2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Decentralized Trajectory Planning for Formation Flight in Unknown and Dense Environments

Jianxin Zeng, Yaonan Wang, Zhiqiang Miao, Wei He, and Hesheng Wang

***Abstract*— For aerial swarms, formation flight has been applied in various scenes. However, most existing works do not consider balancing the conflicting requirements among keeping formation, keeping the smoothness of trajectories, and obstacle avoidance within the limited time. To address this issue, we propose a decentralized trajectory planning framework for formation flight in unknown and dense environments. To ensure that feasible trajectories can be found within the limited time, the formation optimization problem is decoupled into formation affine transformation and iterative trajectory generation. Firstly, the optimization problem based on affine transformation is designed to obtain the optimal affine transformation sequence, which provides the formation reference of trajectory optimization. Secondly, the iterative optimization framework of trajectory planning is designed, which balances the conflicting requirements of formation, smooth flight, and obstacle avoidance. Besides, to escape the local minima caused by non-convex dense environments, the method of topological path planning is designed to provide distinctive initial solutions for trajectory optimization. Finally, the proposed methods are proven to be effective through the simulations and real-world experiments.**

I. Introduction

Quadrotor swarms are widely employed in collaborative tasks, such as collaborative mapping [1], package transportation [2], target tracking [3], and so on. The above tasks usually require swarms to maintain a specific formation. It is widely believed that the technology of formation flight is vital for swarms to execute complex tasks.

Many scholars have put forward many mature solutions for formation flight, such as virtual structures [4], reactive behaviors [5], and leader-follower [6]. However, the above methods only consider formation flight in an obstacle-free or known environment, which hinders its applications in complex environments. It is urgent to develop the technology of formation flight in complex environments.

This work was supported in part by the National Key Research and Development Program of China (No. 2022YFB3903804), the Natural Science Foundation of China under Grant 62273138, the Science and Technology Innovation Program of Hunan Province under Grant 2021RC3060. (Corresponding author: Zhiqiang Miao.)

Jianxin Zeng, Yaonan Wang, and Zhiqiang Miao are with the Department of Electrical and Information Engineering, Hunan University, Changsha 410082, China, and also with the National Engineering Research Center for Robot Visual Perception and Control, Changsha 410082, China. {zengjianxin, yaonan, miaozhiqiang}@hnu.edu.cn

Wei He is with the School of Intelligence Science and Technology, University of Science and Technology Beiing, Beijing 100083, China, and also with the Key Laboratory of Intelligent Bionic Unmanned Systems, Ministry of Education, University of Science and Technology Beijing, Beijing 100083, China. weihe@ieee.org

Hesheng Wang is with the Department of Automation, Shanghai Jiao Tong University, Shanghai 201100, China. wanghesheng@sjtu.edu.cn

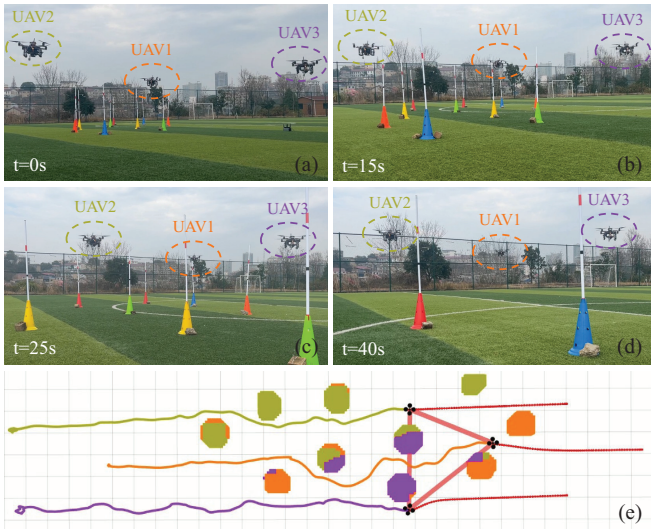


Fig. 1. The swarm in a triangle formation flew in real-world experiments. (a)-(d) are the snapshots of the swarm. (e) is the visualization of the map, formation, planned trajectories, and executed trajectories.

Recently, formation flight in unknown and dense environments has been widely researched [7], [8]. However, most methods do not consider the following challenges brought by the complex environments. (a) The unknown environments put high demands on the real-time trajectory planning of quadrotors. Quadrotors should map through sensors and replan new trajectories in the limited time, otherwise, it may cause collisions. (b) The contradiction among keeping formation, keeping the smoothness of trajectories, and obstacle avoidance is inevitable. However, most methods find the trajectories without considering it. How to balance these three conflicting requirements is different for formation flight. (c) The dense environments make the solution space non-convex and make it difficult to solve high-quality trajectories. Most methods find the trajectories for formation with a topologically equivalent class, which is easy to cause local minima for trajectory planning. It may lead to large formation errors or unsatisfactory smoothness of trajectories.

To settle the above difficulties, we propose a decentralized trajectory planning framework for formation flight in unknown and dense environments, as shown in Fig. 2. Each quadrotor maps, plans trajectories, and shares trajectories among the swarm. To ensure that feasible trajectories can be found within the limited time, we decouple the formation optimization problem into solving formation affine transformation and generating flight trajectories, which lightens the optimization problem. Firstly, we design the optimiza-

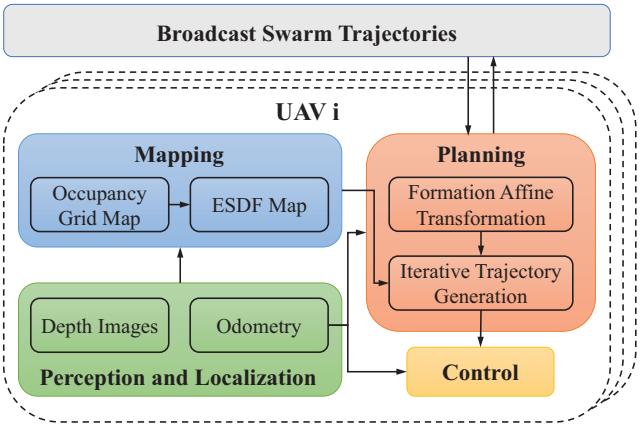


Fig. 2. An overview of our formation method, where the part of planning is the core of our work, the parts of mapping, localization, and control can be realized by general methods.

tion problem of formation affine transformation to find the optimal affine transformation sequence as the formation reference. It allows the formation to scale and rotate, so it can keep formation better with obstacle avoidance compared with the rigid formation. Secondly, to balance the conflicting requirements of formation and smooth flight, we establish an iterative trajectory optimization framework that dynamically adjusts the formation reference when generating flight trajectories. Besides, to escape the local minima for trajectory planning in dense environments, we design a search-based topological path planning method with Uniform Visibility Deformation(UVD) [9], which provides several initial solutions for the trajectory optimization problem. The contributions of this paper are summarized as follows.

