

Whole-Body Integrated Motion Planning for Aerial Manipulators

Weiliang Deng[✉], Hongming Chen[✉], Biyu Ye[✉], Haoran Chen[✉], Ziliang Li[✉] and Ximin Lyu[✉]

Abstract—Expressive motion planning for Aerial Manipulators (AMs) is essential for tackling complex manipulation tasks, yet achieving coupled trajectory planning adaptive to various tasks remains challenging, especially for those requiring aggressive maneuvers. In this work, we propose a novel whole-body integrated motion planning framework for quadrotor-based AMs that leverages flexible waypoint constraints to achieve versatile manipulation capabilities. These waypoint constraints enable the specification of individual position requirements for either the quadrotor or end-effector, while also accommodating higher-order velocity and orientation constraints for complex manipulation tasks. To implement our framework, we exploit spatio-temporal trajectory characteristics and formulate an optimization problem to generate feasible trajectories for both the quadrotor and manipulator while ensuring collision avoidance considering varying robot configurations, dynamic feasibility, and kinematic feasibility. Furthermore, to enhance the maneuverability for specific tasks, we employ Imitation Learning (IL) to facilitate the optimization process to avoid poor local optima. The effectiveness of our framework is validated through comprehensive simulations and real-world experiments, where we successfully demonstrate nine fundamental manipulation skills across various environments.

Index Terms—Aerial manipulator, delta arm, motion planning, collision avoidance, waypoint constraint.

I. INTRODUCTION

AERIAL manipulators, which consist of Unmanned Aerial Vehicles (UAVs) and manipulators, have garnered significant interest from researchers and industries due to their wide operational range and agile manipulation capabilities [1]. These systems have proven to operate effectively in challenging or hazardous environments, facilitating applications across various fields such as 3D printing, inspection, transportation, grasping, repair and perching [2]–[7]. Various aspects of aerial manipulators have been extensively studied, including

Manuscript received: January 15, 2025; Revised June 28, 2025; Accepted September 26, 2025.

This paper was recommended for publication by Editor Paolo Robuffo Giordano upon evaluation of the Associate Editor and Reviewers comments. This work was supported by the National Key Research and Development Program of China (Grant No. 2023YFB4706600), the National Natural Science Foundation of China (Grant No. 62303495), the Guangdong-Hongkong-Macao Joint Research of Science and Technology Planning Funding from Guangdong Province (Grant No. 2023A0505010019), and the Young Talent Support Project of Guangzhou Association for Science and Technology (Grant No. QT-2025-004). (Weiliang Deng and Hongming Chen contributed equally to this work.) (Corresponding authors: Ximin Lyu.)

Weiliang Deng, Hongming Chen, Biyu Ye, Haoran Chen, Ziliang Li, and Ximin Lyu are with the School of Intelligent Systems Engineering, Sun Yat-sen University, Guangzhou, 510275, China (e-mail: {dengwliang, chenhm223, yeby9, chenhr66, litleong3}@mail2.sysu.edu.cn, lyxm6@mail.sysu.edu.cn). Ximin Lyu is also with Differential Robotics Technology Co., Ltd., Hangzhou, China.

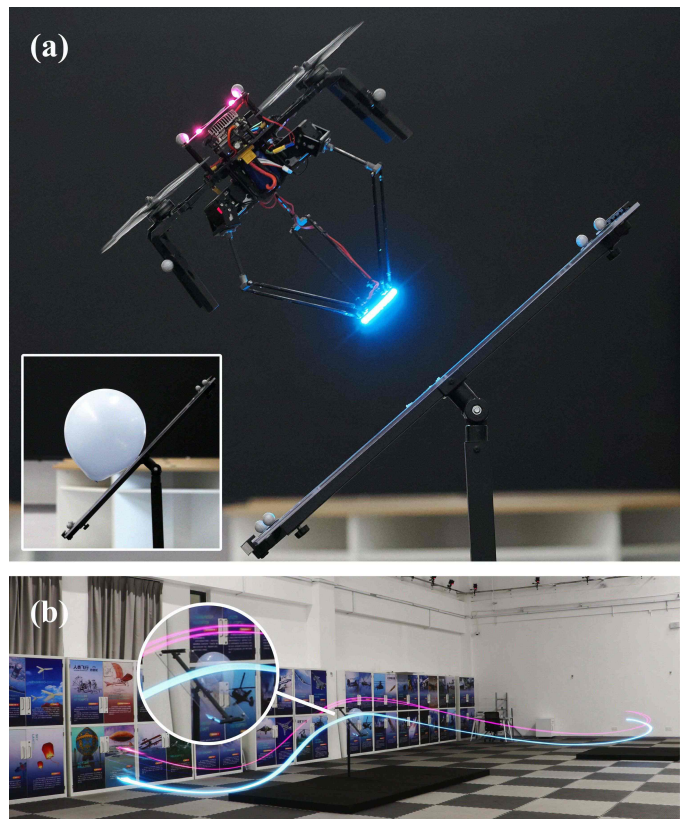


Fig. 1. Demonstration of aerial striking using whole-body integrated motion planning. (a) The aerial manipulator executes a planned trajectory to reach a constrained waypoint, maintaining an attitude parallel to the inclined plane while configuring its delta arm for balloon striking. (b) Time-lapse visualization capturing the complete execution of the planned motion.

platform design [8]–[11], motion planning [12], [13], and precise control [14]–[16]. However, most existing approaches have been developed for specific tasks with corresponding algorithms and hardware [17]. Developing integrated planning frameworks that can autonomously accomplish multiple manipulation skills, particularly those requiring aggressive maneuvers, remains challenging.

Unlike decoupled approaches that separately plan UAV and manipulator trajectories, coupled planning enables coordinated motion generation for the entire system, preserving its full maneuverability for complex tasks [18]. However, coupled planning for aerial manipulators presents significant challenges, with the most fundamental stemming from the high-dimensional state space. The inclusion of additional joints from the manipulators greatly increases the system's

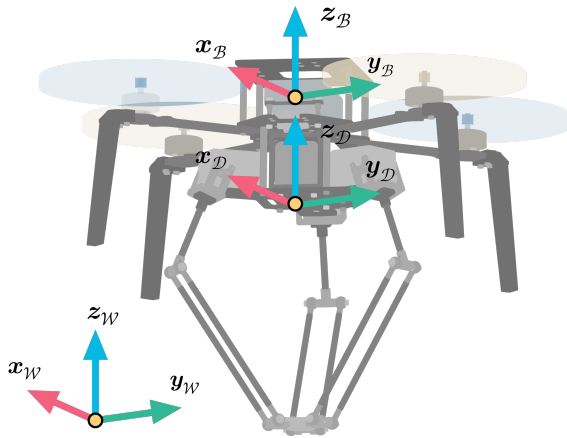


Fig. 2. Coordinate frames of the aerial manipulator system: World frame ($\mathcal{F}_W$), Body frame ($\mathcal{F}_B$), and Delta frame ($\mathcal{F}_D$).

dimensionality, resulting in a vastly expanded search space. To address the challenges of high-dimensional motion planning, various approaches have been developed. Early work proposes an optimization-based framework for aerial manipulators, focusing on aerial grasping of moving targets [19]. However, this approach primarily prioritizes time minimization while overlooking other critical performance metrics, such as energy consumption. The method in [20] performs trajectory search in the task space and employs a controller as a local planner to generate whole-body motions. However, this approach lacks global optimization considerations and therefore cannot guarantee overall trajectory smoothness or energy efficiency. The approach in [21] integrates end-effector and UAV dynamics within an NMPC framework for aerial writing tasks. More recently, a discrete-mechanics-based time-optimal trajectory planning framework with complementarity constraints was developed for aerial manipulators [4]. However, both approaches are designed for specific single tasks and their formulations make them difficult to systematically determine more constrained intermediate states, for example striking a point or being inclined at a certain angle, and lack flexibility to address broader manipulation tasks.

To enable versatile aerial manipulation, a planning framework should be capable of accommodating various simple skills, including grasping, striking, pushing, pulling, writing, and so on [22], [23]. Intuitively, these complex skills can be naturally broken down into fundamental subtasks: point traversing for the quadrotor, point reaching for the end-effector, and combining with extra velocity and orientation requirements as needed at these points. Therefore, we introduce flexible partial waypoint constraints, which allow selective specification of position, velocity, or orientation requirements for individual quadrotors or end-effectors at different waypoints, rather than imposing complete state constraints at every point. However, the diverse types of waypoints pose significant challenges due to the need to satisfy precise intermediate-state requirements while maintaining feasibility across the

entire trajectory [24]. Existing methods for handling waypoint constraints fall into two main categories: multi-stage sampling and optimization integration methods. Multi-stage sampling divides the trajectory into segments by setting multiple start and goal points [25]. While this approach can effectively handle partial waypoint constraint problems, it primarily focuses on position without considering orientation information. Introducing orientation constraints may lead to a significantly enlarged search space or even result in the inability to find feasible solutions. For optimization integration methods, several approaches have been proposed with varying strategies. The method in [26] treats both polynomial trajectories and waypoint arrival times as optimization variables, then penalizes the distance between the end-effector and target points at those specific times. This method provides valuable insights, but for multiple waypoint constraints, it should introduce more time optimization variables to determine the waypoint constraints scope and more variable constraints which may make it difficult to solve. Gaussian penalty functions have been used as soft constraints to incorporate waypoint constraints into MPC [27]. However, since arrival times are provided by higher-level planners, this approach still lacks adaptability and an inappropriate time may lead to an unfeasible solution. Recently, a general spatio-temporal optimization framework with geometric constraints, named MINCO [28], has been proposed to address whole-body trajectory generation for quadrotors. It operates by optimizing the positions of intermediate waypoints and the intervals between these points and then solves the complete trajectory. This optimization structure makes it particularly well-suited for incorporating waypoint constraints on these intermediate points, which forms the foundation for our implementation approach.

For most skills, given a target waypoint along with some orientation or velocity constraints, it is generally possible to optimize a feasible trajectory that accomplishes the task. However, for highly dynamic maneuvers, such as parallel grasping or striking on an inclined plane, as shown in Figure 1 (a), directly using a single waypoint constraint may converge to poor suboptimal solutions. For such highly attitude-constrained trajectories, there exist inherent underlying motion patterns that should be leveraged. For example, quadrotors tend to follow arc-shaped trajectory patterns to satisfy their dynamics, as shown in Figure 1 (b), with this tendency becoming more pronounced as the attitude angle increases. Therefore, we propose a novel guided optimization approach for aerial manipulations that initially imposes additional waypoint constraints to steer the trajectory optimization toward desired local optima that better satisfy both task requirements and dynamic feasibility constraints, and then progressively relaxes these auxiliary constraints to allow for further refinement. The question then becomes: how should we properly select these waypoints? Recently, IL has demonstrated significant capabilities in the robot manipulation domain. For instance, Diffusion Policy [29] learns data distributions through imitation learning to achieve policy learning, while OpenVLA [30] and π_0 [31] achieve task generalization through imitation learning on large-scale robot training data. A recent work, Flying Hand [17], is the first to incorporate teleoperation into aerial manipulators, training

an ACT [32] policy and achieving some basic tasks. Current mainstream IL approaches typically learn to output single-step actions or complete trajectories based on environmental information. However, we aim to leverage this capability of imitation learning through a lightweight network that imitates our aerial manipulator's local behavior near target points when performing large-attitude manipulation, using this information as a prior for guided optimization.

In this paper, we propose a whole-body integrated planning framework for aerial manipulators that incorporates flexible waypoint constraints which can be used to freely combine into many fundamental skills. First, we develop an improved safe flight corridor (SFC) generation strategy that not only expands the feasible solution space but also guarantees the feasible waypoint constraint implementation. Second, we modify the dimension and the dynamics from [28] and then formulate a spatio-temporal optimization problem for aerial manipulators to generate a 6-dimensional trajectory, which includes the quadrotor's trajectory in the world frame and the manipulator's trajectory in the local frame. In addition, we improve the constraint formulation from [28], specifically by introducing a varying ellipsoid to model the changing collision volume of the aerial manipulator caused by the motion of the manipulator. Third, we introduce a comprehensive approach for handling waypoint constraints and incorporate various task-specific constraints, including partial or complete position constraints (with individual x , y , z axis constraint), velocity, and orientation requirements. Building upon this constraint handling approach, we further introduce the concept of IL-guided optimization that employs IL to generate guide points, which serve as waypoint constraints to improve trajectory convergence. Finally, we validate our framework through comprehensive comparative and ablation studies in simulation, as well as real-world aerial manipulation experiments to demonstrate its effectiveness. The main contributions of this work are:

- 1) **Whole-body Planning Framework:** We propose a novel whole-body integrated motion planning framework for aerial manipulators. This framework simultaneously optimizes the trajectory of both the quadrotor and the manipulator to achieve coordinating whole-body motion and complete a series of tasks, strike, grasp, lift, write, pull, push, wind, cross and press. Additionally, we develop a novel varying ellipsoid method to model the changing collision volume to guarantee the whole-body collision-free manipulations under any requirements.
- 2) **Flexible Partial Waypoint Constraint:** We introduce a comprehensive waypoint constraint formulation that enables selective specification of position, velocity, or orientation requirements for individual quadrotors or end-effectors at different waypoints. This approach allows flexible task-specific partial waypoint constraints that only constrain the necessary degrees of freedom while leaving others free for optimization, providing significantly greater flexibility and more constraint options compared to existing methods.
- 3) **IL-Guided Optimization for Complex Tasks:** We introduce an IL-guided optimization approach to overcome

poor local optima in challenging manipulation scenarios, such as grasping or striking on inclined surfaces. The method learns from high-quality demonstration data to generate a sequence of waypoints to guide the optimization process in the first stage, and then progressively relaxes these guided constraints in the second stage to find better solutions.

- 4) **Comprehensive Validation and Open-source Implementation:** We demonstrate the versatility and effectiveness of our framework through extensive validation, including diverse benchmark tasks, ablation studies, simulations on different types of manipulators, and real-world aerial manipulation experiments. The complete framework is made publicly available to support the research community and promote reproducible research in aerial manipulation planning.¹

Our framework overview is depicted in Figure 3. The paper is organized as follows. Section II introduces the notation, system model, and trajectory representation. Section III presents the improved SFC generation strategy, while Section IV elaborates on the basic optimization problem formulation. Section V introduces the implementation of waypoint constraint and IL-guided optimization. Section VI validates the effectiveness of the proposed method through simulations and experiments. Finally, conclusions are presented in Section VII.

II. PRELIMINARIES

A. Notation

The aerial manipulator system discussed in this paper consists of a quadrotor platform integrated with a 3-DOF parallel delta robot with 3-RSS configuration similar to [33], as illustrated in Figure 2. We experimentally determine the Center of Mass (CoM) of the entire aerial manipulator system by placing the delta arm in its initial state and measuring the resulting CoM, which we denote as c_{m0} . For convenience, we define three distinct coordinate frames:

- 1) **World Frame ($\mathcal{F}_W$):** This frame is denoted by $\{x_W, y_W, z_W\}$, where z_W points upwards, opposing the direction of gravity.
- 2) **Body Frame ($\mathcal{F}_B$):** This frame is represented by $\{x_B, y_B, z_B\}$. The origin of this frame is located at c_{m0} . In this frame, x_B points forward and z_B aligns with the total thrust generated by the rotors.
- 3) **Delta Frame ($\mathcal{F}_D$):** This frame is specified as $\{x_D, y_D, z_D\}$ and is related to the body frame $\mathcal{F}_B$ through a translational transformation only.

