2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 1-5, 2023. Detroit, USA

Decentralized Swarm Trajectory Generation for LiDAR-based Aerial Tracking in Cluttered Environments

Longji Yin*, Fangcheng Zhu*, Yunfan Ren*, Fanze Kong, and Fu Zhang

***Abstract*— Aerial tracking with multiple unmanned aerial vehicles (UAVs) has wide potential in various applications. However, the existing works for swarm tracking typically lack the capability of maintaining high target visibility in cluttered environments. To address this deficiency, we present a decentralized planner that maximizes target visibility while ensuring collision-free maneuvers for swarm tracking. In this paper, each drone's tracking performance is first analyzed by a decentralized kinodynamic searching front-end, which renders an optimal guiding path to initialize safe flight corridors and visible sectors. Afterwards, a polynomial trajectory satisfying the corridor constraints is generated by a spatial-temporal optimizer. Inter-vehicle collision and occlusion avoidance are also incorporated into the optimization objectives. The advantages of our methods are verified by extensive benchmark comparisons against other cutting-edge works. Integrated with an autonomous LiDAR-based swarm system, the proposed planner demonstrates its efficiency and robustness in real-world experiments with unknown cluttered surroundings.**

I. Introduction

Boosted by the advancement of onboard sensing and computing technologies, autonomous aerial tracking with UAVs has been widely applied in cinematography, surveillance, and industrial inspection. In recent literature, target tracking using multiple UAVs has drawn increasing attention. Compared to a single UAV, swarm tracking possesses larger system redundancy and team cooperation capability. However, there still exist three main technical challenges in swarm tracking:

1) **Safety:** The trajectory of each tracker must be collision free w.r.t. surrounding obstacles, target, and other teammate trackers in the swarm. Each trajectory should also respect the dynamic constraints of the robot.
2) **Visibility:** The target should be visible during the flight. Each tracker needs to prevent the target from being occluded by obstacles and other teammate robots.
3) **Portability:** The trajectory planner should be decentralized and computationally efficient, which is crucial for the application in real-world tracking tasks.

In practice, the movements of the tracker are constrained by collision avoidance and dynamic feasibility, which may cause the target visibility to be compromised. And the portability imposes more strict limitations on the planner implementation. Thus, how to systematically trade off the safety and visibility in a portable manner is the key to achieving high-visibility swarm tracking in cluttered scenes. Despite the recent progress in multiple-UAV tracking, the method that

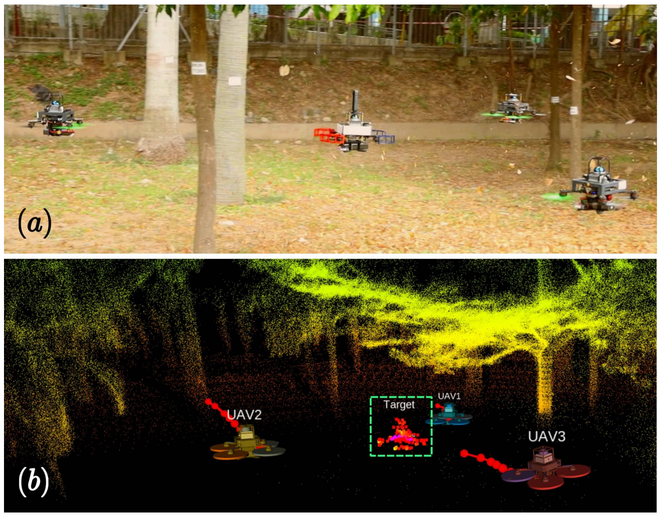

Fig. 1: **(a)** A swarm of three drones is tracking a manually operated target drone in the middle. **(b)** Visualization of the point cloud map and planned trajectories. The points measured on the target are marked in a dashed box. More details can be found in the video at https://youtu.be/04-ls0PHkuU.

can concurrently address the three challenges is still lacking in existing works. Several studies [1, 2] chase a target as a leader by leveraging constant leader-follower formations. Their frameworks can retain safe displacements between tracker and target but neglect the visibility requirements. Instead of prescribing formations, a group of works [3, 4] coordinate the swarm by searching visibility-aware paths using a centralized front-end. However, their centralized schemes make the system vulnerable to single-point-of-failure. In many aerial tracking studies [5]–[8], both collision and occlusion avoidance rely on constructing Signed Distance Fields (SDFs), which could cause an extra computational burden. Therefore, an efficient swarm tracking planner that tackles all three challenges is rare in the literature.

To bridge this gap, we introduce a hierarchical swarm trajectory generation framework that handles all the aforementioned challenges systematically. In this work, we use a 360° LiDAR sensor as the onboard sensor. Compared to traditional cameras [1, 8, 9], one 360° LiDAR is able to perform target sensing and environmental perception simultaneously, which endows the swarm with more possible tracking strategies. In our planning pipeline, a decentralized kinodynamic search is first carried out as the front-end to generate guiding paths for each agent. Front-end costs that quantify the tracking quality are formulated to select the best motion primitives for the task objectives. Subsequently, a safe flight corridor is constructed along the searched path. For the back-end, we present a spatial-temporal optimiza-

L. Yin, F. Zhu, Y. Ren, F. Kong, and F. Zhang are with the Department of Mechanical Engineering, University of Hong Kong {ljyin, zhufc, renyf, kongfz}@connect.hku.hk, fuzhang@hku.hk.
* **Equal contribution**.

2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-9190-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/IROS55552.2023.10341567

tion module that optimizes the trajectories over collision penalty, tracking distance, and visibility costs jointly. The advantage of our method is verified by extensive benchmark comparisons and ablation studies. Afterwards, we integrate our framework into a real decentralized LiDAR-based swarm system, where drone trajectories are broadcast to others via wireless communication. Finally, we validate the practicality and efficiency of our tracking planner in a real-world forest.

The contributions of this paper are summarized as:

1) We propose a decentralized kinodynamic front-end to initiate guiding paths that maximize the tracking quality (e.g., target visibility) for back-end optimizations.
2) We propose a decentralized spatial-temporal trajectory optimizer for swarm tracking, which considers tracker safety and target visibility concurrently.
3) We present a decentralized swarm tracking system that demonstrates the efficiency and robustness of our method in real-world aerial tracking tasks.

