Exploring in Uneven Terrain: A Novel Map Representation and Decision-Making with Frontier Observations

Hongru Zhang[1] and Li Fan[2]

***Abstract*— In this paper, we propose an exploration framework for UGVs in uneven terrain. The uneven environment is presented in a novel map called Plane-Oct, which takes the local geometric features to extract frontiers dynamically. To the best of our knowledge, this is the first time that a frontier maintenance strategy in uneven environments has been proposed in the field of wheeled robot exploration. Additionally, we propose viewpoints sampling and trajectory optimization methods especially designed for uneven terrain. Extensive simulation experiments have shown that the framework we proposed can not only robustly maintain frontiers in uneven environments with varying fluctuations, but also reduce ineffective transfer trajectories, thereby shortening the exploration path and improving efficiency. Compared with the baseline, the exploration strategy we proposed consumes less than 10% of exploration time and 15% of trajectory length. Our algorithm code is made publicly available.** [3]

I. INTRODUCTION

Exploration is one of the important issues in the field of robotics. The problem is set as follows: In a completely unknown environment, the robot uses its sensor information to make decisions independently, plan its movement path in real-time, and map the limited space as soon as possible. Exploration algorithms in even environments are relatively mature[1][2]. However, in some mission scenarios, especially in search and rescue and planetary exploration, the environment is not as flat as that of indoors, which usually has varying degrees of fluctuations. In some extreme situations, there are even cliffs and hills in the scene. Thus, classical algorithms will not have good performance in such circumstances. The focus of this research is challenges posed by uneven terrain to wheeled robot exploration tasks: In terms of perception, we need to design a map that has facilitation in both terrain representation and deployments of off-the-shelf planning algorithms. What's more, frontiers[3] should be maintained dynamically in this map. In terms of decision-making, we need to evaluate the observation benefits of trajectories in uneven terrain, thus improving the efficiency of missions.

Our main work can be summarized as the following:

1. We propose a new map representation Plane-Oct based on Octomap[4]. Plane-Oct parameterizes local geometric features of uneven terrain and stores them in voxels. The surface of the terrain is regularized evenly in an octree, which is favorable for memory management and indexing in the planning step.

2. We propose an algorithm to extract frontiers under the assumption of plane continuation, which proposes a good solution to the real-time maintenance of frontiers in uneven terrain.

3. We propose a decision-making strategy especially designed for exploration tasks in uneven terrain, including viewpoints sampling, trajectory optimization, and re-planning. Experiments show that observation benefits and re-planning we proposed successfully cut down unnecessary transfer processes and time overhead.

*This research was supported by the Intelligent Aerospace System Leading Innovation Team Program of Zhejiang (Grant No. 2022R01003).

[1]Hongru Zhang is with Faculty of Control Science and Engineering, Zhejiang University, Hangzhou, China. 88huru88@gmail.com

[2]Li Fan is with Huzhou Institute, Zhejiang University, Huzhou, China. fanli77@zju.edu.cn

[3]https://github.com/Ru-hulu/Exploring-in-Uneven-Terrain

II. RELATED WORK

Autonomous exploration of wheeled robots in uneven terrain is a challenging problem. The approach described in this paper is based on key results in autonomous navigation in uneven terrain and exploration in unstructured environments briefly discussed in this section.

A. Autonomous Navigation in Uneven Environments

Navigation is the key process of the exploration mission. Krüsi's method[5]is based on the point clouds map. The map controls the density of point clouds data through filtering to save memory overhead. In the planning process, points within a small range are iteratively fitted into planes to approximate the uneven environment. Atas, etc. [6][7] take a further step by directly using surfels, regular patches with normal vectors, to represent the terrain in local space. They perform real-time local fitting, thereby enhancing the efficiency of the path search process. Ruetz, etc. [8] makes use of the fast convex hull algorithm [9] to reconstruct the mesh of local terrain. Instead of performing plane fitting, they use the inclination of triangles in mesh to quickly determine the safety of the local area. Chaoqun[10], etc. directly apply navigation algorithms on octomap. By improving the resolution accuracy, detailed information of uneven terrain can be detected, ensuring the safety of navigation.

