RINGO: Real-time Navigation with a Guiding Trajectory for Aerial Manipulators in Unknown Environments

Zhaopeng Zhang, Shizhen Wu, Chenfeng Guo, Yongchun Fang, *Senior Member, IEEE*,
Jianda Han, *Member, IEEE*, Xiao Liang, *Senior Member, IEEE*

Abstract—Motion planning for aerial manipulators in constrained environments has typically been limited to known environments or simplified to that of multi-rotors, which leads to poor adaptability and overly conservative trajectories. This paper presents RINGO: Real-time Navigation with a Guiding Trajectory, a novel planning framework that enables aerial manipulators to navigate unknown environments in real time. The proposed method simultaneously considers the positions of both the multi-rotor and the end-effector. A pre-obtained multi-rotor trajectory serves as a guiding reference, allowing the end-effector to generate a smooth, collision-free, and workspace-compatible trajectory. Leveraging the convex hull property of B-spline curves, we theoretically guarantee that the trajectory remains within the reachable workspace. To the best of our knowledge, this is the first work that enables real-time navigation of aerial manipulators in unknown environments. The simulation and experimental results show the effectiveness of the proposed method. The proposed method generates less conservative trajectories than approaches that consider only the multi-rotor.

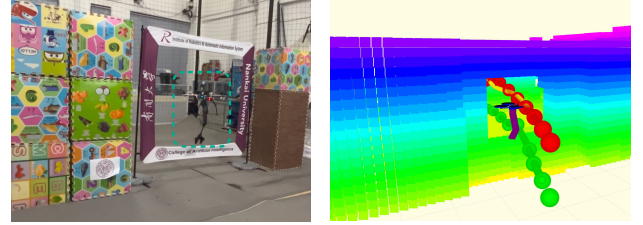*Index Terms*—Aerial manipulator, motion planning, trajectory optimization.

I. INTRODUCTION

AERIAL manipulators, typically consisting of a multi-rotor and a robotic arm, integrates the strengths of both components: the robotic arm provides the multi-rotor with manipulation capabilities, while the multi-rotor overcomes the fixed workspace limitation of the robotic arm, enhancing the aerial manipulator's flexibility for large-scale movements. In recent years, aerial manipulators have attracted considerable attention for the robust control [1], [2], visual servo [3], [4], and various practical applications [5]–[7].

Motion planning is a fundamental problem in robotics, aiming to generate collision-free and dynamically feasible trajectories for robotic systems. Although some studies have addressed the motion planning problem for aerial manipulators, there are still two key issues that need to be addressed.

Corresponding author: Xiao Liang

The authors are with the Institute of Robotics and Automatic Information System, College of Artificial Intelligence, and Tianjin Key Laboratory of Intelligent Robotics, Nankai University, Tianjin 300350, China, and also with Institute of Intelligence Technology and Robotic Systems, Shenzhen Research Institute of Nankai University, Shenzhen 518083, China (e-mail: zhangzp@mail.nankai.edu.cn; szwu@mail.nankai.edu.cn; guocf@mail.nankai.edu.cn; liangx@nankai.edu.cn; fangyc@nankai.edu.cn; hanjianda@nankai.edu.cn)

(a) aerial manipulator passing through a ring-shaped obstacle

(b) visualization in RViz

Fig. 1: Our proposed method is validated in a real-world environment. Experimental details are given in Sec. VI.

Firstly, the motion planning algorithm for aerial manipulators is typically conducted in known environments [8]–[13]. However, addressing the motion planning problem in an unknown environment presents significant challenges in terms of computational complexity and real-time performance. Secondly, the motion planning for an aerial manipulator in constrained environments is, in some cases, simplified to that of a multi-rotor [10]. However, this approach, which encloses the entire system within a large bounding sphere, tends to yield overly conservative trajectories. While minimizing the sphere's radius by retracting the robotic arm can mitigate this issue, it introduces a periodic planning approach, potentially increasing the time required for the arm to reach its goal state [14].

Building on the extensive previous research and the vibrant open-source community in the field of multi-rotor motion planning, we propose a novel motion planning method, called **R**eal-t**I**me **N**avigation with a **G**uiding traject**O**ry (**RINGO**) for aerial manipulators in unknown environments. Our proposed method employs a leader-follower-inspired motion planning framework for the aerial manipulator. Based on a previously planned and parameterized B-spline trajectory of the multi-rotor, we then plan the trajectory for the end-effector. Firstly, the initial trajectory for the end-effector is generated by a second-order Bézier curve from the initial position to the goal position and refined into a B-spline curve. Then, the gradient-based optimization method is used to optimize the trajectory of the end-effector. The linear and convex-hull properties of the B-spline curve guarantee that the trajectory is smooth, collision-free, and compatible with the available workspace.

Additionally, considering the motion capabilities related to the yaw angle of the multi-rotor, we incorporate an extra yaw rate cost into the trajectory optimization problem.

Compared with the existing works, our proposed method is able to generate the trajectory for the aerial manipulator in real-time without reducing the system to a multi-rotor-only model. Through theoretical analysis, the trajectory generated by the proposed method is guaranteed to be within the workspace and collision-free. The simulation and experimental results show that the proposed method can generate a collision-free and workspace-compatible trajectory for the aerial manipulator in real time. The main contributions of this paper are listed as follows:

1) Unlike most existing works that plan trajectories for aerial manipulators in known constrained environments, this paper presents the first real-time motion planning algorithm for aerial manipulators in unknown environments without degrading the problem to that of a multi-rotor.
2) With the convex hull property and the linear property of the B-spline curve, the trajectory generated by the proposed method can be guaranteed to be collision-free and workspace-compatible theoretically.

The rest of the paper is organized as follows: Section II presents the related work on motion planning for both multi-rotors and aerial manipulators. Section III introduces some preliminaries including the aerial manipulator system and B-spline curve. Section IV and V present the main part of the proposed method. In Section VI, the details of the algorithm implementation, simulation and experimental results are presented. Finally, Section VII concludes the paper and outlines future directions.

II. RELATED WORK

A. Motion Planning for Multi-rotors

By invoking the differential flatness property [15], the motion planning problem is simplified to consider only the position and the yaw angle of the multi-rotor [16]. In cases where a 360-degree LiDAR, rather than a front-view camera, is mounted on the multi-rotor [17], the yaw angle of the multi-rotor can be disregarded. Consequently, the motion planning problem is further simplified to find a collision-free and dynamically feasible trajectory for a point in $\mathbb{R}^3$, with obstacle avoidance ensured by inflating the obstacles. Alternatively, the multi-rotor can be enclosed within an ellipsoid [18] or a convex polyhedron [19] to reduce the conservatism of the trajectory by formulating the problem in $SE(3)$. Most existing studies on motion planning can be divided into two main stages: path search and trajectory optimization. In [17], the initial path is generated using the hybrid A* algorithm, followed by trajectory optimization through a gradient-based method after the trajectory has been parameterized as a B-spline curve. Similarly, the jump point search (JPS) algorithm is employed to determine the initial waypoints for the multi-rotor, and the trajectory is optimized by parameterizing it as a Bézier curve, as demonstrated in [20].

