Autonomous Robots (2020) 44:1431–1450
https://doi.org/10.1007/s10514-020-09936-7

Online coverage and inspection planning for 3D modeling

Soohwan Song[1] · Daekyum Kim[1] · Sungho Jo[1]

Received: 30 June 2019 / Accepted: 15 July 2020 / Published online: 8 August 2020

Abstract

In this study, we address an exploration problem when constructing complete 3D models in an unknown environment using a Micro-Aerial Vehicle. Most previous exploration methods were based on the Next-Best-View (NBV) approaches, which iteratively determine the most informative view, that exposes the greatest unknown area from the current partial model. However, these approaches sometimes miss minor unreconstructed regions like holes or sparse surfaces (while these can be important features). Furthermore, because the NBV methods iterate the next-best path from a current partial view, they sometimes produce unnecessarily long trajectories by revisiting known regions. To address these problems, we propose a novel exploration algorithm that integrates coverage and inspection strategies. The suggested algorithm first computes a global plan to cover unexplored regions to complete the target model sequentially. It then plans local inspection paths that comprehensively scans local frontiers. This approach reduces the total exploration time and improves the completeness of the reconstructed models. We evaluate the proposed algorithm in comparison with other state-of-the-art approaches through simulated and real-world experiments. The results show that our algorithm outperforms the other approaches and in particular improves the completeness of surface coverage.

Keywords Active sensing · Exploration planning · Autonomous inspection · Next-best-view · Motion planning

1 Introduction

Reconstructed 3D models of large environments are becoming more useful in many industrial fields, including agriculture, engineering, and construction. With the development of various mobile robots, many studies suggest various methods to realize autonomous modeling systems (Blaer and Allen 2009; Ramanagopal et al. 2018; Roberts et al. 2017). Recently, because of rapid technological advances, Micro-Aerial Vehicles (MAVs) have become the most widely-used robots in the modeling systems. With their high maneuverability, MAVs can acquire information from almost any vantage points. However, due to their limited battery life, it is important to efficiently plan viewpoints when modeling a target environment.

The problem of computing the optimal trajectory of an MAV to reconstruct an environment is known as a view path planning problem. This problem is addressed differently depending on the availability of environment's prior geometric information. First, when prior information is available, inspection approaches (Englot and Hover 2012; Bircher et al. 2016; Li et al. 2012) are used. The inspection approaches precompute a view path that provides complete surface coverage of the prior model. These methods are able to provide an optimal solution for given geometric information of a target. However, prior information may not be available in many real-world situations. Second, when prior information is not available, the exploration approaches (Yamauchi 1997; Juliá et al. 2012; Vasquez-Gomez et al. 2017) are used. The exploration approaches incrementally complete a 3D model of unexplored regions. They iteratively plan view paths in an online manner from the partially acquired environment.

In this study, we focus on the MAV 3D exploration problem to model an unknown environment using a forward-facing depth sensor. Most exploration algorithms are based

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10514-020-09936-7) contains supplementary material, which is available to authorized users.

✉ Sungho Jo
shjo@kaist.ac.kr

Soohwan Song
dramanet30@kaist.ac.kr

Daekyum Kim
daekyum@kaist.ac.kr

[1] School of Computing, KAIST, Daejeon 34141, Yuseong-gu, Republic of Korea

on the Next-Best-View (NBV) method (Connolly 1985). This method iteratively determines a view configuration that provides the most informative from the current partially reconstructed environment. Some studies (Bircher et al. 2018; Charrow et al. 2015) determine the most informative sequence of views rather than a single optimal view. The "informativeness", the quantitative measurement of information, is estimated by analyzing unknown volumes (Bircher et al. 2018), frontiers (Blaer and Allen 2009; Yamauchi 1997), or information-theoretic measures (Moorehead 2001; Julian et al. 2014; Jadidi et al. 2015). All these approaches are greedy strategies that determine local solutions from current partial information. However, some factors of greedy strategies degrade the modeling performance. First, they focus only on sensing a large unknown region while ignoring less informative regions. Thus, the constructed model may be incomplete, and therefore inaccurately reconstructed regions, such as holes or sparse surfaces, are prevalent. Second, they do not consider a global route of an entire environment. Therefore, they may produce unnecessarily long trajectories that frequently overlap. This results in inefficient reconstruction performance in time by repeatedly revisiting known regions. Third, they ignore additional sensing targets that may emerge from the continuous model updates while a robot follows a planned path. For more accurate modeling, it is necessary to refine the planned path according to the updated information.

To address the aforementioned, this paper proposes a new exploration planning method that provides an inspection strategy to model an unknown structure. The proposed method integrates both global coverage and local inspection planning. For global coverage planning, an online map partitioning method is introduced. It decomposes an entire target space into uniformly distributed sectors representing a topological map. Then, it consistently refines shapes of the sectors based on currently updated map in real time. For every iterative planning step, a global coverage path of the entire environment is generated by computing a visitation sequence of the currently refined sectors. For the local inspection planning algorithm, heuristics that actively determine sampling region are proposed. This generates high quality inspection samples even with a small number of sampling points, which enables the existing inspection method to be processed online. The proposed local inspection method is to provide a path that fully covers local frontiers in the current iteration. The obtained path is iteratively replanned based on the currently updated map until completing the modeling of a local area. Integrating the individual components, the proposed method is to obtain a global coverage and local inspection path consistently to improve 3D modeling performances; it reduces the total moving distance and time during inspection and enhances the completeness of surface modeling. Figure 1 depicts the overview of the proposed exploration planning method.

1.1 Contributions and outline

This paper suggests the following contributions:

- Unlike past solutions, the proposed solution includes online coverage and inspection approaches for the exploration problem to completely model 3D environments.
- For global coverage planning, this paper proposes an online map partitioning method to construct a topological map. This method decomposes a map into sectors by clustering free space. To this end, a new distance measure for clustering is introduced. This enforces the compactness of complexity of each sector region.
- For online inspection planning, an informative sampling algorithm is introduced. This aims to incrementally reduce sampling regions through a streaming set cover algorithm (Emek and Rosén 2016). The advantages of the proposed method are verified through experiments.
- The proposed method is evaluated in both simulated and real-word environments. There are two sets of simulated scenarios: a classical 2D environment and 3D infrastructure. In the real-world scenario, the performances of the proposed method is compared with the performances of the offline inspection method (Englot and Hover 2012).

A preliminary version of this paper has been presented (Song and Jo 2017). In the current paper, we extended the previous study to take into account global coverage planning. The sector decomposition and coverage planning methods are newly presented. We provide more detailed explanations about the inspection algorithm and present a thorough evaluation of the method using various experimental scenarios. As compared to the previous study, we also experimentally addressed the effect of the global coverage planning.

The remainder of this paper is structured as follows. Sect. 2 presents the related work on mobile robot exploration and inspection. Section 3 describes the considered problem and its basic setup. Section 4 provides an overview of the proposed approach, which is divided into two stages. Global path planning is detailed in Sect. 5 and local inspection planning is detailed in Sect. 6. Sections 7 and 8 describe the simulation experiments and real-world experiments, respectively. Finally, in Sect. 9 we discuss our findings and the study limitations and in Sect. 10 summarize the contributions of this study.

2 Related works

In this section, we discuss prior studies regarding autonomous modeling systems for *exploration* and *inspection* of mobile robots. Approaches for the inspection task plan a coverage path offline because they assume that prior information about environments is given. In contrast, approaches for the exploration task assume that the target environment is unknown. Therefore, they consistently plan paths in an online manner according to acquired information.