Although various navigation methods in uneven terrain have been proposed, there are still challenges when it comes to exploration tasks[11]. Compared with navigation, exploration imposes higher demands on the robot's perceptual capabilities: A global map is necessary, and the spectral distribution of frontiers should be maintained in real-time. Maps used in methods based on point clouds[5] and surfels[6][7] are not regular, posing challenges to indexing and frontier extraction. Although mesh-based methods[8][9] propose a

solution to frontier extraction in graph structure, the entire map should be updated when new sensor data frames arrive, which is not conducive to exploration efficiency. Due to its intricate data structure, Octomap is helpful for indexing in the planning process. However, to guarantee the safety of navigation, the resolution should be accurate enough. This implies that this strategy is only suitable for exploration missions in limited areas. The perceptual method we proposed adopts the data structure of the octree. It regularizes the uneven environment, fitting the local surface with parameters as in [6][7]. Plane-Oct is especially designed for navigation and exploration tasks in large environments as we can weigh parameters in a coarse resolution to check safety efficiently. With the geometric information saved in voxels, a frontier extraction strategy is proposed based on the assumption of surface continuation, providing crucial guidance to the decision-making process.

B. Evaluation of Observation Benefits

Map representation and frontier extraction strategies form the basis of decision-making. The decision-making process in unstructured environments can be divided into two steps: viewpoint sampling and path planning. The sampling and evaluation of viewpoints directly affect the efficiency of exploration. Common indicators include information gain, distance cost, and truncation entropy [12], etc. While dense sampling of viewpoints [13] [14] ensures that observation benefits of the path at discrete locations are maximized, there are drawbacks when applying this strategy to uneven terrain. The field of view is prone to be occluded by the fluctuation of the surface. Algorithms such as ray-casting can help evaluate viewpoints in such circumstances, but the computation becomes burdensome when candidates are numerous. Additionally, determining the posture of viewpoints is not conducive to path optimization.

To evaluate the observation benefits in uneven terrain, we introduce the frontier distance field to assist decision-making and optimization. Instead of optimizing several discrete states, the frontier distance field is used to optimize the entire trajectory, maximizing observation benefits while considering smoothness and safety. Additionally, we use the frontier distance field to segment the transition stages of the path, improving the efficiency of exploration tasks by avoiding redundant transfer paths.

III. Methodology

The overall architecture of the proposed method is shown in Fig. 1. The system mainly consists of two parts: a map server and an exploration planner. The map server parameterizes sensor data and updates Plane-Oct in real-time. The planner conducts viewpoints sampling and generates a smooth path, guiding the robot to explore unknown areas.

A. Map Server

1) Map Representation of Uneven Terrain: To efficiently represent uneven terrain, we propose a novel map called Plane-Oct, based on Octomap and adopting the octree data

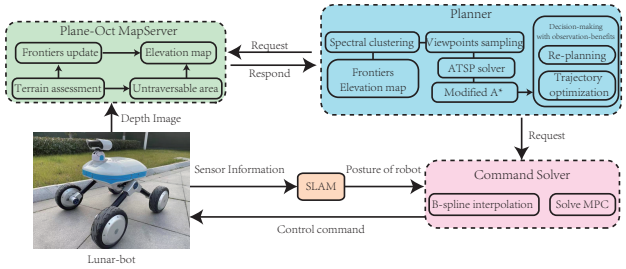

Fig. 1. The overall architecture of the proposed method.

structure. However, in contrast to storing occupancy probabilities, Plane-Oct focuses on the assessment of terrain characteristics, as illustrated in Fig. 2.

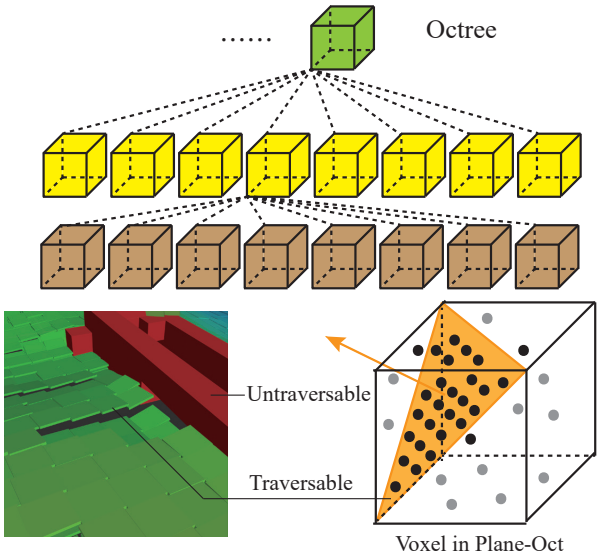

Fig. 2. The basic element of Plane-Oct is illustrated here. The map adopts the data structure of octree. It stores terrain assessment information, including states, flatness, sparsity, and geometric details. By dividing terrain surfaces into uniform spaces, Plan-Oct is memory-efficient and friendly to off-the-shelf navigation algorithms designed for uneven terrain.