B. Motion Planning for Aerial Manipulators

Motion planning for aerial manipulators involves generating safe and feasible trajectories that account for both the multi-rotor and the robotic arm. Some studies adopt decoupled motion planning frameworks, in which the multi-rotor and the manipulator are planned in separate stages [14]. For instance, Kim *et al.* [8] integrate informed-RRT* with the local planner in [9] to generate collision-free trajectories in known environments. Cao *et al.* [10] propose a two-stage decoupled method for pick-and-place tasks, where the aerial manipulator is enclosed within a large sphere to simplify collision avoidance.

More recent efforts have incorporated whole-body planning strategies. Alvaro *et al.* [11] consider the kinematic model of the aerial robotic system with two arms for long-reach manipulation and use the RRT*-based method to plan the trajectory for the multi-rotor and the robotic arm in a known environment. Deng *et al.* [12] propose a dynamic ellipsoidal approximation method that adapts to varying manipulator configurations, enabling precise collision checking for the aerial manipulator with a delta arm. However, this method may not generalize well to serial-link arms due to their asymmetric and configuration-dependent collision volumes. Zhang *et al.* [13] formulated a coupled motion planning method by enclosing the aerial manipulator within a convex polyhedron and optimizing its trajectory in structured known environments.

Meanwhile, other works [21]–[24] explore task-constrained planning without addressing collision avoidance, limiting their applicability in real-world environments.

III. PRELIMINARY

A. Aerial Manipulator

In this paper, three frames are defined as $\{I\} = \{\boldsymbol{i}_1, \boldsymbol{i}_2, \boldsymbol{i}_3\}$, $\{B\} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$ and $\{V\} = \{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3\}$ which represent the inertia frame, the body-fixed frame, and the virtual body-fixed frame, respectively, which is presented in Fig. 2a. The three axes of coordinate frame $\{V\}$ are parallel to the corresponding three axes of coordinate frame $\{I\}$ and the origin of frame $\{V\}$ coincides with that of frame $\{B\}$.

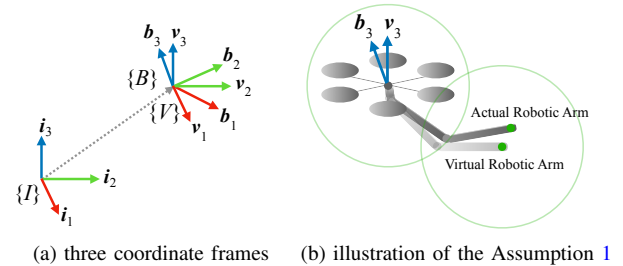(a) three coordinate frames   (b) illustration of the Assumption 1

Fig. 2: Illustration of the aerial manipulator and the defined coordinate frames.

The aerial manipulator combines a multi-rotor and a 2-DoF pitch-pitch robotic arm, as shown in Fig. 2b, whose degree of

freedom is 8. The state variables of the aerial manipulator are defined as follows:

$$q = \begin{Bmatrix} x \\ R \\ \boldsymbol{\theta} \end{Bmatrix} \in \mathbb{R}^3 \times SO(3) \times \Theta, \qquad (1)$$

where $x \in \mathbb{R}^3$ is the multi-rotor's position with respect to the inertial frame $\{I\}$, $R \in SO(3)$ is the rotation matrix from the body-fixed frame $\{B\}$ to the inertial frame $\{I\}$, $\boldsymbol{\theta} = \{\theta_1, \theta_2\} \in \Theta$ represents the joint angles of the robotic arm and $\Theta = \Theta_1 \times \Theta_2$. θ_i represents the joint angle of the i-th joint, where $\Theta_i \subset \mathbb{R}(i = 1, 2)$ specifies the allowable range of θ_i. $x_e \in \mathbb{R}^3$ is the position of the end-effector with respect to the inertial frame $\{I\}$, which is defined as follows:

$$x_e = x + R(b_3, \psi)^b x_e(\boldsymbol{\theta}). \qquad (2)$$

The rotation matrix $R(b_3, \psi)$ can be divided into two parts, including the tilt motion and the yaw angle ψ, as follows [25]:

$$R(b_3, \psi) = H_2(b_3)H_1(\psi).$$

Assumption 1. *The direction of the thrust b_3 is not too far from v_3 of the virtual body-fixed frame.*

With the aforementioned assumption, it can be derived that

$$R(b_3, \psi) \approx R(v_3, \psi) = H_2(v_3)H_1(\psi) = H_1(\psi).$$

The position of the end-effector x_e in (2) can be derived as

$$x_e \approx x + R(\psi)^b x_e(\boldsymbol{\theta}) = x + {}^v x_e(\boldsymbol{\theta}^+), \qquad (3)$$

where $\boldsymbol{\theta}^+ = \{\psi, \boldsymbol{\theta}\}$ is defined as the generalized joint angle vector. ${}^v x_e \in \mathbb{R}^3$ represents the end-effector position with respect to the virtual body-fixed frame $\{V\}$, which is denoted as x_{ve} in the following context.

As shown in Fig. 2b, the light-colored robotic arm represents the virtual arm, while the dark-colored one corresponds to the actual arm. The two green dots indicate the positions of the end-effectors associated with the virtual and actual arms, respectively. Notably, two appropriately sized spheres centered at the positions of the multi-rotor and the end-effector are sufficient to enclose the entire aerial manipulator system. This geometric abstraction facilitates subsequent safety margin analysis and collision checking in the planning process.
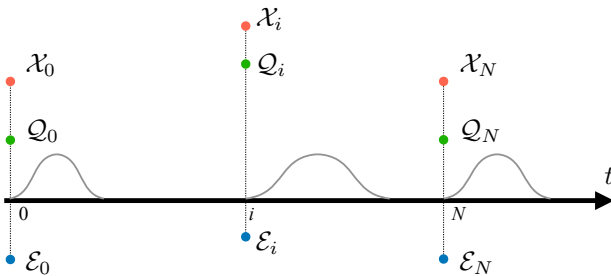
B. B-spline Curve

Fig. 3: Illustration of the linear property of the B-spline curve.

An s-order B-spline curve is determined by a set of $N + 1$ control points $\mathcal{E} = \{\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_N\}$ and a knot vector $\mathcal{T} =$ $[t_0, t_1, \ldots, t_M]^\top \in \mathbb{R}_+^{M+1}$, where $\mathcal{E}_i \in \mathbb{R}^3$, $t_i \in \mathbb{R}_+$ and $M = N + s + 1$.

