This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A Collision-Aware Motion Planning Framework With Goal State Generation for Aerial Manipulation in Constrained Environments

Zhaopeng Zhang, *Graduate Student Member, IEEE*, Hai Yu, *Member, IEEE*, Wei He, Yongchun Fang, *Senior Member, IEEE*, Jianda Han, *Member, IEEE*, and Xiao Liang, *Senior Member, IEEE*

***Abstract*—In this article, we propose a collision-aware motion planning framework with goal state generation (GSG) for aerial manipulation in constrained environments. The framework generates nominal trajectories that satisfy the collision checks and are dynamically consistent. Owing to the kinematic redundancy of aerial manipulators, a given end-effector (EE) pose can correspond to multiple feasible system states. We introduce a *collision-avoidance margin* concept that quantifies the collision risk for aerial manipulators, and generate an optimal goal state by minimizing it while satisfying the target EE pose and collision-avoidance constraints. Then, a goal-state-biased path searching algorithm is developed in configuration space, where the generated optimal goal state guides the search to efficiently obtain a whole-body path that passes the geometric collision checks, and a weighted-sum cost is adopted to properly measure the distance between states. Finally, the generated path is converted into a dynamically consistent trajectory satisfying feasibility constraints for the flat outputs of the aerial manipulator by solving a gradient-based optimization problem that incorporates both collision-avoidance constraints and feasibility constraints, including velocity limits, acceleration limits, joint-angle limits, and maximum tilt-angle limit. The proposed method incorporates a geometrically accurate whole-body collision model for aerial manipulation, which explicitly accounts for the full geometry of both the multirotor and the robotic arm through multiple sets of spheres, supporting whole-body collision avoidance in constrained environments. The proposed motion planning framework is validated and shown to be effective in both simulation and real-world experiments.**

***Index Terms*—Aerial manipulation, collision-aware, constrained environments, motion planning.**

I. Introduction

MULTIROTOR aerial vehicles have been widely studied for diverse applications, such as coverage missions [1] or payload transportation [2]. An aerial manipulator, integrating a multirotor with a robotic arm, extends from mobility to manipulation. The robotic arm provides manipulation capability, while the multirotor eliminates the fixed workspace constraint of conventional manipulators, thereby significantly enhancing operational flexibility. Robust control methods have been developed to ensure stable and reliable operation of aerial manipulators under uncertainties and disturbances [3], [4], [5]. It has been widely applied in various fields, such as grasping an object [6], controlled shaking [7], contact-based inspection [8] or tool docking [9]. In constrained environments, motion planning is crucial for safe and efficient collision avoidance.

Mobile manipulation tasks can be performed on the ground or in the air. A robotic arm on a wheeled platform forms a wheeled mobile manipulator. Pilania et al. [10] propose a decoupled framework, planning the mobile base and manipulator in separate phases. Some methods consider all degrees of freedom (DoFs) simultaneously [11]. Tang et al. [12] establish differential flatness for wheeled mobile manipulators and exploit it for motion planning. Wu et al. [13] propose an environment-adaptive search framework with spatial-temporal optimization for collision-free, dynamically feasible trajectories. Beyond wheeled platforms, legged manipulators have also been studied for loco-manipulation. Bellicoso et al. [14] integrate motion planning and whole-body control for quadrupedal manipulators, while Sleiman et al. [15] unify locomotion and manipulation within a multicontact MPC framework.

Different from ground-based mobile manipulators that can translate while maintaining a horizontal orientation, aerial manipulators achieve horizontal motion by tilting the platform, making the motion planning problem more challenging. Motion

Received 30 April 2026; revised 31 July 2026 and 28 September 2026; accepted 15 October 2026. This work was supported in part by the National Key Research and Development Program of China under Grant 2024YFB4710400, in part by the National Natural Science Foundation of China under Grant 62673279 and Grant U25A20473, and in part by the Postdoctoral Fellowship Program of CPSF under Grant GZB20260452. *(Corresponding authors: Jianda Han; Xiao Liang.)*

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

This article has supplementary downloadable material available at https://doi.org/10.1109/TIE.2026.3748210, provided by the authors.

Digital Object Identifier 10.1109/TIE.2026.3748210

1557-9948 © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

planning methods for aerial manipulators can be categorized into two types [16]. One type is the decoupled method, separating planning into approach and manipulation phases. The other type is the coupled method, or whole-body planning, which simultaneously plans the multirotor and the robotic arm. In [17], the aerial manipulator is driven to a target area and then the robotic arm trajectory is planned to grasp a target object from a moving platform. Wang et al. [18] approach the target before executing the peg-in-hole task using the robotic arm. Cao et al. [19] propose a two-phase planning method for pick-and-place tasks, using a large sphere to enclose the aerial manipulator for obstacle avoidance. Such decoupled methods reduce motion efficiency due to sequential planning and limit task flexibility by treating the multirotor as a floating platform during manipulation.

Unlike decoupled methods, whole-body planning simultaneously considers the motion of the multirotor and the robotic arm. Luo et al. [20] propose a time-optimal handover trajectory planning method based on discrete dynamics. Welde et al. [21] develop coordinate-free dynamics and analyze differential flatness for 6-DoF aerial manipulators, later extending the analysis to any-configuration systems with task-space planning [22]. He et al. [23] employ imitation learning for end-effector (EE) planning and trajectory tracking. However, these methods do not address obstacle avoidance in constrained environments.

For real-world deployments, addressing collision-avoidance problem remains a critical challenge in confined environments. Regarding this problem, optimization-based and safety-theoretic methods have provided foundational insights. Classical trajectory optimization methods, such as CHOMP [24] and TrajOpt [25], combine smooth collision costs with hinge penalties, motivating the collision-avoidance margin proposed in this work. Safety-critical frameworks based on Hamilton-Jacobi reachability [26] or control barrier functions (CBFs) [27] emphasize formal safety guarantees through rigorous mathematical bounds. However, Hamilton-Jacobi reachability often suffers from the curse of dimensionality when applied to high-dimensional systems [28], while the synthesis of valid CBFs for complex underactuated systems in constrained environments remains highly challenging [29].

In contrast to these methods, which prioritize formal verification, motion planning for aerial manipulators specifically focuses on the practical and efficient trajectory generation. Caballero et al. [30], [31] propose a rapidly-exploring random tree star (RRT*)-based motion planning method for an aerial manipulator with two arms for long-reach manipulation and present the experimental validation, while this method does not consider the dynamic model of the aerial manipulator. Seo et al. [32] propose a locally optimal trajectory planning method for aerial manipulation in constrained environments. However, this method has not been validated in extremely confined environments. Kim et al. [33] employ informed-RRT* method for pick-and-place tasks, where the algorithm is used separately for the pick and place trajectory segments, combined with the local planner in [34] to generate collision-free trajectories. These methods [32], [33], [34] parameterize trajectories based on the multirotor position while neglecting the dynamic coupling between the multirotor and the robotic arm, failing to account for dynamic consistency. Besides, only the EE position is considered, without considering the orientation, which is insufficient for precise manipulation tasks. Deng et al. [35] incorporate intermediate waypoints derived from jump point search (JPS) and safe flight corridors into a trajectory optimization framework, while employing dynamic ellipsoidal approximations. Zhang et al. [36] propose a coupled motion planning method for aerial manipulators in constrained environments and use a dynamic convex polyhedron to enclose the aerial manipulator. Zhang et al. [37] adopt a two-sphere model for aerial manipulators to support real-time navigation in unknown environments, but the focus is real-time navigation rather than fine-scale manipulation in highly constrained environments. Although these geometric approximations [35], [36], [37] effectively handle collision-avoidance problems in relatively open environments, the coarse approximations may restrict the efficacy of these methods in highly constrained environments. Lee et al. [38] propose a two-step optimization-based method using multiple ellipsoids for omnidirectional aerial manipulators. However, it is not directly applicable to underactuated platforms, which must tilt to achieve horizontal motion.

Collision-aware goal-state selection has been investigated for fixed and mobile manipulators [39], [40], [41]. However, such goal-state generation has received limited attention in aerial manipulation. Existing methods generally do not explicitly select a goal state according to the surrounding collision information, but instead formulate the planning problem with a prescribed goal state or waypoint constraints [30], [31], [32], [33], [34], [35], [36], [37], [38].

Although many studies have focused on the motion planning problem for aerial manipulation, two issues remain to be addressed. First, some decouple the problem into approach and manipulation phases [17], [18], [19], leading to reduced efficiency and flexibility. Second, the collision-avoidance problem remains insufficiently addressed. Some formulate motion planning solely in unconstrained environments [20], [21], [22]. Others adopt coarse models, such as fixed spheres [19], dynamic ellipsoids [35], convex polyhedra [36] or two-spheres [37]. While they are effective for large-scale navigation tasks, these coarse geometric approximations limit applicability in confined environments, especially for precise manipulation tasks.

Aiming to solve the above-mentioned issues, we propose a complete motion planning framework for aerial manipulation, including goal state generation (GSG), path searching and trajectory optimization stages. In the GSG stage, an optimal goal state is generated by minimizing the collision-avoidance margin. In the path searching stage, a bidirectional RRT* (Bi-RRT*) algorithm generates waypoints connecting the start and goal states. In the trajectory optimization stage, the trajectory is parameterized using the flat outputs, in which the position component is the CoM of the whole system, and is combined with the proposed link-level whole-body collision model and feasibility constraints for collision-aware trajectory refinement. Both simulation and experimental results demonstrate the

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

effectiveness of the proposed framework for aerial manipulation. The main contributions are listed as follows.

1) Several existing motion-planning methods for aerial manipulation are primarily designed for or evaluated in relatively open environments [19], [30], [31], [33], [34], [35], [36], [37]. However, whole-body motion planning for aerial manipulators is particularly challenging in highly constrained environments due to the limited clearance for collision avoidance and the high-dimensional configuration space. This article proposes a collision-aware whole-body motion planning framework that achieves high planning success rates with low computation times in the evaluated constrained environments, and introduces a collision-aware GSG method that selects a goal state by minimizing the whole-body collision margin, thereby providing a favorable goal state for the subsequent planning pipeline.
2) The proposed method integrates a link-level whole-body collision model for aerial manipulators based on multiple spheres and incorporates an adjustment to account for the underactuated aerial platform, providing a theoretically guaranteed collision margin under bounded tilt motion. This adjusted collision-modeling idea may also be applied to other underactuated platforms, such as legged robots.
3) Unlike previous works [32], [33], [34], [35], [36], [37], which utilize the multirotor position and joint angles as optimization variables under simplified models that neglect their dynamic coupling, the proposed trajectory optimization utilizes the flat outputs of the complete aerial-manipulator system formulated by Welde et al. [22], with the CoM as the position component, thereby generating dynamically consistent trajectories by construction. Building on this formulation, our method further accounts for whole-body collisions during trajectory optimization.
4) The code will be released at https://github.com/cheungsiupaang/ManiPlanner.

