2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Multi-Fov-Constrained Trajectory Planning for Multirotor Safe Landing

Dong Wang, Jingping Wang, Suqin He, Jinxin Huang, Bangyan Zhang,
Yinian Mao, Guoquan Huang, Chao Xu, and Fei Gao

***Abstract*— In recent years, multirotors have become more and more widely used, such as in aerial photography and delivery. Ensuring a safe landing in emergencies is the most basic requirement, and it is important to make full use of all the sensors of the multirotor. To improve the safety of UAV landing in unknown unstructured scenes, this paper proposes a multi-FOV-constrained trajectory planning algorithm. Due to the discontinuity of multi-FOV constraints and the nonlinearity of UAV dynamics, the entire trajectory planning problem is a nonlinear optimization problem with non-convex constraints. To address this problem, our algorithm contains two stages, a multi-fov-constrained path search algorithm and a safe landing trajectory optimization algorithm. The multi-fov-constrained path search algorithm is used to generate a safe initial path that satisfies the FOV constraint. Then, the safe landing trajectory optimization algorithm generates a safe trajectory, which considers FOV constraints, dynamics, smoothness, and obstacle avoidance. We conducted simulation experiments and real-world experiments to verify the robustness and effectiveness of our algorithm.**

I. Introduction

In recent years, multirotors have become more and more widely used, such as aerial photography and delivery. In crowded places such as urban scenes, drone applications require strong safety guarantees. Ensuring a safe landing in emergencies is the most basic requirement, but it is still challenging for multirotors due to their limited sensor FOV. This paper focuses on trajectory planning for multirotors safe landing with multiple sensors.

The navigation of multirotors in unknown environments relies on onboard sensors for environmental perception. Considering cost, versatility, and lightweight, vision sensors are the most common choice. Most unmanned vehicles and applications cruise within a certain altitude range. To ensure cruising safety, horizontal observation is required, a forward-looking camera is often used and is sufficient since the heading direction of the drone can be independently controlled. In addition, to have the ability to make an autonomous emergency landing, vertical observation is required to avoid ground obstacles, and a downward-looking camera is sufficient. In summary, for an autonomous navigation drone, the simplest sensor configuration consists of a forward-looking camera and a downward-looking camera. Based on the above analysis of sensor capabilities, and without loss of generality, we assume that there is a certain blind spot angle between the two cameras to design our safe landing trajectory generation method.

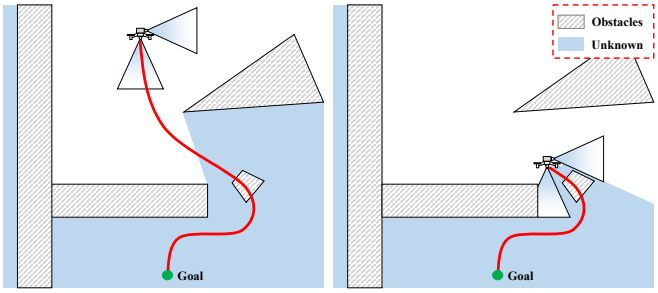

Fig. 1: An example of planning without considering FOV constraints. The drone has two onboard cameras. Due to the FOV constraints are not satisfied, part of the trajectory is in the visual blind spot, causing danger to the drone.

Ensuring the safety of the flight process depends on timely observation and replanning, therefore, movement in the direction of the blind zone should be avoided, as shown in Fig.1. For the above sensor configuration, to complete a safe landing and avoid the risk of blind spot, the future trajectory has to switch between two FOVs. A straightforward idea is to add multi-FOV constraints within the existing trajectory generation framework. Existing methods [1]–[6] focused on generating single-FOV-constrained trajectories, but due to the discontinuity of multi-FOV constraints, it is difficult to directly apply them to the problem of generating multi-FOV-constrained trajectories. As shown in Fig.7(c), the trajectory switches between the two FOVs at high frequency, resulting in poor trajectory quality. This phenomenon inspired us to divide the trajectory into multiple sub-segments connected by switching points, so that each trajectory segment only needs to consider a single FOV constraint, eliminating discontinuity of constraints. Finding suitable switching point locations can greatly improve the quality of trajectory optimization since the switching points not only include location information but also include the decision-making information of FOV switching. Based on this analysis, we propose a trajectory generation framework, which consists of a multi-FOV-constrained path search algorithm and a safe landing trajectory optimization method. The path search algorithm can generate a safe initial path that satisfies the FOV constraints and contains switching point decision information. We propose a heuristic function to speed up the algorithm. Using this decision information, the original multi-FOV-

Dong Wang, Jingping Wang, Chao Xu, and Fei Gao are with the Institute of Cyber-Systems and Control, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China, and also with the Huzhou Institute, Zhejiang University, Huzhou 313000, China.

Suqin He, Jinxin Huang, Bangyan Zhang, Yinian Mao, and Guoquan Huang are with the Meituan UAV, Beijing, China.

This work was supported by the National Natural Science Foundation of China under grant no. 62322314 and the Pioneer and Leading Goose R&D Program of Zhejiang under Grant 2024C01170.

Email: (dongwangab, fgaoaa)@zju.edu.cn

constrained optimization problem is decomposed into several single-FOV-constrained optimization sub-problems, which are efficiently solved through our joint optimization method.

The contributions are summarized as:

1) We proposed a path search algorithm with multi-FOV constraints and corresponding heuristic functions, which can generate an initial path with decision-making information.
2) We propose an efficient trajectory optimization algorithm for safe trajectory generation, which decomposes the original problem into several sub-problems and then solves them by joint optimization.
3) We verify the effectiveness and robustness of the algorithm in simulations and real-world experiments.

II. Related Work

A. Multirotor Trajectory Planning

Trajectory planning for multirotors is a widely studied problem. One of the earliest works [7] proposes to generate minimum-snap trajectory by representing the trajectory as piecewise polynomials and solving a quadratic programming(QP) problem. Richter et al. [8] shows a closed-form solution of minimum-snap trajectory, which generates a safe trajectory by iteratively inserting points. Since the obstacle environment is nonconvex, the trajectory generation problem is difficult to solve. To this end, some works [9]–[13] adopt corridor-based methods, which are multiple convex units extracted from safe areas in the environment. These methods can ensure safe and smooth trajectories but strongly rely on time allocation to satisfy dynamic constraints. Wang et al. [14] propose the MINCO trajectory class based on optimality conditions of an unconstrained control effort minimization problem. Han et al. [15] develop parallel computing for this planning problem in the racing scene, significantly enhancing efficiency. Wang et al. [16] implements time-uniform MINCO to enhance the timeliness of replanning for high-speed flight in dynamic environments.