$$v_i = \{a_i, b_i, c_i, d_i, \sigma_i, \gamma_i, s_i, \varphi_i\} \tag{1}$$

In Eq. 1, a_i, b_i, c_i, d_i are parameters of the local plane, which are obtained by fitting point clouds within v_i. σ_i and γ_i are flatness and sparsity can be calculated in fitting process.

$$\sigma_i = \frac{1}{n}\sum_{k=1}^{n}\frac{|a_i x_k + b_i y_k + c_i z_k + d_i|}{\sqrt{a_i^2 + b_i^2 + c_i^2}}, \gamma_i = \frac{sizeof(P_{inline})}{n_{max}} \tag{2}$$

When the vehicle is driving, we often pay less attention to the road conditions in small areas. Instead, we are more concerned about the information of ground fluctuation and flatness. Based on the above considerations, we define the traversability of a voxel v_i by weighting three criteria: the slope θ_i, the flatness σ_i, and the sparsity γ_i:

$$\varphi_i = \alpha_1(1-\gamma_i) + \alpha_2\frac{\min(\theta_{th}, \theta_i)}{\theta_{th}} + \alpha_3\frac{\min(\sigma_{th}, \sigma_i)}{\sigma_{th}} \tag{3}$$

In Eqution 4, α_1, α_2, α_3 are weights sum to one. It implies that if terrain in a voxel is traversable, it's likely to have a

comparatively small φ_i, which means that the slope, flatness, and sparsity are acceptable. Conversely, when φ_i approaches 1, the condition of the surface worsens.

For the convenience of illustration, we define M as the workspace to be explored. M can be divided into four subsets, $M_{traverable}$, M_{danger}, $M_{frontier}$, and $M_{unknown}$. The border of $M_{traverable}$ and $M_{unknown}$ are defined as frontiers. Thus, a voxel can be labeled based on the subset to which it belongs. We use s_i to categorize the states of voxels. Specifically, $s_i = \{-1, 0, 1\}$ corresponds to the voxel belonging to M_{danger}, $M_{traverable}$, and $M_{frontier}$, respectively.

The map server updates sensor data in real-time to the Plane-Oct, as outlined in Algorithm 1. Inspired by the frontier extraction algorithm [15] designed for grid maps, Algorithm 1 maintains active voxels to monitor their status. (The strategy of frontiers extraction will be explained in the next section.) The workspace currently scanned by the sensor can be divided into a known set V_{know} and an unknown set $V_{unknown}$. Each unknown voxel and point clouds P_i within it constitute the basic element of $\widetilde{P}_{unknown}$ (line 1). For previously observed voxels (V_{know}), we extract the subset $V_{fcheck} \in M_{frontier}$ (line 2) and identify voxels that are no longer frontiers ($V_{NowTrav}$) from them (line 6). The algorithm employs RANSAC to perform plane fitting on point clouds in $\widetilde{P}_i$ (line 3). Newly observed frontiers extracted from this process will be updated to $M_{frontier}$ (line 4, line 9, line 10).

Algorithm 1 Update Plane-Oct and FrontierSet.

Input: A Frame of Depth Image F, Plane-Oct, Camera Posture T_i

Output: Plane-Oct, FrontierSet

1: $\{V_{know}, \widetilde{P}_{unknown}\} \leftarrow$ DistributePoints2Voxel(T_i,F)
   //$\widetilde{P}_{unknown} = \{v_i, P_i | P_i$ *is point clouds within* $v_i\}$
2: $\{V_{fcheck}\} \leftarrow$ GetFrontierSet(V_{know})
3: Plane-Oct.UpdateParameter($\widetilde{P}_{unknown}$)
4: $\{V_{NewFrontier}, V_{NewTrav}\} \leftarrow$ CheckNeigh($V_{unknown}$)
5: $\{V_{NowTrav}\} \leftarrow$ CheckNeigh(V_{fcheck})
6: Plane-Oct.UpdateState($V_{NowTrav}$)
7: FrontierSet.DeleteSubSet($V_{NowTrav}$)
8: FrontierSet.InsertSubSet($V_{NewFrontier}$)
9: Plane-Oct.UpdateState($V_{NewFrontier}$)

2) Frontiers Extraction in Uneven Terrain: Extracting frontiers in real-time is a challenge when exploring uneven terrain. Most of the algorithms designed for Octomap will check surrounding units (6 or 26 voxels) of a voxel. If a voxel is adjacent to an unknown unit, it will be labeled as a frontier. However, this strategy is not suitable for uneven environments because a voxel is used to assess a patch of surface. Thanks to the geometric information stored in Plane-Oct, we propose an extraction strategy based on the assumption of local plane continuity: by analyzing the intersection of the fitting plane in a voxel with each face of the cube, we can determine which unit around it may be passed through as it extending in space.