Unless explicitly stated otherwise, the pre-superscript of a vector indicates the frame in which the vector is expressed. For conciseness, when a vector is expressed in the world frame, the pre-superscript will be omitted. For instance, p_b represents the position of the CoM of the aerial manipulator, in $\mathcal{F}_W$, while ${}^D p_e$ denotes the position of the end-effector in $\mathcal{F}_D$. Analogously, the matrix $\mathbf{R}_B$ represents the rotation from $\mathcal{F}_W$ to $\mathcal{F}_B$.

¹<https://github.com/SYSU-HILAB/am-planner.git>

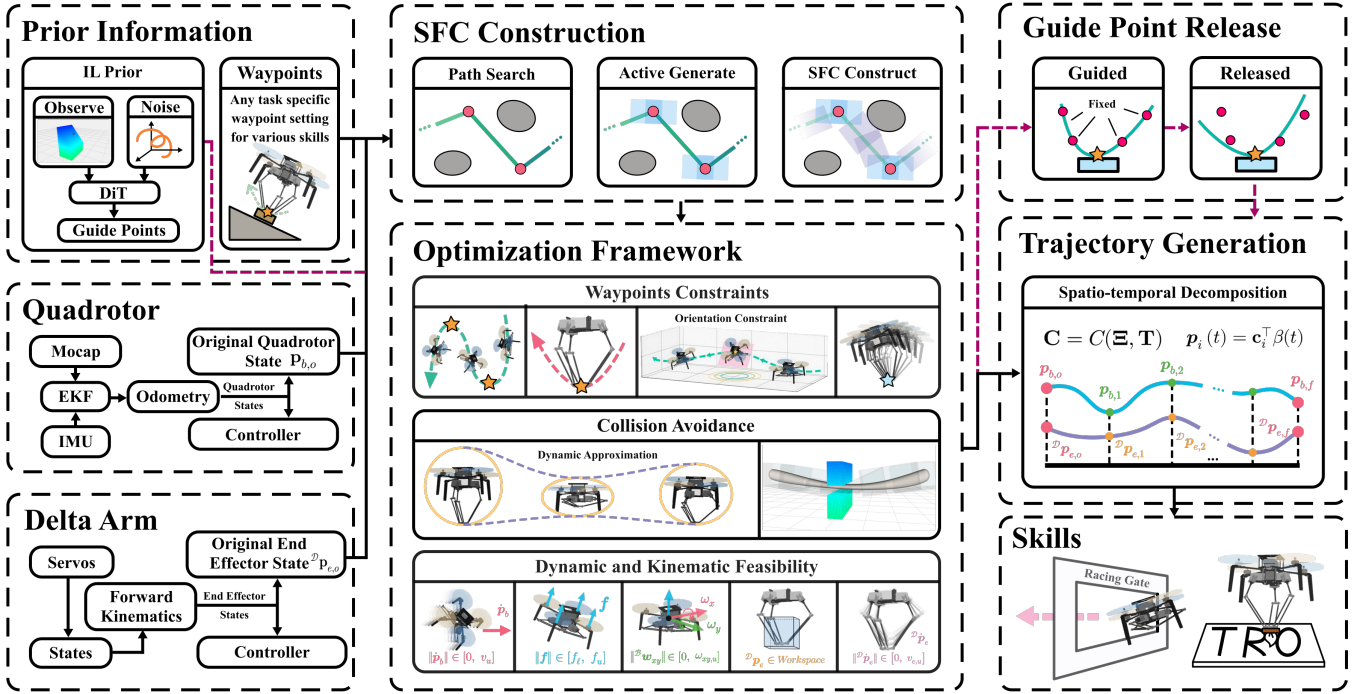


Fig. 3. Overview of the proposed integrated motion planning framework for aerial manipulation. The framework consists of seven key components: (a) Prior Information module that employs a Diffusion Transformer to generate trajectory tendencies and processes user-defined waypoint constraints; (b) Quadrotor module integrating motion capture (Mocap), Extended Kalman Filter (EKF), and IMU for robust state estimation; (c) Delta Arm module handling servo control and forward kinematics for Cartesian space state computation; (d) Safe Flight Corridor (SFC) Construction module for path searching, active polyhedron generation, and conventional SFC construction; (e) Optimization Framework addressing waypoint constraints, collision avoidance, and dynamic/kinematic feasibility; (f) IL-guided two-stage optimization where guide points serve as hard constraints in the initial stage and are subsequently released for free optimization; (g) Trajectory Generation module producing smooth polynomial trajectories; and (h) Skills module performing various basic manipulation skills.

Let $\mathbf{p}_o = [\mathbf{p}_{b,o}^\top, \mathbf{p}_{e,o}^\top]^\top \in \mathbb{R}^6$ and $\mathbf{p}_f = [\mathbf{p}_{b,f}^\top, \mathbf{p}_{e,f}^\top]^\top \in \mathbb{R}^6$ denote the origin and final positions of the aerial manipulator in $\mathcal{F}_W$ and end-effector in $\mathcal{F}_D$, respectively. We define a waypoint constraint set in frame $\mathcal{F}_W$ that consists of arbitrary combinations of two types of waypoint constraints:

$$\mathbf{W}_{\text{cons}} := \{\mathbf{w}_\lambda \mid \lambda \in \{1, \dots, n_\Lambda\}\} \quad (1)$$

The two types of waypoint constraints are defined as:

$$\mathbf{w}_\lambda := \begin{cases} \mathbf{p}_{b,\lambda}^{\text{cons}}, & \text{if } \lambda \in \Lambda_b \\ \mathbf{p}_{e,\lambda}^{\text{cons}}, & \text{if } \lambda \in \Lambda_e \end{cases} \quad (2)$$

where Λ_b and Λ_e are disjoint index sets with $\Lambda_b \cup \Lambda_e = \{1, \dots, n_\Lambda\}$, representing quadrotor and end-effector waypoint constraints, respectively.

B. Modeling

We use Newton-Euler equations to model the whole-body dynamics. Following [34], we treat the aerial manipulator as a rigid body system and focus on quasi-static forces from manipulator motion and environmental interactions. This approach is reasonable as the delta arm's mass represents only a small fraction of the total system mass [35], and we will strictly limit the manipulator's velocity during planning. For instantaneous actions such as grasping and striking, we cannot model these transient disturbances, but we do model

the environmental external forces, that occur during sustained operations like pulling and holding grasped objects.

1) **Aerial Manipulator Dynamics:** The coupled dynamics of the aerial manipulator system are described by:

$$\dot{\mathbf{p}}_b = \mathbf{v}_b \quad (3)$$

$$\dot{\mathbf{v}}_b = \frac{1}{m_c} \mathbf{R}_B (\mathbf{f} + \mathbf{f}_e) - g \mathbf{z}_W \quad (4)$$

$$\mathbf{I}_c \dot{\boldsymbol{\omega}}_b = \mathbf{I}_c^{-1} (\mathbf{B} \boldsymbol{\tau} + \mathbf{B} \boldsymbol{\tau}_e - \mathbf{B} \boldsymbol{\omega}_b \times \mathbf{I}_c \cdot \mathbf{B} \boldsymbol{\omega}_b) \quad (5)$$

$$\mathbf{I}_c = \mathbf{I} + m_e \cdot \text{diag}(\mathbf{p}_e - \mathbf{p}_{e0})^2 \quad (6)$$

where $\mathbf{p}_b$ and $\mathbf{v}_b$ represent the position and linear velocity of the aerial manipulator, respectively. The combined system mass and the end-effector mass are denoted by m_c and m_e respectively. $\mathbf{f}$ and $\boldsymbol{\tau}$ are the thrust and torque generated by the quadrotor, while $\mathbf{f}_e$ and $\boldsymbol{\tau}_e$ are the force and moment generated by the end-effector. $\mathbf{I}_c$ is the combined inertia, where $\mathbf{I} = \text{diag}(I_x, I_y, I_z)$ is the inertia tensor of the aerial manipulator with the arm at its initial state. Here, $\mathbf{p}_{e0}$ and $\mathbf{p}_e$ denote the initial and current positions of the end-effector in the body frame $\mathcal{F}_B$, respectively.

Now we consider the force $\mathbf{f}_e$ and moment $\boldsymbol{\tau}_e$ caused by the end-effector:

$$\mathbf{f}_e = \mathbf{R}_W \mathbf{f}_{\text{ext}} \quad (7)$$

$$\boldsymbol{\tau}_e = \mathbf{p}_e \times \mathbf{f}_e + (\mathbf{p}_e - \mathbf{p}_{e0}) \times (\mathbf{R}_W \cdot m_e g \mathbf{z}_W) \quad (8)$$

where $\mathbf{f}_{\text{ext}}$ is the external forces applied to the end-effector, for example, the gravitational force when grasping an object, or the frictional force when pulling or pushing.

2) **Delta arm Kinematics:** Our quadrotor incorporates a 3-DOF delta arm mounted directly beneath the vehicle. This configuration offers significant advantages in terms of lightweight construction, cost-effectiveness, and operational speed, making it widely adopted in recent research [2], [35]. The kinematic model for the delta arm is formulated as follows:

$$\mathbf{p}_e = \mathbf{R}_B({}^B\mathbf{p}_D + {}^D\mathbf{p}_e) + \mathbf{p}_b \quad (9)$$

$$\mathbf{v}_e = [\boldsymbol{\omega}_B]^\times \mathbf{R}_B({}^B\mathbf{p}_D + {}^D\mathbf{p}_e) + \mathbf{R}_D {}^D\mathbf{v}_e + \mathbf{v}_b \quad (10)$$

where $\mathbf{p}_e$ and $\mathbf{v}_e$ represent the position and velocity of the end-effector in frame $\mathcal{F}_W$, and $[\cdot]^\times$ denotes the skew-symmetric matrix operator.

Since the above representation operates in Cartesian space, inverse kinematics (IK) must be employed to establish the mapping between the end-effector position ${}^D\mathbf{p}_e$ and the joint angles $\mathbf{q} = [q_1, q_2, q_3]^\top \in \mathbb{R}^3$. In this formulation, L_u and L_l denote the lengths of the upper arm and lower arm, respectively, while r_s represents the circumradius of the static platform and r_d represents the circumradius of the end-effector. Following the methodology outlined in [36], the relationship between the end-effector position in the delta frame ${}^D\mathbf{p}_e$ and the joint angles $\mathbf{q}$ is obtained by solving the following equation:

$$\left\| {}^D\mathbf{p}_e - {}^D\mathbf{R}_i \cdot \begin{bmatrix} r_s + L_u \sin q_i \\ 0 \\ -L_u \cos q_i \end{bmatrix} \right\|_2^2 = L_l^2, \quad (11)$$

where $i \in \{1, 2, 3\}$, and ${}^D\mathbf{R}_i$ denotes the z -axis rotation matrix by $\{0, \frac{2}{3}\pi, \frac{4}{3}\pi\}$ radians, respectively. By differentiating both sides of the above equation with respect to time and rearranging the terms, we obtain:

$$\dot{\mathbf{q}} = \mathbf{J} {}^D\dot{\mathbf{p}}_e, \quad (12)$$

where $\mathbf{J}$ denotes the Jacobian matrix. The explicit expression for this matrix is detailed in [36].

C. Trajectory Representation

In this study, we adopt a hybrid coordinate representation for trajectory planning: the quadrotor's trajectory is planned in the frame $\mathcal{F}_W$ while the end-effector trajectory is planned in the frame $\mathcal{F}_D$. This definition offers two key advantages: (1) it enables better capture of the end-effector's motion relative to the quadrotor, facilitating accurate collision volume modeling and workspace constraint enforcement; (2) when no manipulation tasks are required, the system can directly operate as a standard quadrotor without representation switching.

Based on this design, we utilize 6-dimensional spatio-temporal polynomial trajectories that decompose multi-segment trajectories into waypoints and their corresponding time intervals. We denote the trajectory as:

$$\mathcal{T} = \{\mathbf{p}(t) : [0, T_i] \mapsto \mathbb{R}^m \mid \mathbf{C} = C(\mathbf{q}, \mathbf{T}), i = 1, \dots, M\} \quad (13)$$

where $m = 6$ represents the dimension of the trajectory. $\mathbf{C} = [\mathbf{c}_1^\top, \dots, \mathbf{c}_M^\top]^\top \in \mathbb{R}^{2Ms \times 6}$ denotes the polynomial coefficient matrix, where M represents the number of segments in the polynomial trajectory and s represents the order of the integrator chain for control effort. $\mathbf{q} = [q_1, \dots, q_{M-1}]^\top \in \mathbb{R}^{(M-1) \times 6}$ represents the intermediate points, where the first 3 dimensions correspond to the quadrotor and the last 3 dimensions correspond to the end-effector. $\mathbf{T} = [T_1, \dots, T_M]^\top \in \mathbb{R}_{\geq 0}^M$ represents the non-negative time vector. $C(\mathbf{q}, \mathbf{T})$ denotes the parameter mapping constructed from Theorem 2 in [28].

For the i -th segment of the trajectory, the position at time t is defined as:

$$\mathbf{p}_i(t) = \mathbf{c}_i^\top \boldsymbol{\beta}(t), \quad t \in [0, T_i], \quad (14)$$

where $i \in \{1, 2, \dots, M\}$, and T_i represents the duration of the i -th segment. The polynomial coefficient matrix for the i -th segment is denoted by $\mathbf{c}_i \in \mathbb{R}^{2s \times 6}$, and the basis vector function $\boldsymbol{\beta}(x)$ is defined as $\boldsymbol{\beta}(x) = [1, x, \dots, x^{2s-1}]^\top \in \mathbb{R}^{2s}$.