B. Navigation With Limited Sensors

Compared with general trajectory planning, less attention is paid to navigation with limited perception in unknown environments. Many methods [12, 17]–[19] adopt the optimistic assumption, which considers unknown regions to be safe. This strategy does not guarantee safety, restricting their application scenarios. Tordesillas et al. [20] propose a strategy that plans an extra backup trajectory in known-free space, but does not consider the sensor model. Much attention has been paid to planning with limited perception applications on localization [21, 22] or target tracking [23]–[25]. Zhou et al. [26] present a robust and perception-aware(PA) replanning framework, the awareness is introduced through a visibility metric, and the heading is independently planned of the translational motion. Nevertheless, this method of heading planning cannot consider unknown areas in the vertical direction.

To guarantee safety, other works regard unknown areas as unsafe and constrain trajectories within known areas [27] or sensor FOV [1]–[6]. Lopez et al. [2, 3] propose motion primitive approach, which pre-generates motion primitives within sensor FOV and selects the optimal and safe one based on sensor observations. This approach is short-sighted and requires additional global planning to complete the task. Nieuwenhuisen et al. [1] employ a search-based method on a modified grid to generate long-distance paths that satisfy FOV constraints, which has good optimality guarantees but only horizontal sensors are considered. Considering a full sensor model, Lu et al. [5] directly penalize the temporal, nonconvex, and noncontinuous FOV constraints in one optimization formulation and adopt a sampling-based MPC to solve it in real time. Recently, Yu et al. [6] propose a motion planner with complete perception awareness, which first generates a PA-satisfied initial path and is then refined by a nonlinear optimization process. However, these two methods only consider single FOV constraints and cannot be generalized to multi-FOV constraints.

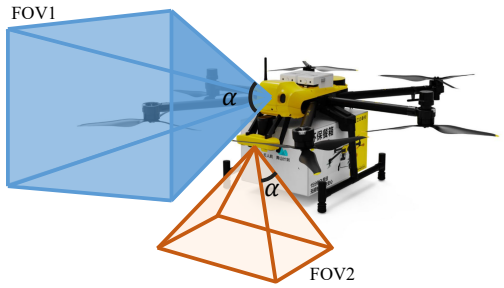

Fig. 2: The hexacopter platform: we use a meituan UAV as our experiment platform, which has two onboard cameras, one with a forward-looking field of view and the other with a downward-looking field of view.

III. Preliminaries

A. Notation

1) Sensor Configuration: The sensor configuration considered in this work is shown in Fig.2. Assume that the camera has equivalent vertical and horizontal sensing angle α, and the two sensor axes are aligned with the x-axis and -z-axis of the body respectively. The two cameras are rigidly connected to the hexacopter, so the observation range is affected not only by the position of the drone but also by its attitude.

2) Differential Flatness: Simultaneously planning the trajectory of the position and attitude is a high-dimensional problem. Fortunately, multirotors are typical differently-flat systems, and their simplified linear dynamics models are widely utilized by the community [14, 16, 26] to reduce computing. We build the dynamics as an s-order linear chain model,

$$\begin{aligned} \dot{\mathbf{x}}(t) &= f\big(\mathbf{x}(t), \mathbf{u}(t)\big) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \\ \mathbf{u}(t) = \boldsymbol{p}^{(s)}(t)^{\mathrm{T}},\ \mathbf{x}(t) &= [\boldsymbol{p}(t), \cdots, \boldsymbol{p}^{(s-1)}(t)]^{\mathrm{T}}, \end{aligned} \tag{1}$$

where $\boldsymbol{p} = [p_x, p_y, p_z]^{\mathrm{T}} \in \mathbb{R}^3$ is the position variable vector and $\cdot^{(s)}$ denotes its s-times derivative of time. With $s \geq 3$,

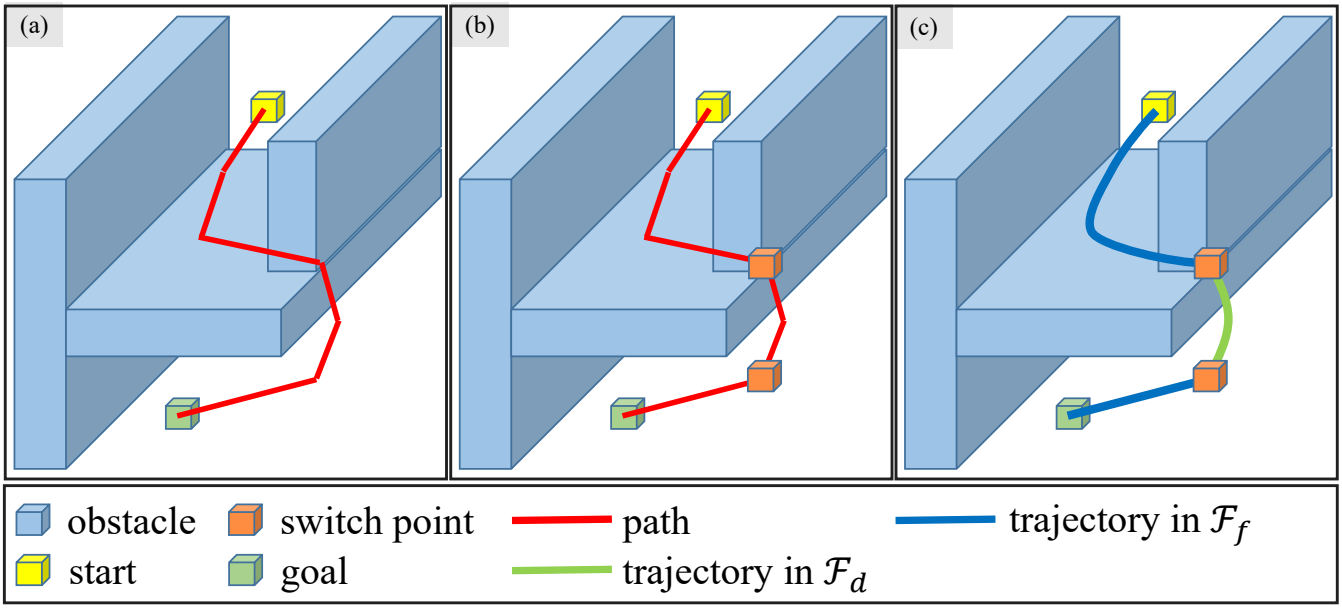

Fig. 3: Framework overview. (a) FOV-constrained pathfinding. (b) Switching points extraction. (c) Trajectory optimization.

the velocity $\boldsymbol{v}$, the rotation $\mathbf{R}$, the angular velocity $\boldsymbol{\omega}$, the thrust f, and other variables that describe the states of the multirotor can then be determined from the chosen flat states through a differential flatness translation,

$$\{\boldsymbol{p}, \boldsymbol{v}, \mathbf{R}, \boldsymbol{\omega}, f\} = \Psi(\mathbf{x}, \mathbf{u}), \tag{2}$$

and the analytical translation of which can be found in our previous work [28].