1) The optimization problem for formation affine transformation is designed, which enhances the ability to keep formation for swarms in dense environments.
2) An iterative optimization framework of trajectory planning with topological initial solutions is designed, which balances the conflicting requirements of formation, smooth flight, and obstacle avoidance.
3) The simulations and real-world experiments in unknown and dense environments validate the efficiency of our method for formation flight.

II. Related Works

A. Distributed Swarm Trajectory Planning

Optimization-based methods have been extensively adopted in distributed swarm trajectory planning. Zhou et al. [10] proposed a decentralized swarm framework to find feasible trajectories parameterized by B-spline in unknown obstacle-rich environments. Park et al. [11], [12] proposed a swarm planning algorithm that generates safe and feasible trajectory parameterized by Bézier in cluttered environments. The above methods leverage the convex hull property of the trajectory basis to ensure the safety of trajectories. However, it may lead to conservative results. To settle this deficiency, Tordesillas et al. [13] proposed the MINVO basis that has the minimum volume enclosing polynomial curves. Based on MINVO, they proposed a planning framework to generate nonconservative trajectories in dynamic environments.

The above methods do not support optimizing the time allocation of trajectories, which may limit the quality of trajectories. To settle this deficiency, Zhou et al. [14] proposed a distributed swarm planning method with MINCO [15] basis, which supports the spatial–temporal deformation of trajectories. Our method of swarm trajectory planning is on the basis of this work.

B. Formation Flight in Complex Environments

Recently, formation flight in complex environments has been extensively researched. Guo et al. [16] proposed a centralized formation method using Gaussian Processes and factor graphs. However, the centralized architecture inevitably leads to a large amount of calculation. Peng et al. [7] proposed a swarm trajectory global optimal algorithm to ensure a strict consensus on formations and avoid local minima. However, as each quadrotor needs to optimize the trajectories of all quadrotors simultaneously, it is unsuited for large-scale formation. Quan et al. [8], [17] proposed a distributed formation method in dense environments, where the formation error was measured by the differentiable Laplacian-based metric.

Our method for swarm formation in unknown and dense environments is inspired by [8]. Different from [8], we adopt the theory of affine transformation to maintain formation instead of the differentiable Laplacian-based metric, which has less computational complexity and allows constraining formation scale directly in optimization problems.

III. Formation Affine Transformation

In this section, we present the method of formation affine transformation to find the optimal affine transformation sequence, which is used as the formation reference of the trajectory optimization in Section IV-B.

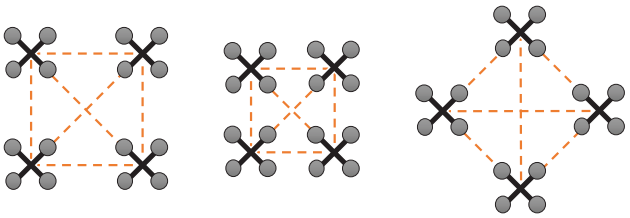
A. Affine Transformation for Formation

A swarm formation of n quadrotors can be described as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, ..., n\}$ is the set of vertices, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. The edge $e_{ij} \in \mathcal{E}$ indicates that quadrotor i and quadrotor j can receive information between each other. In our work, each quadrotor can obtain the information of all other quadrotors, so the graph $\mathcal{G}$ is complete. Let $\boldsymbol{p}_{des} = [\boldsymbol{p}_{des,1}^T, \boldsymbol{p}_{des,2}^T, ..., \boldsymbol{p}_{des,n}^T]^T \in \mathbb{R}^{3n}$ be the desired formation of the swarm, where $\boldsymbol{p}_{des,i} \in \mathbb{R}^3$ is the desired position of quadrotor i.

We adopt affine transformation [18] to model the deformation of the swarm formation. The new position $\boldsymbol{p}_i$ of quadrotor i can be obtained by

$$\boldsymbol{p}_i = \boldsymbol{A}\boldsymbol{p}_{des,i} + \boldsymbol{B} \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{3\times3}$ describes the rotation and scaling of the formation, and $\boldsymbol{B} \in \mathbb{R}^3$ describes the translation of the



(a) Nominal (b) Scaling (c) Rotation

Fig. 3. Illustration of affine transformations. The formations in (b) and (c) are obtained by scaling and rotating from the original formation in (a) respectively.

formation. In our work, we only consider formation scaling and rotation in the x-y plane depicted in Fig. 3, as well as three-dimensional translation, so we define $\boldsymbol{A}$ and $\boldsymbol{B}$ as

$$\boldsymbol{A} \overset{\text{def}}{=} \begin{bmatrix} a & -b & 0 \\ b & a & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{B} \overset{\text{def}}{=} \begin{bmatrix} c \\ d \\ e \end{bmatrix} \tag{2}$$

where $a, b, c, d, e \in \mathbb{R}$. The formation scale s and the horizontal rotation angle θ_z can be obtained by

$$\begin{cases} s = \sqrt{a^2 + b^2} \\ \theta_z = \arctan(b/a) \end{cases} \tag{3}$$

B. Optimal Affine Transformation Sequence

In order to provide the formation reference for subsequent trajectory planning, we define the optimal affine transformation sequence for formation. The affine transformation sequence for quadrotor i in the future period is defined as

$$\boldsymbol{D}_i = \{\boldsymbol{D}_{i,1}, \boldsymbol{D}_{i,2}, ..., \boldsymbol{D}_{i,M_f}\} \tag{4}$$

where $\boldsymbol{D}_{i,k} = \{\boldsymbol{A}_{i,k}, \boldsymbol{B}_{i,k}\}$, $k \in [1, M_f]$, $\boldsymbol{A}_{i,k}$, $\boldsymbol{B}_{i,k}$ are the matrices of the affine transformation for quadrotor i at time stamp k, M_f is the number of sampling points with the fixed time interval.

In order to solve the optimal affine transformation sequence for quadrotor i, we define the optimization problem

$$\min_{\boldsymbol{D}_i} \lambda_f J_f + \lambda_s J_s + \lambda_d J_d + \lambda_c J_c \tag{5}$$

where λ_f, λ_s, λ_d, λ_c are the positive weights. The cost function consists of formation cost J_f, smooth cost J_s, desired scale cost J_d, and scale constraint cost J_c.