Property 1. *If two B-spline curves with the same order share the same knot vector, the linear combination of them is still a B-spline curve with the same order.*

It is assumed that the trajectory of the multi-rotor's position $x(t)$ is a B-spline curve, determined by one set of control points $\mathcal{X} = \{\mathcal{X}_0, \mathcal{X}_1, \ldots, \mathcal{X}_N\}$, and the trajectory of the end-effector's position $x_e(t)$ is also a B-spline curve, determined by another set of control points $\mathcal{Q} = \{\mathcal{Q}_0, \mathcal{Q}_1, \ldots, \mathcal{Q}_N\}$. The two B-spline curves share the same time knots vector $\mathcal{T}$ and are defined as

$$x(t) = \sum_{i=0}^{N} B_i(t)\mathcal{X}_i, \qquad (4)$$

$$x_e(t) = \sum_{i=0}^{N} B_i(t)\mathcal{Q}_i, \qquad (5)$$

where $B_i(t)$ is the B-spline basis function of order s. Then, $x_{ve}(t)$ in (3) can be derived as

$$x_{ve}(t) = x_e(t) - x(t)$$
$$= \sum_{i=0}^{N} B_i(t)(\mathcal{Q}_i - \mathcal{X}_i) = \sum_{i=0}^{N} B_i(t)\mathcal{E}_i, \qquad (6)$$

where it can be concluded that $x_{ve}(t)$ is also a B-spline curve with the control points $\mathcal{E} = \{\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_N\}$ and share the same time knots vector $\mathcal{T}$ with $x(t)$ and $x_e(t)$, as shown in Fig. 3.

IV. METHOD OVERVIEW

A. Problem Statement

The problem this paper aims to address is *how to plan a collision-free trajectory for the aerial manipulator in real time within a constrained environment, given its initial and goal states.*

By inflating the obstacles, the collision avoidance constraint is formulated as two points in the $\mathbb{R}^3 \times \mathbb{R}^3$ space, where it is represented by the multi-rotor's position x and the end-effector position x_e. As illustrated in Fig. 2b, two green circles are placed in the multi-rotor and the virtual end-effector. Once Assumption 1 holds, b_3 will remain sufficiently close to e_3, and the large sphere attached to the virtual end-effector position will be adequately sized to encompass the actual robotic arm. As a result, the end-effector's trajectory will be decoupled from the multi-rotor's tilt motion.

B. Main Method

As shown in Fig. 4, the multi-rotor trajectory is firstly planned as the guiding trajectory and then parameterized as a B-spline curve, determined by the control points $\mathcal{X}$ and time knots $\mathcal{T}$. Then, the motion planning problem for the aerial manipulator is formulated as follows:
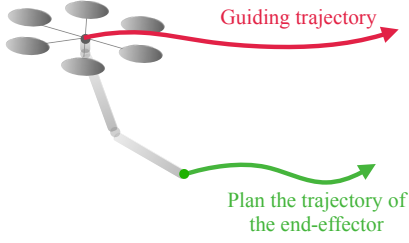
Fig. 4: Illustration of the proposed guiding trajectory-based motion planning method.

Algorithm 1 Trajectory Planning for the robotic arm

Input: the control points $\mathcal{X}$ and time knots $\mathcal{T}$ of the multi-rotor's trajectory $\boldsymbol{x}(t)$; the initial and goal position of the end-effector $\boldsymbol{x}_e(t_0)$, $\boldsymbol{x}_e(t_M)$.

Output: the trajectory of the end-effector $\boldsymbol{x}_e(t)$.

1: **Generate the initial trajectory** for the end-effector by combining $\boldsymbol{x}(t)$ and a second-order Bézier curve detailed in Section IV-C.

2: **Parameterize the trajectory** as a B-spline using control points $\mathcal{Q}$.

3: **Optimize the control points** $\mathcal{Q}$ with the gradient-based method detailed in Section V.

4: **Inverse Kinematics** to obtain the generalized joint trajectory $\boldsymbol{\theta}^+(t)$.

C. Initial Trajectory Generation

The initial trajectory of the end-effector $\boldsymbol{x}_{ve}(t)$ is generated by a second-order Bézier curve. The Bézier control point is determined by the initial and goal end-effector position with respect to the virtual body frame $\{V\}$, which is shown in Fig. 5. The Bézier control point is calculated as follows:

$$\boldsymbol{P} = \frac{1}{2}\lambda\left(\boldsymbol{x}_{ve}(t_0) + \boldsymbol{x}_{ve}(t_M)\right), \tag{7}$$

where $\lambda = \log\left(\frac{1}{2}\left|\arccos\left(\boldsymbol{x}_{ve}^\top(t_0)\boldsymbol{x}_{ve}(t_M)\right) + 1\right| + 1\right)$. It is equivalent to push the middle point of the initial and goal end-effector position far away. The bigger the angle between the initial and goal end-effector position vector, the larger the coefficient λ, which is designed to get the smooth trajectory in the joint space.

The initial trajectory of the end-effector can be generated by combining the pre-obtained trajectory $\boldsymbol{x}(t)$ of the multi-rotor with the second-order Bézier curve for the end-effector. Subsequently, by refining the initial trajectory, the end-effector's trajectory is parameterized using B-spline control points $\mathcal{Q} = \{\mathcal{Q}_0, \mathcal{Q}_1, \ldots, \mathcal{Q}_N\}$ along with time knots $\mathcal{T} = [t_0, t_1, \ldots, t_M]^\top$.

V. TRAJECTORY OPTIMIZATION

According to the procedure in Section IV-C, the initial trajectory of the robotic arm with respect to $\{V\}$ is generated by a second-order Bézier curve. Then, the initial trajectory of the end-effector with respect to $\{I\}$ is obtained. It is also a B-spline curve sharing the same time knot vector $\mathcal{T}$ with the multi-rotor's trajectory $\boldsymbol{x}(t)$, which is parameterized using the
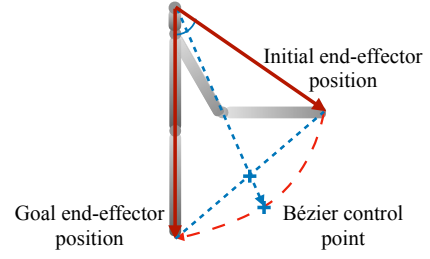

Fig. 5: Initial trajectory for the end-effector.

control points $\mathcal{X} = \{\mathcal{X}_0, \mathcal{X}_1, \ldots, \mathcal{X}_N\}$. The obstacle avoidance problem is not considered in the initial trajectory generation. Then, the following trajectory optimization problem is formulated:

$$\min_{\mathcal{Q}} f = \lambda_s f_s + \lambda_w f_w + \lambda_y f_y + \lambda_d f_d, \tag{8}$$

where f represents the total cost, while f_s, f_w, f_y, and f_d correspond to the smoothness cost, workspace cost, yaw rate cost, and obstacle avoidance cost, respectively. The weights for these cost components are denoted as λ_s, λ_w, λ_y, and λ_d. In the following context, both $\mathcal{Q} = \{\mathcal{Q}_0, \mathcal{Q}_1, \ldots, \mathcal{Q}_N\}$ and $\mathcal{E} = \{\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_N\}$ will be utilized, and the relationship of them has been given in Section III-B.
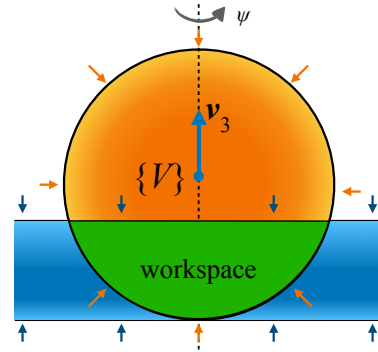
A. Workspace Cost

Fig. 6: Illustration of the workspace.

