Robotica (2025), **43**, pp. 2855–2871
doi:10.1017/S0263574725101860

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

SEAC: a simultaneous exploration and coverage planner for online aerial 3-D modeling

Shiyong Zhang[1], Xuebo Zhang[1], Qianli Dong[1], Tianyi Li[1], Haobo Xi[1], Ziyu Wang[1], Chaoqun Wang[2] and Jingjin Yu[3]

[1]Institute of Robotics and Automatic Information System, Nankai University, Tianjin, China
[2]School of Control Science and Engineering, Shandong University, Jinan, China
[3]Department of Computer Science, Rutgers, the State University of New Jersey, Piscataway, NJ, USA
Corresponding author: Xuebo Zhang; Email: zhangxuebo@nankai.edu.cn

Received: 5 September 2024; **Revised:** 25 April 2025; **Accepted:** 19 May 2025; **First published online:** 22 July 2025

Keywords: micro aerial vehicle; informative path planning; autonomous exploration; online 3-D modeling

Abstract

In this paper, we propose a novel online informative path planner for 3-D modeling of unknown structures using micro aerial vehicles. Different from the *explore-then-exploit* strategy, our planner can cope with exploration and coverage simultaneously and thus obtain complete and high-quality 3-D models. We first devise a set of evaluation metrics considering the perception constraints of the sensor for efficiently evaluating the coverage quality of the reconstructed surfaces. Then, the coverage quality is utilized to guide the subsequent informative path planning. Specifically, our hierarchical planner consists of two planning stages – a local coverage stage for inspecting surfaces with low coverage quality and a global exploration stage for transiting the robot to unexplored regions at the global scale. The local coverage stage computes the coverage path that takes into account both the exploration and coverage objectives based on the estimated coverage quality and frontiers, and the global exploration stage maintains a sparse roadmap in the explored space to achieve fast global exploration. We conduct both simulated and real-world experiments to validate the proposed method. The results show that our planner outperforms the state-of-the-art algorithms and especially decreases the reconstruction error (at least 12.5% lower on average).

1. Introduction

Accurate 3-D models are essential for many applications, such as search and rescue [1], cave surveying [2, 3], and environment perception [4]. Generally, 3-D models of large structures can be reconstructed by robotic coverage. However, existing robotic coverage methods typically require a prior 3-D model of the target as input and plan the robotic coverage path offline. In contrast, robotic exploration methods do not require any prior knowledge and can explore unknown environments online. Nevertheless, existing exploration planning works usually utilize a volumetric map to describe the environment instead of the reconstructed 3-D surface model.

In this paper, we investigate the informative path planning problem for autonomous online 3-D modeling of unknown structures by using micro aerial vehicles (MAVs) equipped with limited on-board sensors. As mentioned above, robotic coverage methods usually plan the coverage path for fully known or at least partially-known structures [5–7]. These methods typically incorporate three steps for constructing accurate 3-D models: first, solving the model-based offline coverage path planning problem under a variety of perception constraints; second, performing online data collection during the MAV flight; lastly, converting the recorded data to a 3-D model utilizing off-the-shelf 3-D reconstruction pipelines. It is a tedious and time-consuming procedure and not suitable for time-sensitive applications (*e.g.* search and rescue missions [8]). Different from robotic coverage, autonomous robotic exploration can model unknown structures online by constructing a volumetric map of the target [9–14]. However, a

© The Author(s), 2025. Published by Cambridge University Press.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

complete volumetric map does not imply that the resulting 3-D model is accurate and complete (*i.e.* it may leave holes on the reconstructed surfaces). To achieve high-quality online 3-D modeling of unknown structures, one should concurrently execute exploration and coverage in a unified planning framework.

We propose the **S**imultaneous **E**xploration **A**nd **C**overage planner (SEAC) for autonomous online 3-D modeling of unknown structures based on informative path planning. First, we devise a set of metrics that take into account the perception constraints of the on-board sensor for evaluating the coverage quality of the reconstructed surfaces. These metrics can be computed quickly during the construction of the Truncated Signed Distance Field (TSDF) model. The evaluation results of the surface coverage quality are then stored in the TSDF model for the subsequent viewpoint utility calculation. Second, SEAC adopts a hierarchical framework consisting of two planning stages to achieve high coverage quality and exploration efficiency simultaneously. At the local coverage stage, the MAV will compute the local coverage path within a local planning horizon. Specifically, the viewpoint with the maximum local exploration utility is first selected as the planning goal. Then, the local coverage path with the highest coverage utility is obtained by solving a submodular orienteering problem. When there are no frontiers and uncovered surfaces in the local planning horizon, the planner will turn to the global exploration stage. At this stage, the MAV can quickly find a safe and efficient global exploration path on a sparse roadmap. The sparse roadmap is incrementally expanded during the MAV flight and the positions of its nodes are continuously updated utilizing the constructed Euclidean signed distance fields (ESDFs). Our method is validated in realistic simulations by comparing it with state-of-the-art methods. The comparative results demonstrate that our method outperforms other methods in terms of both reconstruction accuracy and completeness, without compromising exploration efficiency. Finally, we integrate SEAC into a self-built MAV and validate it through an indoor experiment.

The main contributions of this work are the following:

1. We devise a set of efficiently computable metrics for evaluating the coverage quality of the reconstructed surfaces that incorporates perception constraints of the on-board sensor. These evaluation metrics can greatly facilitate the viewpoint utility evaluation that is essential to the performance of SEAC.
2. We propose a novel planner that performs simultaneous exploration and coverage (SEAC). Our planner optimally balances between local coverage effort for inspecting low-quality surfaces and global exploration effort for extending the map.
3. We benchmark the performance of SEAC through simulation trials, demonstrating its superiority. In addition, the effectiveness of SEAC is also validated by the real-world experiment.

2. Related work

2.1. Path planning for robotic coverage

The objective of coverage path planning (CPP) is to compute the motion path for robots achieving complete coverage of the target with the on-board sensor. Existing algorithms to CPP problem typically require the prior information of the target and compute the coverage path offline. The typical workflow of CPP algorithms consists of three steps, *i.e.* viewpoints generation, CPP and execution, and offline data processing [5–7, 15, 16]. Different from most CPP algorithms, Roberts et al. [17] proposed the *explore-then-exploit* strategy to achieve CPP without any prior knowledge of the target. In the *explore* phase, the MAV is commanded to execute a default trajectory at a safe distance above the target while acquiring a sequence of images. After that, a coarse 3-D model of the target can be generated by processing the obtained images. In the *exploit* phase, the 3-D coverage path is computed based on the coarse model to maximize the fidelity of the resulting 3-D model. Although the strategy results in higher-quality 3-D reconstructions, it still can not be directly used in time-sensitive applications.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

2.2. Path planning for robotic exploration

Exploration path planning (EPP) algorithms can be roughly classified into sampling-based methods and frontier-based methods [9–14, 18]. Sampling-based methods usually sample viewpoints or paths based on the rapidly-exploring random tress (RRT) [19] and select the candidate with the largest exploration utility as the exploration goal [9, 10]. In contrast to sampling-based methods, frontier-based methods guide the exploration process by visiting frontiers [11–14]. Therefore, the important problem for frontier-based methods to solve is the sequence to visit frontiers. Generally, the optimal frontier visiting sequence can be computed by solving the traveling salesman problem (TSP) [20]. Up to now, the EPP problem has been widely investigated, and there are many promising algorithms that have been proposed for mapping unknown environments online. Nevertheless, EPP algorithms only focus on observing unexplored space, and the reconstruction quality of the target structure is not explicitly considered, which usually results in poor reconstruction quality or even leaves holes on the obtained 3-D model.

2.3. Path planning for simultaneous exploration and coverage

Recently, the path planning problem for online 3-D modeling of unknown structures has been investigated [21–26]. Heng et al. [21] first proposed the strategy that can simultaneously explore and cover unknown environments. However, their strategy requires indispensable pre-computation operations before the online planning. The work in [22] proposed to utilize surface frontiers to extend the model of the target structure. Sungho et al. [23] proposed a novel hierarchical online 3-D modeling planner. This planner first computes the sector visitation order and then plans the local inspection path to enhance the accuracy of the resulting 3-D model. However, in their work, the confidence of surfaces is estimated through Poisson surface reconstruction, which requires high computational cost. Similarly, Feng et al. [24] proposed a seminal hierarchical planner for online aerial 3-D modeling. One limitation of the planner is that it requires pre-training of the surface prediction module, which is utilized to guide the subsequent surface coverage. By contrast, Schmid et al. [26] proposed an online informative path planner for 3-D modeling by continuously expanding a single tree. Their planner employs surface frontiers and weights stored in TSDFs to guide the evaluation of coverage viewpoints. However, it does not take into account the perception constraints of the on-board sensor. Furthermore, maintaining a single tree in the explored environment often requires significant computational costs. Most recently, Zhang et al. [27] proposed a promising simultaneous exploration and photographing algorithm for fast autonomous 3-D reconstruction of unknown structures using multiple heterogeneous MAVs.

