Autonomous Exploration Method for Fast Unknown Environment Mapping by Using UAV Equipped With Limited FOV Sensor

Yinghao Zhao, Li Yan, *Member, IEEE*, Hong Xie, Jicheng Dai, and Pengcheng Wei

***Abstract*—Autonomous exploration is crucial and highly challenging for various intelligent operations of unmanned aerial vehicles. However, low-efficiency problems caused by low-quality trajectory or back-and-forth maneuvers (BFMs) still exist. To improve the exploration efficiency in unknown environments, a fast autonomous exploration planner is proposed in this article. We first design a novel frontier exploration sequence generation method to significantly reduce BFM during the exploration by considering multilevel factors in an asymmetric traveling salesman problem. Then, according to the exploration sequence and the distribution of frontiers, an adaptive yaw planning method is proposed to cover more frontiers by yaw change during an exploration journey. In addition, to increase the fluency of flight, a dynamic replanning strategy is also adopted. The main technical contributions of the proposed approach are to provide a comprehensive global coverage path with less BFM and use adaptive yaw planning to generate more reasonable yaw trajectory. We present extensive comparison and real-world experiments to validate the effectiveness and correctness of our method. Experimental results show that the proposed approach has better performance compared to typical and state-of-the-art methods in these flat scenarios without lots of scattered obstacles.**

***Index Terms*—Adaptive yaw planning, autonomous exploration, exploration sequence, mapping.**

I. Introduction

AN INTELLIGENT robot has always been the goal pursued in lots of fields, and many application cases have emerged through the efforts of many researchers [1]. Unmanned aerial vehicles (UAVs) as a data collection and material transportation platform with its unique advantages have been widely used in the field of surveying and mapping [2], environmental protection [3], rescue [4], military [5], and other fields in recent years. However, in most operation scenarios, it is still in the state of human operation, and autonomous operation ability still lacks. As one of the key parts of UAV autonomous mapping capability, autonomous exploration has attracted extensive attention, and many excellent autonomous exploration algorithms have emerged [6], [7], [8], [9].

A. Existing Problems

Although the existing exploration methods can explore environments by using frontiers or sampling viewpoints, there are still many problems to be solved [6], [7], [10], [11], [12]. The methods of sampling viewpoints can easily generate the candidate goals, but they always cause a low exploration rate and efficiency [6]. Most of them are using the greedy strategy, which pays attention to the local information gain but ignores the global efficiency. Selin et al. [10] improve the efficiency by using a frontier-based strategy to guide the "next-best-view" planning. However, they do not consider the dynamics of the UAV, which will cause unsmooth exploration trajectory, low-speed flight, and lots of stop-and-go maneuvers. Although the methods [7], [11] using frontiers can quickly explore the whole environment by searching frontiers and generating an exploration sequence, the process of finding and describing frontiers is always computationally expensive. FUEL [12] is a state-of-the-art fast and autonomous exploration algorithm. Its heuristic framework can achieve rapid and efficient UAV exploration through the designed frontier information structures (FISs) and hierarchical planning. It can generate a smooth and high-speed exploration trajectory in high frequency. However, although this algorithm has greatly improved the exploration rate and exploration efficiency compared with other algorithms, it still faces problems affecting its exploration efficiency, such as back-and-forth maneuvers during the exploration process.

B. Contributions

To reduce the back-and-forth maneuvers that cause inefficient exploration mentioned in Section I-A, based on the framework of FUEL, this article proposes a novel fast autonomous exploration planner (FAEP). To improve the rationality of the exploration trajectory and the efficiency of exploration, we design a comprehensive exploration sequence generation method for global

Manuscript received 31 July 2022; revised 21 November 2022, 4 January 2023, 13 March 2023, and 16 April 2023; accepted 5 June 2023. Date of publication 19 June 2023; date of current version 18 December 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFD1100200 and in part by the Science and Technology Major Project of Hubei Province under Grant 2021AAA010. *(Corresponding authors: Li Yan; Hong Xie.)*

The authors are with the School of Geodesy and Geomatics, Wuhan University, Wuhan 430079, China (e-mail: zhaoyinghao@whu.edu.cn; lyan@sgg.whu.edu.cn; hxie@sgg.whu.edu.cn; daijicheng@whu.edu.cn; wei.pc@whu.edu.cn).

We release our implementation as an open-source package (code is available at https://github.com/Zyhlibrary/FAEP) for the community.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIE.2023.3285921.

Digital Object Identifier 10.1109/TIE.2023.3285921

coverage planning, which not only considers the flight-level factors but also innovatively considers the frontier-level factors (spatial features of frontiers) to reduce the back-and-forth exploration maneuvers in various simulation scenarios and typical complex real-world environments designed in this article. According to the spatial features, we evaluate the potential possibility of the frontier causing back-and-forth maneuvers and get a more reasonable exploration sequence based on the possibility. After determining the local exploration target, an adaptive yaw planning strategy is designed to achieve efficient exploration by yaw change during flight, which contains two yaw planning modes: normal planning mode and two-stage planning mode. The appropriate mode can be selected to cover more areas according to the distribution of frontiers and its own motion state. In addition, to improve the stability and speed of the flight, a dynamic replanning is also adopted.

We compare our method with three typical and state-of-the-art methods in different simulation environments. The experimental results show that our method achieves a better performance than that of the other three methods. Compared to the two typical methods, i.e., "next-best-view" planning (NBVP) and autonomous exploration planner (Aeplanner), the exploration process of our method is three to seven times faster. Compared with the state-of-the-art method FUEL, our method reduces the flight time and flight distance by more than 20%. In addition, we also verify the effectiveness of our method through onboard real-world exploration. The contributions of this article are summarized as follows:

1) a comprehensive frontier exploration sequence generation method, which innovatively considers not only flight-level factors but also frontier-level factors to reduce the back-and-forth maneuvers;
2) an adaptive yaw planning strategy, which can generate a smooth yaw trajectory to cover more unknown space when flying to the local target by two yaw planning modes: normal planning mode and two-stage planning mode;
3) sufficient quantitative comparison experiments are conducted in simulation. Real-world experiments are also carried out to validate our method in various environments. More details about the experiments are provided in https://youtu.be/0Y671mEwJ_A.

II. Related Work

The problem of autonomous exploration has been studied by many scholars in recent years, and lots of methods from multiple angles have been published, which are mainly divided into the following three categories: sampling-based exploration, frontier-based exploration, and algorithms based on machine learning, which have emerged recently. This article only discusses the previous two types of algorithms, which have been widely used in various exploration tasks.

A. Sampling-Based Exploration Methods

Sampling-based exploration methods use randomly sampled viewpoints in the free space, which find the next best view by obtaining a path with the highest information gain [13]. A receding horizon NBVP is designed to explore the 3-D environments by considering the information gained over the entire path in [6]. The NBVP is the first method that uses the concept of the next best view for exploration in a receding horizon fashion, and many methods are derived from this method. These methods select the path with the highest information gain in the incrementally rapidly exploring random trees for UAVs to execute. The Aeplanner in [10] combined frontier exploration and the NBVP to avoid getting stuck in large environments not exploring all the regions, and the method also makes the process of estimating potential information gain faster by using cached points from earlier iterations. Incremental sampling and probabilistic roadmap are used in [14] to improve the efficiency of planning. The method [15] uses a combination of sampling- and frontier-based methods to reduce the impact of finding unexplored areas in large scenarios. There are also some two-stage methods [16], [17] to cover the entire environment efficiently by different planning strategies in the global and local map.

