FUEL: Fast UAV Exploration Using Incremental Frontier Structure and Hierarchical Planning

Boyu Zhou, Yichen Zhang, Xinyi Chen, and Shaojie Shen

***Abstract*—Autonomous exploration is a fundamental problem for various applications of unmanned aerial vehicles(UAVs). Existing methods, however, were demonstrated to insufficient exploration rate, due to the lack of efficient global coverage, conservative motion plans and low decision frequencies. In this letter, we propose FUEL, a hierarchical framework that can support *F*ast *U*AV *E*xp*L*oration in complex unknown environments. We maintain crucial information in the entire space required by exploration planning by a frontier information structure (FIS), which can be updated incrementally when the space is explored. Supported by the FIS, a hierarchical planner plans exploration motions in three steps, which find efficient global coverage paths, refine a local set of viewpoints and generate minimum-time trajectories in sequence. We present extensive benchmark and real-world tests, in which our method completes the exploration tasks with unprecedented efficiency (3-8 times faster) compared to state-of-the-art approaches. Our method will be made open source to benefit the community[1].**

***Index Terms*—Aerial systems: applications, aerial systems: perception and autonomy, motion and path planning.**

I. Introduction

UNMANNED aerial vehicles, especially quadrotors have gained widespread popularity in various applications, such as inspection, precision agriculture, and search and rescue. Among the tasks, autonomous exploration, in which the vehicle explores and maps the unknown environment to gather information, is a fundamental component.

Various exploration planning methods have been proposed in recent years, with some real-world experiments presented [1]–[4]. However, most of them demonstrate a low/medium exploration rate, which is unsatisfactory for many large-scale real-world applications. First of all, many existing planners plan exploring motions in greedy manners, such as maximizing the immediate information gain, or navigating to the closest unknown region. The greedy strategies ignore global optimality and therefore result in low overall efficiency. Besides, most methods generate rather conservative motions in order to guarantee information gain and safety simultaneously in previously unknown environments. Low-speed exploration, however, disallows quadrotors to fully exploit their dynamic capability to fulfill the missions. Last but not least, many methods suffer from high computational overhead and can not respond quickly and frequently to environmental changes. However, to enable faster exploration, it is desirable to replan new motions immediately whenever new information of the environment is available.

Motivated by the above facts, this paper proposes **FUEL**, a hierarchical framework that can support **F**ast **U**AV **E**xp**L**oration in complex environments. We introduce a frontier information structure (FIS), which contains essential information in the entire space required by exploration planning. The structure can be updated efficiently and incrementally when new information is collected, so it is capable of supporting high-frequency planning. Based on the FIS, a hierarchical planner generates exploring motion in three coarse-to-fine steps. It starts by finding a global exploration tour that is optimal in the context of accumulated environment information. Then, viewpoints on a local segment of the tour are refined, further improving the exploration rate. Finally, a safe, dynamically feasible and minimum-time trajectory is generated. The planner produces not only efficient global coverage paths, but also safe and agile local maneuvers. Moreover, the planner is triggered whenever unvisited regions are explored, so that the quadrotor always responds promptly to any environmental changes, leading to consistently fast exploration.

We compare our method with classic and state-of-the-art methods in simulation. Results show that in all cases our method achieves complete exploration in much shorter time (3-8 times faster). What's more, we conduct fully onboard real-world exploration in various challenging environments. Both the simulation and real-world tests demonstrate an unprecedented performance of our method compared to state-of-the-art ones. To benefit the community, we will make the source code public. The contributions of this paper are summarized as follows:

1) An incrementally updated FIS, which captures essential information of the entire explored space and facilitates exploration planning in high frequency.

2) A hierarchical planning method, which generates efficient global coverage paths, and safe and agile local maneuvers for high-speed exploration.

3) Extensive simulation and real-word tests that validate the proposed method. The source code of our system will be made public.

Manuscript received October 15, 2020; accepted December 13, 2020. Date of publication January 14, 2021; date of current version February 1, 2021. This letter was recommended for publication by Associate Editor M. Bangura and Editor P. Pounds upon evaluation of the reviewers; comments. This work was supported by Research Grants Council (RGC) Project 16213717, ITC Project ITT/027/19GP, HDJI Lab. *(Corresponding author: Boyu Zhou.)*

The authors are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong, China (e-mail: 405540572@qq.com; yzhangec@connect.ust.hk; xchencq@connect.ust.hk; eeshaojie@ust.hk).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3051563, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3051563

[1]To be released at https://github.com/HKUST-Aerial-Robotics/FUEL

2377-3766 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Harbin Engineering Univ Library. Downloaded on April 17,2023 at 06:31:46 UTC from IEEE Xplore. Restrictions apply.

II. Related Work

A. Exploration Path Planning

Robotic exploration, which uses mobile robots to map unknown environments, has been studied for years. Some of the works focus on exploring the space quickly [1], [5], as this paper does. Meanwhile, other methods place more emphasis on accurate reconstruction [2], [6]. Among the various proposed methods, the frontier-based approaches are one type of classic approaches. The methods are first introduced in [7] and evaluated more comprehensively afterwards in [8]. To detect frontiers in 3D space, a stochastic differential equation-based method is proposed in [9]. In the original method [7], the closest frontier is selected as the next target. [1] presented a different scheme. In every decision, it selects the frontier within the FOV that minimizes the velocity change to maintain a consistently high flight speed. This scheme is shown to outperform the classic method [7]. In [10], a differentiable measure of information gain based on frontiers is introduced, allowing paths to be optimized with gradient information.

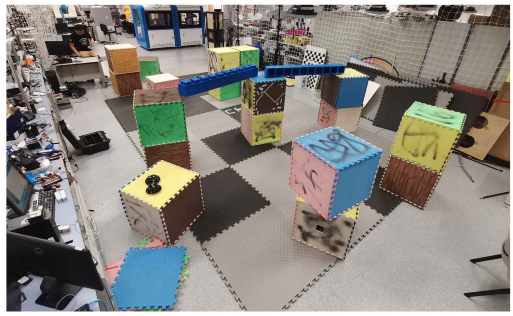
Sampling-based exploration, as another type of major approaches, generate viewpoints randomly to explore the space. These methods are closely related to the concept of next best view (NBV) [11], which computes covering views repeatedly to obtain a complete model of a scene. [12] first used NBV in 3D exploration, in which it expands RRTs with accessible space and executes the edge with the highest information gain in a receding horizon fashion. The method was extended to consider uncertainty of localization [13], visual importance of different objects [14] and inspection tasks [15] later. To avoid discarding the expanded trees directly, roadmaps are constructed in [16], [17] to reuse previous knowledge. [2] maintains and refines a single tree continuously using a rewiring scheme inspired by RRT*. To achieve faster flight, [5] samples safe and dynamically feasible motion primitives directly and execute the most informative one.