2.1 Exploration planning

Existing approaches for the exploration problem are based on the NBV strategy, which iteratively selects the optimal viewpoint and computes a path to reach the next-best viewpoint. Frontiers are one of the most widely used evaluation metrics on NBV methods for mobile robot exploration. The frontier, first defined by Yamauchi (1997), represents the discrete border between an explored and an unexplored space. His approach always determines an exploration path toward the closest frontiers. The frontier approach has been extended to 3D volumetric representations and applied in MAV 3D exploration tasks (Cieslewski et al. 2017; Bircher et al. 2018; Charrow et al. 2015). Shen et al. (2012) suggested an indoor exploration method using MAVs by utilizing the expansion of a particle system, which is a similar/extended concept of the frontier. Cieslewski et al. (2017) determined target frontiers from the current Field of View (FoV) in order to explore a large area at high speed. Vasquez-Gomez et al. (2014) proposed a method that directly samples candidate views in the configuration space. The NBV is determined by evaluating the samples using a utility function considering some factors such as unknown volume, overlap and path distance.

Information-theoretic measures have also been used for NBV evaluations in robot exploration. Some studies aimed to minimize the map entropy (Moorehead 2001) or to maximize the mutual information (Julian et al. 2014; Jadidi et al. 2015) of future sensor measurements and the current map. These studies were focused on exploring only a 2D occupancy grid map constructed from the data provided by an omnidirectional range sensor. The computational cost of evaluating mutual information is high and scales linearly with the map's resolution and the dimension of the configuration space; thus, these methods are not appropriate for 3D exploration with sensors having a limited FoV.

Recently, several studies (Hepp et al. 2018; Wang et al. 2019; Zhu et al. 2018) employed machine learning methods for NBV planning. Hepp et al. (2018) proposed an exploration method that predicts the utility of a viewpoint via a 3D convolutional neural network (CNN). This 3D-CNN used a multi-scale voxel representation of the current volumetric map as an input to predict the utility score of a viewpoint. Wang et al. (2019) proposed an information gain metric for NBV determination, which integrates an entropy-based volumetric utility with a data-driven metric. The data-driven metric is estimated based on a 2D-CNN architecture instead of the 3D-CNN (Hepp et al. 2018). The 2D-CNN computes the ranked motion directions from a depth image input. Zhu et al. (2018) used deep reinforcement learning to learn topological and structural information of an office-like environment. This learned model enables the robot to compute a long-term visiting sequence for unexplored areas.

The objectives of several recent studies (Bircher et al. 2018; Charrow et al. 2015; Heng et al. 2015; Song et al. 2020) was to evaluate view sequences or optimize view paths toward the determined the next-best viewpoint. Shade and Newman (2011) computed the steepest descent path from a 3D vector field toward frontiers to obtain shorter 3D exploration paths. Bircher et al. (2018) proposed an exploration method based on a receding horizon that samples feasible configurations in a rapidly exploring random tree (RRT). The method extracts the most informative branch in the random tree and moves the first node of the branch. Charrow et al. (2015) proposed an information-theoretic planning method that determines the trajectory that maximizes the information-theoretic objective from global and local motions. Although this method takes into account the locally and globally uncertain parts of a map simultaneously, it does not consider their full coverage. Similar to our approach, Heng et al. (2015) considers the coverage of local unknown parts along the path to the NBV. For the coverage computation, their method requires precomputed 3D state lattices that contain motion-constrained edges and view frustums in every state. It also ignores updated sensing information for the path, whereas our approach incrementally refines the inspection path according to updates. Oßwald et al. (2016) introduced a method that computes a global exploration strategy from a topological graph provided by the user. The method significantly reduces the overall path length required to explore the entire environment. However, using no prior information our method generates a topological graph online and plans a global exploration strategy.

2.2 Coverage and inspection planning

Coverage planning is defined as a task of computing a mobile robot's path that guarantees of visiting all the points in a target area. In particular, inspection planning, referred to as visual coverage planning, is a sub-problem of coverage planning that determines a path by gathering the surface information of all the target points using vision sensors. In this paper, for clarification of meaning, we differentiate the *inspection planning problem* from the original *coverage planning problem*, where robots perform contact or passage operations.

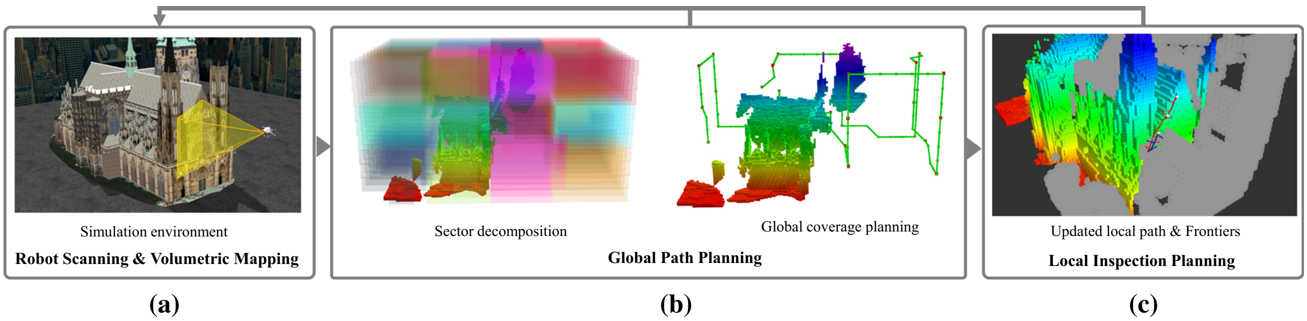


Fig. 1 Overview of the proposed 3D modeling system. **a** The micro-aerial vehicle explores a structure while simultaneously constructing its volumetric map. The proposed approach is composed of two steps. **b** First, a global path is determined by computing global coverage of the unknown space. To compute the global coverage, we decompose the entire map into a set of sectors and then compute the visitation order of the unexplored sectors. The global path to move toward the next sector is then determined. **c** The second step is local inspection planning, which provides an inspection path for the local frontiers near the global path. The inspection path is iteratively updated according to the updated frontiers

2.2.1 Coverage path planning

A large body of studies has been focused on coverage path planning in 2D environments for many robot applications, such as autonomous vacuum cleaners and lawn mowers, and for search-and-rescue missions. Most coverage methods decomposed the entire map into sub-regions or cells and then computed their visitation sequence. Some approaches exactly decomposed the map into non-overlapping sub-regions using trapezoidal (Oksanen and Visala 2009), boustrophedon (Choset 2000), or Morse cell decomposition (Choset et al. 2000). However, in several studies (Zelinsky et al. 1993; Gabriely and Rimon 2002) the map was approximately decomposed into grid cells and then their coverage was planned. 2D coverage algorithms have been extended to online applications (Yang and Luo 2004; Shnaps and Rimon 2014) or to non-planar surfaces in 3D spaces (Atkar et al. 2001, 2005). Atkar et al. (2001) utilized exact cellular decomposition for coverage of 3D structures. They decomposed an offset surface into cells, the boundaries of which were defined by the critical points of a series of 2D planners. Hess et al. (2012) investigated coverage planning of redundant manipulators to clean 3D surfaces. They computed coverage paths in terms of the amount of displacement of joint configurations rather than Euclidean distance of surface patches. These methods addressed coverage planning for 3D surfaces, whereas our approach is focused on 3D coverage of the entire free space. We constructed a topological map by decomposing the entire free space into disjoint sectors and then computes their coverage. Our approach is an online algorithm that updates the global coverage according to the incremental map construction.