Algorithm 1 Active SFC Generation

```

1: Input: path,  $\{\mathbf{p}_{x,\lambda}^{\text{cons}}\}_{\lambda=1}^{n_\Lambda}$  where  $x \in \{b, e\}$ 
2: Output:  $\mathcal{S}$  //  $\mathcal{S}$ : Corridor
3:  $\mathcal{S} \leftarrow \emptyset, \mathcal{Q} \leftarrow \emptyset$  //  $\mathcal{Q}$ : Polyhedron queue
4: for  $\lambda = 1$  to  $n_\Lambda$  do
5:    $(\mathcal{P}_1, \mathcal{P}_2) \leftarrow \text{ACTIVEGENERATE}(\mathbf{p}_{x,\lambda}^{\text{cons}})$ 
6:    $\mathcal{Q}.\text{push}(\mathcal{P}_1), \mathcal{Q}.\text{push}(\mathcal{P}_2)$ 
7: end for
8:  $i \leftarrow 1$ 
9: while  $i < |\text{path}|$  do
10:  for  $j = i + 1$  to  $|\text{path}|$  do
11:    if  $\text{ISFARTHESTPOINT}(\text{path}[i], \text{path}[j])$  then
12:       $\mathcal{S}.\text{push}(\text{GENERATE}(\text{path}[i], \text{path}[j]))$ 
13:       $i \leftarrow \text{LASTPOINTINPOLY}(\text{path}, \mathcal{S}[-1])$ , break
14:    end if
15:    if  $\mathcal{Q} \neq \emptyset$  and  $\text{INSIDEPOLY}(\mathcal{Q}[0], \text{path}[j])$  then
16:       $\mathcal{S}.\text{push}(\mathcal{Q}.\text{pop}()), \mathcal{S}.\text{push}(\mathcal{Q}.\text{pop}())$ 
17:       $i \leftarrow \text{LASTPOINTINPOLY}(\text{path}, \mathcal{S}[-1])$ , break
18:    end if
19:  end for
20: end while
21: return  $\mathcal{S}$ 

```

III. ACTIVE SFC GENERATION

Given the constrained waypoint positions $\mathbf{p}_{b,\lambda}^{\text{cons}}$ and $\mathbf{p}_{e,\lambda}^{\text{cons}}$ ($\lambda \in \{1, 2, \dots, n_\Lambda\}$), our objective is to generate a safe and feasible trajectory for the aerial manipulator while satisfying all waypoint constraints. However, according to [37], waypoints should intersect with non-empty regions formed by the intersection of two consecutive polyhedra. To achieve this objective, we propose a modified SFC generation strategy that combines multi-stage path searching with an active polyhedron generation method at constrained points, which not only enables waypoint constraint implementation but also enlarges the solution space. The complete algorithm is presented in Algorithm 1.

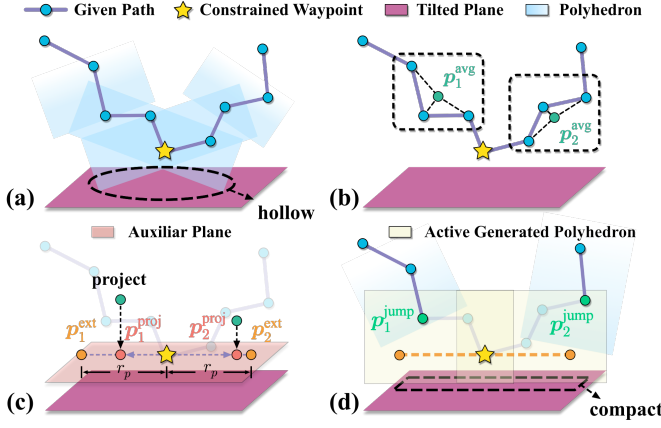


Fig. 4. Active polyhedrons generation algorithm: (a) hollow spaces from SFC generation on inclined surfaces; (b) computation of averaged points p_1^{avg} and p_2^{avg} from path points; (c) projection onto auxiliary planes yielding p_1^{proj} , p_2^{proj} and extensions p_1^{ext} , p_2^{ext} ; (d) polyhedron generation using constrained waypoints with points p_1^{ext} and p_2^{ext} for compact adherence to the surface.

A. Active Polyhedron Generation

Prior to the SFC construction, we leverage existing path searching algorithms JPS [38] to obtain a reference path that sequentially traverses through all waypoints $p_{b,\lambda}^{cons}$ and $p_{e,\lambda}^{cons}$ from the start to the end. Traditional construction methods [37], [39] directly generate a sequence of continuously intersecting polyhedrons by identifying and linking the farthest unblocked points along the reference path. However, obstacle occlusion and environmental factors introduce significant uncertainties during the polyhedron generation process. These uncertainties make it difficult to accurately locate intersection regions and can lead to reduced feasible solution spaces, such as hollow areas that may form between surfaces and polyhedrons, as illustrated in Figure 4 (a). To address the challenge, we develop an active polyhedron generation strategy that extends the polyhedron around constrained waypoints of interest, inspired by the method proposed in [40].

As illustrated in Figure 4 (a), we define an arbitrary and winding purple path that orderly connects the start point $p_{b,o}$, several constrained waypoints ($p_{b,\lambda}^{cons}$ or $p_{e,\lambda}^{cons}$), and the end point $p_{b,f}$. For clarity, we focus on the situation at one of the constrained points. According to [37], the convex polyhedron generation is strongly related to the input line segment, obstacle distribution, and the bounding box configuration. This relationship naturally leads us to consider a key geometric constraint: constructing two line segments that connect the constrained waypoint to its previous and subsequent points can guarantee that the two corresponding polyhedrons form a non-empty intersection region containing the constrained waypoint. More crucially, if the expanding ellipsoid's x -axis is aligned parallel to the inclined plane, the bounding box can be optimally oriented to maximize coverage of the available space on the slope, thereby enlarging the feasible solution space for trajectory optimization. To determine the directions towards and away from the constrained points, we calculate the average of a given number of points before and after the constrained point along the path, denoted as p_1^{avg} and

p_2^{avg} respectively, as shown in Figure 4 (b). These points are then projected onto an auxiliary plane parallel to the previous plane but passing through the constrained waypoint, resulting in two projected vectors p_1^{proj} and p_2^{proj} as illustrated in Figure 4 (c). After normalizing and scaling these vectors by r_p , we obtain two new generation points, p_1^{ext} and p_2^{ext} . By connecting the constrained point to these generation points, we can directly generate two polyhedrons that are tightly fitted to the inclined plane, as illustrated in Figure 4 (d). These constrained polyhedrons are then stored in a temporary queue $\mathcal{Q}$. Additionally, we define two jump points, p_1^{jump} and p_2^{jump} , which are the first and last points along the path within these two polyhedrons respectively. Their specific applications will be discussed in the following subsection.

B. SFC Generation

After generating two polyhedrons for each constrained point as described above, the complete SFC is constructed through an iterative process along the path. During this iteration, when the current point reaches a predefined threshold distance from the last point, the algorithm pauses to generate a new polyhedron and then resumes iteration from the last point within this newly generated polyhedron. When the iterative point reaches p_1^{jump} , the iteration immediately pauses, and the two constrained polyhedrons are popped from the temporary queue $\mathcal{Q}$ and incorporated into the SFC. The iteration then resumes from the corresponding p_2^{jump} and continues the standard generation process. Ultimately, we obtain the SFC in $\mathcal{H}$ -representation as a series of consecutively intersected polyhedrons [41]:

$$\mathcal{S} := \bigcup_{i=1}^M \mathcal{H}_i \subset \mathbb{R}^3 \quad (15a)$$

$$\mathcal{H}_i := \{x \in \mathbb{R}^3 \mid \mathbf{A}_i x \preceq \mathbf{b}_i\} \quad (15b)$$

where M is the total number of polyhedrons in the SFC.

IV. BASIC PROBLEM FORMULATION

This section presents our basic optimization problem formulation for aerial manipulators, which guarantees trajectory smoothness, safety, and dynamic and kinematic feasibility without considering task-specific waypoint constraints. We begin by formulating a comprehensive objective function that balances control effort and time consumption metrics. The objective function is subject to multiple constraint categories: collision-free constraints implemented through a varying ellipsoid, and maximum velocity, acceleration, thrust, body rate, and workspace constraints.

A. Objective Function

We focus on generating a 6-dimensional trajectory $p(t) = [p_b^T(t), p_e^T(t)]^T \in \mathbb{R}^6$ for an aerial manipulator. As discussed in Section II-C, we decompose the complete trajectory into intermediate points and intervals. For convenience, we

separate these intermediate points for the quadrotor and the manipulator into:

$$\mathbf{P} = [\mathbf{p}_{b,0}, \mathbf{p}_{b,1}, \dots, \mathbf{p}_{b,M}]^\top \in \mathbb{R}^{(M+1) \times 3}, \quad (16)$$

$$\mathbf{Q} = [\mathbf{p}_{e,0}, \mathbf{p}_{e,1}, \dots, \mathbf{p}_{e,M}]^\top \in \mathbb{R}^{(M+1) \times 3}. \quad (17)$$

Accordingly, the coefficient matrix is also decomposed as $\mathbf{C} = [\mathbf{c}_b, \mathbf{c}_e]$. Combined with the previously mentioned time vector $\mathbf{T}$, we can formulate the optimization problem as:

$$\min_{\mathbf{P}, \mathbf{Q}, \mathbf{T}} \mathcal{J} = \int_0^{T_\sigma} \|\mathbf{p}^{(s)}(t)\|^2 dt + \rho T_\sigma \quad (18a)$$

$$\text{s.t. } \mathcal{G}(\mathbf{p}(t), \dots, \mathbf{p}^{(s)}(t)) \preceq \mathbf{0}, \quad \forall t \in [0, T_\sigma] \quad (18b)$$

$$\mathbf{p}_1^{[s-1]}(0) = \mathbf{p}_o, \quad \mathbf{p}_M^{[s-1]}(T_M) = \mathbf{p}_f \quad (18c)$$

$$T_i > 0, \quad i \in \{1, 2, \dots, M\} \quad (18d)$$

where $T_\sigma = \sum_{i=1}^M T_i$ is the total time consumption with regularization factor $\rho > 0$, and $\mathcal{G}$ denotes the inequality constraints over $[0, T_\sigma]$. The notation $\mathbf{p}^{[s-1]}$ represents the multi-order differentials matrix $[\mathbf{p}, \dot{\mathbf{p}}, \dots, \mathbf{p}^{(s-1)}] \in \mathbb{R}^{6 \times s}$. Let $\mathbf{p}_o, \mathbf{p}_f \in \mathbb{R}^{6 \times s}$ denote the initial and terminal conditions for the aerial manipulator in $\mathcal{F}_W$ and the end-effector in $\mathcal{F}_D$. T_i denotes the duration of the i -th segment, and M is the total number of segments.

The inequality constraints in Equation (18b) encompass multiple aspects, including safety, dynamic, kinematic and attitude constraints. We adopt a penalty-based approach to handle these constraints efficiently by incorporating them into the objective function with appropriate weights, thereby transforming the original constrained optimization problem into an unconstrained one. To reduce the computational complexity, we approximate the continuous integral using discrete numerical integration:

$$\int_0^{T_\sigma} J_G dt \approx \sum_{i=1}^M \sum_{n=0}^{N-1} \frac{T_i}{N} J_G^i\left(\frac{nT_i}{N}\right) \quad (19)$$

where J_G^i represents the soft penalty terms for the i -th segment, and N denotes the number of discretization points per segment.

B. Control Effort and Total Time Penalty

We define the control effort penalty J_c as the integral of the squared third-order derivatives (i.e., minimum jerk) for both the aerial manipulator and the end-effector. The total control effort is given by:

$$J_c = \int_0^{T_\sigma} \|\mathbf{p}^{(3)}(t)\|^2 dt = \sum_{i=1}^M \int_0^{T_i} \|\mathbf{p}_i^{(3)}(t)\|^2 dt. \quad (20)$$

In addition, to ensure trajectory efficiency, we incorporate a time penalty term J_t , defined as the total trajectory duration:

$$J_t = T_\sigma = \sum_{i=1}^M T_i, \quad (21)$$

where T_i represents the duration of the i -th trajectory segment.

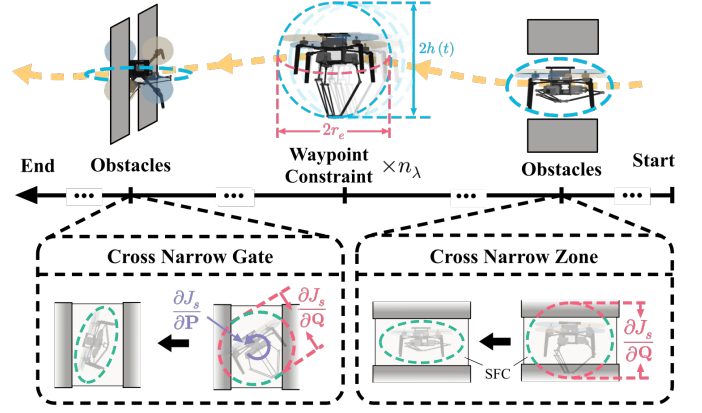


Fig. 5. Visualization of the safety penalty mechanism: The aerial manipulator achieves whole-body collision avoidance by optimizing its configuration through orientation adjustment and delta arm extension, while maintaining task constraints at specified waypoints within the SFC limits.

C. Safety Penalty

We extend the approaches proposed by [41] and [37] by modeling the entire aerial manipulator system as a varying ellipsoid during task execution. The ellipsoid, designed with fixed semi-major axes of length r_e in the xy -plane, incorporates a variational function $h(t)$ to dynamically adjust its length along the z -axis, accounting for the manipulator's motion range and ensuring complete coverage of the entire system, as shown in Figure 5. The formulation of this ellipsoidal model is given by:

$$\mathcal{E}(\xi, t) = \mathbf{G}(t) \cdot (\mathbf{R}_B(t)\xi) + \mathbf{p}_b(t), \quad \xi^\top \xi \leq 1, \quad (22a)$$

$$\mathbf{G}(t) = \text{diag}(r_e, r_e, h(t)) \in \mathbb{R}_{>0}^{3 \times 3}, \quad (22b)$$

$$h(t) = (\mathcal{D}\mathbf{p}_B - \mathcal{D}\mathbf{p}_e(t))^\top \mathbf{e}_3, \quad (22c)$$

where $\xi \in \mathbb{R}^3$ represents any vector within a unit sphere, and $\mathcal{E}$ represents the approximated ellipsoid in the world frame $\mathcal{F}_W$. The matrix $\mathbf{G}(t)$ encodes the semi-lengths of the ellipsoid's three axes along its diagonal. $\mathbf{e}_3 = [0, 0, 1]^\top$ is the unit vector. $\mathcal{D}\mathbf{p}_B$ denotes the translation from $\mathcal{F}_D$ to $\mathcal{F}_B$.

To guarantee safety, we constrain the dynamic ellipsoid to remain within the SFC using the aforementioned methodologies, as shown in Figure 5. This containment constraint can be mathematically expressed as:

$$\mathcal{E} \subset \mathcal{S} \quad (23)$$

where $\mathcal{S}$ is the space defined in Equation (15a). To achieve this, we define the following penalty functions:

$$J_s = \sum_{i=1}^M \sum_{n=0}^{N-1} \sum_{k=1}^{K_i} \frac{T_i}{N} \mathcal{K}(\mathcal{P}_{s,i,n,k}), \quad (24a)$$

$$\mathcal{P}_{s,i,n,k} = \left(\mathbf{p}_{i,n}^k(\tau) - \hat{\mathbf{p}}_i^k \right)^\top \hat{\mathbf{n}}_i^k + d_s \quad (24b)$$

where $\tau = \frac{nT_i}{N}$, a definition that holds for all subsequent derivations. Here, $\mathcal{K}(\cdot)$ denotes the cubic smooth function defined as $\mathcal{K}(x) = \max(x^3, 0)$. For the i -th polyhedron containing K_i half-planes, $\hat{\mathbf{p}}_i^k$ represents an arbitrary point on the k -th half-plane with $\hat{\mathbf{n}}_i^k$ as its outward-pointing normal

vector, and d_s defines a small safety margin. The term $\mathbf{p}_{i,n}^k$ represents the point on the ellipsoid's surface where its normal vector is perpendicular to the k -th plane. This point can be solved by inverting the ellipsoidal tangent point equation and be computed as follows:

$$\begin{cases} \mathbf{p}_{i,n}^k(\tau) = \mathbf{R}_{\mathcal{B},i}(\tau) \cdot {}^{\mathcal{B}}\mathbf{p}_{i,n}^k(\tau) + \mathbf{p}_{b,i}(\tau), \\ {}^{\mathcal{B}}\mathbf{p}_{i,n}^k(\tau) = \pm \frac{\mathbf{G}^2(\tau) \cdot (\mathbf{R}_{\mathcal{B},i}^\top(\tau) \hat{\mathbf{n}}_i^k)}{\|\mathbf{G}(\tau) \cdot (\mathbf{R}_{\mathcal{B},i}^\top(\tau) \hat{\mathbf{n}}_i^k)\|_2}, \\ {}^{\mathcal{B}}\mathbf{p}_{i,n}^k(\tau) \cdot (\mathbf{R}_{\mathcal{B},i}^\top(\tau) \cdot \hat{\mathbf{n}}_i^k) \geq 0. \end{cases} \quad (25)$$

Then, we can easily derive the gradient of the safety cost with respect to the optimization variables $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{T}$.

