect to each weight  $w_{ij}$  can be computed as follow

$$\frac{\partial f_s}{\partial w_{ij}} = \text{tr}\left\{\left(\frac{\partial f_s}{\partial \hat{\mathbf{L}}}\right)^T \left(\frac{\partial \hat{\mathbf{L}}}{\partial w_{ij}}\right)\right\}, \\ \frac{\partial f_s}{\partial \hat{\mathbf{L}}} = \frac{\partial \|\hat{\mathbf{L}} - \hat{\mathbf{L}}_{des}\|_F^2}{\partial \hat{\mathbf{L}}} = 2(\hat{\mathbf{L}} - \hat{\mathbf{L}}_{des}), \\ \frac{\partial \hat{\mathbf{L}}}{\partial w_{ij}} = -\frac{\partial (\mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2})}{\partial w_{ij}}. \quad (7)$$

Then the gradient  $\partial f_s / \partial \mathbf{w}_i$  can be written as

$$\partial f_s / \partial \mathbf{w}_i = [\partial f_s / \partial w_{i1}, \partial f_s / \partial w_{i2}, \dots, \partial f_s / \partial w_{iN}]^T. \quad (8)$$

As for  $\partial \mathbf{w}_i / \partial \mathbf{p}_i$ , the Jacobian can be easily derived since the weight function (1) is a differentiable quadratic form.

##### C. Optimal Formation Position Sequence

In our previous work [4], we incorporated  $f_s$  directly into the trajectory optimization, making formation flight a coupled trajectory optimization problem. While this method is suitable for small-scale formation flight, it becomes computationally inefficient as the number  $N$  of robots increases. Considering the simplified equation for coupled trajectory optimization

$$\min_{\mathbf{p}_{i,0}, \dots, \mathbf{p}_{i,M_c}} \sum_{j=0}^{M_c} f_s(\mathbf{p}_{i,j}) + J_{other}, \quad (9)$$

where  $\mathbf{p}_{i,j}$  represent the  $j^{th}$  sample point of  $i^{th}$  robot trajectory in (19) for convenience.  $J_{other}$  represents all other cost functions, and  $M_c$  is the number of sample points with corresponding timestamps. The primary purpose of calculating  $f_s$  is to supply gradient information for minimizing formation similarity error. However, since the graph  $\mathcal{G}$  is a complete graph, computing  $f_s$  has a complexity of  $O(N^2)$ . Consequently, the coupled trajectory optimization (9) also exhibits high complexity of  $O(N^2)$  in each iteration, limiting its applicability to large-scale swarm operations.

To address this issue, we must identify an equivalent approach with reduced computational complexity to replace the function of  $f_s$  in (9). We introduce the concept of **optimal formation position**  $\mathbf{p}_{i,j}^*$  for robot  $i$  at timestamp  $j$ , which is the position that minimizes the formation similarity error  $f_s$ . Fig. 3(a) illustrates this concept using a 2D formation as an example. It is evident from the figure that there exists an optimal formation position  $\mathbf{p}_{i,j}^*$  that results in a minimal formation similarity error, and the partial derivative is  $\partial f_s / \partial \mathbf{p}_{i,j} = 0$ . In the future period with a sequence of timestamps  $\{0, \dots, j, \dots, M_c\}$ , we represent the expected positions of robot  $i$  with the **optimal formation position sequence**  $\mathbf{p}_i^* = \{\mathbf{p}_{i,0}^*, \dots, \mathbf{p}_{i,j}^*, \dots, \mathbf{p}_{i,M_c}^*\}$ , as shown in Fig. 3(b). By precomputing  $\mathbf{p}_i^*$ , we can utilize its quadratic distance to replace the gradient information offered by  $f_s$  in (11), thus decreasing the computational requirements as follows

$$f_s(\mathbf{p}_{i,j}) \Rightarrow \|\mathbf{p}_{i,j} - \mathbf{p}_{i,j}^*\|^2. \quad (10)$$

Since the optimal solutions of  $f_s$  and quadratic distance cost are equivalent, the trajectory approaches the positions with minimal formation similarity error, maintaining the desired formation. Thus, we can effectively solve the coupled trajectory optimization with a two-step procedure

$$\begin{aligned} \textcircled{1} \quad \mathbf{p}_i^* &= \arg \min \sum_{j=0}^{M_c} f_s(\mathbf{p}_{i,j}), \\ \xrightarrow{\mathbf{p}_i^*} \textcircled{2} \quad \min_{\mathbf{p}_{i,0}, \dots, \mathbf{p}_{i,M_c}} &\|\mathbf{p}_{i,j} - \mathbf{p}_{i,j}^*\|^2 + J_{other}. \end{aligned} \quad (11)$$

As a result, the previously required calculation of  $f_s$  in each trajectory optimization process is replaced by the computation of the quadratic distance, simplifying the optimization problem. This significantly reduces computational demands and enables large-scale swarm formation.

Formula (11) indicates that trajectory optimization in Sec.V is performed on discretized points. Non-uniform discretized points may lead to poor trajectories and sub-optimal performance. Therefore it is crucial to ensure a uniform distribution of these points to maintain the effectiveness of the optimization process. In engineering practice, since graphs  $\mathcal{G}$  are constructed from a series of discretized timestamps as depicted in Fig. 3(b), each  $\mathbf{p}_{i,j}^*$  is independent.

To ensure a smoother trajectory, we introduce the uniform optimal formation position sequence  $\hat{\mathbf{p}}_i^*$ , which is generated by considering the formation similarity error  $J_s$  and the uniform distribution cost  $J_u$

$$\hat{\mathbf{p}}_i^* = \arg \min \lambda_s J_s + \lambda_u J_u, \quad (12)$$

$$\begin{aligned} J_s &= \sum_{j=0}^{M_c} f_s(\hat{\mathbf{p}}_{i,j}^*), \\ J_u &= \mathbb{E}(\mathbf{U}^2) - \mathbb{E}(\mathbf{U})^2 = \frac{\|\mathbf{U}\|_2^2}{M_c} - \frac{\|\mathbf{U}\|_1^2}{(M_c)^2}, \end{aligned} \quad (13)$$

where  $\lambda_s$  and  $\lambda_u$  are the relative weights.  $\mathbb{E}(\cdot)$  is mathematic expectation and the squared distance vector  $\mathbf{U} \in \mathbb{R}^{M_c}$  is

$$\mathbf{U} = (\|\hat{\mathbf{p}}_{i,1}^* - \hat{\mathbf{p}}_{i,0}^*\|_2^2, \dots, \|\hat{\mathbf{p}}_{i,M_c}^* - \hat{\mathbf{p}}_{i,M_c-1}^*\|_2^2). \quad (14)$$

We use the quasi-Newton method [37] to solve this unconstrained optimization problem (12) and generate uniform  $\hat{\mathbf{p}}_i^*$  for the later trajectory optimization (18). By doing so, the trajectory resulting from these discretized points in Sec.V can be smoother and avoid sudden spatial changes.

## V. SPATIAL-TEMPORAL TRAJECTORY OPTIMIZATION FOR FORMATION FLIGHT

### A. Trajectory Representation

The differential flatness of multicopters [38] benefits trajectory generation without integrating differential equations. Moreover, the motion planning of multicopters can be performed on low-dimensional smooth trajectories. In this paper, we adopt a state-of-the-art trajectory representation named MINCO [39] to achieve minimum control effort spatial-temporal trajectory planning for swarm aerial robots in three-dimensional environments. MINCO conducts spatial-temporal deformation of the flat-output  $M$ -piece trajectory  $p(t)$  by decoupling the space and time parameters with a linear-complexity mapping  $\mathcal{M}$

$$p(t) = \mathcal{M}_{\mathbf{q}, \mathbf{T}}(t), \quad \forall t \in [t_0, t_M], \quad (15)$$

where  $\mathbf{q} = (\mathbf{q}_1, \dots, \mathbf{q}_{M-1})^T \in \mathbb{R}^{3 \times (M-1)}$  are the adjacent intermediate points between each pair of connected pieces and  $\mathbf{T} = (T_1, \dots, T_M)^T \in \mathbb{R}_{>0}^M$  the time duration of each piece.

A  $m$ -dimensional  $M$ -piece trajectory  $p(t)$  is represented by piecewise polynomials. And  $i^{th}$  piece  $p_i(t)$  is defined as a multi-degree polynomial ( $Q = 5$  in this paper)

$$p_i(t) = \mathbf{c}_i^T \boldsymbol{\beta}(t), \quad \forall t \in [0, T_i], \quad (16)$$

where  $\mathbf{c}_i \in \mathbb{R}^{(Q+1) \times m}$  is the coefficient matrix and  $\boldsymbol{\beta}(t) = [t^0, t^1, \dots, t^Q]^T$  is the natural basis.

For an  $s$ -integrator ( $s = 3$  in this paper) chain dynamics system, a  $M$ -piece  $2s - 1$  degree trajectory  $p(t)$  is defined by constant boundaries and minimum control effort  $\{\mathbf{q}, \mathbf{T}\}$ . Furthermore, MINCO is advanced in convert  $\{\mathbf{q}, \mathbf{T}\}$  to  $\{\mathbf{c}, \mathbf{T}\}$  using a linear-time and space parameter mapping  $\mathbf{c} = \mathcal{M}(\mathbf{q}, \mathbf{T})$ , where  $\mathbf{c} = (\mathbf{c}_1^T, \dots, \mathbf{c}_M^T)^T$  is polynomial coefficients.

### B. Problem Formulation

After determining the desired formation shape in Sec.IV, we expect a cluster of trajectories for swarm robots, which are smooth, collision-free, and formation maintained. In practice, navigating swarm robots in an unknown dense environment with FOV-limited sensors and onboard computer requires an efficient real-time planner focusing on local information. Besides, centralized optimization is limited by the scale of

the swarm. Therefore, we choose a distributed local trajectory optimization for formation flight as follows

$$\min_{\mathbf{q}, \mathbf{T}} \int_{t_0}^{t_M} \|p^{(s)}(t)\|^2 dt + \rho \cdot T_\Sigma, \quad (17a)$$

$$s.t. \quad p(t) = \mathcal{M}_{\mathbf{q}, \mathbf{T}}(t), \forall t \in [t_0, t_M], \quad (17b)$$

$$\mathbf{p}^{[s-1]}(0) = \bar{\mathbf{p}}_0, \quad (17c)$$

$$\mathbf{p}^{[s-1]}(t_M) = \bar{\mathbf{p}}_f, \quad (17d)$$

$$\mathcal{H}(p(t), \dots, p^{(s)}(t)) \preceq \mathbf{0}, \forall t \in [t_0, t_M]. \quad (17e)$$

We define costs (17a) for smoothness and aggressiveness to achieve smooth and efficient flight.  $\rho$  is time regularization parameter,  $T_\Sigma = \sum_{i=1}^M T_i$ . The state of robot  $p(t)$  (17b) is parameterized by the optimization variables  $\{\mathbf{q}, \mathbf{T}\}$ .  $\mathbf{p}^{[s-1]}(t) = (p(t)^T, \dot{p}(t)^T, \dots, p^{(s-1)}(t)^T)^T \in \mathbb{R}^{ms}$  represents the higher-order derivatives of a chain dynamic system with  $s$ -integrator. Boundary conditions involve initial state  $\bar{\mathbf{p}}_0 \in \mathbb{R}^{ms}$  (17c) and terminal state  $\bar{\mathbf{p}}_f \in \mathbb{R}^{ms}$  (17d). Continuous-time constraints  $\mathcal{H}$  (17e) include swarm formation similarity, dynamic feasibility, obstacle avoidance, and swarm reciprocal avoidance.

### C. Constraints Transcription

To solve the continuous constrained optimization problem (17) in real-time, we use the optimization variable of MINCO (15) to eliminate all kinds of equality constraints (17b)-(17d). And penalty function method [40] is used to deal with the inequality constraints (17e). Then, every integral is evaluated by a finite sum of sample points. Finally, the continuous constrained optimization problem is converted to a discrete unconstrained optimization problem

$$\min_{\mathbf{q}, \mathbf{T}} \sum_x \lambda_\star \tilde{J}_\star(\mathbf{q}, \mathbf{T}, \delta), \quad (18)$$

where  $\tilde{J}_\star$  are various terms of cost function or penalties, and  $\lambda_\star$  are relative weights. Subscripts  $\star = \{f, e, t, o, r, d\}$  ( $f$ ) swarm formation similarity, ( $e$ ) denote control effort, ( $t$ ) total time, ( $o$ ) obstacle avoidance, ( $r$ ) swarm reciprocal avoidance, ( $d$ ) dynamic feasibility.  $\delta$  is the sampling time interval.