The workspace for the end-effector $\boldsymbol{x}_{ve}(t)$, as depicted in Fig. 6, is formulated as the intersection of a large sphere and two half-spaces, each defined by a plane, as the green region shown in the figure. The workspace, denoted as $\mathbb{W} \subset \mathbb{R}^3$, is a convex space.
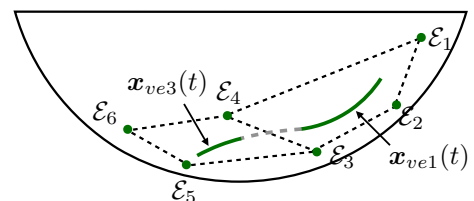

Fig. 7: Illustration of the convex hull in the workspace.

Remark 1. *The shape of workspace is determined by the configuration of the robotic arm. For other configurations, the workspace can be formulated into different shapes as long as they are convex.*

Two segments of the B-spline curve $\boldsymbol{x}_{ve1}(t)$ and $\boldsymbol{x}_{ve3}(t)$ are shown in Fig. 7, which are determined by the control points set $\{\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4\}$ and $\{\mathcal{E}_3, \mathcal{E}_4, \mathcal{E}_5, \mathcal{E}_6\}$, respectively.

For the sake of simplicity, let $\mathbb{C}_1 \subset \mathbb{R}^3$ represent the interior space of a tetrahedron with $\{\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4\}$ as the vertices. Due to the convex hull property of the B-spline curve, the segment $\boldsymbol{x}_{ve1}(t) \in \mathbb{C}_1 \subset \mathbb{W}, t \in [t_1, t_2]$. Similarly, the whole B-spline curve $\boldsymbol{x}_{ve}(t)$, in which $t \in [t_0, t_M]$ is theoretically guaranteed to be in the convex workspace $\mathbb{W}$ theoretically, if all the control points are in the convex workspace.

According to the above analysis, for the purpose of constraining $\boldsymbol{x}_{ve}$ in the workspace, the corresponding cost is formulated as follows:

$$f_w = \sum_{i=s}^{N-s} \frac{1}{k} \log \left(e^{h_o k F_o(\mathcal{Q}_i)} + e^{h_l k F_l(\mathcal{Q}_i)} \right), \qquad (9)$$

where $F_o(\mathcal{Q}_i)$ and $F_l(\mathcal{Q}_i)$ are the approximated signed distance functions determined by the circle and the two lines, respectively. k is a positive constant.
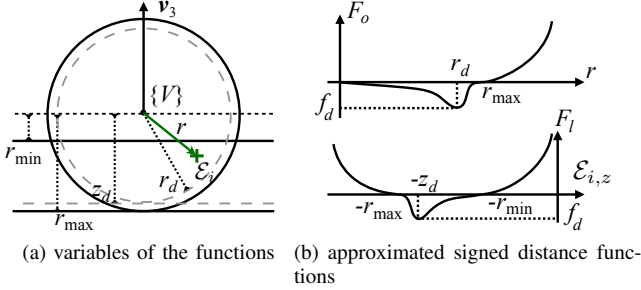

(a) variables of the functions   (b) approximated signed distance functions

Fig. 8: Illustration of two functions in the workspace cost.

As shown in Fig. 8b, the specific form of F_o and F_l are as follows:

$$F_o(\mathcal{Q}_i) =$$
$$\begin{cases} b_{o,1} r^2 + a_{o,1} r^3, & 0 \leq r \leq r_d \\ b_{o,2}(r - r_{\max})^2 + a_{o,2}(r - r_{\max})^3, & r_d \leq r \leq r_{\max} \\ (r - r_{\max})^2, & r_{\max} \leq r \end{cases} \quad (10)$$

$$F_l(\mathcal{Q}_i) =$$
$$\begin{cases} (\mathcal{E}_{i,z} + r_{\max})^2, & \mathcal{E}_{i,z} \leq -r_{\max} \\ b_{l,1}(\mathcal{E}_{i,z} + r_{\max})^2 + \\ \quad a_{l,1}(\mathcal{E}_{i,z} + r_{\max})^3, & -r_{\max} \leq \mathcal{E}_{i,z} \leq -z_d \\ b_{l,2}(\mathcal{E}_{i,z} + r_{\min})^2 + \\ \quad a_{l,2}(\mathcal{E}_{i,z} + r_{\min})^3, & -z_d \leq \mathcal{E}_{i,z} \leq -r_{\min} \\ (\mathcal{E}_{i,z} + r_{\min})^2, & -r_{\min} \leq \mathcal{E}_{i,z} \end{cases} \quad (11)$$

where $r_{\max}$ and $r_{\min}$ are the parameters of the convex region. $\mathcal{E}_{i,x}, \mathcal{E}_{i,y}$, and $\mathcal{E}_{i,z}$ are three components of the control point $\mathcal{E}_i$. $r = \sqrt{\mathcal{E}_i^\top \mathcal{E}_i}$ is the 2-norm of the vector $\mathcal{E}_i$. By guaranteeing that $F_o(\mathcal{Q}_i)$ and $F_l(\mathcal{Q}_i)$ are continuously differentiable, the

coefficients $a_{o,j}$, $b_{o,j}$, $a_{l,j}$, and $b_{l,j}$ $(j = 1, 2)$ can be derived by combining r_d, z_d and f_d, while r_d and z_d are related by the goal state of the end-effector, and f_d is a constant.
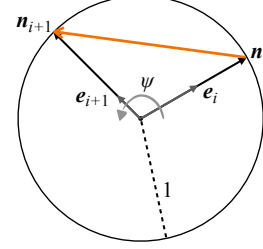
B. Yaw Rate Cost

Fig. 9: Illustration of the yaw rate cost.

As shown in Fig. 9, $\boldsymbol{e}_i$ and $\boldsymbol{e}_{i+1}$ are the projections of the control points $\mathcal{E}_i$ and $\mathcal{E}_{i+1}$ on the x-y plane, expressed as follows:

$$\boldsymbol{e}_i = \begin{bmatrix} \mathcal{E}_{i,x} \\ \mathcal{E}_{i,y} \\ 0 \end{bmatrix}, \quad \boldsymbol{e}_{i+1} = \begin{bmatrix} \mathcal{E}_{i+1,x} \\ \mathcal{E}_{i+1,y} \\ 0 \end{bmatrix}.$$

$\boldsymbol{n}_i$ and $\boldsymbol{n}_{i+1}$ are the normalized vector of $\boldsymbol{e}_i$ and $\boldsymbol{e}_{i+1}$, respectively, which means that

$$\boldsymbol{n}_i = \frac{\boldsymbol{e}_i}{\|\boldsymbol{e}_i\|} = \begin{bmatrix} \frac{\mathcal{E}_{i,x}}{\sqrt{\mathcal{E}_{i,x}^2 + \mathcal{E}_{i,y}^2}} \\ \frac{\mathcal{E}_{i,y}}{\sqrt{\mathcal{E}_{i,x}^2 + \mathcal{E}_{i,y}^2}} \\ 0 \end{bmatrix},$$

$$\boldsymbol{n}_{i+1} = \frac{\boldsymbol{e}_{i+1}}{\|\boldsymbol{e}_{i+1}\|} = \begin{bmatrix} \frac{\mathcal{E}_{i+1,x}}{\sqrt{\mathcal{E}_{i+1,x}^2 + \mathcal{E}_{i+1,y}^2}} \\ \frac{\mathcal{E}_{i+1,y}}{\sqrt{\mathcal{E}_{i+1,x}^2 + \mathcal{E}_{i+1,y}^2}} \\ 0 \end{bmatrix}.$$

