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

VRExplorer: An Efficient View-Region based Autonomous Exploration Method in Unknown Environments for UAV

Kai Xu[1], Lanxiang Zheng[2], Mingxin Wei[3], Hui Cheng*[2]

***Abstract*— Autonomous exploration plays a crucial role in robotics applications like rescue and scene reconstruction. This work addresses the challenges of autonomous exploration in intricate unknown environments by presenting a novel UAV autonomous exploration method based on a new concept of the view-region. Our proposed approach leverages the view-region to replace the conventional viewpoint generation and selection process, streamlining the planning process for exploration. Simultaneously, we model the problem of maximizing frontier coverage within the field of view during exploration, and jointly optimize it with the exploration path optimization problem. This approach ensures exploration path safety and effectiveness while being aggressive. Additionally, a gimbal is incorporated beneath the camera, with an associated optimization problem designed to minimize UAV self-rotation and enhance exploration efficiency. Simulations and real-world experiments demonstrate that the proposed method outperforms existing state-of-the-art methods in terms of runtime and distance traveled.**

I. INTRODUCTION

In recent years, unmanned aerial vehicles (UAV), especially quadrotor, has gained increasing attention due to their high flexibility and maneuverability, and are widely applied in search and rescue, scene reconstruction, and other applications. In these applications, robots autonomously exploring the environment utilize planning algorithms to determine their motion, cover the environment, and acquire information.

Currently, many methods focus on researching autonomous exploration, which can be roughly divided into sampling-based [1]–[3] and frontier-based methods [4]–[6]. Sampling-based methods generate a series of candidate viewpoints in known free space by methods such as sampling or rapidly-exploring random trees (RRTs) [1], [2] to guide the exploration. Frontier-based methods generate frontiers on the boundaries of known space, cluster them using algorithms like [5], [6] or principal component analysis (PCA) [4], and create candidate viewpoints between frontier clusters and known space. After obtaining candidate viewpoints, they all need to choose the path that connects the viewpoints by weighing factors such as shortest path length, minimum pose transformation, and maximum information gain. Finally, the path is optimized to satisfy the robot's dynamics.

This work was supported by the National Natural Science Foundation of China (U22A2095).

[1]Kai Xu is with the School of Systems Science and Engineering, Sun Yat-Sen University, Guangzhou 511400, China.

[2]LanXiang Zheng and Hui Cheng are with the School of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou 511400, China.

[3]Mingxin Wei is with the School of Artificial Intelligence, Sun Yat-Sen University, Zhuhai 519082, China.

Kai Xu and Lanxiang Zheng contributed equally to this work. *Corresponding author: chengh9@mail.sysu.edu.cn

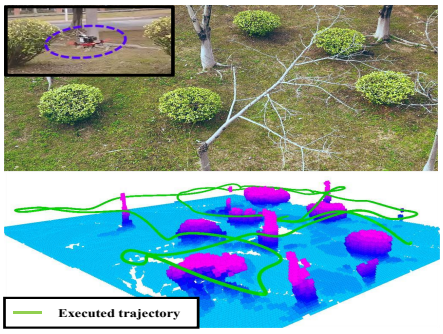


Fig. 1: Autonomous exploration using the proposed method.

Although these methods exhibit good exploration performance, the efficiency of exploration is directly influenced by the steps of viewpoint generation and selection. Generating a large number of viewpoints can lead to finer exploration paths, but increases computation consumption, which is not conducive to subsequent path planning. Conversely, a limited number of viewpoints may result in paths that are too conservative to efficiently cover the frontiers. Moreover, traditional viewpoint generation methods typically employ a greedy strategy, resulting in some viewpoints having low or no exploration value [7]. Furthermore, constrained by the load capacity of UAVs, lightweight sensors such as depth cameras are often utilized to provide environmental information for exploration. These sensors typically have a limited field of view (FoV), necessitating UAVs to hover and rotate large angles to achieve full coverage during exploration, thereby increasing the risk of collision and exploration time.

To address the aforementioned challenges, this paper introduces a novel and efficient autonomous exploration framework based on the view-region. The exploration result of the proposed method is shown in Fig. 1. In this framework, after getting the frontier clusters, a list of star-convex polyhedrons is generated between the frontiers clusters and the known space. Instead of generating viewpoints like common methods [4], view-regions are created within star-convex polyhedrons. These view-regions are not only used to guide the camera, but also to constrain the UAV to avoid collisions in the final trajectory optimization. Subsequently, a coarse search is performed within the view-region to generate a tour for guiding the UAV to explore each frontier clusters in turn. To minimize excessive rotation of the quadrotor, a single-axis gimbal is added to rotate the camera in place of a portion of the fuselage rotation. Ultimately, a joint path optimization problem is employed to co-optimize the quadrotor's final trajectory and gimbal rotation, ensuring fast exploration while maintaining the quadrotor's safety. The overview of

process is illustrated in Fig. 2. To verify the effectiveness of the algorithms, comparisons are made with the state-of-the-art algorithms FUEL [4] and TARE [8] in both simulation and real-world experiments. The results show that compared to the benchmarks, the proposed method achieves more efficient and safe exploration in several scenarios with the shortest exploration time and path length.