In our previous work [4], we used the fixed number sampling points  $\hat{\mathbf{p}}_{i,j} = p_i((j/\kappa_i) \cdot T_i)$  to transform the optimization problem, where  $p_i(t)$  is the  $i^{th}$  piece trajectory and  $\kappa_i$  is the fixed sample number on this piece. However, considering that the total time  $T_\Sigma$  changes during the optimization process, the fixed number sampling points  $\hat{\mathbf{p}}_{i,j}$  are difficult to space on the whole trajectory equally. Therefore, we take fixed time interval sampling points for the whole trajectory to ensure the accuracy of the penalty function sampling transformation

$$\tilde{\mathbf{p}}_j(t) = p_i(j\delta - \sum_{l=1}^{i-1} T_l), \quad (19)$$

$$j \in \{0, \dots, \kappa\}, \kappa = \lfloor \frac{T_\Sigma}{\delta} \rfloor,$$

where  $\kappa$  is the sample number and  $T_l$  is the preceding time for any  $1 \leq l < i$ .

For the trajectory planning of swarm robots, the fixed time interval sampling points  $\tilde{\mathbf{p}}_j(t)$  can simplify the optimization problem. Compared with  $\hat{\mathbf{p}}_{i,j}$ , the timestamp corresponding to  $\tilde{\mathbf{p}}_j(t)$  is fixed, so the states of other robots at this timestamp

are also constant during the optimization process. Therefore, it is feasible to calculate the states of other robots w.r.t  $\tilde{\mathbf{p}}_j(t)$  according to the broadcast trajectories before optimization. Then we can solve the uniform formation position sequence optimization (12) in advance and use  $\hat{\mathbf{p}}_i^*$  to replace the formation similarity metric  $f_s$  in trajectory optimization (17a) of  $i^{th}$  robot. This decoupled formation trajectory optimization results in higher computational efficiency, making our method suitable for large-scale swarm robots.

Despite the optimization problem is not differentiable when sampling number  $\kappa$  changes, the cost function remains continuous w.r.t. time duration  $\mathbf{T}$ . In this paper, we use the quasi-Newton method proposed in [37] to solve the non-smooth discrete unconstrained optimization problem (18).

### D. Cost Functions and Gradients

Given the fixed sampling time interval  $\delta$ , we can evaluate the cost functions and gradients of the whole trajectory by a finite sum of sampling points  $\tilde{\mathbf{p}}_j(t)$ . The cost of various general purpose penalties at  $j^{th}$  sampling points is

$$\mathcal{P}_\star(\mathbf{c}, \mathbf{T}, j\delta) = \mathcal{P}_\star(\tilde{\mathbf{p}}_j(t)), \quad (20)$$

then the cost function  $\tilde{J}_\star$  in (18) is calculated as follows

$$\begin{aligned} \tilde{J}_\star(\mathbf{q}, \mathbf{T}, \delta) &= J_\star(\mathbf{c}, \mathbf{T}, \delta), \\ &= \delta \sum_{j=0}^{\kappa} \bar{\omega}_j \mathcal{P}_\star(\mathbf{c}, \mathbf{T}, j\delta) + \\ &\quad \frac{1}{2} (T_\Sigma - \kappa\delta) [\mathcal{P}_\star(\mathbf{c}, \mathbf{T}, \kappa\delta) + \mathcal{P}_\star(\mathbf{c}, \mathbf{T}, T_\Sigma)], \end{aligned} \quad (21)$$

where  $(\bar{\omega}_0, \bar{\omega}_1, \dots, \bar{\omega}_{\kappa-1}, \bar{\omega}_\kappa) = (1/2, 1, \dots, 1, 1/2)$  are the orthogonal coefficients following the trapezoidal rule [41]. And MINCO allows any second-order continuous cost function  $\tilde{J}_\star(\mathbf{q}, \mathbf{T})$  to be represented by  $J_\star(\mathbf{c}, \mathbf{T})$ . Hence,  $\partial \tilde{J}_\star / \partial \mathbf{q}$  and  $\partial \tilde{J}_\star / \partial \mathbf{T}$  can be efficiently obtained from  $\partial J_\star / \partial \mathbf{c}$  and  $\partial J_\star / \partial \mathbf{T}$  respectively, which is benefit to the construction and solution of the optimization problem. In (19), the sampling time  $t = j\delta - \sum_{l=1}^{i-1} T_l$  is related to the preceding time  $T_l$ , so the gradient of  $J_\star$  w.r.t  $\mathbf{c}_i$  and  $T_l$  are computed as

$$\frac{\partial J_\star}{\partial \mathbf{c}_i} = \frac{\partial J_\star}{\partial \mathcal{P}_\star} \frac{\partial \mathcal{P}_\star}{\partial \tilde{\mathbf{p}}_j(t)} \frac{\partial \tilde{\mathbf{p}}_j(t)}{\partial \mathbf{c}_i}, \quad (22)$$

$$\frac{\partial J_\star}{\partial T_l} = \frac{\partial J_\star}{\partial \mathcal{P}_\star} \frac{\partial \mathcal{P}_\star}{\partial \tilde{\mathbf{p}}_j(t)} \frac{\partial \tilde{\mathbf{p}}_j(t)}{\partial t} \frac{\partial t}{\partial T_l}, \quad (23)$$

$$\frac{\partial \tilde{\mathbf{p}}_j(t)}{\partial \mathbf{c}_i} = \beta(t), \frac{\partial \tilde{\mathbf{p}}_j(t)}{\partial t} = \dot{\tilde{\mathbf{p}}}_j(t), \frac{\partial t}{\partial T_l} = \begin{cases} 0, & l = i, \\ -1, & l < i, \end{cases} \quad (24)$$

where the calculation of  $\partial J_\star / \partial \mathcal{P}_\star$  is simple and the details of  $\mathcal{P}_\star(\tilde{\mathbf{p}}_j(t))$  for various general purpose are given as follow.

1) *Cost of Swarm Formation Similarity  $\mathcal{P}_f$* : In Sec.IV-C, we decouple the formation similarity error metric from trajectory optimization by constructing an unconstrained optimization problem to calculate the uniform optimal formation position sequence  $\hat{\mathbf{p}}_i^*$  for each sampling point. This improvement avoids multiple calculations of formation similarity metric  $f_s$ . Then, we use the quadratic form to calculate the cost of swarm formation similarity

$$\mathcal{P}_f(\tilde{\mathbf{p}}_j(t)) = \max\{\|\tilde{\mathbf{p}}_j(t) - \hat{\mathbf{p}}_{i,j}^*\|^2, 0\}^3. \quad (25)$$

2) *Control Effort  $J_e$* : The  $s^{th}$  ( $s = 3$  in this paper) control input for the trajectory and its gradients are written as

$$J_e = \sum_{i=1}^M \int_0^{T_i} \|p_i^{(s)}(t)\|^2 dt, \quad (26)$$

$$\frac{\partial J_e}{\partial \mathbf{c}_i} = 2 \left( \int_0^{T_i} \beta^{(s)}(t) \beta^{(s)}(t)^T dt \right) \mathbf{c}_i, \quad (27)$$

$$\frac{\partial J_e}{\partial T_i} = \mathbf{c}_i^T \beta^{(s)}(T_i) \beta^{(s)}(T_i)^T \mathbf{c}_i. \quad (28)$$

3) *Total Time  $J_t$* : In order to ensure the aggressiveness of the trajectory, we minimize the total time  $J_t = \sum_{i=1}^M T_i$ . The gradients are given by  $\partial J_t / \partial \mathbf{c} = \mathbf{0}$  and  $\partial J_t / \partial \mathbf{T} = \mathbf{1}$ .

4) *Cost of Obstacle Avoidance  $\mathcal{P}_o$* : Inspired by [42], obstacle avoidance penalty  $J_o$  is computed using Euclidean Signed Distance Field (ESDF). We penalize the sampling points which are too close to the obstacles

$$\mathcal{P}_o(\tilde{\mathbf{p}}_j(t)) = \max\{\psi_o(\tilde{\mathbf{p}}_j(t)), 0\}^3, \quad (29)$$

$$\psi_o(\tilde{\mathbf{p}}_j(t)) = d_o - d_o(\tilde{\mathbf{p}}_j(t)), \quad (30)$$

where  $d_o$  is the safety threshold set according to the actual situation and  $d_o(\tilde{\mathbf{p}}_j(t))$  is the distance between  $\tilde{\mathbf{p}}_j(t)$  and the closest obstacle around it. The gradient of  $\mathcal{P}_o$  w.r.t  $\tilde{\mathbf{p}}_j(t)$  is

$$\frac{\partial \mathcal{P}_o}{\partial \tilde{\mathbf{p}}_j(t)} = -\nabla d^T, \quad (31)$$

where the  $\nabla d$  is the gradient of ESDF in  $\tilde{\mathbf{p}}_j(t)$ .

5) *Cost of Swarm Reciprocal Avoidance  $\mathcal{P}_r$* : We penalize  $\tilde{\mathbf{p}}_j(t)$  when it is too close to the trajectories  $p_\phi(t)$ ,  $\phi \in \Phi$  at the fixed timestamp  $t = j\delta$ , where  $\Phi$  represents the all other robots in the swarm. Compared to our previous work [4], the state of other robots with fixed timestamp  $p_\phi(j\delta)$  are constant during the optimization process and do not produce a gradient w.r.t  $\mathbf{T}$  for the cost function  $J_r$ . So the optimization problem and the gradients are simplified.

The cost of swarm reciprocal avoidance is defined as

$$\mathcal{P}_r(\tilde{\mathbf{p}}_j(t)) = \sum_{\phi} \max\{\psi_r(\tilde{\mathbf{p}}_j(t), p_\phi(j\delta)), 0\}^3, \quad (32)$$

$$\psi_r(\tilde{\mathbf{p}}_j(t), p_\phi(j\delta)) = d_r^2 - \|\tilde{\mathbf{p}}_j(t) - p_\phi(j\delta)\|^2, \quad (33)$$

where  $d_r$  is the safe clearance between each robot. And the gradient of  $\mathcal{P}_r$  w.r.t  $\tilde{\mathbf{p}}_j(t)$  is

$$\frac{\partial \mathcal{P}_r}{\partial \tilde{\mathbf{p}}_j(t)} = -2(\tilde{\mathbf{p}}_j(t) - p_\phi(j\delta))^T. \quad (34)$$

6) *Cost of Dynamic feasibility  $\mathcal{P}_d$* : We limit the maximum value of velocity and acceleration to guarantee that the robots can execute the trajectory.

$$\begin{aligned} \mathcal{P}_d(\tilde{\mathbf{p}}_j(t)) &= \mathcal{P}_{d,v}(\tilde{\mathbf{p}}_j(t)) + \mathcal{P}_{d,a}(\tilde{\mathbf{p}}_j(t)), \\ \mathcal{P}_{d,v}(\tilde{\mathbf{p}}_j(t)) &= \max\{\|\dot{\tilde{\mathbf{p}}}_j(t)\|^2 - v_m^2, 0\}^3, \\ \mathcal{P}_{d,a}(\tilde{\mathbf{p}}_j(t)) &= \max\{\|\ddot{\tilde{\mathbf{p}}}_j(t)\|^2 - a_m^2, 0\}^3, \end{aligned} \quad (35)$$

where  $v_m$  and  $a_m$  are the maximum velocity and acceleration.

## E. Discussion on solution quality of trajectory optimization

The proposed trajectory optimization process (17) aims to solve a challenging multi-stage Linear Quadratic Minimum Time (LQMT) problem, which is inherently non-convex and non-linear. Additionally, incorporating ESDF for obstacle avoidance introduces further non-convex constraints. As a result, guaranteeing the global optimal solution with the quasi-Newton method is not always possible. To address concerns regarding local minima and infeasible solutions, we have implemented measures that prioritize safety and dynamic feasibility while maintaining high-performance formation flight.

Firstly, we utilize hybrid-A\* searching algorithm [42] to generate initial trajectories that are collision-free and dynamically feasible, ensuring a valid final solution trajectory. During optimization, we give greater weight to obstacle avoidance and dynamic constraints to prioritize safety and feasibility. Additionally, we conduct collision checks on trajectories to enhance safety. Moreover, our distributed swarm optimization framework effectively mitigates the impact of local minima on overall formation performance. Implementing these measures, our method reliably achieves robust formation flight while maintaining computational efficiency.

## VI. SWARM REORGANIZATION METHOD

During the formation navigation, the swarm could encounter many unfavorable conditions, such as highly constrained space, inappropriate assignment of tasks, and sudden formation switching commands. To recover from these situations, we present a swarm reorganization method. The method aims to generate high-quality local goals which satisfy the desired formation distribution and respect the current states of each robot. With these local goals, the swarm can reform the desired shape quickly, even in highly constrained environments such as narrow corridors or holes.

Unlike high-frequency distributed formation trajectory optimization, the swarm reorganization method only runs at a low frequency when the stable state of formation flight is destroyed or about to be destroyed. The method first calculates the formation constraint awareness and then solves an optimal formation **AL**ignment and task **AS**ignment problem (ALAS). The former awareness distributedly quantifies the conflict between formation maintenance and obstacle avoidance of each robot, while the latter solves the ALAS problem centrally.

