



Search-based hierarchical whole-body motion planning and control for quadrotors

Peiyu Cui¹ , Hao Zhang¹ , Zhixu Du², Chao Huang¹, Jian Sun³

Keywords:

Motion and path planning, trajectory optimization, autonomous navigation

Citation: Cui, P.; Zhang, H.; Du, Z.; Huang, C.; Sun, J. Search-based hierarchical whole-body motion planning and control for quadrotors. *Intell. Robot.* 2026, 6(3), 504-23. <https://dx.doi.org/10.20517/ir.2026.24>

Received: 5 May 2026

First Decision: 4 Jun 2026

Revised: 17 Jul 2026

Accepted: 10 Aug 2026

Published: 19 Aug 2026

Academic Editor:

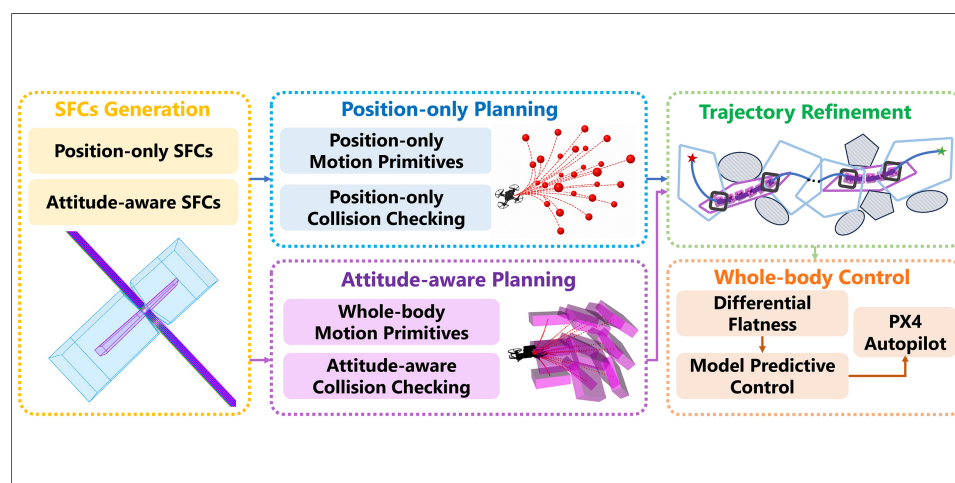
Chaomin Luo

Copy Editor:

Pei-Yun Wang

Production Editor:

Pei-Yun Wang



Abstract

In this paper, we propose a novel search-based hierarchical whole-body motion planning framework that can divide the planning process of a quadrotor into two parts: position-only planning and attitude-aware planning. A safe flight corridor (SFC) containing spatial scale information is designed to partition collision-free regions to identify narrow regions in the environment, and the attitude-aware planning process is triggered only in narrow regions. Based on the spatial-scale SFC, a carefully designed hierarchical collision detection mechanism can constrain the attitude of the quadrotor along the trajectory to the safe corridor to ensure collision-free and accelerate the planning process. Afterward, a trajectory refinement strategy is introduced to preserve the searched safe attitude-aware trajectory segments and optimize the neighboring position-only trajectory segments with position, velocity, and acceleration continuity, ensuring smooth transitions between different planning stages. Subsequently, a differentially flat model predictive controller is used to track aggressive trajectories. Benchmark results show that the proposed method generates safe and aggressive trajectories while reducing computation time by approximately 4-9 times relative to the search-based whole-body baseline. High-fidelity simulation experiments are conducted in various unknown environments to demonstrate the feasibility of the generated trajectories.



¹College of Electronic and Information Engineering, Tongji University, Shanghai 201210, China.

²Shanghai Research Institute for Intelligent Autonomous Systems, Tongji University, Shanghai 201210, China.

³College of Transportation Engineering, Tongji University, Shanghai 201210, China.

Correspondence to: Prof. Hao Zhang, College of Electronic and Information Engineering, Tongji University, Shanghai 201210, China. E-mail: zhang_hao@tongji.edu.cn

1. INTRODUCTION

Quadrotors have attracted much attention in the field of aircraft due to their flexible maneuverability and diverse application scenarios. Traditional quadrotor flight motion planning achieves obstacle avoidance by modeling the quadrotor as a sphere or simply inflating the obstacles. The resulting trajectory is relatively conservative and is mainly used for low-speed, stable flight missions^[1-5]. However, when performing tasks such as high-maneuverability flight, stunts, or dealing with dynamic obstacles in complex environments, quadrotors need to adjust their attitude to pass through narrow gaps that are much smaller than themselves, as shown in Figure 1. In such scenarios, motion planning must be carried out in the full configuration space, where both position and orientation are considered, which significantly increases planning complexity.

Whole-body motion planning is crucial for applications such as quadrotor navigation^[6], robotic manipulation^[7], and vehicle control. Unlike simpler motion planning in Euclidean space, where only point-mass objects are considered^[8], whole-body motion planning involves additional complexity because rotational motion must also be taken into account. Despite recent progress, whole-body motion planning for quadrotors still faces three main challenges.

(1) **Efficient extraction and characterization of feasible regions.** Whole-body motion planning relies on environmental perception and free-space representation to identify regions where the quadrotor attitude must be explicitly considered. Some methods use onboard cameras to detect narrow regions^[9], but they typically rely on specific gap geometries or visual features. Other methods require external motion-capture systems to obtain prior position and orientation information of narrow gaps^[6,10], limiting their applicability in unstructured environments. Another line of work models obstacles or feasible regions using convex polyhedra^[11,12]. However, high-quality convex decomposition can be computationally expensive. Voronoi-based methods^[13] can also partition the feasible space and provide safe paths with large clearance, but they are often conservative. Safe flight corridors (SFCs)^[14-16] have been widely adopted for quadrotor trajectory generation because they provide compact convex constraints for optimization. Nevertheless, conventional SFCs usually describe only the geometric boundary of the collision-free space and lack explicit spatial-scale information, making it difficult to distinguish spacious regions suitable for position-only planning from narrow regions that require attitude-aware planning.

(2) **Real-time whole-body motion planning.** Existing whole-body motion planning methods can be broadly classified into search-based methods^[9,17], learning-based methods^[18,19], and optimization-based methods^[6,10,12,20]. Search-based methods generate motion primitives by sampling the control space to find feasible trajectories. Liu *et al.* modeled the quadrotor as an ellipsoid and detected collisions by testing whether obstacle points lie inside it^[17]. Although finer sampling can improve feasibility, the search may require several seconds, while sampling in jerk space to ensure acceleration continuity leads to the curse of dimensionality. Learning-based methods have shown promising performance in narrow-gap traversal. Wu *et al.* proposed a pixel-to-action policy using reinforcement learning^[18], while Zhang *et al.* developed a differentiable-simulation-based end-to-end framework for irregular gaps^[19]. However, such methods typically require extensive simulation training and careful reward design, and may face generalization challenges in unseen cluttered environments. Optimization-based methods typically use convex decomposition to generate SFCs at the front end and optimize trajectory smoothness, safety, and dynamic feasibility at the back end^[6,10,12,20]. However, imposing attitude constraints along the entire trajectory increases the computational burden. Therefore, a real-time model-based framework that exploits spatial-scale information and selectively activates attitude-aware planning in narrow regions remains necessary.

(3) **Real-time control with constraints.** Conventional quadrotor tracking controllers usually adopt a cascaded feedback structure for position and attitude tracking. However, dynamic constraints are typically not explicitly handled in the control layer, placing a heavy burden on the planning module^[21-23]. If the

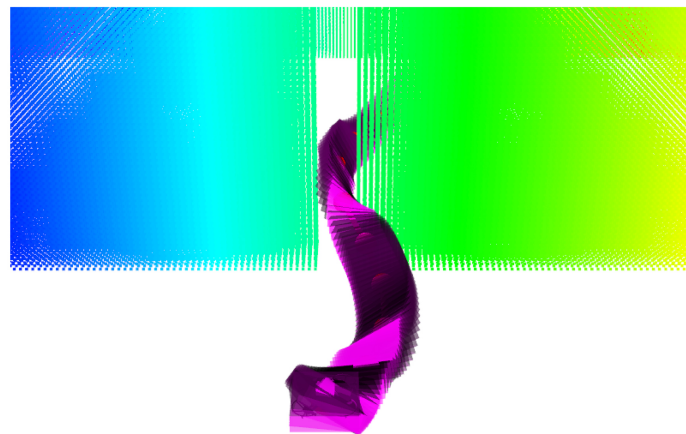


Figure 1. By finely dividing the environment into safe corridors with spatial scale information, our method can separate attitude-aware planning from position-only planning, which can greatly speed up the search efficiency compared with existing methods.

reference trajectory violates the dynamic limits of the quadrotor, the controller may suffer from large tracking errors or even tracking failure. Model predictive control (MPC) is a widely used strategy for explicitly handling dynamic constraints by predicting future states and optimizing control inputs online^[24]. However, quadrotor dynamics are inherently nonlinear, especially during aggressive flight with large roll and pitch angles. Directly applying nonlinear MPC to such systems requires solving a constrained nonlinear optimization problem at each control step, which can introduce considerable computational burden. When high-dimensional states, input limits, and fast control frequencies are considered simultaneously, the resulting nonlinear MPC problem may be difficult to solve in real time, thereby limiting its applicability as a high-frequency control module for aggressive quadrotor motion.