1) Formation cost J_f: The cost of formation error is the key to the above optimization problem, which is defined as

$$J_f = \sum_{k=1}^{M_f} \sum_{j=1, j\neq i}^{n} \left\| \boldsymbol{p}_{j,k} - \boldsymbol{p}_{j,k}^b \right\|^2 \tag{6}$$

where $\boldsymbol{p}_{j,k} = \boldsymbol{A}_{i,k}\boldsymbol{p}_{des,j} + \boldsymbol{B}_{i,k}$ denotes the new position of quadrotor j with affine transformation at time stamp k, and $\boldsymbol{p}_{j,k}^b$ is the planned position of quadrotor j at time stamp k from the planned trajectory broadcast by quadrotor j.

2) Smooth cost J_s: To ensure a smooth affine transformation sequence, the elastic band function [19] is adopted as the smooth cost

$$J_s = \sum_{k=1}^{M_f - 1} \left\| \boldsymbol{p}_{i,k+1} - 2\boldsymbol{p}_{i,k} + \boldsymbol{p}_{i,k-1} \right\|^2 \tag{7}$$

where $\boldsymbol{p}_{i,0}$ is the current position of quadrotor i, and $\boldsymbol{p}_{i,k} = \boldsymbol{A}_{i,k}\boldsymbol{p}_{des,i} + \boldsymbol{B}_{i,k}$ is the new position of quadrotor i with affine transformation at time stamp k.

3) Desired scale cost J_d: To ensure that the formation follows the desired scale as much as possible, we design the desired scale cost

$$J_d = \sum_{k=1}^{M_f} \left\| s_k - s_{des} \right\|^2 \tag{8}$$

where s_{des} is the desired formation scale, and s_k is the formation scale at time stamp k obtained by (3) with $\boldsymbol{A}_{i,k}$.

4) Scale constraint cost J_c: Too large formation scale increases the distance between quadrotors and puts pressure on communication, while too small formation scale increases the risk of collision among quadrotors, so the formation scale should be limited. The scale constraint cost is defined as

$$J_c = \sum_{k=1}^{M_f} \psi^3 \left(s_k \right) \tag{9}$$

$$\psi\left(s\right) = \begin{cases} 0, & s_{\min} \leq s \leq s_{\max} \\ s_{\min} - s, & s < s_{\min} \\ s_{\max} - s, & s > s_{\max} \end{cases} \tag{10}$$

where $s_{\max}$ and $s_{\min}$ are the maximum and minimum formation scales respectively.

IV. Trajectory Generation for Formation Flight

In this section, we present the method of iterative trajectory optimization to generate trajectories for the swarm, which balances the conflicting requirements of formation, smooth flight, and obstacle avoidance. The formation reference of the optimization is obtained by Section III-B.

A. Trajectory Representation

We adopt the minimum control effort polynomial trajectory $\mathfrak{T}_{\text{MINCO}}$ [15] as the representation of trajectories for quadrotors, which can be deformed in time and space through optimization. The $\mathfrak{T}_{\text{MINCO}}$ is written as

$$\begin{aligned} \mathfrak{T}_{\text{MINCO}} = \{ & p(t) : [0, T] \mapsto \mathbb{R}^m \mid \boldsymbol{c} = \mathcal{M}(\boldsymbol{q}, \boldsymbol{T}), \\ & \boldsymbol{q} \in \mathbb{R}^{m \times (M-1)}, \boldsymbol{T} \in \mathbb{R}_{>0}^{M} \} \end{aligned} \tag{11}$$

where $p(t)$ is an m-dimensional piece-wise polynomial trajectory with pieces M and degree $N = 2\sigma - 1$, σ is the order of the integrator chain. $\boldsymbol{q} = (q_1, ..., q_{M-1})$ is the intermediate points, and $\boldsymbol{T} = (T_1, ..., T_M)^T$ is the time vector. The trajectory $\mathfrak{T}_{\text{MINCO}}$ is parameterized only by $\boldsymbol{q}$ and $\boldsymbol{T}$. The polynomial coefficient $\boldsymbol{c} = \left(\boldsymbol{c}_1^T, ..., \boldsymbol{c}_M^T\right)^T \in \mathbb{R}^{2M\sigma \times m}$ can be obtained by the mapping $\mathcal{M}(\boldsymbol{q}, \boldsymbol{T})$, where $\boldsymbol{c}_k \in \mathbb{R}^{2\sigma \times m}$ is the coefficient matrix of the k-th piece. The k-th piece of $p(t)$ is represented by

$$p_k(t) = \boldsymbol{c}_k^T \boldsymbol{\beta}(t) \tag{12}$$

where $\boldsymbol{\beta}(t) = \left(1, t, ..., t^N\right)^T$ is the natural basis, $t \in [0, T_k]$.

B. Optimization Problem Formulation

In our work, the trajectory generation is modeled as a multi-objective optimization problem, which includes the requirements of smoothness, safety, aggressiveness, dynamical feasibility, and small formation error. The optimization problem for formation flight is formulated as

$$\min_{\mathbf{c},\mathbf{T}} w_c f_c + w_t f_t + w_f f_f + w_I f_I \tag{13}$$

where w_c, w_t, w_f and w_I are the positive weights. The cost function consists of control energy cost f_c, time cost f_t, formation cost f_f, and penalty cost of soft constraints f_I.

1) Control energy cost f_c: The σ-th (in our work, $\sigma = 3$) control input for the trajectory is written as

$$f_c = \sum_{k=1}^{M} \int_0^{T_k} \left\| p_k^{(\sigma)}(t) \right\|^2 dt \tag{14}$$

2) Time cost f_t: To realize that the swarm can reach the destination quickly, we define the time cost of flight as

$$f_t = \sum_{k=1}^{M} T_k \tag{15}$$

3) Formation cost f_f: In Section III-B, we obtain the optimal affine transformation sequence with the fixed time interval. With the fixed time interval and the reference positions obtained by inputting the optimal affine transformation sequence to (1), the formation reference trajectory p_{af}^* with the representation $\mathfrak{T}_{\text{MINCO}}$ can be constructed. It is used to calculate the formation cost defined as

$$f_f = \sum_{k=1}^{M} \left(\frac{T_k}{\kappa_k} \sum_{j=0}^{\kappa_k} \varpi_j \left(P_d\left(\frac{j}{\kappa_k} T_k + \sum_{q=1}^{k-1} T_q\right) \right)^3 \right) \tag{16}$$

$$P_d(t) = \begin{cases} \left\| p(t) - p_{af}^*(t) \right\|^2, t \le T_{af} \\ 0, t > T_{af} \end{cases} \tag{17}$$

where κ_k is the sample number on the k-th piece trajectory, $[\varpi_0, \varpi_1, ..., \varpi_{\kappa_k - 1}, \varpi_{\kappa_k}] = [1/2, 1, ..., 1, 1/2]$ are the quadrature coefficients, and T_{af} is the overall time of the formation reference trajectory p_{af}^*.