B. Frontier-Based Exploration Methods

In contrast, the frontier-based methods are mainly comprised of two processes: finding frontiers (the boundary between known and unknown areas) and solving a sequence problem for a global path to visit frontiers. The first frontier-based exploration method is introduced by Yamauchi [18] to explore a generic 2-D environment, which selects the closest frontier as the next goal. Then, a stochastic-differential-equation-based exploration algorithm [19] is contributed to achieve exploration in 3-D environments. To achieve the high-speed flight, Cieslewski et al. [11] presented a method that extracts frontiers in the field of view (FOV) and selects the frontier minimizing the velocity change. For finding a reasonable frontier exploration sequence, the traveling salesman problem (TSP) is employed in [7]. A wise exploration goal is selected by adopting an information-driven exploration strategy in [20]. However, many methods are facing the problems of inefficient global coverage, conservative flight trajectory, and low decision frequencies. For solving these issues, Zhou et al. [12] achieved fast exploration by adopting an incremental frontier structure and hierarchical planning.

III. Proposed Approach

To improve the efficiency of the exploration mapping, we propose an FAEP. The main operation flow of our planner is illustrated in Fig. 1. First, we perform map maintenance and update frontiers based on the UAV state and the sensor data. Second, the global exploration planning is conducted by considering the flight- and frontier-level factors. Third, based on the result of global planning, the path planning and the adaptive yaw planning that contains two planning modes are performed to generate the high-quality local path. Finally, we will submit the planning results to the flight control for execution, and the above process will be continuously executed according to the replanning strategy.

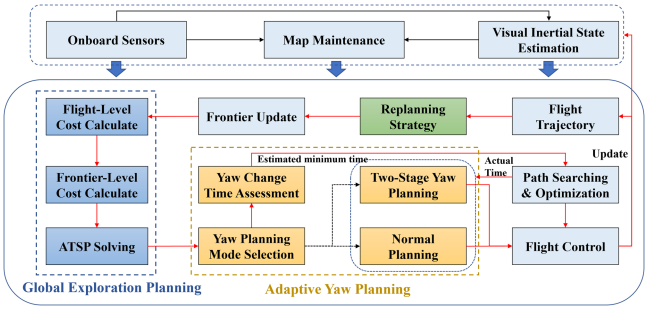

Fig. 1. Overview of the proposed FAEP. The main flow of the planner is shown in the red line. Our main contributions are shown in dark color (blue, yellow, and green).

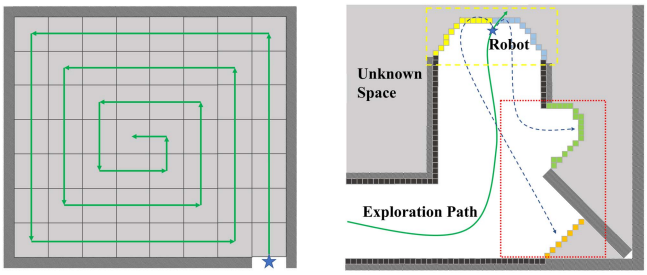

Fig. 2. (a) Diagram of the exploration process by using the spiral filling algorithm. (b) Exploration process without independent small area priority. The yellow box in (b) shows the frontiers being explored. The red box in (b) shows the frontiers that will cause back-and-forth path (the dotted line) in the future.

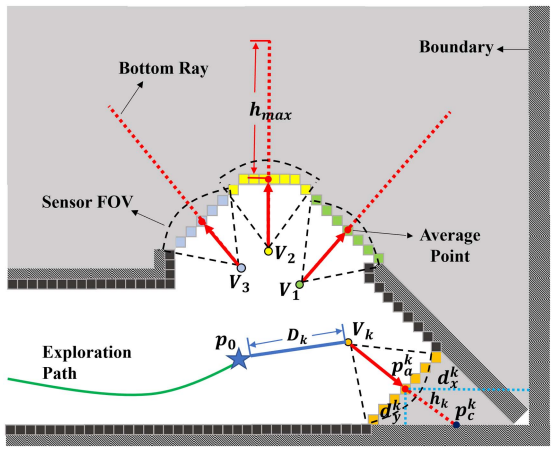

Fig. 3. Diagram of evaluating the spatial features of frontiers. V_1, V_2, V_3, and V_k are the corresponding optimal viewpoints for covering each frontier. According to the average point p_a^k of frontier F_k (orange voxels), we calculate the distance (light blue line) from F_k to the edge of the exploration area. And we use the Bottom Ray (red line, generated by the vector direction of V_k and p_a^k) to roughly evaluate the scope of the unknown area (gray area) behind each frontier (color voxels). Once the ray touches the occupied, free voxel, boundary or maximum distance $h_{\max}$, we regard the distance h_k of ray (red dashed line) as the scope of the frontier.

A. Frontier Exploration Sequence Generation

The frontier exploration sequence is crucial for the frontier-based exploration method, whose rationality directly affects the efficiency of the whole exploration process. Many methods use the TSP to obtain the exploration sequence. However, most methods only take the Euclidean distance between the frontiers as the cost of the TSP, which is simple but insufficient. Although some methods take into account factors, such as yaw change and speed direction to improve the quality of the exploration sequence on the basis of the Euclidean distance, there are still some deficiencies. Different from these methods, first, this article does not use the conventional TSP but uses a more reasonable asymmetric traveling salesman problem (ATSP) for the solution. Second, this article not only considers flight-level factors (Euclidean distance, yaw change, and speed direction change between frontiers and UAV) as the cost but also takes frontier-level factors as terms of the cost function to generate a better global exploration sequence by solving the ATSP.

According to the evaluation criteria of complete coverage path planning problem [21], the fewer the back-and-forth maneuvers, the higher the efficiency. Therefore, when an unknown area needs to be explored quickly, one of the optimal strategies is to use spiral filling algorithms [22]. As shown in Fig. 2(a), the spiral algorithm has low path overlap while satisfying high coverage. In addition, to prevent unnecessary back-and-forth maneuvers, small unknown areas should not be left behind where they have already passed. Otherwise, as shown in Fig. 2(b), there will be lots of areas that will cause the maneuvers. Inspired by this, this article holds that when the frontier is close to the boundary of the exploration area or an independent small area, the corresponding exploration priority should be higher. If this area is not preferentially explored, it will cause back-and-forth maneuvers and reduce the efficiency of global exploration. Therefore, different from other algorithms that only consider the factors of the current flight level, this method also takes the spatial features of the frontier into consideration, which obtains a more reasonable frontier exploration sequence by achieving boundary area higher priority and independent small area higher priority, where the boundary means the edge of the exploration area determined by mission.

For the boundary area to achieve higher priority, we calculate the boundary cost $c_b(k)$ for each frontier F_k in FISs

$$c_b(k) = \begin{cases} d_{\min}(k), & D_k < r_s \\ d_{\min}(k) \cdot \left(1 + w_d \cdot \frac{D_k - r_s}{r_s}\right), & D_k \geq r_s \end{cases} \quad (1)$$

$$d_{\min}(k) = \min\left(d_x^k, d_y^k, d_z^k\right), \quad k \in \{1, 2, \ldots, N_{\text{cls}}\} \quad (2)$$

where N_{cls} represents the number of frontiers. As shown in Fig. 3, d_x^k, d_y^k, and d_z^k are the shortest distance from the frontier to the boundary along the x-axis, y-axis, and z-axis, respectively, and x-axis, y-axis, and z-axis are the same as the initial body frame of the UAV at takeoff. r_s is the maximum range of the sensor, and D_k is the distance between the viewpoint V_k of the frontier F_k and the current position p_0. We directly regard the minimum distance $d_{\min}(k)$ as $c_b(k)$ when the frontier is close to the current position ($D_k < r_s$). The farther the frontier is

from the boundary, the greater the boundary cost $c_b(k)$ is. To raise the priority of the boundary frontiers, we regard $c_b(k)$ as a member of the cost function in the ATSP to obtain a sequence, where the frontiers near the boundary will be explored in higher priority. Meanwhile, to avoid exploring the frontier close to the boundary but far from the current position, we penalize the frontiers by a positive penalty factor w_d when D_k is greater than r_s. In this way, the larger D_k is, the larger the frontier cost is.