To address these challenges, we develop a search-based hierarchical whole-body motion planning and control framework for quadrotors. First, a spatial-scale-aware SFC strategy classifies the free space into spacious regions suitable for position-only planning and narrow regions requiring attitude-aware planning. Attitude-aware whole-body collision checking is selectively activated only in narrow regions, reducing unnecessary state expansion and improving search efficiency. Next, attitude-aware trajectory segments are retained, while the neighboring position-only segments are refined subject to position, velocity, and acceleration continuity. Finally, a differentially flat model predictive controller (DF-MPC) is used to track the resulting aggressive trajectory while explicitly enforcing dynamic constraints. The main contributions of this paper are summarized as follows:

- A spatial-scale-aware SFC generation method is proposed. Compared with widely used SFC generation methods^[14–16], the proposed method not only provides geometric containment constraints for trajectory generation but also characterizes the local spatial scale of the free space. This information enables the environment to be classified into spacious regions suitable for position-only planning and narrow regions requiring attitude-aware planning.
- A hierarchical collision-checking and trajectory-refinement mechanism is proposed. Using the spatial-scale information provided by the SFCs, attitude-aware whole-body collision checking is selectively activated in narrow regions, while efficient position-only collision checking is performed in spacious regions. The resulting attitude-aware trajectory segments are retained, whereas the neighboring position-only segments are refined to improve the higher-order continuity of the overall trajectory.

• A model predictive controller based on differential flatness is developed for whole-body trajectory tracking. The controller explicitly enforces dynamic constraints and maintains accurate tracking as the reference trajectory approaches the dynamic limits of the quadrotor. Its tracking performance is evaluated using the high-fidelity AirSim simulator.

The remainder of this paper is organized as follows. Section 2 introduces the differential flatness and collision-checking formulations. Section 3 presents the hierarchical whole-body motion planning and trajectory refinement methods. Section 4 develops the DF-MPC. Section 5 reports benchmark comparisons and high-fidelity simulation results. Finally, Section 6 concludes the paper.

2. PRELIMINARIES

2.1. Differential flatness and attitude reconstruction

The quadrotor has been shown to be a differentially flat system, whose state and control inputs can be represented by the flat output $\sigma = [x, y, z, \psi]^T$ and its higher-order derivatives. In whole-body planning, the key step is to recover the attitude of the quadrotor from the planned translational trajectory. Following the differential-flatness formulation in^[17], the rotation matrix of the quadrotor is denoted by

$$R = [\mathbf{x}_b(\sigma), \mathbf{y}_b(\sigma), \mathbf{z}_b(\sigma)] \in SO(3), \quad (1)$$

where $\mathbf{x}_b$, $\mathbf{y}_b$, and $\mathbf{z}_b$ are the body-frame axes expressed in the world frame. Let

$$a_x = \ddot{x}, \quad a_y = \ddot{y}, \quad a_z = \ddot{z} + g. \quad (2)$$

Then, the body z -axis is aligned with the desired total force direction:

$$\mathbf{z}_b = \frac{1}{\sqrt{a_x^2 + a_y^2 + a_z^2}} \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}. \quad (3)$$

The desired yaw angle ψ defines an intermediate heading direction:

$$\begin{aligned} \mathbf{x}_c &= [\cos \psi, \sin \psi, 0]^T, \\ \mathbf{y}_c &= [-\sin \psi, \cos \psi, 0]^T. \end{aligned} \quad (4)$$

According to $\mathbf{y}_b = \mathcal{N}(\mathbf{z}_b \times \mathbf{x}_c)$, the body y -axis can be explicitly written as

$$\mathbf{y}_b = \frac{1}{\sqrt{a_z^2 + (a_x \sin \psi - a_y \cos \psi)^2}} \begin{bmatrix} -a_z \sin \psi \\ a_z \cos \psi \\ a_x \sin \psi - a_y \cos \psi \end{bmatrix}. \quad (5)$$

Finally, from $\mathbf{x}_b = \mathbf{y}_b \times \mathbf{z}_b$, the body x -axis is obtained as

$$\mathbf{x}_b = \frac{1}{\sqrt{a_x^2 + a_y^2 + a_z^2} \sqrt{a_z^2 + (a_x \sin \psi - a_y \cos \psi)^2}} \begin{bmatrix} a_z^2 \cos \psi + a_y^2 \cos \psi - a_x a_y \sin \psi \\ a_z^2 \sin \psi + a_x^2 \sin \psi - a_x a_y \cos \psi \\ -a_z (a_x \cos \psi + a_y \sin \psi) \end{bmatrix}. \quad (6)$$

If the collective thrust f and the body angular velocity ω are used as control inputs, they can be computed from the flat outputs as

$$\begin{aligned} f &= m\sqrt{a_x^2 + a_y^2 + a_z^2}, \\ \omega &= (R^T \dot{R})^\vee, \end{aligned} \quad (7)$$

where $(\cdot)^\vee: \mathfrak{so}(3) \rightarrow \mathbb{R}^3$ denotes the vee map. From Equations (3)–(7), we observed that the collective thrust is directly determined by the desired acceleration, while the attitude is algebraically determined by the desired acceleration and yaw angle. Moreover, the body angular velocity is related to the derivative of the attitude, and therefore to higher-order derivatives of the flat output, such as jerk. This property is useful for both attitude-aware whole-body planning and differential-flatness-based control.

2.2. Collision checking

SFCs decompose the collision-free space into an ordered sequence of convex polyhedra. Each SFC is represented in half-space form as

$$\mathcal{P}^{\mathcal{H}} = \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{A}\mathbf{x} \preceq \mathbf{b}\}, \quad (8)$$

where each row of $\mathbf{A}$ defines the normal vector of a supporting plane and the corresponding element of $\mathbf{b}$ defines its offset. Based on this representation, we introduce position-only and attitude-aware collision-checking conditions for various spatial regions.

2.2.1. Position-only collision checking

In regions with sufficient clearance, the quadrotor is approximated as a point mass, allowing collision checking to be performed using only its center position. Let the SFC associated with the i -th trajectory segment be

$$\mathcal{P}_i^{\mathcal{H}} = \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{A}_i \mathbf{x} \preceq \mathbf{b}_i\}, \quad (9)$$

where $\mathbf{A}_i \in \mathbb{R}^{m_i \times 3}$, $\mathbf{b}_i \in \mathbb{R}^{m_i}$, and m_i is the number of planes of the i -th SFC. The position-only collision-free condition is then given by

$$\mathbf{A}_i \mathbf{p}(t) - \mathbf{b}_i \preceq \mathbf{0}, \quad t \in [0, T_i], \quad i = 0, 1, \dots, M-1, \quad (10)$$

where $\mathbf{p}(t)$ denotes the position of the quadrotor center and T_i denotes the duration of the corresponding trajectory segment. This condition ensures that the quadrotor center remains within its associated SFC throughout the segment.

To maintain additional clearance from the SFC boundary, we introduce a safety distance $d_{\text{safe}} > 0$ and strengthen the position-only collision-free condition as

$$\mathbf{A}_i \mathbf{p}(t) - \mathbf{b}_i + d_{\text{safe}} \mathbf{1}_{m_i} \preceq \mathbf{0}, \quad t \in [0, T_i], \quad (11)$$

where the rows of $\mathbf{A}_i$ are normalized and $\mathbf{1}_{m_i} \in \mathbb{R}^{m_i}$ is a vector of ones.

2.2.2. Attitude-aware collision checking

In narrow regions, a point-mass approximation is insufficient because the collision-free configuration of the quadrotor depends on its attitude. We therefore represent the quadrotor body as a convex polyhedron and

perform attitude-aware collision checking using all its vertices. Let $\mathbf{Q} \in \mathbb{R}^{3 \times N_v}$ collect the body-frame coordinates of the N_v vertices:

$$\mathbf{Q} = [\mathbf{q}_1 \quad \mathbf{q}_2 \quad \cdots \quad \mathbf{q}_{N_v}]. \quad (12)$$

The corresponding world-frame vertex coordinates at time t are

$$\mathbf{V}(t) = \mathbf{R}(t)\mathbf{Q} + \mathbf{p}(t)\mathbf{1}_{N_v}^\top, \quad (13)$$

where $\mathbf{R}(t) \in SO(3)$ is the rotation matrix recovered from the differential-flatness mapping, $\mathbf{p}(t)$ is the position of the quadrotor center, and $\mathbf{1}_{N_v} \in \mathbb{R}^{N_v}$ is a vector of ones.

Because both the quadrotor model and the SFC are convex, the entire quadrotor body lies within the i -th SFC if and only if all its vertices lie within that SFC. The attitude-aware collision-free condition can therefore be expressed as

$$\mathbf{A}_i \left(\mathbf{R}(t)\mathbf{Q} + \mathbf{p}(t)\mathbf{1}_{N_v}^\top \right) - \mathbf{b}_i\mathbf{1}_{N_v}^\top \preceq \mathbf{0}, \quad t \in [0, T_i], \quad i = 0, 1, \dots, M-1. \quad (14)$$

Unlike the position-only condition in Equation (10), the attitude-aware condition explicitly accounts for attitude. Because the rotation matrix is derived from the planned acceleration and yaw angle, this condition captures the effect of acceleration-induced attitude changes and is applied in narrow regions where attitude adjustment is essential for collision-free traversal.

3. WHOLE-BODY MOTION PLANNING

In this section, we first present a method for generating whole-body motion primitives based on the geometry of the quadrotor and clarify the conditions required to ensure collision-free whole-body motion within convex polyhedra. We then propose a spatial-scale-aware SFC classification method to distinguish spacious regions suitable for position-only planning from narrow regions requiring attitude-aware planning. Finally, we develop a trajectory-refinement strategy to ensure continuous attitude transitions across trajectory segments.