There are also methods combining the advantages of frontier-based and sampling-based approaches. [4], [18] plan global paths toward frontiers and sample paths locally. [18] also presented a gradient-based method to optimize the local path. [3] samples viewpoints around frontiers and finds the global shortest tour passing through them. [6] generates inspection paths that cover the frontier completely using a sampling-based algorithm.

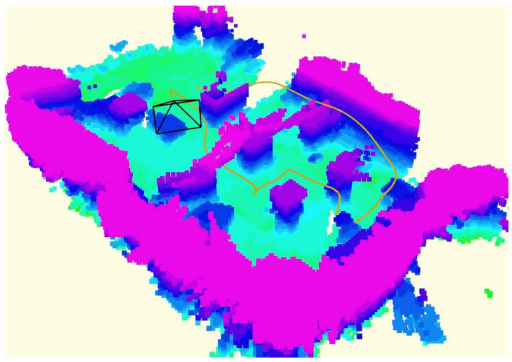
Most of existing methods make decision greedily and does not account for the dynamics of the quadrotor, which leads to inefficient global tours and conservative maneuvers. In contrast, we plan tours that efficiently cover the entire environment and generate dynamically feasible minimum-time trajectories to enable agile flight.

B. Quadrotor Trajectory Planning

Trajectory planning for quadrotor has been widely studied, which can be categorized into the hard-constrained and soft-constrained approaches in major. The former is pioneered by minimum-snap trajectory [19], whose closed-form solution was presented [20] later. Based on [19], [21]–[23] extract convex safe regions for safe trajectory generation. To obtain a more reasonable time allocation, fast marching [22], kinodynamic search [23] and mixed integer-based methods [24] are proposed. [22] also introduced an efficient Bézier curve-based method to guarantee feasibility.



(a) A cluttered environment for the exploration tests.



(b) The online built map and executed trajectory.

Fig. 1. A quadrotor autonomous exploration test conducted in a complex indoor scene. Video of the experiments is available at: https://www.youtube.com/watch?v=_dGgZUrWk-8.

Soft-constrained methods typically formulate a non-linear optimization trading off several objectives. Recently [25]–[30] applied them to local replanning, demonstrating their attractive performance. The methods were revived by [31] and extended to continuous-time trajectories [25] later. To relieve the issue of local minima, [26] initializes the optimization with collision-free paths. [27] introduces uniform B-splines for replanning. More recently, [28] further exploited B-splines and demonstrated fast flight in field tests. [28] is further improved with topological guiding paths and perception-awareness in [29], [30].

In this paper, we base our trajectory planning on [28] but extend it to optimize all parameters of B-splines. In this way, the total trajectory time can be minimized so that the unknown space is explored with a higher navigation speed.

III. System Overview

The proposed framework operates upon a voxel grid map. As illustrated in Fig. 2, it is composed of an incremental update of the FIS (Sect.IV) and a hierarchical exploration planning approach (Sect.V). Whenever the map is updated using sensor measurements, it is examined whether any frontier clusters are influenced. If so, FISs of influenced clusters are removed while new frontiers along with their FISs are extracted (Sect.IV).

Authorized licensed use limited to: Harbin Engineering Univ Library. Downloaded on April 17,2023 at 06:31:46 UTC from IEEE Xplore. Restrictions apply.

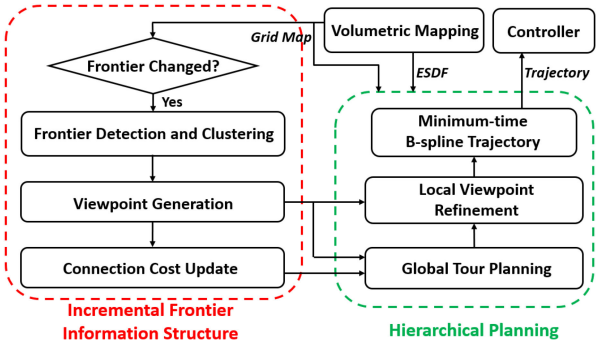


Fig. 2. An overview of the proposed exploration framework.

TABLE I
DATA CONTAINED BY A FIS FI_i OF CLUSTER F_i

Data	Explanation
C_i	Frontier cells that belong to the cluster
$\mathbf{p}_{\text{avg},i}$	Average position of C_i
B_i	Axis-aligned bounding box of C_i
VP_i	Viewpoints covering the cluster
$L_{\text{cost},i}$	Doubly linked list of connection costs to all other clusters

After that, the exploration planning is triggered, which finds global exploration tour, refines local viewpoints, and generates trajectory to a selected viewpoint successively (Sect.V). The exploration is considered finished if no frontier exists.

IV. Incremental Frontier Information Structure

As presented in classic frontier-based exploration [7], frontiers are defined as known-free voxels right adjacent to unknown voxels, which are grouped into clusters to guide the navigation. Traditionally, the extracted information is too coarse to do fine-grained decision making. Besides, frontiers are retrieved by processing the entire map, which is not scalable for large scenes and high planning frequencies. In this work, we extract richer information from frontiers to enable more elaborate planning, and develop an incremental approach to detect frontiers within the locally updated map.

A. Frontier Information Structure

A frontier information structure FI_i is computed when a new frontier cluster F_i is created. It stores all cells C_i belonging to the cluster and the average position $\mathbf{p}_{\text{avg},i}$ of C_i. The axis-aligned bounding box (AABB) B_i of the cluster is also computed, in order to accelerate the detection of frontier changes (Sect.IV-B). To serve the exploration planning (Sect.V), candidate viewpoints VP_i are generated around the cluster. Besides, a doubly linked list $L_{\text{cost},i}$ containing connection costs between F_i and all other clusters is computed. Data stored by a FIS is listed in Table I.