3) Trajectory Representation: For trajectory representation, we adopt time-uniform MINCO trajectory class [16]. An s-order m-dimensional N-piece time-uniform MINCO is a s'-order piece-wise polynomial, denoted as $p(t)$, where $s' = 2s-1$. The parameters of a trajectory are denoted by $\{\mathbf{C}, \mathbf{T}\}$, where $\mathbf{C} = (\mathbf{c}_1^{\mathrm{T}}, \ldots, \mathbf{c}_N^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{2Ns\times m}$ and $\mathbf{T} = (T_1, \ldots, T_N)^{\mathrm{T}} \in \mathbb{R}^N_{>0}$ are the coefficients and the time durations for each piece. The i-th piece polynomial is defined as

$$p_i(t) = \mathbf{c}_i^{\mathrm{T}} \beta(t), \quad \forall t \in [0, T_i], \tag{3}$$

where $\beta(x) := (1, x, \ldots, x^{s'})^{\mathrm{T}}$ is the natural basis. We denote $p^{[x,y]} \in \mathbb{R}^{m\times(y-x+1)}$ as

$$p^{[x,y]} = \left(p^{(x)}, p^{(x+1)}, \ldots, p^{(y)}\right), \quad x < y. \tag{4}$$

The time-uniform MINCO trajectory class provides a mapping function to parameterize high-dimensional polynomial coefficients $\mathbf{C}$ with fewer variables. The mapping function is defined as

$$\mathbf{C} = \mathcal{M}(\mathbf{Q}, \mathbf{T}, \mathbf{Z}), \tag{5}$$

where $\mathbf{T} = (T/N)\mathbf{1} \in \mathbb{R}^N_{>0}$ is uniform time allocation, $\mathbf{Q} = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{N-1}) \in \mathbb{R}^{m\times(N-1)}$ is composed of the middle waypoints in-between polynomial pieces, and $\mathbf{Z} = (\mathbf{z}_o, \mathbf{z}_f) \in \mathbb{R}^{m\times 2s}$ are the initial and terminal boundary conditions that take derivatives of the position from 0 to $s-1$ order, respectively. The analytical translation of Eq.5 can be found in our previous work [16].

B. Problem Formulation

To generate a smooth trajectory, the quadratic control effort with time regularization is adopted as a cost function of $p(t)$. To ensure trajectory safety and feasibility, the trajectory should satisfy constraints of dynamical feasibility, obstacle avoidance, and perception awareness. Taking all requirements into account, our problem takes the following form:

$$\min_{\mathbf{C},\mathbf{T}} \quad J_o = \int_0^T \frac{1}{2} p^{(s)}(t)^{\mathrm{T}} p^{(s)}(t) dt + \rho T, \tag{6a}$$

$$s.t. \quad p^{[0,s-1]}(0) = \mathbf{z}^o, \tag{6b}$$

$$p^{[0,s-1]}(T) = \mathbf{z}^f, \tag{6c}$$

$$p^1(t) \in \mathcal{F}(t), \forall t \in [0, T] \tag{6d}$$

$$\mathcal{G}_x\left(p(t), ..., p^{(s)}(t), t\right) \preceq \mathbf{0}, \ \forall x \in \mathcal{X}, \forall t \in [0, T], \tag{6e}$$

where Eq.6b-6c are the boundary conditions. Eq.6d is the multi-FOV constraints, where $\mathcal{F}(t) = \mathcal{F}_f(t) \cup \mathcal{F}_d(t)$ is the union of front FOV and downward FOV as shown in Fig.2. Eq.6e is the continuous-time constraints, the set $\mathcal{X} = \{t, b, v, o\}$ include actuator physical limits on thrust (t) and body rate (b), velocity (v), and obstacle avoidance (o).

C. Framework Overview

To solve the above problem, a multi-stage method is adopted in this paper, as shown in Fig.3. First, a multi-FOV-constrained path search algorithm is adopted to find a collision-free path in Sec.IV. Then, after switching points extraction in Sec.V-A, the original problem is decomposed into several single-FOV-constrained sub-problems. Finally, these sub-problems are efficiently solved by our joint optimization method in Sec.V-B.

Algorithm 1 Multi-FOV-constrained FMT*

$V \leftarrow \{x_{init}, x_{goal}\} \cup$ **SampleFree**(n); $E \leftarrow \emptyset$
$V_{unvisited} \leftarrow V \setminus \{x_{init}\}$; $V_{open} \leftarrow \{x_{init}\}$, $V_{closed} \leftarrow \emptyset$
$z \leftarrow x_{init}$
$N_z \leftarrow$ **Near**$(V \setminus \{z\}, z, r_n)$
while $z \neq x_{goal}$ **do**
  $V_{open\_new} \leftarrow \emptyset$
  $X_{near} = N_z \cap V_{unvisited}$
  for $x \in X_{near}$ **do**
    $N_x \leftarrow$ **Near**$(V \setminus \{x\}, x, r_n)$
    $Y_{near} \leftarrow N_x \cap V_{open}$
    $y_{min} \leftarrow \arg\min_{y \in Y_{near}} \{y.c + Cost(y, x)\}$
    if InFovRange(y_{min}, x) **then**
      if CollisionFree(y_{min}, x) **then**
        $E \leftarrow E \cup \{(y_{min}, x)\}$
        $V_{open\_new} \leftarrow V_{open\_new} \cup \{x\}$
        $V_{unvisited} \leftarrow V_{unvisited} \setminus \{x\}$
        $x.c = y_{min}.c + Cost(y_{min}, x)$
        $x.h =$ **Heuristic**(x, x_{goal})
      end if
    end if
  end for
  $V_{open} \leftarrow (V_{open} \cup V_{open\_new}) \setminus \{z\}$
  $V_{closed} \leftarrow V_{closed} \cup \{z\}$
  if $V_{open} = \emptyset$ **then**
    return Failure
  end if
  $z \leftarrow \arg\min_{y \in V_{open}} \{y.c + y.h\}$
end while
return Path$(z, T = (V_{open} \cup V_{closed}, E))$