For the independent small area to achieve higher priority, a method called Bottom Ray is designed, as shown in Fig. 3. We use the ray (red line) to roughly detect the range of the unknown area behind the frontier, and then, the probability $c_s(k)$ for an independent small area can be evaluated according to the detection result. The main process is as follows. First, we obtain the frontiers whose distance D_k between the viewpoints $V_k(p_k, \xi_k)$ and the current position p_0 of the UAV is less than D_{thr}, where D_{thr} is the distance threshold to avoid raising the priority of too far frontiers. We use the same method in [12] to generate the viewpoint V_k, which contains a position p_k and a yaw angle ξ_k. The method can be divided into three steps: 1) uniformly sampling points in the cylindrical coordinate system whose origin locates at the frontier's center; 2) the points with coverage higher than a threshold are reserved and sorted in descending order of coverage; and 3) selecting the viewpoint that needs less yaw change and flight distance as the final viewpoint. Second, if a frontier is adjacent to the obstacle in both the directions perpendicular to the z-axis at the same time, we believe that the unknown area behind the frontier is an independent area similar to rooms or corridors, which should be explored in higher priority. For this case, we set $c_s(k) = 1$. If not, execute the next step. Third, the vector $\overrightarrow{p_k p_a^k}$ (red line in Fig. 3) from the viewpoint V_k to the average point p_a^k of F_k is calculated, and we extend it by mapping resolution until it touches the occupied, free voxel, boundary or exceeds the maximum distance $h_{\max}$; then, a bottom point p_c^k and the distance h_k between p_a^k and p_c^k (red dashed line in Fig. 3) are obtained. We calculate the independent small area probability $c_s(k)$ for these frontiers by

$$c_s(k) = \frac{h_{\max} - h_k}{h_{\max}}. \tag{3}$$

Finally, we regard $c_s(k)$ and $c_b(k)$ as the factors of frontier level and integrate them with flight-level factors used in [12] into the cost matrix M_{tsp} of the ATSP as follows:

$$\begin{aligned} M_{\text{tsp}}(0,k) &= t_{lb}\left(V_0,\ V_k\right) + w_c \cdot c_c\left(V_k\right) \\ &\quad + w_b \cdot c_b(k) - w_f \cdot c_s(k) \end{aligned} \tag{4}$$

$$\begin{aligned} t_{lb}\left(V_0,\ V_k\right) = \max\Bigg\{ &\frac{\text{length}\left(p_0, p_k\right)}{v_{\max}} \\ &\frac{\min\left(\left|\xi_0 - \xi_k\right|, 2\pi - \left|\xi_0 - \xi_k\right|\right)}{\dot{\xi}_{\max}} \Bigg\} \end{aligned} \tag{5}$$

$$c_c\left(V_k\right) = \cos^{-1}\frac{\left(p_k - p_0\right)\cdot v_0}{\left\|p_k - p_0\right\|\left\|v_0\right\|} \tag{6}$$

where V_0 indicates the current state of UAV, which contains the current position p_0 and yaw angle ξ_0. V_k represents the viewpoint of frontier F_k. v_0 is the current speed of UAV. $t_{lb}(V_0, V_k)$ and $c_c(V_k)$ represent the flight-level factors, such as distance, yaw change, and speed change. w_c, w_b, and w_f are the corresponding weights of the three cost terms, respectively. According to (4), the frontier with the minimum cost should be the frontier that has high probability of causing the back-and-forth maneuvers, and it should be the best frontier that needs to be explored promptly to reduce the low-efficiency exploration in the later stage. The calculation method of the rest of M_{tsp} is as follows:

$$\begin{aligned} M_{\text{tsp}}\left(k_1, k_2\right) &= M_{\text{tsp}}\left(k_2, k_1\right) \\ &= t_{lb}\left(V_{k_1}, V_{k_2}\right), \quad k_1, k_2 \in \{1, 2, \ldots, N_{\text{cls}}\} \end{aligned} \tag{7}$$

$$M_{\text{tsp}}(k, 0) = 0, \quad k \in \{0, 1, 2, \ldots, N_{\text{cls}}\}. \tag{8}$$

B. Adaptive Yaw Planning

According to the frontier exploration sequence generation method that considers the flight- and frontier-level factors described in Section III-A, the optimal local frontier target and corresponding target viewpoint can be determined, and yaw planning can be conducted to cover the target frontier by changing current yaw to the yaw of target viewpoint. Through a large number of experiments, we observe that once there is an excellent yaw planning, the UAV can explore more areas to improve the efficiency of exploration by yaw change during the flight to the local target frontier. However, covering more areas may need more time. If the planner keeps using wide range of yaw change to cover more space during the exploration, it will cause the exploration process at extremely slow speed. Therefore, the timing of using this strategy is also important.

Based on the above analysis, this article designs an adaptive yaw planning strategy after determining the local target frontier F_g by Section III-A. Different from other yaw planning methods, two planning modes are adopted: normal yaw planning and two-stage yaw planning. The normal yaw planning is used to plan a smooth yaw trajectory from current yaw ξ_0 to the yaw ξ_e of the target viewpoint (the viewpoint of F_g). The two-stage yaw planning is designed to generate a reasonable yaw trajectory from ξ_0 to ξ_e while passing through a middle yaw, which utilizes a larger range of yaw change to cover more frontiers when the time required for the yaw change is less than or approximately equal to the flight time of the current trajectory. The middle yaw we select is the yaw with the largest yaw change, so most of the local unknown areas can be covered without more stages of the yaw planning. The main process is described in the adaptive yaw planning strategy, where VPs is a set containing all the viewpoints. V_1 and X_0 are the next target viewpoints determined by Section III-A and current motion state, respectively.

At first, owing to the limited range of the sensor, only the area whose distance from the current position p_0 is less than sensor range r_s can be covered. Therefore, we use **ViewpointsInLocal()** to calculate the number of viewpoints V_k whose distance D_k between V_k and current position p_0 is less than the threshold d_{thr} ($d_{\text{thr}} < r_s$), and the two points are intervisible. In addition,

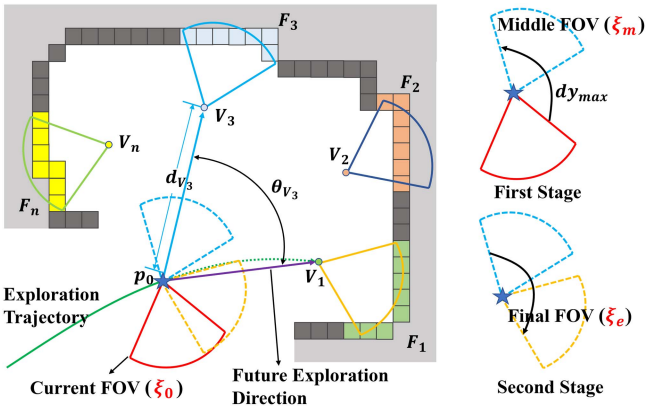

Fig. 4. Schematic of the proposed two-stage yaw planning for the case of multiple viewpoints in the local range. In the first stage, we calculate the middle yaw ξ_m (blue FOV) according to the magnitude of the yaw change, and then, the corresponding yaw planning from the current yaw ξ_0 (red FOV) to the middle yaw is conducted to cover more area during the flight process (green dashed line). In the second stage, the yaw planning from the middle yaw to the final yaw ξ_e (yellow FOV) is conducted to explore the target frontier (green grids).