The yaw rate cost is formulated as

$$f_y = \sum_{i=s}^{N-s-1} F_y(\mathcal{Q}_i, \mathcal{Q}_{i+1}) = \sum_{i=s}^{N-s-1} \|\boldsymbol{n}_{i+1} - \boldsymbol{n}_i\|^2. \quad (12)$$

C. Smoothness Cost

The smoothness cost is formulated as minimizing the normalization of the acceleration control points. The smoothness cost is formulated as

$$f_s = \sum_{i=s-2}^{N-s} F_s(\mathcal{Q}_i, \mathcal{Q}_{i+1}, \mathcal{Q}_{i+2}) = \sum_{i=s-2}^{N-s} \|\mathcal{A}_i\|^2. \quad (13)$$

Different from the uniform B-spline, the acceleration control points $\mathcal{A}_i$ of the non-uniform B-spline is as follows:

$$\mathcal{A}_i = M_i \begin{bmatrix} \mathcal{E}_i \\ \mathcal{E}_{i+1} \\ \mathcal{E}_{i+2} \end{bmatrix} = M_i \begin{bmatrix} \mathcal{Q}_i \\ \mathcal{Q}_{i+1} \\ \mathcal{Q}_{i+2} \end{bmatrix} - M_i \begin{bmatrix} \mathcal{X}_i \\ \mathcal{X}_{i+1} \\ \mathcal{X}_{i+2} \end{bmatrix}.$$

The coefficients M_i is defined as

$$M_i = [M_{i,0}, M_{i,1}, M_{i,2}]$$
$$= \frac{s(s-1)}{t_{i+2,s-1}} \left[\frac{1}{t_{i+1,s}} \quad -\left(\frac{1}{t_{i+2,s}} + \frac{1}{t_{i+1,s}} \right) \quad \frac{1}{t_{i+2,s}} \right],$$

in which $t_{i,s}$ denotes the time span from t_i to t_{i+s}, which means that $t_{i,s} = t_{i+s} - t_i$. The time knot vector $\mathcal{T} = [t_0, t_1, \ldots, t_M]^\top$ and all the control points $\mathcal{Q}_i$ have been derived from the multi-rotor's trajectory.

D. Obstacle Avoidance Cost

The obstacle avoidance cost is formulated as the distance between the control point $\mathcal{Q}_i$ and the closet obstacle, formulated as follows:

$$f_d = \sum_{i=s}^{N-s} F_d(d(\mathcal{Q}_i)), \tag{14}$$

where $d(\mathcal{Q}_i)$ is the distance between the control point $\mathcal{Q}_i$ and the closest obstacle, which can be obtained by the ESDF map [26]. The cost function on the i-th control point $F_d(d(\mathcal{Q}_i))$ is as follows:

$$F_d(d(\mathcal{Q}_i)) = \begin{cases} (d(\mathcal{Q}_i) - d_{thr})^2, & d(\mathcal{Q}_i) \leq d_{thr} \\ 0, & d(\mathcal{Q}_i) > d_{thr} \end{cases} \tag{15}$$

where d_{thr} is a distance threshold.

The gradients of the above-mentioned cost functions with respect to the control points are derived in Appendix A.

VI. IMPLEMENTATION AND RESULTS

In order to verify the effectiveness of the proposed method, the implementation details, simulation results and experimental results are introduced in this section.

A. Algorithm Implementation and Experiment Setup

1) Algorithm Implementation: The order s of the B-spline curve is set to 3 and the optimization problem presented in (8) is solved by NLopt Library[1]. The simulation platform is adapted from the Fast-Planner framework [17], incorporating the multi-rotor dynamics model, random map generator, and point cloud rendering module. All simulations are conducted on an Intel Core i7−13700 CPU and GeForce GTX 3070 Ti GPU. To ensure a fair comparison, all computations are conducted with the same aforementioned computation capability. In real world experiments, all the state estimation, mapping and motion planning modules run on an Intel Core i7−13700 CPU with 16GB RAM.

2) Aerial Manipulator: The aerial manipulator system consists of an F550 hexrotor and a custom-built robotic arm. Specifically, the hexrotor is equipped with a Pixhawk FMUv5 flight controller, which connects to the onboard computer via the Mavlink communication protocol. The flight controller employs the cascaded P-PID controller incorporating the feedforward term from the robotic arm, which is embedded within the Pixhawk FMUv5[2]. The flight controller powers six pairs of T-Motors and 10-inch propellers through the Electronic Speed Controllers (ESCs). The robotic arm is conducted by Dynamixel servo motors and a few self-designed connectors.
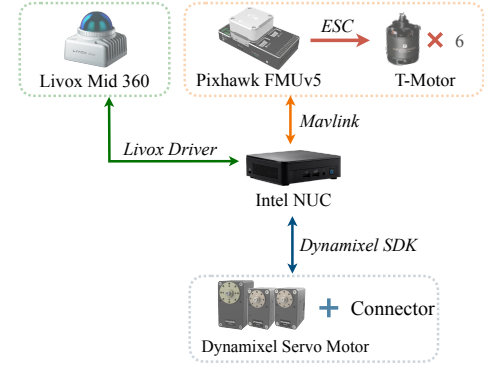
[1]https://nlopt.readthedocs.io/en/latest/
[2]https://ardupilot.org/copter/docs/common-cuav-v5plus-overview.html

Fig. 10: Experimental platform.

3) State Estimation and Mapping: The Livox Mid 360 LiDAR is mounted on the multi-rotor and integrated using the Livox Driver[3]. Fast-lio2 [27] is employed to estimate the odometry of the multi-rotor and to generate a dense point cloud map. To enhance the motion planning algorithm, we increase the frequency of the multi-rotor's odometry estimation provided by Fast-lio2. The state estimation for the robotic arm is handled by the Dynamixel SDK, while the end-effector's position is computed using the robotic arm's forward kinematics.
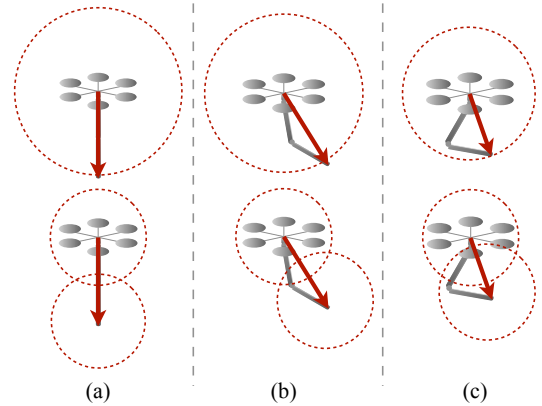
B. Simulation Results

Fig. 11: Illustration of the simulation scenario.