IV. Multi-FOV-constrained Path Search Algorithm

For initial path planning, we employ a modified FMT* [29] path search method with multi-FOV constraints. Nieuwenhuisen et al. [1] propose to use a modified voxel grid to satisfy the FOV constraints and perform an A* search on it. However, the modified regular grid only applies to single-FOV constraints, and it is difficult to directly construct a graph that satisfies multi-FOV constraints. Fortunately, the FMT* concurrently performs graph construction and graph search, and lazily skips collision-checks when evaluating local connections. Moreover, the lazy checking feature can be applied to checking FOV constraints. Based on this idea, our algorithm details are described in Alg.1. Compared with the original FMT*, we add a function InFovRange$(\cdot)$ before the collision check to satisfy the FOV constraints, and inspired by the heuristic design of Nieuwenhuisen et al. [1], Euclidean distance heuristic underestimates altitude changes, we propose a modified heuristic function to reduce unnecessary node expansion.

Considering that the yaw angle of the drone can be controlled independently, when checking whether the child node is in the FOV range, we assume the yaw angle is towards the child node. For a parent node position p_p and a child position

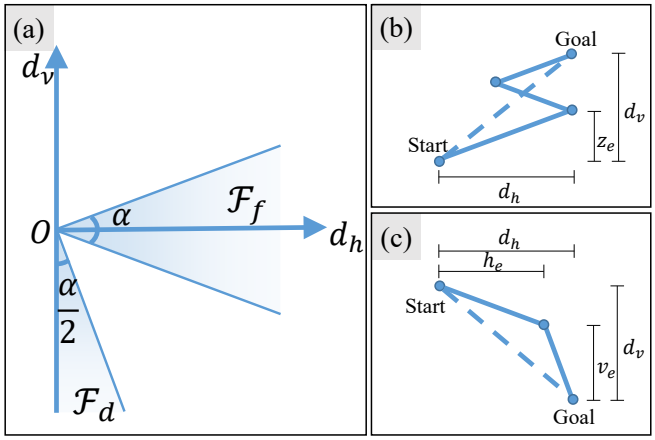

Fig. 4: Illustration of FOV constraints and heuristic design. (a) The sensor model on position difference d. (b)(c) The Euclidean distance underestimates the path length in some cases of our planning problem.

p_c, we define the InFovRange$(p_c - p_p)$ = InFovRange(d) on the position difference d as

$$\text{InFovRange}(d) = \text{InFovRange}_f(d) \vee \text{InFovRange}_d(d), \tag{7a}$$

$$\text{InFovRange}_f(d) = \begin{cases} \text{True}, & -\frac{\alpha}{2} < \phi < \frac{\alpha}{2} \\ \text{False}, & \text{otherwise} \end{cases}, \tag{7b}$$

$$\text{InFovRange}_d(d) = \begin{cases} \text{True}, & \phi < \frac{\alpha}{2} - \frac{\pi}{2} \\ \text{False}, & \text{otherwise} \end{cases}, \tag{7c}$$

where $d_v = d_z$ is vertical distance and $d_h = \sqrt{d_x^2 + d_y^2}$ is horizontal distance, $\phi = \arctan(d_v/d_h)$ is the position difference angle, α is the sensing angle as shown in Fig.4. The resulting plans with and without FOV constraints are illustrated in Fig.5.

Since the path direction is limited within the FOV range, the Euclidean distance heuristic underestimates this constraint, we propose a modified heuristic. Fig.4 illustrates the modified heuristic. For a node position p_n and a target position p_t, we define the heuristic $h(p_t - p_n) = h(d)$ on the position difference d as a piecewise function. When $\alpha/2 - \pi/2 < \phi < \alpha/2$, the shortest possible path has two parts, $d_1 = h_e/\cos(\alpha/2)$ is the shortest possible detour with angle $-\alpha/2$ to bring the target point into the FOV range and $d_2 = v_e/\cos(\alpha/2)$ is the shortest path with angle $\alpha/2 - \pi/2$. As shown in Fig.4(c),

$$\frac{d_1}{h_e + |d_v| - v_e} = \frac{d_2}{v_e + d_h - h_e} = \frac{1}{\cos\frac{\alpha}{2} + \sin\frac{\alpha}{2}} \tag{8a}$$

$$\frac{d_1 + d_2}{h_e + |d_v| - v_e + v_e + d_h - h_e} = \frac{1}{\cos\frac{\alpha}{2} + \sin\frac{\alpha}{2}} \tag{8b}$$

$$h(d) = \frac{d_h + |d_v|}{\cos\frac{\alpha}{2} + \sin\frac{\alpha}{2}} \tag{8c}$$

When $\phi > \alpha/2$, $h(d) = |d_v|/\sin\frac{\alpha}{2}$ is the shortest possible detour to overcome the altitude difference d_v with maximum

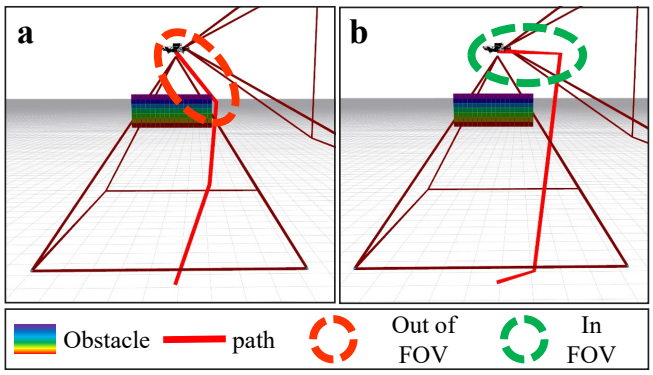

Fig. 5: Planning under FOV constraints. (a) Without FOV constraints, the shortest path from the start point to the end point is out of the sensor range. (b) With FOV constraints, the path switches between the forward-looking FOV and the downward-looking FOV twice.