In summary, robotic coverage excels in achieving high-quality 3-D reconstructions of known structures, whereas robotic exploration is suited for constructing models of unknown structures. It is intuitive to achieve high-quality online 3-D modeling of unknown structures by integrating these two techniques within a unified framework. However, existing methods struggle to balance computational efficiency with modeling quality. Therefore, we propose a computationally efficient simultaneous exploration and coverage planner for autonomous online 3-D modeling of unknown structures using an MAV.

3. Problem formulation

3.1. MAV and sensor models

In this work, we consider the robot to be a quadrotor-type MAV equipped with a forward-looking depth sensor, and we assume the MAV can localize itself perfectly.

MAV Model: We follow the idea proposed in ref. [28] to exploit the differential flatness of MAVs, that is, only four selected *flat outputs* are considered as follows

$$\mathrm{x} := \{\mathbf{p}, \psi\} \in \mathcal{X} \subset \mathbb{R}^4, \tag{1}$$

where $\mathbf{p} = [x, y, z]^T \in \mathbb{R}^3$ is the position of the MAV and $\psi \in [-\pi, \pi)$ is the yaw angle at that position. At a state (also called viewpoint) x, the MAV collects depth measurements from the sensor and integrates them into the map.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

Sensor Model: In order to model the sensor accurately, we adopt the view frustum model that is usually used in the field of robotic coverage [29]. We assume that the sensor has its inherent and/or user-defined perception constraints such as a limited field of view (FoV), *e.g.,* it has a truncated perception range, $[d_{min}, d_{max}]$, $d \in \mathbb{R}^+$, and horizontal and vertical view angles, $\alpha_h \in (0, 360^\circ]$, $\alpha_v \in (0, 360^\circ]$, respectively.

3.2. Map representation

Generally, for the exploration problem, the volumetric map is a popular map type for the representation of unknown environments. On the other hand, the spline surface [16] or polygon mesh [29] is usually adopted to model the target by robotic coverage works. However, all these representations are not suitable for online 3-D modeling.

In this paper, we decide to utilize an efficient surface reconstruction method, Voxblox [30], for online 3-D modeling. Voxblox is an open-source library that builds TSDFs of the target incrementally and computes the corresponding ESDFs directly from the obtained TSDFs. While Voxblox is also a volumetric-type mapping method, it can generate human-readable surfaces with sub-voxel resolution. To cope with our problem features, Voxblox is modified as follows: first, it computes a high-resolution TSDF model for accurate 3-D modeling as well as the coverage quality of each reconstructed surface for the subsequent viewpoint evaluation using the proposed evaluation metrics; second, it also maintains the ESDFs from a low-resolution TSDF model for efficient collision checking and roadmap maintenance.

3.3. Problem description

In this work, we consider a path planning problem for exploring an unknown but spatially bounded 3-D space $\mathcal{V} \subset \mathbb{R}^3$ and simultaneously covering the surfaces $\mathcal{S}$ of a structure contained in this space (*i.e.* $\mathcal{S} \subset \mathcal{V}$). We can obtain a volumetric representation $\mathcal{M}$ and a surface model $\mathcal{S}$ from the TSDFs of the target space. The volumetric representation $\mathcal{M}$ can divide $\mathcal{V}$ into tree types: occupied $\mathcal{M}_{occ}$, free $\mathcal{M}_{free}$ and unknown $\mathcal{M}_{un}$, thus we have $\mathcal{M}_{un} = \mathcal{M} \setminus \{\mathcal{M}_{occ} \cup \mathcal{M}_{free}\}$. The surfaces perceived by the onboard sensor at a viewpoint x under the perception constraints will be labeled as *covered* $\mathcal{S}_{cov}$ and the remaining surfaces belonging to *uncovered* $\mathcal{S}_{un}$. Similar to $\mathcal{M}$, we have $\mathcal{S}_{un} = \mathcal{S} \setminus \mathcal{S}_{cov}$. Our objective is to simultaneously reduce the voxels belonging to $\mathcal{M}_{un}$ and surfaces belonging to $\mathcal{S}_{un}$ as much as possible and thus obtain a complete and accurate 3-D model of the target. The problem can be formulated as follows.

Problem 1: Given $\mathcal{M}_{un}$, $\mathcal{S}_{un}$, and the current state of the MAV x_{cur}, we compute the optimal path X^* consisting of a sequence of viewpoints x_i $(i = 1, 2, \ldots)$. The path X^* can be further optimized to be a feasible trajectory $\mathrm{X}^*(t)$ using the off-the-shelf algorithms. After that, the MAV will follow $\mathrm{X}^*(t)$ and thus observe $\mathcal{M}_{un}$ and cover $\mathcal{S}_{un}$. To achieve the objective more efficiently, we want to maximize the utility function when computing the MAV path in each planning epoch as follows

$$\begin{aligned} &\max\ \mathcal{U}_e(\mathcal{L}(\mathrm{X}),\ \mathcal{I}_e(\mathrm{X})) + \mathcal{U}_c(\mathcal{L}(\mathrm{X}),\ \mathcal{I}_c(\mathrm{X})) \\ &s.t. \begin{cases} \mathrm{X} \subset \mathcal{M}_{free}, \\ \mathcal{L}(\mathrm{X}) \le L_B, \\ g(\mathrm{X}) \ge 0, \end{cases} \end{aligned} \tag{2}$$

where $\mathcal{L}(\mathrm{X})$ is the motion cost for the robot following the path, X, $\mathcal{I}_e(\mathrm{X})$ and $\mathcal{I}_c(\mathrm{X})$ are the expected exploration gain and coverage information gain that the MAV can obtain along with the path, L_B denotes the motion cost budget (*e.g.* maximum path length or maximum flight time), and $g(\mathrm{X}) \ge 0$ denotes that the planned path should meet designated constraints (*e.g.* maximum linear velocity and maximum yaw rate). We can learn that $\mathcal{U}_e(\cdot)$ takes as input parameters the motion cost $\mathcal{L}(\mathrm{X})$ and the expected exploration gain $\mathcal{I}_e(\mathrm{X})$ and returns an evaluation result. $\mathcal{U}_c(\cdot)$ takes as input parameters the motion cost $\mathcal{L}(\mathrm{X})$ and the expected coverage gain $\mathcal{I}_c(\mathrm{X})$ and also returns an evaluation result. The path that

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

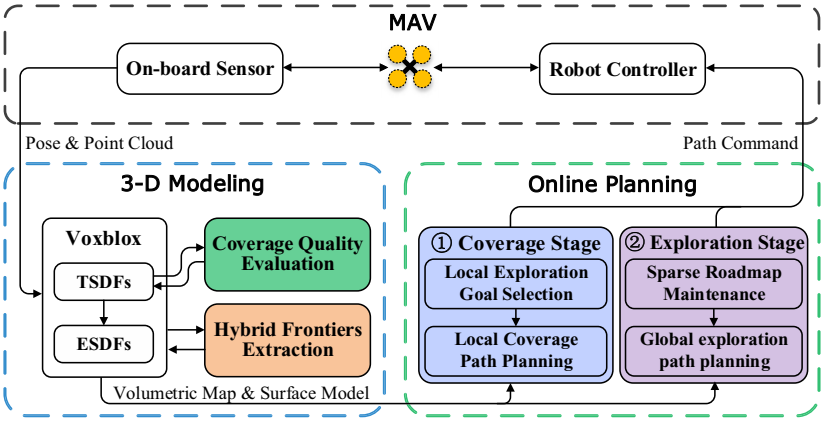


Figure 1. *Overview of our simultaneous exploration and coverage planning framework for online 3-D modeling.*

maximizes $\mathcal{U}_e(\mathcal{L}(\mathrm{X}),\ \mathcal{I}_e(\mathrm{X})) + \mathcal{U}_c(\mathcal{L}(\mathrm{X}),\ \mathcal{I}_c(\mathrm{X}))$ will be selected to explore and cover the environment simultaneously.