In this section, we evaluate the proposed method in three scenarios, as illustrated in Fig. 11. Scenario (a), (b) and (c) depict three different initial and goal positions for the robotic arm. The red arrows extending from the center of the multi-rotor to the arm's end-effector represent the maximum distance from the multi-rotor's center to its outer edge. The proposed method incorporates trajectory planning for both the multi-rotor and the robotic arm, whereas the baseline method keeps the robotic arm fixed, which requires a larger inflation radius. Fig. 12 shows the multi-rotor's travel trajectories across the three scenarios within the same environment.

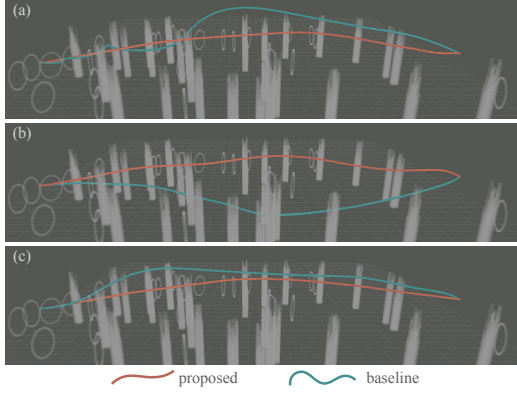[3]https://github.com/Livox-SDK/Livox-SDK

Fig. 12: Multi-rotor's travel trajectories in different scenarios.

Fig. 13: The detail of the robotic arm's computation time.

TABLE I: Quantitative Comparison

Scenario	Method	Travel Trajectory		Computation Time (ms)		
		Length (m)	Time (s)	multi-rotor	robotic arm	total
(a)	proposed	**42.52**	**15.79**	0.79	0.22	**1.01**
	baseline	49.63	22.87	1.73	-	1.73
(b)	proposed	**42.58**	**16.03**	1.00	0.21	**1.21**
	baseline	44.08	17.72	1.55	-	1.55
(c)	proposed	**42.39**	**15.88**	1.05	0.12	**1.17**
	baseline	43.82	16.65	1.22	-	1.22

The quantitative comparison results are presented in TABLE I. From the above comparison, the proposed method achieves shorter trajectory length and flight time for the multi-rotor's trajectory than the baseline method, which is intuitive. In the baseline method, the robotic arm is assumed to remain in a fixed configuration throughout the entire trajectory. To account for potential collisions without modeling the manipulator's motion, the entire aerial manipulator is conservatively enclosed within a large bounding sphere. The quantitative results also show that the baseline method exhibits shorter travel distance and time in scenarios (b) and (c), attributed to the reduced inflation radius in these cases. It's more like an ablation study.

In addition to travel trajectories, computation time is compared as well. The computation time in TABLE I is the average time of each planning process in the whole travel trajectory. The baseline method does not include computation time for the robotic arm, while the proposed method includes both the multi-rotor and the robotic arm's computation time. The proposed method has a shorter time than the baseline method in all scenarios. This is because the baseline method requires trajectory planning for the multi-rotor in a more constrained environment, leading to higher computation cost in both path searching and trajectory optimization.

For the purpose of further exploring the impact of the proposed method on the computation time of the robotic arm, we present a detailed boxplot in Fig. 13. The boxplot provides a detailed illustration of the computation time of Algorithm 1 across different scenarios. Even in the most extreme case for scenario (a), the time remains *less than 0.6 ms*.

C. Experimental Results

In this paper, we present two fully autonomous flight experiments conducted in unknown environments, as illustrated
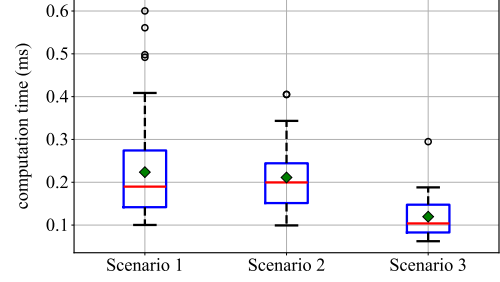
in Fig. 14 and Fig. 15.

In Experiment 1, the aerial manipulator can successfully plan a collision-free trajectory for both the multi-rotor and the end-effector through a ring-shaped obstacle. In Experiment 2, the aerial manipulator starts behind the vertical obstacles, and it can successfully plan a collision-free trajectory that involves hurdling over the horizontal obstacles and navigating through an unknown ring-shaped obstacle.

To the best of our knowledge, this is the first instance of an aerial manipulator achieving autonomous flight in unknown environments. The experimental results demonstrate the effectiveness of the proposed method in real-world applications.
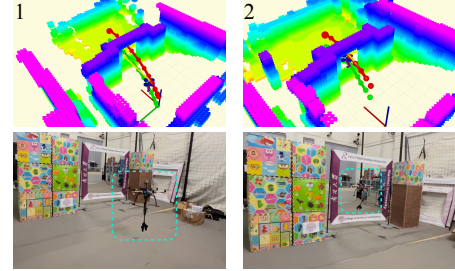

Fig. 14: The snapshots and visualization of Experiment 1.
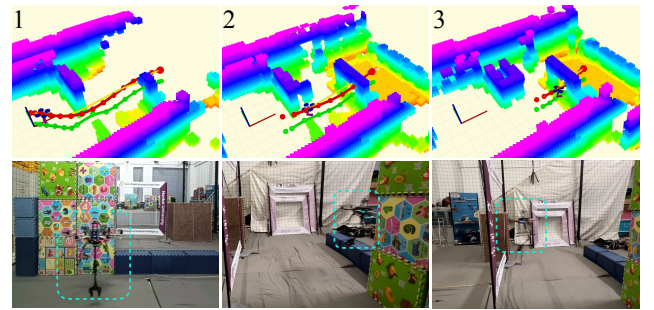

Fig. 15: The snapshots and visualization of Experiment 2.

VII. Conclusion

In this paper, the motion planning problem for aerial manipulators is formulated as the positions of two points in the space, including the positions of the multi-rotor and the end-effector. By utilizing the planned and parameterized trajectory of the multi-rotor, we propose a real-time algorithm

to plan the trajectory for the end-effector. An initial trajectory for the end-effector is generated by a second-order Bézier curve. Then, the gradient-based optimization method is used to optimize the trajectory of the end-effector. The linear and convex-hull properties of the B-spline curve ensure that the trajectory is smooth, collision-free and workspace-compatible. The simulation and experimental results show that the proposed method's effectiveness. Compared to motion planning strategies that only consider the multi-rotor, our approach yields less conservatism.

Although it is the pitch-pitch 2-DOF robotic arm that this paper focuses on, the proposed method can be easily extended to robotic arms with other configurations. In future work, we will develop a planner that simultaneously considers the positions of both the multi-rotor and the end-effector, and validate the algorithm in field environments.