The construction of a topological map is a major issue in coverage planning. Topological maps represent the entire environment as a graph consisted of the nodes, which are decomposed into sub-regions, and the edges, which are the connectivity between sub-regions. A constructed topological map is used to compute a coverage sequence of sub-regions. Each sub-region is covered by simple zigzag-like patterns (Brown and Waslander 2016; Das et al. 2014). Compact and convex decomposition of sub-regions is required for effective coverage planning. Brown and Waslander (2016) proposed a constriction decomposition method, which decomposes indoor environments based on narrow passages, called constriction points. The constructed topological maps are similar to the results of room-based decomposition. Das et al. (2014) proposed greedy convex polygon decomposition and a coverage planning framework. The framework is extended to a partially known search environment by iteratively re-planning the coverage path after the current sector has been fully searched. Liu et al. (2015) also proposed an incremental topological segmentation algorithm for topological mapping of partially known 2D environments. Blochliger et al. (2018) presented a 3D topological mapping framework for MAV path planning. They composed a topological map as a set of convex-shaped free spaces, which was constructed by clustering the free voxels from a given sparse feature-based map. Most approaches for online topological mapping (Das et al. 2014; Liu et al. 2015) were designed for 2D environments. The method for 3D topological mapping (Blochliger et al. 2018) requires prior information of the entire environment and does not operate online. In this paper, an online algorithm is proposed for constructing 3D topological maps for coverage planning in partially known environments. The algorithm decomposes a map into a set of sectors by clustering free spaces while enforcing the convexity of the cluster shape.

2.2.2 Inspection path planning

Traditional inspection methods plan a path offline using a prior 3D model of the environment. Cheng et al. (2008) simplified 3D urban structures into hemispherical and cylindrical models and planned time-optimal coverage trajectories for the simplified models. Englot and Hover (2012) introduced a sampling-based approach for inspecting the complex 3D structures of a ship's hull. The approach separately solves the coverage sampling and the multi-goal planning problem. The approach first performs coverage sampling, which determines the smallest set of views that guarantees full coverage, and then solves the multi-goal planning problem: visiting all the sampled views. Instead of selecting the smallest set of views, Bircher et al. (2016) attempted to resample the coverage viewpoints iteratively to reduce the distance cost of the overall trajectory.

Only a few studies have addressed online inspection planning in 3D environments. Vidal et al. (2017) proposed an online method for the inspection of underwater structures that does not require any prior information. Their method samples candidate viewpoints for sonar and camera sensors and then determines the nearest viewpoint as the NBV, similarly to the closest frontier method (Yamauchi 1997). This method determines a viewpoint only using local sensing information without considering global trajectories. Galceran et al. (2015) took localization uncertainty into account for inspection tasks using an underwater vehicle. Similarly to our method, their method iteratively computes an inspection path according to sensing measurements. However, their system assumes that a prior model of the target is provided. Also, it computes initial paths offline using the prior model. In contrast, our algorithm consistently provides an inspection path without using prior model, in real time and online.

Algorithm 1 Proposed exploration planning algorithm

Input: Volumetric map $\mathcal{M}$, Current configuration q_{curr}, Previous sector set $\overline{\mathbf{S}}$, and Initial sampling distance d_{sample}.
/* Global Path Planning */
1: $\mathbf{S} \leftarrow SectorDecomposition(\mathcal{M}, \overline{\mathbf{S}})$
2: $\mathbf{S}_{\Pi} \leftarrow SectorCoveragePlanning(\mathbf{S}, q_{curr})$
3: $[S_{curr}, S_{next}] \leftarrow Current\&NextSectors(\mathbf{S}_{\Pi})$
4: $R_{search} \leftarrow GetSearchRegion(S_{curr}, S_{next})$
5: $[q_{goal}, \xi_{global}] \leftarrow CompGlobalPath(q_{goal}, S_{next}, R_{search})$
/* Local Inspection Planning */
6: **while** $q_{curr} \neq q_{goal}$ **do**
7: $\quad V_{front} \leftarrow GetFrontierCells(\mathcal{M}, R_{search})$
8: $\quad$ **if** $|V^{new}_{front}| > \theta_{front}$ **then**
9: $\quad\quad [\xi_{local}, Queue_{Q^*}] \leftarrow InspectionPathPlanning(q_{curr}, q_{goal}, Queue_{Q^*}, V_{front}, d_{sample})$
10: $\quad$ **end if**
11: $\quad MoveToward(\xi_{local})$
12: $\quad Update(\mathcal{M}, q_{curr}, Queue_{Q^*})$
13: **end while**

3 Problem description and basic setup

In this study, we consider the exploration problem of an unknown and spatially bounded 3D space $V \subset \mathbb{R}^3$. We assume that the target structure is bounded within V. An MAV is to simultaneously construct a 3D model of the target structure by exploring 3D space V. We assume that the MAV is equipped with a forward-looking depth sensor that collects dense 3D data. The sensor has innate and userdefined constraints like a limited Field of View (FoV) and max/min sensing ranges. The estimated 3D data are integrated into a probabilistic volumetric map using an OctoMap (Hornung et al. 2013). The volumetric map $\mathcal{M}$ discretizes the entire space V into discrete volumes as octree structures. Each volume is classified to three space states: occupied $V_{occ} \subset V$, free $V_{free} \subset V$, and unknown $V_{unk} \subset V$. It simultaneously represents a workspace $W \subset \mathbb{R}^3$ and the volumetric model of an environment.

We assume that the configuration of the MAV is a flat state comprising a 3D position and yaw angle $q = \{x, y, z, \psi\}^T$ with zero roll and pitch (Bircher et al. 2018). We denote the maximum translational speed v_{max} and the rotational speed by $\dot{\psi}_{max}$. Both speed limits are set to be small to achieve increased sensing accuracy and exact path following. Paths that MAV navigates are computed in real-time. The paths are planned only within the known free space that guarantees a collision-free navigation. Let Q be a feasible configuration space, which contains all possible configuration of the MAV. We define a path $\xi : [0, 1] \rightarrow Q$ as a sequence of configurations. Let $V^* \subset V$ be as a set of all volumes that are visible from any configuration in Q. The volumetric model is complete when $V_{occ} \cup V_{free}$ in $\mathcal{M}$ is equal to V^*. Ultimately, our objective is to generate a complete volumetric model with the minimum exploration time.

4 Proposed method

To generate a 3D model of an environment, we iteratively plan an exploration path for an MAV by utilizing a current volumetric model $\mathcal{M}$ until the entire model is completely explored. Figure 1 shows an overview of our proposed approach and system. The proposed approach is a two-stage planning algorithm. The first stage constitutes global path planning, which determines the goal configuration and global path by computing the global coverage of the entire space. The second stage is local inspection planning. It provides an inspection path for the local frontiers near the global path. Both the global and local planning methods are online algorithms that can be operated on the partially known and consistently updated map. Using this approach, the MAV can rapidly explore the entire area while completely modeling the local surfaces of the target structure.

Algorithm 1 depicts the pseudocode of the proposed exploration planning algorithm, which is an iterative step in a loop. The algorithm first decomposes the entire space into a set of polygonal regions denoted as sectors $\mathbf{S}$ (line

1 and Sect. 5.1). It then estimates the global coverage of the unexplored area by computing a visitation sequence among the sectors (line 2 and Sect. 5.2). The global coverage is represented as a sequence of unexplored sectors $\mathbf{S}_{\Pi} = \{S_{\pi_1}, \ldots, S_{\pi_N}\}$, where $\Pi = \{\pi_1, \ldots, \pi_N\}$ is a permutation of the unexplored sector indices $\{1, \ldots, N\}$. The MAV sequentially constructs the model following the global coverage. The sector set and its coverage are consistently updated based on the changing information of the map. The search region R_{search} is defined as a region inside the sectors from a current sector S_{curr} to a subsequent sector S_{next} (line 4). The goal configuration q_{goal} is determined to be the sensor position that maximizes the potential visibility of unknown volume while being close to the center of S_{next}. The algorithm then computes the global path ξ_{global} from the current configuration q_{curr} to the q_{goal} (line 5 and Sect. 5.3).