Additional implementation details, discussions of practical considerations, and supporting illustrations for the ablation and sensitivity analysis section will be made available in the open-source code repository.

II. Preliminary

As shown in Fig. 1, several coordinate frames are involved, where $\{I\} = \{\boldsymbol{i}_1, \boldsymbol{i}_2, \boldsymbol{i}_3\}$ represents the inertia frame, $\{B\} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$ is the body-fixed frame, $\{F_l\} = \{\boldsymbol{x}_l, \boldsymbol{y}_l, \boldsymbol{z}_l\}$ is the frame of the l-th link, and $\{E\} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3\}$ is the EE frame.

The aerial manipulator combines an underactuated multirotor and an L-DoF robotic arm, with a total of $6 + L$ DoFs. The state variable $\boldsymbol{q}$ of the aerial manipulator is defined as

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

where $\boldsymbol{x} \in \mathbb{R}^3$ is the position of the center of mass (CoM) of the aerial manipulator with respect to the inertial frame $\{I\}$, $R \in SO(3)$ is the rotation matrix from the frame $\{B\}$ to the inertial frame $\{I\}$, and $\boldsymbol{\theta} = \{\theta_1, \theta_2, \ldots, \theta_L\} \in \Theta = \Theta_1 \times \Theta_2 \times \cdots \times \Theta_L$ is the generalized joint angle state of the robotic arm, where each $\theta_l \in \Theta_l = [\theta_{l,\min}, \theta_{l,\max}] \subset \mathbb{R}$ $(l = 1, \ldots, L)$ represents the specific range for each joint angle.

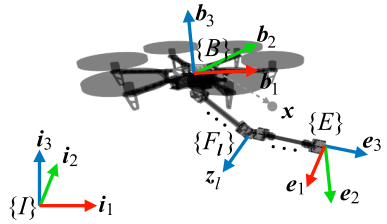
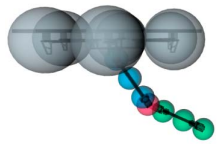


Fig. 1. Several coordinate frames of aerial manipulators.



$$\mathcal{O}_b = \{O_{b,1}, \ldots, O_{b,n_b}\}$$
$$\mathcal{O}_1 = \{O_{1,1}, \ldots, O_{1,n_1}\}$$
$$\vdots$$
$$\mathcal{O}_L = \{O_{L,1}, \ldots, O_{L,n_L}\}$$

Fig. 2. Collision model consists of multiple sets of spheres covering the aerial manipulator.

The aerial manipulator is differentially flat [22], and its flat outputs are $\boldsymbol{y} = [\boldsymbol{x}^\top, \psi, \boldsymbol{\theta}^\top]^\top$, where ψ is the yaw angle of the multirotor. Define $\boldsymbol{g}_e = \{R_e, \boldsymbol{x}_e\} \in SE(3)$ as the pose of the EE, where $\boldsymbol{x}_e \in \mathbb{R}^3$ is the position of the EE with respect to $\{I\}$, and $R_e \in SO(3)$ is the rotation matrix from the EE frame $\{E\}$ to $\{I\}$. Let $\mathcal{S} = \mathbb{R}^3 \times \mathbb{S}^1 \times \Theta$ be the configuration space of the aerial manipulator. The corresponding configuration variable $\boldsymbol{s}$ is defined as

$$\boldsymbol{s} = (\boldsymbol{x}_b, \psi, \boldsymbol{\theta}) \in \mathcal{S} = \mathbb{R}^3 \times \mathbb{S}^1 \times \Theta. \tag{2}$$

Specifically, $\boldsymbol{x}_b$ is the position of the multirotor in $\{I\}$. The pose $\boldsymbol{g}_e$ and the configuration $\boldsymbol{s}$ can be expressed as functions of the flat outputs $\boldsymbol{y}$ and their derivatives but *not vice versa*.

When the aerial manipulator hovers at a fixed position $\boldsymbol{x}_b$ with zero velocity and acceleration, the flat outputs $\boldsymbol{y}$ and configuration variable $\boldsymbol{s}$ coincide, making the two representations convertible. We define an injective mapping from $\boldsymbol{s}$ to $\boldsymbol{y}$ as

$$\begin{bmatrix} \boldsymbol{x} \\ \psi \\ \boldsymbol{\theta} \end{bmatrix} = \begin{bmatrix} f_x(\boldsymbol{s}) \\ \psi \\ \boldsymbol{\theta} \end{bmatrix} = \begin{bmatrix} \frac{1}{m}\left(m_b \boldsymbol{x}_b + \sum_l m_l \boldsymbol{x}_{cl}(\boldsymbol{x}_b, \psi, \boldsymbol{\theta})\right) \\ \psi \\ \boldsymbol{\theta} \end{bmatrix} \tag{3}$$

where m_b is the mass of the multirotor, m_l is the mass of the l-th link, and $\boldsymbol{x}_{cl}(\boldsymbol{x}_b, \psi, \boldsymbol{\theta})$ is the position of the mass center of the l-th link with respect to $\{I\}$.

III. Problem Formulation and Method Overview

A. Collision Model

As depicted in Fig. 2, the collision model of the aerial manipulator is represented by multiple sets of spheres covering the system, which is defined as $\mathcal{O} = \{\mathcal{O}_b, \mathcal{O}_1, \ldots, \mathcal{O}_L\}$.

$\mathcal{O}_b$ and $\mathcal{O}_l$ are the sets of spheres covering the multirotor and the l-th link of the robotic arm, respectively. n_b and n_l are the numbers of spheres. $\mathcal{B}(\boldsymbol{c}, r)$ denotes a sphere with center $\boldsymbol{c}$ and radius r as

$$\mathcal{B}(\boldsymbol{c}, r) = \{\boldsymbol{x} \in \mathbb{R}^3 \mid \|\boldsymbol{x} - \boldsymbol{c}\| \leq r\}. \tag{4}$$

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

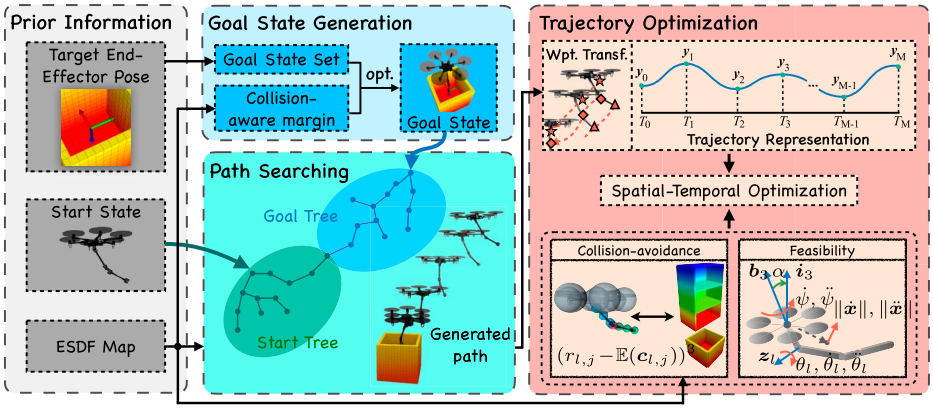


Fig. 3. Method overview. In the goal state generation (GSG) and path searching stages, the planning variables are in the configuration space $\mathcal{S}$. In the trajectory optimization stage, the optimization variables are the spatial-temporal parameters of the trajectories of the flat outputs.

$O_{b,j} = \mathcal{B}(\boldsymbol{c}_{b,j}, r_{b,j})$ and $O_{l,j} = \mathcal{B}(\boldsymbol{c}_{l,j}, r_{l,j})$ denote the j-th sphere covering the multirotor and the l-th link, respectively. $r_{b,j}, r_{l,j} \in \mathbb{R}$ and $\boldsymbol{c}_{b,j}, \boldsymbol{c}_{l,j} \in \mathbb{R}^3$ are the radii and centers of the corresponding spheres with respect to $\{I\}$, respectively. $\boldsymbol{c}_{b,j}$ and $\boldsymbol{c}_{l,j}$ can be obtained through the forward kinematics of the robotic arm, as follows:

$$\begin{bmatrix} \boldsymbol{c}_{b,j} \\ 1 \end{bmatrix} = T_b(\boldsymbol{q}) \begin{bmatrix} {}^b\boldsymbol{c}_{b,j} \\ 1 \end{bmatrix}, \quad \begin{bmatrix} \boldsymbol{c}_{l,j} \\ 1 \end{bmatrix} = T_l(\boldsymbol{q}) \begin{bmatrix} {}^l\boldsymbol{c}_{l,j} \\ 1 \end{bmatrix} \tag{5}$$

where $T_b(\boldsymbol{q}) \in \mathbb{R}^{4\times4}$ and $T_l(\boldsymbol{q}) \in \mathbb{R}^{4\times4}$ are the transformation matrices from $\{B\}$ and $\{F_l\}$ to $\{I\}$, respectively. ${}^b\boldsymbol{c}_{b,j} \in \mathbb{R}^3$ is the center of the j-th sphere with respect to $\{B\}$, and ${}^l\boldsymbol{c}_{l,j} \in \mathbb{R}^3$ is the center of the j-th sphere with respect to $\{F_l\}$.

B. Problem Statement

Unlike decoupled methods [17], [18], [19], which plan motions of the multirotor and the robotic arm periodically and thus compromise efficiency, this article investigates a whole-body planning method for aerial manipulation in constrained environments. We focus on manipulation-oriented tasks executed in a non-aggressive regime. To accomplish aerial manipulation tasks in constrained environments, the collision model introduced in Section III-A is employed to describe potential contacts between the aerial manipulator system and its surroundings. In contrast to [19], [35], [36], [37], a more accurate collision model is adopted, facilitating a more precise representation of collision interactions under cluttered and tightly constrained environments. The problem that this article aims to address is stated below.

Given a start state $\boldsymbol{q}(0) = \boldsymbol{q}_0$ with zero generalized velocity and acceleration, and the target EE pose $\boldsymbol{g}_e(T) = \boldsymbol{g}_{e,d}$ with zero target generalized velocity and acceleration, this article aims to find a nominal trajectory $\boldsymbol{y}(t)$ over $t \in [0, T]$ that is dynamically consistent, satisfies the selected feasibility constraints, and meets the collision constraints under the assumed static map and collision model.

C. Method Overview

As illustrated in Fig. 3, the proposed method, taking the start state $\boldsymbol{q}_0$, the target EE pose $\boldsymbol{g}_{e,d}$, and an Euclidean Signed Distance Field (ESDF) map as inputs, consists of the following three stages.