II. Related works

A. Single-UAV Target Tracking

Optimization-based trajectory generation for single-drone aerial tracking is extensively investigated in the literature. Penin *et al.* [10] directly evaluate the target visibility in the camera image space and incorporate the constraints into a nonlinear programming problem. However, they assume that all the obstacles are ellipsoid-shaped, which is not applicable to unstructured scenes. Han *et al.* [11] proposed a tracking planner containing a spatial-temporal back-end optimizer and kinodynamic front-end searcher. But they only include tracking distance as the objective and neglect target visibility. Bonatti *et al.* [5] present an aerial cinematography planning framework that jointly takes obstacle avoidance, target occlusion, and motion smoothness into account. Nevertheless, their optimization relies on the gradients from a SDF, which is time-consuming to construct for large scenes. In [8], Wang *et al.* design a visibility cost that penalizes the intersection area of obstacles and the field of view (FOV). However, the proposed constraint is too hard to be satisfied in dense environments. Besides, it is not compatible with omnidirectional sensors like 360° LiDAR, since the intersection area could always exist in the FOV. In [12], Ji *et al.* design a more generic method to enforce target visibility. They generate a sequence of sector-shaped visible regions for the target by raycasting and then constrain the tracker's trajectory in those sectors via spatial-temporal optimization. In this work, we adopt the visibility formulation in [12] and extend it to the settings of multiple-UAV tracking.

B. Multiple-UAV Target Tracking

Multiple-UAV target tracking has drawn increasing attention in the recent literature [1]–[4, 13]. Zhou *et al.* [1] demonstrate a swarm of drones following a target in accordance with a constant leader-follower formation. In their work, the target position observed by one tracker is broadcast to others so as to improve the overall occlusion resistance. For the tracking task in [2], Tallamraju *et al.* exploit the formation configuration to minimize the fused uncertainty of target estimation, and utilize model predictive controllers (MPC) to handle both formation maintenance and obstacle avoidance. Although the drones in [1, 2] can communicate the target observations to temporarily survive the occlusion, their planners lack the motion strategy to actively reduce the risk of visibility loss. To address this issue, Nageli *et al.* [13] proposed an MPC-based scheme that calculates a horizon plane according to the obstacle to split the visible and invisible areas. However, their strategy still relies on modeling the obstacles as ellipsoids. Bucker *et al.* [3] discretize the space around the target into cells and rate each cell based on occlusion avoidance performance. Then centralized greedy searching is conducted to generate motion sequences with the lowest visibility loss for each drone. However, their greedy searching has a predefined priority, which could lead to sub-optimality of the result. Ho *et al.* [4] introduce a framework that produces swarm trajectories for aerial 3D reconstruction. In their work, the follower formation is allowed to rotate around the target, and a centralized front-end planner searches the optimal sequence of rotation angles via dynamic programming. Nevertheless, the strict assumption of fixed formation can be easily violated in cluttered scenes. And the centralized system is vulnerable w.r.t. the failure of central nodes. In contrast to [4], our work is fully decentralized and encodes the formation as a soft cost for trade-off, but still possesses collaborative maneuverability while preserving high target visibility.

III. Kinodynamic Searching

Our kinodynamic front-end searches for a safe and feasible reference path by expanding motion primitives with a discretized control input space, which is similar to other hybrid A* schemes for quadrotor navigation [9, 11, 14]. Rather than deriving a path to the terminal state with minimal control effort, our searching method evaluates each primitive based on its tracking performance, so as to achieve high consistency of the front-end and the overall task objectives.

A. Node Expansion

The state vector $x \in \mathbb{R}^6$ of a tracker drone is comprised of drone position $p = [p_x, p_y, p_z]^T$ and velocity $v = [v_x, v_y, v_z]^T$. For each dimension, we use acceleration as the control input and discretize the input space as $u_d = \{-a_{max}, 0, a_{max}\}$, where a_{max} indicates the acceleration limit. Before the searching process, a prediction module renders a series of future target positions according to a time step δT and a prediction horizon T_p. The target sequence is

$$\mathcal{Q}_{target} = \{q_k \in \mathbb{R}^3 \mid 0 \le k \le N_p,\ t_k = k \cdot \delta T\}, \qquad (1)$$

where N_p is the total prediction number and t_k is the timestamp corresponding to the target position at the k^{th} step from now. In the front-end, we expand motion primitives of the tracker drone directly using the prediction interval δT, so that the timestamp t_k of every new node x_n of the tracker drone is aligned with the target prediction q_k. With the above control inputs and time steps, the motion primitives are expanded using double-integrator dynamics.

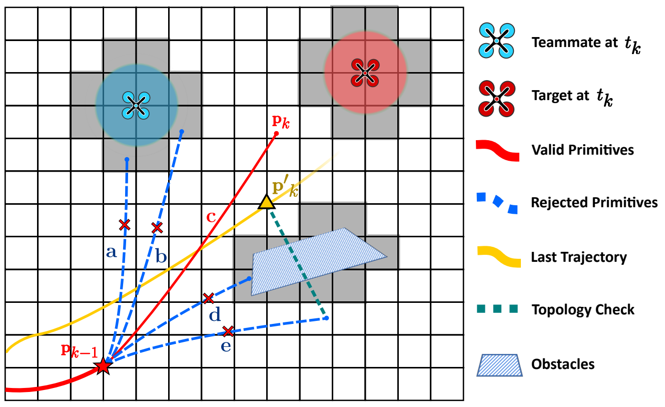

Fig. 2: An illustration of primitive rejection. **a** and **b** are rejected by inter-vehicle safety check, **d** is rejected by obstacles, **e** fails the topology check.

B. Primitive Rejection

To facilitate the task requirements and accelerate the searching, we design pruning strategies to reject the infeasible nodes in the primitive expansion process timely. After each expansion step, all resultant nodes are checked in terms of obstacle avoidance, dynamic feasibility, inter-vehicle safety, and topological consistency. For inter-vehicle collision avoidance, we first query the current positions of teammate drones on the broadcast trajectories. Then a node is considered safe if the distances between the node and all its teammates are larger than a clearance r_s.

During swarm flights, each drone is expected to maintain a coherent trajectory homotopy, since rapid changes in topological structures could raise the risk of inter-vehicle collision, especially in the presence of communication latency. Thus, for a proximal time horizon T_s, the topological consistency is checked between the current node and the corresponding node on the trajectory planned in the last replan cycle, to prevent abrupt changes. Let p'_k denote the position at time $t_k < T_s$ on the last trajectory, then the node p_k being expanded is considered inconsistent if line $\overline{p_k\, p'_k}$ is not collision-free. Fig. 2 shows the rejection mechanism for the primitives expanded from t_{k-1} to t_k, where $t_k < T_s$.

C. Cost Functions

After the pruning process, every remaining node x_n is assigned a cost g_n as a coarse assessment of its tracking quality. We formulate the front-end costs as follows.