4) Penalty cost of soft constraints f_I: To ensure the safety and dynamical feasibility of quadrotors, the penalty cost of soft constraints is defined as

$$f_I = w_I^o f_I^o + w_I^r f_I^r + w_I^v f_I^v + w_I^a f_I^a \tag{18}$$

where w_I^o, w_I^r, w_I^v and w_I^a are the positive weights. The penalty cost of soft constraints consists of obstacle avoidance penalty f_I^o, swarm reciprocal avoidance penalty f_I^r, velocity feasibility penalty f_I^v, and acceleration feasibility penalty f_I^a, which are defined as follows

$$f_I^\star = \sum_{k=1}^{M} f_k^\star, \ \star \in \{o, r, v, a\} \tag{19}$$

$$f_k^\star = \frac{T_k}{\kappa_k} \sum_{j=0}^{\kappa_k} \varpi_j \left(G_\star\left(\frac{j}{\kappa_k} T_k + \sum_{q=1}^{k-1} T_q\right) \right)^3 \tag{20}$$

Algorithm 1: Topological Paths Generation

Input: initial trajectory τ_{init}, grid map
Output: the set of topological paths S_π^{tp}

$S_\pi^{init} \leftarrow$ **SegmentedPath**(τ_{init}), $S_\pi^{pair} \leftarrow \oslash$;
foreach $\pi_0 \in S_\pi^{init}$ **do**
  $\pi_1 \leftarrow$ **AstarSearch** $(\pi_0(0), \pi_0(1))$;
  $\pi_2 \leftarrow$ **OppositeSearch** (π_0, π_1);
  $S_\pi^{pair} \leftarrow S_\pi^{pair} \cup \{\pi_1, \pi_2\}$;
$S_\pi^{tp} \leftarrow$ **CombinedPaths**(S_π^{pair});
$S_\pi^{tp} \leftarrow$ **ShortPaths**(S_π^{tp})

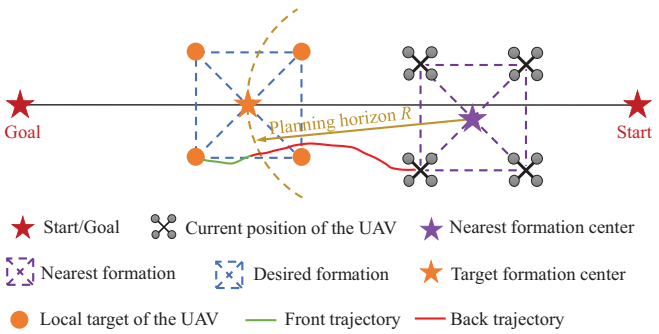


Fig. 4. Illustration of how to obtain the initial trajectory at time stamp η. The initial trajectory consists of the front trajectory and the back trajectory, where the back trajectory is the unexecuted part of the planned trajectory at time stamp $\eta - 1$.

where G_o, G_r, G_v and G_a are the constraint functions of obstacle avoidance, swarm reciprocal avoidance, velocity feasibility, and acceleration feasibility. They are defined as

$$\begin{cases} G_o(t) = \max\left(d_o - d_e(p(t)), 0\right) \\ G_r(t) = \sum_\Phi \max\left((d_r^2 - \|p(t) - p_\phi(t)\|^2), 0\right) \\ G_v(t) = \max\left(\left\|p^{(1)}(t)\right\|^2 - v_m^2, 0\right) \\ G_a(t) = \max\left(\left\|p^{(2)}(t)\right\|^2 - a_m^2, 0\right) \end{cases} \tag{21}$$

where d_o is the threshold of safe distance away from obstacles, $d_e(p(t))$ is the distance between $p(t)$ and its closest obstacle computed by Euclidean Signed Distance Field(ESDF) [19], d_r is the threshold of safe distance among quadrotors, ϕ represents the collection of other quadrotors, v_m and a_m are the maximal velocity and acceleration respectively.

C. Initial Solution Guess

As the optimization problem constructed in Section IV-B is non-convex and non-linear, the initial solution may affect the solving quality and solving time of optimization. Therefore, the initial solution guess is also an important part of the optimization process.

If general methods of path planning such as the A* algorithm are directly used to provide the initial solution, it is easy to cause local minima. Topological path planning can be used to escape local minima by providing distinctive initial paths. Uniform Visibility Deformation(UVD) [9] is efficiently used to capture distinctive initial paths in topological path planning, which is defined as

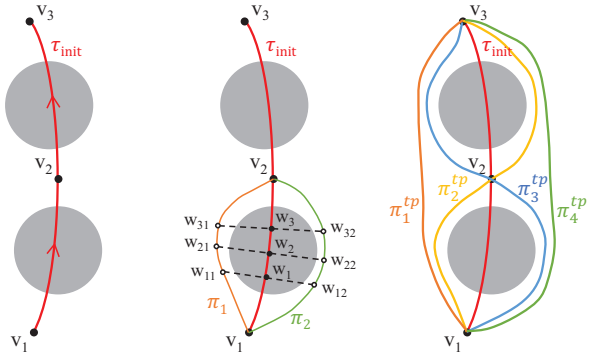


Fig. 5. Illustration of topological path generation. (a) The initial trajectory is segmented by collision checking. (b) Search two topological paths from different sides of the obstacle for each segment of the initial path. (c) The new topological paths are obtained by combining the topological paths of each segment and shorting them.

Definition 1: Two paths $\pi_1(\zeta)$, $\pi_2(\zeta)$ parameterized by $\zeta \in [0,1]$ belong to the same UVD class if satisfying $\pi_1(0) = \pi_2(0)$, $\pi_1(1) = \pi_2(1)$, and the line from $\pi_1(\zeta)$ to $\pi_2(\zeta)$ is collision-free for all ζ.

Inspired by [10], we propose the method of generating topological paths as shown in Algorithm 1. Different from [10] which combines path and trajectory planning, it searches the topological paths and constructs the initial solutions for trajectory optimization. It provides collision-free initial solutions, which ensure the safety of the final optimized trajectory for the non-convex problem.