### A. Formation Constraint Awareness

First, we need to derive weights to indicate how severely the environment constrains the robots. These weights are called formation constraint awareness, which should be determined by the current formation status and obstacle information.

Inspired by our previous work [43], we hope to describe the conflict degree based on the relationship between different gradient information. Firstly, we retrieve the ESDF distance  $d_o(p)$  and the corresponding obstacle gradient  $\nabla d(p)$ . Meanwhile, we calculate the current gradient  $\nabla f_s(p)$  of the formation similarity term. Secondly, we calculate the cosine  $\beta$  of the angle between gradients  $\nabla d(p)$  and  $\nabla f_s(p)$

$$\beta = \frac{\nabla d(p) \nabla f_s(p)}{\|\nabla d(p)\| \|\nabla f_s(p)\|}. \quad (36)$$



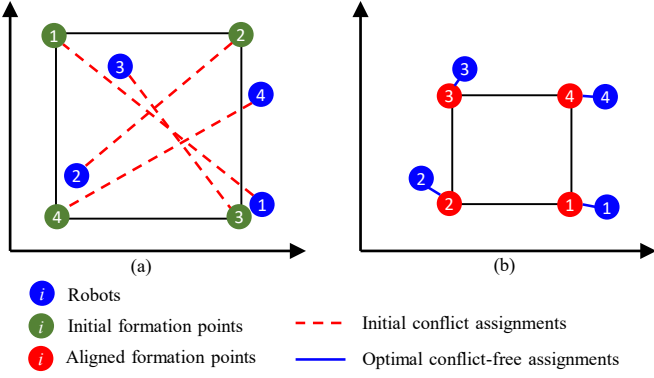


Fig. 4. Illustration of formation alignment and task assignment. (a) Before solving ALAS problem, the initial formation goals suffer from a large transition distance to robots and disordered assignments that may lead to deadlock. (b) With robots at the same positions, after solving ALAS, the formation goals enjoy low distance costs and better assignments.

Then we utilize the sigmoid function to map the cosine of the angle to a conflict coefficient  $\eta$

$$\eta(\beta) = \frac{1}{1 + e^{(\alpha\beta + \gamma)}}, \quad (37)$$

where  $\alpha$  regulates how fast this awareness rises as the cosine value  $\beta$  increases,  $\gamma$  regulates the dead zone and the activation zone of this angle-based awareness. The conflict coefficient  $\eta$  is maximum when the directions of  $\nabla d(p)$  and  $\nabla f_s(p)$  are opposite, which indicates the most conflicting case. And  $\eta$  reaches a minimum when the two gradients have the same direction, which means no conflict.

The formation constraint awareness should also consider the influence of the gradient magnitude and the distance of the current closest obstacle. Hence, we design the formation constraint awareness  $g_i$  of  $i^{th}$  robot as

$$g_i = \lambda \cdot \eta(\beta) \cdot \frac{\|\nabla J_f(p)\|}{d_o(p)}. \quad (38)$$

We apply the calculation to each robot in the swarm and thus get a formation constraint awareness vector  $\mathbf{g} = \{g_1, \dots, g_N\}$  of the whole swarm. To distinguish the most constrained ones, we use softmax function to amplify the variance of awareness vector  $\mathbf{g}$  and normalize the vector

$$\mathbf{w} = \text{softmax}(\mathbf{g}). \quad (39)$$

$\lambda$  in (38) is used to adjust the variance of elements in  $\mathbf{w}$ .  $\mathbf{w}$  is the final awareness vector describing the degree of formation-obstacle conflict of the robots in the swarm.

### B. Formation Alignment and Task Assignment Optimization

For formation flights in constrained scenes, e.g. in narrow corridors, the unconstrained robots with lower constraint awareness possess larger space to freely coordinate with other robots, since the obstacles don't hinder the formation requirement. On the contrary, the constrained robots with larger awareness always fall into the conflict between formation maintenance and obstacle avoidance. Hence, refining the positions of unconstrained robots to match up with the constrained ones is more reasonable when generating local goals for formation reorganization. In this work, we use  $\mathbf{w}$

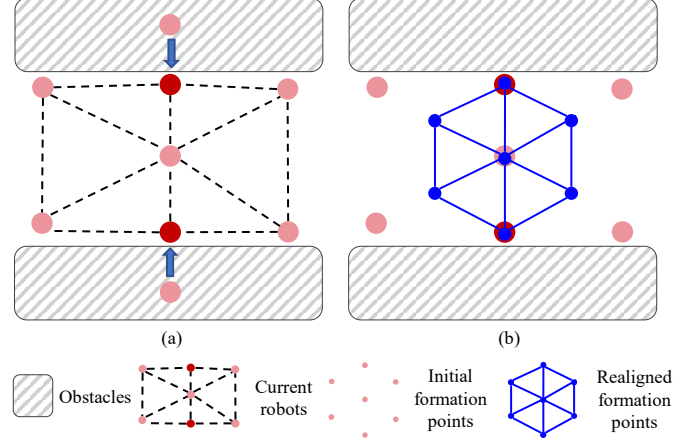


Fig. 5. Illustration of the weighted formation alignment problem in a narrow corridor. When a hexagon formation enters the corridor, the original formation distribution (pink points) cannot be maintained anymore. The upper and lower robots (red points) are severely pressed by the obstacles, and hence have the largest constraint awareness. After solving the awareness-weighted formation alignment, the swarm obtains a new desired formation distribution (blue points) that best matches up with the constrained robots.

in Sec. VI-A to weigh the robots when adjusting the formation distributions and place more weights on the constrained robots.

In this section, we only elaborate on the ALAS problem for local goal generation. Afterward, the global-remap-local-replan strategy uses the generated local goals to reorganize the formation, which is detailed in Sec. VII-A.

Let  $\mathbf{l}g_i = [lg_{ix}, lg_{iy}, lg_{iz}]^T, i = 1, \dots, N$  represents the current positions of robots. The desired formation shape template is given by  $N$  positions  $\mathbf{q}_j = [q_{jx}, q_{jy}, q_{jz}]^T, j = 1, \dots, N$ . Then, the aligned formation positions  $\mathbf{q}'_j$  can be written as

$$\mathbf{q}'_j = s \cdot \mathbf{q}_j + \mathbf{d}, \quad (40)$$

where  $s \in \mathbb{R}$  is the scaling factor,  $\mathbf{d} \in \mathbb{R}^3$  denotes the translation factor. In this work, the formation alignment is determined by a scaling factor and a translation factor.

ALAS is composed of formation alignment and task assignment as shown in Fig. 4. The former aims to find the optimal alignment of the desired formation based on a weighted Euclidean distance cost. And the latter solves the optimal assignment that matches the agents with the local goals.

The task assignment problem is formulated as

$$\min_{\sigma} \sum_i^n \|\mathbf{l}g_i - (s^* \cdot \mathbf{q}_{\sigma(i)} + \mathbf{d}^*)\|^2, \quad (41)$$

where  $\sigma \in S_N$  is the assignment map of the formation task and  $S_N$  is the symmetric group of all permutations from the set  $\{1, \dots, N\}$  to itself. Problem (41) solves the assignment that minimizes the overall transition distance between current robots and the aligned local goals.

The formation alignment problem is formulated as

$$\min_{s, \mathbf{d}} \sum_i^n w_i \cdot \|\mathbf{l}g_i - (s \cdot \mathbf{q}_{\sigma^*(i)} + \mathbf{d})\|^2, \quad (42)$$

where  $\sigma^*$  represents the optimal assignment,  $w_i$  is the awareness weight from Sec. VI-A. Problem (42) generates a standard formation that best fits into the current robot positions according to a distance cost weighted by the constraint awareness.

Fig. 5 illustrates how the alignment adjusts the formation distribution when the swarm is traversing a corridor.

Problems (41) and (42) are coupled. The whole ALAS problem has three decision variables: scaling factor  $s$ , translation  $\mathbf{d}$ , and assignment  $\sigma$ . The goal of ALAS is to find an optimal set of decision variables that minimize both (42) and (41). Note that there is no awareness weight  $w_i$  multiplied in formulation (41). Because in the assignment optimization, we only care about the total Euclidean distance cost, which is irrelevant to the degree of the constraint of any agent.

However, for formation alignment using only scaling factor  $s$  and translation  $\mathbf{d}$ , [25] proves that the corresponding assignment  $\sigma$  can be optimized in a decoupled manner, rather than alternating the two optimization phases iteratively. In [25], the optimal assignment solution  $\sigma^*$  is shown invariant w.r.t the changes in formation scaling factor  $s$  and translation  $\mathbf{d}$ . And the solution of (41) can be directly optimized by solving the following integer programming with new pseudo costs  $\kappa_{ij}$

$$\min_{\sigma=(x_{ij})} \sum_{i=1}^n \sum_{j=1}^n \kappa_{ij} x_{ij}, \quad (43)$$

$$\text{where } \kappa_{ij} = -\mathbf{lg}_i^T \mathbf{q}_j.$$

The formulation (43) is independent of the scale parameter  $s$  and translation  $\mathbf{d}$ . Hence, (43) can be first solved prior to the formation alignment phase. Then we determine the best alignment using the optimized assignment  $\sigma^*$ . The formation alignment problem with awareness weights is still convex and the closed-form solution to (41) is given In Appendix.A.

Given the solution of ALAS, the position of generated local goal  $\mathbf{lg}'_i$  for the  $i^{\text{th}}$  robot is calculated by

$$\mathbf{lg}'_i = s^* \cdot \mathbf{q}_{\sigma^*(i)} + \mathbf{d}^*. \quad (44)$$

After the ALAS optimization, the distribution of generated local goals is in the desired formation shape, and respects the formation-obstacle conflict of each robot.

## VII. SWARM AGREEMENT METHOD

### A. Global-remap-local-replan Strategy

The swarm system cannot quickly recover from disordered states caused by unknown obstacles or sudden changes in the desired formation shape. To address this challenge, we propose a novel approach that utilizes a global-remap-local-replan (GRLR) strategy for trajectory generation. This approach enables us to efficiently navigate complex environments while maintaining the swarm in a “critical state” of the desired formation, effectively balancing coordination and adaptability.

For distributed framework, communication helps the robot to obtain information about others and generate better coordination behavior. Especially in the case of formation transformation, collaborative decision-making can make the swarm formation converge quickly. However, circular dependencies sometimes occur due to communication delays, making it difficult to guarantee the consistency of decisions. Therefore, We design a **Global-Remap-Local-Replan** (GRLR) strategy for the swarm formation system, which only centrally remaps crucial parameters of formation and maintains the formation coordination through distributed replanning local trajectory.

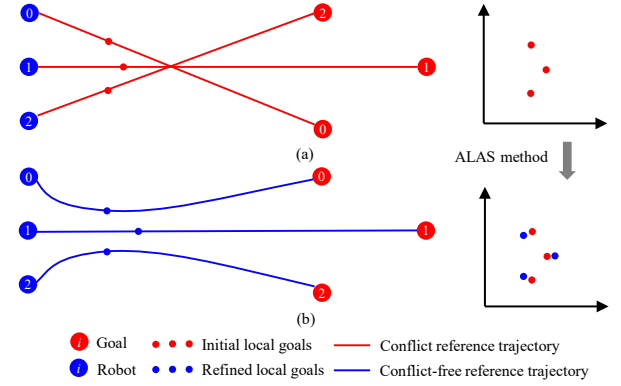


Fig. 6. Illustration of GRLR strategy. (a) The local-replan strategy generates initial local goals within the planning horizon. Due to the conflict reference trajectories, the swarm robots are expected to deadlock with the current formation behavior. Therefore, the global-remap strategy calls ALAS method to realign the shape and reassign the task of initial local goals. (b) After solving the refined local goals, the global-remap strategy generates conflict-free reference trajectories. In this way, the swarm formation will converge quickly with the new formation behavior.