Algorithm 1: Adaptive Yaw Planning Strategy.

Input: $VPs, V_1(p_e, \xi_e), X_0(V_0, v_0, a_0), D_k, c_s(k)$
Output: yaw Trajectory Y

1: $N_v \leftarrow \mathbf{ViewpointsInLocal}(VPs)$
2: **if** $N_v > 1$ **then**
3: $\xi_m \leftarrow \mathbf{FindMiddleYaw}(N_v)$
4: $T_1, T_2 \leftarrow \mathbf{EstimateMinYawTime}(\xi_0, \xi_m, \xi_e)$
5: $T_{\min} \leftarrow \tau \cdot (T_1 + T_2), R \leftarrow T_1/T_{\min}$
6: **if** $T_{\min} <= D_k/v_0 \,||\, c_s(k) > 0.5$ **then**
7: $T_{\text{real}} \leftarrow \mathbf{PathPlanningAndOpt}(X_0, p_e, T_{\min})$
8: **if** $T_{\text{real}} >= T_{\min}$ **then**
9: $Y_1 \leftarrow \mathbf{YawPlanning}(\xi_0, \xi_m, T_{\text{real}} * R)$
10: $Y_2 \leftarrow \mathbf{YawPlanning}(\xi_m, \xi_e, T_{\text{real}} * (1 - R))$
11: **return** $Y(Y_1, Y_2)$
12: $T_{\min} \leftarrow \mathbf{EstimateMinYawTime}(\xi_0, \xi_e)$
13: $T_{\text{real}} \leftarrow \mathbf{PathPlanningAndOpt}(X_0, p_e, T_{\min})$
14: $Y \leftarrow \mathbf{YawPlanning}(\xi_0, \xi_e, T_{\text{real}})$
15: **return** Y

the angle θ_{V_k} between $\overrightarrow{p_0 V_k}$ and $\overrightarrow{p_0 V_1}$ needs to be less than 90° to ensure the ability of coverage (line 1). Then, if there are viewpoints that meet the above conditions, we adopt the two-stage yaw planning mode (lines 2–11) . Otherwise, the normal yaw planning method is used (lines 12–15). If the two-stage planning is adopted, as shown in Fig. 4, we use **FindMiddleYaw()** to calculate the yaw change between each viewpoint and the current state and find the viewpoint (blue FOV) with the largest yaw change ξ_m that meets the time condition (line 3). Later, according to the geometric relationship between ξ_m, the current yaw ξ_0, the yaw ξ_e of the target frontier, and the max angular velocity $\dot{\xi}_{\max}$, the minimum time $T_{\min}$ required for the two yaw changes is roughly calculated by **EstimateMinYawTime()** as follows:

$$T_1 = \frac{\min\left(|\xi_m - \xi_0|, 2\pi - |\xi_m - \xi_0|\right)}{\dot{\xi}_{\max}} \tag{9}$$

$$T_2 = \frac{\min\left(|\xi_e - \xi_m|, 2\pi - |\xi_e - \xi_m|\right)}{\dot{\xi}_{\max}} \tag{10}$$

$$T_{\min} = \tau \cdot (T_1 + T_2) \tag{11}$$

where τ is the expansion coefficient that can be used to improve the feasibility of yaw change. To avoid the excessive influence of yaw change on flight speed, we only carry out the two-stage yaw planning when the time of yaw change is less than the flight time (line 6). However, once the area is probably a small area judged by Section III-A, the two-stage yaw planning will be used to cover it without hesitation for avoiding possible back-and-forth maneuvers later (line 6). Then, if the above conditions are met, $T_{\min}$, current motion state X_0, and the position p_e of the next target viewpoint will be provided for **PathPlanningAndOpt()** introduced in Section III-C to generate a flight path (line 7), where $T_{\min}$ is regarded as the minimum flight time constraint.

Finally, if the actual flight time T_{real} generated by **PathPlanningAndOpt()** is more than $T_{\min}$, the two-stage yaw planning will be executed by **YawPlanning()** (lines 8–15). In this function, we use an uniform B-spline to represent the trajectory of yaw angle $\phi(t)$, which is parameterized by the $N+1$ control points $\Phi := \{\phi_0, \ldots \phi_n\}$ and knot span δt_ϕ. T is the total time of the trajectory. Owing to the convex hull property of B-spline, the smoothness and dynamic feasibility of the trajectory can be optimized by changing the control points Φ and solving the problem

$$\begin{aligned} \arg\min_{\xi_{cp}} \; & \gamma_1 f_s + \gamma_2 \left(\phi\left(t_0\right) - \xi_0\right) + \gamma_3 \left(\phi(T) - \xi_e\right) \\ & + \gamma_4 \left(f_{\dot{\xi}} + f_{\ddot{\xi}}\right) \end{aligned} \tag{12}$$

where f_s represents the smoothness. The second and third terms are soft waypoint constraint enforcing $\phi(t)$ to pass through current yaw ξ_0 and target yaw ξ_e. The last two terms are the soft constraints for the dynamic feasibility of angular velocity and acceleration. γ_1, γ_2, γ_3, and γ_4 are the corresponding weights of each item. The calculation methods of f_s, $f_{\dot{\xi}}$, and $f_{\ddot{\xi}}$ are similar to [23]. However, owing to the limitations in UAV dynamics feasibility, **PathPlanningAndOpt()** and **YawPlanning()** may fail or T_{real} generated by the part is less than $T_{\min}$ when the time $T_{\min}$ required by the two-stage yaw planning is large. At this moment, the normal yaw planning will be conducted (lines 12–15) to achieve the local planning. Replanning for the whole process will be triggered when the normal yaw planning also fails.

C. Path Searching and Optimization

Another core component of achieving fast exploration is path planning, which is used to generate a safe, smooth, and dynamically feasible trajectory for fast flight. According to the content, path planning can be divided into two parts: path searching and path optimization. In path searching, kinodynamic

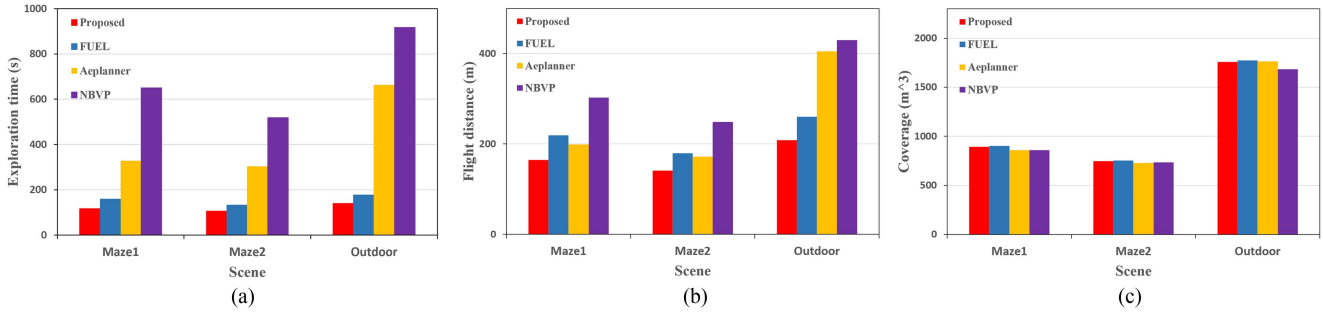

Fig. 5. (a) Average exploration time, (b) average flight distance, and (c) average coverage presented in Table I.

path searching [24] is widely adopted, which can use motion primitives respecting the UAV dynamic to find a collision-free and dynamic feasible path. However, when the UAV is in special scenes, such as searching path from inside to outside in a house or passing through a narrow space, if only the conventional kinodynamic path searching is adopted, the search process will take a relatively long time or even fail due to the discrete control space. To reduce the occurrence of this situation, we adopt our previous work [25] to search the initial path, which use a guiding path to improve the stability of kinodynamic path searching. In path optimization, we optimize the initial path by applying the B-spline trajectory optimization [24]. The method also utilizes the convex hull property of uniform B-splines to generate smooth, safe, and dynamically feasible trajectory by minimizing the cost function.