As the input of Algorithm 1, the initial trajectory τ_{init} at time stamp η is obtained by merging the front trajectory τ_{fr} and the back trajectory τ_{ba}, where the back trajectory is the unexecuted part of the planned trajectory at time stamp $\eta - 1$, as depicted in Fig. 4. In order to obtain the front trajectory τ_{fr}, it is necessary to obtain the target formation center, which is related to the nearest formation. According to (1), the nearest formation is computed by

$$\min_{\boldsymbol{A}_d, \boldsymbol{B}_d} \sum_{i=1}^{n} \left\| \boldsymbol{A}_d \boldsymbol{p}_{des,i} + \boldsymbol{B}_d - \boldsymbol{p}_{cur,i} \right\|^2 \tag{22}$$

where $\boldsymbol{A}_d \in \mathbb{R}^{3\times3}$ describes the rotation and scaling of the nearest formation, $\boldsymbol{B}_d \in \mathbb{R}^3$ is the center of the nearest formation, $\boldsymbol{p}_{des,i} \in \mathbb{R}^3$ is the position of quadrotor i in the desired formation, and $\boldsymbol{p}_{cur,i} \in \mathbb{R}^3$ is the current position of quadrotor i.

The target formation center $\boldsymbol{B}_g \in \mathbb{R}^3$ is on the global trajectory τ_{gl} and within planning horizon R from $\boldsymbol{B}_d$, where the space of τ_{gl} is the line from the start to the goal and the time of τ_{gl} follows the trapezoidal time allocation. The target formation velocity $\boldsymbol{v}_g \in \mathbb{R}^3$ is obtained by τ_{gl} with $\boldsymbol{B}_g$, and the local target position $\boldsymbol{p}_g \in \mathbb{R}^3$ for quadrotor i is obtained by

$$\boldsymbol{p}_g = \boldsymbol{p}_{des,i} + \boldsymbol{B}_g \tag{23}$$

With the local goal $\boldsymbol{p}_g$, the front trajectory τ_{fr} can be obtained, where $\tau_{fr} = [\tau_{fr,x}, \tau_{fr,y}, \tau_{fr,z}]^T$ consists of

Algorithm 2: Iterative Trajectory Optimization

Input: initial guess trajectories set S_τ^{tp}, formation reference trajectory p_{af}^* with total time T_{af}, maximum iterations $iter_{\max}$

Output: the final optimized trajectory τ_f

1 $\tau_f.cost \leftarrow +\infty$;
2 **foreach** $\tau_i^{tp} \in S_\tau^{tp}$ **do**
3 $\tau_i^{sol} \leftarrow$ **SolveOpt**(τ_i^{tp}, p_{af}^*);
4 **if** $\tau_i^{sol}.cost < \tau_f.cost$ **then**
5 $\tau_f \leftarrow \tau_i^{sol}$;
6 $f_{cur} \leftarrow 0, f_{base} \leftarrow w_c^b + w_f^b$;
7 $f_c^b \leftarrow \tau_f.f_c, f_f^b \leftarrow \tau_f.f_f$;
8 $iter \leftarrow 0$;
9 **while** $f_{cur} < f_{base} \wedge iter < iter_{\max}$ **do**
10 $iter \leftarrow iter + 1$;
11 $p_{af}^* \leftarrow \tau_f[0, \min(T_{af}, \tau_f.T)]$;
12 $\tau_f \leftarrow$ **SolveOpt**(τ_f, p_{af}^*);
13 $f_{cur} \leftarrow w_c^b \cdot \tau_f.f_c / f_c^b + w_f^b \cdot \tau_f.f_f / f_f^b$;

three polynomial functions over time [20] acquired by the Pontryagins minimum principle [21]

$$\tau_{fr,\mu}(t) = \frac{1}{6}\alpha_\mu t^3 + \frac{1}{2}\beta_\mu t^2 + v_{s,\mu} + p_{s,\mu} \tag{24}$$

$$\begin{bmatrix}\alpha_\mu \\ \beta_\mu\end{bmatrix} = \frac{1}{(T_{fr})^3}\begin{bmatrix}-12 & 6T_{fr} \\ 6T_{fr} & -2(T_{fr})^2\end{bmatrix}\begin{bmatrix}p_{g,\mu} - p_{s,\mu} - v_{s,\mu}T_{fr} \\ v_{g,\mu} - v_{s,\mu}\end{bmatrix} \tag{25}$$

where subscript $\mu \in \{x, y, z\}$, $\boldsymbol{p}_s = [p_{s,x}, p_{s,y}, p_{s,z}]^T$ and $\boldsymbol{v}_s = [v_{s,x}, v_{s,y}, v_{s,z}]^T$ are the terminal position and velocity of the back trajectory, $\boldsymbol{p}_g = [p_{g,x}, p_{g,y}, p_{g,z}]^T$ and $\boldsymbol{v}_g = [v_{g,x}, v_{g,y}, v_{g,z}]^T$ are the local target position and velocity, and T_{fr} is the total time of τ_{fr} from trapezoidal time allocation

$$T_{fr} = \begin{cases} \sqrt{d_{sg}/a_m}, & v_m^2/a_m < d_{sg} \\ (v_m^2 + d_{sg}a_m)/(v_m a_m), & v_m^2/a_m \geq d_{sg} \end{cases} \tag{26}$$

where d_{sg} is the Euclidean distance from $\boldsymbol{p}_s$ to $\boldsymbol{p}_g$.

Algorithm 1 firstly segments the initial trajectory by collision checking in Line 1 if the initial trajectory is unsafe, as depicted in Fig. 5(a). Secondly, search two topological paths from different sides of the obstacle for each segment of the initial path in Line 2-5, as depicted in Fig. 5(b). One side path is obtained by A*, and the anchor point w_{i1} at the obstacle surface are assigned for the point w_i on the initial path. Search from w_i along vector $\overrightarrow{w_{i1}w_i}$, find the opposite point w_{i2} outside the obstacle, and then connect opposite points by A* to obtain the opposite path. Finally, obtain new topological paths by combining the topological paths of each segment and shorten them by the method of [9] in Line 6-7, as depicted in Fig. 5(c).