In summary, the contributions of this work are as follows:

- Introducing a novel autonomous exploration framework based on view-region, streamlining the exploration process of clasisic methods.
- Incorporating a gimbal onto the quadrotor to minimize unnecessary rotations and expedite boundary coverage. Constructing a joint optimization problem to simultaneously optimize trajectory and gimbal control, ensuring optimal exploration trajectory.
- Simulations and real-world experiments in diverse scenarios validate the effectiveness of proposed framework.

II. Related Work

As previously discussed, autonomous exploration refers to a robot independently collecting comprehensive map information by making decisions and taking actions in unknown environment. The task is considered complete when all available space is successfully classified as occupied or free.

Depending on specific application, research in autonomous exploration can be divided into the speed of reconstruction [4], [9] and the accuracy of reconstruction [3], [10]. This paper mainly focuses on rapid environment exploration by UAVs. Rapid autonomous exploration can be broadly classified into frontier-based and sampling-based approaches. The term ”frontier” denotes the region where free space meets unknown space [11]. Frontier-based approaches explore the environment by continuously identifying and visiting new frontiers. The exploration task is considered complete when no frontiers remain, ensuring comprehensive map coverage. The order of frontier visits affects the exploration efficiency, and [11] proposes to visit the nearest frontier clusters sequentially to fully explore the unknown environments. However, adopting this strategy may result in repeated exploration in known spaces [12]. To address this, [9] focuses on selecting frontiers within the sensor's FoV and minimizing velocity variations of the robot to achieve higher-speed exploration.

The sampling-based methods compute the next-best-view (NBV) [13] by sampling candidate viewpoints in known accessible space. In [1], the viewpoints are generated using the receding horizon next-best-view (RH-NBVP). [14] proposes the use of motion primitives to sample candidate viewpoints in order to search for an initial exploration path for the robot. The advantage of sampling-based methods is that they allow any type of information gain formulations, making them more widely used in exploration tasks [5], [15]. However, these methods tend to produce local optimum solutions, resulting in the generation of sub-optimal trajectories, sometimes failing to achieve full coverage of the map [7].

Corah et al. [16] employ a fixed FoV strategy to construct motion primitive trees for the identification of promising

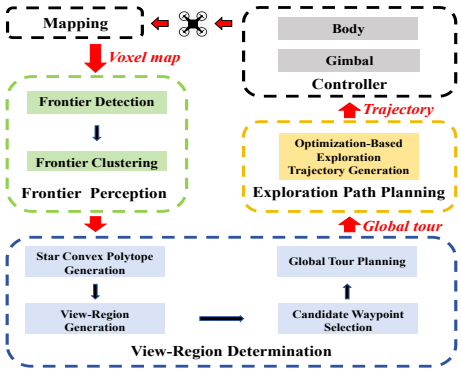


Fig. 2: Overview of the proposed autonomous exploration system.

local trajectories in their work on communication-efficient exploration. They approximate areas of potential information gain in the map by utilizing a global library of sampled views. To prevent convergence into local minima, local trajectories are penalized for deviating from the library view with a fixed minimum gain. Hybrid methods stand out as particularly promising solutions in the field of autonomous exploration. AEP [2] effectively avoiding local minima by combining RH-NBVP [1] for local planning with a frontier-based global planning approach. [17] proposes to combine frontier-based approach with sample-based approach to guide autonomous exploration. [18] suggests frontier pose sampling as a means to reduce the required number of samples.

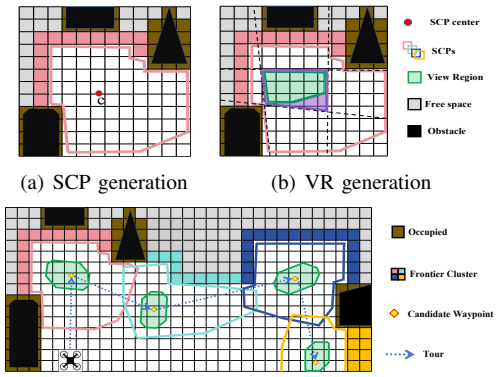
Meng et al. [19] present a two-stage planning approach, wherein viewpoints of frontiers are initially sampled, followed by the computation of global shortest paths from all sampled viewpoints. In the context of exploration efficiency, [4] introduces an incrementally updated Frontier Information Structure (FIS), which efficiently stores spatial information regarding exploration for planning purposes. Their proposed hierarchical planner storing the cost of movement between each set of frontier regions within the FIS to facilitate the computation of global travel. Additionally, they refine local exploration paths to generate safe routes, considering robot dynamics for rapid and secure exploration. The two-stage planning methods mentioned above inspire our work.

III. View-Region Determination

The method proposed in this letter utilizes voxel map [11] to represent the environment, as depicted in Fig. 3, where different colors denote distinct occupancy states. The area covered by the camera's FoV is defined as the known region. Frontiers are the boundaries separating the known region from the unexplored area, which are incrementally updated as the quadrotor moves. Prior to constructing the view-region, the frontiers are clustered into appropriately sized clusters using PCA [4].