3.1. Whole-body motion primitives

According to the differential flatness of the quadrotor, the system state and control input can be represented by a set of appropriately selected flat outputs $\sigma = [x, y, z, \psi]^\top$ and their derivatives^[4]. Because the quadrotor yaw angle ψ is decoupled, it is set to follow the trajectory heading rather than being optimized as an independent planning variable. Therefore, the planning problem is formulated in the translational flat-output space, where the position $\mathbf{p}(t) = [p_x(t), p_y(t), p_z(t)]^\top$ and its higher-order derivatives are planned. The corresponding attitude can then be recovered from the planned acceleration and yaw angle through the differential-flatness mapping.

Consider an s -th-order integrator chain in the flat-output space. The state variable and control input are defined as

$$\begin{aligned} \mathbf{x}(t) &= [\mathbf{p}^\top(t), \dot{\mathbf{p}}^\top(t), \dots, \mathbf{p}^{(s-1)\top}(t)]^\top \in \mathcal{X} \subset \mathbb{R}^{3s}, \\ \mathbf{u}(t) &= \mathbf{p}^{(s)}(t) \in \mathcal{U} := [-u_{\max}, u_{\max}]^3. \end{aligned} \quad (15)$$

The corresponding state-space model is

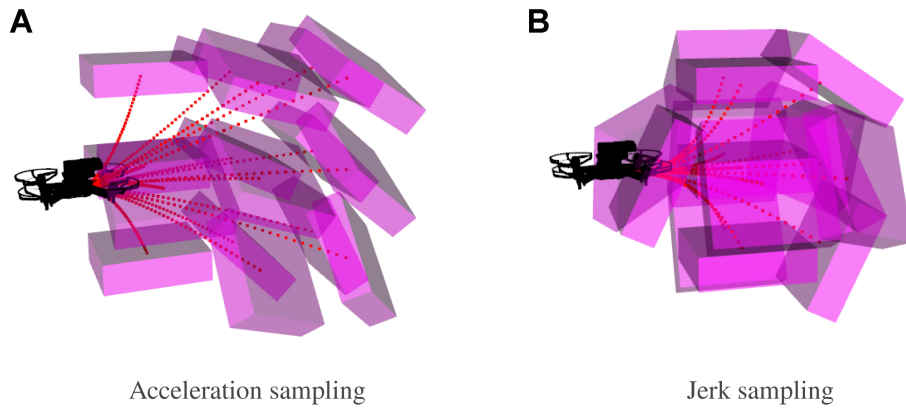


Figure 2. Examples of three-dimensional whole-body motion primitives starting from an initial state $\mathbf{x}_0$. (A and B) show the primitives generated by sampling the acceleration space ($s = 2$) and jerk space ($s = 3$), respectively. The purple cuboids represent the terminal whole-body states after applying a constant control input for $\tau = 0.2$ s. The initial velocity and acceleration are $\mathbf{v}_0 = [1, 0, 0]^T$ and $\mathbf{a}_0 = [1, 0, 0]^T$, respectively.

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u},$$

$$\mathbf{A} = \begin{bmatrix} 0 & \mathbf{I}_3 & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{I}_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 & \mathbf{I}_3 \\ 0 & \cdots & \cdots & 0 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \mathbf{I}_3 \end{bmatrix}. \quad (16)$$

Under a constant control input, each component of $\mathbf{p}(t)$ is represented by an s -th-order polynomial. By uniformly sampling the control space $\mathcal{U}$ at a resolution of $u_{\max}/r$, a total of $(2r + 1)^3$ motion primitives can be generated over a fixed duration τ ^[5].

To achieve whole-body obstacle avoidance, the geometry of the quadrotor must be considered. In^[17], the quadrotor is modeled as an ellipsoid. To accommodate a more general body shape, we model it as a convex polyhedron. In particular, we use a cuboid with half-lengths along the three body axes denoted by d_x , d_y , and d_z . Its body-frame occupied set is

$$\mathcal{Q}_{\text{cub}} = \{\mathbf{q} = [q_x, q_y, q_z]^T \in \mathbb{R}^3 \mid |q_x| \leq d_x, |q_y| \leq d_y, |q_z| \leq d_z\}. \quad (17)$$

Accordingly, its eight vertices are

$$\mathcal{V}_{\text{cub}} = \{\mathbf{q}_i = [\pm d_x \quad \pm d_y \quad \pm d_z]^T, \quad i = 1, 2, \dots, 8\}. \quad (18)$$

According to the differential-flatness formulation introduced in Section 2.1, the rotation matrix $\mathbf{R}(t)$ is determined by the acceleration $\ddot{\mathbf{p}}(t)$ and yaw angle $\psi(t)$, and can therefore be expressed as $\mathbf{R}(\psi(t), \ddot{\mathbf{p}}(t))$. Based on Equation (16) and the cuboid representation, the occupied set of a whole-body motion primitive in the world frame is defined as

$$\mathcal{E}(t) = \{\mathbf{R}(\psi(t), \ddot{\mathbf{p}}(t))\mathbf{q} + \mathbf{p}(t) \mid \mathbf{q} \in \mathcal{Q}_{\text{cub}}\}. \quad (19)$$

Figure 2 shows whole-body motion primitives generated over a fixed duration τ by sampling the acceleration and jerk spaces.

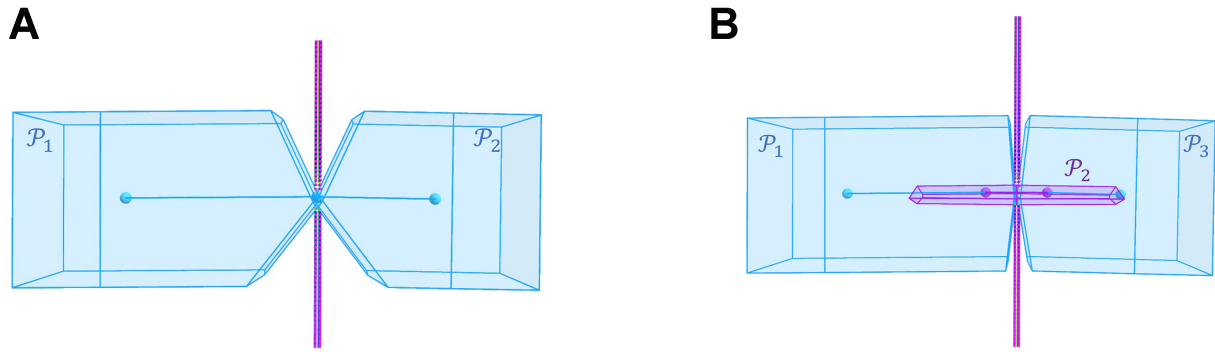


Figure 3. Comparison of corridor construction around a narrow passage. The blue points represent position-only seed endpoints, whereas the purple points represent attitude-aware seed endpoints. (A) The conventional method generates two polyhedra whose overlap is concentrated near the narrow opening, leaving insufficient space for attitude adjustment and corridor transition; (B) The proposed method inserts an Att-SFC $\mathcal{P}_2$ between the Pos-SFCs $\mathcal{P}_1$ and $\mathcal{P}_3$, forming an ordered and overlapping corridor sequence for attitude-aware traversal of the narrow passage. SFC: Safe flight corridor.

The expansion cost of a motion primitive is defined as

$$J(T) = \int_0^T \|\mathbf{u}(t)\|_2^2 dt + \rho T, \quad (20)$$

where ρ is the time-penalty coefficient, which encourages faster trajectories. For a constant control input $\mathbf{u}_d$ and a fixed sampling duration τ , the primitive cost becomes $J(\tau) = (\|\mathbf{u}_d\|_2^2 + \rho)\tau$. At the same control resolution and sampling duration, jerk-controlled search ensures acceleration continuity but incurs substantially higher computational and memory costs as the system order increases. Because the attitude is directly related to acceleration, the proposed method adopts an acceleration-controlled system to improve search efficiency. The resulting acceleration discontinuities at the junctions of motion primitives are addressed by the trajectory-refinement method presented in Section 3.2.3.

3.2. Generation of a collision-free trajectory

This subsection introduces the generation of a collision-free trajectory. SFCs with spatial-scale information are used to divide the free space into Pos-SFCs and Att-SFCs. Efficient position-only planning is performed in Pos-SFCs, whereas attitude-aware planning is activated in Att-SFCs to explicitly account for attitude. After the search stage, a trajectory-refinement strategy is introduced to ensure continuous transitions between position-only and attitude-aware trajectory segments.

3.2.1. Generation of position-only and attitude-aware SFCs

To improve the efficiency of search-based whole-body planning, we construct an ordered sequence of SFCs along a center-feasible guide path. By restricting the expansion of motion primitives to these corridors, the proposed method avoids exploring redundant regions, thereby reducing the number of expanded nodes and the overall search time. SFCs generated in spacious and narrow regions generally exhibit substantially different spatial scales. This observation motivates us to divide the generated corridors into Pos-SFCs and Att-SFCs, as illustrated in Figure 3.

The guide path is first generated by a front-end path-searching algorithm and is represented as $\Gamma = \{\mathbf{q}_0, \mathbf{q}_1, \dots, \mathbf{q}_{N_g-1}\}$, where $\mathbf{q}_0$ and $\mathbf{q}_{N_g-1}$ denote the initial and goal positions, respectively. At this stage, the guide path guarantees only collision-free traversal of the quadrotor center and does not explicitly account for its attitude. To identify guide-path intervals where attitude adjustment may be required, we introduce a bounding sphere that conservatively encloses the cuboid model. Its radius is defined as $r_s = \sqrt{d_x^2 + d_y^2 + d_z^2}$,

where d_x , d_y , and d_z are the half-lengths defined in Equation (17). Because the radius is independent of attitude, the bounding sphere provides an efficient and conservative measure of local clearance along the guide path.