1) *Goal State Generation*: Generate the optimal goal state $\boldsymbol{s}_d \in \mathcal{S}$ satisfying the target EE pose $\boldsymbol{g}_{e,d}$.
2) *Path Searching*: Search for the path (waypoints) $\boldsymbol{s}_{\text{wpts}} = \{\boldsymbol{s}_0, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_d\}$ from the start state $\boldsymbol{q}_0$ to the optimal goal state $\boldsymbol{s}_d$ in the configuration space $\mathcal{S}$.
3) *Trajectory Optimization*: Generate the trajectory of the flat outputs $\boldsymbol{y}(t)$ over $t \in [0, T]$, where the searched path serves as the initial guess for the optimization.

In the above three stages, collision avoidance for the aerial manipulator system is considered. An ESDF map is available as an input and is assumed to be sufficiently accurate to represent the obstacles in the environment for planning.

IV. Collision-Aware Goal State Generation

We aim to generate a goal state $\boldsymbol{s}_d$ with zero velocity and acceleration, and owing to the redundancy of the aerial manipulator system, the same target EE pose $\boldsymbol{g}_{e,d}$ can be reached with different states. Therefore, the decision variable can be chosen from the subset $\mathcal{S}_d = \{\boldsymbol{s} \in \mathcal{S} \mid g_e(\boldsymbol{s}) = g_{e,d}\}$. For geometric collision avoidance, we introduce the concept of the *collision-avoidance margin*, which is used in generating the *optimal goal state* to meet the requirements of the collision-avoidance constraints and the target EE pose. The collision-avoidance margin is defined as follows:

$$d_{b,j} = \begin{cases} (\lambda_r r_{b,j} - \mathbb{E}(\boldsymbol{c}_{b,j}))^3, & \mathbb{E}(\boldsymbol{c}_{b,j}) < \lambda_r r_{b,j}, \\ 0, & \mathbb{E}(\boldsymbol{c}_{b,j}) \geq \lambda_r r_{b,j} \end{cases} \tag{6a}$$

$$d_{l,j} = \begin{cases} (\lambda_r r_{l,j} - \mathbb{E}(\boldsymbol{c}_{l,j}))^3, & \mathbb{E}(\boldsymbol{c}_{l,j}) < \lambda_r r_{l,j}, \\ 0, & \mathbb{E}(\boldsymbol{c}_{l,j}) \geq \lambda_r r_{l,j} \end{cases} \tag{6b}$$

where $\lambda_r \in \mathbb{R}_{>1}$ is a positive constant to determine the collision-avoidance margin. The distance $\mathbb{E}(\cdot)$ is the minimum distance which can be extracted from ESDF map. The margin in (6) is formulated as a hinge penalty, which is activated when

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

the ESDF distance falls below $\lambda_r r$ and remains zero otherwise. A cubic exponent is employed to ensure the continuous differentiability of the cost function. In contrast to lower-order penalties that may introduce gradient discontinuities, this formulation mitigates numerical oscillations and facilitates stable convergence.

The optimal goal state $\boldsymbol{s}_d$ is generated by satisfying both the collision-avoidance constraints and the desired EE pose requirement. The optimization problem is formulated to minimize the sum of the collision-avoidance margins of all spheres, which is expressed as

$$\boldsymbol{s}_d = \arg\min_{\boldsymbol{s}} \sum_{j=1}^{n_b} d_{b,j} + \sum_{l=1}^{L}\sum_{j=1}^{n_l} d_{l,j} \tag{7a}$$

$$s.t.\ \boldsymbol{s} \in \mathcal{S}_d \tag{7b}$$

$$\mathbb{E}(\boldsymbol{c}_{b,j}) \geq r_{b,j},\ \forall j = 1,2,\ldots,n_b \tag{7c}$$

$$\mathbb{E}(\boldsymbol{c}_{l,j}) \geq r_{l,j},\ \forall j = 1,2,\ldots,n_l,\ \forall l = 1,2,\ldots,L. \tag{7d}$$

By solving the optimization problem (7), an optimal goal state $\boldsymbol{s}_d$ is obtained that satisfies both the collision-avoidance constraints and the desired EE pose.

V. Path Searching in the Configuration Space

This section aims to find a path $\boldsymbol{s}_{\text{wpts}}$ that passes the geometric collision checks and connects the start state $\boldsymbol{s}_0$ to the goal state $\boldsymbol{s}_d$ in the configuration space $\mathcal{S}$. Since the motion costs of all the states in the searching variable $\boldsymbol{s} = \{\boldsymbol{x}_b, \psi, \boldsymbol{\theta}\}$ need to be considered, a weighted cost formulation is required, as detailed in Section V-A. During the path searching stage, the velocities and accelerations are not taken into account and therefore the collision model would be adjusted, as illustrated in Section V-B. The Bi-RRT* algorithm is employed to find the path, as detailed in Section V-C.

A. Weighted Sum Cost

Since the configuration space $\mathcal{S}$ contains heterogeneous components ($\boldsymbol{x}_b$, ψ, and $\boldsymbol{\theta}$), a weighted sum is adopted to define the cost $J(\boldsymbol{s}_1, \boldsymbol{s}_2)$ between two states $\boldsymbol{s}_1, \boldsymbol{s}_2 \in \mathcal{S}$

$$J(\boldsymbol{s}_1, \boldsymbol{s}_2) = \omega_t J_x(\boldsymbol{s}_1, \boldsymbol{s}_2) + \omega_r J_\psi(\boldsymbol{s}_1, \boldsymbol{s}_2) + \omega_r J_\theta(\boldsymbol{s}_1 \boldsymbol{s}_2)$$

where ω_t and ω_r are the weights, and the cost components are defined as

$$\begin{aligned} J_x(\boldsymbol{s}_1, \boldsymbol{s}_2) &= \|\boldsymbol{x}_{b,1} - \boldsymbol{x}_{b,2}\|, \\ J_\psi(\boldsymbol{s}_1, \boldsymbol{s}_2) &= \min\left(|\psi_1 - \psi_2|, 2\pi - |\psi_1 - \psi_2|\right), \\ J_\theta(\boldsymbol{s}_1, \boldsymbol{s}_2) &= \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_1. \end{aligned}$$

Here, J_x measures the translational distance, J_ψ computes the minimum angular difference accounting for yaw periodicity, and J_θ measures the total joint angle changes. The weights ω_t and ω_r are introduced to normalize the numerical scale difference between translational and angular motions. This normalization helps mitigate bias in Bi-RRT* tree expansion toward the subspace with larger numerical ranges.

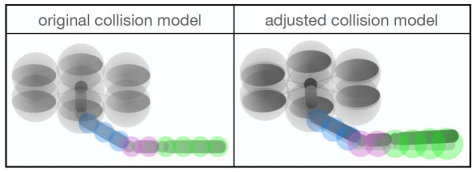


Fig. 4. Original collision model used in the goal state generation and trajectory optimization stages, and the adjusted collision model used in the path searching stage.

B. Adjusted Collision Model

In the path searching stage, the tilt motion of the multirotor is not considered to enable efficient searching. To account for tilt-induced geometric displacement under the tilt-angle bound $\alpha_{\max}$, an adjusted collision model is adopted using simplified forward kinematics and inflated sphere radii, as shown in Fig. 4.

1) The sphere centers are adjusted via forward kinematics

$$\begin{bmatrix} \boldsymbol{c}'_{b,j} \\ 1 \end{bmatrix} = T_b(\boldsymbol{s}) \begin{bmatrix} {}^b\boldsymbol{c}_{b,j} \\ 1 \end{bmatrix}, \quad \begin{bmatrix} \boldsymbol{c}'_{l,j} \\ 1 \end{bmatrix} = T_l(\boldsymbol{s}) \begin{bmatrix} {}^l\boldsymbol{c}_{l,j} \\ 1 \end{bmatrix} \tag{9}$$

where $T_b(\boldsymbol{s}) \in \mathbb{R}^{4\times4}$ and $T_l(\boldsymbol{s}) \in \mathbb{R}^{4\times4}$ are the transformation matrices from $\{B\}$ and $\{F_l\}$ to $\{I\}$, respectively, through simplified kinematics, where the tilt motion of the multirotor is neglected.

2) The sphere radii are inflated to account for the neglected tilt motion

$$r'_{b,j} = r_{b,j} + 2\|{}^b\boldsymbol{c}_{b,j}\|_{\max} \cdot \sin\frac{\alpha_{\max}}{2} \tag{10a}$$

$$r'_{l,j} = r_{l,j} + 2\|{}^b\boldsymbol{c}_{l,j}\|_{\max} \cdot \sin\frac{\alpha_{\max}}{2} \tag{10b}$$

where $\alpha_{\max}$ is the maximum tilt angle of the multirotor ($0 \leq \alpha \leq \pi/2$).

Remark 1: The inflated radii $r'_{b,j}$ and $r'_{l,j}$ conservatively account for tilt-induced displacement for all $\alpha \leq \alpha_{\max}$. Taking $r'_{l,j}$ as an example

$$\begin{aligned} r'_{l,j} &= r_{l,j} + 2\|{}^b\boldsymbol{c}_{l,j}\|_{\max} \cdot \sin\frac{\alpha_{\max}}{2} \\ &\geq r_{l,j} + 2\left(\left\|{}^b\boldsymbol{c}_{l,j}\right\| \cdot \sin\frac{\alpha}{2}\right)_{\max} \\ &\geq r_{l,j} + 2\left\|{}^b\boldsymbol{c}_{l,j}\right\| \cdot \sin\frac{\alpha}{2}. \end{aligned}$$

The inflation term provides an upper bound derived from the triangle inequality and the spectral norm of the attitude rotation difference for tilt angles bounded by $\alpha_{\max}$. Therefore, it provides a conservative containment bound for the adjusted collision model without relying on empirical heuristics.

The adjusted collision model is used only in the path searching stage to provide an initial guess for the subsequent trajectory optimization, and the optimized trajectory is not constrained to pass through the searched waypoints. The radius inflation in (10) is employed to maintain waypoint generation under bounded tilt uncertainty while ensuring computational efficiency, while the finalized whole-body motion is determined by the trajectory optimization stage.

C. Bi-RRT* Algorithm

We present a Bi-RRT* algorithm in the configuration space $\mathcal{S}$, as illustrated in Alg. 1. The objective of the algorithm is to

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Algorithm 1: Bi-RRT* in the configuration space $\mathcal{S}$.