D. Kinematics and Dynamics Penalty

To ensure the physical feasibility of the generated trajectories, we need to consider both kinematic and dynamic constraints. The kinematic constraints govern the system's workspace, velocities, and body rates, while the dynamic constraints regulate the thrust forces.

We first consider the kinematic constraints. For the delta arm to operate safely and effectively, its end-effector must remain within a prescribed workspace. Following [12], we model this workspace as a cuboid in the Cartesian coordinate system, where $\kappa \in \{x, y, z\}$ denotes any of the three axes. The workspace is bounded by:

$$\kappa_{e,\ell} \leq \mathbf{e}_\kappa^\top \mathbf{D} \mathbf{p}_e(\tau) \leq \kappa_{e,u}, \quad (26a)$$

where $\kappa_{e,\ell}$ and $\kappa_{e,u}$ denote the lower and upper bounds along axis κ , respectively, and $\mathbf{e}_\kappa$ represents the unit vector along the κ -axis. To enforce these workspace constraints, we introduce the following penalty terms:

$$J_k = \sum_{i=1}^M \sum_{n=0}^{N-1} \sum_{\kappa} \frac{T_i}{N} \left(\mathcal{K}(\mathcal{P}_{k_\ell,i,n,\kappa}) + \mathcal{K}(\mathcal{P}_{k_u,i,n,\kappa}) \right), \quad (27a)$$

$$\mathcal{P}_{k_\ell,i,n,\kappa} = (\mathbf{e}_\kappa^\top \mathbf{D} \mathbf{p}_{e,i}(\tau))^2 - \kappa_{e,\ell}^2, \quad (27b)$$

$$\mathcal{P}_{k_u,i,n,\kappa} = \kappa_{e,u}^2 - (\mathbf{e}_\kappa^\top \mathbf{D} \mathbf{p}_{e,i}(\tau))^2. \quad (27c)$$

To prevent excessive motion, we also constrain the velocities of both the aerial manipulator and the end-effector. The aerial manipulator's velocity is limited to an upper bound $v_{b,u}$, while the end-effector velocity is constrained by $v_{e,u}$. These constraints are incorporated through the following penalty terms:

$$J_{k,vel} = \sum_{i=1}^M \sum_{n=0}^{N-1} \frac{T_i}{N} \left(\mathcal{K}(\mathcal{P}_{b,v,i,n}) + \mathcal{K}(\mathcal{P}_{e,v,i,n}) \right), \quad (28a)$$

$$\mathcal{P}_{b,v,i,n} = \|\dot{\mathbf{p}}_{b,i}(\tau)\|^2 - v_{b,u}^2, \quad (28b)$$

$$\mathcal{P}_{e,v,i,n} = \|\dot{\mathbf{p}}_{e,i}(\tau)\|^2 - v_{e,u}^2, \quad (28c)$$

where $\dot{\mathbf{p}}_{b,i}(\tau)$ represents the velocity of the aerial manipulator in $\mathcal{F}_W$, and $\dot{\mathbf{p}}_{e,i}(\tau)$ represents the velocity of the end-effector in $\mathcal{F}_D$.

As another kinematic constraint, we consider the aerial manipulator's body rates. Given that yaw angle variations are

neglected in our framework, we focus only on the body rates in the xy -plane. The corresponding penalty terms are defined as:

$$J_{k,bdr} = \sum_{i=1}^M \sum_{n=0}^{N-1} \frac{T_i}{N} \mathcal{K}(\mathcal{P}_{\omega,i,n}), \quad (29a)$$

$$\mathcal{P}_{\omega,i,n} = \|\mathcal{B}\boldsymbol{\omega}_{xy,i}\|^2 - \omega_{xy,u}^2, \quad (29b)$$

where $\mathcal{B}\boldsymbol{\omega}_{xy,i} = [\omega_x, \omega_y]^\top$ represents the body angular rates in the xy -plane, and $\omega_{xy,u}$ denotes the upper limit of the body rates.

For dynamic constraints, we consider the aerial manipulator's thrust capabilities. The thrust f must remain within physically realizable bounds:

$$f_\ell \leq f \leq f_u, \quad (30)$$

where f_ℓ and f_u denote the lower and upper thrust limits. We formulate the corresponding penalty terms as:

$$J_{d,thr} = \sum_{i=1}^M \sum_{n=0}^{N-1} \frac{T_i}{N} \left(\mathcal{K}(\mathcal{P}_{f_u,i,n}) + \mathcal{K}(\mathcal{P}_{f_\ell,i,n}) \right), \quad (31a)$$

$$\mathcal{P}_{f_u,i,n} = \|m_c(\ddot{\mathbf{p}}_{b,i}(\tau) + g\mathbf{z}_W) - \mathbf{f}_{\text{ext}}\|^2 - f_u^2, \quad (31b)$$

$$\mathcal{P}_{f_\ell,i,n} = f_\ell^2 - \|m_c(\ddot{\mathbf{p}}_{b,i}(\tau) + g\mathbf{z}_W) - \mathbf{f}_{\text{ext}}\|^2, \quad (31c)$$

where $\ddot{\mathbf{p}}_{b,i}(\tau)$ denotes the aerial manipulator's acceleration.

The gradients of all kinematics and dynamics penalty terms with respect to the optimization variables $\mathbf{P}$, $\mathbf{Q}$ and $\mathbf{T}$ can be efficiently computed using the method described in [39], [42].

With the complete formulation of constraints and their corresponding gradients for the aerial manipulator's basic flight planning, we can solve the optimization problem using L-BFGS to obtain collision-free flight trajectories for the aerial manipulator.

V. WAYPOINT CONSTRAINTS AND IL-GUIDED OPTIMIZATION

Having established the basic optimization formulation in the previous section, we now turn to the incorporation of task-specific waypoint constraints, which can be applied to either the quadrotor or the end-effector in the frame $\mathcal{F}_W$ independently, as shown in Figure 6. These constraints include higher-order velocity and orientation requirements at designated waypoints. Building upon this framework, we further propose an IL-guided optimization approach to improve performance for specific manipulation tasks.

A. Waypoint Constraints Implementation

Based on our modified SFC generation strategy, each constrained waypoint corresponds to an intersection of two polyhedrons and an intermediate point. We define the mapping

$$\phi: \{1, \dots, n_\Lambda\} \rightarrow \{1, \dots, M-1\} \quad (32)$$

to map constrained waypoint indices to their corresponding intermediate point indices.

1) Quadrotor Waypoint Constraints: Since the quadrotor trajectory is planned in frame $\mathcal{F}_W$, we exploit the spatio-temporal trajectory decomposition property by fixing certain

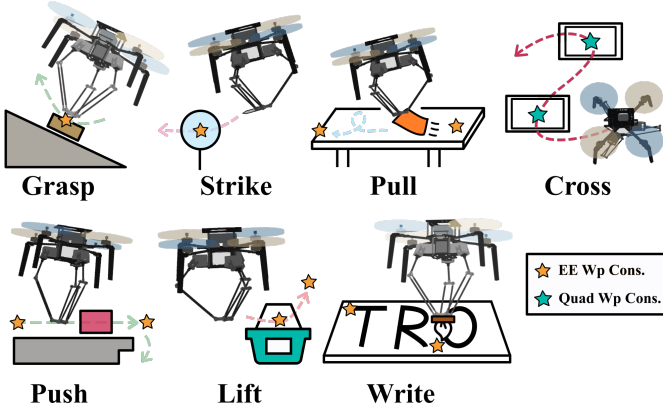


Fig. 6. Illustration of representative manipulation primitives achieved through waypoint constraints. The figure demonstrates various fundamental skills including *Grasp*, *Strike*, *Pull*, *Cross*, *Push*, *Lift*, and *Write*. All these manipulation tasks can be effectively accomplished by implementing appropriate waypoint constraints combined with motion restrictions during execution.

waypoints as constants while maintaining others as optimization variables. The quadrotor waypoint constraints are denoted as:

$$\mathbf{P}_b^{\text{cons}} := \{p_{b,\phi(\lambda)} = p_{b,\lambda}^{\text{cons}} \mid \lambda \in \Lambda_b\}. \quad (33)$$

The modified optimization variables become:

$$\bar{\mathbf{P}} = \mathbf{P} \setminus \mathbf{P}_b^{\text{cons}} \quad (34)$$

where $\setminus$ denotes set difference. Due to the spatio-temporal trajectory properties, the resulting trajectory is guaranteed to traverse all constrained waypoints in $\mathbf{P}_b^{\text{cons}}$.

2) End-Effector Waypoint Constraints: The end-effector trajectory is planned in $\mathcal{F}_D$ and cannot be directly fixed like the quadrotor trajectory. Instead, we enforce end-effector waypoint constraints through a soft penalty approach. We have generated polyhedron intersections near each $p_{e,\lambda}^{\text{cons}}$ to ensure the corresponding trajectory intermediate points pass through their vicinity. Therefore, using the end-effector's world frame position from Equation (9), we formulate the penalty term:

$$J_e = \sum_{\lambda \in \Lambda_e} \left\| p_{e,\phi(\lambda)} - p_{e,\lambda}^{\text{cons}} \right\|^2. \quad (35)$$

B. Task-Specific Constraints

1) Axis-Wise End-Effector Motion Constraint: When executing specific manipulation skills such as pulling, pushing, and writing tasks, the aerial manipulator's end-effector must maintain its position parallel to a defined plane [43]. More restrictively, operations involving sliding mechanisms require the end-effector to remain constrained to a line throughout the motion. These geometric constraints pose significant challenges for trajectory planning, as conventional point-to-point planning methods cannot guarantee such continuous spatial restrictions.

To address these requirements, we introduce axis-wise end-effector position motion constraints that enforce planar or

Algorithm 2 IL-guided Optimization

```

1: Input:  $p_o, p_f, p_{\text{inc}}, o_{\text{inc}}, \epsilon_{\text{loose}}, \epsilon_{\text{strict}}$ 
2: Output:  $p(t), t \in [0, T_\sigma]$ 
3:  $\mathbf{a} \leftarrow \text{IMITATIONLEARNING}(p_{\text{inc}}, o_{\text{inc}}, p_o)$ 
4:  $\mathbf{w}_{\text{guide}} \leftarrow \text{SAMPLEPOINTS}(\mathbf{a})$ 
5:  $\mathbf{w}_{\text{task}} \leftarrow \text{TASKSPECIFIC}(p_{\text{inc}}, o_{\text{inc}})$ 
6:  $\mathbf{x}_1 \leftarrow \text{INIT}(p_o, [\mathbf{w}_{\text{guide}}, \mathbf{w}_{\text{task}}], p_f)$ 
7:  $\mathbf{x}_2 \leftarrow \text{OPTIMIZATION}(\mathbf{x}_1, \epsilon_{\text{loose}}, [\mathbf{w}_{\text{guide}}, \mathbf{w}_{\text{task}}])$ 
8: // Release the IL guided constraints
9:  $\mathbf{x}_3 \leftarrow \text{OPTIMIZATION}(\mathbf{x}_2, \epsilon_{\text{strict}}, \mathbf{w}_{\text{task}})$ 
10: return  $\text{POLYNOMIAL}(\mathbf{x}_3)$ 

```

linear motion during designated trajectory segments. The constraints are formulated as follows:

$$J_{pe} = \sum_{i=1}^M \mathcal{I}_p(i) \sum_{n=0}^{N-1} \frac{T_i}{N} \left\| \mathbf{c}_p^\top (\mathbf{p}_{e,i}(\tau) - \mathbf{p}_{\text{des},\lambda}) \right\|^2 \quad (36)$$

where $\mathcal{I}_p(i) \in \{0, 1\}$ that activates constraints for specific trajectory segments, and $\mathbf{c}_p \in \mathbb{R}^3$ is a user-defined binary vector where setting any dimension to 1 indicates that the corresponding coordinate should be constrained. These axis-wise motion constraints are typically employed in conjunction with two end-effector waypoint constraints to ensure that the constrained dimensions maintain identical values between consecutive waypoints.

2) Velocity Constraint: In certain manipulation tasks, the end-effector's velocity must be constrained to ensure precise operation. For instance, during object grasping maneuvers, the end-effector may need to maintain zero velocity or a small downward velocity along the z_B to achieve stable contact.

The velocity constraint is formulated as:

$$J_{ve} = \sum_{\lambda \in \Lambda_e} \mathcal{I}_v(\lambda) \left\| \mathbf{c}_v^\top (\dot{\mathbf{p}}_{e,\phi(\lambda)} - \dot{\mathbf{p}}_{\text{des},\lambda}) \right\|^2 \quad (37)$$

where $\mathcal{I}_v(\lambda) \in \{0, 1\}$ is a binary selector that activates velocity constraints for waypoint λ , $\mathbf{c}_v \in \mathbb{R}^3$ is a user-defined binary vector specifying which velocity components to constrain, and $\dot{\mathbf{p}}_{\text{des},\lambda}$ is the desired end-effector velocity at the constrained waypoint.

3) Orientation Constraint: Since the end-effector in the delta arm configuration shares the same orientation as the quadrotor, orientation constraints can be naturally imposed on the quadrotor's intermediate waypoints. This inherent coupling eliminates the need to distinguish between quadrotor and end-effector orientations. The orientation constraint is formulated as:

$$J_{oe} = \sum_{\lambda \in \Lambda_e} \mathcal{I}_o(\lambda) \left\| \mathbf{f}_\lambda - \|\mathbf{f}_\lambda\| \mathbf{o}_{\text{des},\lambda} \right\|^2 \quad (38)$$

where $\mathcal{I}_o(\lambda) \in \{0, 1\}$ is a binary selector for activating orientation constraints, $\mathbf{f}_\lambda = m_c(\ddot{\mathbf{p}}_{b,\phi(\lambda)} + g\mathbf{z}_W) - \mathbf{f}_{\text{ext}}$ is the actual thrust vector, and $\mathbf{o}_{\text{des},\lambda}$ is the desired unit orientation vector at the λ -th waypoint.