A. Star-Convex Polyhedron Generation

The star-convex polyhedron (SCP) is a special polyhedron generated by finding visible points directly on the point cloud. The point on the surface of a SCP is visible to any points in its kernel [20]. Leveraging this property, a SCP is first generated near a frontier cluster within the known space,



(a) SCP generation (b) VR generation

(c) Exploration tour covering frontier clusters

Fig. 3: Illustration of exploration tour generation process with following steps: (a) generating a SCP (area within the red box) quickly in free region between frontiers and obstacles, and c is the center used in sphere flipping. (b) Generating view-region (green area) in the kernel of a SCP (purple area). (c) Generating a global tour (blue dotted line) which connecting individual view-regions.

as shown in Fig. 3(a). Subsequently, a view-region can be generated from its kernel, as illustrated in Fig. 3(b), where the quadrotor located within it with a specific attitude can cover all frontiers.

Proposed framework quickly generates SCPs in free region between frontier clusters and obstacles, as shown in the area within the red box in Fig. 3(a). This process firstly uses a nonlinear transformation, named sphere flipping [21] which is defined as:

$$\hat{x} = \mathcal{F}(x) = x - c + 2(R - \|x - c\|)\frac{x - c}{\|x - c\|}, \quad (1)$$

where the point x represents the frontier within the cluster or an obstacle, $\hat{x}$ is the result after sphere flipping. c is the center used in sphere flipping, and a circle centered at c with a radius of R can encompass all x.

After obtaining the sphere flipping result $\hat{x}$, an efficient convex hull algorithm [22] is used to compute the convex hull $\mathcal{X}$ of the point set $\hat{x}$. The vertices of the SCP are then obtained by applying the inverse function of Eq. 1 to the $\mathcal{X}$. Finally, a scaling operation is applied to the obtained SCP to prevent the size of the SCP from becoming too large and encompassing multiple frontier clusters.

B. View-Region Generation

View-region is a subset of the kernels of the SCP, and when the quadrotor is positioned within the view-region, the frontiers can be covered by its FoV at a suitable angle.

For a given SCP P, its kernel can be obtained by "cut" operations [23] along the inner normal vector corresponding to its surface. The Algorithm 1 we proposed outlines the details of the kernel computation process. In this algorithm, $(\cdot).faces$ and $(\cdot).verts$ represent the sets of faces and vertices of the polyhedron, respectively. Initially, Line 1 defines B as the axis-aligned bounding box (AABB) of the SCP P, and then function **calculatePlane**$(\cdot)$ is applied to determine the input faces and their corresponding normal

Algorithm 1 Computing the View-Region of Star-Convex Polyhedron

Input: Star-convex polyhedron P, inward-pointing face normals N of polyhedron P;
Output: Polyhedron K

1: B := AABB of P
2: **for** Face f in $P.faces$ **do**
3: Plane p := **calculatePlane**$(f, N(f))$
4: **for** Face f_B in $B.faces$ **do**
5: Points v_P := vertices in $P.verts$ relative to f_B
6: **if** all vertices in v_P are strictly below p **then**
7: continue
8: **else if** all vertices in v_P are weakly above p **then**
9: $B.verts \leftarrow v_P$, $B.faces \leftarrow f_B$
10: **else**
11: $aboveV, aboveF$:=**planeIntersection**(v_P, f_B, p)
12: $B.verts \leftarrow aboveV$, $B.faces \leftarrow aboveF$
13: **end if**
14: **end for**
15: **end for**
16: K := **regionRefine**(B)
17: **return** K

vectors. Lines 4-14 constitute the core of the algorithm, which segments the bounding box B according to plane p and recursively divides it into the kernel of polyhedron. Function **planeIntersection**$(\cdot)$ is used to obtain vertices $aboveV$ and face $aboveF$ above the plane p through splitting. After multiple "cut" operations, the kernels are obtained, as illustrated in purple region in Fig. 3(b). Subsequently, the view-region is derived by segmenting the region of the kernel where the distance from the frontier cluster is less than the camera's FoV using function **regionRefine**$(\cdot)$, as shown in the green region in Fig. 3(b). It is noteworthy that if the view-region computed by Algorithm 1 is empty, the frontier cluster is re-segmented using PCA, and the algorithm is applied iteratively until the corresponding view-region is obtained.

IV. Exploration Path Planning

To ensure the quadrotor efficiently completes the exploration task, comprehensive whole-body motion planning is indispensable. By integrating the view-region obtained in Sec. III-B with UAV motion planning, a joint optimization problem is formulated to optimize the final exploration path.

A. Global Tour Planning

During exploration, the frontier clusters are incrementally updated, and the sequential visitation order of multiple frontier clusters directly influences the efficiency of exploration tasks. A global tour typically guides UAV to sequentially visit existing frontier clusters to enable exploration of unknown areas.

The proposed framework follows a strategy similar to [4] for global tour planning but differs in generating only one candidate waypoint within each view-region to guide the exploration, as shown by the yellow point in Fig. 3(c).

Subsequently, the global tour planning is formulated as an open-loop traveling salesman problem (TSP), starting from the UAV's current position. The problem-solving process considers factors such as the quadrotor's rotation cost, highest coverage, and shortest distance. By transforming the problem into a standard asymmetric TSP (ATSP), the global tour can be rapidly planned, as detailed in [4].