D. Adaptive Dynamic Replanning

The speed of the target point is usually set to zero by default, and the cost time of each replanning is dynamic and unknown. Therefore, a low-frequency replanning strategy will cause low-speed exploration due to the stop-and-go maneuvers. Besides, if the current position is used as the starting point for planning, the flight will be unstable when the planning process takes a long time due to a long distance between the starting point of the new path and the current position of the UAV. To solve these problems, this article adopts the adaptive dynamic starting point for exploration replanning inspired by Tordesillas et al. [26]. In the ith planning step, we do not use the current position as the starting point of the planning but select the position at the time t_i in the future as the starting point of the planning, and t_i is not constant but determined by the previous planning result

$$t_i = \max\left(\rho \cdot t_{i-1}, t_{\min}\right) \tag{13}$$

where t_i and t_{i-1} represent the cost time of ith and $(i-1)$th planning steps, respectively. $t_{\min}$ is the minimum time for one planning. ρ is the expansion coefficient to improve reliability. If the planning is successful and the actual planning time is less than t_i, update the path after time t_i with the planning result. Otherwise, execute replanning. Besides, to maintain the speed and fluency of the flight, we make a replanning when the duration of the remaining flight path is less than 1 s.

IV. Experimental Results

A. Benchmark Comparisons

In simulation experiments, we compare our method with three state-of-the-art methods in three different environments. The three methods are FUEL [12], Aeplanner [10], and NBVP [6]. For all the methods, we adopt their open-source implementation and default configuration (the maximum number of nodes of Aeplanner and NBVP is set to 200). In each scenario, they are tested in five different initial positions (5 m far from each other), and each method is tested five times at a position. We use the same configurations (shown in Table II) in all the experiments. We test these methods on a computer with Intel Core i9-9900K@ 3.6 GHz, 64-GB memory.

Remark: Although there are many sensors with wide FOV, they are essentially still the limited FOV sensors. Moreover, since we concentrate on the quality of the exploration method, we believe that it would be more convincing that the method could achieve better performance even with the sensor with a narrow observation range, which is more challenging compared with using the ones with a wider observation range.

1) Maze Scenario: Maze is the most effective environment to verify the efficiency of the exploration method. Therefore, we first compare the exploration efficiency in two maze environments: rectangular maze (maze1) and square maze (maze2). The scene range of maze1 is $30 \times 16 \times 2$ m^3, and that of maze2 is $20 \times 20 \times 2$ m^3. The experimental results are shown in Figs. 5 and 6 and Table I. It can be seen that the NBVP takes the longest time and flight distance, and its efficiency is also unstable. The Aeplanner is an improved method of the NBVP; its efficiency is improved by combining the frontier exploration strategy with the NBVP. Owing to the efficient global coverage path and the minimum-time flight path, the proposed method and FUEL have obvious advantages over the above two methods. Not only the flight path is smoother, but also flight time and flight distance are less while ensuring high coverage rate. Meanwhile, owing to the fewer back-and-forth maneuvers and yaw planning strategy, the proposed method achieves more efficient exploration than that of FUEL. Compared with FUEL, the exploration time and flight distance of our method are reduced by 26.8% and 24.8% in maze1 and 20.3% and 21.5% in maze2, respectively, and the exploration ratio tends to be more linear.

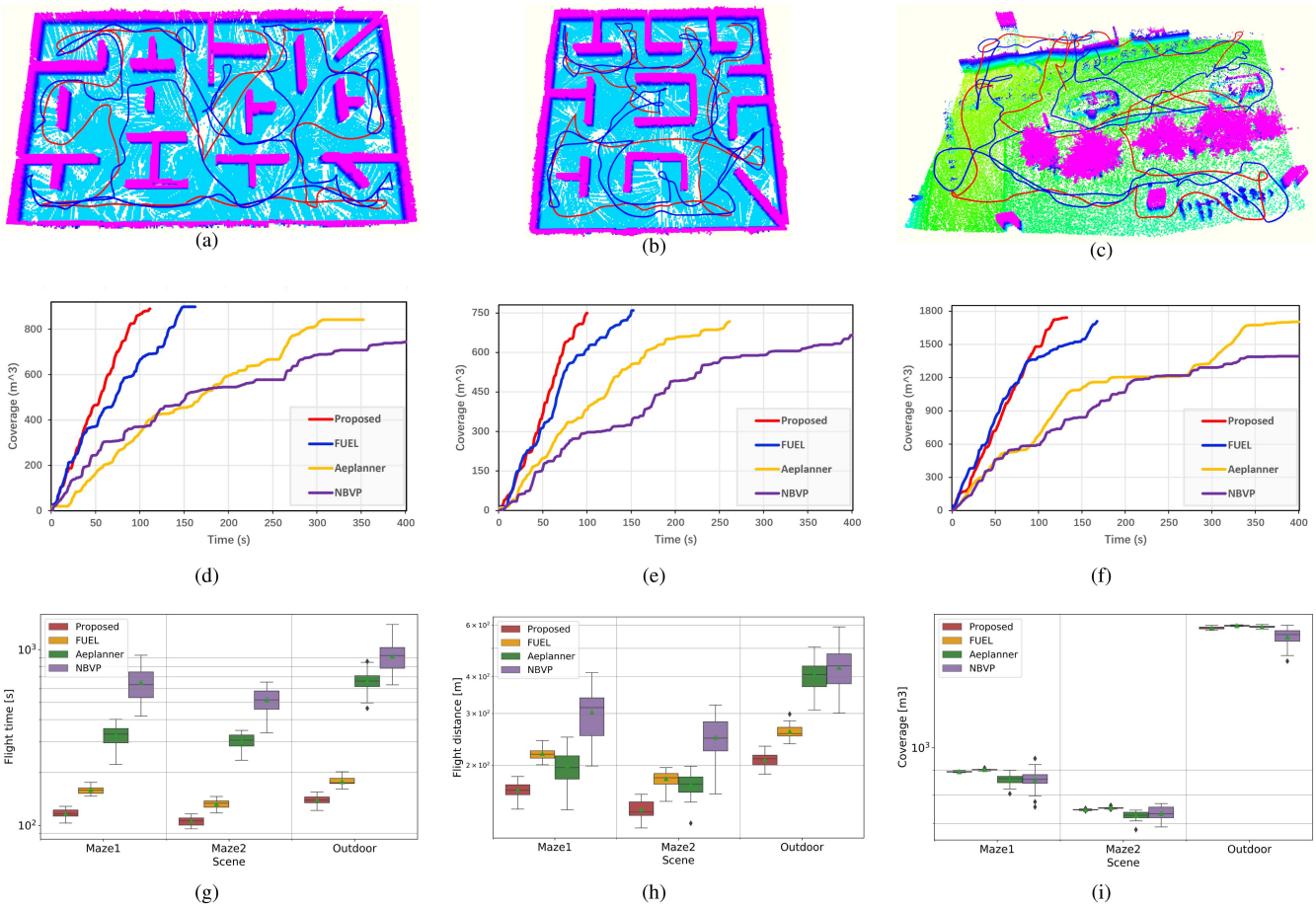

Fig. 6. Benchmark comparison of the experimental results in different scenarios. (a)–(c) Correspond to maze1, maze2, and outdoor scenarios, respectively (the red and blue trajectories in them represent the best performance of the proposed method and FUEL at the same initial position. And we only show the above two methods for clarity). (d)–(f), respectively, illustrate the coverage performance of four comparative experiments in three different scenarios. (g)–(i) show the specific distribution of flight time, flight distance, and coverage, respectively, in the experimental results (presented in Table I) of the four methods in three different scenarios. More comparison can be found in https://github.com/Zyhlibrary/FAEP.