4. Methodology

The system overview of SEAC is shown in Figure 1. SEAC has two main components: a 3-D modeling module and a path planning module. The 3-D modeling module integrates the estimated pose and collected depth measurements into Voxblox for volumetric mapping and surface modeling. In addition, whenever the 3-D modeling module is invoked, the coverage quality of each reconstructed surface (Sec. 4.1) as well as hybrid frontiers will be also computed at the same time for facilitating the subsequent path planing (Sec. 4.2). After that, the hierarchical path planning module works to find the optimal path for simultaneously covering uncovered surfaces and exploring unknown space (Sec. 4.3 and Sec. 4.4). Finally, we utilize the SE(3) controller [31], which takes the current MAV pose and the optimal path provided by the online planner as input, to command the MAV to complete the task. The online 3-D modeling will be terminated when there are no *uncovered* surfaces and frontiers.

4.1. Surface coverage quality evaluation

The TSDF model consists of a set of voxels v with a user-defined voxel size $r \in \mathbb{R}^+$. We define the occupied voxel v_s containing a reconstructed surface $s \in \mathcal{S}$ as *surface voxel*. To achieve the best coverage quality, the on-board sensor should observe the target as perpendicularly as possible to its surfaces. Thus, the viewing angle between the ray from the on-board sensor center through the center of v_s and the normal to the reconstructed surface s should be less than a threshold $\theta_{\max}$. However, the surface normal is difficult to compute directly from the volumetric map. Existing methods usually compute the surface normal by reconstructing the whole surface of the target to obtain the surface normals [23] or fitting a plane around the surface voxel v_s [32]. Although the above methods can obtain accurate surface normals, the computational complexity is too high, making it intractable to evaluate the surface coverage quality online. Therefore, we present a method to calculate the approximate surface normal and also a set of coverage quality metrics.

Given a reconstructed surface $s \in \mathcal{S}$, its normal $\mathbf{n}_{v_s}$ can be approximately computed from the TSDF model as the distance gradient $\nabla d_s\,(v_s)$ of v_s as follows

$$\mathbf{n}_{v_s} = \sum_{v_{nbr} \in N_{v_s}^k} d_s\,(v_{nbr}) \cdot \frac{\mathbf{p}_{v_{nbr}} - \mathbf{p}_{v_s}}{\left\| \mathbf{p}_{v_{nbr}} - \mathbf{p}_{v_s} \right\|}, \tag{3}$$

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

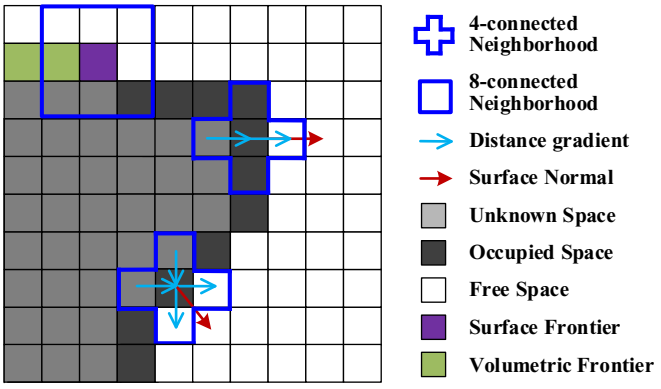


Figure 2. *Simplified 2-D examples of the surface normal calculation and frontier extraction. The 4/8-connected neighborhood in the 2-D case is a simplification of the 6/26-connected neighborhood in the 3-D case.*

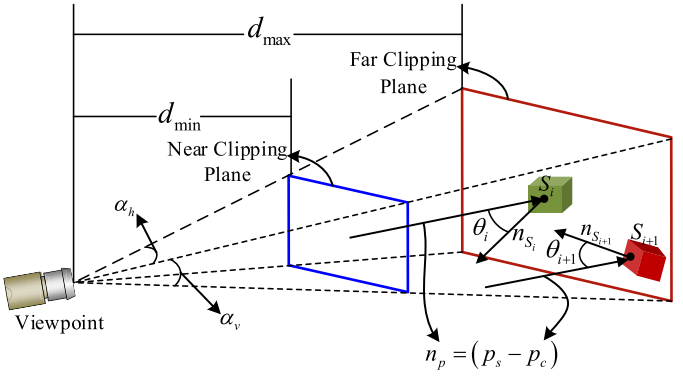


Figure 3. *Diagram of the frustum sensor model and its perception constraints. It illustrates the relationship between one viewpoint and the normals of surfaces.*

where $v_{nbr} \in N_{v_s}^k$ denotes the k-connected neighboring voxels of v_s, $d_s(v_{nbr})$ indicates the signed distance of v_{nbr} to its nearest surface, and $\mathbf{p}_{v_{nbr}}$ and $\mathbf{p}_{v_s}$ are the positions of v_{nbr} and v_s, respectively.

Let the index of v_s of the TSDF model be (x_s, y_s, z_s), we have $d_s(v_s) = d_s(x_s, y_s, z_s)$. As shown in Figure 2, when $k = 6$, the Eq. (3) can be written as

$$\mathbf{n}_{v_s} = \begin{bmatrix} d_s(x_s+1, y_s, z_s) - d_s(x_s-1, y_s, z_s) \\ d_s(x_s, y_s+1, z_s) - d_s(x_s, y_s-1, z_s) \\ d_s(x_s, y_s, z_s+1) - d_s(x_s, y_s, z_s-1) \end{bmatrix}, \tag{4}$$

similarly, we can quickly compute the normal to each reconstructed surface when $k = 18$ or $k = 26$.

To improve the coverage quality of reconstructed surfaces and thereby effectively reduce the reconstruction error of the target, we adopt the frustum sensor model (see Figure 3) to achieve accurate evaluation of the surface coverage quality. Based on the frustum sensor model and the extracted surface normals, we propose our evaluation metrics that incorporate the perception constraints of the sensor when evaluating the coverage quality of reconstructed surfaces. Specifically, if a surface $s_i \in \mathcal{S}$ is effectively covered by the sensor (indicating it has a high coverage quality), we mark s_i as *covered*, otherwise s_i will be marked as *uncovered*.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

4.1.1. Observing angle

$$\left|\cos^{-1}\frac{\mathbf{n}_{s_i}\cdot\left(\mathbf{p}_{s_i}-\mathbf{p}_c\right)}{\left\|\mathbf{n}_{s_i}\right\|\left\|\mathbf{p}_{s_i}-\mathbf{p}_c\right\|}\right|\leq\theta_{\max},\tag{5}$$

where $\mathbf{n}_{s_i}$ is the normal to surface s_i and $\theta_{\max}$ is the designated threshold of the observing angle. This metric is used to determine whether the ray from sensor center to the surface s_i is cast appropriately.

4.1.2. Observing distance

$$d_{\min}\leq\left\|\mathbf{p}_{s_i}-\mathbf{p}_c\right\|\leq d_{\max},\tag{6}$$

where $\mathbf{p}_{s_i}$ denotes the position of v_{s_i} and $\mathbf{p}_c$ is the position of the on-board sensor's center. The observing distance metric is used to judge whether the reconstructed surface has been observed with a proper distance.

4.1.3. Surface completeness

$$N_{v_{s_i}}^{18}\geq l,\ l\in\mathbb{N}^+,\tag{7}$$

where $N_{v_{s_i}}^{18}$ denotes the number of neighboring surfaces of v_{s_i} in its 18-connected neighborhood, and the number of neighboring surfaces of v_{s_i} should be not less than l. The surface completeness metric is used to mitigate the effects of isolated surfaces induced by sensor noise, thereby enhancing the completeness of the reconstructed mesh surface.

When new depth measurements arrives, we first update the TSDF model and compute the normals of newly reconstructed surfaces, and then sequentially determine whether a newly reconstructed surface s meets the above metrics. If s does not satisfy all the metrics, it will be considered to have low coverage quality (*i.e.* s will be marked as *uncovered*) and thus will be carefully inspected later. If s satisfies all the evaluation metrics, it will be considered to have high coverage quality (*i.e.* s will be marked as *covered*) and will not contribute to the coverage utility evaluation of candidate viewpoints in the subsequent planning iteration.

4.2. Hybrid frontiers extraction

In order to efficiently achieve simultaneous coverage and exploration, we extract two types of frontiers, i.e., the *volumetric frontier* and the *surface frontier*.

4.2.1. Volumetric frontier F_v

A set of connecting free voxels $v_{mf}\in\mathcal{M}_{free}$ adjacent to at least one unknown voxel, i.e., $\mathrm{F}_v=\{v_{mf}\in\mathcal{M}_{free}:\exists\, neighbor(v_{mf})\in\mathcal{M}_{un}\}$.