B. Safe Flight Corridors Construction

Global tour planning determines the order for accessing all frontier clusters, yet autonomous UAV exploration necessitates a continuous trajectory that satisfying its dynamics. Therefore, further optimization is required. To ensure both security and optimality of the final exploration trajectory, a list of safe flight corridors (SFCs) [24] is constructed along the global tour to constrain the final trajectory optimization process. The set of SFCs can be represented by the $\mathcal{H}$-representation [25], i.e. the intersection of m_p half-spaces:

$$\mathcal{F}_{sfc} = \{\mathbf{p} \in \mathbb{R}^3 | \mathbf{A}_{sfc} \cdot \mathbf{p} - \mathbf{b}_{sfc} \leq 0\}, \tag{2}$$

where $\mathbf{A}_{sfc} \in \mathbb{R}^{m_p \times 3}$, $\mathbf{p} \in \mathbb{R}^3$and $\mathbf{b}_{sfc} \in \mathbb{R}^3$.

In order to connect SFCs with view-regions to jointly constrain the final trajectory, the SFCs $\mathcal{S}_k$ and $\mathcal{S}_{k+1}$ on both sides of the i-th view-region $\mathcal{V}_i$ should be properly scaled so that they satisfy the following conditions: 1) $\mathcal{V}_i \cup \mathcal{S}_k \cup \mathcal{S}_{k+1} \neq \mathcal{S}_k \cup \mathcal{S}_{k+1}$; 2) $\mathcal{V}_i \cap \mathcal{S}_k \neq \emptyset$ and $\mathcal{V}_i \cap \mathcal{S}_{k+1} \neq \emptyset$. Similarly to the definition of SFCs, the set of view-region is defined as:

$$\mathcal{F}_{vr} = \{\mathbf{p} \in \mathbb{R}^3 | \mathbf{A}_{vr} \cdot \mathbf{p} - \mathbf{b}_{vr} \leq 0\}. \tag{3}$$

C. Optimization-Based Exploration Trajectory Generation

The $\mathfrak{T}_{MINCO}$ [26] is a widely used trajectory represent and optimize tool, for the obtained global tour path, it can be defined as follow:

$$\begin{aligned} \mathfrak{T}_{MINCO} = \{p(t) : [0, T_\Sigma] \mapsto \mathbb{R}^m | \mathbf{c} = \mathcal{M}(\mathbf{q},\mathbf{T}), \\ \mathbf{q} \in \mathbb{R}^{m(M-1)}, \mathbf{T} \in \mathbb{R}^m_{>0}, \\ \mathbf{c} = (\mathbf{c}_1^T, ..., \mathbf{c}_M^T)^T \in \mathbb{R}^{2Ms \times m}\}, \end{aligned} \tag{4}$$

where the trajectory $p(t)$ is represent as a m-dimensional polynomial with M segments and the number of times $N = 2s - 1$, where s is the order of the associated chain of integrals. $\mathbf{c}$ is the polynomial coefficients, $\mathbf{q}$ denotes the intermediate points, $\mathbf{T}$ denotes the the time interval between each segment, and the total time is $T_\Sigma = \sum_{i=1}^{M} T_i$.

The i-th segment of $p(t)$ is represented as:

$$p_i(t) = \mathbf{c}_i^T \beta(t) \quad \forall t \in [0, T_i), \tag{5}$$

where $\mathbf{c}_i \in \mathbb{R}^{(N+1)\times m}$ represents the coefficient matrix, $\beta(t) = [1, t, ..., t^N]^T$ denotes the natural basis, and $T_i = t_i - t_{i-1}$ indicates the duration of the i-th segment. This minco trajectory is uniquely represented by the pair $(\mathbf{q}, \mathbf{T})$. The parameter mapping matrix $\mathcal{M}$ is built using Theorem 2 in [26], which allows the gradients of the trajectory coefficients and times to propagate back to the intermediate point and duration of each segment trajectory with linear time complexity. The $\mathfrak{T}_{MINCO}$ uses $\mathbf{c} = \mathcal{M}(\mathbf{q}, \mathbf{T})$ to convert $(\mathbf{q}, \mathbf{T})$ into $(\mathbf{c}, \mathbf{T})$, enabling any continuous cost function $J(\mathbf{c}, \mathbf{T})$ to be expressed as $H(\mathbf{q}, \mathbf{T}) = J(\mathcal{M}(\mathbf{q}, \mathbf{T}), \mathbf{T})$.

In the proposed method, the trajectory of the UAV must satisfy observability and gimbal motion constraints in addition to quadrotor dynamics. Therefore the generation of the final exploration trajectory is constructed as an unconstrained multi-objective optimisation problem:

$$\min_{\mathbf{q},\mathbf{T}} J(\mathbf{c}, \mathbf{T}) = \lambda_s J_s + \lambda_m J_m + \lambda_d J_d + \lambda_o J_o + \lambda_g J_g + \lambda_t J_t, \tag{6}$$

where J_s, J_m, J_d, and J_t represent the penalties for safety and visibility, smoothness, dynamic feasibility and running time, respectively. Penalty terms for observability and gimbal are the focus of this paper, which represented by J_o and J_g, respectively. λ_s, λ_m, λ_d, λ_o, λ_g, and λ_t are the weights of the penalty terms above, respectively. The proposed method uses $\mathfrak{T}_{MINCO}|_{m=3}$ to denote the trajectory $[x(t), y(t), z(t)]^T$ of the quadrotor and $\mathfrak{T}_{MINCO}|_{m=2}$ to represent the yaw of the quadrotor itself and the rotation of the gimbal.