### Algorithm 1 Global-remap strategy

**Notation:** Global reference trajectory  $\mathcal{T}_{ref}$ , Local trajectory  $\mathcal{P}_i$ , Initial local goals  $\mathbf{lg}_i$ , Global goals  $\mathbf{G}_i$ , Assignments  $\sigma$ , Scale  $s$ , Translation  $\mathbf{d}$ ;

- 1: **Initialize:**  $CallGlobalRemap \leftarrow False$ ,
- 2: **for** each robot  $i$  **do**
- 3:  $g_i \leftarrow \text{ConstrainedAwareness}(\mathcal{P}_i)$ ;  $\triangleright$  detailed in (38)
- 4: **if**  $g_i > g_d$  **then**  $\triangleright g_d$  is threshold of  $g_i$
- 5:  $CallGlobalRemap \leftarrow True$ ;
- 6:  $\mathbf{w} \leftarrow \text{softmax}(\mathbf{g})$ ;
- 7: **end if**
- 8: **end for**
- 9: **if**  $\text{SimilarityError}(\cdot) > e_{sim,d}$  **then**  $\triangleright$  detailed in (5)
- 10:  $CallGlobalRemap \leftarrow True$ ;
- 11:  $\mathbf{w} \leftarrow \text{softmax}(\mathbf{I})$ ;
- 12: **end if**
- 13: **if**  $CallGlobalRemap$  **then**
- 14:  $\sigma^* \leftarrow \text{Assignment}(\sigma, s, \mathbf{d})$ ;  $\triangleright$  detailed in (41)
- 15:  $s^*, \mathbf{d}^* \leftarrow \text{Alignment}(\mathbf{w}, \sigma^*)$ ;  $\triangleright$  detailed in (42)
- 16:  $\mathbf{lg}_i^* \leftarrow \text{RemapLocalGoals}(\sigma^*, s^*, \mathbf{d}^*)$ ;
- 17:  $\mathbf{G}_i^* \leftarrow \text{RemapGlobalGoals}(\mathbf{G}_i, \sigma^*)$ ;
- 18:  $\mathcal{T}_{ref}^* \leftarrow \text{GlobalTrajectoryReplan}(\mathbf{lg}_i^*, \mathbf{G}_i^*)$ ;
- 19: **Return**  $\mathcal{T}_{ref}^*$ ;
- 20: **end if**

GRLR strategy comprises the local-replan for a single robot and the global-remap for a formation-level system. The local-replan is a receding horizon incremental planning strategy [44], which allows each robot plans a trajectory within its limited sensing range. The local goals are selected on the global reference trajectories within planning horizon  $\Psi_p$ , as shown in Fig. 6 (a). The global-remap is an efficient centralized strategy that only remaps the local goals by solving ALAS method and refines the global reference trajectory, as shown in Fig. 6 (b). GRLR strategy is very suitable for distributed asynchronous systems, and there is no deadlock in swarm systems even in the presence of network delays.

The main workflow of the proposed global remap strategy is described in Algorithm.VII-A. Before generating the new formation behavior, the global-remap strategy checks if there are any emergence events (Line 1-12), such as the stable state of the swarm formation being destroyed (Line 9) or about to be destroyed (Line 4). Unlike the local-replan strategy is triggered at a fixed frequency, the global-remap strategy is started by emergent events (Line 13). Then the ALAS method is called to solve the optimal assignment  $\sigma^*$  and alignment  $s^*, \mathbf{d}^*$  (Line 14-15). Global-remap strategy remaps the local goals  $\mathbf{lg}_i^*$  and global goals  $\mathbf{G}_i^*$  and generates a new global trajectory  $\mathcal{T}_{ref}^*$  for each robot (Line 16-19). Finally, robots form the new formation by executing the local-replan strategy.

In this work, we utilize this semi-distributed GRLR strategy to make the swarm formation adaptable to unknown obstacles or sudden changes in the desired formation shape by replanning local trajectories at 1 Hz and checking emergent events for triggering the global-remap strategy at 20 Hz.

### B. Formation-level Global Path Finding

---

#### Algorithm 2 Formation-level Global Path Finding

---

**Notation:** Tree  $\mathcal{T}$ , State  $\mathbf{z}$ , Path cost  $c$ , Path  $\mathbf{P}$ ;

```

1: Initialize:  $\mathcal{T}_a \leftarrow \emptyset \cup \{\mathbf{z}_{start}\}$ ,  $\mathcal{T}_b \leftarrow \emptyset \cup \{\mathbf{z}_{goal}\}$ ,
    $c_{best} \leftarrow \infty$ ,  $FoundSolution \leftarrow False$ ;
2: for  $i = 1$  to  $N$  do
3:    $\mathbf{z}_{random} \leftarrow \text{Sample}(\mathbf{z}_{start}, \mathbf{z}_{goal}, c_{best})$ ;
4:   if not  $FoundSolution$  then
5:      $\mathbf{z}_{new} \leftarrow \text{GreedyExtendTree}(\mathcal{T}_a, \mathbf{z}_{random})$ ;
6:      $\mathbf{z}_{conn} \leftarrow \text{NearestVertice}(\mathbf{z}_{new}, \mathcal{T}_b)$ ;
7:      $c_{new} \leftarrow \text{Connect}(\mathbf{z}_{new}, \mathbf{z}_{conn}, \mathcal{T}_a, \mathcal{T}_b)$ ;
8:     if  $c_{new} < c_{best}$  then
9:        $c_{best} \leftarrow c_{new}$ ;
10:       $FoundSolution \leftarrow True$ ;
11:    end if
12:  else
13:     $\mathbf{z}_{new} \leftarrow \text{ExtendTree}(\mathcal{T}_a, \mathbf{z}_{random})$ ;
14:     $\mathbf{z}_{near} \leftarrow \text{NearVertex}(\mathbf{z}_{new}, \mathcal{T}_a)$ ;
15:     $\mathcal{T}_a \leftarrow \text{Rewire}(\mathbf{z}_{new}, \mathbf{z}_{near})$ ;
16:     $\mathbf{z}_{conn} \leftarrow \text{NearestVertice}(\mathbf{z}_{new}, \mathcal{T}_b)$ ;
17:     $c_{new} \leftarrow \text{Connect}(\mathbf{z}_{new}, \mathbf{z}_{conn}, \mathcal{T}_a, \mathcal{T}_b)$ ;
18:    if  $c_{new} < c_{best}$  then
19:       $c_{best} \leftarrow c_{new}$ ;
20:    end if
21:  end if
22:   $\text{SwapTrees}(\mathcal{T}_a, \mathcal{T}_b)$ ;
23: end for
24:  $\mathbf{P} \leftarrow \text{RetrievePath}(\mathcal{T}_a, \mathcal{T}_b)$ ;
25: Return  $\mathbf{P}$ ;

```

---

We propose a method for formation-level global path finding. Given a start and goal configuration, the planner generates a feasible path connecting them with collision-free intermediate formations. A bidirectional RRT approach is employed to address this path-finding problem.

Many navigation tasks expect the formation to maneuver with a desired scale. In practice, an oversized formation could reduce the vehicle's communication quality, while an overly

small formation scale could increase the risk of inter-vehicle collisions. Unlike the method in [45] which only samples position  $\mathbf{p} \in \mathbb{R}^3$  of the formation center, our method adds the formation scale  $s$  into the sampling space and makes the formation configuration  $\mathbf{z} = \{\mathbf{p}, s\} \in \mathbb{R}^3 \times \mathbb{R}^+$ . In this way, the objective of maintaining desired scale, i.e., minimizing the changes in scale along the path, can be handled by minimizing the  $L_2$ -norm distance of path in the configuration space  $\mathbf{z}$ .

Navigation in dense environments requires the robots to maintain a formation while letting the obstacles pass through the formation. The method in [46] samples the center position  $\mathbf{p}$ , and then the scale factor  $s$  is solved by optimizing the formation placement in obstacle-free convex regions. However, this approach does not allow any obstacle to intersect with the convex hull of the formation and hence wastes many solutions. In contrast, our method directly samples the whole states of the formation configuration  $\mathbf{z}$  to fully explore the solution space. For each edge of our RRT algorithm, a collision check is conducted on each robot rather than the formation's whole convex hull to allow obstacles to pass.

The main workflow of our bidirectional RRT planner is described in Algorithm.2, where two trees  $\mathcal{T}_a$  and  $\mathcal{T}_b$  grow towards each other from the initial state  $\mathbf{z}_{start}$  and the goal state  $\mathbf{z}_{goal}$  respectively. Before the first solution is found, the bidirectional planner extends the trees in an RRT-Connect [47] manner (Line 5-7). In **GreedyExtendTree()** and **Connect()**, the greedy heuristic [47] is adopted to aggressively explore the environment and make tree-connection attempts. After a feasible solution is found, i.e. a finite path cost  $c_{new}$  is returned by **Connect()**, the function **Sample()** computes an informed sampling set with the new cost  $c_{new}$  as depicted in [48]. Then the standard Bidirectional-RRT\* [49] procedures are conducted in each loop to update the trees (Line 13-17). Since the path cost is  $L_2$ -norm distance in the configuration space  $\mathbf{z}$ , informed sampling [48] and Bidirectional-RRT\* [49] can guarantee the asymptotic optimality of the path solution.

This formation-level path planner is deployed to render a global path when the global environment information is available. Then global trajectories connecting the waypoints of the global path are generated using MINCO [39], and the framework in Sec.V is employed to optimize the local motions.

## VIII. BENCHMARK

In the benchmark, it is important to assess the distortion degree of the current formation  $\mathcal{F}^c$  fairly relative to the desired one  $\mathcal{F}^d$  during flight. Inspired by [22], we solve the following nonlinear optimization problem to find the best similarity transformation ( $Sim(3)$  transformation) that aligns  $\mathcal{F}^c$  with  $\mathcal{F}^d$ . Then the average formation distance degree  $\bar{e}_{dist}$  is calculated at the normalized formation scale

$$\bar{e}_{dist} = \frac{1}{s_o \cdot L} \int_{\mathcal{L}} \min_{\mathbf{R}, \mathbf{t}, s} \sum_{i=1}^n \|\mathbf{p}_i^d - (s \mathbf{R} \mathbf{p}_i^c + \mathbf{t})\|^2 dl, \quad (45)$$

where  $\mathbf{p}_i^d$  and  $\mathbf{p}_i^c$  represent the position of  $i^{th}$  robot in formation  $\mathcal{F}^d$  and  $\mathcal{F}^c$ , respectively. The  $Sim(3)$  transformation is composed of a rotation  $\mathbf{R} \in SO(3)$ , a translation  $\mathbf{t} \in \mathbb{R}^3$  and a scale expansion  $s \in \mathbb{R}_+$ . Moreover,  $s_o$  is the initial formation

TABLE I  
FORMATION PARAMETERS OF THE PROPOSED METHOD

Parameter	Symbol	Value
Similarity error threshold	$e_{sim,d}$	0.05
Constraint awareness threshold	$g_d$	$2/N$
Parameter for regulation in (37)	$\alpha$	5
Parameter for regulation in (37)	$\lambda$	25
Parameter for regulation in (37)	$\gamma$	-1
Sampling time interval (s)	$\delta$	0.5
Planing Horizon (m)	$\Psi_p$	7.5
Max velocity (m/s)	$v_m$	1.0
Max acceleration (m/s <sup>2</sup> )	$a_m$	6.0
Weight for control effort	$\lambda_e$	10000.0
Weight for total time	$\lambda_t$	80.0
Weight for swarm reciprocal avoidance	$\lambda_r$	10000.0
Weight for obstacle avoidance	$\lambda_o$	10000.0
Weight for swarm formation similarity	$\lambda_f$	10000.0
Weight for dynamic feasibility	$\lambda_d$	10000.0

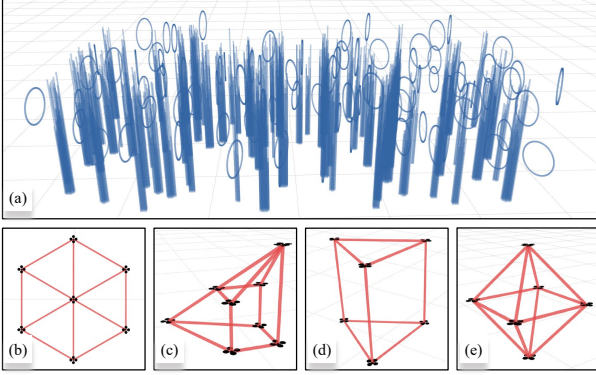


Fig. 7. Random forest map and formation types of benchmarks. (a) Random forest map. (b) Regular hexagon shape. (c) Irregular shape. (d) Triangular prism shape. (e) Octahedron shape.

scale, and  $L$  is the length of formation trajectory  $\mathcal{L}$ . Optimizing the transformation in (45) and applying it to formations, the influence of scaling and rotation is squeezed out so that all the formations can be equitably rated by measuring the position error w.r.t the desired formation. A larger  $\bar{e}_{dist}$  represents a larger distortion from the desired formation  $\mathcal{F}^d$ . Besides, we also calculate the average formation similarity degree  $\bar{e}_{sim}$

$$\bar{e}_{sim} = \frac{1}{s_o \cdot L} \int_{\mathcal{L}} \|\hat{\mathbf{L}} - \hat{\mathbf{L}}_{des}\|_F^2 dl, \quad (46)$$

where  $\hat{\mathbf{L}}$  and  $\hat{\mathbf{L}}_{des}$  are detailed in (5) and the formation similarity error  $\|\hat{\mathbf{L}} - \hat{\mathbf{L}}_{des}\|_F^2$  is proposed in Sec.IV. We show important parameters in Table I used in the following benchmarks, simulations, and real-world experiments. All benchmarks are run on a desktop with an Intel i7-12700 CPU.