APPENDIX A
THE GRADIENT OF THE COST FUNCTION

A. Workspace cost

The gradient of the distance r to the control point $\mathcal{E}_i$ is calculated as

$$\frac{\partial r^2}{\partial r} \cdot \frac{\partial r}{\partial \mathcal{E}_i} = 2r \cdot \frac{\partial r}{\partial \mathcal{E}_i} = 2\mathcal{E}_i \quad \Rightarrow \quad \frac{\partial r}{\partial \mathcal{E}_i} = \frac{\mathcal{E}_i}{r}.$$

The gradient of the workspace cost $F_w(\mathcal{Q}_i)$ is given as follows:

$$\frac{\partial f_w}{\partial \mathcal{Q}_i} = \frac{h_o e^{h_o k F_o(\mathcal{Q}_i)}}{e^{h_o k F_o(\mathcal{Q}_i)} + e^{h_l k F_l(\mathcal{Q}_i)}} \frac{\partial F_o(\mathcal{Q}_i)}{\partial \mathcal{Q}_i} + \frac{h_l e^{h_l k F_l(\mathcal{Q}_i)}}{e^{h_o k F_o(\mathcal{Q}_i)} + e^{h_l k F_l(\mathcal{Q}_i)}} \frac{\partial F_l(\mathcal{Q}_i)}{\partial \mathcal{Q}_i}.$$

The gradient of the circle cost $F_o(\mathcal{Q}_i)$ is calculated as

$$\frac{\partial F_o(\mathcal{Q}_i)}{\partial \mathcal{Q}_i} = \frac{\partial F_o(\mathcal{Q}_i)}{\partial r} \frac{\partial r}{\partial \mathcal{E}_i} \frac{\partial \mathcal{E}_i}{\partial \mathcal{Q}_i} = \frac{\partial F_o(\mathcal{Q}_i)}{\partial r} \frac{\mathcal{E}_i}{r}.$$

$$\frac{\partial F_o(\mathcal{Q}_i)}{\partial r} = \begin{cases} 2b_{o,1}r + 3a_{o,1}r^2, & 0 \leq r \leq r_d \\ 2b_{o,2}(r - r_{\max}) + 3a_{o,2}(r - r_{\max})^2, & r_d \leq r \leq r_{\max} \\ 2(r - r_{\max}). & r_{\max} \leq r \end{cases}$$

The gradient of the line cost $F_l(\mathcal{Q}_i)$ is calculated as

$$\frac{\partial F_l(\mathcal{Q}_i)}{\partial \mathcal{Q}_i} = \frac{\partial F_l(\mathcal{Q}_i)}{\partial \mathcal{E}_i} \frac{\partial \mathcal{E}_i}{\partial \mathcal{Q}_i} = \begin{bmatrix} 0 \\ 0 \\ \frac{\partial F_l(\mathcal{Q}_i)}{\partial \mathcal{E}_{i,z}} \end{bmatrix}.$$

$$\frac{\partial F_l(\mathcal{Q}_i)}{\mathcal{E}_{i,z}} = \begin{cases} 2(\mathcal{E}_{i,z} + r_{\max}), & -r_{\max} \leq \mathcal{E}_{i,z}, \\ 2b_{l,1}(\mathcal{E}_{i,z} + r_{\max}) + \\ \qquad 3a_{l,1}(\mathcal{E}_{i,z} + r_{\max})^2, & -r_{\max} \leq \mathcal{E}_{i,z} \leq -z_d \\ 2b_{l,2}(\mathcal{E}_{i,z} + r_{\min}) + \\ \qquad 3a_{l,2}(\mathcal{Q}_{i,z} + r_{\min})^2, & -z_d \leq \mathcal{E}_{i,z} \leq -r_{\min} \\ 2(\mathcal{E}_{i,z} + r_{\min}). & -r_{\min} \leq \mathcal{E}_{i,z} \end{cases}$$

B. Yaw rate cost

The gradient of the yaw rate cost $F_y(\mathcal{Q}_i, \mathcal{Q}_{i+1})$ is given as follows:

$$\frac{\partial F_y}{\partial \mathcal{Q}_i} = -2\left[(\boldsymbol{n}_{i+1} - \boldsymbol{n}_i)^\top \frac{\partial \boldsymbol{n}_i}{\partial \mathcal{Q}_i}\right]^\top$$
$$= -2\left[(\boldsymbol{n}_{i+1} - \boldsymbol{n}_i)^\top \frac{\partial \boldsymbol{n}_i}{\partial \mathcal{E}_i}\right]^\top$$
$$= -2\left(\frac{\partial \boldsymbol{n}_i}{\partial \mathcal{E}_i}\right)^\top (\boldsymbol{n}_{i+1} - \boldsymbol{n}_i).$$

$$\frac{\partial F_y}{\partial \mathcal{Q}_{i+1}} = 2\left[(\boldsymbol{n}_{i+1} - \boldsymbol{n}_i)^\top \frac{\partial \boldsymbol{n}_{i+1}}{\partial \mathcal{Q}_{i+1}}\right]^\top$$
$$= 2\left[(\boldsymbol{n}_{i+1} - \boldsymbol{n}_i)^\top \frac{\partial \boldsymbol{n}_{i+1}}{\partial \mathcal{E}_{i+1}}\right]^\top$$
$$= 2\left(\frac{\partial \boldsymbol{n}_{i+1}}{\partial \mathcal{E}_{i+1}}\right)^\top (\boldsymbol{n}_{i+1} - \boldsymbol{n}_i).$$

where,

$$\frac{\partial \boldsymbol{n}_i}{\partial \mathcal{E}_i} = \begin{bmatrix} \frac{\partial \boldsymbol{n}_{i,x}}{\partial \mathcal{E}_{i,x}} & \frac{\partial \boldsymbol{n}_{i,x}}{\partial \mathcal{E}_{i,y}} & 0 \\ \frac{\partial \boldsymbol{n}_{i,y}}{\partial \mathcal{E}_{i,x}} & \frac{\partial \boldsymbol{n}_{i,y}}{\partial \mathcal{E}_{i,y}} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$
$$= \begin{bmatrix} \frac{\mathcal{E}_{i,y}^2}{(\mathcal{E}_{i,x}^2 + \mathcal{E}_{i,y}^2)^{\frac{3}{2}}} & \frac{-\mathcal{E}_{i,x}\mathcal{E}_{i,y}}{(\mathcal{E}_{i,x}^2 + \mathcal{E}_{i,y}^2)^{\frac{3}{2}}} & 0 \\ \frac{-\mathcal{E}_{i,x}\mathcal{E}_{i,y}}{(\mathcal{E}_{i,x}^2 + \mathcal{E}_{i,y}^2)^{\frac{3}{2}}} & \frac{\mathcal{E}_{i,x}^2}{(\mathcal{E}_{i,x}^2 + \mathcal{E}_{i,y}^2)^{\frac{3}{2}}} & 0 \\ 0 & 0 & 0 \end{bmatrix} = \left(\frac{\partial \boldsymbol{n}_i}{\partial \mathcal{E}_i}\right)^\top.$$

It can be concluded that

$$\frac{\partial f_y}{\partial \mathcal{Q}_i} = -2\left(\frac{\partial \boldsymbol{n}_i}{\partial \mathcal{E}_i}\right)^\top (\boldsymbol{n}_{i+1} - \boldsymbol{n}_i) + 2\left(\frac{\partial \boldsymbol{n}_i}{\partial \mathcal{E}_i}\right)^\top (\boldsymbol{n}_i - \boldsymbol{n}_{i-1}).$$