Only if the initial trajectory is unsafe, Algorithm 1 is used to obtain topological paths and construct initial guess solutions. Otherwise, the initial trajectory is directly as the initial guess solution. The topological paths are divided into

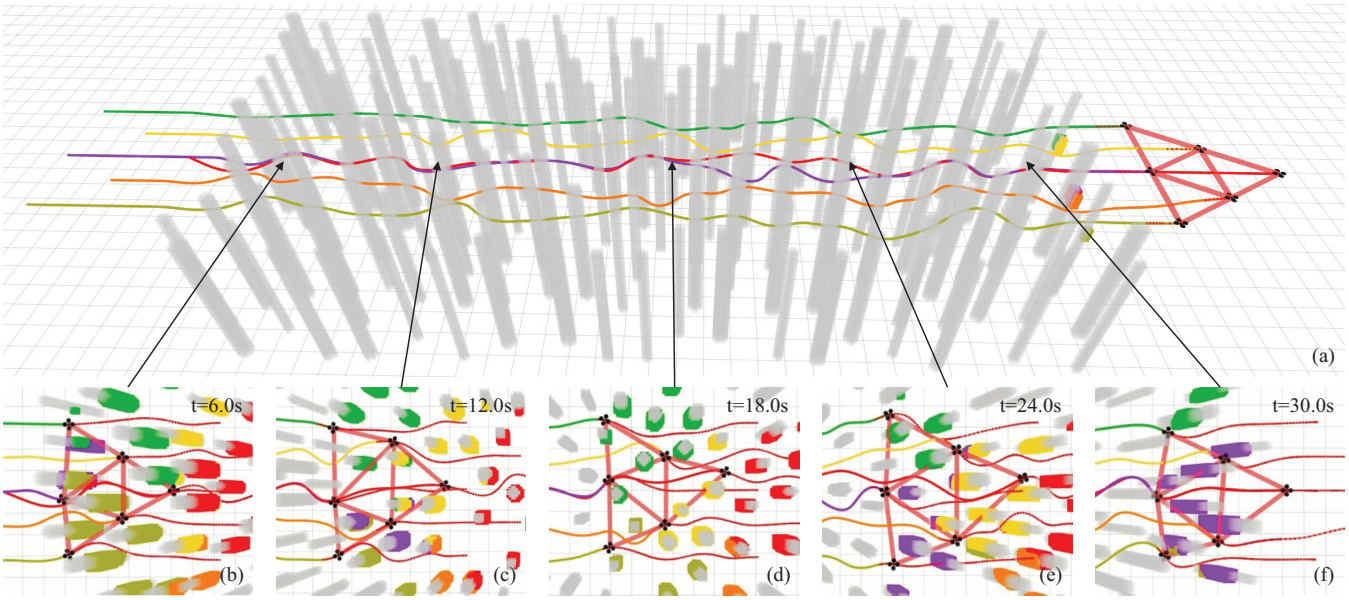


Fig. 6. A triangular formation consisting of six quadrotors flew from the left side to the right side in unknown and dense environments (0.3 obs./m^2). (a) The executed trajectories of the swarm shown by colored curves. (b)-(f) The snapshots of swarm formation flight at different times.

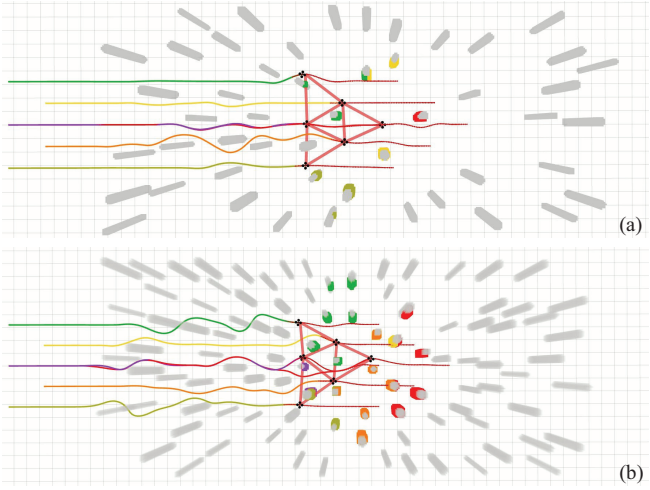


Fig. 7. The snapshots of a triangular formation consisting of six quadrotors flying in unknown and dense environments. (a) The snapshot of the formation in the map with density 0.1 obs./m^2. (b) The snapshot of the formation in the map with density 0.2 obs./m^2.

M segments at equal distances d_c, and the trapezoidal time allocation is applied to guess the time t_{gi} of the i-th segment. With d_c and t_{gi}, the topological paths are converted into the initial guess trajectories with the representation $\mathfrak{T}_{\text{MINCO}}$. The initial guess trajectories are used as initial solutions to the optimization problem in Section IV-B.

D. Iterative Optimization

To improve computational efficiency, we decouple formation optimization into solving the optimal affine transformation sequence and solving flight trajectories. However, it may lead to a new problem. As optimal affine transformation sequence is obtained without considering obstacle avoidance, the sequence may be unreasonable, which affects the quality of trajectory planning. To address this issue, we establish an iterative optimization framework that dynamically adjusts the affine transformation sequence when solving the trajectory optimization problem.

The method of iterative trajectory optimization is shown in Algorithm 2. Line 2-5 solve the optimal trajectory τ_f with different initial guess trajectories, where $\tau.cost$ expresses the cost of trajectory τ obtained by (13), and the function **SolveOpt**(τ, p^*_{af}) solves the optimization problem (13) with the initial guess solution τ and the formation reference trajectory p^*_{af}. Line 6-8 initialize the parameters for the iterative optimization in Line 9-13. The iterative optimization is used to deal with the problem that the control energy cost of the optimized trajectory may be too large because of the unreasonable formation reference trajectory, where w^b_c, w^b_f are the given positive weights satisfying $w^b_c + w^b_f = 1$, $\tau.f_c$, $\tau.f_f$ are the control energy cost of τ defined in (14) and formation cost of τ defined in (16). In order to balance the costs of control energy and formation errors, $w^b_f = 3w^b_c$ is set in our work. The formation reference trajectory p^*_{af} is dynamically updated to the part of the trajectory τ_f with time $t \in [0, \min(T_{af}, \tau_f.T)]$ by Line 11, where $\tau_f.T$ is the overall time of trajectory τ_f.

V. Experiments

The method we proposed was implemented in C++ with ROS2 as the swarm communication architecture and the open-source library LBFGS-Lite[1] as the optimization solves. Simulation experiments and real-world experiments were designed to prove the efficiency of our method.

[1]https://github.com/ZJU-FAST-Lab/LBFGS-Lite

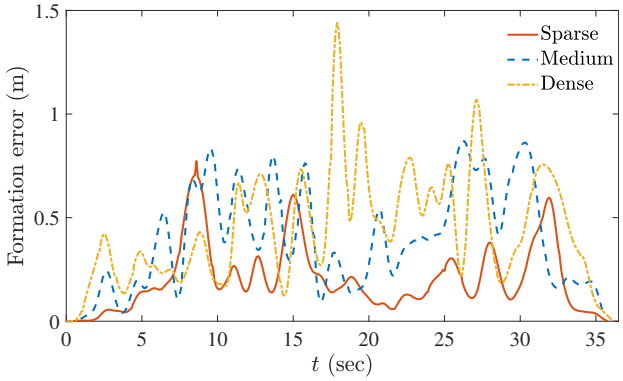


Fig. 8. The formation errors over time in three maps with different densities.