B. Incremental Frontier Detection and Clustering

As depicted in Fig. 3, every time the map is updated by sensor measurements, the AABB of the updated region B_m is also recorded, within which outdated frontier clusters are removed and new ones are searched. It starts with going through all clusters and returning only those whose AABBs (B_i) intersect

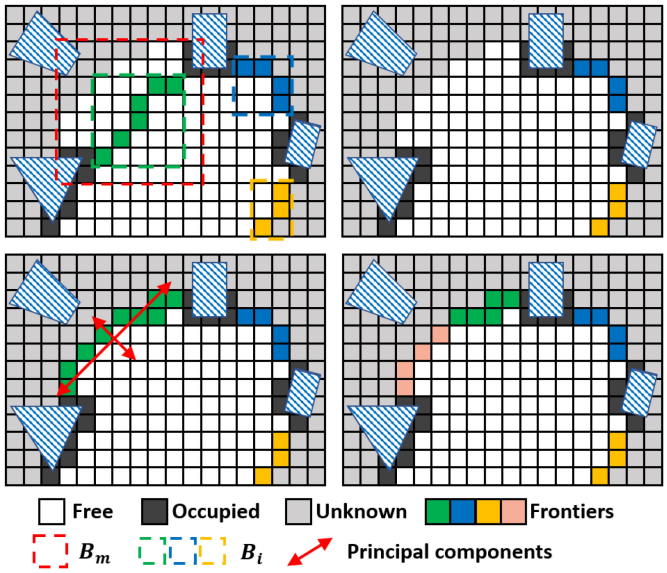


Fig. 3. Incremental frontier detection and clustering. Top: detecting and removing outdated frontiers. Bottom: new frontier is detected (left) and PCA is performed, the large cluster is split into two smaller ones (right).

with B_m. Then, precise checks are conducted for the returned clusters, among which the ones containing cells that are no longer frontier are removed. These two processes are inspired by the broad/narrow phase collision detection algorithms [32], which eliminate most unaffected clusters in a fast way and significantly reduce the number of expensive precise checks.

After the removal, new frontiers are searched and clustered into groups by the region growing algorithm, similar to the classic frontier-based method. Among the groups, the ones with a small number of cells typically resulting from noisy sensor observations are ignored. The remaining groups, however, may contain large-size clusters which are not conducive to distinguishing distinctive unknown regions and making elaborate decisions. Therefore, we perform Principal Component Analysis (PCA) for each cluster and split it into two uniform ones along the first principal axis, if the largest eigenvalue exceeds a threshold. The split is conducted recursively so that all large clusters are divided into small ones.

C. Viewpoint Generation and Cost Update

Intuitively, a frontier cluster implies a potential destination to explore the space. However, unlike previous approaches which simply navigate to the center of a cluster, we desire more elaborate decision making. To this end, when a cluster F_i is created, we generate a rich set of viewpoints $VP_i = \{\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \ldots, \mathbf{x}_{i,n_i}\}$ covering it, where $\mathbf{x}_{i,j} = (\mathbf{p}_{i,j}, \xi_{i,j})$. VP_i are found by uniformly sampling points in the cylindrical coordinate system whose origin locates at the cluster's center, as displayed in Fig. 4. For each of the sampled points $\mathbf{p}$ lying within the free space, the yaw angle ξ is determined as the one maximizing sensor coverage to the cluster, by using a yaw optimization method similar to [16]. The coverage is evaluated as the number of frontier cells that comply with the sensor model and are not occluded by occupied voxels. Then, viewpoints with coverage

Authorized licensed use limited to: Harbin Engineering Univ Library. Downloaded on April 17,2023 at 06:31:46 UTC from IEEE Xplore. Restrictions apply.

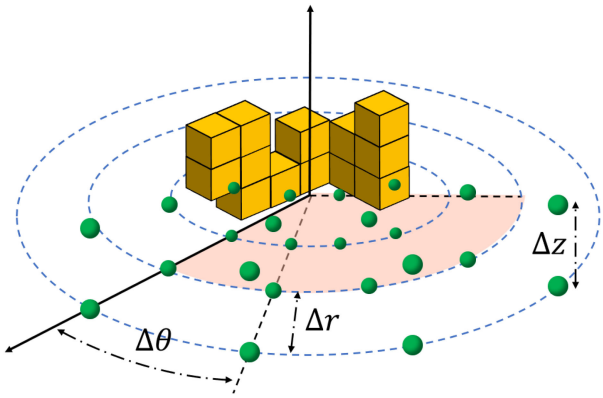


Fig. 4. Generating candidate viewpoints for a frontier cluster. Within the cylindrical coordinate system centered at the average position of the cluster, points are sampled uniformly.

higher than a threshold are reserved and sorted in descending order of coverage. We reserve at most N_{view} viewpoints in VP_i ($n_i \leq N_{\text{view}}$) to make the local viewpoint refinement (Sect.V) tractable.

To perform global planning of exploration tour (Sect.V), a connection cost between each pair of clusters (F_{k_1}, F_{k_2}) is required. Let $t_{\text{lb}}(\mathbf{x}_{k_1,j_1}, \mathbf{x}_{k_2,j_2})$ denotes a time lower bound when moving between two viewpoints $\mathbf{x}_{k_1,j_1}$ and $\mathbf{x}_{k_2,j_2}$, it is computed by:

$$t_{\text{lb}}(\mathbf{x}_{k_1,j_1}, \mathbf{x}_{k_2,j_1}) = \max \left\{ \frac{\text{length}\left(P\left(\mathbf{p}_{k_1,j_1}, \mathbf{p}_{k_2,j_2}\right)\right)}{v_{\max}}, \right.$$
$$\left. \frac{\min\left(|\xi_{k_1,j_1} - \xi_{k_2,j_2}|, 2\pi - |\xi_{k_1,j_1} - \xi_{k_2,j_2}|\right)}{\dot{\xi}_{\max}} \right\} \tag{1}$$

where $P(\mathbf{p}_{k_1,j_1}, \mathbf{p}_{k_2,j_2})$ denote a collision-free path between $\mathbf{p}_{k_1,j_1}$ and $\mathbf{p}_{k_2,j_2}$ found by a path searching algorithm, $v_{\max}$ and $\dot{\xi}_{\max}$ are the limits of velocity and angular rate of yaw. For each pair (F_{k_1}, F_{k_2}), we select the viewpoints with highest coverage and estimate the cost as $t_{\text{lb}}(\mathbf{x}_{k_1,1}, \mathbf{x}_{k_2,1})$, in which $P(\mathbf{p}_{k_1,1}, \mathbf{p}_{k_2,1})$ is searched on the voxel grid map using the A* algorithm.

Note that computing connection costs between all pairs of N_{cls} clusters from scratch requires $O(N_{\text{cls}}^2)$ A* searching, which is considerably expensive. Fortunately, the costs can also be computed in an incremental manner. When any outdated clusters are removed (Sect.IV-B), associated cost items in $L_{\text{cost},i}$ of all remaining FISs are erased. After that, connection costs from each new cluster to all other clusters are computed and inserted into $L_{\text{cost},i}$. Suppose there are k_{new} new clusters in each frontier detection, the above update scheme takes $O(k_{\text{new}} \cdot N_{\text{cls}})$ time. Practically, k_{new} is small and can be regarded as a constant factor, resulting in a linear time update of connection costs.