C. IL Prior

Several complex manipulation tasks requiring extreme orientations, such as inclined surface grasping and striking,

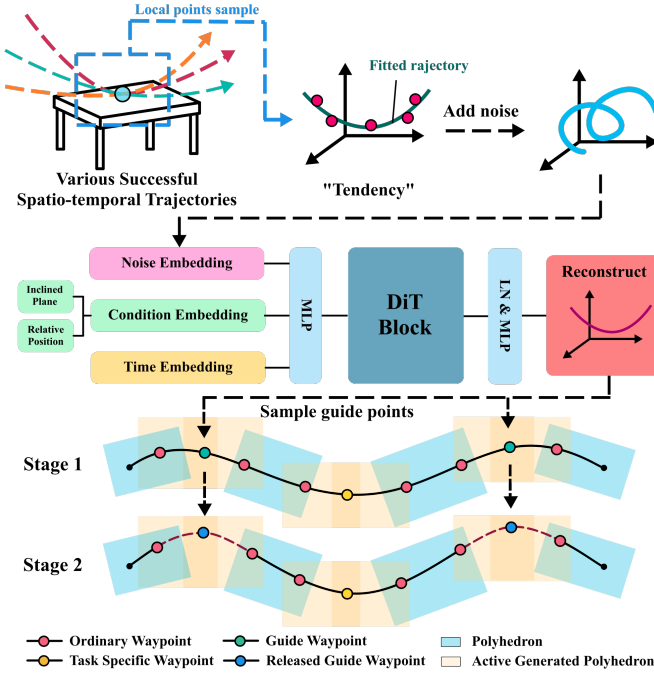


Fig. 7. Visualization of the IL-guided optimization framework. The process begins with collecting various successful spatio-temporal trajectories from demonstration data. These trajectories are fitted using polynomial approximation to extract motion tendencies. A DiT model learns to reconstruct these motion tendencies through noise embedding, condition embedding, and time embedding. The optimization process consists of two stages: Stage 1 employs guide waypoints as hard constraints; Stage 2 releases these hard constraints, allowing free optimization.

present significant challenges for trajectory optimization. These tasks involve inherently conflicting constraints - for example, orientation requirements conflict with maximum body rate limitations, and zero end-effector velocity during grasping conflicts with time penalty objectives. The complexity of these constraint interactions significantly amplifies the non-convex nature of the optimization problem. Since optimization fundamentally involves balancing competing objectives, poor initial solutions can lead the optimizer to converge to local optima that completely sacrifice task constraints [44].

We focus on two tasks requiring aggressive maneuvers: inclined grasping and inclined striking. For these complex tasks, designing simple waypoint constraints w_{task} that contain position p_{inc} and corresponding orientations o_{inc} often results in poor local optima that fail to satisfy overall task requirements. To address this challenge, we analyze successful trajectories from both tasks and discover that they exhibit distinct flight "tendencies." For example, striking tasks consistently demonstrate arc-like trajectory patterns. These observed patterns provide valuable priors that can guide more effective flight planning.

Given the recent influence of imitation learning in trajectory planning [17], we employ IL to incorporate the observed trajectory priors into our planning framework. To promote training efficiency and reduce unnecessary input complexity, we adopt a compact observation space $o := [p_{\text{rel},xy}, \theta_{\text{inc}}]$, where $p_{\text{rel},xy}$ encodes the relative position of the quadrotor in

the horizontal plane with respect to the inclined surface, and θ_{inc} denotes the surface's inclination angle, derived from the surface orientation vector o_{inc} . The action space comprises local polynomial trajectory parameters near the surface, defined as $a := [c_x, c_y, c_z]$. This polynomial parameterization enables a compact network architecture to learn trajectory "tendencies" more effectively and stably than approaches based on discrete trajectory points.

Moreover, for analogical observations, multiple reasonable trajectories may exist. A deterministic network modeling ($o \rightarrow a$) risks learning averaged action patterns that may not represent valid individual trajectories [45]. Diffusion models can naturally handle this multi-modality by modeling the conditional probability $p(a|o)$. Therefore, we design a Diffusion model to address the multi-modal nature of valid trajectories [46]. The model structure can be seen in Figure 7.

We collect many high-quality, effective single-task trajectories as training data. For each clean action a^0 , we perform K Gaussian noise addition steps to obtain noisy action a^K :

$$a^k = \sqrt{\bar{\alpha}_k} a^0 + \sqrt{1 - \bar{\alpha}_k} \epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathbf{I}), \quad k = 1, \dots, K \quad (39)$$

where $\bar{\alpha}_k = \prod_{i=1}^k \alpha_i$ and $\alpha_k = 1 - \beta_k$ with β_k being the noise schedule. The denoising network f_θ is trained to predict the noise:

$$\mathcal{L}(\theta) = \mathbb{E}_{k, a^0, \epsilon} [\|\epsilon - f_\theta(a^k, k, o)\|^2] \quad (40)$$

During inference, we sample from the learned distribution through the reverse process:

$$a^{k-1} = \frac{1}{\sqrt{\alpha_k}} \left(a^k - \frac{\beta_k}{\sqrt{1 - \bar{\alpha}_k}} f_\theta(a^k, k, o) \right) + \sigma_k z \quad (41)$$

where $\sigma_k = \sqrt{\beta_k}$ and $z \sim \mathcal{N}(0, \mathbf{I})$. This reverse diffusion process generates diverse trajectory parameters that capture the multi-modal distribution of successful manipulation strategies.

D. IL-guided Optimization

After learning polynomial trajectories through IL, we leverage the IL prior trajectories as guidance for optimization by incorporating them as quadrotor waypoint constraints. We sample k guide points from the IL prior polynomial trajectory and insert them into Equation (1):

$$w_{\text{guide}} := \{p_{\text{guide},1}, \dots, p_{\text{guide},k}\} \subset \mathbf{W}_{\text{cons}} \quad (42)$$

where $p_{\text{guide},i}$ represents the i -th guide point. These sampled guide points ensure that the aerial manipulator follows the desired trajectory direction, thereby improving success rates to some extent.

However, trajectories constrained to fixed guide points may still be suboptimal due to their rigidity. To address this limitation, we implement a novel two-stage optimization strategy. In the first stage, we enforce guide point constraints with a loose gradient tolerance condition ϵ_{loose} , which ensures that the optimization is guided toward the desired direction during the initial search phase. In the second stage, we remove the guide point constraints and apply strict convergence criteria ϵ_{strict} , allowing the optimizer to discover locally superior solutions

while maintaining the beneficial trajectory structure established in the first stage. This two-stage approach combines the global guidance from successful demonstrations with local optimization refinement, significantly improving both success rates and trajectory quality for complex manipulation tasks. The complete procedure is detailed in Algorithm 2.

VI. EXPERIMENTS

This section presents a comprehensive experimental evaluation of the proposed framework across multiple dimensions. We begin by detailing the imitation learning data collection and training procedures for the DiT model. Subsequently, we conduct ablation studies to validate key components: the SFC enlarge strategy and the IL-guided optimization approach. We then benchmark different collision volume approximation methods, comparing static ellipsoid and multi-sphere fitting approaches. Simulation experiments demonstrate the framework's versatility across various manipulator configurations, including telescopic manipulators, 1-DOF, and 2-DOF robotic arms. Finally, real-world experiments validate the system's performance through the execution of 9 fundamental manipulation skills: *strike*, *grasp*, *lift*, *press*, *wind*, *write*, *cross*, *push*, and *pull*, confirming the practical applicability and versatility of our integrated approach.

A. Data Collection and IL Training

For the IL-guided optimization, we focus on grasp and strike skills. Expert trajectories are generated on inclined plane scenarios with angles uniformly sampled from 20° to 60° . Initial robot positions are randomly sampled within annular regions of 1.5m to 3.0m radius centered on the inclined plane, with a task-specific waypoint constraint for the end-effector positioned at 0.05m directly above the plane. Due to low success rates in direct planning, we apply the two-stage optimization algorithm similar to Section V-D. However, the guide points are strategically selected through expert knowledge before and after critical strike and grasp phases. Generated trajectories with total optimization cost exceeding 1×10^5 are discarded to ensure solution quality.

To obtain stable trajectory tendencies within local regions, we extract trajectory segments of ± 1 s duration around the grasping and striking phases. From each segment, we sample 15 discrete points that are subsequently fitted using a three-dimensional fifth-order polynomial $[c_x, c_y, c_z]$. The resulting 15 polynomial coefficients constitute the trajectory tendency representation that serves as training targets for the diffusion model. Through this process, we collected approximately 3,000 high-quality trajectory samples for each skill type (grasping and striking) along with their corresponding polynomial representations. The data collection process requires an average generation time of about 33s per sample, which allows for practical data collection at scale.

The imitation learning component employs a Diffusion Transformer (DiT) architecture specifically designed for trajectory tendency learning. The model utilizes a 128-dimensional embedding space with 8 attention heads across 6 transformer

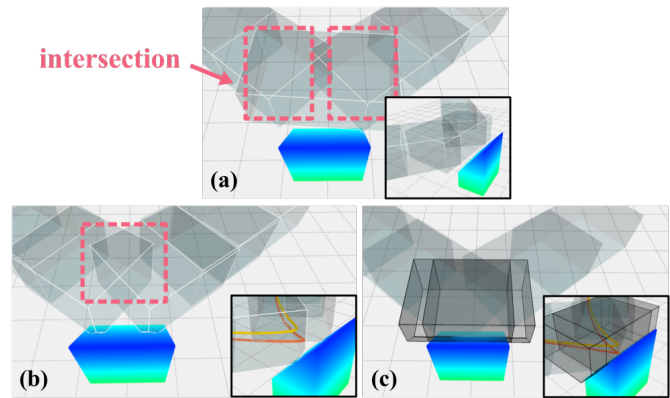


Fig. 8. Comparison of different polyhedron generation methods for inclined surface grasping: (a) original method from [13] without modifications, (b) ablated method with manually constrained intersection above the inclined surface, and (c) our proposed Active Polyhedron Generation method. The pink boxes indicate the locations of adjacent polyhedron intersections.

TABLE I
QUANTITATIVE COMPARISON OF ACTIVE POLYHEDRON GENERATION METHODS

Method	Utilization Rate (%)	Final Cost	Distance (m)
Ablated	27.1	9.51×10^5	0.267
Proposed	100.0	3.54×10^3	0.027

layers, incorporating adaptive layer normalization and cross-attention mechanisms for effective condition-context fusion. With approximately 3M parameters, the network maps 3-dimensional input conditions (relative position $p_{rel,xy}$ and inclination angle θ_{inc}) to 15-dimensional polynomial coefficients representing trajectory tendencies. Training is conducted on an NVIDIA RTX 4090 with 24GB memory for approximately one hour using mixed precision and a learning rate of 3×10^{-4} with cosine annealing.

B. Ablation Study

We designed two ablation experiments to validate the effectiveness of our Active Polyhedron Generation and demonstrate the improvement provided by IL-guided Optimization.

1) Active Polyhedron Generation: To verify that our Active Polyhedron Generation not only maintains compatibility with our waypoint constraint framework but also expands the solution space, we designed an inclined surface grasping scenario with three different configurations: (1) the original method from [13] without any modifications; (2) a modified version that ensures the intersection of two polyhedrons exists directly above the inclined surface to enable waypoint constraint application; and (3) our proposed Active Polyhedron Generation method.

The results are shown in Figure 8. In Figure 8 (a), without predetermined intervention points in SFC generation, the resulting SFC exhibits significant randomness. The polyhedron intersections are located on both sides of the inclined surface, making it difficult to effectively apply waypoint constraints. In Figure 8 (b), although we constrained the intersection to exist

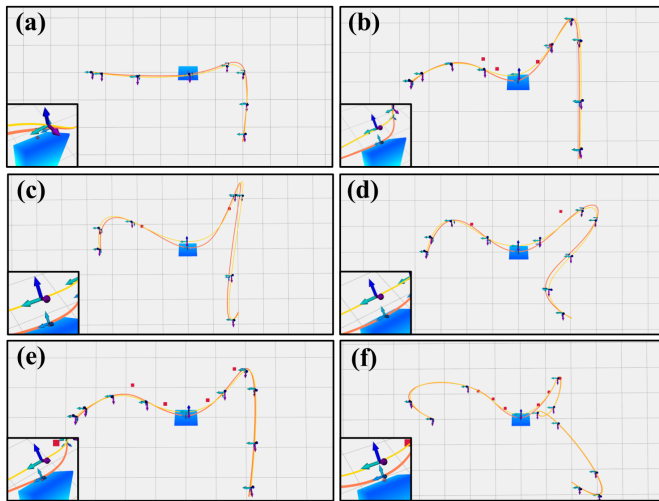


Fig. 9. Convergence results for IL-guided optimization variants. Yellow trajectories represent the motion of the quadrotor, while orange trajectories show the end-effectors movement in $\mathcal{F}_{VV}$. Red points indicate IL-derived guide points, and aerial manipulator orientations at strike instant are shown in the lower-left corners.

directly above the inclined surface, substantial unused space remains, particularly near the center of the inclined surface, resulting in the "hollow" region we mentioned. In contrast, Figure 8 (c) demonstrates our proposed Active Polyhedron Generation, which actively generates polyhedrons that nearly achieve full utilization of the feasible space above the inclined surface.

Since configuration (1) cannot accommodate waypoint constraints, we conducted quantitative analysis only on configurations (2) and (3). We applied our optimization framework to optimize the grasp skill trajectory, with results shown in the lower panels of Figure 8 (a) and (c), where orange trajectories represent the end-effector path and yellow trajectories represent the quadrotor path. Our proposed method successfully reaches the target, while the ablated method suffers from conflicts between end-effector waypoint constraints and safety constraints due to limited solution space. This leads to failure in reaching the target point, and excessive safety penalties in the vicinity result in significantly degraded trajectory quality.

Table I presents a quantitative comparison of three key metrics. The Utilization Rate measures the percentage of free space coverage within a 0.5m radius sphere around the target point. The Final Cost represents the objective function value after convergence. The Distance quantifies the end-effector positioning error at the waypoint constrained location.

2) IL-guided Optimization: To validate the effectiveness of our IL-guided optimization framework, we conduct a comprehensive ablation study examining strike success rates on inclined surfaces. This study addresses three key questions: whether our IL-guided optimization can improve strike success rates, the rationale for sampling points from polynomial trajectories rather than learning discrete points directly, and the optimal number of sampling points for trajectory guidance.

We design the experimental scenario with varying inclined surfaces and designated target points for strike operations.

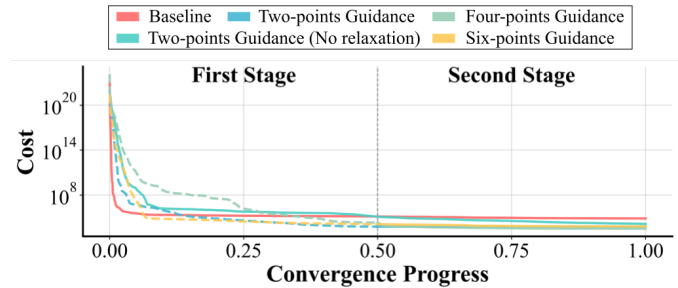


Fig. 10. Convergence curves comparison across optimization methods. Normalized iteration lengths account for varying convergence requirements. Dashed lines: first-stage optimization; solid lines: second-stage optimization.