Input: step sizes δ_t, δ_r; thresholds σ_t, σ_r; states $\boldsymbol{s}_0, \boldsymbol{s}_d$; limits $I_{\max}, c_{\max}$
Output: waypoints $\boldsymbol{s}_{\text{wpts}} = \{\boldsymbol{s}_0, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_d\}$

$\mathcal{T}_{\text{start}} \leftarrow \{\boldsymbol{s}_0\}$, $\mathcal{T}_{\text{goal}} \leftarrow \{\boldsymbol{s}_d\}$
$flag \leftarrow$ true, $iter \leftarrow 0$
while $iter < I_{\max}$ **do**
  if $flag$ **then**
    $\mathcal{T}_{\text{current}} \leftarrow \mathcal{T}_{\text{start}}$, $\mathcal{T}_{\text{target}} \leftarrow \mathcal{T}_{\text{goal}}$
  else
    $\mathcal{T}_{\text{current}} \leftarrow \mathcal{T}_{\text{goal}}$, $\mathcal{T}_{\text{target}} \leftarrow \mathcal{T}_{\text{start}}$
  $flag \leftarrow \neg flag$, $iter \leftarrow iter + 1$
  $\boldsymbol{s}_{\text{rand}} \leftarrow \texttt{Sample}(\mathcal{T}_{\text{target}})$
  $\boldsymbol{s}_{\text{near}} \leftarrow \texttt{Nearest}(\mathcal{T}_{\text{current}}, \boldsymbol{s}_{\text{rand}})$
  $\boldsymbol{s}_{\text{new}} \leftarrow \texttt{Steer}(\boldsymbol{s}_{\text{near}}, \boldsymbol{s}_{\text{rand}}, \delta_t, \delta_r)$
  if $\texttt{CollisionFree}(\boldsymbol{s}_{new}, \boldsymbol{s}_{new})$ **then**
    $\texttt{AddNode}(\mathcal{T}_{\text{current}}, \boldsymbol{s}_{\text{new}})$
    $\texttt{AddEdge}(\boldsymbol{s}_{\text{near}}, \boldsymbol{s}_{\text{new}})$
    $\texttt{RewireTree}(\mathcal{T}_{\text{current}}, \boldsymbol{s}_{\text{new}})$
    $\boldsymbol{s}_{\text{cand}} \leftarrow \texttt{FindCandidate}(\boldsymbol{s}_{\text{new}}, \mathcal{T}_{\text{target}})$
    if $\texttt{Connectable}(\boldsymbol{s}_{new}, \boldsymbol{s}_{cand}, \sigma_t, \sigma_r)$ **then**
      if $flag$ **then**
        $\boldsymbol{s}_{\text{wpts}} \leftarrow \{\boldsymbol{s}_0, \ldots, \boldsymbol{s}_{\text{new}}, \boldsymbol{s}_{\text{cand}}, \ldots, \boldsymbol{s}_d\}$
      else
        $\boldsymbol{s}_{\text{wpts}} \leftarrow \{\boldsymbol{s}_0, \ldots, \boldsymbol{s}_{\text{cand}}, \boldsymbol{s}_{\text{new}}, \ldots, \boldsymbol{s}_d\}$
    return $\boldsymbol{s}_{\text{wpts}} \leftarrow \texttt{Sparse}(\boldsymbol{s}_{\text{wpts}}, c_{\max})$

find a sequence $\boldsymbol{s}_{\text{wpts}} = \{\boldsymbol{s}_0, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_d\}$ that connects the start state $\boldsymbol{s}_0$ to the goal state $\boldsymbol{s}_d$ and passes collision checks using the adjusted collision model.

- $\texttt{Sample}(\mathcal{T}_{\text{target}})$: Randomly sample a state $\boldsymbol{s}_{\text{rand}} \in \mathcal{S}$. A certain probability p_{root} is assigned to directly sample the root node ($\boldsymbol{s}_0$ or $\boldsymbol{s}_d$) of $\mathcal{T}_{\text{target}}$ as $\boldsymbol{s}_{\text{rand}}$.
- $\texttt{Nearest}(\mathcal{T}_{\text{current}}, \boldsymbol{s}_{\text{rand}})$: Find the nearest node $\boldsymbol{s}_{\text{near}}$ in the tree $\mathcal{T}_{\text{current}}$ to the state $\boldsymbol{s}_{\text{rand}}$.
- $\texttt{Steer}(\boldsymbol{s}_{\text{near}}, \boldsymbol{s}_{\text{rand}}, \delta_t, \delta_r)$: Steer the state $\boldsymbol{s}_{\text{near}}$ toward the state $\boldsymbol{s}_{\text{rand}}$ with the steps δ_t and δ_r.

$$\begin{cases} \boldsymbol{x}_{b,\text{new}} = \boldsymbol{x}_{b,\text{near}} + \delta_t(\boldsymbol{x}_{b,\text{rand}} - \boldsymbol{x}_{b,\text{near}})/J_x(\boldsymbol{s}_{\text{near}}, \boldsymbol{s}_{\text{rand}}), \\ \psi_{\text{new}} = \psi_{\text{near}} + \delta_r(\psi_{\text{rand}} - \psi_{\text{near}})/J_\psi(\boldsymbol{s}_{\text{near}}, \boldsymbol{s}_{\text{rand}}), \\ \boldsymbol{\theta}_{\text{new}} = \boldsymbol{\theta}_{\text{near}} + \delta_r(\boldsymbol{\theta}_{\text{rand}} - \boldsymbol{\theta}_{\text{near}})/J_\theta(\boldsymbol{s}_{\text{near}}, \boldsymbol{s}_{\text{rand}}). \end{cases}$$

- $\texttt{Sparse}(\boldsymbol{s}_{\text{wpts}}, c_{\max})$: Sparsify overly dense waypoints induced by the fixed step size while keeping the segment span bounded in practice, by limiting the maximum number of skipped waypoints between two retained waypoints to $c_{\max}$.

The Bi-RRT* module is employed as a sampling-based planner to efficiently obtain a waypoint sequence that passes the geometric collision checks in configuration space $\mathcal{S}$ as an initial guess for the subsequent trajectory optimization. Due to practical heuristics such as biased sampling, threshold-based connection rules, and local rewiring, we do not claim the asymptotic optimality. These heuristics are introduced to enhance practical planning efficiency and facilitate the rapid generation of feasible waypoint sequences.

VI. Trajectory Optimization

In this section, we present the proposed method for trajectory optimization with the flat outputs $\boldsymbol{y}$. Unlike prior works [32], [33], [34], [35], [36], [37], which utilize the multirotor position and joint angles as optimization variables under simplified models that neglect their dynamic coupling, the proposed method uses the complete aerial-manipulator flat outputs as established by Welde et al. [22], thereby generating dynamically consistent trajectories by construction. The original collision model illustrated in Fig. 2 is utilized in trajectory optimization to evaluate whole-body collision avoidance.

A. Trajectory Representation and Initial Guess

M pieces of N $(= 4 + L)$-dimensional polynomials with degree $D - 1$ are utilized to represent the whole trajectory of the aerial manipulator. The i-th $(i = 1, 2, \ldots, M)$ piece of the trajectory is expressed as

$$\boldsymbol{y}_i(\tau) = \mathbf{C}_i^\top \boldsymbol{\beta}(\tau), \ \tau \in [0, \mathrm{T}_i] \tag{11}$$

where $\mathbf{C}_i \in \mathbb{R}^{D \times N}$ is the coefficient matrix of the i-th piece of the trajectory, $\boldsymbol{\beta}(\tau) = [1, \tau, \tau^2, \ldots, \tau^{(D-1)}]^\top \in \mathbb{R}^D$ is the polynomial basis function, τ is the local time variable, and T_i is the duration of the i-th piece. Let $\mathbf{T} = [\mathrm{T}_1, \mathrm{T}_2, \ldots, \mathrm{T}_\mathrm{M}] \in \mathbb{R}^\mathrm{M}$ denote the time durations for each piece and $\mathbf{C} = [\mathbf{C}_1^\top, \mathbf{C}_2^\top, \ldots, \mathbf{C}_M^\top]^\top \in \mathbb{R}^{MD \times N}$ denote the coefficients of the whole trajectory. The whole time of the trajectory is $\mathrm{T} = \sum_{i=1}^{M} \mathrm{T}_i$. With the aforementioned notation, the trajectory for flat outputs $\boldsymbol{y}(t)$ is parametrized by the coefficients $\mathbf{C}$ and the time durations $\mathbf{T}$.

The generated waypoints $\boldsymbol{s}_{\text{wpts}}$ derived from the path searching stage in Section V have been sparsified to obtain M waypoints $\{\boldsymbol{s}_0, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_M\}$, where $\boldsymbol{s}_d = \boldsymbol{s}_M$ is the goal state. The searched waypoints are then converted into waypoints of the flat outputs $\boldsymbol{y}_{\text{wpts}} = \{\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_M\}$ as initial guesses for the following optimization process through the injective mapping in (3). The initial guess of the time durations $\mathbf{T}$ is set as the maximum of the distances between two consecutive waypoints for each component divided by their respective average velocities, i.e.,

$$\mathrm{T}_i = \max \left\{ \begin{array}{c} \|\boldsymbol{x}_{b,i} - \boldsymbol{x}_{b,i-1}\|/v_{t,\text{avg}} \\ |\psi_i - \psi_{i-1}|/v_{r,\text{avg}} \\ \max\limits_{l=1,\ldots,L} |\theta_{l,i} - \theta_{l,i-1}|/v_{r,\text{avg}} \end{array} \right\} \tag{12}$$

where $i = 1, \ldots, M$, and $v_{t,\text{avg}}$, $v_{r,\text{avg}}$, and $v_{r,\text{avg}}$ are given parameters representing the average velocities for the position, yaw angle, and joint angles, respectively.

B. Spatial-Temporal Trajectory Optimization

With the trajectory representation in (11), we can formulate the trajectory optimization problem as follows:

$$\begin{aligned} \min_{\mathbf{C},\mathbf{T}} \sum_{i=1}^{M} & J_{s,i} + J_{t,i} + J_{\text{vel},x,i} + J_{\text{vel},\psi,i} + J_{\text{vel},\theta,i} \\ & + J_{\text{acc},x,i} + J_{\text{acc},\psi,i} + J_{\text{acc},\theta,i} \\ & + J_{\text{tilt},i} + J_{\text{joint},i} + J_{\text{col},i}. \end{aligned} \tag{13}$$

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

The cost function balances the smoothness cost $J_{s,i}$ and the time allocation $J_{t,i}$, which are defined as follows:

$$J_{s,i} = \int_0^{\mathrm{T}_i} \boldsymbol{y}_i^{(s)\top}(\tau)\boldsymbol{y}_i^{(s)}(\tau)\mathrm{d}\tau \tag{14a}$$

$$J_{t,i} = \lambda_T \mathrm{T}_i \tag{14b}$$

where λ_T is a scalar constant. The cost function also balances the other constraints, which would be illustrated in Section VI-C and Section VI-D.