#### A. Adaptability of Graph-based Formation Definition

To demonstrate the adaptability of graph-based formation definition in Sec.IV, we conduct numerous benchmarks compared to the mainstream formation definition methods concluded in [50], which are categorized based on the controlled variables, namely position-based [51], distance-based [52] and displacement-based methods [53].

We implement these methods in our framework and adapt them to the dense environments by replacing the original cost

$J_s$  in (12) to generate uniform optimal formation position sequence  $\hat{\mathbf{p}}_i^*$  for each robot  $i$ . For the position-based method, we set drone\_0 as the leader and predefined the absolute relative positions for all other robots to specify the desired formation. So its cost is  $J_{s,1} = 0$ . The distance-based method optimizes the error of desired inter-agent distances

$$J_{s,2} = \sum_{j \in N} (\|\mathbf{p}_i - \mathbf{p}_j\| - \|\mathbf{p}_i^d - \mathbf{p}_j^d\|)^2, \quad (47)$$

where  $N$  is the number of robots, and  $\mathbf{p}_i^d$  is the desired position vector for the  $i^{th}$  robot. The displacement-based method optimizes the error of desired relative displacements

$$J_{s,3} = \sum_{j \in N} \|(\mathbf{p}_i - \mathbf{p}_j) - (\mathbf{p}_i^d - \mathbf{p}_j^d)\|^2. \quad (48)$$

Then we simulate four different geometric formation types in a high-density environment of  $40 \times 15m$  size with randomly generated obstacles, as shown in Fig. 7 (a). 2D and 3D formations with irregular and regular geometries are considered, namely formation types in a regular hexagon, irregular geometry, triangular prism, and octahedron, as shown in Fig. 7 (b)-(e). To fully compare the adaptability of these methods, we design four different scenarios considering both scaling and rotational variation of formation shape. The formation's initial and final positions may differ in scale and rotation. Then the scenarios are corresponding categorized as 'Same to same', 'Rotation change', 'Scale change', and 'Scale & rotation change'. We test each method 20 times for each scenario and formation type. The corresponding results over  $\bar{e}_{dist}$  (45) and  $\bar{e}_{sim}$  (46) are summarized in Table II.

Unlike our graph-based formation definition, in other methods, changing the scale and rotation of the formation is not permitted during the flight. As shown in Table II, in the same formation type and same scenario, the data states that our method achieves promising results with almost the lowest  $\bar{e}_{sim}$  and  $\bar{e}_{dist}$ . Moreover, our method shows the lowest error growth rate when the scenario becomes more complicated. In the same scenario, the distortion degrees of all methods decrease with the change of formation type from 2D to 3D centrosymmetric structure, which shows that the formation maintenance is also related to the structural stability of the formation itself. In addition, the distance-based method is invariant to the rotation and achieves relatively acceptable performance in the 'Rotation change' scenario. Nevertheless, it can not handle size-variant cases. Similarly, other methods are sensitive to rotation or scaling, leading to significant performance degradation in such scenarios. Generally speaking, our graph-based formation definition method achieves scaling and rotational invariance. The invariance improves the formation flight's adaptability and outperforms the mainstream methods in complicated scenarios.

#### B. Predictability of Spatial-Temporal Trajectory Optimization

To prove the predictability of formation trajectory optimization in Sec.V, we compare our work with the virtual rigid body (VRB) method [26], a SOTA formation control framework that avoids obstacles using potential fields. Moreover, we also compare the performance between the spatial-only and spatial-temporal optimization to illustrate the importance of the time



TABLE II  
PERFORMANCE COMPARISON BETWEEN FORMATION DEFINITION METHODS

Scenario	Formation type	Regular hexagon		Irregular shape		Triangular prism		Octahedron	
	Error Method	$\bar{e}_{dist}(\%)$	$\bar{e}_{sim}(\%)$	$\bar{e}_{dist}(\%)$	$\bar{e}_{sim}(\%)$	$\bar{e}_{dist}(\%)$	$\bar{e}_{sim}(\%)$	$\bar{e}_{dist}(\%)$	$\bar{e}_{sim}(\%)$
Same to same	Position	39.120	0.384	35.649	0.384	34.506	0.374	21.534	0.341
	Displacement	16.231	0.172	15.023	0.159	14.952	0.125	11.645	0.153
	Distance	15.489	0.164	14.295	0.131	<b>14.009</b>	<b>0.118</b>	10.285	0.113
	Ours	<b>15.443</b>	<b>0.161</b>	<b>14.287</b>	<b>0.128</b>	14.012	0.119	<b>10.281</b>	<b>0.112</b>
Rotation change	Position	57.456	0.945	51.298	0.732	49.821	0.612	34.124	0.542
	Displacement	39.456	0.412	31.012	0.439	29.546	0.345	23.125	0.353
	Distance	27.513	0.312	22.312	0.234	21.031	0.201	14.173	0.159
	Ours	<b>19.234</b>	<b>0.218</b>	<b>17.032</b>	<b>0.171</b>	<b>15.013</b>	<b>0.151</b>	<b>12.146</b>	<b>0.138</b>
Scale change	Position	59.654	1.098	58.416	0.784	53.246	0.741	37.845	0.555
	Displacement	42.516	0.629	40.021	0.624	39.412	0.398	29.845	0.395
	Distance	59.542	1.030	59.105	0.799	54.126	0.632	38.451	0.578
	Ours	<b>18.332</b>	<b>0.192</b>	<b>18.196</b>	<b>0.185</b>	<b>16.023</b>	<b>0.179</b>	<b>12.264</b>	<b>0.164</b>
Scale & rotation change	Position	62.584	1.304	60.124	0.796	56.213	0.832	41.856	0.635
	Displacement	45.627	0.755	40.194	0.631	40.168	0.423	31.288	0.504
	Distance	62.154	1.250	61.059	0.804	54.317	0.684	42.138	0.684
	Ours	<b>20.231</b>	<b>0.243</b>	<b>18.345</b>	<b>0.204</b>	<b>16.851</b>	<b>0.183</b>	<b>12.357</b>	<b>0.175</b>

TABLE III  
PERFORMANCE COMPARISON BETWEEN FORMATION NAVIGATION METHODS

Scenario	Formation type	Regular hexagon			
	Error Method	<i>success</i>	<i>rate</i> (%)	<i>length</i> (m)	$\bar{e}_{dist}(\%)$ $\bar{e}_{sim}(\%)$
Sparse	VRB [26]	75		22.978	57.962 0.984
	Spatial-only	100		21.923	15.023 0.152
	Spatial-temporal	<b>100</b>		<b>21.756</b>	<b>11.240</b> <b>0.138</b>
Medium	VRB [26]	25		-	- -
	Spatial-only	100		22.130	14.927 0.158
	Spatial-temporal	<b>100</b>		<b>21.932</b>	<b>13.274</b> <b>0.153</b>
Dense	VRB [26]	0		-	- -
	Spatial-only	100		22.283	17.630 0.185
	Spatial-temporal	<b>100</b>		<b>22.133</b>	<b>15.443</b> <b>0.161</b>

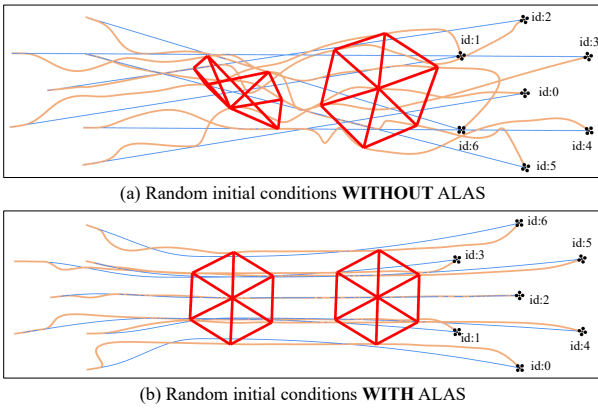


Fig. 8. Comparison of formation flight under improper initial conditions with and without ALAS. (a) In the case without ALAS, the executed trajectories (orange lines) are winding, and the formation shape (red lines) converges slowly due to the crossed global trajectories (blue lines). (b) In the case of ALAS, swarm reorganization makes the formation flight process orderly.

domain for formation flight. We simulate seven drones flying in a regular hexagon from one side to another with a velocity limit of  $0.5m/s$ . The cluttered area is of  $30 \times 15m$  size, and three obstacle densities are tested for comparison. Parameters are finely tuned for the best performance of each method.

The results are summarized in Table III, which indicates that the VRB method [26] has an unsatisfactory success rate when dealing with medium and dense obstacles. This is mainly due to the short-term obstacle avoidance generated by multiple interacting potential fields, which often leads to local minima near the corridors, causing robots to become trapped and fail. However, optimization methods consider the future movement of formation, so they can balance the formation maintenance and obstacle avoidance but not break the formation shape. Therefore, optimization methods achieve better performance and maintain the success rate.

We can also conclude that the spatial-temporal method is much more effort-efficient, robust, and flexible when considering temporal optimization. The spatial-only method cannot adjust the trajectory in the time domain, which leads to excessive spatial deformations of the trajectory. So the trajectory length and the formation error  $\bar{e}_{sim}$  and  $\bar{e}_{dist}$  are larger in the spatial-only method.

### C. Elasticity of Swarm Reorganization Method

To validate the swarm reorganization methods in Sec.VI, we design two benchmarks to illustrate the necessity of task assignment and the adaptability of formation alignment.

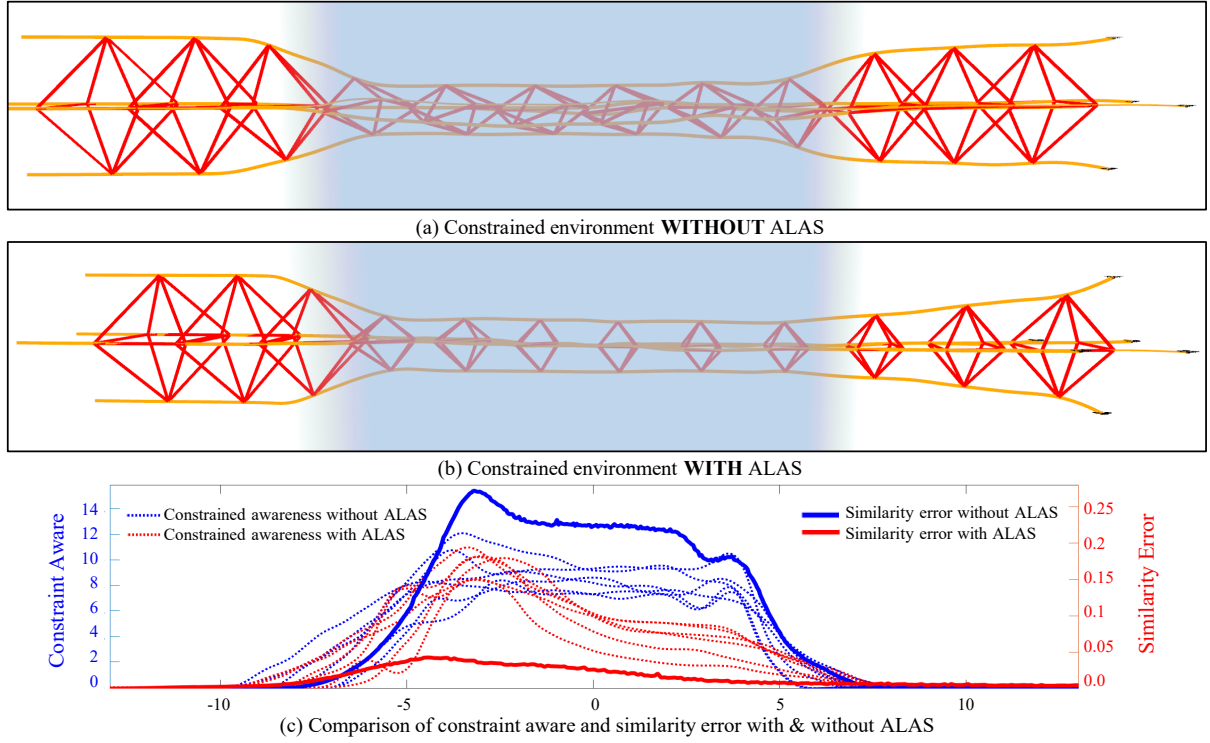


Fig. 9. Comparison of formation flight in a constrained environment with and without ALAS. The light blue area represents a wall with a hole in which swarm robots should shrink to pass through. (a) In the case without ALAS, the formation shape (red lines) is deformed to force through the area. (b) In the case with ALAS, the formation shape actively gets smaller to pass the area. (c) Formation flight with ALAS can decrease each robot's constraint awareness (red dotted lines). This improvement maintains the formation shape with lower similarity error (red lines).