A successful strike requires two simultaneous conditions at contact: the aerial manipulator maintains parallel alignment with the inclined surface, and the end-effector remains within 0.03m of the target center. Initial positions are randomly sampled within a 1.5 – 3m annular region around the surface for training, while out-of-distribution (OOD) testing uses a 3 – 4m annular region to evaluate generalization performance. We assume prior knowledge of target location and surface inclination angle, establishing fixed end-effector waypoint constraints and corresponding orientations. Our experimental design comprises six configurations tested across four scenarios: 25°, 40°, and 55° surface angles with in-distribution initial positions, 40 with OOD initial positions.

The **Baseline** completely removes IL-guided optimization, relying solely on direct trajectory optimization. The second configuration (**Learned Points Guidance + Relaxation**) trains the IL model to predict four discrete points, maintaining consistency with our training methodology but reducing the projection dimension from 15 to 12. The third approach (**Two-Point Guidance, No Relaxation**) implements our polynomial trajectory imitation learning, sampling one point before and after strike as guide points with strict convergence criteria. Our primary method (**Two-Point Guidance + Relaxation**) extends the third configuration by incorporating relaxed convergence criteria and two-stage optimization, removing guide points in the second stage for stricter optimization. Finally, we evaluate **Four-Point and Six-Point Guidance + Relaxation** variants, differing only in the number of polynomial trajectory sampling points.

Figure 9 illustrates convergence results for challenging cases where initial positions lie behind the inclined surface. The baseline approach in Figure 9 (a) demonstrates complete failure, with the end-effector trajectory (orange) missing the target entirely due to convergence to poor local optima. In contrast, learned point guidance in Figure 9 (b) successfully captures required trajectory tendencies, achieving target strikes. Our two-point sampling approach Figure 9 (c) produces quadrotor trajectories (yellow) that accurately traverse the red guide points, resulting in desired trajectory generation. Both two-point and four-point guidance Figure 9 (d) and (e) consistently accomplish the task, while six-point guidance Figure 9 (f) suffers from over-constraint, leading to suboptimal solutions. Figure 10 quantitatively demonstrates that all proposed IL-guided methods achieve superior local optima compared to

TABLE II
COMPARISON OF ROBOTIC MANIPULATION OPTIMIZATION METHODS ACROSS DIFFERENT ROLL ANGLES

Method	Roll (°)	Performance		Trajectory Quality			Efficiency	
		Success Rate (%)	Cost	Time (s)	Length	Jerk	Iteration	Runtime (s)
Baseline (Complete Ablation)	25	100	2137.4	3.7	6.2	277.7	1536	5.1
	40	50 [†]	4541.7	6.8	9.6	281.1	1726	6.1
	55	0 [†]	—	—	—	—	—	—
	40(OOD)	70	3703.1	4.6	8.5	488.2	2619	7.9
Learned Points Guidance + Relaxation	25	90	3903.5	6.6	8.1	338.5	34306 [†]	132.4
	40	70	5527.5 [†]	10.1 [†]	9.7 [†]	400.8	37209	139.4 [†]
	55	80	5127.2	8.2	11.1	886.0	19653	86.5
	40(OOD)	60	4081.4	6.2	8.8	440.6	47118 [†]	151.2 [†]
Two-Point Guidance (No Relaxation)	25	100	3994.9 [†]	5.8	8.0	401.5 [†]	8272	31.7
	40	70	3243.0	4.7	8.0	579.1 [†]	10653	35.4
	55	100	4137.0	6.1	12.6	974.4 [†]	5590	21.9
	40(OOD)	90	3035.4	4.9	9.7	580.1	6904	27.2
Two-Point Guidance + Relaxation (Ours)	25	100	2568.9	4.4	6.8	317.2	9364	34.6
	40	100	3584.1	5.4	9.0	563.1	16357	50.9
	55	100	4612.0	7.3	13.2 [†]	857.1	6741	27.4
	40(OOD)	100	2719.2	4.2	8.8	584.9 [†]	9566	35.6
Four-Point Guidance + Relaxation (Ours)	25	80	2842.3	4.4	7.3	377.8	22529	88.2
	40	70	3112.3	5.0	8.5	575.8	37550	126.2
	55	70	4376.9	6.9	11.9	852.5	17751	88.4
	40(OOD)	90	4053.7 [†]	7.1 [†]	11.4 [†]	446.8	7460	34.0
Six-Point Guidance + Relaxation (Ours)	25	70 [†]	3838.8	6.9 [†]	8.6 [†]	325.7	33666	172.9 [†]
	40	50	4250.7	6.8	9.1	519.4	56956 [†]	239.6
	55	60	6097.0 [†]	10.5 [†]	12.6	737.4	23621 [†]	146.0 [†]
	40(OOD)	30 [†]	3389.2	5.9	9.3	406.6	37985	149.3

Bold: Best performance within each roll [†] Worst performance within each roll OOD: Given out-of-distribution initial states

baseline optimization, as evidenced by their lower convergence costs. Single-stage optimization outperforms baseline but remains inferior to two-stage approaches, confirming that the relaxation strategy further enhances local optimum extraction by allowing more effective exploration in the first stage followed by refined convergence in the second stage.

Table II presents comprehensive quantitative analysis across performance metrics, trajectory quality, and computational efficiency. Our Two-Point Guidance + Relaxation achieves **100%** success rates across all scenarios, significantly outperforming alternatives, particularly in the challenging 55° case where baseline success rate drops to zero. This demonstrates the critical importance of IL guidance for steep surface manipulation. Two-Point Guidance without relaxation ranks second, while Four-Point and Six-Point Guidance exhibit reduced success rates due to over-constraint in the first optimization stage, where two points already provide sufficient information for trajectory guidance, and additional points excessively constrain the trajectory during the first stage, thereby compromising the second stage's search capability.

Regarding generalization performance, while Learned Points Guidance + Relaxation achieves comparable success rates in standard scenarios, it significantly underperforms in OOD conditions. This performance gap stems from polynomial fitting's superior ability to integrate multi-point information, producing more stable results compared to discrete point prediction, which suffers from data sparsity as identical trajectories generate numerous possible point combinations.

Additionally, Learned Points Guidance + Relaxation exhibits significantly longer convergence times and more iteration times compared to our proposed methods, indicating that the learned points deviate substantially from the desired optimal trajectory, further highlighting the deficiencies of this approach.

Computational analysis reveals that increasing guide points extends convergence time compared to baseline performance, as guide points direct optimization toward potentially distant desired directions, which increases the iteration requirements. This indicates that setting more guide points leads to increased solution time and complexity for the optimization problem.

Interestingly, Two-Point Guidance + Relaxation occasionally produces higher final costs than the no-relaxation variant, reflecting optimization randomness where relaxed first-stage criteria may lead to different local optima. However, cost differences remain minimal, which is still acceptable. In terms of trajectory quality, we observe that IL successfully learns high-quality trajectory tendencies from demonstration data. Furthermore, increasing the number of guide points yields lower final jerk values, indicating that more guide points drive the optimization closer to these learned high-quality trajectories.

These findings collectively demonstrate that our IL-guided optimization framework successfully enhances performance, with our polynomial approach exhibiting superior stability and generalization compared to learned point methods. Moreover, selecting merely two points can provide sufficient trajectory

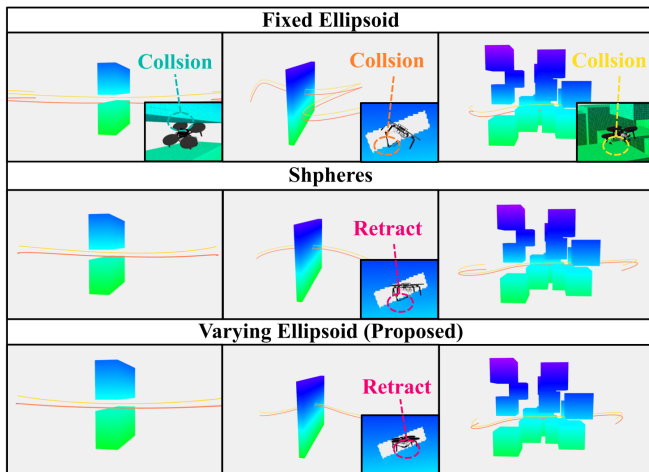


Fig. 11. Collision avoidance performance comparison across three benchmark scenarios: narrow gate (left), tilted hole (middle), and random cubes (right). The fixed ellipsoid method (top row) results in collisions due to rigid volume representation. The spheres method (middle row) achieves collision-free navigation. Our varying ellipsoid method (bottom row) demonstrates superior performance through adaptive manipulator retraction.

tendency information while maintaining computational efficiency.

C. Benchmark

To validate the superiority of our varying ellipsoid method for collision volume approximation, we designed three obstacle-rich scenarios and evaluated the aerial manipulator's collision avoidance performance through each environment.

The first scenario, termed **"Narrow Gate"**, consists of two rectangular obstacles positioned to create a narrow passage between them, as illustrated in the first column of Figure 11. We systematically varied the gap width from 0.60m to 0.25m in 0.05m decrements, resulting in eight experimental trials. Throughout these experiments, the aerial manipulator was manually configured to maintain an extended state at both the initial and final positions, with the end-effector position set to $\mathcal{D}_{p_{e,z}} = -0.2\text{m}$ to ensure consistent testing conditions. The second scenario, **"Tilted Hole,"** features a wall with an inclined hole, as shown in the second column of Figure 11. We examined three different inclination angles: 20° , 40° , and 60° . For each angle, the manipulator's initial end-effector positions were set to $\mathcal{D}_{p_{e,z}} = -0.07\text{m}$, -0.13m , and -0.2m respectively, yielding a total of nine experimental configurations. The third scenario, **"Random Cubes"**, involves randomly distributing 12 cubic obstacles within a three-dimensional workspace. Ten different random configurations were generated to provide diverse navigation challenges for the aerial manipulator. For all experiments, a trial was considered successful if the aerial manipulator reached the target position without any collision occurrence during the entire trajectory.

We compared our approach against two established collision volume approximation methods: Fast-Racing [39] and REMANI [47]. The Fast-Racing method employs a single fixed ellipsoid to approximate the entire aerial manipulator system. To ensure collision safety, this fixed ellipsoid must be

sized to accommodate the manipulator in its fully extended configuration. The REMANI method utilizes multiple spheres to represent the aerial manipulator's geometry: four spheres with 0.08m radius for the quadrotor body, one sphere with 0.025m radius for the end-effector, and fifteen spheres with 0.025m radius distributed along the upper and lower arm segments. All experimental comparisons maintained identical constraint weights and hyperparameters across the 27 total trials.

The results presented in Figure 11 demonstrate clear performance distinctions between methods. The fixed ellipsoid approach failed to navigate collision-free through gaps smaller than or equal to 0.45m, as the rigid geometric representation prevented adaptive arm retraction during narrow passage traversal. Similarly, this method failed in the tilted hole scenario when the initial end-effector positions were set to -0.13m and -0.2m . The random cubes scenario also resulted in collisions when narrow passages were encountered.

As summarized in Table III, the fixed ellipsoid method achieved only 37.5% and 33.3% success rates in the first two scenarios, respectively, while maintaining 70% success in the random cubes scenario due to the relatively sparse obstacle distribution.

In contrast, both the sphere-based method and our proposed varying ellipsoid approach achieved 100% success rates in the narrow gate and tilted hole scenarios. However, their collision avoidance strategies differed significantly, as observable in the bottom-right panel of the second column in Figure 11. The sphere-based method tends to induce lateral (xy -axis) manipulator movements, while our approach primarily achieves collision avoidance through strategic end-effector retraction along the z -axis.

For the random cubes scenario, our proposed method achieved an 80% success rate compared to 70% for the sphere-based approach. The failures in both methods were primarily attributed to path planning limitations that forced the aerial manipulator into excessively constrained regions where sufficient SFC volume could not be established. Additionally, the sphere-based method's failures resulted from the cumulative high costs generated by multiple spheres in proximity to obstacles, preventing the optimizer from adequately balancing other essential constraints and leading to localized collisions.

The computational efficiency comparison in Table III reveals that our method's average iteration time closely matches that of the fixed ellipsoid approach while significantly outperforming the sphere-based method. This efficiency stems from our approach's $O(n)$ computational complexity, as only a single ellipsoid-to-polyhedron cost evaluation is required per iteration, compared to the $O(mn)$ complexity of the sphere-based method that necessitates individual cost calculations for each sphere.

In summary, our varying ellipsoid fitting method successfully combines the computational efficiency of ellipsoidal representations with comprehensive whole-body collision avoidance capabilities for aerial manipulators.

TABLE III
PERFORMANCE COMPARISON OF OBSTACLE AVOIDANCE METHODS

Method	Narrow Gate	Tilted Hole	Random Cubes	Avg. Iteration Time
Fast-Racing [39]	37.5%	33.3%	70.0%	33.85 ms
REMANI [47]	100.0%	100.0%	70.0%	123.73 ms
Proposed	100.0%	100.0%	80.0%	45.85 ms

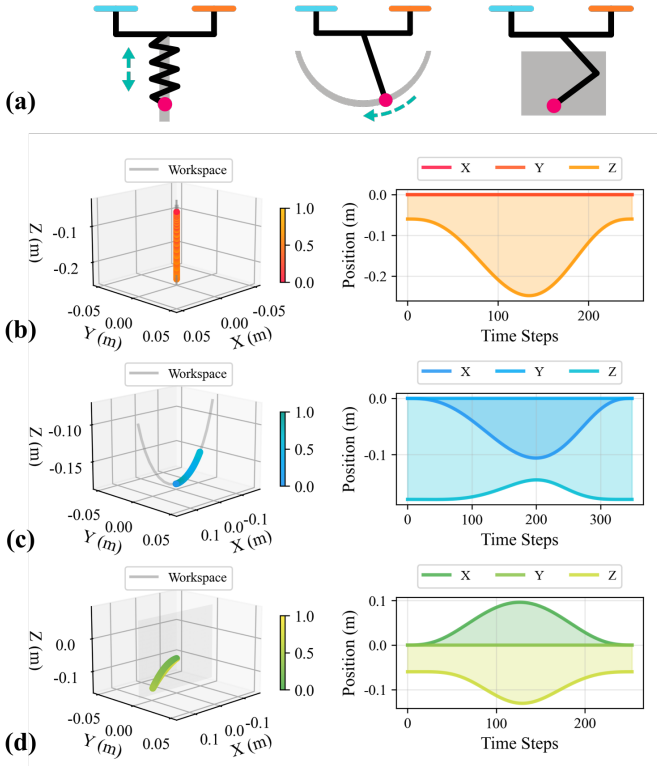


Fig. 12. Validation of the planning framework across different manipulator configurations. (a) Three manipulator types: vertical motion (top), 1-DOF circular arc motion (middle), and 2-DOF planar motion (bottom). Left column shows 3D end-effector trajectories within their respective workspaces (gray regions). Right column displays the corresponding position profiles for each axis over time. All trajectories remain strictly within workspace boundaries, demonstrating the framework's adaptability to various kinematic constraints.

D. Simulation

We formulate the end-effector planning in Cartesian space specifically to accommodate different aerial manipulator configurations. This section validates that our approach can work with various manipulator types through comprehensive simulation studies.

Due to hardware limitations, we evaluate the feasibility of our approach through simulation analysis. As illustrated in Figure 12 (a), we consider three distinct manipulator configurations with different kinematic constraints. The first type is a telescopic manipulator that permits only vertical (z -axis) motion of the end-effector. The second type is a 1-DOF manipulator that allows the end-effector to move along a circular arc in the xz -plane. The third type is a 2-DOF manipulator that enables planar motion within the entire xz -plane. The respective workspaces for each configuration are visualized as gray regions in the left column of Figure 12.