4.2.2. Surface frontier F_s

A set of connecting free voxels $v_{sf}\in\mathcal{M}_{free}$ adjacent to at least one unknown voxel and at least one occupied voxel, i.e.,

$$\begin{cases}\exists v_u\in N_{v_{sf}}^{8} & s.t.\ \omega\left(v_u\right)<W_{th},\\ \exists v_o\in N_{v_{sf}}^{26} & s.t.\ \omega\left(v_o\right)\geq W_{th}\wedge d_s\left(v_o\right)\leq 0,\end{cases}\tag{8}$$

where v_u is an unknown voxel, v_o is an occupied voxel, $N_{v_{sf}}^{8}$ denotes the 8-connected neighborhood of v_{sf}, and $N_{v_{sf}}^{26}$ denotes the 26-connected neighborhood of v_{sf}. ω is a weight that representing the confidence for the signed distance stored in that voxel, and W_{th} is the weight threshold to determine whether a voxel is unknown or not.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

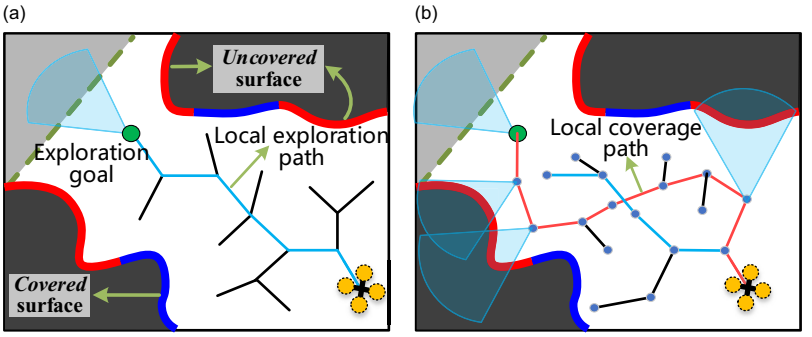


Figure 4. *Local coverage stage. (a) The MAV finds the optimal local exploration goal as well as the corresponding local exploration path, which will serve as the initial solution of the local coverage planning problem. (b) The MAV obtains the local coverage pah by solving the submodular orienteering problem.*

In this work, we utilize the frontier detection method proposed in ref. [13] to extract frontiers. F_v and F_s will be used to compute the exploration gain and the coverage gain of every viewpoint, respectively.

4.3. Local coverage stage

This planning stage is to generate the local coverage path for the MAV inspecting the *uncovered* surfaces and extending the map boundary simultaneously. As shown in Figure 4, the planning objective can be achieved by sequentially executing two operations: local exploration goal selection and local CPP, in a local planning horizon $\mathcal{H}$.

4.3.1. Local exploration goal selection

To find the optimal local exploration goal, our method expands an RRT tree $\mathbb{T}$ by sampling $\mathcal{N}_{exp}$ nodes in $\mathcal{H}$. Meanwhile, the exploration utility of every tree node is evaluated and the node with the largest exploration utility is selected as the local exploration goal x_l^*. The expected exploration utility of one tree node can be calculated similarly to [9] as follows

$$\mathcal{U}_e(\mathrm{x}_i) = \mathcal{I}_e(\mathrm{x}_i)\, e^{-\lambda_1 \mathcal{L}(\gamma_e(\mathrm{x}_r, \mathrm{x}_i))}, \tag{9}$$

where x_i is a node on the tree $\mathbb{T}$ (i.e., the exploration viewpoint), $\mathcal{I}_e(\mathrm{x}_i)$ is the exploration gain of x_i, $\mathcal{L}(\gamma_e(\mathrm{x}_r, \mathrm{x}_i))$ denotes the motion cost, λ_1 is a tunable parameter to penalize the motion cost. $\mathcal{I}_e(\mathrm{x}_i)$ can be calculated by

$$\mathcal{I}_e(\mathrm{x}_i) = \sum_{v \in Vis(\mathrm{x}_i)} \begin{cases} 1, & v \in \mathcal{M}_{un}, \\ 1, & v \in \mathrm{F}_v, \\ 0, & \text{otherwise}, \end{cases} \tag{10}$$

where $Vis(\mathrm{x}_i)$ denotes the collection of all visible voxels that can be observed at the exploration viewpoint x_i. We can learn that $\mathcal{I}_e(\mathrm{x}_i)$ is the total number of unknown voxels and volumetric frontier voxels that can be observed at x_i.

The workflow of the local exploration goal selection is presented in Alg. 1. First, the exploration tree $\mathbb{T}$ is initialized (Alg. 1, lines 3–6). After that, $\mathcal{N}_{exp}$ feasible nodes will be sampled in the free space of $\mathcal{H}$ to expand $\mathbb{T}$ (Alg. 1, lines 7–21), meanwhile, the exploration utility $\mathcal{U}_e(\mathrm{x})$ of each sampled node is computed by the function **Exploration_Utility**(), according to Eq. (9) and Eq. (10) (Alg. 1, line 15). When the tree expansion is finished, the node with the largest exploration utility will be selected as the local exploration goal x_l^*, and the corresponding local exploration path $\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*)$ can be recursively extracted from x_l^* to x_r on the tree $\mathbb{T}$ by the function **Extract_Exploration_Path**() (Alg. 1, lines 22-24, see Figure 4(a)).

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

Algorithm 1. Local exploration goal selection

Input: Volumetric map $\mathcal{M}$, Local planning horizon $\mathcal{H}$, MAV state x_r, Max iteration $\mathcal{N}_{exp}$, Step size Δ_{exp};
Output: Local exploration goal x_l^*, Local exploration path $\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*)$;
$N_{itr} \leftarrow 0$;
$\mathcal{U}_e^{best} \leftarrow 0$;
$\mathrm{x}_{best} \leftarrow \mathrm{x}_r$;
$\mathbb{T}$.**Initialize**(x_r);
while $N_{itr} < \mathcal{N}_{exp}$ **do**
  $\mathrm{x}_{rand} \leftarrow$ **Sample_Free_Space**($\mathcal{M}, \mathcal{H}$);
  $\mathrm{x}_{nearest} \leftarrow$ **Nearest_Node**($\mathbb{T}$, x_{rand});
  $\mathrm{x}_{new} \leftarrow$ **Steer**($\mathrm{x}_{nearest}$, x_{rand}, Δ_{exp});
  if Collision_Free($\mathcal{M}$, $\mathrm{x}_{nearest}$, x_{new}) **then**
    $N_{itr} \leftarrow N_{itr} + 1$;
    $\mathbb{T}$.**Add_Node**(x_{new});
    $\mathbb{T}$.**Add_Edge**($\mathrm{x}_{nearest}$, x_{new});
    $\mathcal{U}_e(\mathrm{x}_{new}) \leftarrow$ **Exploration_Utility**(x_{new}, $\mathcal{M}$);
    if $\mathcal{U}_e(\mathrm{x}_{new}) > \mathcal{U}_e^{best}$ **then**
      $\mathrm{x}_{best} \leftarrow \mathrm{x}_{new}$;
      $\mathcal{U}_e^{best} \leftarrow \mathcal{U}_e(\mathrm{x}_{new})$;
    end
  end
end
$\mathrm{x}_l^* \leftarrow \mathrm{x}_{best}$;
$\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*) \leftarrow$ **Extract_Exploration_Path**($\mathbb{T}$, x_r, x_l^*);
return x_l^* and $\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*)$.

4.3.2. Local coverage path planing

With the local exploration goal in hand, the local coverage path $\gamma_c\left(\mathrm{x}_r, \mathrm{x}_l^*\right)$ can be obtained by solving a maximization problem. Specifically, the CPP problem is described as follows

$$\gamma_c^* = \arg\max_{\gamma_c \subset \mathcal{H}} \mathcal{U}_c\left(\gamma_c(\mathrm{x}_r, \mathrm{x}_l^*)\right) \tag{11}$$

$$s.t. \begin{cases} \gamma_c \subset \mathcal{M}_{free}, \\ \mathcal{L}(\gamma_c) \le L_B, \end{cases} \tag{12}$$

where $\mathcal{L}(\gamma_c) \le L_B$ denotes that the motion cost budget (i.e., the path length) should be considered when the MAV achieves the optimal coverage, $\mathcal{U}_c\left(\gamma_c(\mathrm{x}_r, \mathrm{x}_l^*)\right)$ is the coverage utility that can be obtained along the coverage path. If the coverage gain of the viewpoint is *modular*, then the above maximization problem is a standard orienteering problem [33]. However, the coverage gain is *submodular* since the viewpoints might have overlapping FoVs, that is, one surface s can be observed by multiple viewpoints. Therefore, the above problem is a submodular orienteering problem.