For each guide point $\mathbf{q}_k$, a radius query is performed on the obstacle-point KD-tree. The corresponding sphere-collision indicator is defined as

$$\chi_k = \begin{cases} 1, & \min_{\mathbf{o} \in \mathcal{O}} \|\mathbf{o} - \mathbf{q}_k\|_2 \leq r_s, \\ 0, & \text{otherwise,} \end{cases} \quad (21)$$

where $\mathcal{O}$ denotes the obstacle-point set. A guide point satisfying $\chi_k = 0$ provides sufficient clearance for the cuboid bounding sphere and is therefore assigned to a position-only guide. In contrast, $\chi_k = 1$ indicates that the bounding sphere intersects the environment even though the quadrotor center remains collision-free. Such a point is assigned to an attitude-aware guide because the cuboid may need to adjust its orientation to traverse the corresponding region. The transition $\chi_{k-1} = 0, \chi_k = 1$ defines the entrance of an attitude-aware interval, whereas $\chi_{k-1} = 1, \chi_k = 0$ marks its end. Consecutive guide points satisfying $\chi_k = 1$ are grouped into the same attitude-aware interval, while the remaining portions of the guide path are assigned to position-only intervals. In this way, the original guide path is converted into an ordered sequence of position-only and attitude-aware portions.

The resulting typed guide path is divided into seed segments, which are used to generate SFCs. Each generated SFC is directly inherited from its corresponding guide seed: position-only seeds generate Pos-SFCs, whereas attitude-aware seeds generate Att-SFCs. This construction restricts the translational search space while identifying regions in which attitude-aware whole-body planning may be required.

Figure 3 compares conventional corridor construction with the proposed typed-seed construction. The blue points represent the endpoints of position-only seeds, whereas the purple points represent the endpoints of attitude-aware seeds. In the conventional method shown in **Figure 3A**, a uniform seed-generation strategy produces two large polyhedra on opposite sides of the narrow passage. Their overlap is concentrated near the opening and does not provide sufficient space for attitude adjustment and corridor transition. In the proposed method shown in **Figure 3B**, the bounding-sphere test identifies the attitude-critical portion of the guide path. The purple attitude-aware seed generates the intermediate Att-SFC $\mathcal{P}_2$, while the blue position-only seeds generate the neighboring Pos-SFCs $\mathcal{P}_1$ and $\mathcal{P}_3$. Consequently, $\mathcal{P}_2$ forms an attitude-aware bridge through the narrow passage and overlaps with the adjacent Pos-SFCs, providing an ordered corridor sequence for the subsequent whole-body search.

3.2.2. Hierarchical collision checking

After obtaining the ordered SFC sequence, the search is restricted to the corresponding corridor regions, thereby avoiding the expansion of redundant nodes. This property also allows the computationally expensive attitude-aware collision checking to be separated from efficient position-only collision checking.

Specifically, **PosCollisionCheck**($\cdot$) determines whether the sampled positions of a motion primitive remain within its associated Pos-SFC according to Equation (11). In contrast, **AttCollisionCheck**($\cdot$) transforms all cuboid vertices according to the attitude and verifies their containment within the associated Att-SFC using Equation (14). Finally, the accepted primitives retain their corridor types for the subsequent trajectory-refinement stage.

Whole-body path searching is built upon the kinodynamic A* algorithm for quadrotors^[3], as summarized in [Algorithm 1](#). The open and closed lists are denoted by O and C , respectively, and n_c denotes the currently expanded node. The function **ComputeShotPath**($\cdot$) computes a direct connection from the current node to the goal by analytically solving a linear quadratic minimum time (LQMT) problem^[25]. A shot trajectory is accepted only if it satisfies both the dynamic constraints and the hierarchical collision-checking conditions. Otherwise, candidate motion primitives are sampled and checked according to the type of their associated SFCs. Only collision-free successor nodes are inserted into or updated in the open list.

Algorithm 1 Whole-Body Path Searching

```

1: Initialize( $O, C$ )
2: GenerateSFC()
3: while  $\neg O.empty()$  do
4:    $n_c \leftarrow O.pop()$ 
5:    $C.insert(n_c)$ 
6:    $\pi_{shot} \leftarrow \text{ComputeShotPath}(n_c, \mathbf{x}_{goal})$ 
7:   if  $\pi_{shot}$  is feasible and HierarchicalCollisionCheck( $\pi_{shot}$ ) then
8:     return RetrievePath( $n_c, \pi_{shot}$ )
9:   end if
10:   $\Pi \leftarrow \text{Sampling}(n_c)$ 
11:  for each  $\pi_i \in \Pi$  do
12:    if  $\pi_i$  satisfies the dynamic constraints then
13:       $c_i \leftarrow \text{DetermineSFC}(\pi_i)$ 
14:      if  $c_i$  is a Pos-SFC then
15:         $valid \leftarrow \text{PosCollisionCheck}(\pi_i, c_i)$ 
16:      else
17:         $valid \leftarrow \text{AttCollisionCheck}(\pi_i, c_i)$ 
18:      end if
19:      if  $valid$  then
20:         $n_i \leftarrow \text{Propagate}(n_c, \pi_i)$ 
21:        ExpandNode( $n_i, O$ )
22:      end if
23:    end if
24:  end for
25: end while
26: return Failure

```

[Figure 4](#) illustrates the hierarchical whole-body motion-planning mechanism. The blue convex polyhedron denotes a Pos-SFC, in which a position-only trajectory is generated, whereas the purple convex polyhedron denotes an Att-SFC, in which an attitude-aware trajectory is generated.

3.2.3. Trajectory refinement

The acceleration-controlled search explicitly determines the duration of each motion primitive, thereby providing a natural initial time allocation for subsequent trajectory refinement. The durations of the searched trajectory segments are collected as

$$\mathbf{T} = [T_0, T_1, \dots, T_{M-1}]^\top \in \mathbb{R}_+^M, \quad (22a)$$

$$T_k = \begin{cases} \tau, & k = 0, 1, \dots, M-2, \\ t_{\text{opt}}, & k = M-1, \end{cases} \quad (22b)$$

where τ denotes the sampling duration of each motion primitive and t_{opt} is the duration of the terminal shot trajectory obtained by solving the LQMT problem.

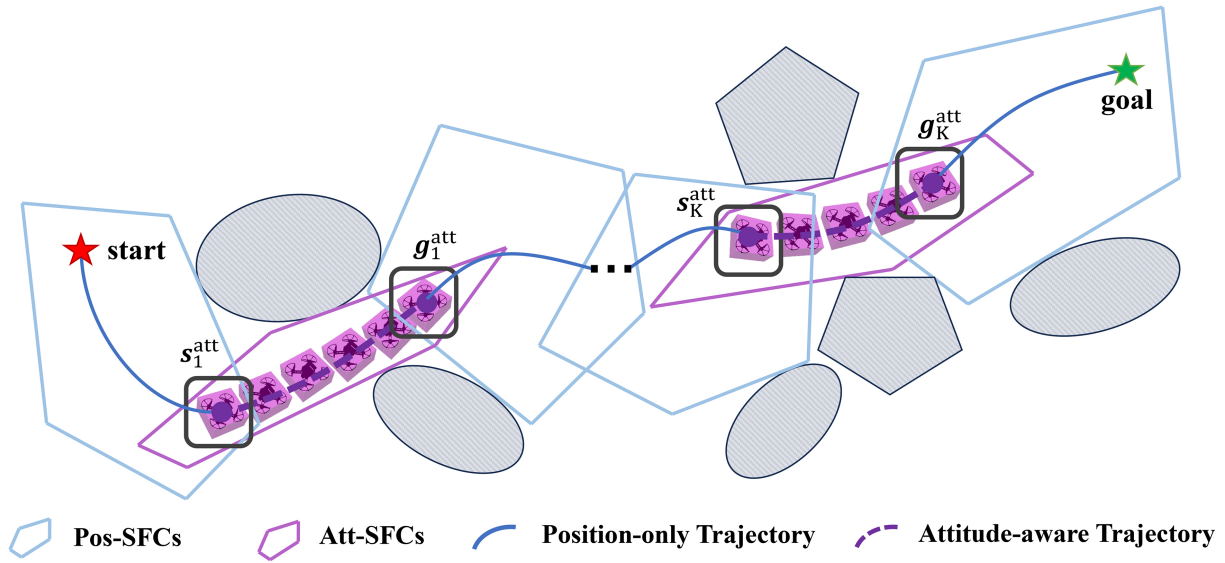


Figure 4. Illustration of the hierarchical whole-body motion-planning mechanism. The blue convex polyhedron denotes a Pos-SFC, in which position-only collision checking is performed. The purple convex polyhedron denotes an Att-SFC, in which attitude-aware collision checking is performed. SFC: Safe flight corridor.

After the hierarchical search, the resulting trajectory is divided into attitude-aware and position-only segments. Let $\mathcal{A}$ and $\mathcal{P}$ denote their index sets, respectively, satisfying

$$\mathcal{A} \cap \mathcal{P} = \emptyset, \quad \mathcal{A} \cup \mathcal{P} = \{0, 1, \dots, M-1\}. \quad (23)$$

For each $k \in \mathcal{A}$, let $\bar{\mathbf{p}}_k^{\text{att}}(t)$ denote the searched attitude-aware trajectory, and let $\mathbf{s}_k^{\text{att}}$ and $\mathbf{g}_k^{\text{att}}$ denote its initial and terminal states, respectively. Because these segments are generated within Att-SFCs and have already passed attitude-aware whole-body collision checking, they are retained unchanged during trajectory refinement. The segments indexed by $\mathcal{P}$ connect the global initial state to the first attitude-aware segment, consecutive attitude-aware segments to each other, and the last attitude-aware segment to the global goal state. These connecting segments lie within Pos-SFCs and are therefore treated as position-only trajectories. The paths obtained from the search stage are used to initialize the subsequent trajectory optimization.