Firstly we design a comparison of regular hexagon formation flight under improper initial conditions with and without ALAS to validate the necessity of formation task assignment, as shown in Fig. 8. The blue lines represent each robot's global trajectory and its assigned tasks in the formation. In Fig. 8(a), the global trajectories are partially crossed due to inappropriate task assignment, leading to trajectory optimization conflicts. So the executed trajectories shown by orange lines look very disordered, and the formation shape shown by red lines converges slowly. In Fig. 8(b), the above problems are effectively resolved by considering ALAS. After one calculation of ALAS, the swarm robots reassign formation tasks and quickly reach a swarm consensus. Then the swarm formation smoothly converges to the desired shape and navigates to the destination in an energy-efficient way. The results of this test validate the necessity of task assignment.

Then, we compare formation flight with and without ALAS when passing through a constrained hole to display the adaptability of ALAS. The results in Fig. 9(a) and Fig. 9(b) show that the formation shape may be deformed when passing through the corridor without ALAS. Otherwise, the case with ALAS can adaptively adjust the formation shape to the constrained environments. From the quantitative analysis results in Fig. 9(c), the case with ALAS can quickly adjust the formation scale and make the swarm reach a consensus so that the formation similarity error and constraint awareness of each robot decline rapidly. However, in the case without ALAS, a higher similarity error and constraint awareness are maintained until the swarm formation leaves the hole, which means the swarm formation is always within the limitations of the

TABLE IV  
COMPARISON BETWEEN GLOBAL PATH FINDING METHODS

	Alonso-Mora's method [46]	Ours
Sampling time (s)	2.0	2.0
Desired scale (m)	3.0	3.0
Path length (m)	50.77	<b>24.10</b>
Min scale along the path (m)	1.88	<b>2.98</b>

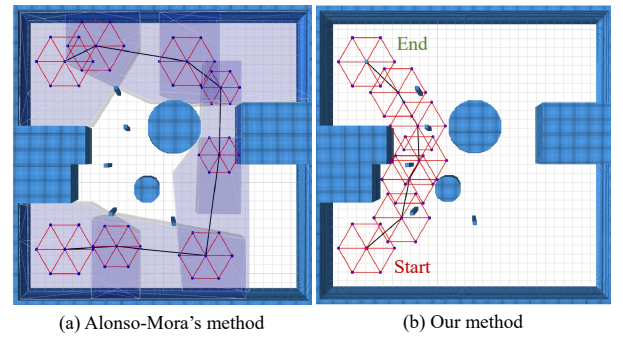


Fig. 10. Comparison of Alonso-mora's method and our formation-level path finding method in the same constrained map. (a) This method samples convex regions (purple polyhedra) in free space and connects them if the intersections are traversable in formation. Because the convex regions must be generated in the safe space, this method is too conservative to generate a longer path with a smaller scale. (b) Our method directly samples the whole states of the 3D-scale formation configuration to fully explore the solution space to allow the formation to pass through tiny obstacles.

environment so that the formation shape cannot converge. This benchmark proves the adaptability of formation alignment.

#### D. Resilience of Swarm Agreement Method

To highlight the resilience of our swarm agreement method, we compare Alonso-Mora's global planning method [46] and

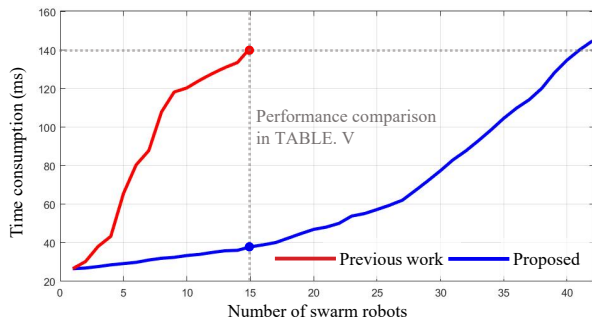


Fig. 11. Comparison of time consumption between previous work and proposed work [4]. We compare the performance of two methods in the 15-robots scenario. The detailed comparison results are shown in Table V.

TABLE V  
PERFORMANCE COMPARISON IN 15-DRONES SCENARIO

	Previous method [4]	Proposed method
Time consumption ( <i>ms</i> )	141.7	<b>38.2</b>
success rate (%)	95.0	<b>100.0</b>
length ( <i>m</i> )	47.387	<b>45.282</b>
$\bar{e}_{dist}$ (%)	12.439	<b>11.724</b>
$\bar{e}_{sim}$ (%)	0.147	<b>0.139</b>

our formation-level path finding method in a constrained map, as shown in Fig. 10. This map comprises several blocks and some tiny obstacles, in which swarm robots need to find the path that allows the formation to pass safely. We test the planners 20 times, and Table IV shows the averaged resultant data. The results state that Alonso-Mora’s method [46] is unsatisfactory in dealing with this scenario. Method [46] has no penalty for scale changes of formation, which could choose the corridor route that leads to sudden changes in formation scale. Moreover, it can not handle tiny obstacles and thus yield to a longer path with smaller scales, as shown in Fig. 10(a). Unlike Alonso-Mora’s method, our formation-level path-finding method directly samples in the augmented 3D-scale space and can better maintain the desired formation scale. As shown in Fig. 10(b), our method generates a much shorter path and only sacrifices a small quantity of formation scale. Therefore, our method can handle the map with blocks and tiny obstacles and find safe guidance for swarm formation, which is more suitable for dense environments.

#### E. Efficiency of Decoupled Formation Optimization

We compare our proposed decoupled formation optimization with the previously coupled formation optimization [4] which directly calls formation similarity distance metric (5) multiple times in the optimization process. To ensure a fair comparison, we exclude the ALAS problem during this benchmark. Both methods’ results are shown in Fig. 11. The previous method [4] only supports small-scale swarm formation since the computation time grows exponentially. Thanks to the decoupled approach, the time consumption of our proposed method for a swarm of 42 robots is not more than 150ms, which can support the real-time application for large-scale swarms.

We select the experimental data from the 15-drones scenario, as presented in Table V. Our method not only achieves



Fig. 12. Illustration of palm-sized swarm aerial robots.

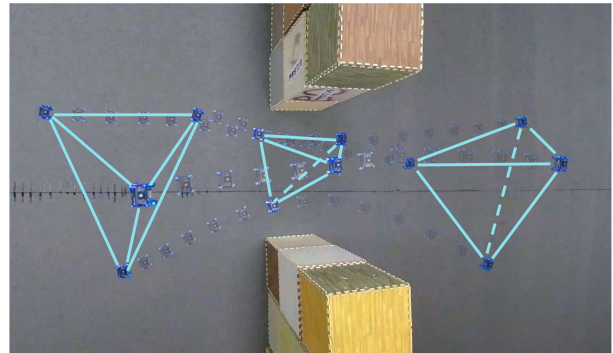


Fig. 13. Composite snapshots of a regular tetrahedron formation passing through a corridor. The swarm rotates and compresses the formation shape to fly through the narrow space from right to left. The blue line shows the outline of the formation shape.

significantly shorter computation times than the previous method, but also demonstrates better performance in terms of formation maintenance, success rate, and trajectory length. This validates the effective decoupling of our method, leading to comprehensive performance improvements.

## IX. REAL WORLD EXPERIMENTS AND SIMULATION

### A. Real world Experiments

Our method is integrated with an autonomous distributed aerial swarm system stated in Sec.III. The swarm shares some information, such as trajectories, through a broadcast network, which is the only connection among all robots. As shown in Fig. 12, we use a palm-sized quadrotor platform [5] with local sensors and an onboard computer. Software modules such as estimation, perception, planning, and control are all running onboard in real-time. The maximum number of swarm robots during real-world experiments is 16. Three different real-world experiments are designed to verify the proposed formation flight system’s characteristics fully.

In the first experiment, as shown in Fig. 13, four quadrotors in a 3-D regular tetrahedron formation manage to pass through a narrow corridor safely. During the flight, the swarm adaptively rotates and compresses the formation shape in response to environmental changes. This test proves that the scaling and rotational invariance provides more flexibility for formation flights in constrained spaces.

Then we design a 3-D formation shape transformation experiment to testify the reorganization ability of our method, as shown in Fig. 14. In Fig. 14(a), the desired formation is cube shape, but the swarm robots navigate from the unconverged



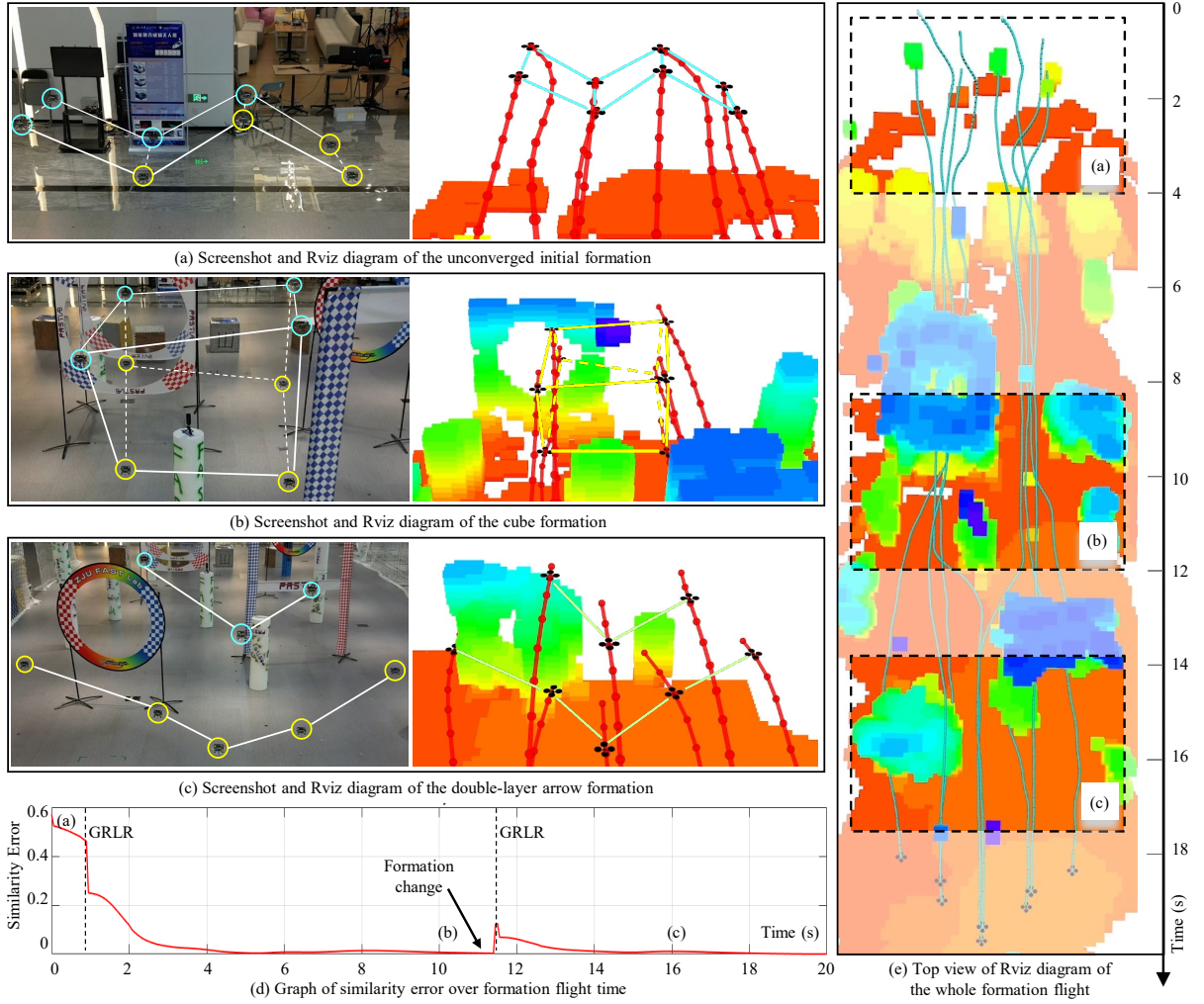


Fig. 14. Illustration of 3-D formation transformation experiment. The blue circle represents quadrotors assigned to the upper position, and the yellow circle represents a lower position. The white line represents the outline of the formation shape. (a) Formation flight starts from unconverged initial positions and improper initial task assignments. So swarm robots call the GRLR strategy to reorganize the formation parameters. (b) After 3 seconds, swarm robots converge to the desired cube shape. Then swarm receives a formation transformation command and quickly calls the GRLR strategy. (c) After 3 seconds, swarm robots converge to the desired double-arrow shape. (e) The executed trajectories (light blue lines) indicate that swarm formation is convergent most time. (d) A more accurate numerical analysis states that similarity error decreases quickly after calling the GRLR strategy.

initial positions and improper initial task assignments. In the beginning, the swarm robots quickly call the GRLR strategy, then make the formation shape quickly converge to the desired square shape, as shown in Fig. 14(b). Then, the swarm robots receive a formation transformation command from the station laptop and converge to a double-arrow shape, as shown in Fig. 14(c). The top view of the navigation process is shown in Fig. 14(e). From the light blue executed trajectories, we can see that the flight behavior of swarm robots tends to be consistent during time [4, 10] and time [14, 20]. Moreover, during time [0, 4] and time [10, 14], swarm robots try to reach a swarm consensus and frequently adjust flight behavior to form the desired formation shape. A more accurate numerical analysis can be seen from Fig. 14(d). When the formation system is far from the convergence state or meets a formation transformation command, swarm robots adjust formation alignment and task assignment by calling the GRLR strategy at the time corresponding to the dotted line. According to the similarity error represented by the red line, we can see that except for the non-convergence state at the initial moment and

sudden formation transformation, the swarm formation can maintain the desired shape while avoiding obstacles.