C. Smoothness cost

The gradient of the smoothness cost is calculated as

$$\frac{\partial f_s}{\partial \mathcal{Q}_i} = \left(2\mathcal{A}_i^\top \frac{\partial \mathcal{A}_i}{\partial \mathcal{E}_i} \frac{\partial \mathcal{E}_i}{\partial \mathcal{Q}_i}\right)^\top + \left(2\mathcal{A}_i^\top \frac{\partial \mathcal{A}_{i-1}}{\partial \mathcal{E}_i} \frac{\partial \mathcal{E}_i}{\partial \mathcal{Q}_i}\right)^\top + \left(2\mathcal{A}_i^\top \frac{\partial \mathcal{A}_{i-2}}{\partial \mathcal{E}_i} \frac{\partial \mathcal{E}_i}{\partial \mathcal{Q}_i}\right)^\top$$
$$= 2\left(\frac{\partial \mathcal{A}_i}{\partial \mathcal{E}_i}\right)^\top \mathcal{A}_i + 2\left(\frac{\partial \mathcal{A}_{i-1}}{\partial \mathcal{E}_i}\right)^\top \mathcal{A}_{i-1} + 2\left(\frac{\partial \mathcal{A}_{i-2}}{\partial \mathcal{E}_i}\right)^\top \mathcal{A}_{i-2}.$$
$$= 2M_{i,0}\mathcal{A}_i + 2M_{i-1,1}\mathcal{A}_{i-1} + 2M_{i-2,2}\mathcal{A}_{i-2}.$$

D. Obstacle Avoidance cost

The gradient of the obstacle avoidance cost $F_o(\mathcal{Q}_i)$ is calculated as

$$\frac{f_d}{\partial \mathcal{Q}_i} = 2\left(d(\mathcal{Q}_i) - d_{\text{thr}}\right) \frac{\partial d(\mathcal{Q}_i)}{\partial \mathcal{Q}_i},$$

where the gradient of the distance $d(\mathcal{Q}_i)$ to the control point $\mathcal{Q}_i$ can be obtained in the ESDF map [26] directly.

REFERENCES

[1] G. Zhang, Y. He, B. Dai, F. Gu, J. Han, and G. Liu, "Robust control of an aerial manipulator based on a variable inertia parameters model," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 11, pp. 9515–9525, 2019.

[2] M. Wang, Z. Chen, K. Guo, X. Yu, Y. Zhang, L. Guo, and W. Wang, "Millimeter-level pick and peg-in-hole task achieved by aerial manipulator," *IEEE Transactions on Robotics*, vol. 40, pp. 1242–1260, 2024.

[3] H. Zhong, Z. Miao, Y. Wang, J. Mao, L. Li, H. Zhang, Y. Chen, and R. Fierro, "A practical visual servo control for aerial manipulation using a spherical projection model," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 12, pp. 10564–10574, 2020.

[4] Y. Chen, L. Lan, X. Liu, G. Zeng, C. Shang, Z. Miao, H. Wang, Y. Wang, and Q. Shen, "Adaptive stiffness visual servoing for unmanned aerial manipulators with prescribed performance," *IEEE Transactions on Industrial Electronics*, vol. 71, no. 9, pp. 11028–11038, 2024.

[5] D. Lee, H. Seo, D. Kim, and H. J. Kim, "Aerial manipulation using model predictive control for opening a hinged door," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1237–1242, IEEE, 2020.

[6] A. González-Morgado, E. Cuniato, M. Tognon, G. Heredia, R. Siegwart, and A. Ollero, "Controlled shaking of trees with an aerial manipulator," *IEEE/ASME Transactions on Mechatronics*, 2024. doi:10.1109/TMECH.2024.3410167.

[7] K. Bodie, M. Brunner, M. Pantic, S. Walser, P. Pfändler, U. Angst, R. Siegwart, and J. Nieto, "Active interaction force control for contact-based inspection with a fully actuated aerial vehicle," *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 709–722, 2021.

[8] H. Kim, H. Seo, J. Kim, and H. J. Kim, "Sampling-based motion planning for aerial pick-and-place," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7402–7408, IEEE, 2019.

[9] H. Seo, S. Kim, and H. J. Kim, "Locally optimal trajectory planning for aerial manipulation in constrained environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1719–1724, IEEE, 2017.

[10] H. Cao, J. Shen, C. Liu, B. Zhu, and S. Zhao, "Motion planning for aerial pick-and-place with geometric feasibility constraints," *IEEE Transactions on Automation Science and Engineering*, 2024.

[11] A. Caballero, A. Suarez, F. Real, V. M. Vega, M. Bejar, A. Rodriguez-Castaño, and A. Ollero, "First experimental results on motion planning for transportation in aerial long-reach manipulators with two arms," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8471–8477, IEEE, 2018.

[12] W. Deng, H. Chen, B. Ye, H. Chen, and X. Lyu, "Whole-body integrated motion planning for aerial manipulators," *arXiv preprint arXiv:2501.06493*, 2025.

[13] Z. Zhang, H. Yu, Y. Chai, Z. Yang, X. Liang, Y. Fang, and J. Han, "An end-effector-oriented coupled motion planning method for aerial manipulators in constrained environments," *IEEE/ASME Transactions on Mechatronics*, 2025. DOI:10.1109/TMECH.2025.3550562.

[14] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, present, and future of aerial robotic manipulators," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 626–645, 2021.

[15] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2520–2525, IEEE, 2011.

[16] B. Zhou, J. Pan, F. Gao, and S. Shen, "Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1992–2009, 2021.

[17] 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.

[18] S. Liu, K. Mohta, N. Atanasov, and V. Kumar, "Search-based motion planning for aggressive flight in se(3)," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2439–2446, 2018.

[19] Z. Han, Z. Wang, N. Pan, Y. Lin, C. Xu, and F. Gao, "Fast-racing: An open-source strong baseline for se(3) planning in autonomous drone racing," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8631–8638, 2021.

[20] J. Tordesillas, B. T. Lopez, M. Everett, and J. P. How, "Faster: Fast and safe trajectory planner for navigation in unknown environments," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 922–938, 2021.

[21] M. Tognon, E. Cataldi, H. A. T. Chavez, G. Antonelli, J. Cortés, and A. Franchi, "Control-aware motion planning for task-constrained aerial manipulation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2478–2484, 2018.

[22] W. Luo, J. Chen, H. Ebel, and P. Eberhard, "Time-optimal handover trajectory planning for aerial manipulators based on discrete mechanics and complementarity constraints," *IEEE Transactions on Robotics*, vol. 39, no. 6, pp. 4332–4349, 2023.

[23] J. Welde and V. Kumar, "Coordinate-free dynamics and differential flatness of a class of 6DOF aerial manipulators," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4307–4313, IEEE, 2020.

[24] J. Welde, J. Paulos, and V. Kumar, "Dynamically feasible task space planning for underactuated aerial manipulators," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3232–3239, 2021.

[25] M. Watterson and V. Kumar, "Control of quadrotors using the hopf fibration on so(3)," in *Proceedings of the Robotics Research: The 18th International Symposium (ISRR)*, pp. 199–215, Springer, 2020.

[26] L. Han, F. Gao, B. Zhou, and S. Shen, "Fiesta: Fast incremental euclidean distance fields for online motion planning of aerial robots," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4423–4430, 2019.

[27] 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.