2) Outdoor Scenario: We also compare the four methods in nature scenario to verify the stability and efficiency. The scenario contains trees, cars, fences, and other objects, with a range of $20 \times 30 \times 3\,\mathrm{m}^3$. The experimental results are also shown in Figs. 5 and 6 and Table I. The results show that the exploration time and distance of the four methods increase compared with mazes due to the increase in scene complexity, but the proposed method still has obvious advantages compared with other methods. Compared with the NBVP and the Aeplanner, our method achieves five to seven times faster. Compared with FUEL, our method still maintains the advantages of 21.1% and 20% in exploration time and flight distance, respectively. In addition, we can see that the efficiency of our method and FUEL is basically the same or FUEL achieves better performance in the early stage of exploration, but the advantage of our method gradually opens up in the middle and late stages; this is because the high information gain will be obtained regardless of which frontier is explored in higher priority in the early stage, so the quality of the global exploration sequence does not affect the exploration efficiency of the early stage. Since our method will promptly explore the frontiers that may have low information gain but will cause the back-and-forth maneuvers, the efficiency of our method may not be the most efficient in the early stages. However, as exploration advances, back-and-forth maneuvers caused by the low-quality global exploration sequence gradually emerge, which leads to the reduced efficiency of FUEL. Owing to the high-quality exploration sequence, as shown in Fig. 6(a)–(c), our method has significantly fewer back-and-forth maneuvers, so our method achieves better performance in the total exploration effectiveness.

3) Further Evaluation: We can also see that the fluctuation range of the results generated by our method in different initial positions is very limited, which implies that different initial positions have a limited impact on the final exploration performance of our method. This is because once the exploration area is determined, no matter where the initial position is, it only affects the exploration area and exploration trajectory in the early stage, and our method will drive the UAV to explore the area according to our exploration principles. In addition, we have conducted a comparison of computation time for the proposed method and the three state-of-the-art methods; the comparison result is shown in Table III. We can see that owing to the use

TABLE I
EXPLORATION STATISTIC IN THE THREE SCENARIOS

Scene	Method	Exploration time (s)				Flight distance (m)				Coverage (m^3)			
		Avg	Std	Max	Min	Avg	Std	Max	Min	Avg	Std	Max	Min
Maze1	Aeplanner	328.1	43.7	400.9	222.07	199.0	26.3	249.1	**140.9**	859.7	24.0	899.0	805.3
	NBVP	650.3	139.8	928.9	418.2	302.7	60.4	413.3	198.6	859.8	41.5	**950.3**	756.8
	FUEL	158.9	8.2	175.6	146.6	219.0	10.5	242.4	200.6	**901.8**	3.2	909.0	**896.0**
	Proposed	**116.3**	**6.4**	**128.3**	**103.1**	**164.6**	**10.0**	**182.9**	141.6	892.9	**2.3**	898.4	889.2
Maze2	Aeplanner	302.0	31.0	346.9	235.0	171.7	17.1	198.2	126.7	726.8	14.3	745.4	680.2
	NBVP	517.9	76.3	653.7	336.2	248.7	37.8	320.8	159.6	733.3	23.5	**768.9**	689.0
	FUEL	132.4	7.6	145.9	117.5	179.7	10.8	196.2	150.5	**753.3**	3.2	762.7	**748.2**
	Proposed	**105.5**	**6.0**	**116.0**	**95.5**	**141.1**	**10.0**	**159.1**	**122.2**	747.1	**2.1**	751.9	742.8
Outdoor	Aeplanner	661.2	101.0	856.2	463.8	405.2	52.5	505.0	308.3	1763.5	9.7	**1783.7**	1746.1
	NBVP	917.1	185.9	1390.4	630.5	429.9	76.4	591.6	300.8	1683.3	76.5	1777.3	1501.9
	FUEL	177.0	9.5	201.7	161.1	260.1	14.4	298.5	236.4	**1773.9**	**4.3**	1781.7	**1767.6**
	Proposed	**139.7**	**8.5**	**154.8**	**121.6**	**208.2**	**11.7**	**232.0**	**186.4**	1755.9	10.5	1776.9	1735.0

The bold entities are the best results.

TABLE II
PARAMETERS USED FOR SIMULATION

Max Velocity	2.0 m/s	Max Accelerate	2.0 m/s^2
Max Yaw Rate	1.0 rad/s	Camera Range	4.5 m
Camera FOV	[80,60] deg	ROS Version	ROS Melodic

TABLE III
AVERAGE COMPUTATION TIME IN DIFFERENT SCENES

Method	Experimental results in three scenes (ms)			
	Maze1	Maze2	Outdoor	Avergae
Aeplanner	623	756	761	713
NBVP	794	912	926	877
FUEL	29	25	33	29
Proposed	24	21	31	25

of a heuristic framework that contains an FIS, our method and FUEL can easily maintain crucial information in the entire space. Therefore, our method and FUEL achieve better performance in computation time.

In addition, as shown in Fig. 8, we also design a simple yet effective comparison experiment in a corridor for our method, our method without adaptive yaw planning (named by Method-1 for clarity), and FUEL to cover our contributions. To finish the exploration, FUEL takes 45.4 s and 74.4 m, and Method-1 takes 32.2 s and 60.5 m. Our method achieves the best performance with 30.7 s and 55.9 m. We can see from Fig. 8 that our trajectory is more reasonable and has fewer back-and-forth maneuvers, which is mainly related to four parts: Part 1, Part 2, Part 3, and Part 4. Part 1 is the area where the exploration begins. Owing to the limitation of the FOV, some small areas will be left behind. Part 2 contains a corner that will cause an area that is difficult to explore promptly. Part 3 is a protruding corridor. Part 4 is an area containing a large corner that is easy to leave areas when turning. For Part 1 and Part 2, since we prioritize the exploration of small independent areas mentioned in Section III-A, our method and Method-1 will explore the areas promptly even with a longer stay in these areas, which slightly sacrifices the local efficiency but avoids the subsequent larger impact caused by back-and-forth maneuvers. Therefore, we can see that the trajectories of our method and Method-1 have a clear detour in the two parts. For Part 3, owing to the use of the spiral filling concept as shown in Fig. 2(a), our method (including Method-1) will explore the part in higher priority to reduce inefficient paths. In Part 4, since our method uses the adaptive yaw planning mentioned in Section III-B, we will cover more frontiers by yaw change. Therefore, compared with Method-1 and FUEL, we will not leave unexplored areas that need to go back and explore again in Part 4. For proving the above analysis, as shown in Fig. 9, we also provide the global exploration statuses of the three methods when UAV flies to Part 4; the distribution of unexplored areas is consistent with our analysis.