We designed a planar grasping scenario with a waypoint

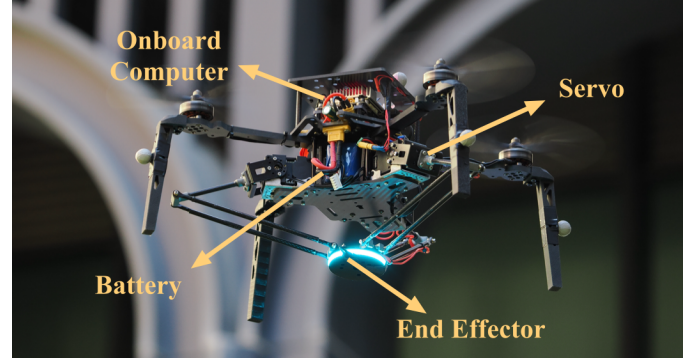


Fig. 13. Real-world aerial manipulator platform used for validation.

constraint for the end-effector in the world frame $\mathcal{F}_W$, setting the terminal velocity to zero at the target position. For the telescopic manipulator, the workspace bounds were set to $\mathcal{D}_{p_{e,z}} \in [-0.25, -0.06]\text{m}$. For the 1-DOF manipulator, the workspace is constrained to a circular arc with radius 0.18m in the xz -plane. For the 2-DOF planar manipulator, the workspace was defined as $\mathcal{D}_{p_{e,x}} \in [-0.2, 0.2]\text{m}$ and $\mathcal{D}_{p_{e,z}} \in [-0.25, -0.06]\text{m}$.

The trajectory planning results demonstrate high accuracy, with minimum distances to the target point of $6.851 \times 10^{-3}\text{m}$, $4.857 \times 10^{-3}\text{m}$, and $2.055 \times 10^{-3}\text{m}$ for the three manipulator types, respectively. These errors are sufficiently small to confirm successful target reaching. The 3D end-effector trajectories in the frame $\mathcal{F}_D$, shown in the left column of Figure 12, strictly adhere to their respective workspace constraints. The right column presents the individual axis variations over time, further confirming that all planned trajectories remain within the prescribed workspace boundaries.

These results demonstrate that our whole-body planning framework successfully adapts to different manipulator types while maintaining workspace compliance and trajectory accuracy.

E. Real-world Implementation

Our system is implemented in C++11 on Ubuntu 20.04 with ROS Noetic. As illustrated in Figure 13, the experimental platform integrates an NVIDIA Jetson Orin NX 16GB as the onboard computer and an NxtPX4v2 flight controller on a quadrotor equipped with a delta arm. The delta arm utilizes three DYNAMIXEL XL430-W250-T servo motors and features a needle end-effector for precise manipulation tasks. For state estimation, the system employs an Extended Kalman Filter (EKF) that fuses data from the NOKOV Motion Capture System and an onboard IMU. We implement a separated control framework inspired by the composite control scheme in [35], which employs an NDOB to compensate for internal coupling and external disturbances, while using a high-pass filter to enhance manipulator agility. Throughout the experimental validation, we assume that interaction forces including gravity and friction are constant and known during the planning phase.

To validate our approach in real-world scenarios, we focus on a fundamental question: whether our flexible waypoint constraints can effectively operate in practical environments

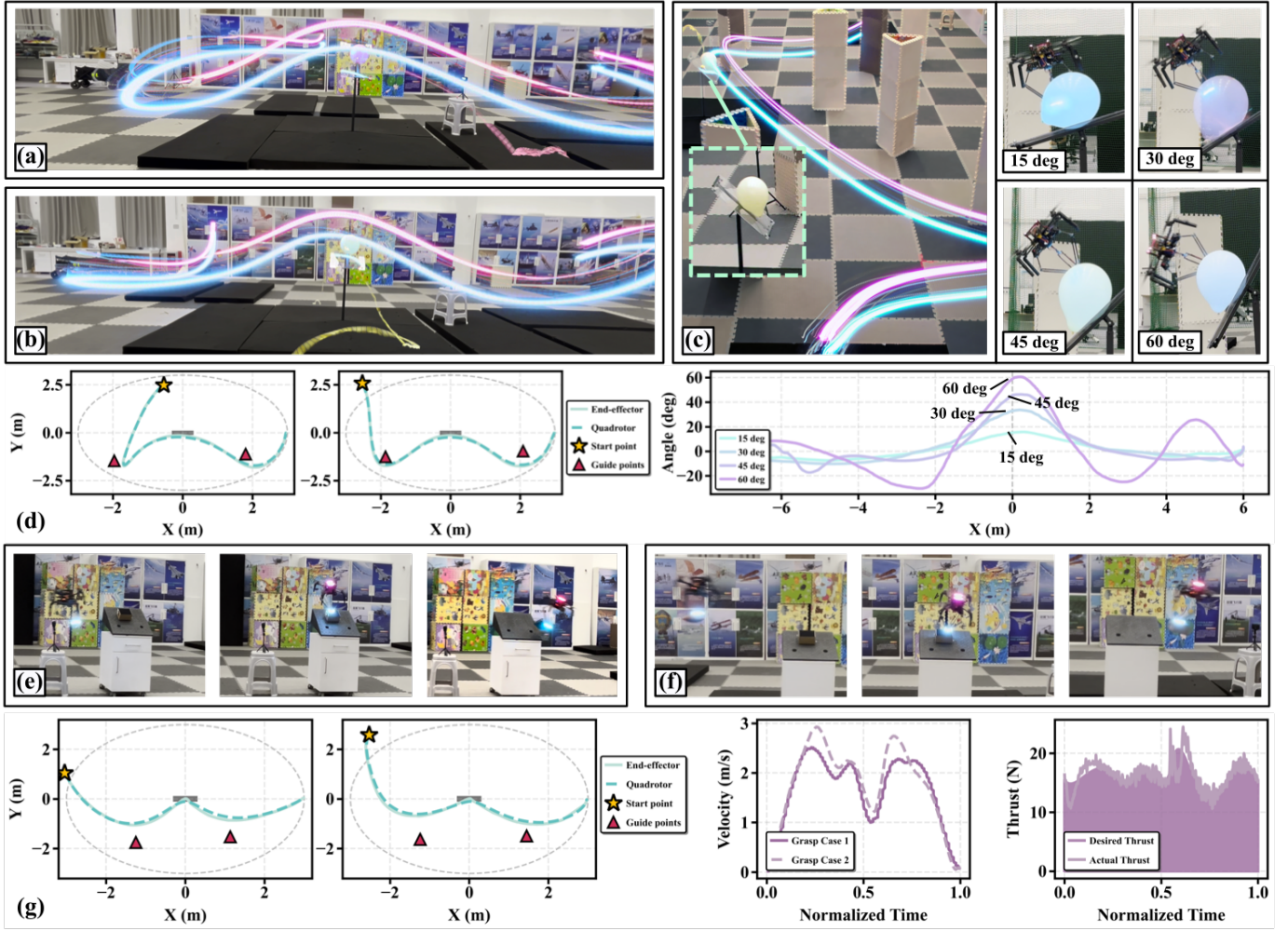


Fig. 14. Experimental results for strike and grasp tasks. (a)-(c) show aerial striking experiments while (e) and (f) present aerial grasping results. (d) The first two plots demonstrate IL-generated guide points and resulting quadrotor and end-effector motion in the xy-plane. The third plot shows roll angle variations across four strike maneuvers. (g) The first two plots show IL guide point generation for grasp skills. The third and fourth plots present quadrotor velocity and comparison between planned and actual thrust curves for the aerial manipulator.

and successfully execute diverse manipulation skills. To address this question systematically, we design 9 fundamental manipulation skills: striking, grasping, lifting, pressing, winding, pulling, pushing, crossing, and writing. We conduct a total of 17 experiments across these skills and analyze their performance comprehensively.

1) Strike: We design a striking scenario with an inclined surface positioned at approximately 37.6 degrees at the center of the test area. To thoroughly validate our IL-guided optimization framework, we position the aerial manipulator behind the inclined surface and conduct experiments from two distinct initial positions: one within the training data distribution (1.5-3m radius) and another outside this distribution (beyond 3m radius). The strike skill requires setting a single end-effector waypoint constraint in $\mathcal{F}_W$ along with its corresponding orientation.

As shown in Figure 14 (a) and (b), given the initial relative position and the surface inclination angle, our IL-guided framework generates guide points as demonstrated in Figure 14 (d) (first two plots). To further validate the

orientation capabilities of our waypoint constraints, we test the aerial manipulator's striking performance on four different inclined surfaces with angles of 15°, 30°, 45°, and 60°. To eliminate the influence of other constraints and focus solely on orientation variations, we specify that the delta arm should extend to ${}^D\mathbf{p}_e = [0.0, 0.0, -0.2]^\top$ m, and subsequently calculate the corresponding quadrotor position to set as the waypoint constraint. This approach enables isolated observation of orientation changes. As illustrated in Figure 14 (c) and the third plot in Figure 14 (d), the aerial manipulator successfully achieves the desired corresponding attitude at the target position.

2) Grasp: We design complementary grasping scenarios with two experimental configurations. In each test, an object of known mass is placed on an inclined surface, with its position tracked using the motion capture system. The aerial manipulator is tasked with planning and executing a grasping maneuver. This grasp skill requires setting an end-effector waypoint constraint, and to improve success rates, we additionally impose a velocity constraint of 0.2 m/s downward

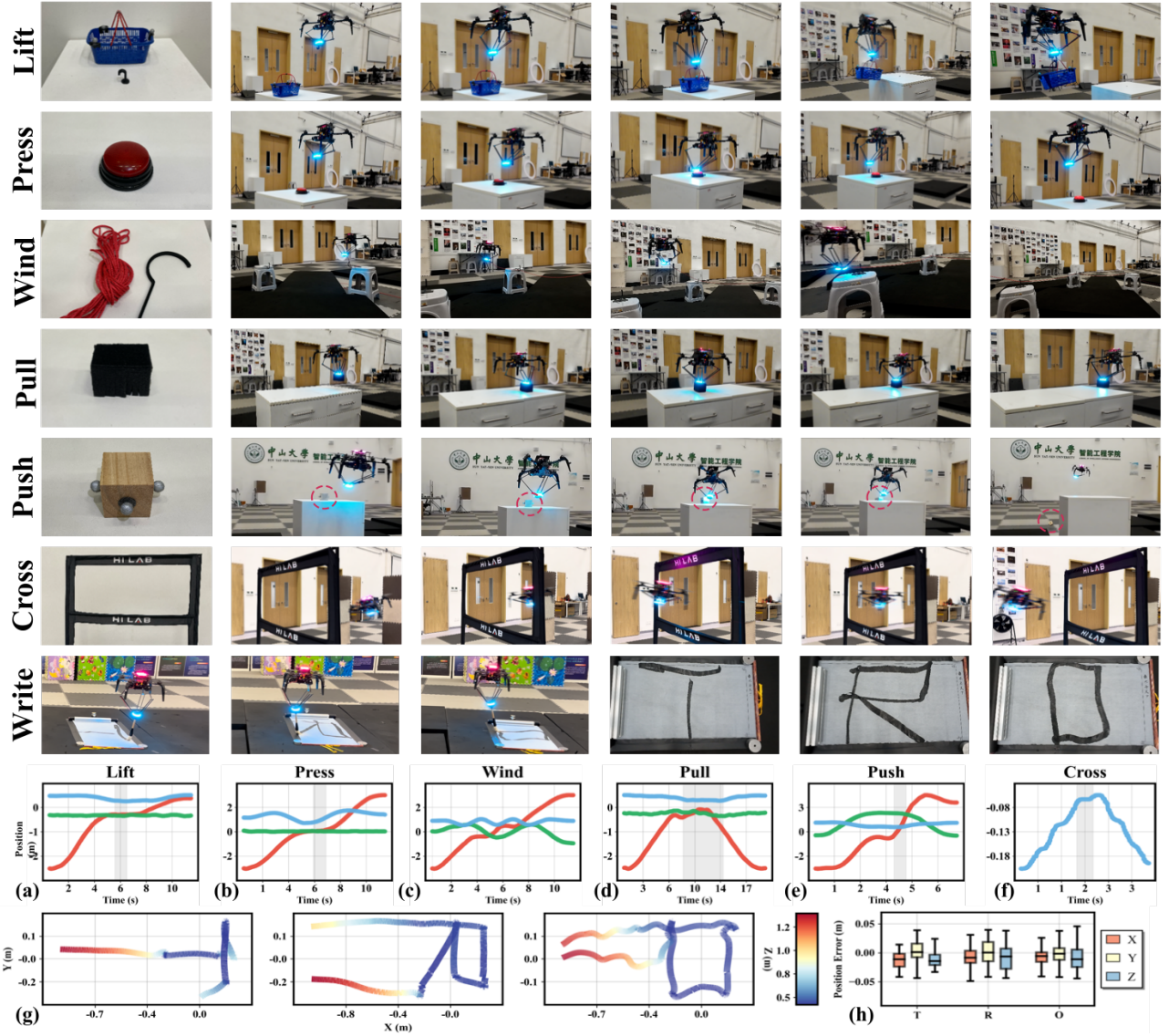


Fig. 15. Real flight results for seven tasks: lift, press, wind, pull, push, cross, and write. (a)-(e) show xyz trajectory variations in the world frame for the first five tasks. (f) presents end-effector z-axis motion trajectory in $\mathcal{F}_D$ during the cross task. (g) and (h) show actual end-effector movement during the write task and corresponding error distributions, respectively.

for end-effector along z_B at the waypoint, while constraining velocities in x_B and y_B directions to zero.

To validate our framework's advantages, we again position the aerial manipulator behind the inclined surface as the initial position. The first experimental group has an initial position at (-3, 1) m with a surface inclination of approximately 42.3° , while the second group starts at (-2.5, 2.5) m with an inclination of 26.7° . The actual flight performance is shown in Figure 14 (e) and (f), demonstrating successful object grasping in both cases. Figure 14 (g) (first two plots) illustrates the corresponding IL-generated guide points and the resulting quadrotor and end-effector trajectories in the xy -plane. The third plot in Figure 14 (g) shows the quadrotor's grasping velocity, demonstrating rapid grasping speeds of nearly 3 m/s while confirming that the quadrotor reduces speed appropriately to satisfy the constraints. The fourth plot compares the planned versus actual thrust forces,

showing that despite using a simplified model that considers only quasi-static forces and assumes fixed external forces, these assumptions remain reasonable for simple manipulation tasks. However, the observed discrepancies during contact phases indicate that more sophisticated contact-rich planning methods would be necessary for more complex or dynamic manipulation scenarios.

3) Lift, Press, Wind: These three tasks can be completed by applying one or several waypoint constraints. The lift task requires the aerial manipulator to use a hook to lift a basket, which is considered successful when the basket is lifted. We apply two waypoint constraints: one below the basket handle and another directly above it. The press task requires the end-effector to press a button, with success indicated by the button emitting red light, achieved by setting a single waypoint constraint at the button's position. The wind task involves threading a line through three hooks at known positions,

with success defined as the line successfully passing through all hooks. This requires setting three end-effector waypoint constraints. The experimental results shown in Figure 15 (**Lift, Press, Wind**) demonstrate successful completion of all three tasks. The corresponding motion trajectory curves are presented in Figure 15 (a)-(c), where gray areas indicate the task execution regions.