Next, the algorithm plans the local inspection path ξ_{local} that provides maximal coverage of the frontier cells in R_{search}. The volumetric map $\mathcal{M}$ (line 12) and frontier cells V_{front} are consistently updated inside R_{search} (line 7). We define a frontier cell as the free volume adjacent to the unknown volume in $\mathcal{M}$. The inspection path is then iteratively planned according to the updated frontier cells (line 9 and Sect. 6). In each iteration, our inspection algorithm performs online refinement of the current local path ξ_{local} by maintaining a configuration set $Q^* \subset Q$. Q^* is a set of sampled configurations that composes the local path ξ_{local}. The sampled configurations are sequentially stored in a queue structure $Queue_{Q^*}$. The configurations that have already been visited by the MAV are removed from $Queue_{Q^*}$ (line 12). The refinement step of the inspection path is performed only if the total number of new frontier cells $|V_{front}^{new}|$ is greater than the constant value θ_{front} front (line 8). If the MAV reaches q_{goal}, the iteration is stopped. These planning steps for global and local paths are repeated until the model is completed.

5 Global path planning

Our approach consistently computes global coverage of the unexplored region to obtain a global path. In contrast to the other exploration methods (Yamauchi 1997; Bircher et al. 2018; Charrow et al. 2015), our method assumes the unknown volumes in $\mathcal{M}$ to be free and computes their coverage. To compute the global coverage, it first decomposes the map into a set of sectors $\mathbf{S}$. A sector $S_i \in \mathbf{S}$ is a volume, changeable in shape, which can be fully covered by the local inspection algorithm. We restrict the sector's maximum size, proportional to the maximum sensing range of the vision sensor. The suggested method then computes the coverage of all unexplored sectors. An unexplored sector is defined as a sector where more than a certain percentage of volumes has not been explored. Figure 2 depicts an execution example of the sector decomposition and global coverage planning. After the sector coverage has been computed, a goal configuration and global path are determined to move toward the next sector and to obtain the maximum visibility of the unknown volume. The global paths are computed within the known free space for a collision free navigation.

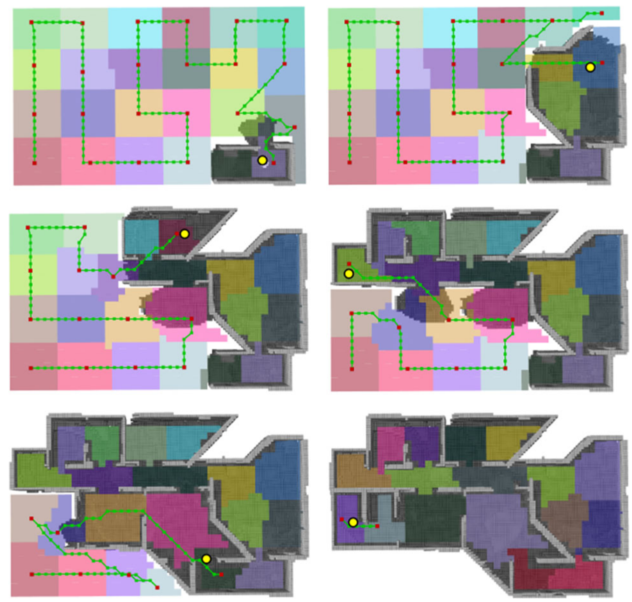


Fig. 2 Examples of sector decomposition and global coverage planning during exploration in an office-like environment (Fig. 5c). Each yellow circle shows the position of an MAV when each volumetric map is constructed. The shape of the sectors is iteratively refined according to the updated map. Each sector maintains a convex shape as far as possible. The shortest path that visits all the center positions of the unexplored sectors is then determined. (Color figure online)

5.1 Online sector decomposition

This section describes the method of dividing the entire map into a sector set, which is defined as a set of unknown and free volumes $V_{free} \cup V_{unk}$ in $\mathcal{M}$. We formulate the sector decomposition problem as a graph-partitioning problem. Let $G_{adj} = (X_{adj}, E_{adj})$ be a weighted, undirected adjacency graph, where the vertex set X_{adj} is composed of the set of unknown and free volumes with specific resolution $\theta_{adj-res}^{sector}$ and the edge set E_{adj} is composed of their connections. The given problem here is to decompose the vertex set X_{adj} into a collection of mutually disjoint non-empty sets $\mathbf{S} = \{S_1, \ldots, S_N\}$, such that all the vertices contained in a particular set are connected by their edges. The set of unknown and free volumes of X_{adj} are extracted by traversing the octree nodes of $\mathcal{M}$ with the fixed octree depth of the resolution $\theta_{adj-res}^{sector}$. The edge set E_{adj} is constructed with the 26-adjacency relation (Papon et al. 2013). The edge weight is defined as the Euclidean distance of adjacent vertices.

Algorithm 2 Online sector decomposition algorithm

Input: Volumetric map $\mathcal{M}$, and Previous sector set $\overline{\mathbf{S}}$.
/* Initialization */
1: $G^{adj} \leftarrow UpdateAdjGraph(\mathcal{M})$
2: $X^{seed} \leftarrow GetCenterVertices(\overline{\mathbf{S}}, G^{adj})$
3: $\mathbf{S} \leftarrow AssignToSectors(X^{seed})$
/* Assignment */
4: **while** $\mathbf{S}$ change **do**
5: **for** all $S_i \in \mathbf{S}$ **do**
6: $X_i^{nbr} \leftarrow GetAdjVertices(S_i, G^{adj})$
7: $RemoveOutOfMaxRangeVertices(X_i^{nbr}, x_i^{seed})$
8: $X*_i^{nbr} \leftarrow GetMinDistNBR(X_i^{nbr}, x_i^{seed}, \mathbf{S}, X^{seed})$
9: $S_i \leftarrow S_i \cup X*_i^{nbr}$
10: **end for**
11: **end while**
/* Post-Processing */
12: $ClusterRemainVertices(\mathbf{S}, G^{adj})$
13: $MergeSmallClusters(\mathbf{S})$
14: $ReExpandClusters(\mathbf{S})$
15: **return** $\mathbf{S}$

To solve this, we employ a local region growing method (Papon et al. 2013), which is a variant of the k-means clustering algorithm. This method generates a set of local clusters by incrementally expanding the cluster regions from each of the seed points distributed evenly in the map. The cluster expansion proceeds up to a certain restricted range, maximum expansion range. This generates compact and nearly uniformed clusters with a low computational overhead.

Similarly, the suggested method aims to expand each cluster region while enforcing the convexity of a cluster shape to the greatest extent possible, which enhances obstacle clearance and the sensor coverage performance in a local area. The method limits the search region and reduces the number of distance comparisons; this gives a significant speed improvement compared to a conventional k-means clustering algorithm that calculates the distances for all vertices. Furthermore, our approach expands a cluster region to the connected vertices at each step, so the additional connectivity check is not required. We provide a new distance measure that includes a metric for star-convexity for convex-shaped clustering. Furthermore, our approach is an online algorithm; it iteratively updates each sector region according to the changed environment. It computes a new seed point for the updated sector region and performs clustering again. The proposed sector decomposition algorithm is summarized in Algorithm 2.

5.1.1 Algorithm pipeline

The proposed algorithm first constructs the adjacent graph G_{adj} from the current map $\mathcal{M}$ (line 1). Vertices are rejected if they are unreachable from the current robot position. The reachability is estimated by traversing the adjacent graph using a breadth-first search starting from the current vertex. Volumes that are not included in G_{adj} are eliminated from the previous sector set $\overline{\mathbf{S}}$. Then, a set of seed points X^{seed} is initialized at vertices located at each center position of $\overline{\mathbf{S}}$ (line 2). If this is the first iteration with no previous sector set, the seeds are initialized as the center positions of the equal-sized rectangular cuboid grids in $\mathcal{M}$. Size of the cuboid grid in the xy-direction is set to be equal to the maximum expansion range while the size in the z-direction is 0.8 times the maximum expansion range, due to the flat state constraint of the MAV. Each seed point $x_i^{seed} \in X^{seed}$ is assigned to each cluster $S_i \in \mathbf{S}$ for the first time (line 3).