C. Feasibility Constraints

In this section, we consider the feasibility of the trajectory, including the velocity limits, the acceleration limits, tilt angle limit, and joint angle limits. To reduce computational complexity, the feasibility constraints are computed using discrete sampling with K sampling points per segment. The sampling points are uniformly distributed in the interval $[0, \mathrm{T}_i]$, i.e., $\tau_k = k\mathrm{T}_i/K$, $k = 0, 1, \ldots, K-1$. To guarantee the continuous differentiability required by the gradient-based optimizer, a smooth penalty function $\mathcal{L}(x)$, adopted from the formulation in [42], is utilized to evaluate constraint violations. This smooth formulation ensures that both the penalty term and its corresponding gradient transition continuously to zero, contributing to the cost function only upon constraint violation.

1) Velocity Limits: To ensure the generated trajectory is physically realizable, we impose bounds on the velocities for each dimension of the system states, which is as follows:

$$J_{\mathrm{vel},x,i} = \lambda_{\mathrm{vel},t} \sum_{k=0}^{K-1} \mathcal{L}\left(\|\dot{\boldsymbol{x}}_i(\tau_k)\|^2 - v_{t,\max}^2\right) \mathrm{T}_i/K \tag{15a}$$

$$J_{\mathrm{vel},\psi,i} = \lambda_{\mathrm{vel},r} \sum_{k=0}^{K-1} \mathcal{L}\left(\dot{\psi}_i(\tau_k)^2 - v_{r,\max}^2\right) \mathrm{T}_i/K \tag{15b}$$

$$J_{\mathrm{vel},\theta,i} = \lambda_{\mathrm{vel},r} \sum_{k=0}^{K-1} \sum_{l=1}^{L} \mathcal{L}\left(\dot{\theta}_{l,i}(\tau_k)^2 - v_{r,\max}^2\right) \mathrm{T}_i/K \tag{15c}$$

where $\lambda_{\mathrm{vel},t}$, $\lambda_{\mathrm{vel},r}$ are scalar weights for the velocity limits, $v_{t,\max}$, $v_{r,\max}$ are the maximum velocities for translational motion and rotational motion, respectively.

2) Acceleration Limits: Similar to the velocity limits, the acceleration limits are computed using discrete sampling with K sampling points per segment. The acceleration limits are defined as follows:

$$J_{\mathrm{acc},x,i} = \lambda_{\mathrm{acc},t} \sum_{k=0}^{K-1} \mathcal{L}\left(\|\ddot{\boldsymbol{x}}_i(\tau_k)\|^2 - a_{t,\max}^2\right) \mathrm{T}_i/K \tag{16a}$$

$$J_{\mathrm{acc},\psi,i} = \lambda_{\mathrm{acc},r} \sum_{k=0}^{K-1} \mathcal{L}\left(\ddot{\psi}_i(\tau_k)^2 - a_{r,\max}^2\right) \mathrm{T}_i/K \tag{16b}$$

$$J_{\mathrm{acc},\theta,i} = \lambda_{\mathrm{acc},r} \sum_{k=0}^{K-1} \sum_{l=1}^{L} \mathcal{L}\left(\ddot{\theta}_{l,i}(\tau_k)^2 - a_{r,\max}^2\right) \mathrm{T}_i/K \tag{16c}$$

where $\lambda_{\mathrm{acc},t}$, $\lambda_{\mathrm{acc},r}$ are scalar weights for the acceleration limits, $a_{t,\max}$, $a_{r,\max}$ are the maximum accelerations for translational motion and rotational motion, respectively.

3) Joint Angle Limit: For each joint angle θ_l, the joint angle limit is defined as $\theta_l \in [\theta_{l,\min}, \theta_{l,\max}]$. The joint angle limit is defined as follows:

$$\begin{aligned} J_{\mathrm{joint},i} = {} & \lambda_{\mathrm{joint}} \sum_{k=0}^{K-1} \sum_{l=1}^{L} \mathcal{L}\left(\theta_{l,i}(\tau_k) - \theta_{l,\max}\right) \mathrm{T}_i/K \\ & + \lambda_{\mathrm{joint}} \sum_{k=0}^{K-1} \sum_{l=1}^{L} \mathcal{L}\left(\theta_{l,\min} - \theta_{l,i}(\tau_k)\right) \mathrm{T}_i/K. \end{aligned} \tag{17}$$

4) Maximum Tilt Angle Limit: The maximum tilt angle $\alpha_{\max}$ constraint is enforced as

$$J_{\mathrm{tilt},i} = \lambda_{\mathrm{tilt}} \sum_{k=0}^{K-1} \mathcal{L}\left(\cos\alpha_{\max} - \cos\alpha_i(\tau_k)\right) \mathrm{T}_i/K \tag{18}$$

where λ_{tilt} is a scalar weight, and the tilt angle $\alpha_i(\tau_k)$ is related to the CoM acceleration $\ddot{\boldsymbol{x}}_i(\tau_k)$ via

$$\cos\alpha_i(\tau_k) = \left(\ddot{x}_{i,3}(\tau_k) + g\right) / \|\ddot{\boldsymbol{x}}_i(\tau_k) + g\boldsymbol{e}_3\| \tag{19}$$

where $\ddot{x}_{i,3}$ is the 3-rd element of the acceleration $\ddot{\boldsymbol{x}}_i$.

Although explicit constraints on thrust and actuator saturation are not directly imposed, the acceleration limits, maximum tilt angle limit, and smoothness term implicitly influence the required thrust and control effort, thereby accounting for actuator limitations in practice.

D. Collision-Avoidance Constraints

The collision-avoidance constraints adopt the concept defined in (6) in Section IV, where the sphere centers are obtained through the forward kinematics based on the system state $\boldsymbol{q}$ derived from the flat outputs $\boldsymbol{y}$. The whole-body collision-avoidance cost is defined as

$$\begin{aligned} J_{\mathrm{col},i} = {} & \lambda_{\mathrm{col}} \sum_{k=0}^{K-1} \sum_{j=1}^{n_b} d_{b,i,j}(\tau_k)\mathrm{T}_i/K \\ & + \lambda_{\mathrm{col}} \sum_{k=0}^{K-1} \sum_{l=1}^{L} \sum_{j=1}^{n_l} d_{l,i,j}(\tau_k)\mathrm{T}_i/K \end{aligned} \tag{20}$$

where $d_{b,i,j}(\tau_k)$ and $d_{l,i,j}(\tau_k)$ are the collision-avoidance margins of the j-th sphere covering the multirotor and the l-th link, respectively, at the k-th sampling point τ_k.

The gradients of all cost functions with respect to the optimization variables are computed analytically to improve efficiency. A postoptimization verification process is conducted to ensure the effectiveness of the solution. The refined trajectory is densely sampled to verify not only the obstacle-avoidance constraints, expressed as $\mathbb{E}(c_{b,j}) > r_{b,j}$ and $\mathbb{E}(c_{l,j}) > r_{l,j}$ for all spheres, but also the feasibility constraints. If any violation is detected, the trajectory is rejected, and the planner re-solves the optimization problem.

Remark 2: The constraints are resolution-dependent without strict continuous-time guarantees. Although evaluated at discrete samples, missed violations are mitigated in practice by jointly controlling the segment span via $\mathtt{Sparse}(\boldsymbol{s}_{\mathrm{wpts}}, c_{\max})$ and the per-segment sampling number K.

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE I
DH PARAMETERS AND MASS PARAMETERS

Component	a_{i-1} (m)	α_{i-1} (rad)	d_i (m)	θ_i (rad)	mass (kg)
Multirotor	/	/	/	/	2.26
Link 1	0.0	0.0	0.0	θ_1	0.17
Link 2	0.2392	0.0	0.0	θ_2	0.16
Link 3	0.0	$-\pi/2$	0.0236	θ_3	0.12
Link 4	0.0	0.0	0.29	0.0	0.23

TABLE II
PARAMETERS USED IN THE PROPOSED FRAMEWORK

Symbol	Description	Value
collision-avoidance parameter		
λ_r	margin scaling factor	3.0
path searching		
ω_t	translational motion weight	0.7
ω_r	rotational motion weight	1.0
δ_t	translational step size	0.05
δ_r	rotational step size	0.025
σ_t	translational threshold	0.06
σ_r	rotational threshold	0.04
p_{root}	root node sampling probability	0.25
$c_{\max}$	maximum skipped waypoints	4
trajectory optimization		
D	number of polynomial coefficients	8
s	derivative order for smoothness cost	3
λ_T	time allocation weight	1.0
$\lambda_{\text{vel},t/r}$, $\lambda_{\text{acc},t/r}$ $\lambda_{\text{joint}}, \lambda_{\text{tilt}}$	weights of feasibility constraints	20.0
λ_{col}	weight of collision-avoidance	1900.0

VII. RESULTS

A. Implementation

The aerial manipulator system consists of an F550 hexrotor and a custom-built robotic arm. The Denavit-Hartenberg (DH) parameters and mass parameters are shown in Table I.

Our algorithm is implemented with C++. The optimization problem (7) is solved through the NLopt Library[1]. The problem (13) is solved by the L-BFGS method [43]. All simulations are conducted on a computer with Intel Core i7–10700K. The used ESDF map is generated [44], with the voxel resolution maintained at 0.02 m. The collision-sphere radii defined in Section III-A are set to $r_{b,j} = 0.25$ m and $r_{l,j} = 0.10$ m for all corresponding spheres. The velocity limits are $v_{t,\max} = 0.5$ m/s and $v_{r,\max} = 0.3$ rad/s, the acceleration limits are $a_{t,\max} = 0.3$ m/s^2 and $a_{r,\max} = 0.2$ rad/s^2, and the maximum tilt is specified by $\cos\alpha_{\max} = 0.98$. A total of $K = 12$ samples per segment are used. The algorithmic parameters listed in Table II are tuned stage by stage in representative constrained scenarios. Tuning continues until each stage consistently yields a valid result: a collision-aware goal state, a valid waypoint sequence, and a feasible trajectory for GSG, path searching, and trajectory optimization, respectively.

[1]https://nlopt.readthedocs.io/en/latest/

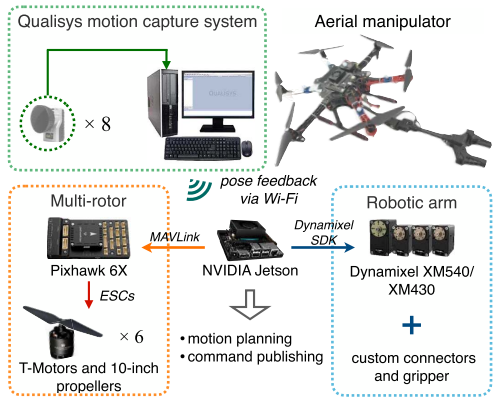


Fig. 5. Experiment setup of the aerial manipulator.