V. Hierarchical Exploration Planning

Instead of adopting greedy exploration strategies or generating conservative maneuvers, we produce global paths to cover the frontiers efficiently and plan safe and agile motions for faster flight. Our planner takes inspiration from the recent hierarchical quadrotor planning paradigm [21], [22], [28], and makes decisions in three phases, as shown in Fig. 5.

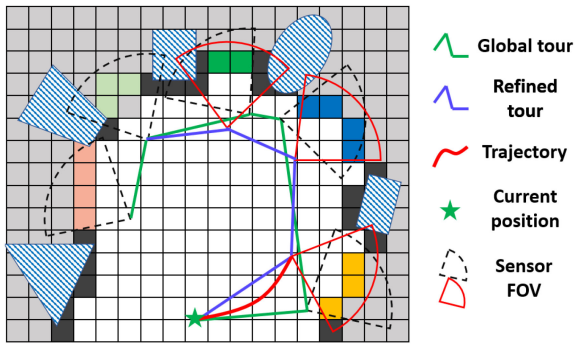


Fig. 5. Generating exploration motion in three coarse-to-fine steps: (1) the shortest tour covering frontier clusters in the entire environment is found. (2) a local segment of the global tour is refined. (3) a safe minimum-time trajectory is generated to the first viewpoint on the refined tour.

A. Global Exploration Tour Planning

Our exploration planning begins with finding a global tour to cover existent frontier clusters efficiently. Inspired by [3], we formulate it as a variant of the Traveling Salesman Problem (TSP), which computes an open-loop tour starting from the current viewpoint and passing viewpoints at all clusters. We reduce this variant to a standard Asymmetric TSP (ATSP) that can be solved quickly by available algorithms, by properly designing the engaged cost matrix $\mathbf{M}_{\text{tsp}}$.

Assume there are N_{cls} clusters totally, $\mathbf{M}_{\text{tsp}}$ corresponds to a $N_{\text{cls}}+1$ dimensions square matrix. The major part is the $N_{\text{cls}} \times N_{\text{cls}}$ block composed of the connection cost between each pair of frontier clusters, which is computed as:

$$\mathbf{M}_{\text{tsp}}(k_1, k_2) = \mathbf{M}_{\text{tsp}}(k_2, k_1)$$
$$= t_{\text{lb}}(\mathbf{x}_{k_1,1}, \mathbf{x}_{k_2,1}),\ k_1, k_2 \in \{1, 2, \ldots, N_{\text{cls}}\} \tag{2}$$

As mentioned in Sect.IV-C, this information is maintained when frontiers are detected. Thus, the $N_{\text{cls}} \times N_{\text{cls}}$ block can be filled without extra overhead.

The first row and column of $\mathbf{M}_{\text{tsp}}$ are associated with the current viewpoint $\mathbf{x}_0 = (\mathbf{p}_0, \xi_0)$ and N_{cls} clusters. Starting from $\mathbf{x}_0$, the cost to the k-th cluster is evaluated by:

$$\mathbf{M}_{\text{tsp}}(0, k) = t_{\text{lb}}(\mathbf{x}_0, \mathbf{x}_{k,1}) + w_{\text{c}} \cdot c_{\text{c}}(\mathbf{x}_{k,1}),$$
$$k \in \{1, 2, \ldots, N_{\text{cls}}\} \tag{3}$$

here a motion consistency cost $c_{\text{c}}(\mathbf{x}_{k,1})$ is introduced, which is generally computed as:

$$c_{\text{c}}(\mathbf{x}_{k,j}) = \cos^{-1} \frac{(\mathbf{p}_{k,j} - \mathbf{p}_0) \cdot \mathbf{v}_0}{\|\mathbf{p}_{k,j} - \mathbf{p}_0\| \, \|\mathbf{v}_0\|} \tag{4}$$

where $\mathbf{v}_0$ is the current velocity. In some cases, multiple tours have comparable time lower bound, so back-and-forth maneuvers may be generated in successive planning steps and slow down the progress. We eliminate this inconsistency with $c_{\text{c}}(\mathbf{x}_{k,1})$, which penalizes large changes in flight direction.

Our problem is different from standard TSP whose solution is a closed-loop tour. However, we can reduce it to an ATSP by assigning zero connection costs from other clusters to the current viewpoint:

$$\mathbf{M}_{\text{tsp}}(k, 0) = 0,\ k \in \{0, 1, 2, \ldots, N_{\text{cls}}\} \tag{5}$$

Authorized licensed use limited to: Harbin Engineering Univ Library. Downloaded on April 17,2023 at 06:31:46 UTC from IEEE Xplore. Restrictions apply.

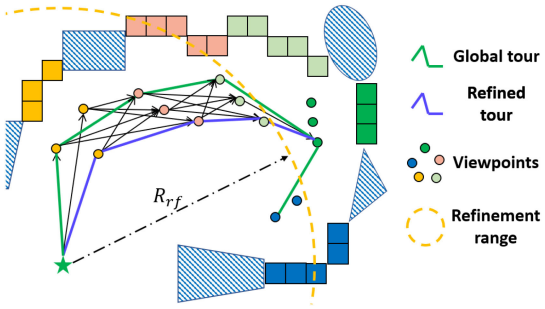


Fig. 6. Refining viewpoints locally using the graph search approach. Along a truncated segment of the global tour, multiple viewpoints of each visited cluster are considered to select the optimal set of viewpoints.

In this way, going back to the current viewpoint in any closed-loop tours contributes no extra cost, so each closed-loop tour always contains an open-loop one with an identical cost. As a result, we can obtain the optimal open-loop tour by finding the optimal closed-loop one and retrieving its equal-cost open-loop tour.

B. Local Viewpoint Refinement

The global tour planning finds a promising order to visit all clusters. Nonetheless, it involves only a single viewpoint of each cluster, which are not necessarily the best combination among all viewpoints.