TABLE I

COMPARISONS OF FORMATION PLANNING METHODS.

Density (obs./m^2)	Method	e_{dist}(m) Mean/Max	E_c (m^2/s^5)	T_{total} (s)	t_{plan} (ms)
Sparse 0.1	Our$_1$	0.25/0.79	248.18	36.04	**2.11**
	Our$_2$	**0.21/0.71**	101.81	**35.86**	5.63
	[8]	0.28/1.45	**42.27**	43.34	16.65
Medium 0.2	Our$_1$	**0.39**/1.04	471.69	36.62	**2.14**
	Our$_2$	0.41/**0.82**	145.81	**36.02**	5.76
	[8]	0.52/1.81	**84.38**	45.71	16.79
Dense 0.3	Our$_1$	0.57/**1.27**	717.06	36.65	**2.23**
	Our$_2$	**0.44**/1.43	240.72	**36.43**	5.87
	[8]	0.68/2.27	**180.95**	46.93	17.24

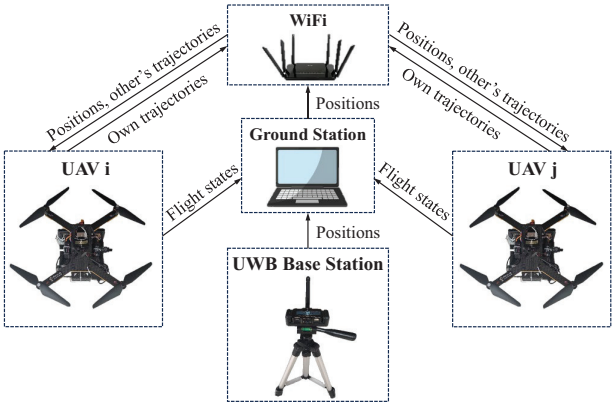
A. Simulations

The simulation experiments were designed on a Ubuntu 22.04 laptop(CPU: i7-13700H, RAM: 16 GB). We designed six quadrotors in a triangular formation flying from one side of the maps($34\text{m} \times 15\text{m} \times 5\text{m}$) to the other side with maximum velocity limit $v_m = 1.5$ m/s and maximum acceleration limit $a_m = 6.0$ m/s^2. The simulations were run on three maps with different obstacle densities, as depicted in Fig. 6 and Fig. 7, where Fig. 6 shows complete trajectories of formation in the dense obstacle map(0.3 obs./m^2).

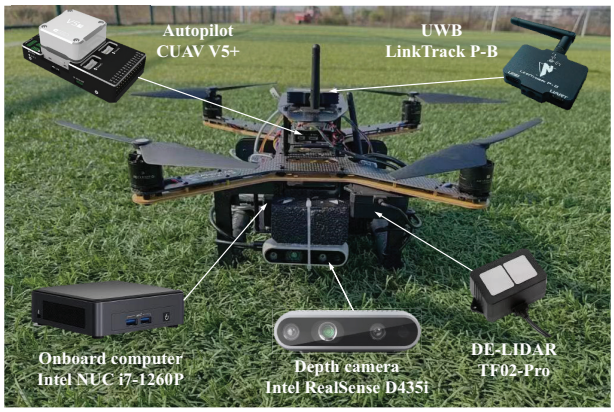
We compared our work with Quan's method [8], as summarized in Table I, where the starting point and goal of the formation center are the same, and the Euclidean distance between them is 42m. In Table I, the methods "our$_1$" and "our$_2$" are our methods without iterative optimization(Algorithm 2 is only run for Lines 1-5) and with iterative optimization respectively, and e_{dist}, E_c, T_{total}, t_{plan} respectively express the formation error, control energy, flight time and average replanning time. Inspired by [22], the formation error e_{dist} for n quadrotors with time variable t is defined according to (1), and it describes the gap between the actual formation of the swarm and the nearest affine transformation formation.

$$e_{dist}(t) = \min_{\boldsymbol{A}(t),\boldsymbol{B}(t)} \sum_{i=1}^{n} \left\| \boldsymbol{A}(t)\boldsymbol{p}_{des,i} + \boldsymbol{B}(t) - \boldsymbol{p}_i(t) \right\|^2 \tag{27}$$

where $\boldsymbol{p}_i(t)$ is the actual position of quadrotor i. The mean



(a)



(b)

Fig. 9. The experimental setup of the formation system. (a) The overall hardware structure of the formation system. (b) Platform of the quadrotor.

of formation errors is written as

$$\bar{e}_{dist} = \frac{1}{T_{total}} \int_0^{T_{total}} e_{dist}(t)dt \tag{28}$$

As shown in Table I, the iterative optimization can take a little computing time to balance the costs of control energy and formation errors. Compared with Quan's method [8], our method with iterative optimization has smaller formation errors, shorter flight time, and shorter replanning time. The formation errors of the method "our$_2$" with three maps are shown in Fig. 8. The simulations proved the efficiency of our method in unknown and dense environments.

B. Real-world Experiments

Apart from simulations, we designed the real-world experiments to verify the efficiency of our method. The main experimental setup of the formation system is shown in Fig. 9. The Intel NUC microcomputer (CPU: i7-1260P, RAM: 16 GB) is used to run software programs and drive hardware devices. The depth camera D435i provides depth sensing, and the Ultrawide bandwidth(UWB) on each quadrotor and four UWB base stations are used for location. A laptop(CPU: i7-13700H, RAM: 16 GB) is used to monitor the status of quadrotors as the ground station.

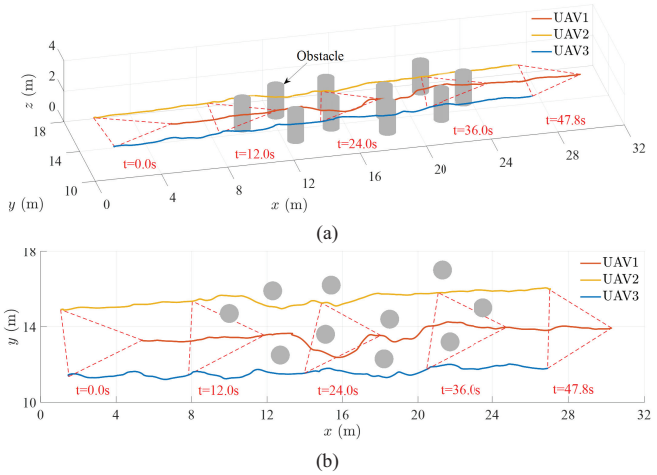


Fig. 10. The executed trajectories and the formations of real-world experiments in three-dimensional view (a) and two-dimensional view (b).