Due to the coverage gain exhibits submodularity, we compute it at one viewpoint x_i by

$$\mathcal{I}_c(\mathrm{x}_i) = \sum_{\substack{v \in Vis(\mathrm{x}_i), \\ v \notin Vis(\mathrm{x}_{\mathrm{pre}})}} \begin{cases} 1, & s \in S_{uncov}, \\ 1, & v \in \mathrm{F_s}, \\ 0, & \text{otherwise}, \end{cases} \tag{13}$$

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

Algorithm 2. Local coverage path planning

Input: Volumetric map $\mathcal{M}$, Surface model $\mathcal{S}$, Local planning horizon $\mathcal{H}$, Budget L_B, Max iteration $\mathcal{N}_{cov}$, Max failure number $\mathcal{N}_{fail}$, Step size Δ_{cov}, Near radius R, Local exploration path $\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*)$;
Output: Local Coverage Path $\gamma_c^*(\mathrm{x}_r, \mathrm{x}_l^*)$;
$N_{itr} \leftarrow 0$, $N_{fail} \leftarrow 0$, $V_{closed} \leftarrow \emptyset$;
$\mathbb{G}$.**Initialize**($\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*)$);
while $N_{itr} < \mathcal{N}_{cov}$ and $N_{fail} < \mathcal{N}_{fail}$ **do**
  $\mathrm{x}_{rand} \leftarrow$ **Sample_Free_Space**($\mathcal{M}$, $\mathcal{H}$);
  $\mathrm{x}_{nearest} \leftarrow$ **Nearest_Node**($\mathbb{G}$, x_{rand});
  $\mathrm{x}_{feasible} \leftarrow$ **Steer**($\mathrm{x}_{nearest}$, x_{rand}, Δ_{cov});
  $\mathbb{N}_{near} \leftarrow$ **Near_Nodes**($\mathrm{x}_{feasible}$, $V \backslash V_{closed}$, R);
  for all $n_{near} \in \mathbb{N}_{near}$ **do**
    $\mathrm{x}_{new} \leftarrow$ **Steer**($\mathrm{x}_{n_{near}}$, $\mathrm{x}_{feasible}$, Δ_{cov});
    if Collision_Free($\mathcal{M}$, $\mathrm{x}_{n_{near}}$, x_{new}) **then**
      $c(\mathrm{x}_{new}) \leftarrow$ **Distance**(x_{near}, x_{new});
      $C_{\mathrm{x}_{new}} \leftarrow C_{n_{near}} + c(\mathrm{x}_{new})$;
      $\mathcal{U}_c(\mathrm{x}_{new}) \leftarrow$ **Coverage_Utility**($\mathcal{M}$, $\mathcal{S}$, $\mathcal{I}_c(\mathrm{x}_{near})$, $C_{\mathrm{x}_{new}}$);
      $n_{new} \leftarrow \langle \mathrm{x}_{new}, C_{\mathrm{x}_{new}}, \mathcal{U}_c(\mathrm{x}_{new}) \rangle$;
      if Prune(n_{new}) **then**
        Delete n_{new};
        $N_{fail} \leftarrow N_{fail} + 1$;
        continue;
      end
      $V \leftarrow V \cup n_{new}$;
      $E \leftarrow E \cup \{n_{near}, n_{new}\}$;
      if Distance(x_new, x_l^*) + $C_{\mathrm{x}_{new}} \geq L_B$ **then**
        $V_{closed} \leftarrow V_{closed} \cup n_{new}$;
      end
      $N_{itr} \leftarrow N_{itr} + 1$;
      $\mathbb{G} \leftarrow (V, E)$;
    end
  end
end
$\gamma_c^*(\mathrm{x}_r, \mathrm{x}_l^*) \leftarrow$ **Extract_Coverage_Path**($\mathbb{G}$, x_r, x_l^*, γ_c^{init});
return $\gamma_c^*(\mathrm{x}_r, \mathrm{x}_l^*)$.

where X_{pre} is the set of previous viewpoints and we have $\mathrm{x}_i \notin \mathrm{X}_{pre}$, $Vis(\mathrm{x}_i)$ represents the set of voxels can be observed at x_i. We can learn that $\mathcal{I}_c(\mathrm{x}_i)$ is the total number of *uncovered* surfaces and surface frontier voxels that have not been covered by any viewpoint in X_{pre} but can be covered at x_i.

To generate the local coverage path efficiently, we modify the rapidly exploring information gathering (RIG) algorithm [34] for solving Eq. (11). RIG is an informative path planning method that combines sampling-based motion planning with branch and bound algorithms to achieve efficient information gathering under motion constraints. The modified RIG for local CPP is presented in Alg. 2. We set the motion cost budget as $L_B = \lambda_2 \mathcal{L}(\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*))$, wherein λ_2 is used to balance the optimality of the resulting path and the real-time performance. When the local exploration path $\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*)$ is obtained, the algorithm first initializes a graph $\mathbb{G} = \{V, E\}$ using $\gamma_e^*(\mathrm{x}_r, \mathrm{x}_l^*)$ (Alg. 2, lines 3–4). After that, $\mathcal{N}_{cov}$ feasible nodes

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

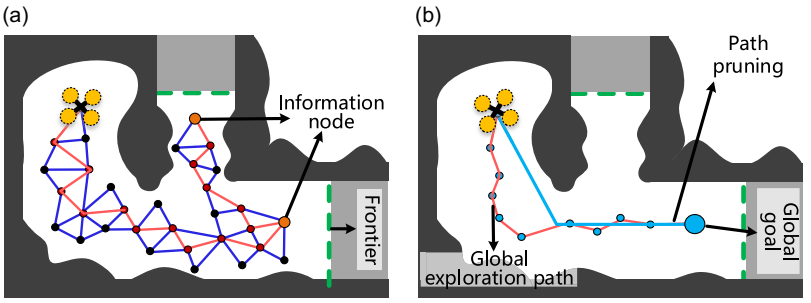


Figure 5. *Global exploration stage. (a) The history graph and the information nodes. (b) The path pruning process of the global exploration path at the current iteration.*

will be sampled for expanding the graph. Each time a random state x_{rand} is sampled, not only the nearest node on the graph will be extended toward x_{rand} but also all near nodes are trying to extend towards x_{rand} (Alg. 2, lines 6–9). Meanwhile, their submodular coverage utilities are recursively calculated along all the newly added edges by the function **Coverage_Utility**(), according to Eqs. (11)–(13) (Alg. 2, line 16). We also prune nodes that are not promising, e.g., those with negligible coverage utility, using the function **Prune**() (Alg. 2, lines 17–21). In addition, if the expected motion cost (**Distance**(x_new, x_l^*) $+ C_{x_{new}}$) of the newly extended node exceeds the budget L_B, it will be added to the *closed set* V_{closed} and will not be extended in the subsequent process to avoid invalid paths (Alg. 2, lines 24–26). Finally, the local coverage path with the largest coverage utility is recursively extracted from x_l^* to x_r on the graph $\mathbb{G}$ by the function **Extract_Coverage_Path**() (Alg. 2, line 32).

4.4. Global exploration stage

Similar to ref. [10], a sparse roadmap is continuously maintained during the MAV flight. The roadmap consists of sampled nodes and edges connecting these nodes. A node that contains the potential exploration information about its surrounding space is called *information node*. When there are no frontiers and *uncovered* surfaces within the local planning horizon, the planner turns to the global exploration stage, and the information nodes are used to guide the global exploration planning. As shown in Figure 5(a), the roadmap is continuously constructed during the MAV flight. The nodes of the roadmap contain two parts: the nodes directly sampled from the trajectory of the MAV (red solid circles) and the nodes sampled from the vicinity space around the MAV (black solid circles). The orange lines are the trajectory of the MAV, while the blue lines are the edges of the roadmap. When planning the global exploration path, the roadmap can efficiently return a feasible path towards any information node. However, the global exploration path may be tortuous and not appropriate for the MAV tracking. As shown in Figure 5(b), the path will be pruned to avoid detours and reduce the path length.