To this end, a richer set of viewpoints on a truncated segment of the global tour are considered to further improve the exploration rate, by using a graph search approach, as depicted in Fig. 6. We found consecutive clusters whose viewpoints on the global tour are closer than R_{rf} to the current position. To simplify the notation, suppose that $F_i, 1 \leq i \leq N_{\text{rf}}$ are the considered clusters. We create graph nodes for their viewpoints VP_i and the current viewpoint $\mathbf{x}_0$. Then each node is connected to other nodes associated with the next cluster with a directed edge, which compose a directed acyclic graph capturing possible variation of the truncated tour. We utilize the Dijkstra algorithm to search for the optimal local tour $\Xi = \{\mathbf{x}_{1,j_1}, \mathbf{x}_{2,j_2}, \ldots, \mathbf{x}_{N_{\text{rf}},j_{N_{\text{rf}}}}\}$ that minimizes the cost:

$$\begin{aligned} c_{\text{rf}}(\Xi) &= t_{\text{lb}}(\mathbf{x}_0, \mathbf{x}_{1,j_1}) + w_{\text{c}} \cdot c_{\text{c}}(\mathbf{x}_{1,j_1}) \\ &\quad + t_{\text{lb}}(\mathbf{x}_{N_{\text{rf}},j_{N_{\text{rf}}}}, \mathbf{x}_{N_{\text{rf}}+1,1}) + \sum_{k=1}^{N_{\text{rf}}-1} t_{\text{lb}}(\mathbf{x}_{k,j_k}, \mathbf{x}_{k+1,j_{k+1}}) \end{aligned} \tag{6}$$

which also consists of time lower bounds and motion consistency. Note that it is straight forward to incorporate information gain [4], [12] to Equ.6, however, evaluating information gain for numerous viewpoints is expensive. Practically, we find that simply adopting viewpoints based on their coverages is much faster and leads to consistently satisfactory results.

C. Minimum-Time B-Spline Trajectory

Given the discrete viewpoints, continuous trajectories are required for smooth navigation. Our quadrotor trajectory planning is based on a method [28] that generates smooth, safe and dynamically feasible B-spline trajectories. We go one step further to optimize all parameters of B-splines, so that the total trajectory time is minimized to enable the quadrotor to fully utilize its dynamic capability.

As the quadrotor dynamics are differentially flat [19], we plan trajectories for the flat outputs $\mathbf{x} \in (x, y, z, \xi)$. Let $\mathbf{X}_{\text{cb}} = \{\mathbf{x}_{\text{c},0}, \mathbf{x}_{\text{c},1}, \ldots, \mathbf{x}_{\text{c},N_b}\}$ where $\mathbf{x}_{\text{c},i} = (\mathbf{p}_{\text{c},i}, \xi_{\text{c},i})$ be the $N_b + 1$ control points of a p_b degree uniform B-spline, and Δt_b be the knot span. We find the B-spline that trades-off smoothness and total trajectory time, and satisfies safety, dynamic feasibility and boundary state constraints. It can be formulated as an following optimization problem:

$$\underset{\mathbf{X}_{\text{c},b}, \Delta t_b}{\arg\min}\ f_{\text{s}} + w_{\text{t}} T + \lambda_{\text{c}} f_c + \lambda_{\text{d}} \left(f_{\text{v}} + f_{\text{a}}\right) + \lambda_{\text{bs}} f_{\text{bs}} \tag{7}$$

Similar to [28], f_{s} is the elastic band smoothness cost:

$$f_{\text{s}} = \sum_{i=0}^{N_b-2} \mathbf{s}_i^{\text{T}} \mathbf{R}_{\text{s}} \mathbf{s}_i,\ \mathbf{s}_i = \mathbf{x}_{\text{c},i+2} - 2\mathbf{x}_{\text{c},i+1} + \mathbf{x}_{\text{c},i} \tag{8}$$

in which $\mathbf{R}_{\text{s}}$ is the penalty matrix:

$$\mathbf{R}_{\text{s}} = \begin{bmatrix} w_{\text{s,p}} \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0}^{\text{T}} & w_{\text{s},\xi} \end{bmatrix} \tag{9}$$

T is the total trajectory time depending on Δt_b and the number of B-spline segments:

$$T = (N_b + 1 - p_b) \cdot \Delta t_b \tag{10}$$

f_{c}, f_{v} and f_{a} are the penalties to ensure safety and dynamic feasibility. Given the following function:

$$\mathcal{P}(\tau_1, \tau_2) = \begin{cases} (\tau_1 - \tau_2)^2 & \tau_1 \leq \tau_2 \\ 0 & \text{else} \end{cases} \tag{11}$$

f_{c} is evaluated as:

$$f_{\text{c}} = \sum_{i=0}^{N_b} \mathcal{P}\left(d\left(\mathbf{p}_{\text{c},i}\right), d_{\min}\right) \tag{12}$$

where $d(\mathbf{p}_{\text{c},i})$ is the distance of point $\mathbf{p}_{\text{c},i}$ to the nearest obstacle, which can be obtained from the Euclidean signed distance field (ESDF) maintained by the mapping module. Practically a clearance larger than our quadrotor's radius (typically $d_{\min} \geq$ 0.5 m) ensures safety in complex scenes. f_{v} and f_{a} penalize infeasible velocity and acceleration:

$$f_{\text{v}} = \sum_{i=0}^{N_b-1} \left\{ \sum_{\mu \in \{x,y,z\}} \mathcal{P}(v_{\max}, |\dot{p}_{\text{c},i,\mu}|) + \mathcal{P}(\dot{\xi}_{\max}, |\dot{\xi}_{\text{c},i}|) \right\} \tag{13}$$

$$f_{\text{a}} = \sum_{i=0}^{N_b-2} \left\{ \sum_{\mu \in \{x,y,z\}} \mathcal{P}(a_{\max}, |\ddot{p}_{\text{c},i,\mu}|) + \mathcal{P}(\ddot{\xi}_{\max}, |\ddot{\xi}_{\text{c},i}|) \right\} \tag{14}$$

in which the control points of derivatives are utilized:

$$\dot{\mathbf{x}}_{\text{c},i} = \left[\dot{p}_{\text{c},i,x}, \dot{p}_{\text{c},i,y}, \dot{p}_{\text{c},i,z}, \dot{\xi}_{\text{c},i}\right]^{\text{T}} = \frac{\mathbf{x}_{\text{c},i+1} - \mathbf{x}_{\text{c},i}}{\Delta t_b} \tag{15}$$

$$\ddot{\mathbf{x}}_{\text{c},i} = \left[\ddot{p}_{\text{c},i,x}, \ddot{p}_{\text{c},i,y}, \ddot{p}_{\text{c},i,z}, \ddot{\xi}_{\text{c},i}\right]^{\text{T}} = \frac{\mathbf{x}_{\text{c},i+2} - 2\mathbf{x}_{\text{c},i+1} + \mathbf{x}_{\text{c},i}}{\Delta t_b^2} \tag{16}$$

In Equ.12, 13 and 14, the convex hull property of B-spline is utilized to ensure the feasibility efficiently. For brevity we refer the reader to [28] for more details.

Authorized licensed use limited to: Harbin Engineering Univ Library. Downloaded on April 17,2023 at 06:31:46 UTC from IEEE Xplore. Restrictions apply.