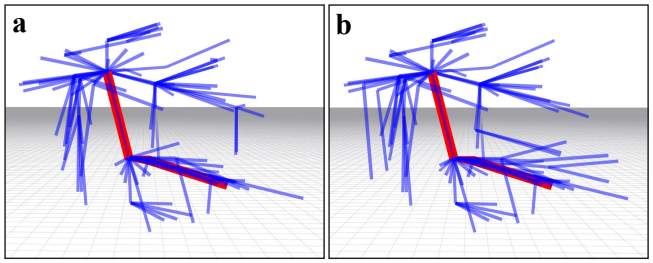

Fig. 6: Planning with different heuristics. (a) Euclidean distance heuristic, $t = 1.97ms, N_{expand} = 8, N_{vertex} = 101$. (b) Our modified heuristic,$t = 1.83ms, N_{expand} = 7, N_{vertex} = 107$.

angle $\alpha/2$, as shown in Fig.4(b). The modified heuristic function is summarized as follows:

$$h(d) = \begin{cases} \dfrac{|d_v|}{\sin\frac{\alpha}{2}}, & \phi > \dfrac{\alpha}{2} \\ \dfrac{d_h + |d_v|}{\cos\frac{\alpha}{2} + \sin\frac{\alpha}{2}}, & \dfrac{\alpha}{2} - \dfrac{\pi}{2} < \phi < -\dfrac{\alpha}{2} \\ \sqrt{d_h^2 + d_v^2}, & \text{otherwise} \end{cases}. \quad (9)$$

The resulting expanded nodes with the Euclidean distance heuristic and our modified heuristic are illustrated in Fig.6. Compared with the Euclidean distance heuristic, our modified heuristic reduces algorithm time by avoiding unnecessary node expansion.

V. Safe Landing Trajectory Optimization

A. Problem Decomposition

Given the initial path $\mathbf{P} = \{P_0, P_1, \dots, P_{N_P}\}$, we perform switching points extraction and decompose the problem accordingly. If the FOV of the path changes after passing through the point P_i, then P_i is a switching point. The algorithm details of switching points extraction are described in Alg.2, where the fucntion SwitchFov$(\cdot)$ is defined as SwitchFov(P_{i-1}, P_i, P_{i+1}) = InFovRange$_f(P_i - P_{i-1}) \oplus$ InFovRange$_f(P_{i+1} - P_i)$. After switching points extraction, we get a switching points set $\mathbf{P_S} = \{P_{S_0}, P_{S_1}, \dots, P_{S_M}\}$, and the initial path between each switching point belongs to a single FOV.

Algorithm 2 Switching Points Extraction

```
P_S ← {P_0}
for i ∈ {1, ..., N_P − 1} do
    if SwitchFov(P_{i−1}, P_i, P_{i+1}) then
        P_S ← P_S ∪ {P_i}
    end if
end for
P_S ← P_S ∪ {P_{N_P}}
return P_S
```

Based on the switching points information, the multi-FOV constraints Eq.6d in the original trajectory generation problem can be simplified. Given the switching points set $\mathbf{P_S}$, we decompose the problem Eq.6 into M sub-problem, each of which has a single-FOV constraint, correspondingly. Assuming it belongs to forward FOV, the i-th sub-problem is formulated as

$$\min_{\mathbf{C}_i, \mathbf{T}_i} \quad J_o = \int_0^{T_i} \frac{1}{2} p_i^{(s)}(t)^{\mathrm{T}} p_i^{(s)}(t) dt + \rho T_i, \quad (10a)$$

$$s.t. \quad p_i^{[0,s-1]}(0) = \mathbf{z}_i^o, \quad (10b)$$

$$p_i^{[0,s-1]}(T_i) = \mathbf{z}_i^f, \quad (10c)$$

$$p_i^1(t) \in \mathcal{F}_f(t), \forall t \in [0, T_i] \quad (10d)$$

$$\mathcal{G}_x\left(p_i(t), ..., p_i^{(s)}(t), t\right) \preceq \mathbf{0}, \ \forall x \in \mathcal{X}, \forall t \in [0, T_i], \quad (10e)$$

where the the multi-FOV constraints Eq.6d is replaced by a single-FOV constraint Eq.10d. The formulation of the sub-problem belonging to downward FOV is similar.

It is worth noting that the speed of the switching point is constrained by two FOVs simultaneously. To achieve this, we constrain the velocity and its high order at switching points to be zero.

B. Joint Trajectory Optimization

Given M sub-problem, we plan an M-segment trajectory. As shown in Fig.3(c), each colored curve represents one segment. The i-th segment $p_i(t)$ is an s-order m-dimensional N_i-piece time-uniform MINCO. We stack the parameters of the whole segmented trajectory $p(t)$ as follows,

$$\begin{aligned} \boldsymbol{\mathcal{C}} &= (\mathbf{C}_1^{\mathrm{T}}, \dots, \mathbf{C}_i^{\mathrm{T}}, \dots, \mathbf{C}_M^{\mathrm{T}})^{\mathrm{T}} && \in \mathbb{R}^{(\sum_{i=1}^M 2N_i s) \times m}, \\ \boldsymbol{\mathcal{T}} &= (T_1, \dots, T_i, \dots, T_M)^{\mathrm{T}} && \in \mathbb{R}^M_{>0}, \\ \boldsymbol{\mathcal{Q}} &= (\mathbf{Q}_1, \dots, \mathbf{Q}_i, \dots, \mathbf{Q}_M) && \in \mathbb{R}^{m \times \left(\sum_{i=1}^M (N_i - 1)\right)}, \\ \boldsymbol{\mathcal{Z}} &= (\mathbf{z}_0, \dots, \mathbf{z}_i, \dots, \mathbf{z}_M)^{\mathrm{T}} && \in \mathbb{R}^{(\sum_{i=1}^M 2N_i s) \times m}. \end{aligned} \quad (11)$$

The mapping function for the i-th segment is

$$\mathbf{C}_i = \mathcal{M}(\mathbf{Q}_i, (T_i/N_i)\mathbf{1}, (\mathbf{z}_{i-1}, \mathbf{z}_i)). \quad (12)$$