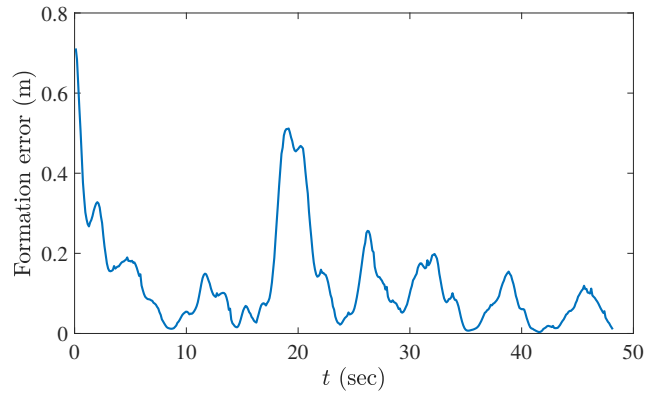


Fig. 11. The formation errors over time in real-world experiments.

Three quadrotors flew in a triangle formation without mapping in advance, as shown in Fig. 1. The maximum velocity limit was set to $v_m = 0.6$ m/s, and the maximum acceleration limit was set to $a_m = 4.0$ m/s^2. The executed trajectories and formations of three quadrotors are shown in Fig. 10, and the formation errors are shown in Fig. 11. The real-world experiments proved that our method enables the swarm to keep formation with obstacle avoidance.

VI. Conclusion

In this paper, we proposed a decentralized planning framework to generate feasible trajectories for formation flight in unknown and dense environments. We decoupled the formation optimization problem into formation affine transformation and iterative trajectory generation, which ensures finding feasible trajectories within the limited time. The formation affine transformation allows the formation to scale and rotate, which is beneficial for maintaining formation with obstacle avoidance. The iterative trajectory generation balances the conflicting requirements of formation and smooth flight. The simulations and real-world experiments proved the efficiency of our method. In the future, we will expand our work to more complex scenes, such as mazes and scenarios with limited communication.

References

[1] N. Mahdoui, V. Frémont, and E. Natalizio, "Communicating multi-UAV system for cooperative SLAM-based exploration," J. Intell. Robot. Syst., vol. 98, no. 2, pp. 325–343, 2020.

[2] Y. Ping, M. Wang, J. Qi, C. Wu and J. Guo, "Collaborative control based on payload-leading for the multi-quadrotor transportation systems," in Proc. IEEE Int. Conf. Robot. Automat., 2023, pp. 5304-5309.

[3] L. Yin, F. Zhu, Y. Ren, F. Kong and F. Zhang, "Decentralized swarm trajectory generation for LiDAR-based aerial tracking in cluttered environments," in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2023, pp. 9285-9292.

[4] M. A. Lewis and K.-H. Tan, "High precision formation control of mobile robots using virtual structures," Auton. Robots, vol. 4, no. 4, pp. 387–403, 1997.

[5] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," IEEE Trans. Robot. Autom., vol. 14, no. 6, pp. 926–939, Dec. 1998.

[6] D. Panagou and V. Kumar, "Cooperative visibility maintenance for leader–follower formations in obstacle environments," IEEE Trans. Robot., vol. 30, no. 4, pp. 831–844, Aug. 2014.

[7] 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 Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2022, pp. 10529–10535.

[8] L. Quan, L. Yin, C. Xu, and F. Gao, "Distributed swarm trajectory optimization for formation flight in dense environments," in Proc. Int. Conf. Robot. Automat., 2022, pp. 4979–4985.

[9] B. Zhou, J. Pan, F. Gao, and S. Shen, "RAPTOR: Robust and perception aware trajectory replanning for quadrotor fast flight," IEEE Trans. Robot., vol. 37, no. 6, pp. 1992–2009, Dec. 2021.

[10] X. Zhou, J. Zhu, H. Zhou, C. Xu, and F. Gao, "Ego-swarm: A fully autonomous and decentralized quadrotor swarm system in cluttered environments," in Proc. IEEE Int. Conf. Robot. Automat., 2021, pp. 4101–4107.

[11] J. Park, D. Kim, G. C. Kim, D. Oh, and H. J. Kim, "Online distributed trajectory planning for quadrotor swarm with feasibility guarantee using linear safe corridor," IEEE Robot. Automat. Lett., vol. 7, no. 2, pp. 4869–4876, Apr. 2022.

[12] J. Park, I. Jang and H. J. Kim, "Decentralized deadlock-free trajectory planning for quadrotor swarm in obstacle-rich environments," in Proc. IEEE Int. Conf. Robot. Automat, 2023, pp. 1428-1434.

[13] J. Tordesillas and J. P. How, "MADER: Trajectory planner in multiagent and dynamic environments," IEEE Trans. Robot., vol. 38, no. 1, pp. 463–476, Feb. 2022.

[14] X. Zhou et al., "Swarm of micro flying robots in the wild," Sci. Robot., vol. 7, no. 66, May 2022, Art. no. eabm5954.

[15] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," IEEE Trans. Robot., vol. 38, no. 5, pp. 3259–3278, Oct. 2022.

[16] S. Guo, B. Liu, S. Zhang, J. Guo and C. Wang, "Continuous-time gaussian process trajectory generation for multi-robot formation via probabilistic inference," in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2021, pp. 9247-9253.

[17] L. Quan et al., "Robust and efficient trajectory planning for formation flight in dense environments," IEEE Trans. Robot., vol. 39, no. 6, pp. 4785-4804, Dec. 2023.

[18] S. Zhao, "Affine formation maneuver control of multiagent systems," IEEE Trans. Autom. Control, vol. 63, no. 12, pp. 4140-4155, Dec. 2018.

[19] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," IEEE Robot. Automat. Lett., vol. 4, no. 4, pp. 3529–3536, Oct. 2019.

[20] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in Proc. Int. Conf. Robot. Automat., 2011, pp. 2520–2525.

[21] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," IEEE Trans. Robot., vol. 31, no. 6, pp. 1294–1310, Dec. 2015.

[22] P. C. Lusk, X. Cai, S. Wadhwania, A. Paris, K. Fathian, and J. P. How, "A distributed pipeline for scalable, deconflicted formation flying," IEEE Robot. Automat. Lett., vol. 5, no. 4, pp. 5213–5220, 2020.