Lastly, we set the 0th to 2nd order derivatives at the start to the instantaneous state $(\mathbf{x}_0, \dot{\mathbf{x}}_0, \ddot{\mathbf{x}}_0)$ for smooth motion. The 0-th order derivative at the end is also determined by the viewpoint $\mathbf{x}_{\text{next}}$ to be visited. In implementation we use cubic B-splines, so the associated cost is:

$$f_{\text{bs}} = \left\| \frac{\mathbf{x}_{\text{c},0} + 4\mathbf{x}_{\text{c},1} + \mathbf{x}_{\text{c},2}}{6} - \mathbf{x}_0 \right\|^2 + \left\| \frac{\dot{\mathbf{x}}_{\text{c},0} + \dot{\mathbf{x}}_{\text{c},1}}{2} - \dot{\mathbf{x}}_0 \right\|^2 + \|\ddot{\mathbf{x}}_{\text{c},0} - \ddot{\mathbf{x}}_0\|^2 + \left\| \frac{\mathbf{x}_{\text{c},N_b-2} + 4\mathbf{x}_{\text{c},N_b-1} + \mathbf{x}_{\text{c},N_b}}{6} - \mathbf{x}_{\text{next}} \right\|^2 \tag{17}$$

VI. Results

A. Implementation Details

We set $w_{\text{c}} = 1.5$ in Equ.3 and 6. In global tour planning, the ATSP is solved using a Lin-Kernighan-Helsgaun heuristic solver [33]. In local viewpoint refinement, we reserve at most $N_{\text{view}} = 15$ viewpoints in each FIS and truncate the global tour within radius $R_{\text{rf}} = 5.0$. For trajectory optimization, we use $w_{\text{s,p}} = 5.0, w_{\text{s},\xi} = 2.5, w_{\text{t}} = 1.0, \lambda_{\text{c}} = \lambda_{\text{bs}} = 10.0, \lambda_{\text{d}} = 2.0$ and $d_{\min} = 0.4$ and solve the problem with a general non-linear optimization solver NLopt.[2] Cubic B-spline ($p_b = 3$) is used as the trajectory representation.

To achieve fast exploration, an efficient mapping framework is essential. In our work, we utilize a volumetric map [34], which has been successfully applied to fast autonomous flights [28], [30] in complex scenes. Similar to [35], which is widely applied in exploration, [34] builds an occupancy grid representation of the space. Meanwhile it also maintains an ESDF incrementally to facilitate the trajectory planning. For brevity we refer interested readers to [34] for more details about our mapping framework.

In all field experiments, we localize the quadrotor by a visual-inertial state estimator [36]. We use a geometric controller [37] for tracking control of the (x, y, z, ξ) trajectory. We equipped our customized quadrotor platform with an Intel RealSense Depth Camera D435i. All the above modules run on an Intel Core i7-8550 U CPU.

B. Benchmark and Analysis

We test our proposed framework in simulation. We benchmark it in a bridge scenario and a large maze scenario. Three methods are compared: the NBVP [12], the classic frontier method [7] and the rapid frontier method [1]. Note that no open source code is available for [1], so we use our implementation. In all tests the dynamic limits are set as $v_{\max} = 2.0$ m/s and $\dot{\xi}_{\max} = 0.9$ rad/s for all methods. The FOVs of the sensors are set as $[80 \times 60]$ deg with a maximum range of 4.5 m. In both scenarios each method is run for 3 times with the same initial configuration. Statistics and exploration progresses of the four methods are shown in Table II and 8 respectively. The computation time of each component of our method is listed in Table III.

1) Bridge Scenario: Firstly, we compare the four methods in a $10 \times 20 \times 5$ m^3 space containing a bridge, as shown in Fig. 7.

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

TABLE II
Exploration Statistic in the Bridge and Large Maze Scenarios

Scene	Method	Exploration time (s)				Flight distance (m)			
		Avg	Std	Max	Min	Avg	Std	Max	Min
Bridge	Classic [7]	575	53	643	511	250	42	285	190
	Rapid [1]	288	15	305	264	286	13	303	269
	NBVP [12]	857	117	1018	740	322	47	377	261
	Proposed	104	1.5	105	102	165	3.8	170	161
Large Maze	Classic [7]	814	104	961	721	419	63	509	373
	Rapid [1]	669	68	766	613	469	32	514	440
	NBVP [12]	1037	152	1253	925	1539	262	1898	1279
	Proposed	168	16	192	156	280	20	310	264

TABLE III
Average Computation Time of Each Proposed Component

Scene	Average computation time (ms)					
	Frontier Sect.IV-B	View.+Cost Sect.IV-C	Global Sect.V-A	Local Sect.V-B	Traj. Sect.V-C	Total
Bridge	4.69	4.86+5.16	1.12	4.10	4.23	24.17
Maze	5.21	6.06+10.97	3.53	4.98	5.47	36.23

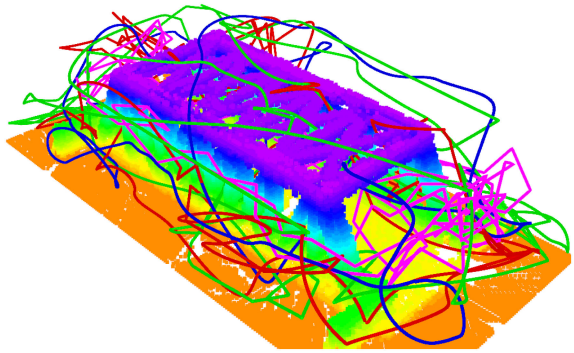


Fig. 7. Benchmark comparison of the proposed method, classic frontier method [7], rapid frontier method [1] and NBVP [12] in a 3D space containing a bridge. The overall exploration paths are shown as blue, red, green and pink respectively.

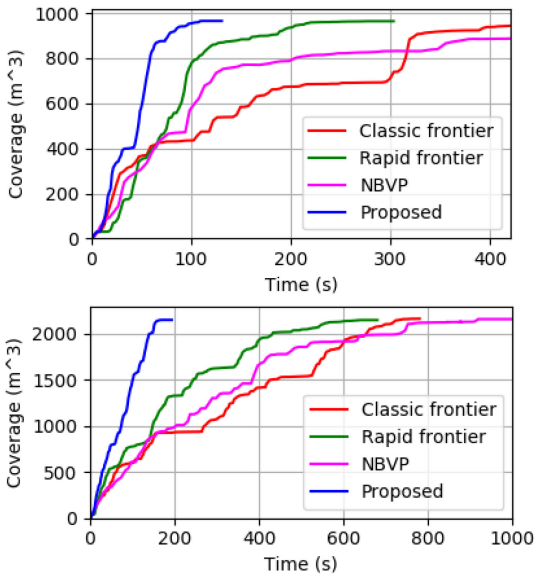


Fig. 8. The exploration progress of four methods in the bridge (top) and large maze (bottom) scenarios.