1) ***Obstacle Occlusion:*** This cost is introduced to preserve target visibility against static obstacle occlusions. Given a target position q_k seen from a tracker positioned at p_k, the target is considered visible if the line of sight(LOS) $\overline{q_k\, p_k}$ hits no obstacles (i.e., collision-free). This definition matches with the principle of LiDAR measurement.

In practice, to avoid binary cost, we penalize the occlusion using a measure of voxel occupancy along the line of sight. Letting L_s denote LOS segment $\overline{q_k\, p_k}$, we have $L_s(\tau) = \tau\, q_k + (1-\tau)\, p_k,\ 0 \le \tau \le 1$. Then the cost is given by

$$g_{vis} = \int_0^1 \mathcal{V}_{occ}(L_s(\tau))\, d\tau, \tag{2}$$

where $\mathcal{V}_{occ}$ returns the voxel occupancy at position $L_s(\tau)$.

2) ***Tracking Distance:*** Inspired by [12], the vertical distance d_z (along z-axis) and the horizontal distance d_h (in x-y plane) between target q_k and tracker p_k are regulated separately using different costs. For vertical distance, a quadratic cost is given as $g_{vrt} = d_z^2$. The purpose is to align the height of the tracker with the target, so that the target can be contained in the vertical FOV of LiDAR at all time. We expect the horizontal distance to satisfy $d_{lb} \le d_h \le d_{ub}$, where d_{ub} and d_{lb} are the upper and lower bounds of desired tracking distance. Then the horizontal cost is defined as

$$g_{hoz} = \begin{cases} 5\,(d_{lb} - d_h)^3, & d_h < d_{lb}, \\ 0, & d_{lb} \le d_h \le d_{ub}, \\ (d_h - d_{ub})^2 / 2, & d_h > d_{ub}. \end{cases} \tag{3}$$

We put slighter penalty on larger tracking distances, since the LiDAR sensing range is much longer than d_{ub}. But the lower bound d_{lb} is strictly enforced to secure target safety.

3) ***Angular Separation:*** Two requirements on swarm angular separation are proposed. Firstly, a requisite for teammate occlusion avoidance is enforced. It demands the trackers to keep an angular distance between each other about the target position, and the distance should be no less than a clearance θ_c, so that the line of sight between tracker and target will not be blocked by teammates. Secondly, apart from the minimal clearance θ_c, we also expect the swarm to maintain an optimal formation with an evenly spaced angular separation $2\pi/N$, where N is the swarm size. With this equidistant formation, the swarm can make full use of the vicinity around the target, which provides the trackers with the largest angular space to respond to any adverse situations, such as occlusion. Fig. 3(a) shows the twofold requirements.

Let $p_{j,k}$ denote the position of the j^{th} drone at time t_k. With the same timestamp, the i^{th} drone locates at $p_{i,k}$ and the target is at q_k. Then for the mutual occlusion cost, we penalize the angles less than clearance θ_c directly by

$$g_{moc} = \sum_{j=1,\, j\ne i}^{N} \max\{0,\ \theta_c - \mathrm{acos}(\frac{e_{i,k} \cdot e_{j,k}}{\|e_{i,k}\|\, \|e_{j,k}\|})\}, \tag{4}$$

where $e_{i,k} = p_{i,k} - q_k$, $e_{j,k} = p_{j,k} - q_k$. This occlusion cost serves as the basic requirement for angular distancing. Cost g_{frm} for the optimal formation has the same expression as cost g_{moc}, but the clearance term θ_c is replaced by $2\pi/N$.

Synthesizing all the terms, we have node cost g_n as

$$g_n = [g_{vis},\, g_{vrt},\, g_{hoz},\, g_{moc},\, g_{frm}] \cdot w, \tag{5}$$

where w is the weight vector to trade off cost priorities. The searching terminates when one primitive reaches the target prediction horizon T_p. In implementation, we employ the remaining expansion time $h_n = T_p - t_k$ as a heuristic function to speed up the searching process.

D. Flight Corridors and Visible Sectors

We adopt the efficient method in [15] for safe flight corridor generation along the searched path. The corridor is composed of connected polyhedra, each is denoted as

$$\mathcal{P} = \{x \in \mathbb{R}^3 \mid \mathbf{A}_c\, x \le b_c\}. \tag{6}$$

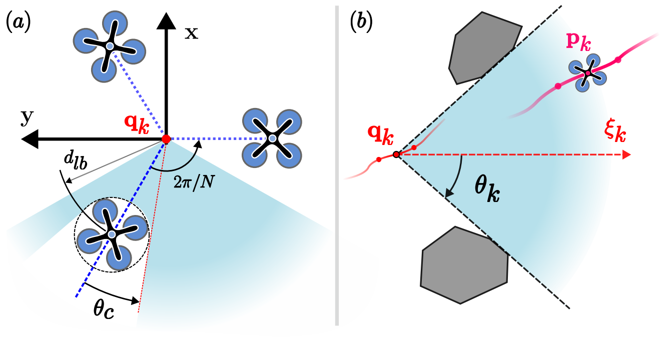

Fig. 3: **(a)** An illustration of the minimal angular clearance θ_c between drones and the expected formation separation $2\pi/N$, $\mathbf{q}_k$ denotes the target position. **(b)** The visible sector constructed in III-D.

For each pair of target position q_k and path waypoint p_k, a visible region as depicted in Fig. 3(b) is constructed following [12]. Each visible sector can be written as

$$\mathcal{V} = \{x \in \mathbb{R}^3 \mid \langle x - q_k, \xi_k \rangle \le \theta_k\}, \tag{7}$$

where $\langle \, , \rangle$ denotes the angle between two vectors.