1) Constraints elimination: Using the above mapping function, combined with Eq.10, we eliminate the equality constraints Eq.6b-6c and FOV constraints at switching

points. The problem formulation of the joint trajectory optimization is

$$\min_{\boldsymbol{\mathcal{Q}},\boldsymbol{\mathcal{T}},\boldsymbol{\mathcal{Z}}} \quad J_o = \sum_{i=1}^{M} \Big(\int_0^{T_i} \frac{1}{2} p_i^{(s)}(t)^{\mathrm{T}} p_i^{(s)}(t) dt + \rho T_i \Big), \tag{13a}$$

$$s.t. \quad p_i^1(t) \in \mathcal{F}_i(t), \forall i \in [1, M], \forall t \in [0, T_i] \tag{13b}$$

$$\mathcal{G}_x \Big(p(t), ..., p^{(s)}(t), t \Big) \preceq \mathbf{0}, \ \forall x \in \mathcal{X}, \forall t \in [0, \sum_{i=1}^{M} T_i], \tag{13c}$$

where Eq.13b is the corresponding FOV constraints from Eq.10. The inequality constraints are formulated as

$$\mathcal{G}\big(p(t), ..., p^{(s)}(t), t\big) = \begin{pmatrix} (f(t) - f_m)^2 - f_r^2 \\ \|^b\boldsymbol{\omega}(t)\|_2^2 - {\Omega_{\max}}^2 \\ \|\boldsymbol{v}(t)\|_2^2 - {\mathrm{v}_{\max}}^2 \\ r_{tol} - f_{\mathcal{DIST}}\big(\boldsymbol{p}(t)\big) \end{pmatrix}, \tag{14a}$$

$$f_m = (f_{max} + f_{min})/2, \ f_r = (f_{max} - f_{min})/2. \tag{14b}$$

The FOV constraints are formulated as

$$\mathcal{F}\big(p(t), ..., p^{(s)}(t), t\big) = \begin{pmatrix} \|\boldsymbol{v}_z(t)\|_2^2 - \|\tan\frac{\alpha}{2}\boldsymbol{v}_{xy}(t)\|_2^2 \\ \|\boldsymbol{v}_{xy}(t)\|_2^2 - \|\tan\frac{\alpha}{2}\boldsymbol{v}_z(t)\|_2^2 \end{pmatrix}, \tag{15a}$$

$$\boldsymbol{v}_{xy}(t) = \sqrt{\boldsymbol{v}_x(t)^2 + \boldsymbol{v}_y(t)^2}. \tag{15b}$$

According to the method used in previous work [16], these inequality constraints Eq.14 can be transformed into a weighted sum of the sampled penalty as

$$\min_{\boldsymbol{\mathcal{Q}},\boldsymbol{\mathcal{T}},\boldsymbol{\mathcal{Z}}} \quad \sum\nolimits_x^{\mathcal{X}} \lambda_x J_x, \tag{16a}$$

$$J_x = \sum\nolimits_{n=1}^{N} \frac{T_n}{\kappa_n} \sum\nolimits_{j=0}^{\kappa_n} \bar{\omega}_j max(\mathcal{G}_x(n,j), 0)^3, \tag{16b}$$

where λ_x is the weight for each cost J_x, κ_n is the sample quantity, we follow the trapezoidal rule [30] $(\bar{\omega}_0, \bar{\omega}_1, \ldots, \bar{\omega}_{\kappa_i}) = (1/2, 1, \cdots, 1, 1/2)$. Apply a similar way to FOV constraints Eq.15 we can get the FOV constraints penalty J_f, J_d.

2) Unconstrained Formulation & Derivative Deducing: So far, we transform Eq.13 to an unconstrainted optimization problem as follows,

$$\min_{\boldsymbol{\mathcal{Q}},\boldsymbol{\mathcal{T}},\boldsymbol{\mathcal{Z}}} \quad \sum_{i=1}^{M} \left\{ \mathcal{J}_{ctrl} + \rho_r T_i + J_f + J_d + \sum_{x}^{\mathcal{X}} \sum_{j=1}^{N_i} J_x \right\}. \tag{17}$$

We now deduce derivatives of each component with respect to $\{\boldsymbol{\mathcal{C}}, \boldsymbol{\mathcal{T}}\}$, and the derivatives with respect to the decision variables $\{\boldsymbol{\mathcal{Q}}, \boldsymbol{\mathcal{T}}, \boldsymbol{\mathcal{Z}}\}$ is provided by time-uniform MINCO. The derivatives of $\mathcal{J}_{ctrl} + \rho_r T_i$ can be easily get. Apply the chain rule to Eq.16b we can obtain $\partial J / \partial \mathcal{G}_x$, and $\partial \mathcal{G}_x / \partial \mathbf{c}_i$ differs for each term. We give the derivatives of each term regarding $\{\mathbf{p}, \boldsymbol{v}, \boldsymbol{q}, \boldsymbol{\omega}, f\}$ as below, while their derivatives relating to $\{\mathbf{x}, \mathbf{u}\}$ is obtained by automatic differentiation[1] according to Eq.2, and finally $\partial \mathcal{G}_x / \partial \mathbf{c}_i$ can be calculated by the chain rule.

For the term about thrust,

$$\frac{\partial \mathcal{G}_t}{\partial f(t)} = 2\left(f(t) - f_m\right). \tag{18}$$

For the term about body rate,

$$\frac{\partial \mathcal{G}_b}{\partial \boldsymbol{\omega}(t)} = 2\boldsymbol{\omega}(t). \tag{19}$$

For the term about velocity,

$$\frac{\partial \mathcal{G}_v}{\partial \boldsymbol{v}(t)} = 2\boldsymbol{v}(t). \tag{20}$$

For the term about collision avoidance,

$$\frac{\partial \mathcal{G}_o}{\partial \mathbf{p}(t)} = -\frac{\partial f_{\mathcal{DIST}}}{\partial \mathbf{p}(t)}, \tag{21}$$

which is obtained by the rebound function in [19] for obstacle avoidance.

For the term about FOV constraints,

$$\frac{\partial \mathcal{F}_f}{\partial \boldsymbol{v}(t)} = 2\boldsymbol{v}_z(t) - 2\tan\frac{\alpha}{2}(\boldsymbol{v}_x(t) + \boldsymbol{v}_y(t)), \tag{22a}$$

$$\frac{\partial \mathcal{F}_d}{\partial \boldsymbol{v}(t)} = 2(\boldsymbol{v}_x(t) + \boldsymbol{v}_y(t)) - 2\tan\frac{\alpha}{2}\boldsymbol{v}_z(t). \tag{22b}$$

VI. Experiment Results

A. Implementation details

To validate our trajectory planning algorithm under multi-FOV constraints, we conducted real-world experiments on a hexacopter platform as fig.2 shows. All computations are performed on an onboard computer: Nvidia Jetson TX2. As Fig.2 shows, the hexacopter is equipped with 2 depth cameras to detect obstacles. The FOV of each depth camera is 0.47rad and their FOVs do not overlap. Furthermore, simulation experiments were conducted to further validate our algorithm and compare it with other methods. All computations are performed with an Intel Core i7-10700 CPU @ 2.90GHZ.