Authorized licensed use limited to: Harbin Engineering Univ Library. Downloaded on April 17,2023 at 06:31:46 UTC from IEEE Xplore. Restrictions apply.

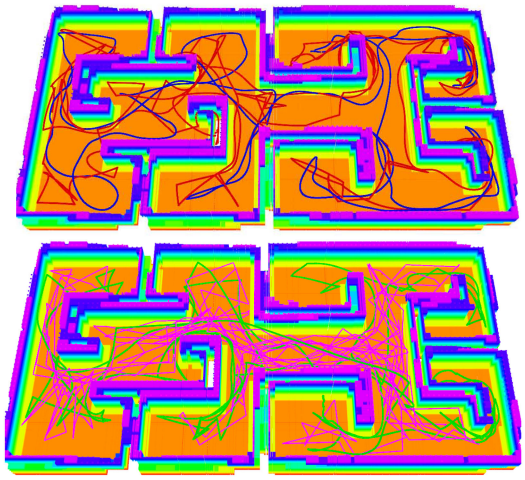


Fig. 9. Path generated by the proposed method (blue), classic frontier method [7](red), rapid frontier method [1] (green) and NBVP [12](pink)

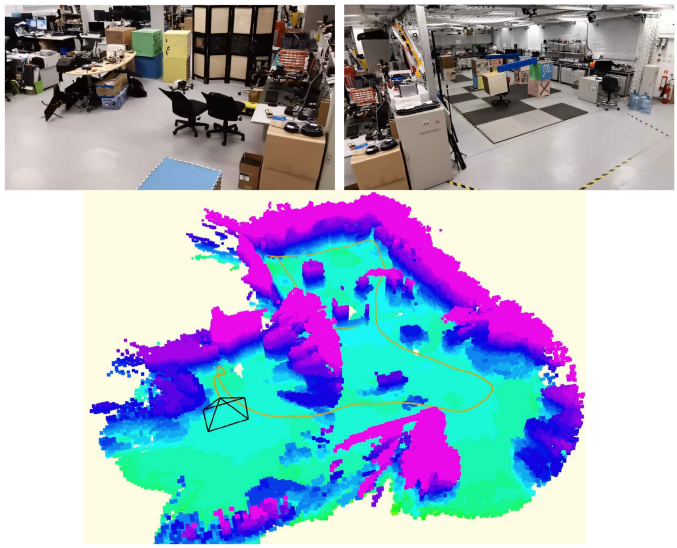


Fig. 10. Experiments in an indoor scene composed of two room: a small one with tables and chairs (top left), a large room cluttered with obstacles (top right).

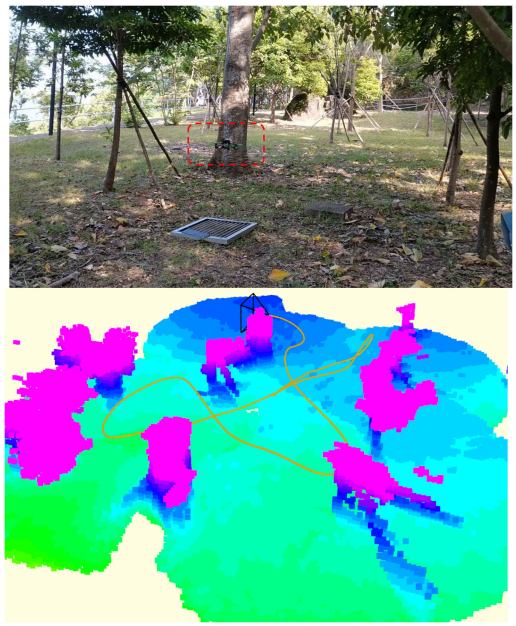


Fig. 11. Exploration experiment conducted in a forest.

The result indicates that we achieve much shorter exploration time and smaller time variance. The overall exploration path of our method is significantly shorter, primarily because we plan tours globally. The executed path is smoother, since we refine motions locally and generate smooth trajectories. Also, we are able to navigate at a higher flight speed, owing to the minimum-time trajectory planning.

2) Large Maze Scenario: We also compare the methods quantitatively in a large maze environment shown in Fig. 9. The explored space is $20 \times 80 \times 3$ m^3 large. In this scenario, all the benchmarked methods take a long time to reach full coverage, due to the complexity of the scene. In contrast, our method completes the exploration 4+ times faster on average. Path executed by the four methods after completion are displayed in Fig. 9. Noticeably, our method explores the maze in a more sensible order, without revisiting the same place frequently. In consequence, it produces a much shorter coverage path and an approximately linear exploration rate (Fig. 8). This behavior is owing to the global plan, without which known regions may be revisited many times and slow down the progress, as the benchmarked methods do. Also note that the computation time in the large maze is longer, mostly due to the larger scene, which naturally involves a greater number of frontier clusters.

C. Field Exploration Tests

To further validate the proposed method, we conduct extensive field experiments in both indoor and outdoor environments. In all tests we set the dynamics limits as $v_{\max} = 1.5$ m/s, $a_{\max} = 0.8$ m/s and $\dot{\xi}_{\max} = 0.9$ rad/s. Note that we do not use any external device for localization and only rely on the onboard state estimator.

First, we present fast exploration tests in two indoor scenes. The first scene is shown in Fig. 1, within which we deploy dozens of obstacles and the quadrotor should perform 3D maneuvers to map the unknown space and avoid obstacles simultaneously. We bound the space to be explored with a $10 \times 6 \times 2$ m^3 box. One sample map and the flight trajectory is presented in Fig. 1. The second indoor scene is a larger environment including two rooms, where one room is similar to scene 1 and the other is a part of an office containing tables and chairs. The space is bounded by a $15 \times 11 \times 2$ m^3 box. The quadrotor starts by exploring the large room, after which it proceeds to the small one. The second scene, the online generated map and trajectory are shown in Fig. 10. Note that in the two scenes the quadrotor starts out at a spot with low visibility, so it only maps a small region of the environment at the beginning. Finally, to validate our method in natural environments, we conduct exploration tests in a forest. The size of the area to explore is $11 \times 10 \times 2$ m^3. The experiment environment and the associated results are displayed Fig. 11.

The above experiments demonstrate the capability of our method in complex real-world scenarios. They also show the merits of our autonomous quadrotor system, among which the

Authorized licensed use limited to: Harbin Engineering Univ Library. Downloaded on April 17,2023 at 06:31:46 UTC from IEEE Xplore. Restrictions apply.

state estimation [36] and mapping modules [34] are also crucial to fullfil the real-world tasks. Video demonstration of all experiments is available (Fig. 1), we refer the readers to it for more details.