Although acceleration-controlled search substantially improves computational efficiency, directly concatenating the searched motion primitives generally leads to acceleration discontinuities and, consequently, abrupt attitude changes. To improve trajectory continuity without compromising the verified safety of the attitude-aware segments, the proposed refinement strategy keeps all attitude-aware segments fixed and optimizes only the position-only connecting segments. Accordingly, each attitude-aware segment satisfies

$$\mathbf{p}_k(t) = \bar{\mathbf{p}}_k^{\text{att}}(t), \quad t \in [0, T_k], \quad k \in \mathcal{A}. \quad (24)$$

The position-only segments are refined by solving the following minimum-jerk problem:

$$\min_{\{\mathbf{p}_k(t)\}_{k \in \mathcal{P}}} \sum_{k \in \mathcal{P}} \int_0^{T_k} \|\mathbf{p}_k^{(3)}(t)\|_2^2 dt, \quad (25a)$$

$$\text{s.t.} \quad \mathbf{x}_0(0) = \mathbf{s}_{\text{start}}, \quad \mathbf{x}_{M-1}(T_{M-1}) = \mathbf{s}_{\text{goal}}, \quad (25b)$$

$$\mathbf{x}_k(T_k) = \mathbf{s}_j^{\text{att}}, \quad (k, j) \in \mathcal{E}_{\text{in}}, \quad (25\text{c})$$

$$\mathbf{x}_k(0) = \mathbf{g}_j^{\text{att}}, \quad (k, j) \in \mathcal{E}_{\text{out}}, \quad (25\text{d})$$

$$\mathbf{x}_{k+1}(0) = \mathbf{x}_k(T_k), \quad k, k+1 \in \mathcal{P}, \quad (25\text{e})$$

$$\mathbf{p}_k(t) \in \mathcal{P}_k^{\mathcal{H}}, \quad t \in [0, T_k], \quad k \in \mathcal{P}. \quad (25\text{f})$$

Here, $\mathcal{E}_{\text{in}}$ denotes the set of stitching relations from position-only segments to their subsequent attitude-aware segments, whereas $\mathcal{E}_{\text{out}}$ denotes the set of stitching relations from attitude-aware segments to their succeeding position-only segments. The state of the k -th trajectory segment is defined as

$$\mathbf{x}_k(t) = [\mathbf{p}_k^\top(t), \dot{\mathbf{p}}_k^\top(t), \ddot{\mathbf{p}}_k^\top(t)]^\top. \quad (26)$$

The constraints in Equations (25c) and (25d) match the boundary states of each position-only segment to those of the adjacent attitude-aware segment. Specifically, the terminal state of each incoming position-only segment is matched to the initial state of the succeeding attitude-aware segment, while the initial state of each outgoing position-only segment is matched to the terminal state of the preceding attitude-aware segment. The constraint in Equation (25e) imposes position, velocity, and acceleration continuity between adjacent position-only segments. Therefore, C^2 continuity is enforced at every junction involving an optimized position-only segment.

4. WHOLE-BODY TRAJECTORY TRACKING WITH A DF-MPC

As discussed in^[5], although trajectory refinement improves the smoothness of the searched trajectory, the refined reference may still violate the dynamic limits of the quadrotor. To track the desired whole-body trajectory while satisfying the prescribed state and control constraints, we employ a DF-MPC. The controller simultaneously reduces position and attitude-related tracking errors by tracking the desired position, velocity, and acceleration over a finite prediction horizon. Because the attitude is algebraically related to the translational acceleration through the differential-flatness mapping, acceleration tracking contributes directly to attitude tracking. The body angular velocity $\boldsymbol{\omega}$ is related to the jerk $\mathbf{j}$. Therefore, a third-order integrator with jerk input is adopted as the prediction model. The state and control input are defined as

$$\mathbf{x}_k = [\mathbf{p}_k^\top \quad \mathbf{v}_k^\top \quad \mathbf{a}_k^\top]^\top = [p_{x,k} \quad p_{y,k} \quad p_{z,k} \quad \dot{p}_{x,k} \quad \dot{p}_{y,k} \quad \dot{p}_{z,k} \quad \ddot{p}_{x,k} \quad \ddot{p}_{y,k} \quad \ddot{p}_{z,k}]^\top, \quad (27\text{a})$$

$$\mathbf{u}_k = \mathbf{j}_k = [\ddot{p}_{x,k} \quad \ddot{p}_{y,k} \quad \ddot{p}_{z,k}]^\top. \quad (27\text{b})$$

Let Δt denote the control interval. Under the zero-order-hold assumption for the jerk input, the discrete prediction model is given by

$$\mathbf{x}_{k+1} = \mathbf{A}_d \mathbf{x}_k + \mathbf{B}_d \mathbf{u}_k, \quad (28)$$

where

$$\mathbf{A}_d = \begin{bmatrix} \mathbf{I}_3 & \Delta t \mathbf{I}_3 & \frac{1}{2} \Delta t^2 \mathbf{I}_3 \\ \mathbf{0} & \mathbf{I}_3 & \Delta t \mathbf{I}_3 \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_3 \end{bmatrix}, \quad \mathbf{B}_d = \begin{bmatrix} \frac{1}{6} \Delta t^3 \mathbf{I}_3 \\ \frac{1}{2} \Delta t^2 \mathbf{I}_3 \\ \Delta t \mathbf{I}_3 \end{bmatrix}. \quad (29)$$

At each control step, DF-MPC predicts the system evolution over N steps and computes the optimal jerk sequence by minimizing a finite-horizon cost function. For trajectory tracking, the objective penalizes both the state-tracking error and the control effort:

$$\mathcal{F} = \sum_{k=0}^{N-1} \left(\|\mathbf{x}_k - \tilde{\mathbf{x}}_k\|_{\mathbf{Q}_x}^2 + \|\mathbf{u}_k\|_{\mathbf{Q}_u}^2 \right) + \|\mathbf{x}_N - \tilde{\mathbf{x}}_N\|_{\mathbf{Q}_f}^2, \quad (30)$$

where $\tilde{\mathbf{x}}_k = [\tilde{\mathbf{p}}_k^\top, \tilde{\mathbf{v}}_k^\top, \tilde{\mathbf{a}}_k^\top]^\top$ is the reference state, and $\mathbf{Q}_x \succeq \mathbf{0}$, $\mathbf{Q}_u \succ \mathbf{0}$, and $\mathbf{Q}_f \succeq \mathbf{0}$, are weighting matrices of appropriate dimensions. The resulting DF-MPC problem is formulated as

$$\begin{aligned} \min_{\{\mathbf{x}_k, \mathbf{u}_k\}} \quad & \sum_{k=0}^{N-1} \left(\|\mathbf{x}_k - \tilde{\mathbf{x}}_k\|_{\mathbf{Q}_x}^2 + \|\mathbf{u}_k\|_{\mathbf{Q}_u}^2 \right) + \|\mathbf{x}_N - \tilde{\mathbf{x}}_N\|_{\mathbf{Q}_f}^2 \\ \text{s.t.} \quad & \mathbf{x}_{k+1} = \mathbf{A}_d \mathbf{x}_k + \mathbf{B}_d \mathbf{u}_k, \quad k = 0, \dots, N-1, \\ & \mathbf{x}_{\min} \preceq \mathbf{x}_k \preceq \mathbf{x}_{\max}, \quad k = 0, \dots, N, \\ & \mathbf{u}_{\min} \preceq \mathbf{u}_k \preceq \mathbf{u}_{\max}, \quad k = 0, \dots, N-1. \end{aligned} \quad (31)$$

To obtain the low-level collective-thrust and body-angular-velocity commands, the optimized flat-output quantities are mapped to the physical control inputs. Define the desired mass-normalized collective thrust as

$$c_d = \mathbf{z}_b^\top (\mathbf{a} + g \mathbf{z}_w). \quad (32)$$

The corresponding collective thrust is $f_d = m c_d$, where m is the mass of the quadrotor. The desired body angular velocity is obtained as

$$\begin{aligned} \omega_x &= -\frac{\mathbf{y}_b^\top \mathbf{j}}{c_d}, \\ \omega_y &= \frac{\mathbf{x}_b^\top \mathbf{j}}{c_d}, \\ \omega_z &= \frac{c_d \dot{\psi} \mathbf{x}_c^\top \mathbf{x}_b + (\mathbf{y}_c^\top \mathbf{z}_b) (\mathbf{x}_b^\top \mathbf{j})}{c_d \|\mathbf{y}_c \times \mathbf{z}_b\|_2}. \end{aligned} \quad (33)$$

Here, $\mathbf{R} = [\mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b]$ is recovered using the differential-flatness mapping introduced in Section 2.1. The mapping assumes $c_d > 0$ and $\|\mathbf{y}_c \times \mathbf{z}_b\|_2 > 0$, thereby excluding zero-thrust and attitude-reconstruction singularities. By minimizing the predicted state error, DF-MPC tracks the desired position through $\mathbf{p}$ and the desired attitude indirectly through $\mathbf{a}$. Meanwhile, the jerk input determines the desired body angular velocity through Equation (33).

5. EXPERIMENTS

5.1. Numerical simulation

5.1.1. Ablation studies and parameter selection

We first conducted an ablation study to evaluate the contribution of the trajectory-refinement module. We then examined the effects of the main parameters involved in the motion-primitive search and selected their values through comparative experiments in narrow-passage scenarios.