We choose the L-BFGS[2] algorithm [31] as a highly efficient quasi-Newton approach for solving numerical optimization problems and use the Lewis-Overton line search [32] to address instances of non-smoothness in the scale that may arise during the optimization process. The joint trajectory optimization method is re-planned at 10HZ and uses the last planning result as the initial optimization value. The multi-FOV-constrained path search algorithm is executed when the last planned trajectory is unsafe.

B. The Simulation Experiments

To fully test the performance of the proposed planner in safe landing applications, we benchmarked it against the start-of-the-art method: Ego-planner [19]. To make the comparison fair enough, we added the proposed multi-FOV-constrained path search method as the initial trajectory module to the Ego-planner. To validate the proposed joint trajectory optimization method, we also benchmarked it against a modified Ego-planner, which adds an extra cost

[1] http://tapenade.inria.fr:8080/tapenade/index.jsp

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

TABLE I: BENCHMARK RESULTS

method	Suc(%)	t_{land}(sec)	l_{avg}(m)	v_{max}(m/s)	v_{avg}(m/s)	t_{plan}(ms)
Ego	85.0	7.09	13.3	4.61	1.74	1.79
Ego+J_{fov}	**100.0**	27.9	22.9	3.18	0.89	4.59
Proposed	**100.0**	17.5	20.0	3.43	1.20	5.16

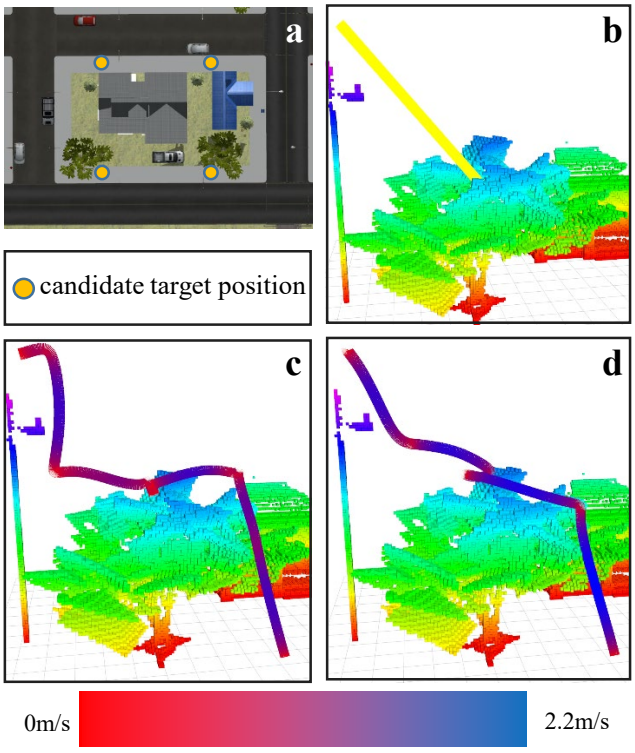

Fig. 7: Benchmark Results. (a) Simulation environments. (b) Ego-planner collided. (c) Ego + J_{fov}, $t_{land} = 20.6s$. (d) Proposed, $t_{land} = 14.2s$.

$J_{fov} = \min(J_f, J_d)$. As shown in Fig.7, we conduct 200 experimental flights for each method in a simulated urban environment[3]. The test range is 54m $\times$ 40m $\times$ 10m. The landing starting position is 10m high and evenly distributed in the horizontal direction. We selected 40 positions and tested each 5 times. The landing target position is selected from the closest candidate target position as shown in Fig.7(a). Fig.7(b)-(d) shows a representative test case.

As shown in Table.I, the success rate of the Ego-planner is lower than other methods, since FOV is not considered. With the cost J_{fov}, the Ego-planner can get a higher success rate. However, due to the discontinuity of the cost, the planning results are of poor quality and the total landing time is very long. Benefiting from multi-FOV-constrained path search and constraints simplification, our method reduces the landing time by 37% and the trajectory length by 12%, and the planning time only increases by 12%.

C. The Real-World Experiment

The real-world experiments involved a hexacopter drone performing landings in urban street environments, where the drone autonomously plans trajectories and avoids obstacles during descent. Two sets of experiments were conducted using a traffic sign and a landscape tree as obstacles, as shown in Fig.8(a1)(b1). During the experiment, the maximum horizontal speed of the hexacopter was limited to 1.5 m/s, while the maximum vertical speed was limited to 2.0 m/s. Fig.8(a2)(b2) shows the real flight trajectories of the drone, along with the fused local maps constructed using onboard sensors. The trajectory variations in the figure highlight the ability of our algorithm to autonomously switch FOV. We demonstrate the effect of FOV constraints by statistically analyzing the angles between the drone's velocity vectors and the horizontal direction. Additionally, we present velocity profiles to demonstrate the algorithm's ability to ensure dynamical feasibility. Results are shown in Fig.9, our

[3]https://github.com/osrf/citysim

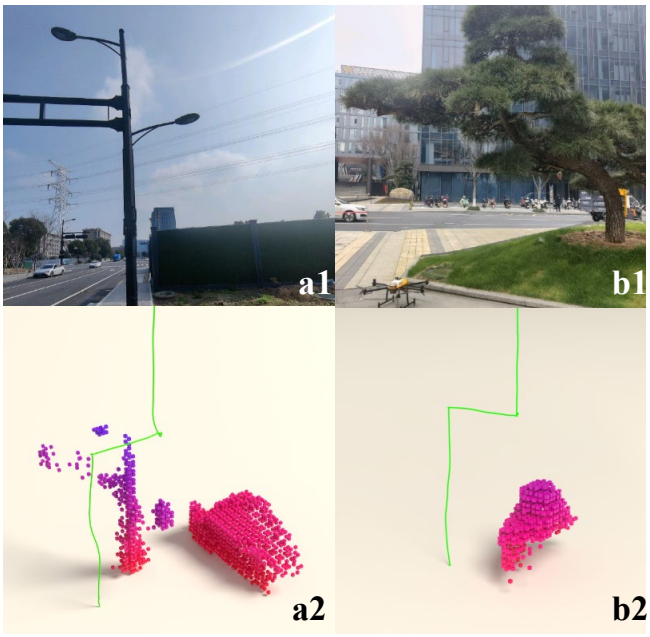

Fig. 8: Real-world experiments: (a1)(b1) The landing environments of the two sets of experiments. (a2)(b2) The corresponding local maps and flight trajectories respectively.