The algorithm incrementally expands each cluster set S_i to the neighboring vertices $X_i^{nbr} \subset X^{adj}$ by measuring their distance to seed point x_i^{seed}. The expansion to the neighboring vertices is sequentially processed for each cluster at each iterative step. When a sector region reaches the boundary of another sector, further expansion is achieved by assigning a vertex to a seed point with the minimum distance. Figure 3 illustrates the expansion of two clusters. To restrict a sector expansion to the maximum expansion range, out of range vertices are removed from X_i^{nbr} (line 7). The vertices that a cluster already visited are excluded from the neighboring vertex set of the cluster. The clustering operation is performed until there is no further expansion. This approach is similar to the breadth-first search of multiple starting points in the adjacency graph, and therefore guarantees the clustered vertices' connectivity and proximity properties.

At the end of this process, the algorithm clusters the unclustered vertices by randomly selecting a vertex and grouping it with neighboring unclustered vertices within the maximum expansion range (line 12). All the small clusters then are merged with the cluster closest to them (line 13). A small cluster is defined as a cluster having a size smaller than a specific volume. Finally, the algorithm determines the last sector shape by re-expanding the clusters to their neighbor vertices (line 14).

5.1.2 Distance measure

An existing clustering method (Papon et al. 2013) considers the spatial compactness of the cluster, but the convexity is not considered. The convexity of the cluster is an important factor to improve the performances of global and local planning. The convex-shaped clustering enhances the sensor coverage performance in a local area by reducing the possibility of blocking the sensor's field of view. The convex-shaped free space guarantees collision-free navigation of a robot with local obstacle clearance. Furthermore, each decomposed sector corresponds to a partially enclosed area such as a room; this area implicitly represents the vertices of a topological map. When assigning a vertex to a cluster, we consider the spatial proximity and cluster convexity simultaneously. We define the distance measure from a vertex x_k to a cluster S_i

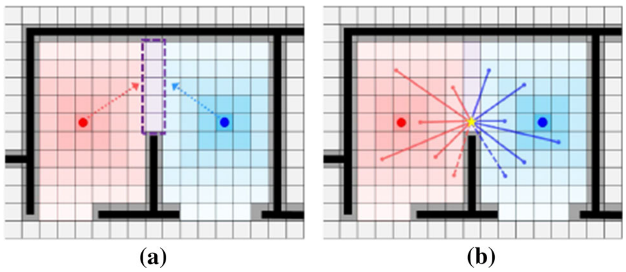


Fig. 3 Illustrations of proposed region growing process. **a** The algorithm incrementally expands each cluster region to the neighboring vertices, starting from each seed point (red and blue points). Vertices with the same color transparency represent neighbor sets processed simultaneously. Each vertex overlapped by more than two clusters (vertices colored purple) compares the distances to each seed point. **b** For the convexity measure of a target vertex to a cluster, the star convexity is evaluated by counting vertices, the line segment of which from the target lies in the cluster region. (Color figure online)

as a spatial distance between x_k and seed point x_i^{seed}, which is penalized by the convexity of S_i from x_k. Given a vertex x_k and a cluster set S_i, the distance function is defined as

$$D_G(S_i, x_k) = D_G^{spatial}(x_i^{seed}, x_k) + \alpha\{1 - D_G^{convex}(S_i, x_k)\} \quad (1)$$

where $D_G^{spatial}$ and D_G^{convex} are spatial distance and convexity measure functions, respectively. The convexity measure has a range [0, 1]; the higher the value, the more convex the cluster shape is. The α is a constant weight for the convexity penalty. The spatial distance $D_G^{spatial}$ is defined as the distance of the shortest path from x_i^{seed} to x_k in G_{adj}. Similarly to Dijkstra's algorithm, our algorithm maintains the distances of the shortest path at every expansion step.

We define the convexity measure D_G^{convex} as the star convexity for a cluster set S_i. A set X is star-convex with respect to a point c, if every line segment from c to a point $x \in X$ lies in the set X. Let $line(x_k, x_j)$ be a set of vertices composing a line segment from x_k to x_j. The star convexity for a sector S_i w.r.t a vertex x_k is defined as

$$D_G^{convex}(S_i, x_k) = \frac{1}{|S_i|} \sum_{x_j \in S_i} \mathbf{1}[line(x_k, x_j) \subset S_i] \quad (2)$$

where $|S_i|$ is the number of vertices in S_i and $\mathbf{1}$ is the indicator function. This measure simply represents the ratio of vertices that satisfy star convexity in S_i from a vertex x_k. Figure 3b illustrates the convexity measure of a vertex between two sectors. The dashed line is a line segment that does not satisfy star convexity. This effectively constructs convexshaped clusters, as shown in Fig. 2.

5.2 Sector coverage planning

After performing sector decomposition, a coverage path that visits every unexplored sector from a current location is planned. The coverage path is computed with the adjacency graph G^{adj} constructed in the previous step. The path in G^{adj} guarantees collision-free navigation, because the vertices are composed of reachable unknown and free volumes. The sector coverage problem is to find the shortest path that starts from a current vertex and visits all center vertices of the uncovered sectors in G^{adj}.

To compute the coverage, we employ an approach similar to that in Das et al. (2014), which applies metric closure to transform the coverage problem into a Hamiltonian path problem. The metric closure is used to compose an augmented graph $G^{aug} = (X^{aug}, E^{aug})$, where the vertex set X^{aug} is composed of vertices located at the center of each unexplored sector in G^{aug} and the edge set E^{aug} is their connections. Each edge has a cost obtained by calculating the shortest path distance between the vertices; the path is computed using an A* search in G^{adj}.

The complete graph G^{aug} is constructed by computing the all-pair shortest paths in G^{adj}. This augmented graph G^{aug} represents the topological map of the entire environment. Finally, the Hamiltonian path in G^{aug} is computed using a heuristic TSP solver (Helsgaun 2000). The resulting path provides the sector visitation order and the sector coverage is estimated by sequentially connecting the computed paths according to the order.

5.3 Goal determination and global path planning

To determine a goal, we first determine a sensor configuration q_{goal} that maximizes the utility function in the search region R_{search} from the current partial model $\mathcal{M}$. The configuration q_{goal} is determined as

$$q_{goal} = \underset{k=1,\ldots,N_{samples}^{global}}{\operatorname{argmax}} \; Util(q_k, R_{search}) \quad (3)$$

where $N_{samples}^{global}$ is the number of sampled configurations. The samples are directly generated in a feasible configuration space by extending the branches of an RRT* (Karaman and Frazzoli 2011) from the current configuration q_{curr}. We generate the sample configurations only within the free space in the search region R_{search}. This approach, similarly to that in Vasquez-Gomez et al. (2018) and Bircher et al. (2018), simultaneously processes the sample evaluation and path planning. Thus, all the samples in the RRT* can feasibly be reached and their paths are inherently collision-free.

We define the utility function as the unknown volumes that are visible from q_k, which is penalized by two distance measures $D_{\mathcal{T}}$ and D_E (Song and Jo 2018):

$$
\begin{aligned}
&Util(q_k, R_{search}) \\
&\quad = Vis(q_k, R_{search}) e^{-\lambda\{D_{\mathcal{T}}(q_{curr}, q_k) + D_E(q_k, S_{next})\}}
\end{aligned}
\tag{4}
$$

where $Vis(q_k, R_{search})$ is the volume of visible and unknown cells from q_k in R_{search}. The volume is estimated through ray casting in the view frustum of the sensor and counting the number of unknown visible cells. For the fast utility computation, the resolution of ray casting is set to a two-times lower resolution than the resolution of volumetric map. The parameter λ is a tuning factor penalizing long distance costs. $D_{\mathcal{T}}(q_{curr}, q_k)$ is the path distance between q_{curr} and q_k in the random tree $\mathcal{T}$. $D_E(q_k, S_{next})$ is the Euclidean distance between q_k and the center position of a sector S_{next}. The sample paths in $\mathcal{T}$ toward the center of S_{next} is assigned a higher weight value by $D_{\mathcal{T}}$ and D_E. Thus, the determined goal configuration q_{goal} covers large unknown volumes in R_{search} and explores toward the center of S_{next} simultaneously. After q_{goal} is determined, a global path ξ_{global} can be obtained by extracting edges from q_{curr} to q_{goal} in $\mathcal{T}$. The path ξ_{global} is then smoothed by using a path smoothing method, as in Kuffner and LaValle (2000).