1) Safety and Visibility Penalty J_s: This penalty term constrains the optimized exploration trajectory within a series of convex polyhedrons comprising view-regions and SFCs, ensuring the safety of the quadrotor while guiding it to cover frontiers. Given that Eq. 3 and Eq. 2 share the same form, they can be combined into the following expression:

$$\mathcal{F} = \{\mathbf{p} \in \mathbb{R}^3 | \mathbf{A} \cdot \mathbf{p} - \mathbf{b} \leq 0\}. \tag{7}$$

Then the penalty function J_s is defined as:

$$\begin{aligned} J_s &= \sum_k \mathcal{C}\left(\mathcal{G}_s(\mathbf{p}(t_k))\right) \\ \mathcal{G}_s(t_k) &= \begin{cases} 0, & \mathbf{A} \cdot \mathbf{p}\left(t_k\right) \leq \mathbf{b} \\ \mathbf{A} \cdot \mathbf{p}\left(t_k\right) - \mathbf{b}, & \text{others}, \end{cases} \end{aligned} \tag{8}$$

where $\mathcal{C}(\cdot) = \max\{\cdot, 0\}^3$ is the cubic penalty, $t_k \in (T_{k-1}, T_k]$.

2) Smoothness Penalty J_m: The proposed method ensures the smoothness of trajectories by minimizing the integral of the third-order derivative of the trajectory, which corresponds to jerk $j(t)$. This penalty function is defined as follows:

$$J_m = \int_0^{T_\Sigma} j^2(t) dt, \tag{9}$$

where $T_\Sigma = \sum_{i=1}^{M} T_i$ is the total time.

3) Dynamic Feasibility Penalty J_d: The optimization process of MINCO involves redistributing the time of segments [26]. This may lead to the optimized trajectory exceeding the quadrotor's actual physical limitations, including maximum velocity $v_{\max}$, acceleration $a_{\max}$, and yaw angle rate $\dot{\phi}_{\max}$. To avoid that, these factors should also be constrained during the optimization process. Using $\mathbf{l}_{\max}$ to represent these limitations, the penalty function is defined as follows:

$$\begin{aligned} J_d &= \sum_k \mathcal{C}\left(\mathcal{G}_d(t_k)\right) \\ \mathcal{G}_d(t_k) &= \begin{cases} 0, & \mathbf{l}(t_k) \leq \mathbf{l}_{\max} \\ ||\mathbf{l}(t_k) - \mathbf{l}_{\max}||^2, & \text{others}. \end{cases} \end{aligned} \tag{10}$$

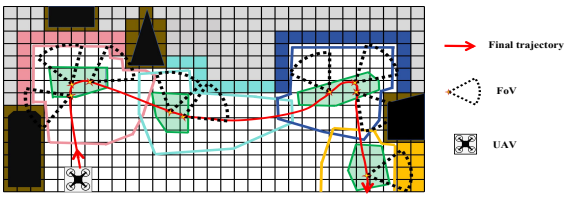


Fig. 4: Final exploration trajectory of proposed method after optimization, as shown by the red line.

4) Observability Penalty J_o: This term guides the quadrotor's FoV to cover as much frontiers as possible within the nearest frontier cluster. The desire angle of the FoV at position $\mathbf{r}(t)$ is defined as $\psi(\mathbf{r}(t)) = \theta(\mathbf{r}(t)) + \phi(\mathbf{r}(t))$, which includes the rotation of the gimbal $\theta(\mathbf{r}(t))$ and the rotation of the quadrotor itself $\phi(\mathbf{r}(t))$. J_o is defined as:

$$J_o = \int_0^{T_\Sigma} ||\frac{1}{1+\mathcal{K}(\psi(\mathbf{r}(t)))}||^2 dt, \tag{11}$$

where $\mathcal{K}(\cdot)$ is a function that statistically counts the number of frontiers covered and smoothes them.

5) Gimbal Penalty J_g: Assuming the gimbal's rotation angle range is $[\theta_{\min}, \theta_{\max}]$, with a maximum angular rate of $\dot{\theta}_{\max}$. The constraint function for the maximum angular rate $\mathcal{G}_{g,1}$ can be calculated using Eq. 10. The constraint on the rotation angle $\mathcal{G}_{g,2}$ can be defined as:

$$\mathcal{G}_{g,2}(t_k) = \begin{cases} ||\theta(t_k)||^2 - \theta_{\min}^2, & \theta(t_k) \le \theta_{\min}, \\ 0, & \theta(t_k) \in [\theta_{\min}, \theta_{\max}], \\ ||\theta(t_k)||^2 - \theta_{\max}^2, & \theta(t_k) \ge \theta_{\max}. \end{cases} \tag{12}$$