5. Results

5.1. Implementation details

We first conduct benchmark evaluations in a realistic simulator, Gazebo [35], since it can provide realistic simulation environments and robot models. As shown in Figure 6(a), a forward-looking depth sensor is mounted on the MAV provided by the open-source library, Rotors [36]. It has a min/max sensing range of $[0.5\,m, 5\,m]$ and field of view of $[90°, 110°]$ in the vertical and horizontal directions. We design two complex scenarios (see Figure 7), *Villa1* ($24\,m \times 20\,m \times 10, m$) and *Villa2* ($29m \times 25m \times 12m$), in Gazebo for the benchmark comparison. To achieve fair evaluations, we implement all algorithms on the same computer, which was configured with an AMD Ryzen 7 3700 CPU and 16-GB memory. The operating system is Ubuntu Linux 18.04 LTS, and the corresponding ROS version is Melodic. In all trials, the resolution r of TSDFs and ESDFs is set to 0.1 m. The maximum linear velocity and the

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

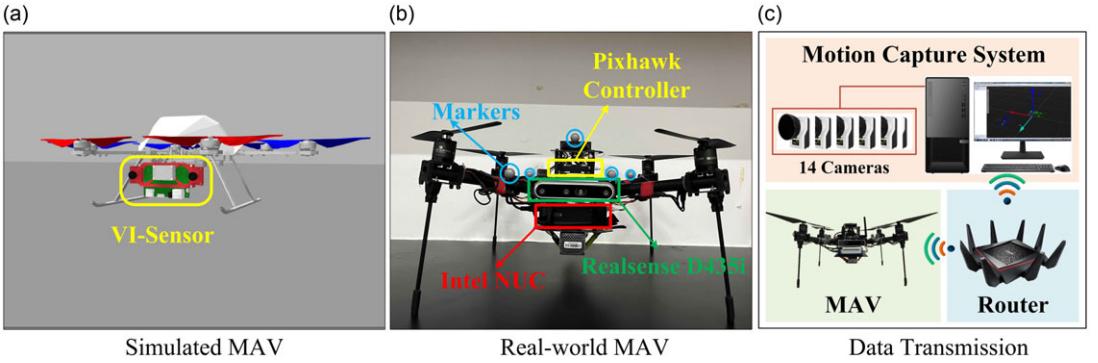


Figure 6. *Simulated and real-world robot platforms. (a) The simulated MAV is equipped with a forward-looking depth sensor, VI-Sensor. (b) The real-world MAV is equipped with a Pixhawk controller, a forward-looking Intel Realsense D455 sensor, and an Intel NUC computer. (c) The data transmission between the motion capture system and the real-world MAV is achieved through the local area network supported by a router.*

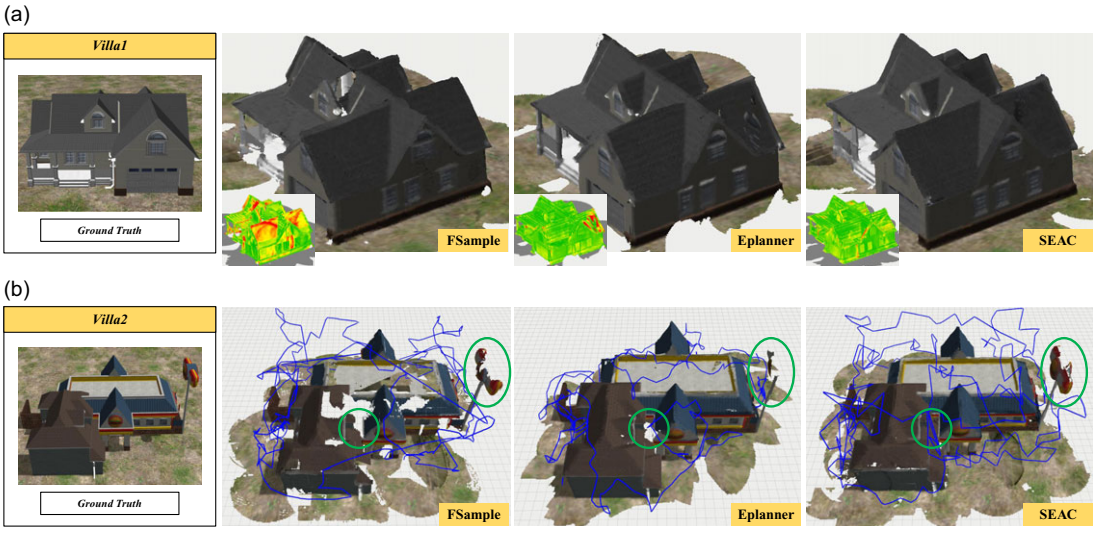


Figure 7. *Reconstruction results of the benchmark comparisons. (a) The obtained 3-D Models of Villa1 and their corresponding reconstruction error (left bottom, wherein red means high error and green means low error) after 600s. Since our method focuses on inspecting the reconstructed surfaces with low coverage quality, we obtain a higher overall accuracy and a more homogeneous error distribution. (b) Qualitative comparison (Reconstructed 3-D models with executed trajectories) of SEAC, FSample, and Eplanner in Villa2 after 900 s. Our method achieves the fullest reconstruction of the target, leaving the fewest holes on the mesh surface.*

maximum yaw rate of the MAV are set to 1.5 m/s and 1.0 rad/s. In order to improve the surface coverage quality, we set $\theta_{\max} = 60^\circ$ in Eq. (5), $d_{min} = 1\ m$ and $d_{max} = 4\ m$ in Eq. (6), and $l = 10$ in Eq. (7). The rest of parameters are summarized in Table I.

It is worth noting that, similar to ref. [26], we add quadratically-scaling Gaussian noise to the ground truth point cloud obtained from the simulator to account for perception uncertainties presented in the real-world scenes as follows:

$$z_{sim} = z + \mathrm{N}\left(\mu\left(z\right), \sigma\left(z\right)\right), \tag{14}$$

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

Table I. *Parameters used in the simulation experiments.*

Parameter	Value	Parameter	Value	Parameter	Value
λ_1	0.1	Δ_{exp}	1.0 m	$\mathcal{N}_{exp}$	15
λ_2	1.5	Δ_{cov}	1.0 m	$\mathcal{N}_{cov}$	15
$\mathcal{H}$	$8m \times 8m \times 8m$	R	2.0 m	$\mathcal{N}_{fail}$	10

where N () denotes the normal distribution function, z is the depth measurement, and μ and σ are the mean and standard deviation of the noise, respectively. In our trials, we set $\mu\ (z) = \sigma\ (z) = 0.0024\ z^2$.

To further validate the effectiveness of SEAC, we also carried out a real-world experiment in a laboratory environment on-board an MAV. Our self-built MAV employed in the real-world experiment is shown in Figure 6(b). The MAV is equipped with an Intel Realsense D435i depth sensor for acquiring the environmental data and an Intel NUC computer with an Intel Core i7-8550U CPU for running the proposed algorithm. We localize the MAV by a motion capture system and control the MAV using the SE(3) control algorithm [31]. As shown in Figure 6(c), the motion capture system comprises fourteen cameras and a ground station computer. It measures pose information for the MAV by identifying markers attached to the MAV. The MAV receives pose measurements from the motion capture system through the local area network supported by a router. The control signals are computed by the NUC computer and transmitted to the Pixhawk controller via the ROS-based communication protocol. For the parameters, we set $\mathcal{H} = 3\ m \times 3\ m \times 3\ m$. The other parameters used in the experiment are kept the same as those used in the simulation (see Table I).

5.2. Benchmark comparisons

We compare SEAC with the state-of-the-art methods for exploring or modeling unknown targets:

- *FSample* [10]: A method that combines the frontier-based and sampling-based strategies in a unified framework to achieve complete exploration of the environment.
- *Eplanner* [26]: A sampling-based method achieves global optimal decisions by continuously expanding a single trajectory tree, but only the coverage utility of the sampled viewpoints is mainly considered in each planing epoch.

Remark 1: FSample is a pure exploration method, while EPlanner is a state-of-the-art online 3-D modeling method that mainly considers the surface coverage quality in its planning. We select these two methods to demonstrate that SEAC takes into account both the coverage quality and the exploration efficiency. To achieve the statistical comparison, all algorithms are run five times in each scenario.