Take case 1 in Fig. 3 as an example. There is a plane representation stored in v_i: $\pi_i = \{a_i, b_i, c_i, d_i\}$. We number

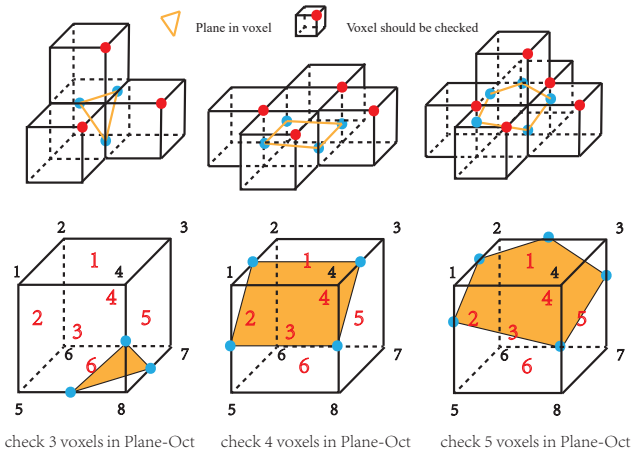

Fig. 3. Frontiers extraction strategy is illustrated in Fig. 3. The direction vector d_i can be calculated based on geometric information. Due to the safety angle and geometric limitations, the index relationship between d_i and neighboring units that the plane will cross can be stored in Plane-Dict in advance. In Fig. 3, we show several representative cases in frontiers extraction, with the remaining cases being spatially symmetrical.

the 8 vertices and the 6 adjacent units of v_i, calculating the position of vertices related to the local plane and obtain a direction vector $\vec{d} = \{1,1,1,1,1,1,1,0\}$ (line 1). Among them, 1 means that the vertex is above the plane, otherwise it is below. For a voxel with $\vec{d}$, the plane can only intersect with the units 3, 5, and 6 (line 2). Therefore, if one of the units is unknown, v_i is determined as a frontier (line 3, line 4). Since how the cube and the plane intersect are limited, the mapping relationship between the direction vector and units should be checked can be obtained in advance.

Algorithm 2 CheckNeigh

Input: Plane-Oct, a voxel v_i

Output: state of v_i

1: $\vec{d}$ = v_i.CalculateDirectVector()
2: $\vec{b}$ = Plane-Dict.search($\vec{d}$)
3: V_{neigh} = Plane-Oct.GetNeigh($\vec{b}$,v_i)
4: s_i = Allobserved(V_{neigh})

In some steep areas, the irregularity of the terrain may lead to dead zones, as shown in Fig. 4: Since the number of points in v_3 is too small, it's likely to be filtered in Fig. 2. Since v_3 is in the plane extension of π_1 and π_2, v_1 and v_2 will be labeled as frontiers. However, the area is completely covered by sensor data and has no value for observation. Therefore, as shown in Fig. 4, if a frontier voxel meets the condition of being surrounded by th_{check} known units in the projection direction along the z-axis, it will be labeled as a traversable space and removed from $M_{frontier}$.

B. Exploration Planner

In the previous section, we store the surface of the terrain in a novel map called Plane-Oct, which is favorable to memory management and indexing. The frontier extraction strategy we proposed leverages the intrinsic characteristics

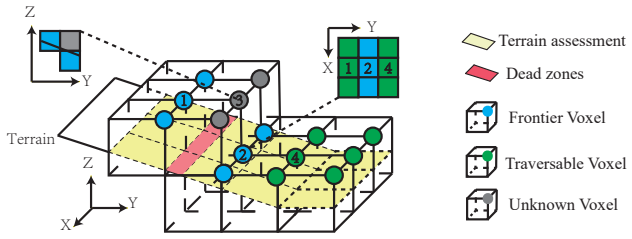

Fig. 4. Dead zones occured in frontiers extraction can be eliminated through checking neighbors in the z-axis. v_2 is surrounded by 8 known units in the z-axis, so it will be deleted from $M_{frontier}$ and labeled as traversable.

of the terrain, demonstrating adaptability to diverse surface features and environmental structures. Having provided a comprehensive perceptual solution for uneven environments, in this section we focus on the decision-making process.