Effect of trajectory refinement: We compared two configurations: (1) Search only, in which the trajectory generated by the acceleration-controlled whole-body search was used directly without subsequent refinement; and (2) Search + refinement, in which the attitude-aware segments were retained, while the position-only segments were refined using the formulation introduced in Section 3.2.3. Each searched motion primitive applies a constant acceleration input over its duration. Although consecutive primitives are

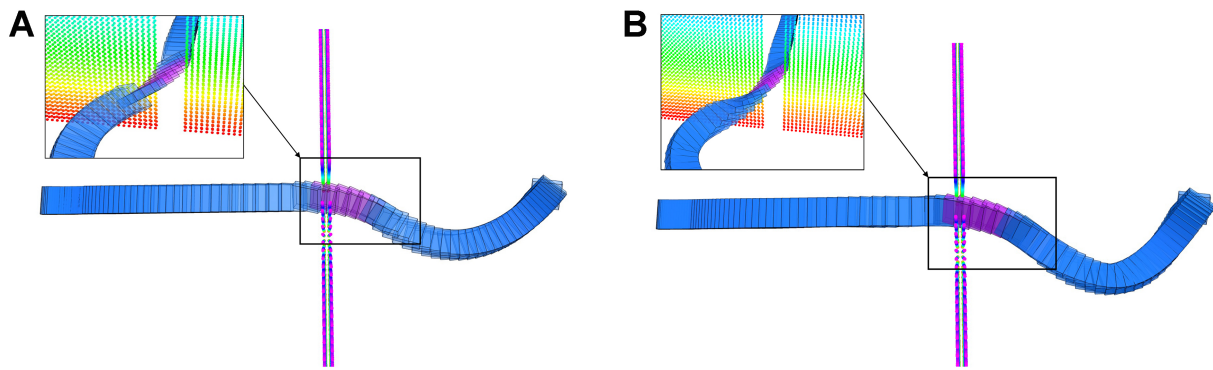


Figure 5. Comparison of the searched and refined trajectories. The purple trajectory denotes the attitude-aware trajectory, while the blue trajectory denotes the position-only trajectory. (A) Acceleration-controlled trajectory without refinement. Abrupt attitude changes occur at the junctions between the two trajectory types; (B) Refined trajectory with smoother transitions. Continuous attitude transitions at their junctions.

connected in position and velocity, their acceleration inputs are not necessarily identical at the junctions. Because the desired roll and pitch angles are recovered from the desired acceleration through differential flatness, these discontinuities can produce abrupt changes in the desired attitude. As shown in Figure 5A, without refinement, such abrupt acceleration-induced attitude changes can be observed at several primitive junctions. The refinement stage improves the higher-order continuity of the position-only portions while preserving the attitude-aware segments. The resulting trajectory therefore provides smoother attitude transitions while traversing narrow passages. As illustrated in Figure 5B, the refined position-only portions exhibit smoother attitude transitions.

Parameter selection. The performance of the motion-primitive search is mainly affected by the primitive cost weight ρ , the primitive duration τ , and the control-space sampling parameter r . The parameter ρ balances traversal time and control effort: a small value favors conservative but longer trajectories, whereas a large value favors faster but potentially more aggressive motions. The duration τ determines the spacing of the state lattice; a small τ increases search resolution and computational cost, while a large τ may produce a lattice that is too sparse to represent feasible transitions near narrow passages. Similarly, increasing r provides a finer control discretization but increases the branching factor as $(2r + 1)^3$, whereas a small r may omit the control inputs required for feasible attitude transitions. Based on comparative experiments in narrow-passage scenarios, we selected $\rho = 4u_{\max}^{-2}$, $\tau = 0.2$ s, and $r = 2$, where $u_{\max}$ denotes the maximum control magnitude along each axis. This setting samples five control inputs per axis and generates 125 candidate motion primitives in the three-dimensional control space, providing a practical compromise among trajectory aggressiveness, state-space coverage, and computational efficiency.

5.1.2. Performance evaluation in narrow-passage scenarios

To evaluate the capability of the proposed planner in geometrically constrained environments, we conducted two groups of narrow-passage experiments. The first group evaluates passages with different widths, whereas the second group evaluates rectangular windows with different orientations. In all experiments, the start and goal states were placed on opposite sides of the obstacle, while the dynamic limits and search parameters were kept unchanged.

Passages with different widths: The quadrotor radius was set to 0.35 m, corresponding to a horizontal span of 0.70 m. As shown in Figure 6, the 0.80 m gap is wider than the quadrotor and can therefore be traversed using a position-only trajectory without triggering attitude-aware planning. In contrast, the 0.65 and 0.50 m gaps are narrower than the horizontal span of the quadrotor, requiring progressively larger attitude

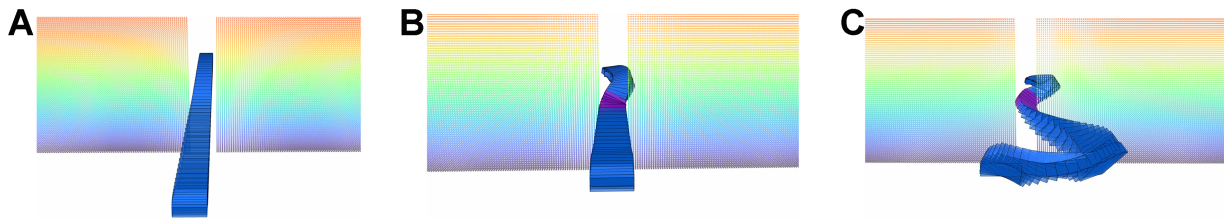


Figure 6. Hierarchical planning results through wall gaps of different widths. (A) 0.80 m gap, where the gap width is larger than the quadrotor's horizontal span and only position planning is required; (B) 0.65 m gap and (C) 0.50 m gap, where attitude-aware planning is activated.

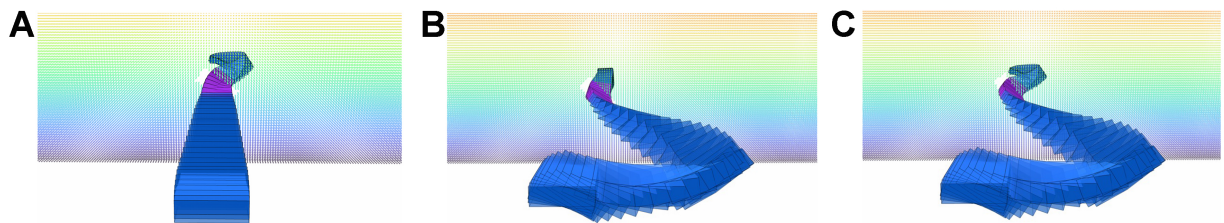


Figure 7. Attitude-aware trajectories through rectangular windows with different orientations. (A) 35°; (B) 45°; (C) 55°.

adjustments to reduce its projected body extent. The results demonstrate that the planner activates attitude-aware motion only when required by the available passage size.

Windows with different orientations: We further evaluated the planner using a narrow rectangular window with fixed dimensions and orientations of 35°, 45°, and 55°. As shown in Figure 7, the planner generates attitude-aware motions that align the projected body shape with the available free space, with larger window inclinations requiring larger roll adjustments.

5.2. Benchmark for whole-body motion planning

In this section, we compare the proposed method with a search-based method^[17], denoted as Base-A, and an optimization-based method, denoted as Base-B. Base-B was evaluated using the publicly available CPU implementation released with Fast-Racing^[10]. Fast-Racing and the method of Yang *et al.*^[20] share the GCOPTER-based polynomial optimization framework^[6] and employ closely related whole-body collision constraints. We therefore used Fast-Racing as the open-source implementation for evaluating Base-B. We test the efficiency and quality of trajectories generated by the proposed method in two different simulated environments: (1) Random forest: 40 m × 20 m × 3 m map containing 150 random obstacles. (2) Office: 60 m × 45 m × 3 m indoor map. All methods accept the same point cloud and goal state inputs, the radius $l_r = 0.35$ m and height $h = 0.15$ m of the quadrotor, and the point cloud's resolution is 0.1. The common dynamic limits were $v_{\max} = 10$ m/s, $a_{\max} = 10$ m/s², and $j_{\max} = 60$ m/s³. For Base-A and our method, ρ represents the time penalty parameter and τ represents the sampling time. For Base-B, λ is the penalty weight and $\mathcal{K}$ is the relative resolution of each trajectory. Since the compared methods employ different objective functions and trajectory-generation mechanisms, their method-specific parameters do not have direct one-to-one correspondences. Base-A adopts a jerk-controlled system with $\rho = 4u_{\max}^2$, $\tau = 0.2$ s, and the same control-space sampling parameter $r = 2$. The resulting jerk sampling interval is $\Delta u = 30$ m/s³. The proposed method uses an acceleration-controlled system with the same sampling parameter $r = 2$, resulting in an acceleration sampling interval of $\Delta u = 5$ m/s². All other shared parameters are kept identical to those of Base-A. Following Yang *et al.*, Base-B used $\rho = 1024$, $\chi_{\text{att}} = 60000$, and $\kappa_{\text{att}} = 16$ ^[20]. All parameter settings, except for the resolution