To minimize unnecessary quadrotor rotation, the proposed method prioritizes gimbal rotation. The quadrotor rotates itself only after the gimbal rotation reaches its physical limits. Additionally, the gimbal should turn in the same direction as the quadrotor's yaw after covering frontiers to provide a forward view. Therefore, the following penalty function is defined:

$$\begin{aligned} \mathcal{G}_{g,3}(t_k) =& \rho_1 ||\psi(t_k) - \theta(t_k)||^2 + \rho_2 ||\theta(t_k)||^2 \\ &+ \mathcal{W}(\mathbf{r}(t_k)) ||\theta(t_k) - \phi(t_k)||^2, \end{aligned} \tag{13}$$

where the first two terms penalize the rotation of the quadrotor itself and the rotation of the gimbal, respectively. The penalty weight $\rho_1 \gg \rho_2$ ensures that the gimbal rotates before the quadrotor itself. The last term adjusts the gimbal rotation to align with the forward direction of the quadrotor when the number of frontiers covered by the FoV is less than the threshold c_{th}. The weighting function $\mathcal{W}(\cdot)$ is a binary function, where $\mathcal{W}(\mathbf{r}(t_k)) = \rho_3$ when $\mathcal{K}(\mathbf{r}(t_k)) < c_{th}$, otherwise, $\mathcal{W}(\mathbf{r}(t_k)) = 0$.

Finally, the gimbal penalty J_g can be expressed as:

$$J_g = \sum_k \left[\mathcal{C}\left(\mathcal{G}_{g,1}(t_k)\right) + \mathcal{C}\left(\mathcal{G}_{g,2}(t_k)\right) + \mathcal{C}\left(\mathcal{G}_{g,3}(t_k)\right)\right], \tag{14}$$

6) Total Time Penalty J_t: This item improves the aggressiveness of the trajectory by minimizing the total time:

$$J_t = \sum_{i=1}^{M} T_i. \tag{15}$$

We employ the L-BFGS[3] [27] to solve the problem Eq. 6. The optimized trajectory is efficient, safe, and satisfies the quadrotor's dynamics, as depicted in Fig. 4.

V. EXPERIMENTS

The proposed method is validated through both simulation and real-world experiments. Comparative experiments are conducted to compare our approach with the state-of-the-art methods, FUEL [4] and TARE [8]. Fig. 5 shows the customized quadrotor platforms for real-world experiments. The platform incorporates a rudder as the gimbal, which is equipped with an depth camera D435i as onboard sensor on the top. All experiments are performed on a NUC with i7-10710U CPU, 32GB RAM, and Ubuntu 20.04 system.

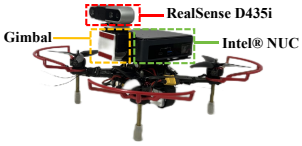
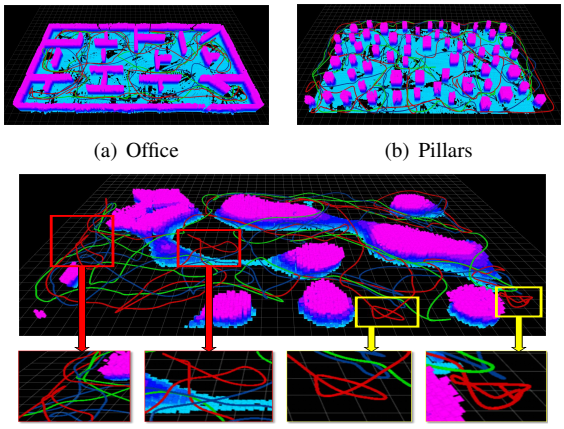


Fig. 5: The customized quadrotor used in real world experiments.

A. Simulation Experiments Evaluation

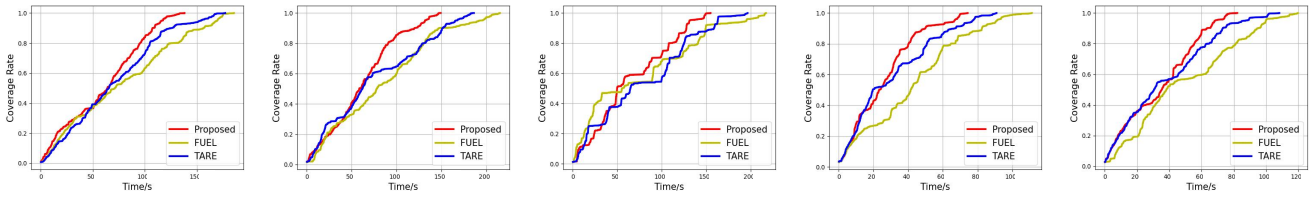


(a) Office (b) Pillars

(c) Hills

Fig. 6: The exploration paths in three scenarios are compared between the proposed method, FUEL, and TARE. Their paths are marked in green, red and blue in the figure, respectively.

Simulation experiments are conducted in three classical scenarios: Office [4], Pillars [4], and Hills [26]. The quadrotor's maximum velocity is set to $v_{\max} = 2.0$m/s, maximum acceleration $a_{\max} = 2.0$m/s^2, maximum rotation rate $\dot{\phi}_{\max} = 0.9$rad/s, and the camera's maximum sensing range is 4.5m. The gimbal's steering angle range is $\theta \in [-60°, 60°]$, with maximum steering rate of $\dot{\theta}_{\max} = 1.5$rad/s. Each method are tested more than 20 times per scenario, ensuring fairness by utilizing the same initial configuration. This work evaluates each method based on the following three metrics: Coverage rate over time, exploration time and length of the path. Wherein, the former term indicates the percentage of the volume of the explored area of the quadcopter UAV to the volume of the total target area over time.