1) Generate Viewpoints in Planning Space: Viewpoints are discrete landmarks on the exploration path. Owning to their observation benefits, the robot will expand the scope of the known map with high efficiency. Therefore, the sampling and evaluation process of viewpoints are essential for the efficiency of exploration. The unknown space that a viewpoint can cover is a main indicator of effectiveness. Because the fluctuation of uneven environment can easily block the sensor's field of view, evaluating viewpoints in 3D will cause an unbearable computational burden. Therefore, sampling and evaluation processes are performed on the elevation map, which is the projection space of Plane-Oct(as shown in Fig. 5). Voxels with $\varphi_i > \varphi_{safe}$ will be marked as dangerous in projection space. Inspired by [14], planning processes are only performed in a set of subspaces, which is a balance between real-time and long-term benefits. To select effective viewpoints, the method proposed has three steps: clustering, sampling, and reachability testing. Spectral clustering is employed to abstract the spatial distribution characteristics of frontiers, so that decision-making will be robust to noise. The centers of clusters, denoted as $C = \{C_i | C_i \in R^3, i = 1, \ldots, n_c\}$, are instrumental in avoiding meaningless sampling. Each cluster center C_i serves as the focal point for a ray casting-based sampling process, which can avoid obstruction of obstacles. Subsequently, a revised A* algorithm which restricts the height gap between two adjacent states is employed. This algorithm is used to select reachable viewpoints denoted as $P = \{\rho_i | \rho_i \in R^3, i = 1, \ldots, n_\rho\}$. In this section, we ensure that the exploration process is concentrated in the short term, as only viewpoints reachable within limited space are considered. What's more, isolated frontiers are neglected during the clustering step, eliminating the trajectory without observation benefits.

2) Exploration Tour Planning: To observe the unknown environment as efficiently as possible, we need to construct the shortest safe trajectory that passes through viewpoints one by one. This process can be abstracted as an ATSP problem. Assuming there are n_ρ reachable viewpoints, the cost matrix dimension is $M^{(n_\rho+1)\times(n_\rho+1)}$, where

$$M(i,j) = M(j,i) = astar_{dist}(\rho_i, \rho_j), 2 < i, j \leq n_\rho + 1 \quad (4)$$

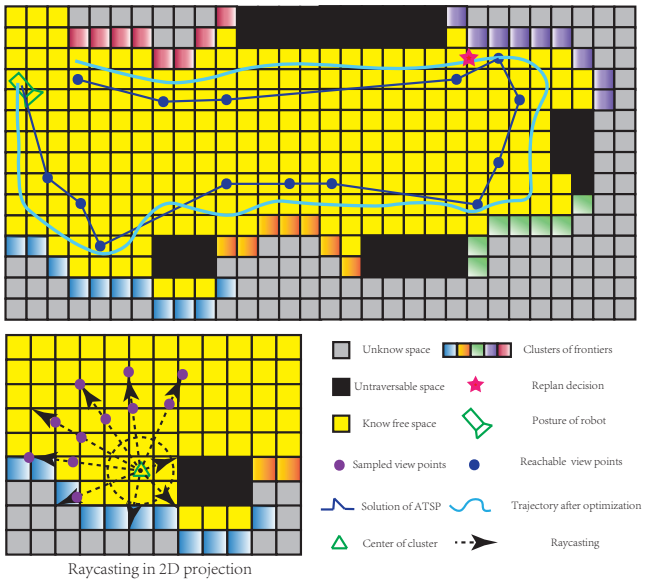

Fig. 5. Three main steps of the decision-making process are illustrated in this figure. Viewpoints sampling is performed in the projection space, and an Asymmetric Traveling Salesman Problem (ATSP) is formulated to determine the most efficient sequence for visiting all viewpoints. A modified A* algorithm is applied to the elevation map to generate a coarse trajectory. An observation cost is introduced in both the optimization and decision-making processes. Instead of maximizing the observation benefit in discrete postures, we optimize the observation benefit of the entire trajectory. The efficiency of exploration is enhanced by promoting increased observations in unknown areas during robot movement and reducing unnecessary transfer stages of the entire task.

$astar_{dist}$ is the collision-free path length searched by A* in the projection space.