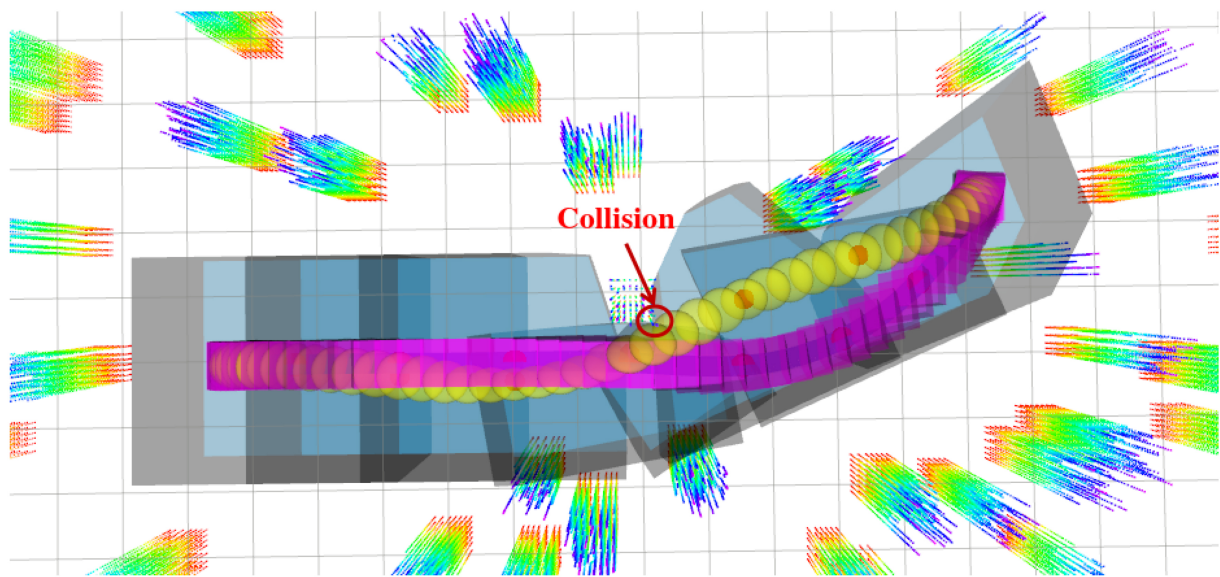


Figure 8. Comparison of the trajectories generated by two search-based methods. The yellow trajectory represents the Base-A, where the trajectory may be too close to the obstacle and may collide when the collision detection resolution is large. The purple trajectory represents our proposed method, which is insensitive to the collision detection resolution.

Table 1. Comparison of search-based methods

	$t_c(s)$	$t_e(s)$	$f(10^3 \times m^2/s^5)$	Node usage
Base-A (res = 0.02 s)	0.172	3.07	8.13	242
Ours (res = 0.02 s)	0.031	3.16	7.09	97
Base-A (res = 0.05 s)	0.166	3.02	7.97	206
Ours (res = 0.05 s)	0.019	3.16	6.71	97

Bold values indicate the best performance for each metric.

explicitly varied in the resolution-sensitivity experiment, were fixed across the evaluated scenarios. All comparisons were performed on an Intel Core i7-6800K CPU, and the source code will be released online (https://github.com/peiyu-cui/uav_motion_planning).

Our method generates SFC through Jump Point Search (JPS) paths and accelerates the generation of collision-free trajectories through hierarchical planning in Section 3.2. Since search-based methods are sensitive to temporal resolution, we first compare our method with Base-A, and the overall comparison is shown in Figure 8.

The results of the benchmark are shown in Table 1. For quantitative comparison, the control cost reported in Tables 1 and 2 is evaluated as $f = \int_0^T \|j(t)\|_2^2 dt$, where $j(t) = \mathbf{p}^{(3)}(t)$ denotes the jerk. The bold represents better performance for each individual metric, that is, shorter calculation time and trajectory execution time, less control cost and node usage. For the Base-A method, high-maneuverability trajectories can be generated quickly when the resolution is large, but due to the large time penalty coefficient ρ , the trajectory is too close to the obstacle and may collide. For our method, trajectories with lower control cost can be generated with shorter computation time and fewer node expansions under different resolutions.

To test the quality of our generated trajectories, we compared our method with the optimization-based method (Base-B). Figure 9 shows the comparative experiments of the three methods in two different scenarios. Our method generates trajectories of similar quality to the optimization-based method by restricting the trajectory search space to SFC. At the same time, through the hierarchical planning framework, our method greatly reduces the computation time of the search-based method while ensuring security. The specific results are given in Table 2.

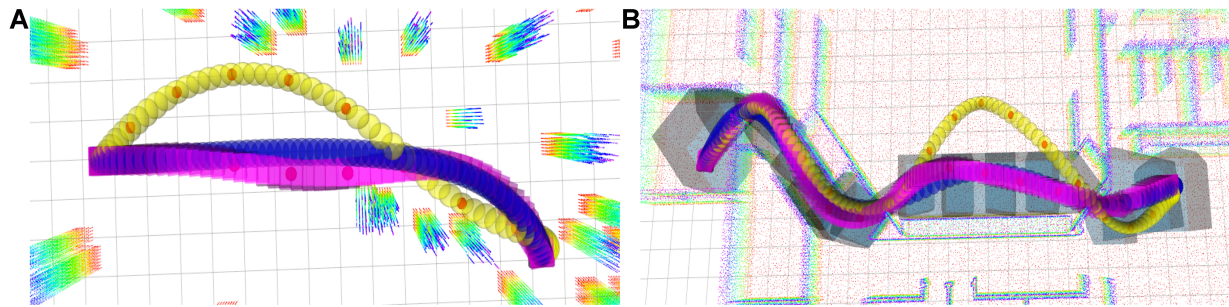


Figure 9. Comparison of the trajectories generated by three methods in two different environments. (A) Trajectories generated by three methods in **random forest**; (B) Trajectories generated by three methods in **Office**. The yellow trajectory represents the search-based method (Base-A), the blue trajectory represents the optimization-based method (Base-B), and the purple trajectory represents our proposed method. It can be seen that our method generates trajectories of comparable quality to the optimization-based method and reduces the trajectory calculation time of the search-based method to the millisecond level.

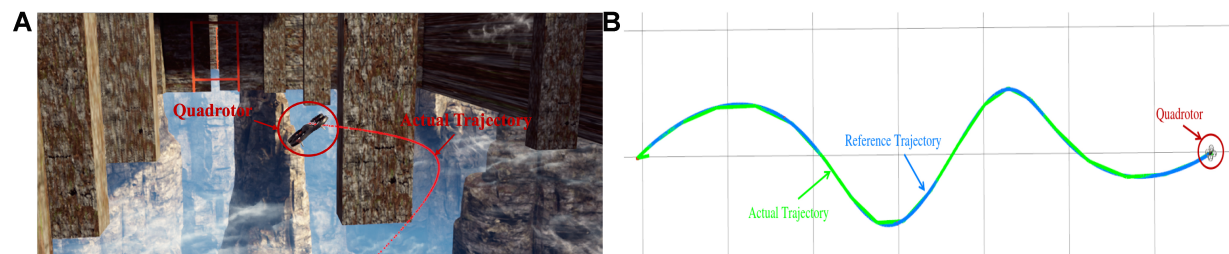


Figure 10. Illustration of a quadrotor tracking an aggressive trajectory in AirSim. (A) AirSim^[26] simulation snapshot of the quadrotor flying in an unknown environment; (B) Trajectory tracking performance. The blue curve denotes the reference trajectory, and the green curve denotes the actual trajectory.

Table 2. Comparison of trajectories generated by three methods

	Random forest			Office		
	$t_c(s)$	$t_e(s)$	$f(10^3 \times m^2/s^5)$	$t_c(s)$	$t_e(s)$	$f(10^3 \times m^2/s^5)$
Base-A	0.916	3.44	9.84	1.923	5.27	10.13
Base-B	0.195	4.29	8.54	0.498	6.04	8.92
Ours	0.138	4.24	8.66	0.455	5.91	9.36

Bold values indicate the best performance for each metric.

5.3. Controller performance under aggressive flight

High-fidelity quadrotor simulator AirSim^[26] is used to verify the tracking performance of the controller under aggressive trajectories. The simulation scene size is $20 \text{ m} \times 70 \text{ m} \times 5 \text{ m}$. The initial position of the quadrotor is $(0, 0, 1)$, the target position is $(0, 66, 1)$, and the reference trajectory duration is set to 11 s to generate an aggressive trajectory. The reference trajectory has a maximum velocity of $v_y = 9.3 \text{ m/s}$ in the y direction and a maximum acceleration of $a_x = 9.5 \text{ m/s}^2$ in the x direction, which are close to the physical limits of actual quadrotors. The DF-MPC problem in Equation (31) is solved using OSQP^[27]. The average solution time remained below 1 ms, satisfying the real-time requirement.

As shown in [Figure 10](#), the scene depicts a quadrotor tracking a very challenging reference trajectory in AirSim. The trajectory is extremely aggressive and is characterized by its significant roll and yaw angle changes. Additional simulation results and visualizations are provided in the [Supplementary Video](#).

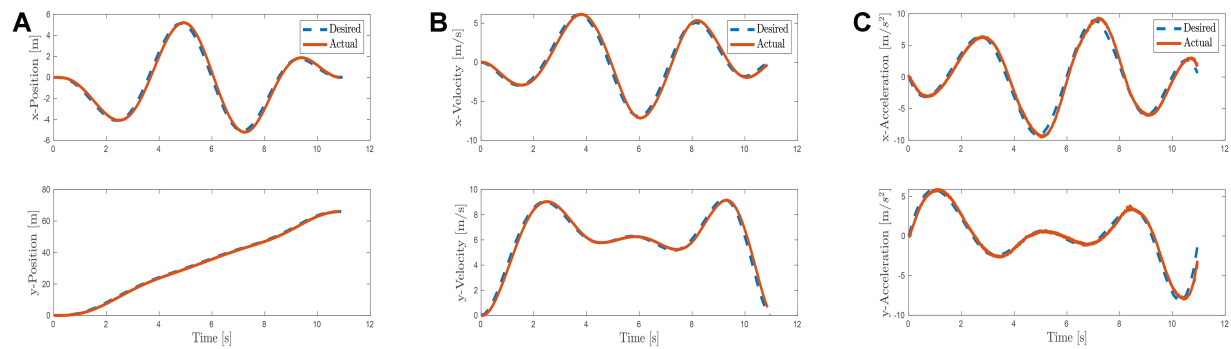


Figure 11. Illustration of the controller's trajectory tracking performance. (A) shows the position tracking performance; (B) shows the velocity tracking performance; and (C) shows the acceleration tracking performance (reflecting the attitude tracking performance).