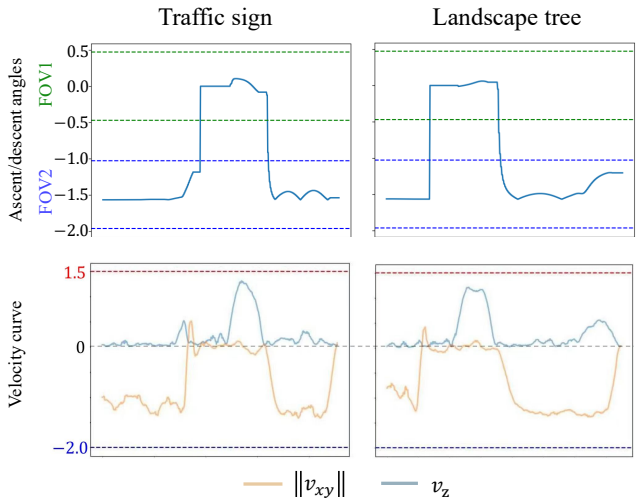

Fig. 9: **Top**: The ascent and descent angles of the hexacopter trajectory. The two FOV ranges are labeled. **Down**: Velocity curve.

algorithm ensures that the flight trajectory always satisfies the FOV constraints.

VII. Conclusion & Future Work

In this paper, we propose to solve the problem of safe landing of UAVs with limited sensing capabilities in unknown environments. We design a multi-FOV-constrained path search algorithm and a joint trajectory optimization method to achieve this. The path search algorithm provides an initial path that contains decision-making information, which is used to split the original problem into several sub-problems. These sub-problems are further efficiently solved by the proposed joint optimization method. The results of simulation experiments and real-world experiments verify the robustness and effectiveness of our algorithm.

In the path search algorithm, future work will consider generating multiple paths and improving efficiency in complex environments. In multi-FOV-constrained trajectory optimization, future work will focus on complete perception awareness.

References

[1] M. Nieuwenhuisen and S. Behnke, "Search-based 3d planning and trajectory optimization for safe micro aerial vehicle flight under sensor visibility constraints," in *International Conference on Robotics and Automation*, 2019, pp. 9123–9129.

[2] B. T. Lopez and J. P. How, "Aggressive 3-d collision avoidance for high-speed navigation," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 5759–5765.

[3] B. T. Lopez and J. P. How, "Aggressive collision avoidance with limited field-of-view sensing," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 1358–1365.

[4] H. Lu, Q. Zong, S. Lai, B. Tian, and L. Xie, "Real-time perception-limited motion planning using sampling-based mpc," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 12, pp. 13 182–13 191, 2022.

[5] H. Lu, Q. Zong, S. Lai, B. Tian, and L. Xie, "Flight with limited field of view: A parallel and gradient-free strategy for micro aerial vehicle," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 9, pp. 9258–9267, 2022.

[6] Q. Yu, C. Qin, L. Luo, H. H.-T. Liu, and S. Hu, "Cpa-planner: Motion planner with complete perception awareness for sensing-limited quadrotors," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 720–727, 2023.

[7] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.

[8] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *The International Symposium of Robotics Research*. Springer, 2016, pp. 649–666.

[9] J. Chen, K. Su, and S. Shen, "Real-time safe trajectory generation for quadrotor flight in cluttered environments," in *IEEE International Conference on Robotics and Biomimetics*, 2015, pp. 1678–1685.

[10] F. Gao and S. Shen, "Online quadrotor trajectory generation and autonomous navigation on point clouds," in *IEEE International Symposium on Safety, Security, and Rescue Robotics*, 2016, pp. 139–146.

[11] F. Gao, W. Wu, Y. Lin, and S. Shen, "Online safe trajectory generation for quadrotors using fast marching method and bernstein basis polynomial," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 344–351.

[12] F. Gao, W. Wu, W. Gao, and S. Shen, "Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *Journal of Field Robotics*, vol. 36, no. 4, pp. 710–733, 2019.

[13] W. Ding, W. Gao, K. Wang, and S. Shen, "An efficient b-spline-based kinodynamic replanning framework for quadrotors," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1287–1306, 2019.

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

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

[16] Q. Wang, D. Wang, C. Xu, A. Gao, and F. Gao, "Polynomial-based online planning for autonomous drone racing in dynamic environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023, pp. 1078–1085.

[17] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3529–3536, 2019.

[18] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[19] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao, "Ego-planner: An esdf-free gradient-based local planner for quadrotors," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 478–485, 2021.

[20] J. Tordesillas, B. T. Lopez, and J. P. How, "Faster: Fast and safe trajectory planner for flights in unknown environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 1934–1940.

[21] L. Bartolomei, L. Teixeira, and M. Chli, "Perception-aware path planning for uavs using semantic segmentation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 5808–5815.

[22] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 1–8.

[23] J. Tordesillas and J. P. How, "Panther: Perception-aware trajectory planner in dynamic environments," *IEEE Access*, vol. 10, pp. 22 662–22 677, 2022.

[24] P. Mali, A. K. Singh, M. Krishnal, and P. Sujit, "Model predictive control for target tracking in 3d with a downward facing camera equipped fixed wing aerial vehicle," in *IEEE International Conference on Automation Science and Engineering*, 2020, pp. 165–172.

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

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

[27] S. Liu, M. Watterson, S. Tang, and V. Kumar, "High speed navigation for quadrotors with limited onboard sensing," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 1484–1491.

[28] Z. Wang, C. Xu, and F. Gao, "Robust trajectory planning for spatial-temporal multi-drone coordination in large scenes," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022, pp. 12 182–12 188.

[29] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 883–921, 2015.

[30] W. H. Press, H. William, S. A. Teukolsky, A. Saul, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 2007.

[31] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1, pp. 503–528, 1989.

[32] A. S. Lewis and M. L. Overton, "Nonsmooth optimization via quasi-newton methods," *Mathematical Programming*, vol. 141, pp. 135–163, 2013.