VII. Conclusions

In this paper, we propose a hierarchical framework for rapid autonomous quadrotor exploration. An incrementally maintained FIS is introduced to provide the exploration planning with essential information. Based on FIS, a hierarchical planner plans exploration motions in three sequential steps, which finds efficient global tours, selects a local set of optimal viewpoints, and generates minimum-time local trajectories. The method makes decisions at high frequency to respond quickly to environmental changes. Both benchmark and real-world tests show the competence of our method.

One limitation of our method is assuming perfect state estimation, as most methods do. We evaluate our method in simulation with ground truth localization, while drifts in pose are not considered. However, error in state estimation exists generally and can not be ignored. In the future we plan to consider the state estimation uncertainty in our method and evaluate its performance under pose drifts.

References

[1] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 2135–2142.

[2] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto, "An efficient sampling-based method for online informative path planning in unknown environments," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1500–1507, Apr. 2020.

[3] Z. Meng *et al.*, "A two-stage optimized next-view planning framework for 3-d unknown environment exploration, and structural reconstruction," *IEEE Robot. Automat. Lett.*, vol. 2, no. 3, pp. 1680–1687, Jul. 2017.

[4] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient autonomous exploration planning of large-scale 3D environments," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 1699–1706, Apr. 2019.

[5] M. Dharmadhikari *et al.*, "Motion primitives-based path planning for fast and agile exploration using aerial robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 179–185.

[6] S. Song and S. Jo, "Online inspection path planning for autonomous 3D modeling using a micro-aerial vehicle," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 6217–6224.

[7] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Automat. CIRA'97. New Comput. Princ. Robot. Automat.*. 1997, pp. 146–151.

[8] M. Julià, A. Gil, and O. Reinoso, "A comparison of path planning strategies for autonomous exploration and mapping of unknown environments," *Auton. Robots*, vol. 33, no. 4, pp. 427–444, 2012.

[9] S. Shen, N. Michael, and V. Kumar, "Stochastic differential equation-based exploration algorithm for autonomous indoor 3 d exploration with a micro-aerial vehicle," *Int. J. Robot. Res.*, vol. 31, no. 12, pp. 1431–1444, 2012.

[10] D. Deng, R. Duan, J. Liu, K. Sheng, and K. Shimada, "Robotic exploration of unknown 2 d environment using a frontier-based automatic-differentiable information gain measure," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics*, 2020, pp. 1497–1503.

[11] C. Connolly, "The determination of next best views," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1985, vol. 2, pp. 432–435.

[12] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon" next-best-view" planner for 3 d exploration," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 1462–1468.

[13] C. Papachristos, S. Khattak, and K. Alexis, "Uncertainty-aware receding horizon exploration and mapping using aerial robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 4568–4575.

[14] T. Dang, C. Papachristos, and K. Alexis, "Visual saliency-aware receding horizon autonomous exploration with application to aerial robotics," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 2526–2533.

[15] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon path planning for 3D exploration and surface inspection," *Auton. Robots*, vol. 42, no. 2, pp. 291–306, 2018.

[16] C. Witting, M. Fehr, R. Bähnemann, H. Oleynikova, and R. Siegwart, "History-aware autonomous exploration in confined environments using mavs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–9.

[17] C. Wang, D. Zhu, T. Li, M. Q.-H. Meng, and C. W. de Silva, "Efficient autonomous robotic exploration with semantic road map in indoor environments," *IEEE Robot. Automat. Lett.*, vol. 4, no. 3, pp. 2989–2996, Jul. 2019.

[18] B. Charrow *et al.*, "Information-theoretic planning with trajectory optimization for dense 3D mapping," in *Proc. Robot.: Sci. Syst.*, vol. 11, 2015.

[19] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 2520–2525.

[20] C. Richter, A. Bry, and N. Roy, "Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments," in *Proc. Int. Symp. Robot. Res.*, Dec. 2013, pp. 649–666.

[21] J. Chen, T. Liu, and S. Shen, "Online generation of collision-free trajectories for quadrotor flight in unknown cluttered environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2016, pp. 1476–1483.

[22] F. Gao, W. Wu, Y. Lin, and S. Shen, "Online safe trajectory generation for quadrotors using fast marching method and bernstein basis polynomial," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2018, pp. 344–351.

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

[24] J. Tordesillas, B. T. Lopez, and J. P. How, "FASTER: Fast and safe trajectory planner for flights in unknown environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 1934–1940.

[25] H. Oleynikova, M. Burri, Z. Taylor, J. Nieto, R. Siegwart, and E. Galceran, "Continuous-time trajectory optimization for online UAV replanning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2016, pp. 5332–5339.

[26] F. Gao, Y. Lin, and S. Shen, "Gradient-based online safe trajectory generation for quadrotor flight in complex environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 3681–3688.

[27] V. Usenko, L. von Stumberg, A. Pangercic, and D. Cremers, "Real-time trajectory replanning for mavs using uniform b-splines and a 3D circular buffer," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 215–222.

[28] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robot. Automat. Lett.*, vol. 4, no. 4, pp. 3529–3536, Oct. 2019.

[29] B. Zhou, F. Gao, J. Pan, and S. Shen, "Robust real-time uav replanning using guided gradient-based optimization and topological paths," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2020, pp. 1208–1214.

[30] B. Zhou, J. Pan, F. Gao, and S. Shen, "Raptor: Robust and perception-aware trajectory replanning for quadrotor fast flight," 2020, *arXiv:2007.03465*.

[31] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 489–494.

[32] C. Ericson, *Real-Time Collision Detection*. Boca Raton, FL, USA: CRC Press, 2004.

[33] K. Helsgaun, "An effective implementation of the lin-kernighan traveling salesman heuristic," *Eur. J. Oper. Res.*, vol. 126, no. 1, pp. 106–130, 2000.

[34] L. Han, F. Gao, B. Zhou, and S. Shen, "FIESTA: Fast incremental euclidean distance fields for online motion planning of aerial robots," 2019, *arXiv:1903.02144*.

[35] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: A probabilistic, flexible, and compact 3 d map representation for robotic systems," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2, May 2010.

[36] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[37] T. Lee, M. Leoky, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on se (3)," in *Proc. IEEE Control Decis. Conf.*, Dec. 2010, pp. 5420–5425.

Authorized licensed use limited to: Harbin Engineering Univ Library. Downloaded on April 17,2023 at 06:31:46 UTC from IEEE Xplore. Restrictions apply.