6 Local inspection planning

This section describes the planning of a local inspection path that provides visual coverage of the local frontier cells. The proposed inspection path planning algorithm is detailed in Algorithm 3. The algorithm first computes the shortest path ξ_{short} from the current configuration q_{curr} to q_{goal} by using an RRT* planner and then performs path smoothing (line 1). If it is the first iteration of the loop of Algorithm 1 (line 6–13), ξ_{short} is equal to the global path ξ_{global}, so ξ_{global} is directly used. We define a sampling region $R_{sample} \subset W$ as a set of positions in radius d_{sample} centered at each discretized position of ξ_{short} (line 2). We restrict feasible sampling positions for path planning to R_{sample} to prevent a situation in which the length of the inspection path is significantly longer than ξ_{short}. As some frontier cells cannot be observed from any positions in R_{sample}, our algorithm determines the best path that provides a certain percentage θ_{cover}^{insp} of coverage rather than full coverage. If there is no sample set that satisfies a θ_{cover}^{insp}-coverage, the algorithm instead finds a path that gives the maximum coverage.

To compute the inspection path, we employ a sampling-based approach (Englot and Hover 2012). The sampling-based approach is efficiently applicable to problems involving a high-dimensional configuration space. Furthermore, it guarantees that the probabilistic completeness (Englot and Hover 2012), meaning the feasible solution, if such exists, will be eventually found by algorithm. The sampling-based approach is composed of a two-step optimization scheme. In the first step, the algorithm solves the coverage sampling problem aimed to determine an optimal set of configurations that cover the frontier cells. In the second step, it solves a multi-goal planning problem, which is aimed to compute the shortest path connecting all sampled configurations. The following subsections (Sects. 6.1 and 6.2) detail the coverage sampling algorithm and multi-goal planning framework, respectively.

Algorithm 3 Local inspection planning algorithm

Input: Current configuration q_{curr}, Goal configuration q_{goal}, Queue of configurations $Queue_{Q^*}$, Frontier cells V_{front}, and Initial sampling distance d_{sample}.

1: $\xi_{short} \leftarrow ComputeShortPath(q_{curr}, q_{goal})$
2: $R_{sample} \leftarrow GetSamplingRegion(\xi_{short}, d_{sample})$
    /∗ Informative Coverage Sampling ∗/
3: **for** $k = 1, \ldots, N_{sample}^{local}$ **do**
4:     $q_k \leftarrow GetFeasibleSample(Queue_{Q^*}, R_{sample})$
5:     $V_k \leftarrow Visible(q_k, V_{front})$
6:     $T_k \leftarrow \mathrm{argmax}_{T_i \subset V_k}\, lev(T_i)$
7:     **for** all $v_i \in T_k$ **do**
8:         $eid(v_i) \leftarrow k$
9:         $eff(v_i) \leftarrow lev(T_k)$
10:     **end for**
11:     $Q^* \leftarrow UpdateSampleSet(eid(\cdot))$
12:     $d_{max} \leftarrow \max_{q_i \in Q^*} w_i$
13:     **if** $(CovRatio(Q^*) > \theta_{cover}^{insp})\&(d_{max} < d_{sample})$ **then**
14:         $d_{sample} \leftarrow d_{max}$
15:         $R_{sample} \leftarrow GetSamplingRegion(\xi_{short}, d_{sample})$
16:     **end if**
17: **end for**
    /∗ Multi-Goal Planning ∗/
18: $\xi^* \leftarrow SolveTSP(\{q_{curr}, q_{goal}\} \cup Q^*)$
19: $[\xi^*, Queue_{Q^*}] \leftarrow SmoothPath(\xi^*)$
20: **return** ξ^* and $Queue_{Q^*}$

6.1 Informative coverage sampling

The goal of the coverage sampling problem is to determine a set of sample configurations that covers more than a certain percentage θ_{cover}^{insp} of the frontier cells. The ideal coverage set consists of a small number of samples with high proximity to the shortest path. We represent the problem as a weighted set cover problem in a set system (V, Q), where V is a finite set of frontier cells and Q is the robot configuration space. Every feasible configuration $q_k \in Q$ maps to a subset $V_k \subset V$ viewed by the sensor (line 5) and has its own weight. The weight w_k is defined as the proximity of q_k and ξ_{short} and can be computed as

$$
w_k = \min_{q_i \in \xi_{short}} D_E(q_k, q_i) \tag{5}
$$

where q_i is a discretized configuration in ξ_{short}. When each configuration q_k is sampled individually, the goal is to con-

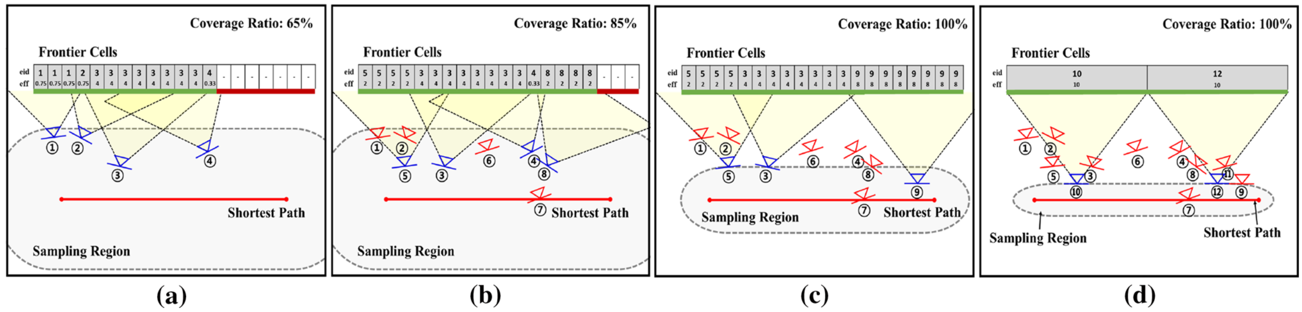


Fig. 4 Illustration of the proposed algorithm's use of the streaming set cover approach (Emek and Rosén 2016) for incrementally reducing a sampling range and improving the coverage sampling problem. The processes are described in 2D for clarity. The red line and the gray ellipsoid represent the shortest path and sampling region, respectively. The successive boxes are frontier cells. Each box stores an identifier eid and effectiveness eff. Each configuration is sequentially sampled, from ① to ⑫. **a** Configurations ① to ④ are assigned to each eid of a frontier cell. **b** The configurations ③ ⑤ ④ ⑧ are included in a suboptimal coverage set Q^*, while ① ② ⑥ ⑦ are excluded. **c** If the coverage ratio reaches 100%, the algorithm starts reducing the sampling region. **d** Finally, the best coverage samples are obtained. (Color figure online)

struct a set of configurations $Q^* \subset Q$ that provides the maximum cover of V with the objective of minimizing the sum of their weights.

Unlike the previous method (Englot and Hover 2012), in this method, the inspection paths should be planned online; therefore, a sufficiently large number of samples cannot be evaluated. Moreover, the suggested algorithm iteratively replans the path according to the changed environment, and therefore, it is important to consider the existing samples from the previous path. Thus, an informed sampling method, which efficiently utilizes the heuristic information such as existing samples and local solutions, is required.