IV. Spatial-Temporal Trajectory Optimization

A. Trajectory Representation

In this work, we use the MINCO representation [16], a minimum control effort polynomial trajectory class to conduct spatial-temporal deformation of the flat-output trajectory

$$\begin{aligned}\Xi_{MINCO} = &\{p(t) : [0, T_\Sigma] \mapsto \mathbb{R}^m \mid \mathbf{c} = \mathcal{C}(\mathbf{q}, \mathbf{T}),\\ &\mathbf{q} \in \mathbb{R}^{m(M-1)}, \mathbf{T} \in \mathbb{R}^M_{>0}\},\end{aligned} \tag{8}$$

where $\mathbf{c} = (c_1^T, \cdots, c_M^T)^T$ is the polynomial coefficient, $\mathbf{q} = (q_1, \cdots, q_{M-1})$ the intermediate points, $\mathbf{T} = (T_1, \cdots, T_M)^T$ the time vector, $\mathcal{C}(\mathbf{q}, \mathbf{T})$ is the linear-complexity parameter mapping from [16], and T_Σ is the total trajectory duration. A m-dimensional M-piece trajectory $p(t)$ is defined as $p(t) = p_i(t - t_{i-1}), \quad \forall t \in [t_{i-1}, t_i)$, and the i^{th} piece trajectory is represented by a $N = 5$ degree polynomial $p_i(t) = c_i^T \beta(t), \quad \forall t \in [0, T_i]$. MINCO trajectories are all compactly parameterized by $\mathbf{q}$ and $\mathbf{T}$. With mapping $\mathcal{C}(\mathbf{q}, \mathbf{T})$, the cost of the polynomial trajectories in Ξ_{MINCO} can be evaluated by $\mathbf{q}$ and $\mathbf{T}$ by $\mathcal{J}(\mathbf{q}, \mathbf{T}) = \mathcal{F}(\mathbf{c}, \mathbf{T}) = \mathcal{F}(\mathcal{C}(\mathbf{q}, \mathbf{T}), \mathbf{T})$. Thus, we can conduct trajectory optimization over objective J using the gradients $\partial\mathcal{J}/\partial\mathbf{q}$ and $\partial\mathcal{J}/\partial\mathbf{T}$, which are propagated from gradients $\partial\mathcal{F}/\partial\mathbf{c}$ and $\partial\mathcal{F}/\partial\mathbf{T}$ accordingly.

We formulate the trajectory generation as an unconstrained optimization problem. The optimization objective is given by

$$\min_{\mathbf{q}, \mathbf{T}} \mathcal{J} = \lambda_c \mathcal{J}_c + \sum_* \lambda_* \mathcal{J}_* + \sum_\star \lambda_\star \mathcal{J}_\star + \sum_\diamond \lambda_\diamond \mathcal{J}_\diamond, \tag{9}$$

where $\mathcal{J}_c$ is the cost for minimizing control effort and time duration of the trajectory:

$$\mathcal{J}_c = \int_0^{T_\Sigma} \parallel p_i^{(3)}(t) \parallel^2 dt + \rho \, T_\Sigma. \tag{10}$$

Aside from control effort and time cost, objective $\mathcal{J}$ also contains the terms that penalize the violations of inequality constraints $\psi(p(t), ..., p^{(3)}(t)) \le 0$. The constraints can be divided into three categories. In the first category, constraints ψ_* act for the basic necessities for single-UAV navigation, and the corresponding violation penalties $\mathcal{J}_*$ are integrated using relative time on the trajectory pieces. For the second category, constraints $\psi_\star$ and penalties $\mathcal{J}_\star$ are concerned with single-UAV target tracking requirements, which are evaluated at the absolute timestamps of target predictions. The third category is associated with swarm coordination, where the constraints and penalties are marked as $\psi_\diamond$ and $\mathcal{J}_\diamond$. Coefficients λ represent the weights for cost trade-offs.

B. Single-UAV Navigation Constraints

The constraints ψ_* for navigation are evaluated on relative times. To efficiently compute violations, we transform the continuous constraints into finite ones using constraint transcription. Penalties $\mathcal{J}_*$ are numerically integrated by sampling the trajectories evenly with time step T_i/κ_i, where κ_i denotes the sample number of the i^{th} piece. So we have

$$\mathcal{J}_* = \sum_i^M \frac{T_i}{\kappa_i} \sum_{j=0}^{\kappa_i} \bar{\omega}_j \max\{\psi_*(p_i(t)), 0\}^3, \tag{11}$$

where $t = (j/\kappa_i)T_i$ refers to the relative time sampled on the i^{th} piece. Integral coefficients $(\bar{\omega}_0, \bar{\omega}_1, \cdots, \bar{\omega}_{\kappa_i - 1}, \bar{\omega}_{\kappa_i}) = (1/2, 1, \cdots, 1, 1/2)$ follow the trapezoidal rule. The gradients of $\mathcal{J}_*$ w.r.t. c_i and T_i can be derived by chain rule

$$\frac{\partial \mathcal{J}_*}{\partial c_i} = \frac{\partial \mathcal{J}_*}{\partial \psi_*} \frac{\partial \psi_*}{\partial p_i} \frac{\partial p_i}{\partial c_i}, \tag{12}$$

$$\frac{\partial \mathcal{J}_*}{\partial T_i} = \frac{\mathcal{J}_*}{T_i} + \frac{\partial \mathcal{J}_*}{\partial \psi_*} \frac{\partial \psi_*}{\partial p_i} \frac{\partial p_i}{\partial t} \frac{\partial t}{\partial T_i}, \tag{13}$$

$$\frac{\partial p_i}{\partial c_i} = \beta(t), \quad \frac{\partial p_i}{\partial t} = \dot{p}_i(t), \quad \frac{\partial t}{\partial T_i} = j/\kappa_i, \tag{14}$$

and $\partial\psi_*/\partial p_i$ is determined by the formulations of ψ_*. The constraints ψ_* are detailed as follows.

1) Obstacle Avoidance: The trajectory of each tracker should be confined in the polyhedral safe corridors generated in Sec.III-D. The corridor constraint ψ_o is written as

$$\psi_o = \mathbf{A}_c \, p_i(t) - b_c. \tag{15}$$

2) Dynamic Feasibility: We limit the maximum amplitudes of the tracker's velocity and acceleration. The dynamic constraints ψ_{vel} and ψ_{acc} are given by

$$\psi_{vel} = \dot{p}_i(t)^2 - v_{max}^2, \; \psi_{acc} = \ddot{p}_i(t)^2 - a_{max}^2, \tag{16}$$

where v_{max} and a_{max} are the upper bounds.

C. Single-UAV Target Tracking Constraints

To impose the single-UAV tracking constraints $\psi_\star$, violation penalties $\mathcal{J}_\star$ should be evaluated at absolute time t_k, which is the k^{th} timestamp of target predictions. We have

$$\mathcal{J}_\star = \sum_{k=1}^{N_p} \delta T \max\{\psi_\star(p(t_k)), 0\}^3, \tag{17}$$

where δT and N_p are the time interval and total number of predictions. Assume that t_k is located on the i^{th} piece of the trajectory, then the corresponding relative time t of t_k on the i^{th} piece becomes $t = t_k - \sum_{l=1}^{i-1} T_l$, where T_l denotes the preceding piece duration. Note that the formulation of t here brings in the gradient dependence on all T_l with $1 \le l \le i$. Thus, the gradients of $\mathcal{J}_\star$ are derived as

$$\frac{\partial \mathcal{J}_\star}{\partial c_i} = \frac{\partial \mathcal{J}_\star}{\partial p}\frac{\partial p}{\partial c_i}, \quad \frac{\partial \mathcal{J}_\star}{\partial T_l} = \frac{\partial \mathcal{J}_\star}{\partial p}\frac{\partial p}{\partial t}\frac{\partial t}{\partial T_l}, \tag{18}$$

$$\frac{\partial p}{\partial c_i} = \beta(t_k - \sum_{l=1}^{i-1} T_l), \quad \frac{\partial t}{\partial T_l} = \begin{cases} 0, & l = i, \\ -1, & l < i. \end{cases} \tag{19}$$

The single-UAV tracking constraints $\psi_\star$ are listed as follows.