B. Complexity Analyzing

Our method has three main parts: global path solving by the ATSP, local path planning and yaw planning, and map maintenance. In the first part, we use Lin-Kernighan heuristic (LKH) [27] to solve the ATSP in the global planning. As proved in [27], the complexity of LKH is $O(n^2)$. Thus, our computation complexity in this part is $O(N_{\text{cls}}^2)$, where N_{cls} represents the number of the frontier, and its size depends on the length of the boundary between known and unknown areas. Once the length of the boundary exceeds the distance threshold (about 2 m), the boundary will be split into two frontiers. Following this principle, N_{cls} can be determined. In the planning part, we use the B-spline to generate the local trajectory and the yaw path, so the computational complexity of this part is $O(N_c)$, where N_c is the number of the control points of B-spline, and N_c is approximately equal to the value of the planning horizon divided by the distance interval of adjacent control points (about 0.5 m). In map maintenance, since we only update the local voxel grids, the time complexity is $O(N_v)$, where N_v is the number of voxel grids affected by the data of the sensor, and its size is determined by the FOV of the sensor.

C. Real-World Experiments

In order to verify the effectiveness of the proposed method, we also conduct two real-world experiments in the park and underground parking lot by using our customized quadrotor, as shown in Fig. 10. We equipped our UAV with a limited

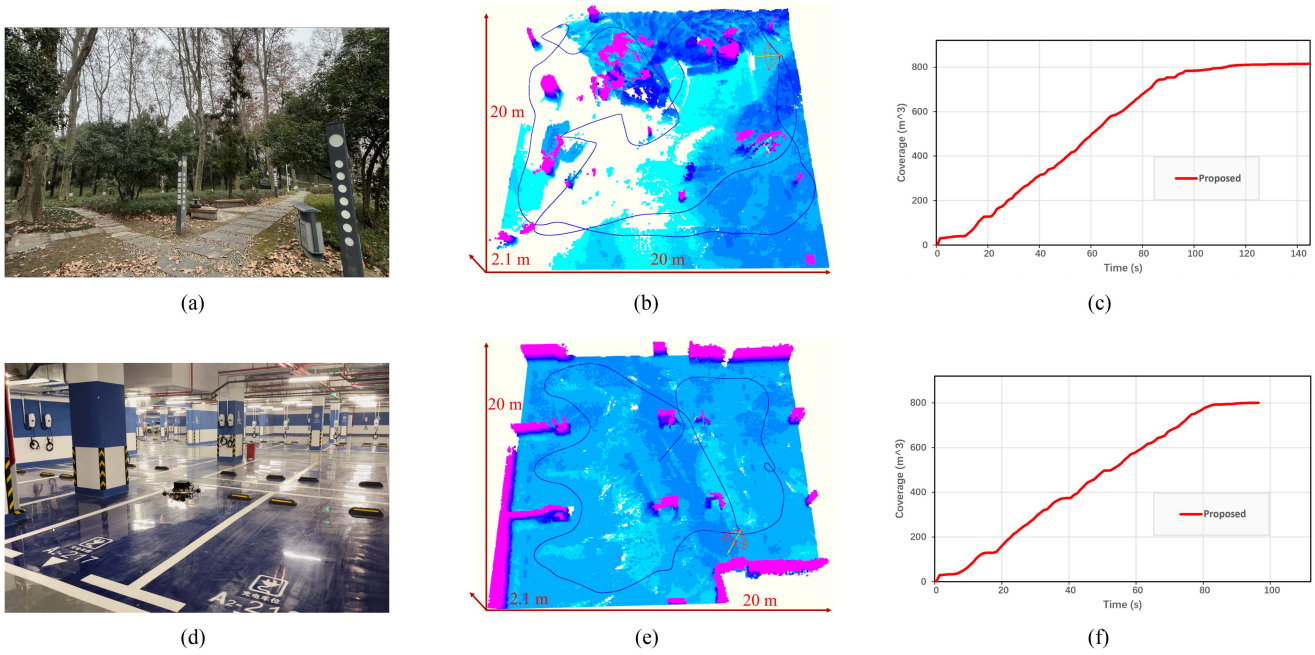

Fig. 7. Results of real-world experiments. (a)–(c) Are the experimental results in the park, (d)–(f) are the results in the underground parking lot. It should be noted that the paths in (b) and (e) are the actual exploration trajectory, which are obtained by recording the position of the UAV provided by SLAM, (e) and (f) show the change in exploration coverage relative to the exploration time, which can be obtained by the number of known voxel grids. Videos of the experiments can be found at https://youtu.be/0Y671mEwJ_A.

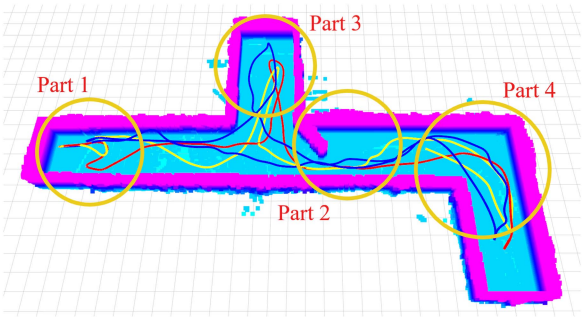

Fig. 8. Corridor scenario that is used to clearly explain the enhancement of our method. Red and blue paths are the actual exploration process of our method and FUEL, respectively. And the yellow path is the exploration process of our method without adaptive yaw planning part. Orange circles represent the key parts that cause the efficiency differences.

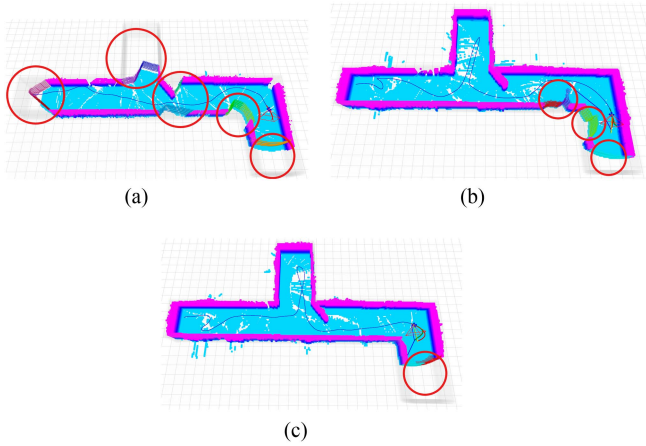

Fig. 9. Global exploration status when the UAV flies to Part 4. (a)–(c) Are the results of FUEL, Method-1, and our method, respectively. The red circles indicate the areas that have not been explored.

FOV sensor and use [28] to provide the quadrotor state. Besides, proportional–integral–derivative is adopted to provide closed-loop control. All the modules run on an onboard computer NUC equipped with Intel Core i5-1135G7@ 2.40 GHz, 16-GB memory and ROS Melodic. The detailed system parameters are shown in Table IV. In the experiments, we set dynamic limits as $v_{\max} = 1.0\,\text{m/s}$, $a_{\max} = 1.0\,\text{m/s}^2$, and $\dot{\xi}_{\max} = 1.0\,\text{rad/s}$. The same UAV and configuration are used in both the experiments.

At first, to validate our method in natural scenarios, we conduct an exploration experiment in park. The scenario contains trees, bushes, stone stools, and other objects. We bound the scenario for exploration by a $20 \times 20 \times 2.1\,\text{m}^3$ box. The

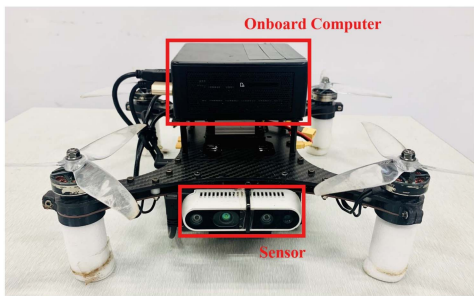

Fig. 10. Customized quadrotor used in real-world experiments.