4) Pull, Push: We design specific scenarios to validate the two skills independently. For the pull task, we attach a sponge to the end-effector and program the aerial manipulator to perform a table-wiping task. We set three coplanar waypoint constraints beyond the table surface and constrain the z -component variation of the end-effector position to zero within these points, enabling movement within a single plane. For the push task, we place a wooden block on a table surface and program the aerial manipulator to push it off the table while restricting end-effector movement to the x -axis direction only, with y and z -axis position components constrained. The results shown in Figure 15 (**Pull, Push**) demonstrate successful task execution. Figure 15 (d) and (e) show that the gray constraint regions effectively limit motion: the pull task maintains minimal z -axis variation, while the push task shows consistent y and z -axis values within the gray constraint region.

5) Cross: We design a narrow racing gate that requires the aerial manipulator to retract its robotic arm to successfully pass through. To increase experimental complexity and comprehensively validate our framework's capability, we extend the end-effector to the -0.2m position before and after crossing, thereby demonstrating both obstacle avoidance and manipulation capabilities. Figure 15 (**Cross**) shows the aerial manipulator's state during gate traversal, while Figure 15 (f) presents the end-effector's z -axis trajectory in $\mathcal{F}_D$ during flight, clearly demonstrating the robotic arm's retraction-extension sequence from initial extension through retraction for gate passage to re-extension afterward.

6) Write: The final task we design is writing, which is comparatively complex, requiring numerous waypoint constraints while ensuring end-effector movement within the same plane. The positional constraints vary dynamically; for example, writing "T" requires initial constraint to x -axis movement, lifting the brush pen, then constraining to y -axis movement. Writing "R" requires multiple constrained movement transitions. The experimental results shown in Figure 15 (**Write**) demonstrate successful completion of writing the letters "TRO" using a brush pen. The actual end-effector movement can be observed in Figure 15 (g). We note slight oscillations at corners when writing "O," attributed to unstable connections between the end-effector and brush, as well as uneven paper surfaces. The position errors for these three experiments, shown in Figure 15 (h), maintain median end-effector errors within 3cm, which falls within acceptable ranges. This further validates that while our planning employs simplified models, most disturbances and unconsidered dynamic coupling effects can be resolved at the controller level, with our planner providing highly effective reference trajectories.

VII. CONCLUSION

This paper introduces a novel whole-body integrated motion planning framework for aerial manipulators that jointly optimizes the motions of the quadrotor and the manipulator. We propose an efficient varying ellipsoid collision avoidance method that dynamically adapts to the aerial manipulator's configuration changes, enabling safe navigation in complex environments. The framework introduces flexible partial waypoint constraints for precise task specification and an IL-guided optimization approach to address the limitations of standard optimization methods in handling large-attitude manipulation tasks. This IL-guided method leverages imitation learning to provide appropriate guide points, significantly improving planning success rates for challenging manipulation scenarios. Our comprehensive experimental validation through 17 real-world experiments spanning 9 fundamental manipulation skills demonstrates the framework's excellent scalability and versatility across diverse manipulation tasks.

While our algorithm demonstrates outstanding performance across various manipulation skills, the current computational requirements prevent real-time operation. This limitation restricts the framework's ability to handle highly dynamic and complex environments where real-time replanning and adaptive adjustments are essential for safe operation. Additionally, as evidenced by the discrepancies observed in thrust force comparisons during grasp experiments, the simplified quasi-static force models limit our ability to consider tasks involving strong interaction forces during the planning stage. Therefore, future work will focus on developing advanced dynamic models and improving computational efficiency, with the goal of enabling real-time performance in complex, contact-rich scenarios.

REFERENCES

- [1] 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, 2022.
- [2] K. Zhang, P. Chermprayong, F. Xiao *et al.*, "Aerial additive manufacturing with multiple autonomous robots," *Nature*, vol. 609, no. 7928, pp. 709–717, 2022. [Online]. Available: <https://doi.org/10.1038/s41586-022-04988-4>
- [3] M. . Trujillo, J. R. Martinez-de Dios, C. Martn, A. Viguria, and A. Ollero, "Novel aerial manipulator for accurate and robust industrial ndt contact inspection: A new tool for the oil and gas inspection industry," *Sensors*, vol. 19, no. 6, 2019. [Online]. Available: <https://www.mdpi.com/1424-8220/19/6/1305>
- [4] P. Foehn, A. Romero, and D. Scaramuzza, "Time-optimal planning for quadrotor waypoint flight," *Science Robotics*, vol. 6, no. 56, p. eabh1221, 2021. [Online]. Available: <https://www.science.org/doi/abs/10.1126/scirobotics.abh1221>
- [5] P. Chermprayong, K. Zhang, F. Xiao, and M. Kovac, "An integrated delta manipulator for aerial repair: A new aerial robotic system," *IEEE Robotics & Automation Magazine*, vol. 26, no. 1, pp. 54–66, 2019.
- [6] K. Hang, X. Lyu, H. Song, J. A. Stork, A. M. Dollar, D. Kragic, and F. Zhang, "Perching and resting paradigm for uav maneuvering with modularized landing gears," *Science Robotics*, vol. 4, no. 28, p. eaau6637, 2019. [Online]. Available: <https://www.science.org/doi/abs/10.1126/scirobotics.aau6637>
- [7] H.-N. Nguyen, S. Park, J. Park, and D. Lee, "A novel robotic platform for aerial manipulation using quadrotors as rotating thrust generators," *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 353–369, 2018.
- [8] M. Xu, Q. De, D. Yu, A. Hu, Z. Liu, and H. Wang, "Biomimetic morphing quadrotor inspired by eagle claw for dynamic grasping," *IEEE Transactions on Robotics*, vol. 40, pp. 2513–2528, 2024.

- [9] J. Thomas, J. Polin, K. Sreenath, and V. Kumar, "Avian-inspired grasping for quadrotor micro UAVs," in *Volume 6A: 37th Mechanisms and Robotics Conference*. American Society of Mechanical Engineers, p. V06AT07A014.
- [10] M. Ryll, G. Muscio, F. Pierri, E. Cataldi, G. Antonelli, F. Caccavale, D. Bicego, and A. Franchi, "6d interaction control with aerial robots: The flying end-effector paradigm," vol. 38, no. 9, pp. 1045–1062. [Online]. Available: <http://journals.sagepub.com/doi/10.1177/0278364919856694>
- [11] 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.
- [12] 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*, vol. 22, pp. 2577–2594, 2025.
- [13] H. Lee, H. Kim, and H. J. Kim, "Planning and control for collision-free cooperative aerial transportation," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 1, pp. 189–201, 2018.
- [14] 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.
- [15] Y. Chen, J. Liang, Y. Wu, Z. Miao, H. Zhang, and Y. Wang, "Adaptive sliding-mode disturbance observer-based finite-time control for unmanned aerial manipulator with prescribed performance," *IEEE Transactions on Cybernetics*, vol. 53, no. 5, pp. 3263–3276, 2023.
- [16] H. Cao, Y. Li, C. Liu, and S. Zhao, "Eso-based robust and high-precision tracking control for aerial manipulation," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 2, pp. 2139–2155, 2024.
- [17] G. He, X. Guo, L. Tang, Y. Zhang, M. Mousaei, J. Xu, J. Geng, S. Scherer, and G. Shi, "Flying hand: End-effector-centric framework for versatile aerial manipulation teleoperation and policy learning." [Online]. Available: <http://arxiv.org/abs/2504.10334>
- [18] M. Spahn, B. Brito, and J. Alonso-Mora, "Coupled mobile manipulation via trajectory optimization with free space decomposition," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 12 759–12 765. [Online]. Available: <https://ieeexplore.ieee.org/document/9561821/>
- [19] R. Spica, A. Franchi, G. Oriolo, H. H. Bulthoff, and P. R. Giordano, "Aerial grasping of a moving target with a quadrotor UAV," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 4985–4992. [Online]. Available: <http://ieeexplore.ieee.org/document/6385771/>
- [20] M. Tognon, E. Cataldi, H. A. T. Chavez, G. Antonelli, J. Corti, 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.
- [21] Suseong Kim, Seungwon Choi, and H. J. Kim, "Aerial manipulation using a quadrotor with a two DOF robotic arm," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 4990–4995. [Online]. Available: <http://ieeexplore.ieee.org/document/6697077/>
- [22] M. Brunner, G. Rizzi, M. Studiger, R. Siegwart, and M. Tognon, "A planning-and-control framework for aerial manipulation of articulated objects," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 689–10 696, 2022.
- [23] X. Guo, G. He, J. Xu, M. Mousaei, J. Geng, S. Scherer, and G. Shi, "Flying calligrapher: Contact-aware motion and force planning and control for aerial manipulation." [Online]. Available: <http://arxiv.org/abs/2407.05587>
- [24] P. Foehn, A. Romero, and D. Scaramuzza, "Time-optimal planning for quadrotor waypoint flight," vol. 6, no. 56, p. eabh1221. [Online]. Available: <https://www.science.org/doi/10.1126/scirobotics.abh1221>
- [25] H. Kim, H. Seo, J. Kim, and H. J. Kim, "Sampling-based motion planning for aerial pick-and-place," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 7402–7408.
- [26] H. Seo, S. Kim, and H. J. Kim, "Locally optimal trajectory planning for aerial manipulation in constrained environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1719–1724. [Online]. Available: <http://ieeexplore.ieee.org/document/8205984/>
- [27] M. Neunert, C. de Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart, and J. Buchli, "Fast nonlinear model predictive control for unified trajectory optimization and tracking," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1398–1404.
- [28] 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.
- [29] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion." [Online]. Available: <http://arxiv.org/abs/2303.04137>
- [30] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn, "OpenVLA: An open-source vision-language-action model." [Online]. Available: <http://arxiv.org/abs/2406.09246>
- [31] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky, "\$_0\$: A vision-language-action flow model for general robot control." [Online]. Available: <http://arxiv.org/abs/2410.24164>
- [32] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware." [Online]. Available: <http://arxiv.org/abs/2304.13705>
- [33] M. Ghazi, Q. Nazir, S. Butt, and A. Baqai, "Accuracy analysis of 3-rss delta parallel manipulator," *Procedia Manufacturing*, vol. 17, pp. 174–182, 01 2018.
- [34] D. Tzoumanikas, F. Graule, Q. Yan, D. Shah, M. Popovic, and S. Leutenegger, "Aerial manipulation using hybrid force and position NMPC applied to aerial writing." [Online]. Available: <http://arxiv.org/abs/2006.02116>
- [35] H. Chen, B. Ye, X. Liang, W. Deng, and X. Lyu, "Ndob-based control of a uav with delta-arm considering manipulator dynamics," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 7505–7511.
- [36] A. Codourey, "Dynamic modelling and mass matrix evaluation of the delta parallel robot for axes decoupling control," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '96)*, vol. 3, 1996, pp. 1211–1218.
- [37] 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*, vol. 2, no. 3, pp. 1688–1695, 2017.
- [38] 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.
- [39] 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.
- [40] Y. Ren, S. Liang, F. Zhu, G. Lu, and F. Zhang, "Online whole-body motion planning for quadrotor using multi-resolution search," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1594–1600.
- [41] S. Yang, B. He, Z. Wang, C. Xu, and F. Gao, "Whole-body real-time motion planning for multicopters," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 9197–9203. [Online]. Available: <https://ieeexplore.ieee.org/document/9561526/>
- [42] Y. Gao, J. Ji, Q. Wang, R. Jin, Y. Lin, Z. Shang, Y. Cao, S. Shen, C. Xu, and F. Gao, "Adaptive tracking and perching for quadrotor in dynamic scenarios," *IEEE Transactions on Robotics*, vol. 40, pp. 499–519, 2024.
- [43] C. A. Dimmig and M. Kobilarov, "Non-prehensile aerial manipulation using model-based deep reinforcement learning," in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, 2024, pp. 2194–2200.
- [44] R. Ni, T. Schneider, D. Panozzo, Z. Pan, and X. Gao, "Robust & asymptotically locally optimal UAV-trajectory generation based on spline subdivision," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 7715–7721. [Online]. Available: <https://ieeexplore.ieee.org/document/9561272/>
- [45] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu, "RDT-1b: a diffusion foundation model for bimanual manipulation." [Online]. Available: <http://arxiv.org/abs/2410.07864>
- [46] W. Peebles and S. Xie, "Scalable diffusion models with transformers." [Online]. Available: <http://arxiv.org/abs/2212.09748>
- [47] C. Wu, R. Wang, M. Song, F. Gao, J. Mei, and B. Zhou, "Real-time whole-body motion planning for mobile manipulators using environment-adaptive search and spatial-temporal optimization," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1369–1375. [Online]. Available: <https://ieeexplore.ieee.org/document/10610192/>



Weiliang Deng received his B.Eng. degree in Intelligence Science and Technology from Sun Yat-sen University, Shenzhen, China, in 2025. He is currently working toward the M.Phil. degree with the Department of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen, China.

His research interests include motion planning, aerial robots and robot manipulation.



Haoran Chen received his B.Eng. degree in Intelligence Science and Technology from Sun Yat-sen University, Shenzhen, China, in 2025. He is currently working toward the M.S. degree in the department of mechanical engineering at the National University of Singapore, Singapore.

His research interests include aerial manipulation, and hardware design, locomotion control, and planning of legged robots.



Hongming Chen received the B.Eng. degree in Software Engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2024. He is currently working toward the M.Phil. degree with the Department of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen, China.

His research interests include unmanned aerial vehicles, semantic navigation, and aerial manipulation.



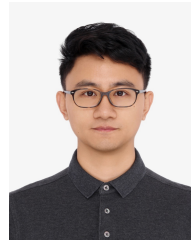
Ziliang Li received his B.Eng. degree in Intelligence Science and Technology from Sun Yat-sen University, Shenzhen, China, in 2024. He is currently working toward the M.Phil. degree with the Department of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen, China.

His research interests include optimal control, motion planning, and aerial manipulation.



Biyu Ye received the B.Eng. degree in Traffic Engineering from Sun Yat-sen University, Guangzhou, China, in 2024. He is currently working toward the M.S. degree in aerial vehicle design with The School of Intelligent Systems Engineering, Sun Yat-sen University.

His research interests include the perception and control of aerial robots.



Ximin Lyu received his B.Eng. and M.Phil. degrees in Aircraft Manufacturing from Harbin Institute of Technology, Harbin, China, in 2012 and 2014, respectively. He received his Ph.D. in Electronic and Computer Engineering from the Hong Kong University of Science and Technology, Hong Kong, China, in 2019.

In 2018, he served as a Senior Flight Control Researcher at Da Jiang Innovation (DJI) Technology Company in China. Since 2021, he has been an Associate Professor with the Department of Intelligent Systems Engineering, Sun Yat-sen University, Shenzhen, China. His research interests encompass robotics and controls, with a specific focus on UAV/UGV design, control, and planning.