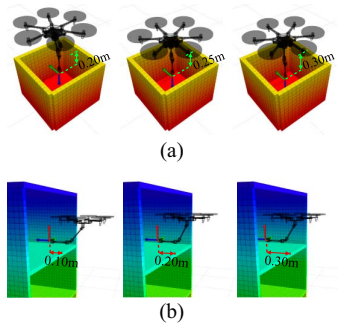


Fig. 6. Illustration of goal state generation. (a) Scenario1: Inserting the end-effector into a bin. (b) Scenario 2: Inserting the end-effector into a shelf.

The experimental setup is shown in Fig. 5. The hexrotor comprises a Pixhawk 6X flight controller and six T-Motor/10-inch-propeller pairs driven through ESCs. It uses a cascaded feedback controller with feedforward compensation for robotic-arm motion, and the flight-control code employed in all demonstration experiments is available online[2]. The robotic arm is built with ROBOTIS Dynamixel XM540/XM430 servos and runs the standard Dynamixel servo controller. An eight-camera Qualisys system sends pose feedback via Wi-Fi to the NVIDIA Jetson NX, which performs motion planning and sends commands to the Pixhawk through MAVLink and to the arm through the Dynamixel SDK.

B. Performance Evaluation

1) Goal State Generation: The computational efficiency of GSG (7) is evaluated by inserting the EE into a bin at depths of 0.20, 0.25, and 0.30 m and into a shelf at depths of 0.10, 0.20, and 0.30 m, as depicted in Fig. 6. All other target-pose components remain fixed. Table III reports the mean computation time and standard deviation over 20 runs, where '0xx' denotes the insertion depth. The method achieves average computation times below 2 ms and a 100% success rate in the evaluated cases.

2) Path Searching: We compare our method with the method proposed by Caballero et al. [31], RRT* w/o GSG, and RRT*

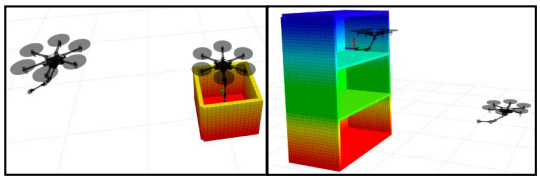
[2]https://github.com/cheungsiupaang/PX4-Autopilot/

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE III
QUANTITATIVE RESULTS IN THE GSG STAGE

Scenario		Aver. (ms)	Std. Dev. (ms)	Success Rate
Scenario 1 (bin)	020	1.52	0.61	100%
	025	1.50	0.60	100%
	030	1.49	0.74	100%
Scenario 2 (shelf)	010	1.82	0.59	100%
	020	1.90	0.37	100%
	030	1.75	0.48	100%



(a) (b)

Fig. 7. Start and goal states for path searching. (a) Scenario 1. (b) Scenario 2.

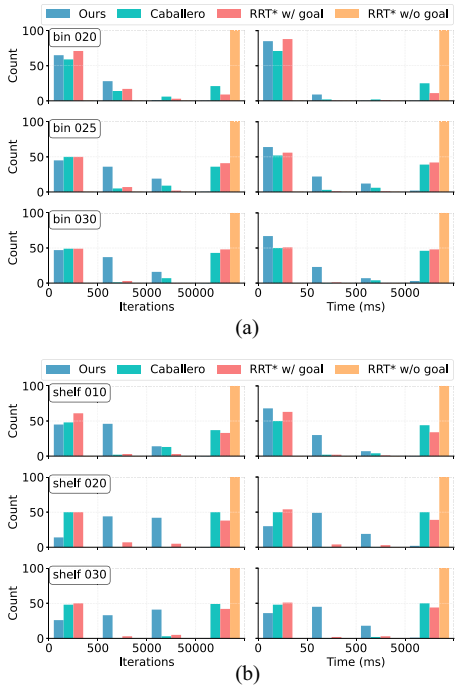


Fig. 8. Controlled path-searching comparison of iterations and computation time among Ours, Caballero et al. [31], RRT* w/ GSG, and RRT* w/o GSG in Scenarios 1 and 2. (a) Scenario 1: Inserting the end-effector into a bin. (b) Scenario 2: Inserting the end-effector into a shelf.

w/GSG. These compared methods perform coupled path searching over the same whole-body configuration space. This is a controlled comparison conducted at the path searching stage, and the reported results pertain only to this stage rather than the complete pipeline. Using the start and goal states in Fig. 7, we test the four probabilistically complete, sampling-based methods for 100 runs each in two scenarios with varying insertion depths. The deeper the insertion, the more challenging the search becomes. The iteration counts and computation times for each run are presented in Fig. 8.

Our method requires fewer iterations and less computation time than the three compared methods. RRT* w/o goal state bias fails in all cases within 50 000 iterations or 5000 ms, indicating the importance of GSG in Section IV for sampling-based planning in high-dimensional spaces. Caballero et al. [31] and RRT* w/ goal state bias exhibit bimodal distributions in both metrics: the left peaks represent rapid solutions enabled by goal-state sampling, whereas the right peaks represent difficult searches. In contrast, our method consistently finds feasible paths with lower variability.

3) Trajectory Optimization: The proposed method is compared with Deng et al. [35] and Zhang et al. [36]. This trajectory-optimization experiment is designed as a controlled comparison of collision-avoidance formulations. Both compared methods are coupled trajectory-optimization formulations for aerial manipulators. To evaluate the performance of trajectory optimization, tests are conducted across two scenarios, including inserting the EE into a bin and a shelf. Both scenarios include three varying insertion depths. For each depth, 50 runs use distinct, valid Bi-RRT* paths as common initial guesses. Only the collision-avoidance cost varies among the proposed multisphere, dynamic ellipsoid [35], and dynamic convex polyhedron [36] models. Since the constraints are penalty-based, success rates are uniformly evaluated using the multisphere model.

Table IV summarizes the quantitative results. The proposed formulation achieves success rates above 92% across all insertion depths. Under the common formulation and verification criterion, the dynamic ellipsoid approximation yields low success rates, while the dynamic convex polyhedron performs moderately in the bin and shallowest shelf cases but fails at greater depths. Its conservatism also leads to longer trajectory execution times and larger smoothness costs J_s. The proposed method completes optimization in 79.77–134.04 ms, compared with 194.59–446.83 ms for Zhang et al. [36], whose convex approximation more severely restricts the feasible solution space. This is a controlled formulation-level evaluation, not an end-to-end comparison. The low ellipsoid success rate reflects its limited effectiveness under our common formulation and verification criterion, not a general limitation of ellipsoidal collision modeling or the pipeline in [35].

C. Ablation and Sensitivity Analysis

To further evaluate the planning performance of the proposed framework, ablation and sensitivity analyses are conducted. For Tables V–VIII, each configuration is evaluated over a fixed budget of 40 independent trials. Success rates are reported with two-sided Wilson 95% confidence intervals, and computation-time statistics include successful trials only. Stage computation times are reported as medians and total time is reported as median [Q1, Q3], where Q1 and Q3 denote the first and third quartiles, respectively.

1) Ablation on Goal State Generation: Collision-aware GSG is compared with random feasible goal-state selection and a

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE IV
QUANTITATIVE COMPARISON OF COLLISION-AVOIDANCE FORMULATIONS IN TRAJECTORY OPTIMIZATION

Scenario		Method	Success Rate	Smoothness Cost J_s	Trajectory Time (s)	Computation Time (ms)
Scenario 1 (bin)	020	Proposed	**98%**	**0.350**	**8.650**	**89.09**
		Deng et al. [35]	0%	—	—	—
		Zhang et al. [36]	84%	0.364	9.019	213.62
	025	Proposed	**94%**	**0.352**	**8.660**	**107.41**
		Deng et al. [35]	0%	—	—	—
		Zhang et al. [36]	80%	0.386	9.369	245.91
	030	Proposed	**92%**	**0.381**	**9.407**	**100.29**
		Deng et al. [35]	0%	—	—	—
		Zhang et al. [36]	74%	0.397	9.447	277.52
Scenario 2 (shelf)	010	Proposed	**98%**	0.245	8.313	**79.77**
		Deng et al. [35]	2%	0.925†	6.418†	51.85†
		Zhang et al. [36]	**98%**	**0.195**	**7.875**	194.59
	020	Proposed	**96%**	**0.256**	**8.502**	**125.07**
		Deng et al. [35]	0%	—	—	—
		Zhang et al. [36]	0%	—	—	—
	030	Proposed	**98%**	**0.243**	**8.540**	**134.04**
		Deng et al. [35]	0%	—	—	—
		Zhang et al. [36]	8%	0.324	9.654	446.83

Smoothness cost J_s is calculated through (14a) and the trajectory time is the time of executing the optimized trajectory. Bold entries indicate the best performance for each metric within each test case, excluding values marked with †. Consequently, these values lack statistical significance and should be interpreted as a coincidental occurrence.
† The quantitative metrics for this specific test case are derived from a single successful trial out of 50 runs.

TABLE V
PLANNING PERFORMANCE WITH DIFFERENT GSG STRATEGIES

GSG method	Success rate (%)	$CI_{95\%}$ (%)	Stage time (ms)			Total time (ms)
			t_g	t_s	t_o	
Collision-aware	95.0	[83.5, 98.6]	2.7	158.8	161.8	331.0 [248.3, 537.5]
Random feasible	85.0	[70.9, 92.9]	1.0	143.2	147.7	314.7 [243.8, 500.4]
Heuristic	82.5	[68.0, 91.3]	1.8	168.6	167.6	371.2 [232.4, 776.1]

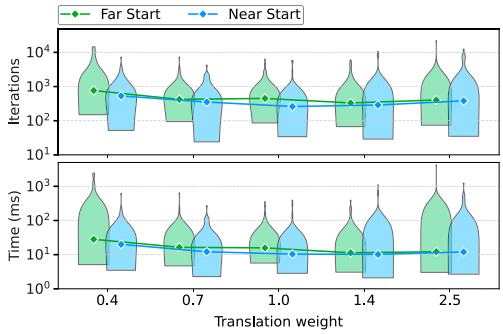


Fig. 9. Iterations and computation times under varying translation weights ω_t, where violin plots and solid lines denote the distribution and medians across Far Start (1.8 m) and Near Start (0.8 m) scenarios.

minimum joint-displacement heuristic in the constrained "bin 030" scenario, with the remaining planning pipeline unchanged. Over 40 trials, it achieves a 95.0% success rate, compared with 85.0% and 82.5% for the two baselines, as shown in Table V, reducing the number of failures from 6 and 7 to 2, respectively. Although the differences are not statistically significant under two-sided Fisher's exact tests ($p = 0.263$ and $p = 0.154$), collision-aware GSG consistently yields a higher empirical success rate.