TABLE IV
SYSTEM PARAMETERS OF THE CUSTOMIZED QUADROTOR

Parameter	Quantity
Weight	2.4 kg
Size	Diagonal: 33 cm, prop diameter: 7 in, height: 15 cm
Payloads	RGBD camera, Flight controller
Communication band	5.0 MHz
RC frequency	2.4 GHz
Propulsion	Four brushless electric motors
Flight controller	CUAV V5+
Control Interface	GCS: Laptop, Software: QGroundControl,Dronecode Project

exploration results are shown in Fig. 7(a)–(c). The exploration time of the whole process is 144.5 s, and the flight distance is 131.0 m. It should be noted that the minimum height of the exploration area preset by us is −0.1 m, but the park is an uneven area that contains a large low-lying area, which causes the visually blank area (no point clouds) in Fig. 7(b). Then, to verify our method in the underground scenario, we also conduct exploration experiments in an underground parking lot, which is bounded by a $20 \times 20 \times 2.1\ m^3$ box. The experimental results are shown in Fig. 7(d)–(f). The exploration time and flight distance of the whole exploration process are 94.3 s and 90.2 m, respectively. In addition, the data collected in the two real-world experiments are stored in an occupancy grid map with 0.1-m resolution [see Fig. 7(b) and (e)], and the sizes of the two maps are 1.82 and 3.13 MB, respectively. The above two experiments prove that our method can achieve the exploration task effectively and safely by using the limited FOV sensor in natural experiments and indoor environments.

V. CONCLUSION

This article proposed an Aeplanner for fast unknown environment mapping by the UAV equipped with limited FOV sensors. First, we designed a comprehensive frontier exploration sequence generation method, which not only considers the cost of flight level (distance, yaw change, and velocity direction change) but also considers the spatial features of the frontier to reduce the back-and-forth maneuvers. Second, according to the distribution of frontiers and flight state of the UAV, an adaptive yaw planning method was proposed to cover more frontiers when the time required for two-stage yaw change was less than or approximately equal to the flight time of the current trajectory. Finally, the path planning and dynamic replanning were adopted to increase the stability and fluency of the flight process by selecting the dynamic start point and corresponding the replanning strategy. Both the simulation and real-world experiments verified the efficiency of our method.

REFERENCES

[1] X. Zhang, Y. Chu, Y. Liu, X. Zhang, and Y. Zhuang, "A novel informative autonomous exploration strategy with uniform sampling for quadrotors," *IEEE Trans. Ind. Electron.*, vol. 69, no. 12, pp. 13131–13140, Dec. 2022.

[2] X. Zheng, F. Wang, and Z. Li, "A multi-UAV cooperative route planning methodology for 3D fine-resolution building model reconstruction," *ISPRS J. Photogrammetry Remote Sens.*, vol. 146, pp. 483–494, 2018.

[3] S. W. Chen et al., "SLOAM: Semantic lidar odometry and mapping for forest inventory," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 612–619, Apr. 2020.

[4] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging UAVs for disaster management," *IEEE Pervasive Comput.*, vol. 16, no. 1, pp. 24–32, Jan.–Mar. 2017.

[5] P. Ramesh and J. M. L. Jeyan, "Comparative analysis of the impact of operating parameters on military and civil applications of mini unmanned aerial vehicle (UAV)," *AIP Conf. Proc.*, vol. 2311, 2020, Art. no. 030034.

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

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

[8] V. An, Z. Qu, and R. Roberts, "A rainbow coverage path planning for a patrolling mobile robot with circular sensing range," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 48, no. 8, pp. 1238–1254, Aug. 2018.

[9] V. An, Z. Qu, F. Crosby, R. Roberts, and V. An, "A triangulation-based coverage path planning," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 50, no. 6, pp. 2157–2169, Jun. 2020.

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

[11] 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.

[12] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "FUEL: Fast UAV exploration using incremental frontier structure and hierarchical planning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 779–786, Apr. 2021.

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

[14] Z. Xu, D. Deng, and K. Shimada, "Autonomous UAV exploration of dynamic environments via incremental sampling and probabilistic roadmap," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2729–2736, Apr. 2021.

[15] V. M. Respall, D. Devitt, R. Fedorenko, and A. Klimchik, "Fast sampling-based next-best-view exploration algorithm for a MAV," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 89–95.

[16] C. Cao, H. Zhu, H. Choset, and J. Zhang, "TARE: A hierarchical framework for efficiently exploring complex 3D environments," in *Proc. Robot.: Sci. Syst. Conf.*, 2021.

[17] H. Zhu, C. Cao, Y. Xia, S. Scherer, J. Zhang, and W. Wang, "DSVP: Dual-stage viewpoint planner for rapid exploration by dynamic expansion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 7623–7630.

[18] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Autom., "Towards New Comput. Princ. Robot. Autom.,"*1997, pp. 146–151.

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

[20] P. Zhong, B. Chen, S. Lu, X. Meng, and Y. Liang, "Information-driven fast marching autonomous exploration with aerial robots," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 810–817, Apr. 2022.

[21] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robot. Auton. Syst.*, vol. 61, no. 12, pp. 1258–1276, 2013.

[22] E. Gonzalez, O. Alvarez, Y. Diaz, C. Parra, and C. Bustacara, "BSA: A complete coverage algorithm," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2005, pp. 2040–2044.

[23] B. Zhou, J. Pan, F. Gao, and S. Shen, "RAPTOR: Robust and perception-aware trajectory replanning for quadrotor fast flight," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1992–2009, Dec. 2021.

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

[25] Y. Zhao, L. Yan, Y. Chen, J. Dai, and Y. Liu, "Robust and efficient trajectory replanning based on guiding path for quadrotor fast autonomous flight," *Remote Sens.*, vol. 13, no. 5, 2021, Art. no. 972.

[26] 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.

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

[28] 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.

Yinghao Zhao received the M.S. degree in photogrammetry and remote sensing from PLA Strategic Support Force Information Engineering University, Zhengzhou, China, in 2019. He is currently working toward the Ph.D. degree in photogrammetry and remote sensing with the School of Geodesy and Geomatics, Wuhan University, Wuhan, China.

His research interests include autonomous exploration, path planning, and intelligent unmanned aerial vehicles.

Li Yan (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 1989, 1992, and 1999, respectively.

He is currently a Luojia Distinguished Professor with the School of Geodesy and Geomatics, Wuhan University. His research interests include photogrammetry, remote sensing, and precise image measurement.

Hong Xie received the B.S., M.S., and Ph.D. degrees in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2007, 2009, and 2013, respectively.

He is currently an Associate Professor with the School of Geodesy and Geomatics, Wuhan University. His research interests include target detection based on image deep learning, point cloud data quality improvement, point cloud information extraction, and model reconstruction.

Jicheng Dai received the M.S. degree in photogrammetry and remote sensing in 2019 from Wuhan University, Wuhan, China, where he is currently working toward the Ph.D. degree in photogrammetry and remote sensing under the supervision of Prof. Li Yan.

His research interests include light detection and ranging simultaneous localization and mapping and multisensor fusion.

Pengcheng Wei received the M.S. degree in photogrammetry and remote sensing from the Beijing University of Civil Engineering and Architecture, Beijing, China, in 2020. He is currently working toward the Eng.D. degree with the School of Geodesy and Geomatics, Wuhan University, Wuhan, China.

His research interests include point cloud registration, segmentation, and classification.