In this study, we employed a streaming set cover algorithm (Emek and Rosén 2016) to perform the informed sampling. The algorithm investigates the set cover problem under the semi-streaming model (Feigenbaum et al. 2005), where the configurations arrive one-by-one. It sequentially processes the configurations and iteratively determines a suboptimal set-cover solution at each step. The streaming algorithm determines coverage samples Q^* that δ–covers frontiers (at least δ% of frontier coverage) with the objective of minimizing $\sum_{v_k \in Q^*} w_k$ and maximizing δ. The algorithm gives $O(\min\{1/(1-\delta), \sqrt{N}\})$-approximation for the original set cover problem where N represents the total number of frontier cells. Each input sample q_k is processed in $O(|V_k| log(|V_k|))$ time where $|V_k|$ is the number of mapped frontiers. The algorithm assigns two variables for each frontier cell $v_i \in V_{front}$: an identifier $eid(v_i)$ of a sample q_k that representatively covers it and a positive variable $eff(v_i)$ that intuitively captures the effectiveness of q_k in covering v_i. For each iterative step, a suboptimal solution is consistently maintained by updating the variables. The suboptimal solution is estimated by extracting the configurations assigned to one more frontier cell from $eid(\cdot)$.

Similarly to the streaming algorithm (Emek and Rosén 2016), our sampling approach (line 3–17) repeatedly processes sampled configurations individually and outputs them for every frontier cell $v_i \in V_{front}$, identifier $eid(v_i)$, and effectiveness $eff(v_i)$. Thus, we consistently maintain a suboptimal coverage set Q^* online. To compute the effectiveness of a sample q_k, we define a level of a subset $T_k \subset V_k$ as

$$lev(T_k) = \frac{|T_k|}{\beta \cdot w_k} \tag{6}$$

where β is a constant value, and $|T_k|$ is the number of elements in T_k. Subset T_k is said to be effective if every $v_i \in T_k$ satisfies $lev(T_k) > eff(v_i)$. For each v_i in an effective set T_k, we assign the ID of sample q_k to $eid(v_i)$ and $lev(T_k)$ to $eff(v_i)$ (lines 7–10). If $lev(T_k) = 0$, the sample q_k is eternally rejected.

The key advantage of this online set cover approach is that it consistently maintains a suboptimal solution Q^* in each iteration. The major difference from Emek and Rosén (2016) is that our method actively determines the sampling region from the suboptimal solution. We can estimate the sampling region of Q^*, which can be used to decrease the size of the sampling domain to possibly improve the solution. Thus, we can efficiently sample a configuration by incrementally reducing the sampling range R_{sample}. First, a sample q_k is sequentially extracted from $Queue_{Q^*}$, which contains the sampled configurations in a previous inspection path. After all samples in $Queue_{Q^*}$ have been extracted, a uniform sample is iteratively generated in R_{sample} (line 4). In each step, a coverage ratio of Q^* is computed. If the ratio is greater than the threshold θ^{insp}_{cover}, the samples are regarded as a first solution of the coverage sampling problem and the algorithm starts reducing R_{sample} (line 13). Let d_{max} be the maximum weight of a sample from Q^*; then, set d_{sample} to d_{max} and recompute R_{sample} (lines 14 and 15). This approach finds effi-

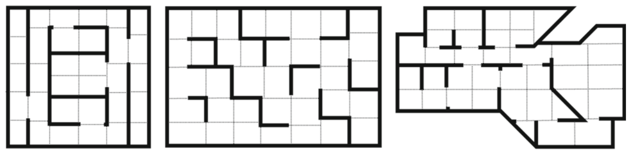

(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Fig. 5 Map layouts of three office-like environments used in the simulator. Scenario 1 has dimensions $30 \times 30 \times 3$ m^3. Scenarios 2 and 3 have the same dimensions $38 \times 23 \times 3$ m^3. The entire area is decomposed into uniformly distributed zones, which are illustrated through the dashed lines. The micro-aerial vehicle explores each map until all zones are visited

cient samples that are closer to the shortest path than those found by the original coverage-sampling method (Englot and Hover 2012). If the first solution is not found in line 13, our algorithm performs the general coverage sampling without informative sampling; the results gives the maximum coverage solution. Figure 4 illustrates the proposed coverage-sampling approach.

6.2 Multi-goal planning

The final inspection path ξ^* is extracted by computing the shortest connecting path over all configurations $q \in Q^*$ by using the TSP solver (Helsgaun 2000) (line 18). We define a connection cost $cost(q_i, q_j)$ of the TSP solver as the execution time of motion (Bircher et al. 2016):

$$cost(q_i, q_j) = max(D_E(q_i, q_j)/v_{max}, \|\psi_i - \psi_j\|/\dot{\psi}_{max}) \tag{7}$$

where $D_E(q_i, q_j)$ is the Euclidean distance directly connecting pairs q_i and q_j. If the connection has a collision, the RRT* planner is used to connect them. Although all configurations are sampled in free space, there may be configurations that are not reachable by the MAV. The unreachable configurations from Q^* are rejected.

After the initial path ξ^* has been extracted, a path smoothing step is performed. In this step, positions and yaws in ξ^* are refined sequentially while preserving the visibility of the corresponding frontiers. First, the positions are refined and the path length is shortened by using the heuristic speed-up improvement procedure in Englot and Hover (2013). If the Euclidean distance of the two subsequent configurations is close, the execution time of the motion depends on the time of yaw rotation, so yaw directions should be smoothed. Each yaw direction is then refined according to the criterion proposed in Bircher et al. (2016). This yaw refinement is additionally applied from the previous algorithm (Song and Jo 2017), which minimizes the decrease in the movement speed of the MAV in yaw rotation. Finally, each configuration in the smoothed path is sequentially stored in $Queue_{Q^*}$.

Table 1 Parameters used in office-like environment scenarios

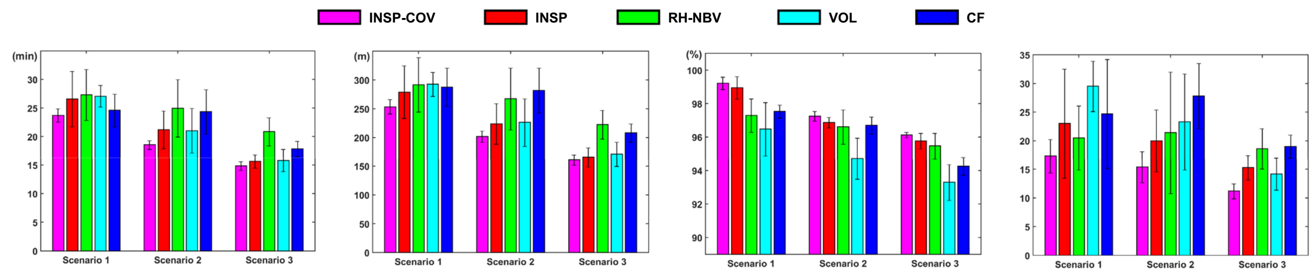
Parameter	Value	Parameter	Value
Resolution of $\mathcal{M}$	0.3 m	d_{sample}	1 m
Max sensing range	5 m	θ_{front}	20
v_{max}	0.2 m/s	$\theta^{sector}_{adj-res}$	1.2 m
$\dot{\psi}_{max}$	0.5 rad/s	θ^{insp}_{cover}	90%
RRT edge length	1 m		

7 Simulation experiments

In this section, we conducted experiments under simulated environments to evaluate the proposed approach. We employed the simulation system used in Bircher et al. (2018), which uses the model of a Firefly hexacopter MAV in the RotorS simulation environment (Furrer et al. 2016). We conducted two sets of comparison experiments. In the first set, a classical 2D exploration performance was evaluated in office-like environments (Sect. 7.1). In the second set, the 3D exploration performance of our approach was evaluated by modeling a large infrastructure (Sect. 7.2). We then evaluated the proposed informative sampling method (Sect. 7.3) and analyzed the computation time of the whole process (Sect. 7.4).