1) Tracking Distance: The constraint ψ_{dis} for distance keeping is in the same form as the front-end in Sec.III-C:

$$\psi_{dis} = g_{hoz} + g_{vrt}. \tag{20}$$

2) Obstacle Occlusion: To confine the trajectory in the visible sectors generated from Sec.III-D following [12], we use the occlusion avoidance constraint ψ_{occ}:

$$\psi_{occ} = \cos\theta_k - \frac{(p(t_k) - q_k)\cdot \xi_k}{\|p(t_k) - q_k\|}. \tag{21}$$

D. Swarm Coordination Constraints

With the communication network, each drone receives teammates' trajectories to formulate swarm constraints $\psi_\diamond$ and optimizes its own trajectory accordingly. In this category, the penalties $\mathcal{J}_\diamond$ consist of $\mathcal{J}_{rep}$ for inter-vehicle collision avoidance and $\mathcal{J}_{moc}$ for swarm angular distancing.

1) Swarm Reciprocal Clearance: Since this continuous constraint involves the teammate trajectories, we need to use a relative time $t = (j/\kappa_i)T_i$ for the ego trajectory and the corresponding absolute timestamp τ to query the teammate positions on received trajectories, where $\tau = \sum_{l=1}^{i-1} T_l + (j/\kappa_i)T_i$. To enforce swarm safety, we expect each drone to maintain the distance clearance r_s to all other teammates at all time. For a swarm of N drones, let $p_\phi(\tau)$ denote the trajectory from the teammate drone ϕ and ψ_{r_ϕ} denote the clearance constraint associated with teammate ϕ, then the violation penalty $\mathcal{J}_{rep}$ is formulated as

$$\mathcal{J}_{rep} = \sum_{i}^{M}\frac{T_i}{\kappa_i}\sum_{j=0}^{\kappa_i}\bar{\omega}_j\sum_{\phi}^{N}\max\{\psi_{r_\phi}(p_i(t), p_\phi(\tau)), 0\}^3, \tag{22}$$

$$\psi_{r_\phi}(p_i(t), p_\phi(\tau)) = r_s^2 - \|p_i(t) - p_\phi(\tau)\|^2. \tag{23}$$

Note that term τ introduces ψ_{r_ϕ} the gradient dependence on its preceding pieces T_l through $p_\phi(\tau)$. So temporal gradient

$$\frac{\partial \psi_{r_\phi}}{\partial T_l} = \frac{\partial \psi_{r_\phi}}{\partial p_i}\frac{\partial p_i}{\partial t}\frac{\partial t}{\partial T_l} + \frac{\partial \psi_{r_\phi}}{\partial p_\phi}\frac{\partial p_\phi}{\partial \tau}\frac{\partial \tau}{\partial T_l}, \tag{24}$$

$$\frac{\partial t}{\partial T_l} = \begin{cases} \frac{j}{\kappa_i}, & l = i, \\ 0, & l < i, \end{cases} \quad \frac{\partial \tau}{\partial T_l} = \begin{cases} \frac{j}{\kappa_i}, & l = i, \\ 1, & l < i. \end{cases} \tag{25}$$

Other components of gradients $\partial\mathcal{J}_{rep}/\partial c_i$ and $\partial\mathcal{J}_{rep}/\partial T_l$ are in the same form as in Eq. 12 and Eq. 13.

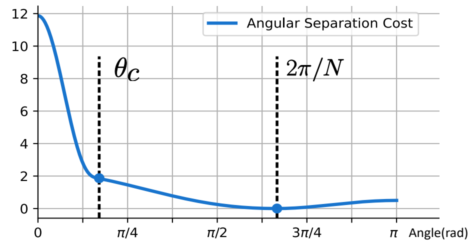

Fig. 4: The profile of ψ_{s_ϕ} w.r.t. the angular distance.

2) Swarm Angular Separation: To handle the twofold angular distancing requirements described in Sec.III-C, a C^2-continuous cost is formulated for the back-end. Since target positions are entailed in the constraints, the angular separation penalty $\mathcal{J}_{sep}$ needs to be evaluated at absolute prediction timestamps t_k. Let ψ_{s_ϕ} denote the angular distancing cost exerted by teammate drone ϕ, then we have

$$\mathcal{J}_{sep} = \sum_{k=1}^{N_p}\delta T\sum_{\phi}^{N}\max\{\psi_{s_\phi}(p(t_k), p_\phi(t_k)), 0\}^3. \tag{26}$$

Let η_ϕ denote the cosine value of the angular distance from teammate ϕ, we have

$$\eta_\phi = \frac{(p(t_k) - q_k)\cdot(p_\phi(t_k) - q_k)}{\|p(t_k) - q_k\|\|p_\phi(t_k) - q_k\|}. \tag{27}$$

If the mutual occlusion clearance θ_c is not violated, the constraint ψ_{s_ϕ} is only induced by the desired formation:

$$\psi_{s_\phi} = (\eta_\phi - \cos\frac{2\pi}{N})^2. \tag{28}$$

When the angular distance is less than θ_c, i.e., $\eta_\phi > \cos\theta_c$, then ψ_{s_ϕ} is augmented by a mutual occlusion penalty to repulse the tracker away from teammate ϕ:

$$\psi_{s_\phi} = (\eta_\phi - \cos\frac{2\pi}{N})^2 + \rho_s(\eta_\phi - \cos\theta_c)^3, \tag{29}$$

where ρ_s is a tunable weight. Unlike the reciprocal collision penalty $\mathcal{J}_{rep}$, the teammate position $p_\phi(t_k)$ in this $\mathcal{J}_{sep}$ produces no extra gradients. It is because both teammate and ego trajectories in $\mathcal{J}_{sep}$ are queried using the same fixed absolute time t_k, thus the teammate position $p_\phi(t_k)$ is a constant throughout the optimization iterations. Therefore, all the gradients of $\mathcal{J}_{sep}$ have identical forms as Eq. 18.