5.2.1. Villa1 reconstruction

The first trial utilizes the MAV to autonomously reconstruct the building *Villa1* online. Since the termination criteria of the evaluated methods are different, we set the maximum run time of each method in this scenario to 600 s. The comparison results of 5 runs are listed in Table II. In addition, Figure 7(a) shows the reconstructed 3-D models and the corresponding reconstruction error. It can be seen that FSample explored the largest volume of the environment (only 0.8% larger than SEAC) while resulting in the highest final reconstruction error (50.0% larger than SEAC). This is because FSample prioritizes maximizing the unknown volume that can be observed by the MAV in each iteration, while ignoring the reconstruction errors of the resulting 3-D model. By contrast, EPlanner explored the smallest volume of the environment and resulted in 31.4% larger reconstruction error than SEAC. This is because EPlanner primarily focuses on improving the surface coverage quality in each planning iteration, which leads the MAV to circle around the central building structure and, consequently, results in a low explored volume.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

Table II. *Result statistic of the benchmark comparisons.*

Scene	Method	Explored Volume (m^3)	Reconstruction Error (cm)	Path Length (m)
Villa1	FSample	**4120.2** ± 23.5	8.1 ± 1.0	316.0 ± 34.7
	EPlanner	3071.7 ± 153.9	7.1 ± 0.6	**257.8** ± **1.6**
	Ours	4088.4 ± **11.1**	**5.4** ± **0.2**	313.7 ± 11.3
Villa2	FSample	**7621.4** ± **37.6**	7.2 ± 0.5	483.2 ± 18.0
	EPlanner	4730.0 ± 98.8	5.6 ± 0.3	**339.1** ± **16.4**
	Ours	7591.5 ± 46.6	**4.9** ± **0.1**	452.1 ± 17.2

Table III. *Computation time used for each planning iteration.*

Scene	Method	Avg (ms)	Std (ms)	Max (ms)	Min (ms)
Villa1	FSample	**91.6**	63.1	410.0	**50.0**
	EPlanner	2054.7	632.4	4034.6	706.8
	Ours	119.1	**58.0**	**316.8**	56.1
Villa2	FSample	**167.5**	**58.2**	**551.1**	115.5
	EPlanner	1956.3	596.4	5262.1	379.1
	Ours	263.6	143.5	780.0	**70.0**

It is worth noting that EPlanner has the shortest trajectory. The reasons are twofold. First, the computational cost for expanding and maintaining a single tree in large environments is high, which might cause the MAV to stay in place and wait for the planning result. Second, EPlanner focuses too much on the low-quality surfaces, thus neglecting to explore the unknown space. On the other hand, our method and FSample have the similar trajectory length as well as similar explored volume.

5.2.2. Villa2 reconstruction

The second trial is to reconstruct *Villa2*. Since this scenario is larger and more complex than *Villa1*, we set the maximum run time of each method to 900 s. As shown in Table II, the results of this scenario are similar to the first one. Figure 7(b) shows the reconstructed 3-D models with the executed trajectories. Specifically, FSample explored the largest volume of the environment (only 0.4% larger than SEAC) while resulting in the highest reconstruction error (46.9% larger than SEAC). Similarly, EPlanner explored the smallest volume of the environment and resulting in 14.3% larger reconstruction error than SEAC. In addition, EPlanner exhibits the shortest trajectory, whereas our method and FSample demonstrate longer trajectories.

5.2.3. Computational time

As shown in Table III, we recorded the computational time for each planning iteration of each method used. It can be seen that the mean computational time of our method and FSample is an order of magnitude less than that of EPlanner. This is because FSample maintains a large single tree during each iteration for computing the globally optimal informative path, which is inherently expensive as it has to evaluate all the candidates to guarantee global optimality. On the contrary, both our method and FSample compute the MAV path within a local planning horizon and thus can reduce the computational time significantly. When comparing our method with FSample, we find that, despite FSample using slightly less time, their mean computational times are quite similar. The reason for our method taking more computational time is that it not only plans an exploration path but also further refines the path to enhance the coverage quality of the reconstructed surfaces.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

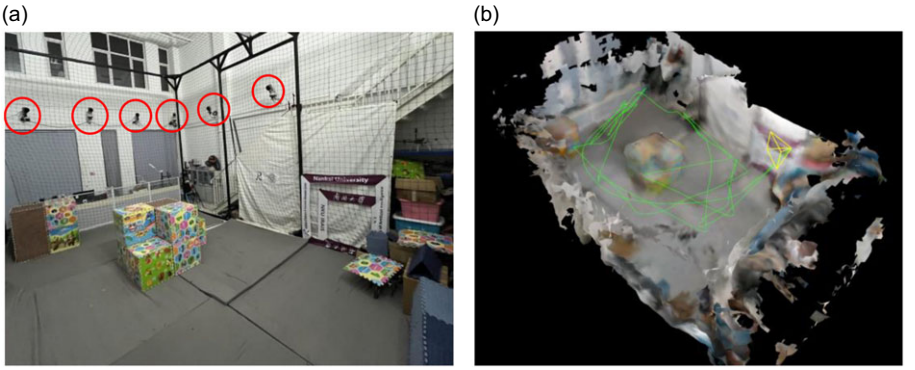


Figure 8. *Reconstruction result of the indoor experiment. (a) The configuration of the indoor environment includes a motion capture system made up of 14 cameras (as shown in the red circles). (b) The 3-D model of the indoor experiment is reconstructed by our method, and the green line represents the executed trajectory of the MAV.*

In summary, our method not only achieves the highest overall reconstruction accuracy but also produces the most complete 3-D models, leaving the fewest holes on the mesh surface. Moreover, our method is computationally efficient, which enables it to be used on resource-limited MAVs.

5.3. Real world experiment

In this section, we present the experiment for online 3-D modeling of an indoor environment (see Figure 8 (a)) by using our sefl-built MAV (see Figure 6 (b)). This size of the indoor environment is $7.0\,m \times 4.2\,m \times 2.5\,m$. In the experiment, we set the min/max sensing range of the Intel D435i sensor as $[0.2\,m, 3.5\,m]$ and the field of view as $[58°, 87°]$. The maximum linear velocity and the maximum yaw rate of the real MAV are set to $1.0\ m/s$ and $0.5\ rad/s$. According to the experiment result, our method can complete the online 3-D modeling task in $165\ s$ and thus obtain almost complete environmental surfaces. The mean computational time used for each planning iteration is $288.0\ ms$, with a standard deviation of $134.7\ ms$, demonstrating the online performance of SEAC. The average length of the MAV's trajectories over five runs is $59.5\ m$, with a standard deviation of $4.5\ m$. The reconstructed 3-D model of the representative run is shown in Figure 8 (b), in which the green line is the executed trajectory of the MAV.

More details of the benchmark comparisons and the real-world experiment can be found at https://youtu.be/Y23AMLUHxBU.

6. Conclusions

This paper proposes a novel online informative path planner for simultaneously exploring and covering unknown 3-D structures. Our method can achieve accurate online 3-D reconstruction without any prior information or offline processing. Specifically, we first devise a set of efficient metrics for evaluating the coverage quality of the reconstructed surfaces, which can facilitate the utility calculation of candidate viewpoints. Then, a hierarchical planner is proposed to cover the entire target by switching between local coverage stage and global exploration stage. The local coverage stage generates local coverage paths for inspecting surfaces with low coverage quality, and the global exploration stage is to output global exploration paths for global exploration. We compared our method with existing algorithms in two challenging scenarios. The results show that our method is capable of significantly reducing the reconstruction error and producing complete 3-D models. Finally, the real-world experiment demonstrated that our method can be run onboard a real MAV.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

The main limitation of our method is that we evaluate it using ground-truth localization, whereas in physical environments, localization errors cannot be ignored. In future work, we aim to further investigate SEAC for online 3-D modeling in scenarios suffering severe odometry drifts. Furthermore, existing literature, as well as our work, primarily relies on simulations or simple qualitative experiments for validating the 3-D modeling performance. Therefore, it is also crucial to design an effective strategy for the quantitative evaluation of SEAC methods in real-world scenarios.

Author contributions. Shiyong Zhang, Xuebo Zhang, and Qianli Dong designed the planner and conducted the code implementation. Qianli Dong, Tianyi Li, and Haobo Xi conducted the simulation comparisons and data gathering. Shiyong Zhang and Ziyu Wang conducted the real-world experiment. Shiyong Zhang, Xuebo Zhang, Chaoqun Wang, and Jingjin Yu wrote the article.

Financial support. This work was supported in part by Natural Science Foundation of China under Grant Number 62303249, in part by the China Postdoctoral Science Foundation-Tianjin Joint Support Program under Grant Number 2023T013TJ, and in part by the China Postdoctoral Science Foundation under Grant Number 2024M751526.

Competing interests. The authors declare no conflicts of interest exist.

Ethical approval. Not applicable.