Finally, we conduct a 16-drone swarm formation flight experiment outdoors. To the best of our knowledge, this is the largest fully autonomous formation flight experiment in a complex outdoor environment. As shown in Fig. 1, 16 drones form a triangular queue shape and successfully traverse an obstacle-rich area without collision. This area has many street trees, stakes, and street lamps. This experiment proves the robustness and large-scale ability of our proposed method. For more details, please view the experimental video<sup>3</sup>.

#### B. Simulation Experiments

In order to comprehensively show the characteristics of the proposed formation flight method, we also conduct several simulation experiments to supplement the real-world experiments. All simulation experiments are run on a desktop with an Intel i7-12700 CPU in real-time.

Firstly, we design a formation multiple transformation simulation to testify swarm reorganization ability when coping

<sup>3</sup><https://www.youtube.com/watch?v=uEMyvPxYqmA>



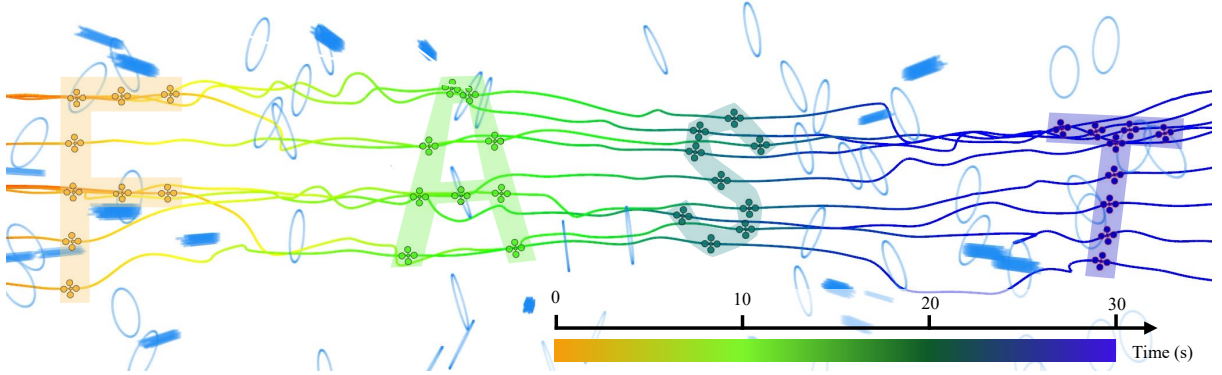


Fig. 15. Formation multiple transformation simulation. The different colors of the trajectories and robots correspond to the different timestamps. We choose four specific timestamps, and the corresponding formation shapes constitute “FAST” (<http://zju-fast.com/>).

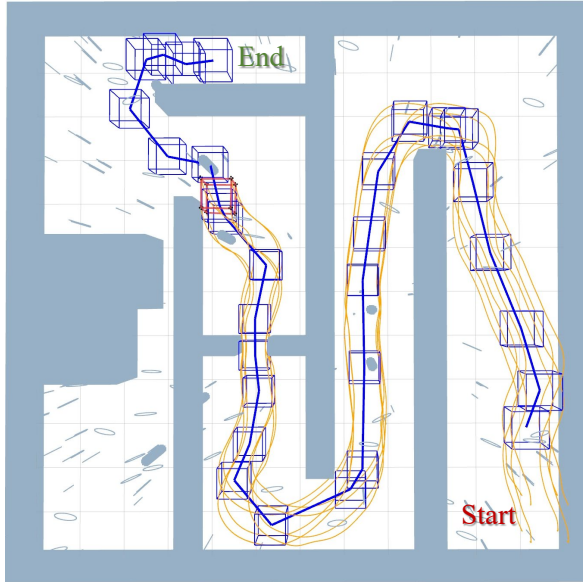


Fig. 16. Formation flight in a maze map. The formation-level global path (blue line) avoids walls while ignoring small obstacles. Then swarm formation follows the global path and generates local trajectories to avoid all obstacles and maintains the formation shape.

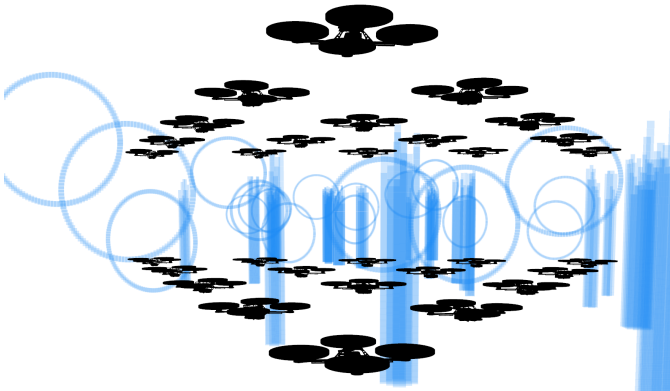


Fig. 17. Large-scale formation flight from far to near. It can be seen from the executed trajectories that the swarm robots still maintain the formation while avoiding obstacles.

with emergency changes in dense environments, as shown in Fig. 15. The different colored trajectories represent the time flow of the formation flight from left to right. The robots of the same color correspond to the formation shape at that timestamp. The command of formation transformation is given in real-time instead of being set in advance. The swarm robots must overcome the instantaneous change of desired formation shape and quickly converge to the new formation state. Thanks to the excellent ability of the proposed method to rapidly transform formation in complex environments, we finally generated the acronym “FAST” for our laboratory.

Then, we set up a special maze simulation environment consisting of walls and many small obstacles such as posts and rings. In this simulation, we aim to verify swarm agreement ability. The formation-level global path finding method first runs in the global map, which is a centralized method proposed in Sec.VII-B. Then, it generates a global path that considers the scale of formation shape, as shown by the blue lines in Fig. 16. The blue cube shapes represent sampling points. After that, 8 robots form a cube formation, navigate following the global path and generate local formation trajectories using distributed methods in Sec.V. This simulation demonstrates that our formation flight system can better accommodate the obstacle constraints and provide safer guidance for the formation flight.

Finally, to test our method’s effectiveness with large-scale irregular formations, we design a double-arrow formation consisting of 30 drones. As depicted in Fig. 17, the swarm successfully avoids the obstacles, and the desired formation is well preserved during the flight.

## X. CONCLUSION AND FUTURE WORK

In this paper, we analyze the core dilemmas to achieve formation flight in dense environments in detail and accurately summarize *PAPER* criteria to solve the above problems. Then we propose a hierarchical formation flight architecture composed of graph-based formation definition, distributed formation trajectory optimization, swarm reorganization method, and swarm agreement methods. The proposed complete formation flight system satisfies all *PAPER* criteria and achieves excellent performance in maintaining cooperative formation flight in dense environments. We design comprehensive benchmarks

in terms of adaptability, predictability, elasticity, resilience, and efficiency to verify the outstanding performance of our proposed method. Finally, we conduct abundant real-world experiments and simulations to prove that we solve the problem of autonomous formation flight in dense environments within large-scale swarms.

In the future, we intend to further improve the efficiency of distributed formation flight method through local information propagation of sub-graphs. Furthermore, while our work currently requires an operator to manually determine the formation shape, the research on optimizing formation shape is a promising area. We believe it has the potential to showcase more intelligent and cooperative swarm behavior, ultimately leading to an enhanced task capacity.

## APPENDIX

### A. The Closed-form Solution of Alignment Problem

Now we derive the closed-form solution to the formation alignment problem. Assume after the assignment phase, we have the current robot position  $\mathbf{l}\mathbf{g}_i$  and desired formation  $\mathbf{q}_i$  with optimal matches. We use  $c_i$  to denote the  $i^{th}$  term of the alignment objective in (42). Then we have

$$\begin{aligned} c_i &= w_i \cdot \|\mathbf{l}\mathbf{g}_i - (s\mathbf{q}_i + \mathbf{d})\|^2, \\ &= w_i (\mathbf{l}\mathbf{g}_i - s\mathbf{q}_i - \mathbf{d})^T (\mathbf{l}\mathbf{g}_i - s\mathbf{q}_i - \mathbf{d}), \\ &= w_i \mathbf{l}\mathbf{g}_i^T \mathbf{l}\mathbf{g}_i + s^2 w_i \mathbf{q}_i^T \mathbf{q}_i - 2s w_i \mathbf{l}\mathbf{g}_i^T \mathbf{q}_i + \\ &\quad 2s w_i \mathbf{q}_i^T \mathbf{d} - 2w_i \mathbf{l}\mathbf{g}_i^T \mathbf{d} + w_i \mathbf{d}^T \mathbf{d}. \end{aligned} \quad (49)$$

We use  $F$  to denote the alignment objective in (42). Then the objective  $F$  can be written as

$$\begin{aligned} F(s, \mathbf{d}) &= \sum_i^n c_i, \\ &= b_{lg} + s^2 b_q - 2s b_{lg,q} \\ &\quad + 2s \hat{\mathbf{q}}^T \mathbf{d} - 2\hat{\mathbf{l}}\mathbf{g}^T \mathbf{d} + \mathbf{d}^T \mathbf{d}, \end{aligned} \quad (50)$$

where

$$\begin{aligned} b_{lg} &= \sum_i^n w_i \mathbf{l}\mathbf{g}_i^T \mathbf{l}\mathbf{g}_i, \\ b_q &= \sum_i^n w_i \mathbf{q}_i^T \mathbf{q}_i, \\ b_{lg,q} &= \sum_i^n w_i \mathbf{l}\mathbf{g}_i^T \mathbf{q}_i, \end{aligned} \quad (51)$$

$$\hat{\mathbf{q}} = \sum_i^n w_i \mathbf{q}_i, \quad \hat{\mathbf{l}}\mathbf{g} = \sum_i^n w_i \mathbf{l}\mathbf{g}_i. \quad (52)$$

Since the awareness weights  $w_i$  are the outputs of a softmax function, we have the property  $\sum_i^n w_i = 1$ . Note that we use this property to simplify the last term in (50). Now we need to prove that the objective  $F$  in (50) is convex w.r.t. scale parameter  $s$  and translation  $\mathbf{d}$ . The Hessian matrix of the objective is given by

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 F}{\partial s^2} & \frac{\partial^2 F}{\partial s \partial \mathbf{d}} \\ \frac{\partial^2 F}{\partial \mathbf{d} \partial s} & \frac{\partial^2 F}{\partial \mathbf{d}^2} \end{bmatrix} = \begin{bmatrix} 2b_q & 2\hat{\mathbf{q}}^T \\ 2\hat{\mathbf{q}} & 2\mathbf{I}_{3 \times 3} \end{bmatrix} \in \mathbb{R}^{4 \times 4}. \quad (53)$$

The eigenvalues  $\lambda$  of the Hessian are as follows

$$\lambda = \left\{ 2, 2, (b_q + 1) \pm \sqrt{(b_q + 1)^2 - 4b_q} \right\}, \quad (54)$$

where

$$\begin{aligned} b_w &= b_q - \hat{\mathbf{q}}^T \hat{\mathbf{q}}, \\ &= \sum_i^n w_i \mathbf{q}_i^T \mathbf{q}_i - \left( \sum_i^n w_i \mathbf{q}_i \right)^T \left( \sum_i^n w_i \mathbf{q}_i \right). \end{aligned} \quad (55)$$

We need to prove that (55) is non-negative, then all the eigenvalues in (54) will be non-negative, thus the Hessian is semi-positive definite, the objective in (50) will be convex w.r.t. variables  $s$  and  $\mathbf{d}$ .