(a) Office. (b) Pillar. (c) Hills. (d) Shrublands. (e) Garden.

Fig. 7: Exploration coverage rate of all methods over time in three simulation scenarios(a-c) and two real-world scenarios(d-e).

TABLE I: Performance Comparison of Different Exploration Methods in Various Simulation Scenes

Scene	Method	Exploration Time /s			Exploration Path /m		
		Avg	Max	Min	Avg	Max	Min
Office	FUEL [4]	168	186	163	223	249	210
	TARE [8]	156	173	151	201	211	193
	Proposed	**121**	**138**	**116**	**178**	**191**	**169**
Pillars	FUEL [4]	170	179	166	210	223	199
	TARE [8]	161	176	155	202	209	196
	Proposed	**131**	**145**	**124**	**182**	**189**	**177**
Hills	FUEL [4]	203	216	194	228	240	211
	TARE [8]	199	210	181	211	217	208
	Proposed	**167**	**178**	**161**	**186**	**180**	**192**

1) Office: The validity of the method is first verified in a $30 \times 20 \times 2.5m^3$ office room. During 3D autonomous exploration, the quadrotor visits the frontier space of interest one by one and is extraordinarily precise in observing all corners. Shown in Fig.6(a), proposed method reduces repetitive explorations with an average path length of 178m, 20.2% and 11.4% shorter compared to FUEL and TARE, respectively. Results in Table I reveal an average improvement of 28.0% and 22.4% in exploration efficiency compared to FUEL and TARE, respectively. In Fig.7(a), the proposed method excels in completing exploration of the entire space while maintaining a high and stable rate.

2) Pillars: In a $32 \times 16 \times 2.5m^3$ pillars scene, our method achieves the fastest 3D exploration rate of the entire scene (refer to Fig.7(b)). As depicted in Table I, compared to FUEL and TARE, our proposed method averages a 22.9% improvement in exploration efficiency and shortens the exploration path by 13.3% and 9.9%, respectively.

3) Hills: In the $32 \times 20 \times 2.5m^3$ hills scene, the proposed method still can complete full coverage of the entire scene at a high and relatively stable speed (see Fig.7(c)). As shown by the green trajectory in Fig.6(c), the method generates safe and concise trajectories even in tight corners, emphasizing its efficiency. As shown in Table I, compared to benchmarks, our method can improve the exploration efficiency by an average of 17.7% and 16.1%, while shortening the exploration path by an average of 18.4% and 11.8%, respectively.

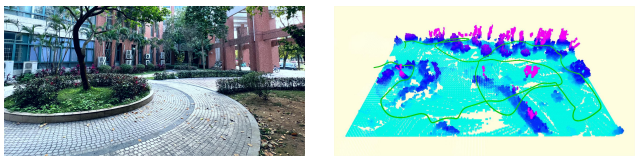
After conducting simulation experiments and comparing them with the benchmark methods, it is evident that the proposed method shows significant improvement in two aspects. Firstly, the benchmark methods often encounter the issue of repeated exploration, as indicated by the red trajectories in the red box in Fig. 6(c). This occurs because these methods select multiple viewpoints in one frontier cluster to guide exploration, and UAVs are unable to cover all frontiers at once during flight due to factors such as control errors, resulting in them needing to return for a second exploration. In contrast, proposed method selects only one candidate waypoint in the view-region and generates a trajectory that covers all frontiers based on its visibility, effectively solving the problem of repeated exploration. Additionally, benchmark methods frequently require hovering and adjusting the camera at a large angle to cover frontiers when exploring narrow spaces, as shown by the red trajectory in the yellow box in Fig. 6(c). The proposed method addresses this issue by loading the camera on a gimbal and rotating it preferentially during exploration according to the designed J_g, enabling the camera to cover all frontiers quickly, thereby improving the speed and safety during local exploration tasks.

(a) The picture of the shrubland scene. (b) 3D exploration path of the proposed method in the shrublands.

Fig. 8: The real-world experiment 1 occurs in shrublands.

B. Real-World Exploration experiments

The real-world experiments are subject to various disturbances and uncertainties, posing significant challenges to the autonomous exploration mission of UAV. To further validate the effectiveness and robustness of proposed method, this work conducts real-world experiments in two scenarios using a self-designed quadrotor. The dynamic limits of the quadrotor are the same as those in simulations. All methods are tested at least five times with the same initial configuration.



(a) The picture of the garden scene. (b) 3D exploration path of the proposed method in garden.

Fig. 9: The real-world experiment 2 occurs in a garden.

1) Shrublands: We first conduct the real-world experiments in a $30 \times 20 \times 2.5m^3$ shrublands with scattered obstacles (Fig. 8(a)). Proposed method accomplishes the 3D exploration task with the highest and most stable speed (see Fig. 7(d)). As indicated in Table II, in comparison to FUEL and TARE, the proposed method enhances exploration time by 32.7% and

18.7%, while simultaneously reducing the exploration path by 20.8% and 15.3%, respectively. These results underscore the effectiveness and stability of proposed method.