V. Benchmark and Ablation Study

A. Benchmark Comparisons

To validate the advantage of our method, benchmark comparisons are conducted with other cutting-edge swarm tracking planners. The proposed method is compared with Zhou's work [1] and Ho's work [4]. To benchmark the

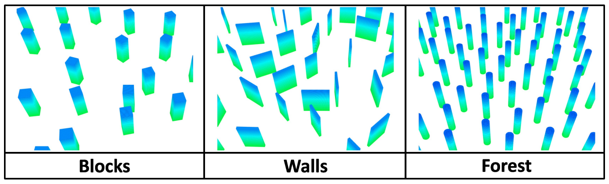

Fig. 5: The random maps for benchmark comparison.

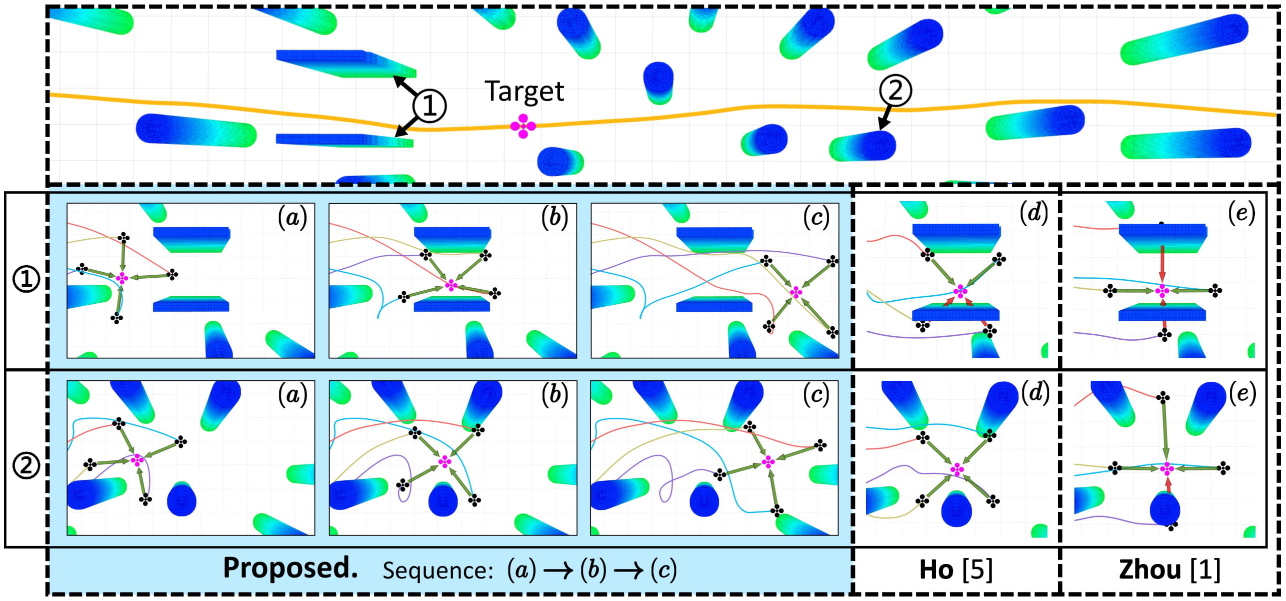

Fig. 6: The benchmark results in simulation. **Top**: The orange trajectory is executed by target. Two areas ① and ② are extracted for illustration. **Middle:** Snapshots of the trackers and target in area ①. Subfigures (a) to (c) shows the swarm behaviour of our method. Benchmark results are shown in (d) and (e). The green arrows indicate the occlusion-free LOS between trackers and target, whereas the red ones refer to occlusion. **Bottom:** Snapshots in area ②.

performance fairly, we simulate three swarms chasing one same target drone simultaneously. Each swarm contains four trackers and runs one compared planner in a cluttered area of $40m \times 40m$ size. Each method is tested with three random maps generated with different obstacle types and densities as shown in Fig. 5. The optimization problems are solved by LBFGS-Lite[1]. The maximum velocity of the target drone is set to $1.5m/s$. The tracker has a vertical FOV of $60°$ and a maximum velocity of $3m/s$. The lower and upper bounds of tracking distance are set to $1.7m$ and $2.3m$.

The tracking performance is evaluated over four metrics: average target visibility (ϑ_{avg}), visibility of the worst case (ϑ_{wst}), time ratio of full visibility (γ_{vis}), and tracking distance (d_{avg}). A tracker is considered losing the target if the line of sight is blocked by an obstacle, or blocked by a teammate drone, or out of vertical FOV. The swarm visibility $\vartheta(t)$ is defined as the count of trackers that are not losing the target at time t. For a tracking task with duration T_b, the average visibility ϑ_{avg} is calculated by

$$\vartheta_{avg} = \frac{1}{T_b} \int_0^{T_b} \vartheta(t) dt. \tag{30}$$

Practically, this metric ϑ_{avg} is numerically integrated by sampling the tracking task with a time interval of 0.2s. The metric ϑ_{wst} indicates the worst swarm visibility sampled in the whole task. Let T_{vis} denote the total time duration in which the target is visible to all trackers in the swarm, then metric γ_{vis} refers to the ratio of T_{vis} to T_b. The tracking distance d_{avg} is averaged in the same manner as ϑ_{avg}.

The results are summarized in Tab.I. As seen from the table, the proposed method outperforms other works in terms of ϑ_{avg}, ϑ_{wst} and γ_{vis} on all three maps with satisfactory

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

TABLE I: Benchmark Results

Scenario	Metric / Method	ϑ_{avg}	ϑ_{wst}	$\gamma_{vis}(\%)$	$d_{avg}(m)$
Blocks	Zhou [1]	3.89	3	89.7	1.98
	Ho [4]	3.97	3	97.6	2.04
	Ours	**4.00**	4	**100.0**	1.94
Walls	Zhou [1]	3.70	2	72.1	2.06
	Ho [4]	3.79	3	79.4	2.03
	Ours	**3.97**	3	**96.9**	1.92
Forest	Zhou [1]	3.64	2	66.9	2.05
	Ho [4]	3.70	2	72.1	2.02
	Ours	**3.96**	3	**96.0**	1.91

distances d_{avg}. Zhou's planner [1] involves no active occlusion avoidance, so their method has the lowest visibility. Ho *et al.* [4] assign a rotatable formation template to the swarm and search for the best rotation to reduce occlusion. However, their assumption of fixed formation is still overly strict for dense maps. For the scenario in Fig. 6, our planner agilely rotates and deforms the swarm to keep the target fully visible, whereas occlusion occurs to other two methods.