Considering the particularity of the exploration problem, the robot does not need to return to the starting point after visiting viewpoints. We set the cost from viewpoints to the start position as 0:

$$M(j,1) = 0, 1 \leq j \leq n_\rho + 1 \quad (5)$$

In this way, going back to the current viewpoint in any closed loop tours contributes no extra cost, so each closed-loop tour always contains an open-loop one with an identical cost. As a result, we obtain the optimal open-loop tour $P\prime = \{\rho_1\prime, \rho_2\prime, \ldots, \rho_{n_\rho}\prime | \rho_i\prime \in P\}$ by finding the optimal closed-loop one with the help of LKH solver. After that, we can search a coarse trajectory $\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n_{tr}} | \xi_i \in SE(2)\}$ on the projected space. Considering that the focus of this paper is not the navigation algorithm on uneven terrain, we simply use the revised A* mentioned before.

3) Trajectory Optimization: The purpose of trajectory optimization is usually to provide the robot with a trajectory that is both smooth and safe, as highlighted in previous studies [16]. In exploration tasks, we want to ensure that more unknown areas can be observed during the movement of the robot. The objective function we designed is as follows, where w_f, w_s, w_o are weights of loss items, satisfying $w_f + w_s + w_o = 1$.

$$c(\Xi) = \sum_{i=1}^{n_{tr}} w_f \cdot c_f(\xi_i) + w_s \cdot c_{smooth}(\xi_i) + w_o \cdot c_{obs}(\xi_i) \quad (6)$$

Among them, c_{smooth} ensures the smoothness of trajectory:

$$c_{smooth}(\xi_i) = \|(\xi_{i+1} - \xi_i) - (\xi_i - \xi_{i-1})\| \quad (7)$$

c_{obs} serves as a safety term designed to steer the trajectory away from hazardous areas, where $SDF_o(\xi_i)$ represents the distance between ξ_i and its nearest obstacle.

$$c_{obs}(\xi_i) = \begin{cases} \alpha_{obs} \left(SDF_{obs}(\xi_i)\right)^2 & SDF_{obs}(\xi_i) < d_{th1} \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

c_{obs} and c_{smooth} are common items in trajectory optimization. To improve the efficiency of exploration tasks, we want the robot to observe as many unknown areas as possible during the tracking process. Some algorithms[17] carefully sample and optimize the postures of viewpoints to maximize the observation benefits at discrete locations. However, this approach falls short in optimizing the observation benefit along the entire path. It can only enhance the observation benefit through intensive viewpoints sampling. Additionally, postures of viewpoints are likely to be adjusted during the optimization, inevitably impacting the observation benefit. A common solution is to divide the path into several parts and optimize them individually. However, this approach sacrifices smoothness and safety, as vehicles have to move drastically at breakpoints. In this paper, we propose a more natural idea: aligning the trajectory as closely as possible to frontiers to ensure comprehensive sensor coverage of unknown space. Simultaneously, the trajectory must maintain a safe distance from frontiers, because unobserved obstacles may exist in unknown areas, and being too close to these regions will increase the risk of collision. For a waypoint in trajectory Ξ, the observation cost is designed as follows:

$$c_f(\xi_i) = \begin{cases} \alpha_{fro} \left(SDF_{fron}(\xi_i) - d_{th2}\right)^2 & SDF_{fron}(\xi_i) < d_{th2} \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

It can be seen that c_{obs} and c_f share similar mathematical forms. Before the optimization, we will prepare the signal distance field (SDF) of frontiers and obstacles. d_{th2} is a constant determined by the safety distance. α_{fro}, α_{obs} are positive real numbers used to adjust the gradient during the optimization. By assigning losses to path points within a specific proximity to frontiers, we break free from the constraint of maximizing observation benefits at discrete locations. Instead, we optimize the entire path, considering safety and smoothness simultaneously.

4) Decision-making with Observation Benefits: During the exploration, the spatial distribution of frontiers changes dynamically. However, due to the onboard resource limitations, the trajectory cannot be updated in real-time as the map updates. This significantly constrains the efficiency of exploration. To improve the efficiency of exploration, we employ a heuristic method to proactive re-planning. After optimization, the path is divided into transfer sections (e.g., BD in Fig. 6) and observation sections (e.g., AB, DE) based on whether they have observation benefits. The transfer segment smoothly connects viewpoints, while the observation segment helps the robot cover unknown areas of the map. During the tracking process, we execute only the initial trajectory with observation benefits and subsequently re-plan. However, determining the observation benefits of trajectory points accurately can impose an unnecessary computation burden. Therefore, we propose a heuristic method based on the distance field. Assuming that the optimized trajectory is $\Xi\prime = \{\xi\prime_1, \xi\prime_2, \ldots, \xi\prime_{n_{tr}} | \xi_i\prime \in SE(2)\}$, then first exploration segment with observation benefits can be expressed as:

$$\begin{aligned} \Xi_{exp} = \{\xi\prime_1, \xi\prime_2, \ldots, \xi\prime_e | \xi_i\prime \in SE(2)\} \\ SDF_f(\xi_i\prime) < th_{view}, n_{tr} \geq e \geq e_{th} \geq 0 \end{aligned} \quad (10)$$