TABLE II: Exploration Performance in Real-World Tests

Scene	Method	Exploration Time /s			Exploration Path /m		
		Avg	Max	Min	Avg	Max	Min
Shrublands	FUEL [4]	110	99	117	168	182	157
	TARE [8]	91	94	88	157	162	151
	Proposed	**77**	**84**	**74**	**133**	**144**	**127**
Garden	FUEL [4]	119	122	117	171	183	162
	TARE [8]	107	112	103	158	161	153
	Proposed	**82**	**79**	**89**	**129**	**137**	**121**

2) Garden: This scenario is a $25 \times 18 \times 2.5m^3$ garden with trees. As evident from Table II, proposed method still outperforms the two benchmarked methods, achieving 31.1% and 23.4% improvements in exploration time, along with 24.6% and 17.7% reductions in exploration path, respectively.

During all real-world experiments, proposed method effectively solves the problems contributing to inefficient exploration, like repeated visits and hovering observations. As shown by the steadily rising red curve in Fig. 7, our method exhibits stable and excellent 3D exploration performance.

VI. CONCLUSIONS

This paper introduces a novel framework for autonomous exploration in complex and unknown environments based on the view-region. The proposed method eliminates the conventional viewpoint generation and selection process and combines the problem of maximizing frontier coverage with the exploration path optimization process, providing a safe, efficient, smooth, physically constrained, and aggressive exploration path for the UAV. Furthermore, the inclusion of a gimbal reduces unnecessary self-rotation of the UAV. Simulations and real-world experiments highlight the superior performance in runtime and traveled distance compared to existing state-of-the-art methods, underscoring the approach's effectiveness in autonomous exploration.

REFERENCES

[1] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3d exploration," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1462–1468.

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

[3] 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 Robotics and Automation Letters*, vol. 5, no. 2, pp. 1500–1507, 2020.

[4] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "Fuel: Fast uav exploration using incremental frontier structure and hierarchical planning," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 779–786, 2021.

[5] D. Duberg and P. Jensfelt, "Ufoexplorer: Fast and scalable sampling-based exploration with a graph-based planning structure," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2487–2494, 2022.

[6] J. Huang, B. Zhou, Z. Fan, Y. Zhu, Y. Jie, L. Li, and H. Cheng, "Fael: Fast autonomous exploration for large-scale environments with a mobile robot," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1667–1674, 2023.

[7] J. Yu, H. Shen, J. Xu, and T. Zhang, "Echo: An efficient heuristic viewpoint determination method on frontier-based autonomous exploration for quadrotors," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 5047–5054, 2023.

[8] C. Cao, H. Zhu, H. Choset, and J. Zhang, "Tare: A hierarchical framework for efficiently exploring complex 3d environments." vol. 5, 2021.

[9] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2135–2142.

[10] S. Song and S. Jo, "Online inspection path planning for autonomous 3d modeling using a micro-aerial vehicle," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 6217–6224.

[11] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, 1997, pp. 146–151.

[12] J. Faigl, M. Kulich, and L. Přeučil, "Goal assignment using distance cost in multi-robot exploration," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 3741–3746.

[13] C. Connolly, "The determination of next best views," in *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, vol. 2, 1985, pp. 432–435.

[14] M. Dharmadhikari, T. Dang, L. Solanka, J. Loje, H. Nguyen, N. Khedekar, and K. Alexis, "Motion primitives-based path planning for fast and agile exploration using aerial robots," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 179–185.

[15] C. Papachristos, S. Khattak, and K. Alexis, "Uncertainty-aware receding horizon exploration and mapping using aerial robots," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 4568–4575.

[16] M. Corah, C. O'Meadhra, K. Goel, and N. Michael, "Communication-efficient planning and mapping for multi-robot exploration in large environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1715–1721, 2019.

[17] G. Best, R. Garg, J. Keller, G. Hollinger, and S. Scherer, "Resilient multi-sensor exploration of multifarious environments with a team of aerial robots," 2022.

[18] A. Dai, S. Papatheodorou, N. Funk, D. Tzoumanikas, and S. Leutenegger, "Fast frontier-based information-driven autonomous exploration with an mav," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9570–9576.

[19] Z. Meng, H. Qin, Z. Chen, X. Chen, H. Sun, F. Lin, and M. H. Ang, "A two-stage optimized next-view planning framework for 3-d unknown environment exploration, and structural reconstruction," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1680–1687, 2017.

[20] D.-T. Lee and F. P. Preparata, "An optimal algorithm for finding the kernel of a polygon," *Journal of the ACM (JACM)*, vol. 26, no. 3, pp. 415–421, 1979.

[21] T. Liu, Q. Wang, X. Zhong, Z. Wang, C. Xu, F. Zhang, and F. Gao, "Star-convex constrained optimization for visibility planning with application to aerial inspection," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE Press, 2022, p. 7861–7867.

[22] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.

[23] T. Sorgente, S. Biasotti, and M. Spagnuolo, "Polyhedron kernel computation using a geometric approach," *Comput. Graph.*, vol. 105, no. C, p. 94–104, jun 2022.

[24] Z. Zhang, Y. Zhong, J. Guo, Q. Wang, C. Xu, and F. Gao, "Auto filmer: Autonomous aerial videography under human interaction," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 784–791, 2023.

[25] C. D. Toth, J. O'Rourke, and J. E. Goodman, *Handbook of discrete and computational geometry*. CRC press, 2017.

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

[27] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical Programming*, vol. 45, pp. 503–528, 1989.