B. Ablation Study

Ablation studies are carried out to further verify the necessity of the proposed kinodynamic front-end and the strength of our angular formation costs. For the front-end ablation, we first replace our kinodynamic searching with a vanilla A* front-end presented in [12]. This ablated planner is marked as **Variant 1**. Secondly, in both front-end and back-end, we remove all cost terms that enforce the uniform angular separation $2\pi/N$. This ablated planner that cancels the formation constraints is marked as **Variant 2**. Finally, the full planner as the **Proposed** and Zhou's work [1] as the **Baseline** are also simulated for comparison.

All variants are tested in the forest maps generated with six levels of densities, varying from $1/40\ tree/m^2$(sparse) to $1/12\ tree/m^2$(very dense). Each tree is a cylinder with

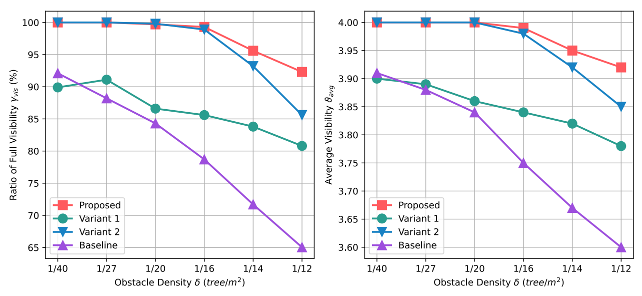

Fig. 7: The ablation results of metrics γ_{vis} and ϑ_{avg}.

a diameter of $0.9m$ and a height of $4m$. In each trial, four trackers collaborate to chase a target drone that traverses the forests with a maximum velocity of $1.5m/s$. The results are shown in Fig. 7. The vanilla A* searching in **Variant 1** greedily minimizes the path distance but neglects the requirements from swarm coordination, which leads to frequent mutual occlusion and hence low swarm visibility even in sparse scenes. **Variant 2** uses the proposed front-end to deconflict the inter-vehicle interference among the swarm and significantly boosts the tracking performance. However, since it cancels the formation, all trackers tend to gather behind the target when traversing the obstacles, which yet raises the risk of mutual occlusion. To address this issue, the **Proposed** uses formation as a reference distribution to disperse the crowded trackers, and thus reduces the blockage in dense areas. The ablation study proves the effectiveness of our front-end and the formation strategy.

VI. Real-world Experiments

A. System Architecture

We integrate the proposed method with a decentralized LiDAR-based aerial swarm system. The swarm system consists of three autonomous drones, each one is mounted with a Livox Mid360 LiDAR, an onboard computer Intel NUC with CPU i7-12700, and a PixHawk flight controller. The swarm is localized by a decentralized swarm LiDAR-inertial odometry (Swarm-LIO) in [17], which is developed based on FAST-LIO2 [18] and provides $100Hz$ state estimation and $25Hz$ point cloud for each tracker. The time offset and extrinsic between LiDAR and IMU are calibrated by [19]. All drones are controlled using an on-manifold MPC [20]. The swarm is connected by a UDP wireless network. Apart from the ego states and mutual observations that are required by Swarm-LIO, the planned trajectories are also transmitted to all other teammates via the UDP network. After receiving a trajectory from a teammate, the agent synchronizes the timestamps and then transforms the trajectory from the teammate's frame into its own global frame using the swarm extrinsic provided by Swarm-LIO. This process is called trajectory alignment.

In experiments, the swarm tracks a manually operated drone as the target. The target is attached with reflective tapes for identification and runs Swarm-LIO for self-localization. Informed of the velocity from the target via broadcast, each drone tracks the target through a constant velocity Error State Kalman Filter (ESKF) that fuses the high-reflectivity

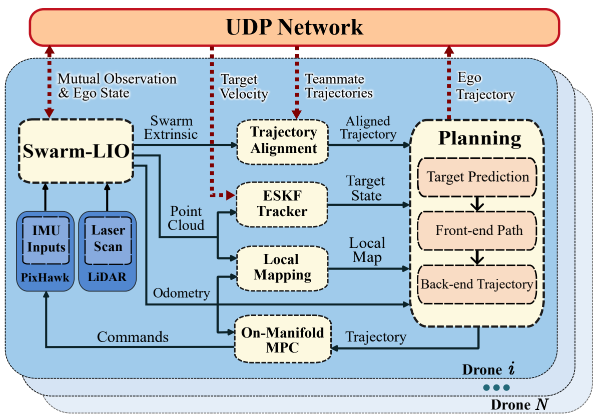

Fig. 8: The diagram of system architecture.

measurements from LiDAR. The target predictions are extrapolated using a fixed velocity model. In implementation, the point measured on the target and teammates are removed in Swarm-LIO map, since these objects are non-stationary. The complete system architecture is depicted in Fig. 8.

B. Real-world Results

To demonstrate the practicality and robustness of our method, we test the swarm tracking system in an unknown real-world cluttered environment. The maximum velocity of the manually controlled target is set to $1.5m/s$. The horizon of target prediction is $2.0s$, and the tracking distance bounds d_{lb} and d_{ub} are $1.4m$ and $2m$. The tracking trajectory is replanned at $10Hz$. In the experiment, three autonomous drones successfully tracked the target in a forest without any collision or occlusion. Fig. 9 shows that the trackers agilely deform and rotate the swarm distribution to avoid potential occlusions. Our supplementary video[2] displays the whole tracking process. The average onboard computation time of our method is detailed in Tab.II.

TABLE II: Onboard Computation Time (ms)

t_{search}	t_{corridor}	t_{sector}	t_{optimize}	t_{total}
2.11	3.09	0.03	1.73	6.96

VII. Conclusions and Future Work

In this paper, we presented a decentralized spatial-temporal trajectory generation framework for LiDAR-based swarm tracking in cluttered environments. Experiments in both simulation and real-world confirmed the efficiency of the proposed framework. It's worth noting that in the experiments the target velocity was directly leveraged from broadcasting to promote the reliability of ESKF tracking and avoid misdetection since this work mainly focused on decentralized trajectory planning. However, a robust onboard LiDAR-based detector for moving objects is also a necessity for tracking systems, which is taken as our future work.

VIII. Acknowledgement

This work was supported by the Information Science Academy of China Electronics Technology Group Corporation (ISA CETC), and the project number is 200010756.