Where e_{th} is a length threshold used to avoid oscillation caused by frequent decision-making. The advantage of this strategy is obvious: when the robot is in observation sections, the map is updated in real-time, potentially updating frontiers nearby. If the robot transfers to different areas, it might lead to unnecessary turnback movements. The re-planning decision proposed aims to minimize such redundant transfers, allowing exploration to focus within a confined range. This concentration is beneficial for enhancing energy utilization.

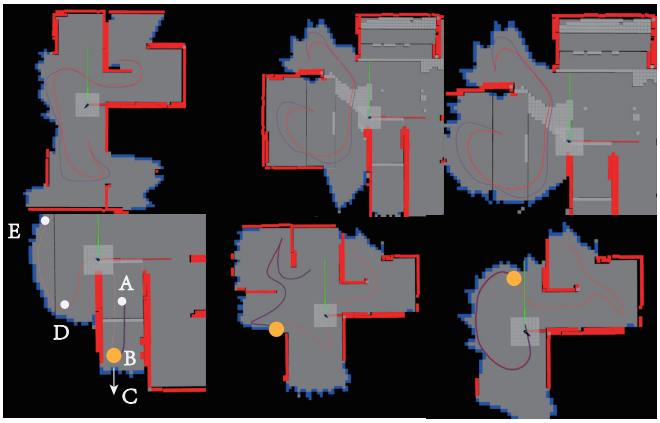

Fig. 6. In the first line, purple trajectories are optimized using c_f, while red trajectories are not. In the second line, purple trajectories are $\Xi\prime$, red trajectories are Ξ, and yellow dots are positions where the re-planning will be triggered.

IV. Experiments

We use the Robot Operating System (ROS) as the platform to deploy the algorithm and build three uneven scenes in Gazebo to conduct experiments. The sizes of them are all $20m \times 20m$, and the difficulty gradually increases (Fig. 7). The baseline is a classical exploration algorithm (TSP), which doesn't have observation benefits in the optimization process and re-planning.

Time consumption and the length of trajectories are shown in Tab. I. We conducted five experiments in each scenario, and the results demonstrate that the exploration framework we proposed can effectively finish exploration tasks in highly challenging landscapes. Additionally, the observation benefit and re-planning strategy we introduced contribute to

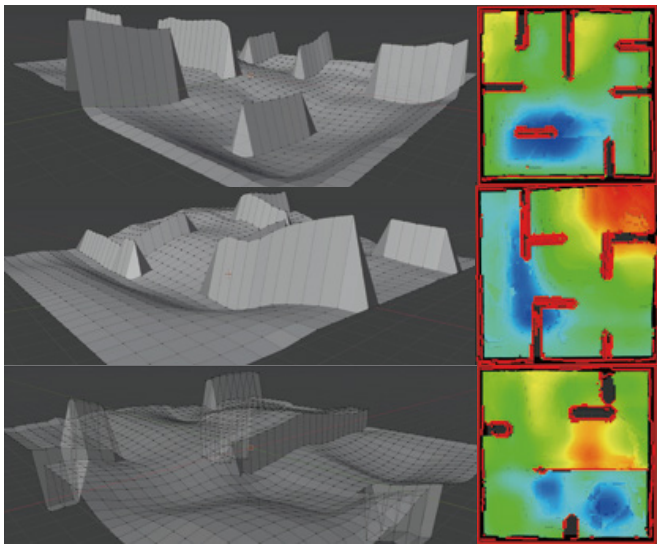

Fig. 7. We show the simulation scenes (left) and mapping results of exploration (right). In Plane-Oct, voxels can be divided into two categories. One is traversable areas, represented as surfels with posture, demonstrating the geometric information. The other is untraversable areas, represented as red cuboids, as illustrated in Fig. 2. Traversable surfels are regularized in an octree and are colored according to their z-axis.