2) Sensitivity to Distance Metric Weights: To evaluate the weight sensitivity during path searching, tests are conducted across scales. In "bin 030", ω_r is kept fixed and ω_t varies under two settings: "Far Start", which features a 1.8 m horizontal distance to evaluate long-distance translational motion, and "Near Start", with a 0.8 m distance to test angular motion. Each setup undergoes 50 runs. Fig. 9 reveals a logical dependency. For "Far Start", larger ω_t accelerates convergence via rapid translational coverage. In "Near Start", a large ω_t restricts fine-grained attitude adjustments in confined spaces by overly penalizing translation. Despite degradation at extremes, all tested ω_t values generate feasible trajectories within 50 000 iterations or 5000 ms in both start settings.

3) Scalability to Collision Discretization: To evaluate the scalability to collision discretization, the computational performance under varying collision discretization levels is analyzed. In the "bin 025" scenario, five (Coarse), seven (Nominal), and 11 (Fine) spheres are employed to envelop the robotic arm. As shown in Table VI, all discretizations achieve success rates of at least 90.0%. Computation times exhibit only a marginal

TABLE VI
PLANNING PERFORMANCE ACROSS COLLISION DISCRETIZATIONS

Discretization	Success rate (%)	$CI_{95\%}$ (%)	Stage time (ms)			Total time (ms)
			t_g	t_s	t_o	
Coarse	90.0	[76.9, 96.0]	2.2	14.6	76.5	98.3 [74.2, 138.7]
Nominal	95.0	[83.5, 98.6]	2.1	23.3	90.1	143.5 [105.6, 187.9]
Fine	95.0	[83.5, 98.6]	2.1	27.4	105.5	148.5 [117.0, 277.4]

increase as the sphere count increases. This efficiency is supported by $O(1)$ distance queries from the ESDF map. Besides, the Fine configuration tightly envelops the physical thickness of the robotic arm, providing the highest geometric fidelity. Notably, even with this model, the median total computation time is only 148.5 ms, demonstrating the applicability of the method for handling fine discretization. The Coarse discretization is more likely to be rejected by the postoptimization feasibility check because of its lower geometric fidelity.

4) Scalability to Arm DoFs: Additional simulations with 4-DoF and 5-DoF arms evaluate DoF scalability in the "bin 025" scenario. To avoid discretization bias, the total arm length, collision model, and enclosing spheres remain fixed. As shown

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE VII
PLANNING PERFORMANCE ACROSS ROBOTIC ARM DOFS

Arm DoFs	Success rate (%)	$CI_{95\%}$ (%)	Stage time (ms)			Total time (ms)
			t_g	t_s	t_o	
3-DoF	95.0	[83.5, 98.6]	2.1	23.3	90.1	143.5 [105.6, 187.9]
4-DoF	95.0	[83.5, 98.6]	2.5	71.7	93.0	180.3 [136.6, 223.7]
5-DoF	92.5	[80.1, 97.4]	4.1	109.6	122.3	229.5 [175.1, 349.0]

Note: The 3-DoF row duplicates the Nominal row in Table VI.

TABLE VIII
PLANNING PERFORMANCE IN LARGE CONSTRAINED SCENARIOS

No. of obstacles	Success rate (%)	$CI_{95\%}$ (%)	Stage time (ms)			Total time (ms)
			t_g	t_s	t_o	
25	97.5	[87.1, 99.6]	1.1	447.2	325.0	744.9 [559.0, 1565.6]
30	87.5	[73.9, 94.5]	1.1	718.1	146.0	832.9 [392.3, 2089.7]

in Table VII, the 3-DoF, 4-DoF, and 5-DoF settings maintain high success rates. Planning time increases with arm DoF but remains below 300 ms for the 5-DoF case, demonstrating computational efficiency. Since the multirotor provides translational DoFs, a 3-DoF arm is sufficient for this task. The 5-DoF arm increases the optimization difficulty due to the higher-dimensional configuration space.

5) Constrained Flight Paths: Two $10\ \mathrm{m} \times 10\ \mathrm{m}$ flight-path scenarios containing 25 and 30 randomly placed cylindrical obstacles are evaluated. As shown in Table VIII, the success rates are 97.5% and 87.5%, with median total computation times of 744.9 ms and 832.9 ms, respectively. These results demonstrate high success rates with sub-second computation in constrained flight paths.

D. Real-World Experiments

1) Repeated Trials: We conduct repeated trials under the same start state and goal EE pose, with each trial involving an independent planning-and-execution process. Table IX reports the tracking RMSEs and measured clearances of successful trials. For each trial, measured clearance is defined as the minimum distance from a collision-sphere surface to the nearest obstacle over all trajectory samples and all collision spheres. Fig. 10 compares planned and measured trajectories and clearance curves for representative executions selected by the median measured clearance. It also shows the EE tracking-error curve and reports its value at the minimum measured-clearance instant. The independently generated trajectories are more similar in bin than in shelf, consistent with the smaller tracking RMSE standard deviations in Table IX. Fig. 11 shows the unsuccessful bin trial, where the EE contacted the bin edge during insertion. The supplementary video, which is available at this link[3], shows complete bin and shelf executions, including the unsuccessful bin trial.

Remark 3: Although the measured clearances in Table IX are smaller than the preset clearance thresholds $(\lambda_r - 1)r$, these thresholds only specify the activation region of the soft collision-avoidance cost rather than hard constraints.

[3]https://youtu.be/sBdlaGuF9AE

TABLE IX
QUANTITATIVE RESULTS IN REAL-WORLD EXPERIMENTS

Type	Metric	Bin	Shelf
Outcome	trials	6	5
	success rate	83.3%	100%
Tracking RMSE	multirotor position (m)	0.112 ± 0.017	0.098 ± 0.012
	yaw (rad)	0.075 ± 0.005	0.127 ± 0.019
	joint 1 (rad)	0.252 ± 0.006	0.127 ± 0.058
	joint 2 (rad)	0.075 ± 0.010	0.078 ± 0.051
	joint 3 (rad)	0.026 ± 0.003	0.005 ± 0.004
	end-effector position (m)	0.159 ± 0.003	0.084 ± 0.048
Clearance	measured clearance (m)	0.119 ± 0.027	0.071 ± 0.028

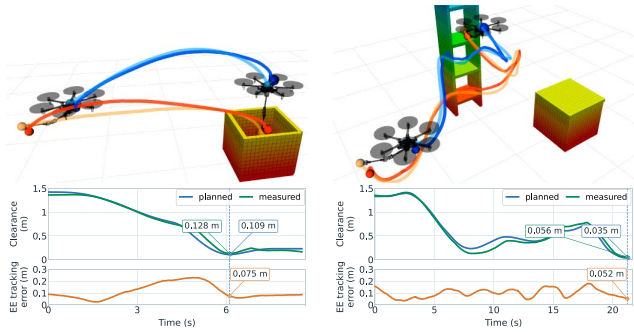


Fig. 10. Planned and measured trajectories, whole-body clearance, and EE tracking error for representative executions.

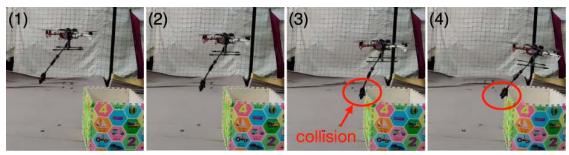


Fig. 11. End-effector collision with the bin edge during insertion.

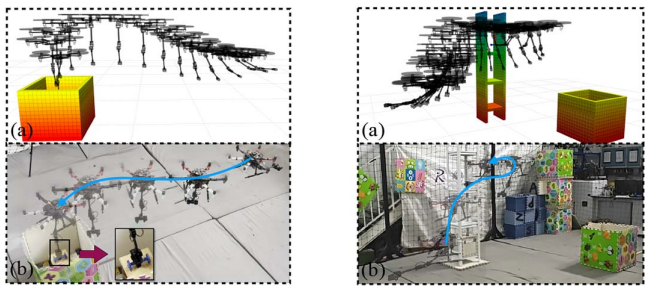


Fig. 12. Object-picking demonstrations in a bin (left) and a shelf (right). For each scenario, (a) shows the generated trajectory, and (b) shows experimental snapshots.

2) Demonstration Experiments: Two object-picking tasks are demonstrated using a bin and a shelf. The generated trajectories and corresponding experimental snapshots are shown in Fig. 12. For the bin task, the aerial manipulator first moves upward to avoid potential collisions with the bin edge and then enters the bin. For the shelf task, the proposed framework generates a

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

nominal trajectory for EE insertion. The total computation times are 345.0 and 421.2 ms for the bin and shelf tasks, respectively.

VIII. Conclusion

For aerial manipulation in constrained environments, a motion planning framework is proposed to generate a nominal trajectory that is dynamically consistent and passes the collision and feasibility checks. The framework comprises GSG, path searching, and trajectory optimization. For the redundant system, GSG selects an optimal goal state by minimizing the defined collision-avoidance margin and biases sampling to improve path-searching efficiency. The collision model is adjusted in the path searching stage to account for the neglected tilt motion, and Bi-RRT* is employed to generate a path connecting the start state and the goal state. In the trajectory optimization stage, the trajectories of the flat outputs are planned, and both the collision-avoidance and the feasibility constraints are incorporated into the optimization problem, which is solved using a gradient-based optimization method. Simulation and experimental results demonstrate that the proposed framework can generate nominal trajectories that pass offline geometric collision checks, with high empirical planning success rates and low computation times in the evaluated scenarios. The hardware experiments further demonstrate successful task execution in the tested scenarios.

While the current framework ensures dynamic consistency, it provides practical collision-aware planning without formal safety guarantees under perception uncertainty, model mismatch, or external disturbances. Future work will investigate robust planning for model mismatch and safety-critical control to provide formal safety guarantees under tracking errors. Furthermore, we will extend the framework to dynamic environments and integrate task and motion planning (TAMP) to address infeasible target poses, recover from execution failures, and handle complex tasks in real-world deployments.

References

[1] W. Zhan et al., "Resilient dynamic coverage control for networked autonomous aerial vehicles under deception attacks," *IEEE/ASME Trans. Mechatronics*, vol. 31, no. 1, pp. 128–139, Feb. 2026.

[2] S. Yang and B. Xian, "Energy-based nonlinear adaptive control design for the quadrotor UAV system with a suspended payload," *IEEE Trans. Ind. Electron.*, vol. 67, no. 3, pp. 2054–2064, Mar. 2020.

[3] G. Zhang, Y. He, B. Dai, F. Gu, J. Han, and G. Liu, "Robust control of an aerial manipulator based on a variable inertia parameters model," *IEEE Trans. Ind. Electron.*, vol. 67, no. 11, pp. 9515–9525, Nov. 2020.

[4] X. Liang, Y. Wang, H. YuZ. Zhang, J. Han, and Y. Fang, "Observer-based nonlinear control for dual-arm aerial manipulator systems suffering from uncertain center of mass," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 1984–1995, 2025.