For the proposed controller, the discrete time $\Delta t = 0.15$ s and the prediction horizon $N = 16$. The tracking performance of the proposed controller is verified by comparing the position, velocity, and acceleration tracking effects of the quadrotor. Figure 11 shows an illustration of the overall tracking effect, where the blue dashed line represents the reference trajectory and the orange line represents the actual trajectory obtained by the simulator. Our method achieves a position closed loop by tracking the position, and since the acceleration of the quadrotor is directly related to the attitude, the attitude closed loop is achieved by tracking the acceleration.

6. CONCLUSIONS

In this paper, we propose a search-based hierarchical whole-body motion-planning framework for quadrotors in unknown environments, which improves computational efficiency by approximately 4-9 times compared with the existing search-based whole-body planning method. We first divide the free space into spacious and narrow regions according to the spatial scale characteristics of SFCs in different regions. By utilizing a specially designed hierarchical collision detection algorithm, the original motion planning problem is decomposed into position-only planning and attitude-aware planning, which significantly improves the planning efficiency. Finally, a model predictive controller based on differential flatness is designed, and a high-fidelity simulator is used to test the tracking performance of the controller under aggressive trajectories.

DECLARATIONS

Authors' contributions

Made substantial contributions to the research, idea generation, algorithm design, and simulation, wrote and edited the original draft: Cui, P.

Performed critical review, commentary, and revision, as well as provided administrative, technical, and material support: Zhang, H.; Du, Z.; Huang, C.; Sun, J.

Availability of data and materials

The data that support the findings of this study are available from the corresponding author upon reasonable request.

AI and AI-assisted tools statement

Not applicable.

Financial support and sponsorship

This work was supported in part by the National Natural Science Foundation of China (62433014 and 62573324) and in part by the Fundamental Research Funds for the Central Universities (22120230311).

Conflicts of interest

Zhang, H. is a Section Chief Editor of the journal *Intelligence & Robotics* and the Guest Editor of the special topic “Low-Altitude Control and Intelligence”. Zhang, H. was not involved in any steps of the editorial processing, notably including reviewer selection, manuscript handling, or decision-making. The other authors declare that there are no conflicts of interest.

Ethical approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Copyright

© The Author(s) 2026.

Supplementary Materials

[Supplementary Materials](#)

REFERENCES

1. Mellinger, D.; Kumar, V. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China. May 09–13, 2011. IEEE; 2011. pp. 2520–5. [DOI](#)
2. Zhou, X.; Wang, Z.; Ye, H.; Xu, C.; Gao, F. EGO-planner: an ESDF-free gradient-based local planner for quadrotors. *IEEE. Robot. Autom. Lett.* **2021**, *6*, 478–85. [DOI](#)
3. Zhou, B.; Gao, F.; Wang, L.; Liu, C.; Shen, S. Robust and efficient quadrotor trajectory generation for fast autonomous flight. *IEEE. Robot. Autom. Lett.* **2019**, *4*, 3529–36. [DOI](#)
4. Gao, F.; Wang, L.; Zhou, B.; Zhou, X.; Pan, J.; Shen, S. Teach-repeat-replan: a complete and robust system for aggressive flight in complex environments. *IEEE. Trans. Robot.* **2020**, *36*, 1526–45. [DOI](#)
5. Liu, S.; Atanasov, N.; Mohta, K.; Kumar, V. Search-based motion planning for quadrotors using linear quadratic minimum time control. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, Canada. Sep 24–28, 2017. IEEE; 2017. pp. 2872–9. [DOI](#)
6. Wang, Z.; Zhou, X.; Xu, C.; Gao, F. Geometrically constrained trajectory optimization for multicopters. *IEEE. Trans. Robot.* **2022**, *38*, 3259–78. [DOI](#)
7. Deng, W.; Chen, H.; Ye, B.; Chen, H.; Li, Z.; Lyu, X. Whole-body integrated motion planning for aerial manipulators. *IEEE. Trans. Robot.* **2025**, *41*, 6661–79. [DOI](#)
8. Sun, Y.; Zhu, H.; Liang, Z.; Liu, A.; Ni, H.; Wang, Y. A phase search-enhanced Bi-RRT path planning algorithm for mobile robots. *Intell. Robot.* **2025**, *5*, 404–18. [DOI](#)
9. Falanga, D.; Mueggler, E.; Faessler, M.; Scaramuzza, D. Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, Singapore. May 29 – Jun 03, 2017. IEEE; 2017. pp. 5774–81. [DOI](#)
10. Han, Z.; Wang, Z.; Pan, N.; Lin, Y.; Xu, C.; Gao, F. Fast-racing: an open-source strong baseline for SE(3) planning in autonomous drone racing. *IEEE. Robot. Autom. Lett.* **2021**, *6*, 8631–8. [DOI](#)
11. Cui, P.; Zhang, H.; Du, Z.; Yan, H. PolyMap: a lightweight polyhedral map based on approximate convex decomposition for aerial robot motion planning. *IEEE. Robot. Autom. Lett.* **2026**, *11*, 3055–62. [DOI](#)
12. Wang, Q.; Wang, Z.; Pei, L.; Xu, C.; Gao, F. A linear and exact algorithm for whole-body collision evaluation via scale optimization. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, London, UK. May 29 – Jun 02, 2023. IEEE; 2023. pp. 3621–7. [DOI](#)
13. Rao, K.; Yan, H.; Lv, Y.; Wu, Z.; Wu, X.; Zhang, Y. Decentralized pursuit of an evader with probabilistic collision-free for differential drive robots. *IEEE. Trans. Cybern.* **2026**, *56*, 4391–402. [DOI](#)
14. Liu, S.; Watterson, M.; Mohta, K.; et al. Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments. *IEEE. Robot. Autom. Lett.* **2017**, *2*, 1688–95. [DOI](#)
15. Deits, R.; Tedrake, R. Efficient mixed-integer planning for UAVs in cluttered environments. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, USA. May 26–30, 2015. IEEE; 2015. pp. 42–9. [DOI](#)
16. Wang, Q.; Wang, Z.; Wang, M.; et al. Fast iterative region inflation for computing large 2-D/3-D convex regions of obstacle-free space. *IEEE. Trans. Robot.* **2025**, *41*, 3223–43. [DOI](#)

17. Liu, S.; Mohta, K.; Atanasov, N.; Kumar, V. Search-based motion planning for aggressive flight in SE(3). *IEEE. Robot. Autom. Lett.* **2018**, *3*, 2439-46. DOI
18. Wu, T.; Chen, Y.; Chen, T.; Zhao, G.; Gao, F. Whole-body control through narrow gaps from pixels to action. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, Atlanta, USA. May 19-23, 2025. IEEE; 2025. pp. 11317-24. DOI
19. Zhang, L.; Hu, Y.; Yu, F.; Deng, Y.; Yu, W.; Zou, D. Vision-based end-to-end learning for UAV traversal of irregular gaps via differentiable simulation. *IEEE. Robot. Autom. Lett.* **2026**, *11*, 7748-55. DOI
20. Yang, S.; He, B.; Wang, Z.; Xu, C.; Gao, F. Whole-body real-time motion planning for multicopters. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China. May 30 - Jun 05, 2021. IEEE; 2021. pp. 9197-203. DOI
21. Lee, T.; Leok, M.; McClamroch, N. H. Geometric tracking control of a quadrotor UAV on SE(3). In *49th IEEE Conference on Decision and Control (CDC)*, Atlanta, USA. Dec 15-17, 2010. IEEE; 2010. pp. 5420-5. DOI
22. Faessler, M.; Falanga, D.; Scaramuzza, D. Thrust mixing, saturation, and body-rate control for accurate aggressive quadrotor flight. *IEEE. Robot. Autom. Lett.* **2017**, *2*, 476-82. DOI
23. Faessler, M.; Franchi, A.; Scaramuzza, D. Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE. Robot. Autom. Lett.* **2018**, *3*, 620-6. DOI
24. Peng, Y.; Yan, H.; Rao, K.; Yang, P.; Lv, Y. Distributed model predictive control for unmanned aerial vehicles and vehicle platoon systems: a review. *Intell. Robot.* **2024**, *4*, 293-317. DOI
25. Verriest, E. I.; Lewis, F. L. On the linear quadratic minimum-time problem. *IEEE. Trans. Autom. Control.* **1991**, *36*, 859-63. DOI
26. Shah, S.; Dey, D.; Lovett, C.; Kapoor, A. AirSim: high-fidelity visual and physical simulation for autonomous vehicles. *arXiv* **2017**, arXiv:1705.05065. Available online: <https://doi.org/10.48550/arXiv.1705.05065>. (accessed on 13 Aug 2026).
27. Stellato, B.; Banjac, G.; Goulart, P.; Bemporad, A.; Boyd, S. OSQP: an operator splitting solver for quadratic programs. In *2018 UKACC 12th International Conference on Control (CONTROL)*, Sheffield, UK. Sep 05-07, 2018. IEEE; 2018. pp. 339. DOI

Disclaimer/Publisher's Note: All statements, opinions, and data contained in this publication are solely those of the individual author(s) and contributor(s) and do not necessarily reflect those of OAE and/or the editor(s). OAE and/or the editor(s) disclaim any responsibility for harm to persons or property resulting from the use of any ideas, methods, instructions, or products mentioned in the content.



© The Author(s) 2026. Open Access This article is licensed under a Creative Commons Attribution 4.0 International License (<https://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, sharing, adaptation, distribution and reproduction in any medium or format, for any purpose, even commercially, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.