TABLE I
COMPARISON OF TRAJECTORIES AND TIME CONSUMPTIONS

Scene	Method	Length of Trajectory (m)	Time (s)
1	Baseline	248m	329s
	Ours	201m	286s
2	Baseline	208m	274s
	Ours	179m	245s
3	Baseline	198m	271s
	Ours	185m	261s

efficiency: the trajectory length has been reduced by 15% on average. Correspondingly, the average exploration time overhead has also decreased by 10%. We observe that as the obstacle density increases in the scene (1-3), the optimization effect becomes more pronounced. This tendency is reasonable: since frontiers tend to be distributed near obstacles, the gradient of c_{obs} is detrimental to exploration efficiency, but the introduction of c_f compensates for this trend. The compensation effect becomes more apparent as the obstacle density increases in the scene. In scenarios where obstacles are rare, the observation efficiency of the path itself is already high because viewpoints are selected based on the distribution of frontiers. Thus, the effect of c_f is not obvious. Additionally, the transfer path is typically located near obstacles in a cluster Fig. 6, indicating that the re-planning strategy will not yield significant efficiency improvements if obstacles are sparse.

A. Conclusion

In this paper, we present a comprehensive solution for wheeled robot exploration tasks in uneven terrain. Our approach includes a novel map representation called Plane-Oct, a frontiers extraction algorithm, and a decision-making strategy specifically designed for uneven terrain. We demonstrate the feasibility of proposed method through simulation experiments, and the algorithm code is publicly available. In the near future, we plan to deploy the framework in the physical world and develop a more sophisticated navigation system to enhance the overall robustness and efficiency of the framework.

REFERENCES

[1] M. Naazare, F. G. Rosas, and D. Schulz, "Online next-best-view planner for 3d-exploration and inspection with a mobile manipulator robot," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3779–3786, 2022.

[2] Z. Wang, L. Chen, H. Chen, H. Chen, and X. Jiang, "Fast and safe exploration via adaptive semantic perception in outdoor environments," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 9445–9451.

[3] J. A. Placed, J. Strader, H. Carrillo, N. Atanasov, V. Indelman, L. Carlone, and J. A. Castellanos, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Transactions on Robotics*, 2023.

[4] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.

[5] P. Krüsi, P. Furgale, M. Bosse, and R. Siegwart, "Driving on point clouds: Motion planning, trajectory optimization, and terrain assessment in generic nonplanar environments," *Journal of Field Robotics*, vol. 34, no. 5, pp. 940–984, 2017.

[6] Z. Jian, Z. Lu, X. Zhou, B. Lan, A. Xiao, X. Wang, and B. Liang, "Putn: A plane-fitting based uneven terrain navigation framework," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 7160–7166.

[7] F. Atas, G. Cielniak, and L. Grimstad, "Elevation state-space: Surfel-based navigation in uneven environments for mobile robots," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 5715–5721.

[8] F. Ruetz, E. Hernández, M. Pfeiffer, H. Oleynikova, M. Cox, T. Lowe, and P. Borges, "Ovpc mesh: 3d free-space representation for local ground vehicle navigation," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8648–8654.

[9] H. Azpúrua, M. F. M. Campos, and D. G. Macharet, "Three-dimensional terrain aware autonomous exploration for subterranean and confined spaces," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 2443–2449.

[10] C. Wang, L. Meng, S. She, I. M. Mitchell, T. Li, F. Tung, W. Wan, M. Q.-H. Meng, and C. W. de Silva, "Autonomous mobile robot navigation in uneven and unstructured indoor environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 109–116.

[11] D. Kim, D. Carballo, J. Di Carlo, B. Katz, G. Bledt, B. Lim, and S. Kim, "Vision aided dynamic exploration of unstructured terrain with a small-scale quadruped robot," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2464–2470.

[12] S. Bai, J. Wang, F. Chen, and B. Englot, "Information-theoretic exploration with bayesian optimization," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1816–1822.

[13] Y. Gao, Y. Wang, X. Zhong, T. Yang, M. Wang, Z. Xu, Y. Wang, Y. Lin, C. Xu, and F. Gao, "Meeting-merging-mission: A multi-robot coordinate framework for large-scale communication-limited exploration," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 13 700–13 707.

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

[15] J. Oršulić, D. Miklić, and Z. Kovačić, "Efficient dense frontier detection for 2-d graph slam based on occupancy grid submaps," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3569–3576, 2019.

[16] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," *Ann Arbor*, vol. 1001, no. 48105, pp. 18–80, 2008.

[17] B. Zhou, H. Xu, and S. Shen, "Racer: Rapid collaborative exploration with a decentralized multi-uav system," *IEEE Transactions on Robotics*, 2023.