For the non-negative weights  $w_i$ , we have the property  $\sum_i w_i = 1$  from the softmax function, so we construct a convex function  $h(\mathbf{x}) = \mathbf{x}^T \mathbf{x}$ , and apply weighted Jensen's Inequality. We get

$$w_1 h(\mathbf{q}_1) + \dots + w_n h(\mathbf{q}_n) \geq h(w_1 \mathbf{q}_1 + \dots + w_n \mathbf{q}_n), \quad (56)$$

$$\Rightarrow \sum_i^n w_i \mathbf{q}_i^T \mathbf{q}_i \geq \left( \sum_i^n w_i \mathbf{q}_i \right)^T \left( \sum_i^n w_i \mathbf{q}_i \right). \quad (57)$$

Thus (55) is nonnegative, the objective  $F$  is convex. Then we can obtain the closed-form solution of this alignment problem by solving

$$\begin{cases} \partial F / \partial s = 2s b_q - 2b_{lg,q} + 2\hat{\mathbf{q}}^T \mathbf{d} = 0, \\ \partial F / \partial \mathbf{d} = 2s \hat{\mathbf{q}} - 2\hat{\mathbf{l}}\mathbf{g} + \mathbf{d} = 0. \end{cases} \quad (58)$$

The close-form solution of problem (42) is given by

$$\begin{aligned} s^* &= \frac{b_{lg,q} - \hat{\mathbf{l}}\mathbf{g}^T \hat{\mathbf{q}}}{b_q - \hat{\mathbf{q}}^T \hat{\mathbf{q}}}, \\ \mathbf{d}^* &= \hat{\mathbf{l}}\mathbf{g} - s^* \hat{\mathbf{q}}. \end{aligned} \quad (59)$$

## REFERENCES

- [1] L. Marconi, C. Melchiorri, M. Beetz, D. Pangercic, R. Siegwart, S. Leutenegger, R. Carloni, S. Stramigioli, H. Bruyninckx, P. Doherty, A. Kleiner, V. Lippiello, A. Finzi, B. Siciliano, A. Sala, and N. Tomatis, "The sherpa project: Smart collaboration between humans and ground-aerial robots for improving rescuing activities in alpine environments," in *2012 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2012, pp. 1–4.
- [2] N. Mahdoui, V. Frémont, and E. Natalizio, "Communicating multi-uav system for cooperative slam-based exploration," *Journal of Intelligent & Robotic Systems*, vol. 98, no. 2, pp. 325–343, 2020.
- [3] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 1, pp. 70–85, 2016.
- [4] L. Quan, L. Yin, C. Xu, and F. Gao, "Distributed swarm trajectory optimization for formation flight in dense environments," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 4979–4985.
- [5] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu *et al.*, "Swarm of micro flying robots in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm5954, 2022.
- [6] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics research*. Springer, 2011, pp. 3–19.
- [7] J. Van Den Berg, J. Snape, S. J. Guy, and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3475–3482.

- [8] D. Bareiss and J. Van den Berg, "Reciprocal collision avoidance for robots with linear dynamics using lqr-obstacles," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 3847–3853.
- [9] S. H. Arul and D. Manocha, "Dcad: Decentralized collision avoidance with dynamics constraints for agile quadrotor swarms," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1191–1198, 2020.
- [10] C. E. Luis and A. P. Schoellig, "Trajectory generation for multiagent point-to-point transitions via distributed model predictive control," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 375–382, 2019.
- [11] J. Park, J. Kim, I. Jang, and H. J. Kim, "Efficient multi-agent trajectory planning with feasibility guarantee using relative bernstein polynomial," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 434–440.
- [12] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager, "Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1047–1054, 2017.
- [13] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 5954–5961.
- [14] T. Baca, D. Hert, G. Loianno, M. Saska, and V. Kumar, "Model predictive trajectory tracking and collision avoidance for reliable outdoor deployment of unmanned aerial vehicles," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 6753–6760.
- [15] M. A. Lewis and K.-H. Tan, "High precision formation control of mobile robots using virtual structures," *Autonomous robots*, vol. 4, no. 4, pp. 387–403, 1997.
- [16] D. Panagou and V. Kumar, "Cooperative visibility maintenance for leader–follower formations in obstacle environments," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 831–844, 2014.
- [17] M. C. De Gennaro and A. Jadbabaie, "Formation control for a cooperative multi-agent system using decentralized navigation functions," in *2006 American Control Conference*. IEEE, 2006, pp. 6–pp.
- [18] T. Balch and R. C. Arkin, "Behavior-based formation control for multi-robot teams," *IEEE transactions on robotics and automation*, vol. 14, no. 6, pp. 926–939, 1998.
- [19] Z. Lin, W. Ding, G. Yan, C. Yu, and A. Giua, "Leader–follower formation via complex laplacian," *Automatica*, vol. 49, no. 6, pp. 1900–1906, 2013.
- [20] K. Fathian, S. Safaoui, T. H. Summers, and N. R. Gans, "Robust distributed planar formation control for higher order holonomic and nonholonomic agents," *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 185–205, 2020.
- [21] A. Weinstein, A. Cho, G. Loianno, and V. Kumar, "Visual inertial odometry swarm: An autonomous swarm of vision-based quadrotors," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1801–1807, 2018.
- [22] P. C. Lusk, X. Cai, S. Wadhwan, A. Paris, K. Fathian, and J. P. How, "A distributed pipeline for scalable, deconflicted formation flying," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5213–5220, 2020.
- [23] M. Turpin, N. Michael, and V. Kumar, "Capt: Concurrent assignment and planning of trajectories for multiple robots," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 98–112, 2014.
- [24] D. Morgan, G. P. Subramanian, S.-J. Chung, and F. Y. Hadaegh, "Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1261–1285, 2016.
- [25] S. Agarwal and S. Akella, "Simultaneous optimization of assignments and goal formations for multiple robots," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 6708–6715.
- [26] D. Zhou, Z. Wang, and M. Schwager, "Agile coordination and assistive collision avoidance for quadrotor swarms using virtual structures," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 916–923, 2018.
- [27] Z. Han, L. Wang, and Z. Lin, "Local formation control strategies with undetermined and determined formation scales for co-leader vehicle networks," in *52nd IEEE Conference on Decision and Control*. IEEE, 2013, pp. 7339–7344.
- [28] S. Zhao, "Affine formation maneuver control of multiagent systems," *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4140–4155, 2018.
- [29] S. Zhao, Z. Li, and Z. Ding, "Bearing-only formation tracking control of multiagent systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 11, pp. 4541–4554, 2019.
- [30] S. Zhao and D. Zelazo, "Bearing rigidity theory and its applications for control and estimation of network systems: Life beyond distance rigidity," *IEEE Control Systems Magazine*, vol. 39, no. 2, pp. 66–83, 2019.
- [31] J. Alonso-Mora, E. Montijano, M. Schwager, and D. Rus, "Distributed multi-robot formation control among obstacles: A geometric and optimization approach with consensus," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 5356–5363.
- [32] P. Peng, W. Dong, G. Chen, and X. Zhu, "Obstacle avoidance of resilient uav swarm formation with active sensing system in the dense environment," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 10 529–10 535.
- [33] R. Van Parys and G. Pipeleers, "Distributed model predictive formation control with inter-vehicle collision avoidance," in *2017 11th Asian Control Conference (ASCC)*. IEEE, 2017, pp. 2399–2404.
- [34] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.
- [35] Y. Wang, X. Wen, L. Yin, C. Xu, Y. Cao, and F. Gao, "Certifiably optimal mutual localization with anonymous bearing measurements," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9374–9381, 2022.
- [36] M. Tantardini, F. Ieva, L. Tajoli, and C. Piccardi, "Comparing methods for comparing networks," *Scientific Reports*, vol. 9, 11 2019.
- [37] A. S. Lewis and M. L. Overton, "Nonsmooth optimization via quasi-newton methods," *Mathematical Programming*, vol. 141, no. 1, pp. 135–163, 2013.
- [38] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.*, Shanghai, China, May 2011, pp. 2520–2525.
- [39] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, 2022.
- [40] L. S. Jennings and K. L. Teo, "A computational algorithm for functional inequality constrained optimization problems," *Automatica*, vol. 26, no. 2, pp. 371–375, 1990.
- [41] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*, 3rd ed. USA: Cambridge University Press, 2007.
- [42] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3529–3536, 2019.
- [43] L. Quan, Z. Zhang, X. Zhong, C. Xu, and F. Gao, "Eva-planner: Environmental adaptive quadrotor planning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 398–404.
- [44] F. Gao, Y. Lin, and S. Shen, "Gradient-based online safe trajectory generation for quadrotor flight in complex environments," in *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, Sept 2017.
- [45] F. Båberg and P. Ögren, "Formation obstacle avoidance using rrt and constraint based programming," in *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, 2017, pp. 1–6.
- [46] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *The International Journal of Robotics Research*, vol. 36, no. 9, pp. 1000–1021, 2017.
- [47] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.
- [48] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt\*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.*, Chicago, IL, Sept 2014.
- [49] A. H. Qureshi and Y. Ayaz, "Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments," *Robotics and Autonomous Systems*, vol. 68, pp. 1–11, 2015.
- [50] K.-K. Oh, M.-C. Park, and H.-S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.
- [51] M. Turpin, N. Michael, and V. Kumar, "Trajectory design and control for aggressive formation flight with quadrotors," *Autonomous Robots*, vol. 33, no. 1, pp. 143–156, 2012.

- [52] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on automatic control*, vol. 49, no. 9, pp. 1520–1533, 2004.
- [53] L. Krick, M. E. Broucke, and B. A. Francis, "Stabilisation of infinitesimally rigid formations of multi-robot networks," *International Journal of control*, vol. 82, no. 3, pp. 423–439, 2009.



**Lun Quan** received the B.Eng. degree in control science and engineering in 2019 from Zhejiang University, Hangzhou, China, where he is currently working toward a Ph.D. degree in control science and engineering.

His research interests include motion planning for multi-robot systems, swarm intelligence, and autonomous navigation for unmanned vehicles.



**Longji Yin** received the B.Eng. degree in automation from Zhejiang University, Zhejiang, China, in 2019 and the M.Sc. degree in robotics from Johns Hopkins University, Maryland, U.S. in 2021. In 2022, he worked as a research assistant at Zhejiang University, Zhejiang, China. He is currently working toward a Ph.D. degree in mechanical engineering at the University of Hong Kong, Hong Kong.

His research interests include motion planning and mapping for autonomous aerial robots.



**Tingrui Zhang** received the B.Eng. degree in automotive engineering from Beijing Institute of Technology, Beijing, China, in 2022. He is currently pursuing an M.Phil. degree in control engineering at Zhejiang University, Hangzhou, China.

His research interests include motion planning for autonomous robots and numerical optimization.



**Mingyang Wang** received the B.Eng. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2022. He is currently working toward an M.Phil. degree in control science and engineering from Zhejiang University, Hangzhou, China.

His research interests include motion planning and control for aerobatic flight.



**Ruilin Wang** received the B.Eng. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2023. He is currently working toward an M.Phil. degree in artificial intelligence at Sun Yat-sen University, Zhuhai, China.

His research interests include aerial robots and motion planning.



**Sheng Zhong** received the B.Eng. degree in control science and engineering from Zhejiang University, China, in 2023. He is currently working toward a Ph.D. degree in control science and engineering at Hunan University, Changsha, China.

His research interests include motion estimation based on event cameras.



**Xin Zhou** received the B.Eng. degree in electrical engineering and automation from China University of Mining and Technology, Xuzhou, China, in 2019. He is currently working toward the Ph.D. degree in control engineering from Zhejiang University, Hangzhou, China.

His research interests include motion planning and mapping for aerial swarm robotics.



**Yanjun Cao** received his Ph.D. degree in computer and software engineering from the University of Montreal, Polytechnique Montreal, Canada, in 2020. He is currently an associate researcher at the Huzhou Institute of Zhejiang University, as a PI in the Center of Swarm Navigation. He leads the Field Intelligent Robotics Engineering (FIRE) group of the Field Autonomous System and Computing Lab (FAST Lab).

His research focuses on key challenges in multi-robot systems, such as collaborative localization, autonomous navigation, perception, and communication.



**Chao Xu** received his Ph.D. in Mechanical Engineering from Lehigh University in 2010. He is currently Associate Dean and Professor at the College of Control Science and Engineering, Zhejiang University. He serves as the inaugural Dean of ZJU Huzhou Institute, as well as plays the role of the Managing Editor for *IET Cyber-Systems & Robotics*.

His research expertise is Flying Robotics, Control-theoretic Learning. Prof. Xu has published over 100 papers in international journals, including *Science Robotics* (Cover Paper), *Nature Machine Intelligence* (Cover Paper), etc. Prof. Xu will join the organization committee of the IROS-2025 in Hangzhou.



**Fei Gao** received the Ph.D. degree in electronic and computer engineering from the Hong Kong University of Science and Technology, Hong Kong, in 2019.

He is currently a tenured associate professor at the Department of Control Science and Engineering, Zhejiang University, where he leads the Flying Autonomous Robotics (FAR) group affiliated with the Field Autonomous System and Computing (FAST) Laboratory. His research interests include aerial robots, autonomous navigation, motion planning, optimization, and localization and mapping.