[2]https://youtu.be/04-ls0PHkuU

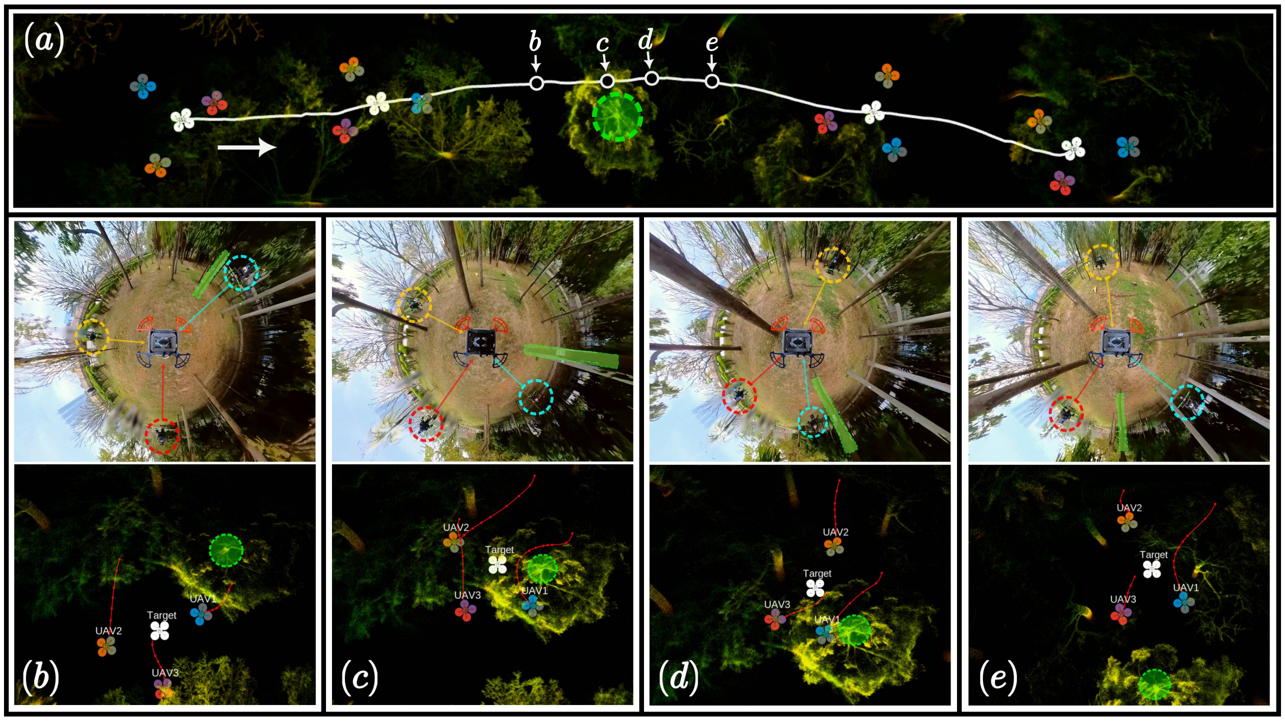

Fig. 9: Snapshots of real-world experiments. Three autonomous drones are tracking a target drone in the middle. **(a)** The white curve represents the target trajectory. An obstacle tree is marked by a green dashed circle. Four keyframes from b to e are selected near the obstacle for further illustration. **(b) - (e)**: The snapshots correspond to the four keyframes in (a). each subfigure contains an image captured by a 360° action camera (top) and the visualization in RViz (bottom). The same obstacle is highlighted by green markers in subfigures (b) - (e). The red curves in RViz are the planned trajectories of the tracker drones. This figure depicts how the swarm maneuvers to avoid the occlusion caused by the obstacle.

References

[1] 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.

[2] R. Tallamraju, E. Price, R. Ludwig, K. Karlapalem, H. H. Bülthoff, M. J. Black, and A. Ahmad, "Active perception based formation control for multiple aerial vehicles," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4491–4498, 2019.

[3] A. Bucker, R. Bonatti, and S. Scherer, "Do you see what i see? coordinating multiple aerial cameras for robot cinematography," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7972–7979.

[4] C. Ho, A. Jong, H. Freeman, R. Rao, R. Bonatti, and S. Scherer, "3d human reconstruction in the wild with collaborative aerial cameras," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5263–5269.

[5] R. Bonatti, W. Wang, C. Ho, A. Ahuja, M. Gschwindt, E. Camci, E. Kayacan, S. Choudhury, and S. Scherer, "Autonomous aerial cinematography in unstructured environments with learned artistic decision-making," *Journal of Field Robotics*, vol. 37, no. 4, pp. 606–641, 2020.

[6] B. Jeon, Y. Lee, and H. J. Kim, "Integrated motion planner for real-time aerial videography with a drone in a dense environment," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1243–1249.

[7] B. F. Jeon and H. J. Kim, "Online trajectory generation of a mav for chasing a moving target in 3d dense environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1115–1121.

[8] Q. Wang, Y. Gao, J. Ji, C. Xu, and F. Gao, "Visibility-aware trajectory optimization with application to aerial tracking," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5249–5256.

[9] Z. Zhang, Y. Zhong, J. Guo, Q. Wang, C. Xu, and F. Gao, "Auto filmer: Autonomous aerial videography under human interaction," *IEEE Robotics and Automation Letters*, 2022.

[10] B. Penin, P. R. Giordano, and F. Chaumette, "Vision-based reactive planning for aggressive target tracking while avoiding collisions and occlusions," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3725–3732, 2018.

[11] Z. Han, R. Zhang, N. Pan, C. Xu, and F. Gao, "Fast-tracker: A robust aerial system for tracking agile target in cluttered environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 328–334.

[12] J. Ji, N. Pan, C. Xu, and F. Gao, "Elastic tracker: A spatio-temporal trajectory planner for flexible aerial tracking," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 47–53.

[13] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges, "Real-time planning for automated multi-view drone cinematography," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–10, 2017.

[14] 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.

[15] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, 2017.

[16] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259–3278, 2022.

[17] F. Zhu, Y. Ren, F. Kong, H. Wu, S. Liang, N. Chen, W. Xu, and F. Zhang, "Swarm-lio: Decentralized swarm lidar-inertial odometry," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 3254–3260.

[18] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[19] F. Zhu, Y. Ren, and F. Zhang, "Robust real-time lidar-inertial initialization," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 3948–3955.

[20] G. Lu, W. Xu, and F. Zhang, "On-manifold model predictive control for trajectory tracking on robotic systems," *IEEE Transactions on Industrial Electronics*, 2022.