References

[1] S. Zhang, X. Zhang, T. Li, J. Yuan and Y. Fang, "Fast active aerial exploration for traversable path finding of ground robots in unknown environments," *IEEE Trans. Instrum. Meas.* **71**, 1–13 (2022).
[2] H. Li, A. V. Savkin and B. Vucetic, "Autonomous area exploration and mapping in underground mine environments by unmanned aerial vehicles," *Robotica* **38**(3), 442–456 (2020).
[3] W. Tabib, K. Goel, J. Yao, C. Boirum and N. Michael, "Autonomous cave surveying with an aerial robot," *IEEE Trans. Robot.* **38**(2), 1016–1032 (2022).
[4] J. Yuan, Y. Huang, F. Sun and T. Tao, "Active exploration using a scheme for autonomous allocation of landmarks," *Robotica* **32**(5), 757–782 (2014).
[5] W. Jing, D. Deng, Z. Xiao, Y. Liu and K. Shimada. "Coverage Path Planning Using Path Primitive Sampling and Primitive Coverage Graph for Visual Inspection." **In:** *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2019) pp. 1472–1479.
[6] H. Song, J. Yu, J. Qiu, Z. Sun, K. Lang, Q. Luo, Y. Shen and Y. Wang. "Multi-UAV Disaster Environment Coverage Planning with Limited-Endurance." **In:** *2022 International Conference on Robotics and Automation (ICRA)* (2022) pp. 10760–10766.
[7] J. Tang and H. Ma, "Mixed integer programming for time-optimal multi-robot coverage path planning with efficient heuristics," *IEEE Robot. Autom. Lett.* **8**(10), 6491–6498 (2023).
[8] J. Delmerico, S. Mintchev, A. Giusti, B. Gromov, K. Melo, T. Horvat, C. Cadena, M. Hutter, A. Ijspeert, D. Floreano, L. M. Gambardella, R. Siegwart and D. Scaramuzza, "The current state and future outlook of rescue robotics," *J. Field Robot.* **36**(7), 1171–1191 (2019).
[9] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova and R. Siegwart, "Receding horizon path planning for 3D exploration and surface inspection," *Auton. Robot.* **42**(2), 291–306 (2018).
[10] V. M. Respall, D. Devitt, R. Fedorenko and A. Klimchik. "Fast Sampling-based Next-Best-View Exploration Algorithm for an MAV." **In:** *2021 IEEE International Conference on Robotics and Automation (ICRA)* (2021) pp. 89–95.
[11] Y. Tao, Y. Wu, B. Li, F. Cladera, A. Zhou, D. Thakur and V. Kumar. "SEER: Safe Efficient Exploration for Aerial Robots Using Learning to Predict Information Gain." **In:** *2023 IEEE International Conference on Robotics and Automation (ICRA)* (2023) pp. 1235–1241.
[12] Y. Zhao, L. Yan, H. Xie, J. Dai and P. Wei, "Autonomous exploration method for fast unknown environment mapping by using UAV equipped with limited FOV sensor," *IEEE Trans. Ind. Electron.* **71**(5), 4933–4943 (2024).
[13] B. Zhou, Y. Zhang, X. Chen and S. Shen, "FUEL: Fast UAV exploration using incremental frontier structure and hierarchical planning," *IEEE Robot. Autom. Lett.* **6**(2), 779–786 (2021).
[14] J. Yu, H. Shen, J. Xu and T. Zhang, "ECHO: An efficient heuristic viewpoint determination method on frontier-based autonomous exploration for quadrotors," *IEEE Robot. Autom. Lett.* **8**(8), 5047–5054 (2023).
[15] M. Sharma and H. K. Voruganti, "Multi-objective optimization approach for coverage path planning of mobile robot," *Robotica* **42**(7), 2125–2149 (2024).
[16] H. Wang, S. Zhang, X. Zhang, X. Zhang and J. Liu, "Near-optimal 3-D visual coverage for quadrotor unmanned aerial vehicles under photogrammetric constraints," *IEEE Trans. Ind. Electron.* **69**(2), 1694–1704 (2022).
[17] M. Roberts, S. Shah, D. Dey, A. Truong, S. Sinha, A. Kapoor, P. Hanrahan and N. Joshi. "Submodular Trajectory Optimization for Aerial 3D Scanning." **In:** *2017 IEEE International Conference on Computer Vision (ICCV)* (2017) pp. 5334–5343.

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press

[18] J. Liu, C. Wang, W. Chi, G. Chen and L. Sun, "Estimated path information gain-based robot exploration under perceptual uncertainty," *Robotica* **40**(8), 2748–2764 (2022).
[19] W. Chi, J. Wang, Z. Ding, G. Chen and L. Sun, "A reusable generalized voronoi diagram-based feature tree for fast robot motion planning in trapped environments," *IEEE Sens. J.* **22**(18), 17615–17624 (2022).
[20] C. Cao, H. Zhu, H. Choset and J. Zhang. "TARE: A hierarchical framework for efficiently exploring complex 3D environments," **In:** *Paper presented at the Robotics: Science and Systems Conference (RSS)*, Virtual, 12 to 16 July 2021 (2021).
[21] L. Heng, A. Gotovos, A. Krause and M. Pollefeys. "Efficient Visual Exploration and Coverage with a Micro Aerial Vehicle in Unknown Environments." **In:** *2015 IEEE International Conference on Robotics and Automation (ICRA)* (2015) pp. 1071–1078.
[22] D. G. Vutetakis and J. Xiao. "An Autonomous Loop-closure Approach for Simultaneous Exploration and Coverage of Unknown Infrastructure Using MAVs." **In:** *2019 International Conference on Robotics and Automation (ICRA)* (2019) pp. 2988–2994.
[23] S. Song and S. Jo. "Surface-based Exploration for Autonomous 3D Modeling." **In:** *2018 IEEE International Conference on Robotics and Automation (ICRA)* (2018) pp. 4319–4326.
[24] C. Feng, H. Li, F. Gao, B. Zhou and S. Shen. "PredRecon: A Prediction-boosted Planning Framework for Fast and High-quality Autonomous Aerial Reconstruction." **In:** *2023 IEEE International Conference on Robotics and Automation (ICRA)* (2023) pp. 1207–1213.
[25] X. Zhang, X. Xu, Y. Liu, H. Wang, X. Zhang and Y. Zhuang, "FGIP: A frontier-guided informative planner for UAV exploration and reconstruction," *IEEE Trans. Ind. Inform.* **20**(4), 6155–6166 (2024).
[26] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart and J. Nieto, "An efficient sampling-based method for online informative path planning in unknown environments," *IEEE Robot. Autom. Lett.* **5**(2), 1500–1507 (2020).
[27] M. Zhang, C. Feng, Z. Li, G. Zheng, Y. Luo, Z. Wang, J. Zhou, S. Shen and B. Zhou. "SOAR: Simultaneous exploration and photographing with heterogeneous UAVs for fast autonomous reconstruction." **In:** *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2024) pp. 10975–10982.
[28] M. Faessler, A. Franchi and D. Scaramuzza, "Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories," *IEEE Robot. Autom. Lett.* **3**(2), 620–626 (2018).
[29] F. Jiang, X. Zhang, X. Chen and Y. Fang, "Distributed optimization of visual sensor networks for coverage of a large-scale 3-D scene," *IEEE/ASME Trans. Mechatron.* **25**(6), 2777–2788 (2020).
[30] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart and J. Nieto. "Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-board MAV Planning." **In:** *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2017) pp. 1366–1373.
[31] T. Lee, M. Leok and N. H. McClamroch. "Geometric Tracking Control of a Quadrotor UAV on SE(3)." **In:** *49th IEEE Conference on Decision and Control (CDC)* (2010) pp. 5420–5425.
[32] S. Jung, S. Song, P. Youn and H. Myung. "Multi-layer Coverage Path Planner for Autonomous Structural Inspection of High-rise Structures." **In:** *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018) pp. 1–9.
[33] P. Vansteenwegen, W. Souffriau and D. V. Oudheusden, "The orienteering problem: A survey," *Eur. J. Oper. Res.* **209**(1), 1–10 (2011).
[34] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," *Int. J. Robot. Res.* **33**(9), 1271–1287 (2014).
[35] Gazebo. [Online]. http://gazebosim.org/. Accessed: May, 2025.
[36] F. Furrer, M. Burri, M. Achtelik and R. Siegwart, "Rotors—A Modular Gazebo Mav Simulator Framework," **In:** *Robot Operating System (ROS): The Complete Reference*, vol. **1** (Springer, Cham, 2016) pp. 595–625.

Cite this article: S. Zhang, X. Zhang, Q. Dong, T. Li, H. Xi, Z. Wang, C. Wang and J. Yu (2025). "SEAC: a simultaneous exploration and coverage planner for online aerial 3-D modeling", Robotica **43**, 2855–2871. https://doi.org/10.1017/S0263574725101860

https://doi.org/10.1017/S0263574725101860 Published online by Cambridge University Press