[5] Z. Li, B. Xiu, Z. LiX. Yu, and J. J. Rodríguez-Andina, "Nonlinear characteristic model-based discrete-time sliding mode control for the hovering flight of an aerial manipulator," *IEEE Trans. Ind. Electron.*, vol. 72, no. 11, pp. 11769–11777, Nov. 2025.

[6] S. Kim, S. Choi, and H. J. Kim, "Aerial manipulation using a quadrotor with a two DOF robotic arm," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS).* Piscataway, NJ, USA: IEEE Press, 2013, pp. 4990–4995.

[7] A. González-Morgado, E. Cuniato, M. TognonG. Heredia, R. Siegwart, and A. Ollero, "Controlled shaking of trees with an aerial manipulator," *IEEE/ASME Trans. Mechatronics*, vol. 30, no. 2, pp. 1026–1037, Apr. 2025.

[8] K. Bodie, M. Brunner, and M. Pantic, "Active interaction force control for contact-based inspection with a fully actuated aerial vehicle," *IEEE Trans. Robot.*, vol. 37, no. 3, pp. 709–722, Jun. 2021.

[9] H. Cao et al., "Proximal cooperative aerial manipulation with vertically stacked drones," *Nature*, vol. 646, no. 8058, pp. 576–583, Oct. 2025.

[10] V. Pilania and K. Gupta, "A hierarchical and adaptive mobile manipulator planner with base pose uncertainty," *Auton. Robots*, vol. 39, no. 1, pp. 65–85, Jun. 2015.

[11] S. Thakar, P. Rajendran, H. Kim, A. M. Kabir, and S. K. Gupta, "Accelerating bi-directional sampling-based search for motion planning of non-holonomic mobile manipulators," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Piscataway, NJ, USA: IEEE Press, 2020, pp. 6711–6717.

[12] C. Tang, P. Miller, V. Krovi, J.-C. Ryu, and S. K. Agrawal, "Differential-flatness-based planning and control of a wheeled mobile manipulator—Theory and experiment," *IEEE/ASME Trans. Mechatronics*, vol. 16, no. 4, pp. 768–773, Aug. 2011.

[13] 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 *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2024, pp. 1369–1375.

[14] C. D. Bellicoso, K. Krämer, M. Stäuble, "Alma-articulated locomotion and manipulation for a torque-controllable robot," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA),* Piscataway, NJ, USA: IEEE Press, 2019, pp. 8477–8483.

[15] J.-P. Sleiman, F. Farshidian, M. V. Minniti, and M. Hutter, "A unified MPC framework for whole-body dynamic locomotion and manipulation," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4688–4695, Jul. 2021.

[16] A. Ollero, M. Tognon, A. Suarez, D. Lee, and A. Franchi, "Past, present, and future of aerial robotic manipulators," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 626–645, Feb. 2022.

[17] G. Zhang et al., "Grasp a moving target from the air: System & control of an aerial manipulator," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Piscataway, NJ, USA: IEEE Press, 2018, pp. 1681–1687.

[18] M. Wang et al., "Millimeter-level pick and peg-in-hole task achieved by aerial manipulator," *IEEE Trans. Robot.*, vol. 40, pp. 1242–1260, 2024.

[19] H. Cao, J. Shen, C. Liu, B. Zhu, and S. Zhao, "Motion planning for aerial pick-and-place with geometric feasibility constraints," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 2577–2594, 2025.

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

[21] J. Welde and V. Kumar, "Coordinate-free dynamics and differential flatness of a class of 6DOF aerial manipulators," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Piscataway, NJ, USA: IEEE Press, 2020, pp. 4307–4313.

[22] J. Welde, J. Paulos, and V. Kumar, "Dynamically feasible task space planning for underactuated aerial manipulators," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3232–3239, Apr. 2021.

[23] G. He et al., "Flying hand: End-effector-centric framework for versatile aerial manipulation teleoperation and policy learning," 2025, *arXiv: 2504.10334.*

[24] M. Zucker et al., "CHOMP: Covariant Hamiltonian optimization for motion planning," *Int. J. Robot. Res.*, vol. 32, nos. 9–10, pp. 1164–1193, 2013.

[25] J. Schulman et al., "Finding locally optimal, collision-free trajectories with sequential convex optimization," *Robot.: Sci. Syst.*, vol. 9, no. 1, pp. 1–10, 2013.

[26] M. Chen and C. J. Tomlin, "Hamilton–Jacobi reachability: Some recent theoretical advances and applications in unmanned airspace management," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 1, no. 1, pp. 333–358, 2018.

[27] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, Aug. 2017.

[28] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-Jacobi reachability: A brief overview and recent advances," in *Proc. IEEE Annu. Conf. Decis. Control (CDC),* Piscataway, NJ, USA: IEEE Press, 2017, pp. 2242–2253.

[29] J. Dong, S. Wu, R. Liu, X. Liang, B. Lu, and Y. Fang, "Robust safety-critical control under multiple state and input constraints: Volume control barrier function method," *IEEE Trans. Ind. Electron.*, vol. 73, no. 2, pp. 2725–2736, Feb. 2026.

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[30] A. Caballero, M. Béjar, A. Rodriguez-Castaño, and A. Ollero, "Motion planning for long reach manipulation in aerial robotic systems with two arms," in *Proc. Eur. Conf. Mobile Robots (ECMR)*, Piscataway, NJ, USA: IEEE Press, 2017, pp. 1–7.
[31] A. Caballero et al., "First experimental results on motion planning for transportation in aerial long-reach manipulators with two arms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Piscataway, NJ, USA: IEEE Press, 2018, pp. 8471–8477.
[32] H. Seo, S. Kim, and H. J. Kim, "Locally optimal trajectory planning for aerial manipulation in constrained environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Piscataway, NJ, USA: IEEE Press, 2017, pp. 1719–1724.
[33] H. Kim, H. Seo, and J. Kim, "Sampling-based motion planning for aerial pick-and-place," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Piscataway, NJ, USA: IEEE Press, 2019, pp. 7402–7408.
[34] H. Lee, H. Kim, and H. J. Kim, "Path planning and control of multiple aerial manipulators for a cooperative transportation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Piscataway, NJ, USA: IEEE Press, 2015, pp. 2386–2391.
[35] W. Deng, H. Chen, B. Ye, H. Chen, Z. Li, and X. Lyu, "Whole-body integrated motion planning for aerial manipulators," *IEEE Trans. Robot.*, vol. 41, pp. 6661–6679, 2025.
[36] Z. Zhang et al., "An end-effector-oriented coupled motion planning method for aerial manipulators in constrained environments," *IEEE/ASME Trans. Mechatronics*, vol. 30, no. 6, pp. 6027–6037, Dec. 2025.
[37] Z. Zhang, S. Wu, and C. Guo, "Ringo: Real-time navigation with a guiding trajectory for aerial manipulators in unknown environments," 2025, *arXiv: 2504.08338.*
[38] D. Lee, B. Kim, and H. J. Kim, "Autonomous aerial manipulation at arbitrary pose in SE(3) with robust control and whole-body planning," *Int. J. Robot. Res.*, vol. 45, no. 7, pp. 1021–1045, 2026.
[39] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *Int. J. Robot. Res.*, vol. 30, no. 12, pp. 1435–1460, 2011.
[40] A. D. Dragan, N. D. Ratliff, and S. S. Srinivasa, "Manipulation planning with goal sets using constrained trajectory optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 4582–4588.
[41] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Robot placement based on reachability inversion," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 1970–1975.
[42] Z. Han et al., "An efficient spatial-temporal trajectory planner for autonomous vehicles in unstructured environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 2, pp. 1797–1814, Feb. 2024.
[43] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Math. Program.*, vol. 45, no. 1, pp. 503–528, 1989.
[44] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast incremental Euclidean distance fields for online motion planning of aerial robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Piscataway, NJ, USA: IEEE Press, 2019, pp. 4423–4430.



Zhaopeng Zhang (Graduate Student Member, IEEE) received the B.S. degree in automation in 2021 from Nankai University, Tianjin, China, where he is currently working toward the Ph.D. degree in control science and engineering with the Institute of Robotics and Automation Information System.

His research interests include motion planning and task planning of aerial manipulators.



Hai Yu (Member, IEEE) received the B.S. degree in automation from Jilin University, Changchun, China, in 2020, and the Ph.D. degree in control theory and control engineering from Nankai University, Tianjin, China, in 2026.

He is currently a Postdoctoral Fellow with the Institute of Intelligence Technology and Robotic Systems, Shenzhen Research Institute of Nankai University, Shenzhen, China. His research interests include nonlinear control and motion planning of robotics.



Wei He received the B.S. degree in intelligent science and technology in 2019 from Nankai University, Tianjin, China, where he is currently working toward the Ph.D. degree in control science and engineering with the Institute of Robotics and Automatic Information System.

His research interests include motion planning, state estimation and intelligent control of robots, and especially autonomous aerial vehicles and aerial manipulation systems.



Yongchun Fang (Senior Member, IEEE) received the B.S. and M.S. degrees in control theory and applications from Zhejiang University, Hangzhou, China, in 1996 and 1999, respectively, and the Ph.D. degree in electrical engineering from Clemson University, Clemson, SC, USA, in 2002.

From 2002 to 2003, he was a Postdoctoral Fellow with Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY, USA. He is currently a Professor with the Institute of Robotics and Automatic Information System, Nankai University, Tianjin, China. His research interests include nonlinear control, visual servo control of underactuated systems, and atomic force microscopy (AFM)-based nano-systems.

Prof. Fang received the National Science Fund for Distinguished Young Scholars of China.



Jianda Han (Member, IEEE) received the Ph.D. degree in mechatronic engineering from Harbin Institute of Technology, Harbin, China, in 1998.

From 1998 to 2003, he was a Visiting Scientist with the City University of Hong Kong, Hong Kong, China, Michigan State University, East Lansing, MI, USA, and Cornell University, Ithaca, NY, USA. He is currently a Professor with the Institute of Robotics and Automatic Information System, Nankai University, Tianjin, China. His research interests include nonlinear estimation and control, robotics, and mechatronics systems.

Prof. Han is one of China's "Ten Thousand Talents Program" Technology Innovation Leading Talents.



Xiao Liang (Senior Member, IEEE) received the B.S. degree in intelligence science and technology from Hebei University of Technology, Tianjin, China, in 2013, and the Ph.D. degree in control theory and control engineering from Nankai University, Tianjin, in 2018.

He is currently a Professor with the Institute of Robotics and Automatic Information System, Nankai University. His research interests include motion planning and nonlinear control of unmanned aerial vehicle systems.

Authorized licensed use limited to: NANKAI UNIVERSITY. Downloaded on November 05,2026 at 01:36:16 UTC from IEEE Xplore. Restrictions apply.