In the comparison experiments, we separately evaluated the performances of the proposed inspection approach with and without the use of global coverage to measure the benefits of the global coverage planning described in Sect. 5. The inspection approach without the global coverage is the same as the method proposed in our previous study (Song and Jo 2017), which is referred to as **INSP**. We refer to the approach with global coverage as **INSP-COV**. The proposed approach was compared with three state-of-the-art methods:

- *Closest frontier method (CF)* (Yamauchi 1997): This method has most frequently been used in 2D exploration applications. We extended the method to 3D environments. We first cluster adjacent frontier cells within a certain distance and then choose the closest cluster as a target of the NBV.
- *Volumetric method (VOL)* (Vasquez-Gomez et al. 2014): This method directly moves to the configuration with the highest utility, similarly to the global planning method described in our previous study (Sect. 6.1 in (Song and Jo 2017)).
- *Receding-horizon NBV (RH-NBV)* (Bircher et al. 2018): This is the most recently developed exploration algorithm for autonomous modeling. It evaluates an exploration path by directly expanding an RRT and moves to the first edge of the best branch of the RRT.



(a) Completion Time **(b)** Path Length **(c)** Covered Volume **(d)** No. of Revisits

Fig. 6 Results in office-like environment scenarios. Comparison of **a** completion time, **b** path length, **c** percentage of covered volume, and **d** number of zone visits of proposed methods, INSP-COV and INSP, and three alternative methods, RH-NBV (Bircher et al. 2018), VOL (Vasquez-Gomez et al. 2014), and CF (Yamauchi 1997). INSP-COV consistently shows the best performance with the minimum standard deviation in every evaluation criteria. The number of revisit for INSP-COV is the lowest in every scenario; this reduces the completion time and path length in comparison to other methods

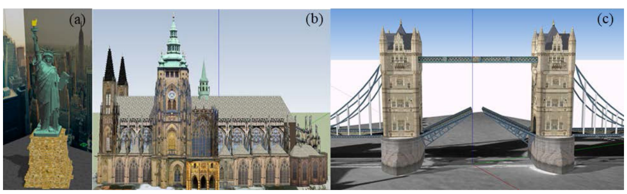


Fig. 7 Three simulated environments of the infrastructure modeling scenarios in the robot operating system (ROS) simulator. **a** Scenario 4: statue of liberty model ($16 \times 16 \times 35\,\mathrm{m}^3$), **b** Scenario 5: St. Vitus Cathedral model ($43 \times 23 \times 25\,\mathrm{m}^3$), and **c** Scenario 6: Tower Bridge model ($18 \times 65 \times 25\,\mathrm{m}^3$)

For every method, the MAV performs continuous scanning and mapping while following a planned path. RH-NBV, VOL, and our approach evaluate the utility while extending a random tree for global path planning. The parameters related to the random tree and the evaluation were applied to each algorithm in the same manner. The number of samples for the random tree was set to $N^{global}_{sample} = 500$ so that the samples would be sufficiently distributed in the entire area of the map. The number of samples for the local inspection planning was set to $N^{local}_{sample} = 500$. The edge length of the tree was set to a different value based on the map scale of each scenario. The λ parameter for the utility evaluation was set to 0.3. For the fast utility computation, the number of unknown cells was counted by using ray casting in a two-fold lower resolution map. For RH-NBV, the RRT cannot be extended when the MAV is located in the middle of a fully explored area, because it rejects a sample having zero utility. Therefore, we modified the RRT in RH-NBV to also extend the zero-utility samples.

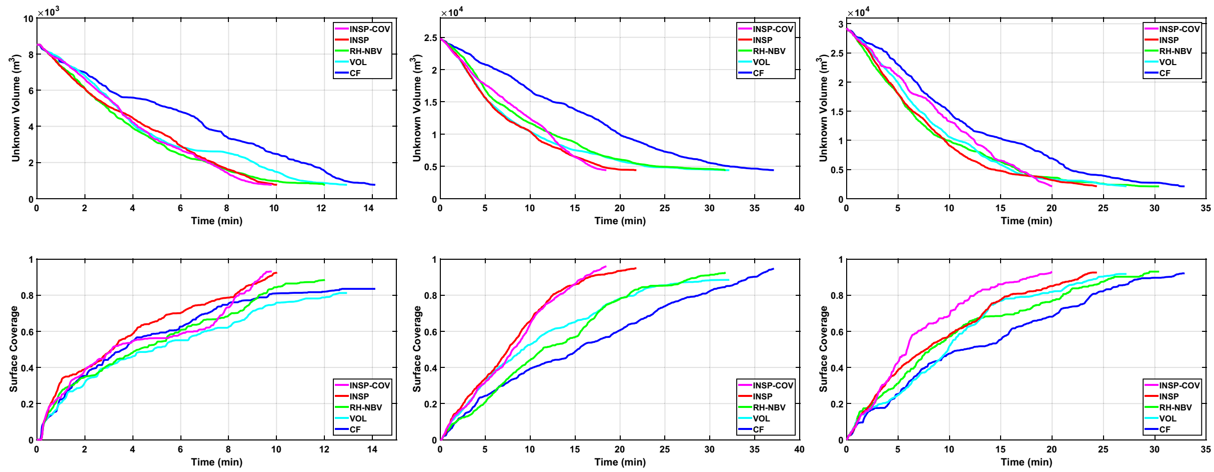
7.1 Exploration in office-like environments

In this experiment, we performed simulations for three different scenarios to evaluate the performance of a 2D exploration task. Figure 5 shows the simulation environments. Scenario 1 (Fig. 5a) is composed of narrow corridors and rooms. Scenario 2 (Fig. 5b) is composed of only relatively wide corridors. Both Scenarios 1 and 2 refer to the map layouts in Oßwald et al. (2016). Scenario 3 (Fig. 5c) is composed of various open rooms, which is the same map as the office scenario in Cieslewski et al. (2017). The volumetric maps of the structured and flat office environments were constructed using each exploration algorithm. Table 1 summarizes the parameters used in the scenarios.

Starting from each fixed position, the MAV explores the entire map. As shown in Fig. 5, we manually divided the entire area of the map into uniformly distributed zones for performance evaluation. When the MAV had visited all the zones, we stopped the exploration and evaluated the performance. For each algorithm, we computed the exploration completion time, path length, percentage of covered volume, and number of zone revisits. The number of zone revisits was computed by counting the number of visits to a zone that the MAV had already visited. This represents the coverage efficiency of the path. Figure 6 shows the results, where each result is the average of 10 executions.

RH-NBV shows the poorest performance in terms of completion time in all three scenarios. RH-NBV frequently changes the best branch, and therefore generates back-and-forth reciprocating motions. VOL shows the poorest coverage performance, although its performance is similar to that of INSP in terms of completion time and path length. VOL generates the simplest and fastest path, because it directly moves to the largest unknown region. However, it frequently misses small, unknown regions. Our methods fully cover the local regions and therefore show high coverage performances.

INSP-COV has the best performance of every evaluation criterion. As can be seen in Fig. 6d, the number of revisit for INSP-COV is the lowest in every scenario, which reduces the completion time and path length in comparison



(a) Scenario 4 **(b)** Scenario 5 **(c)** Scenario 6

Performances when 95% of the map had been explored.

	Scenario 4			Scenario 5			Scenario 6		
	Time (min)	Path Length (m)	Surf Cover (%)	Time (min)	Path Length (m)	Surf Cover (%)	Time (min)	Path Length (m)	Surf Cover (%)
INSP-COV	**9.81**	183.3	**93.10**	**18.48**	**366.6**	**96.07**	**20.05**	**383.0**	92.87
INSP	10.02	**168.9**	92.60	21.82	430.5	94.99	24.36	446.2	92.53
RH-NBV	12.02	223.5	88.40	31.74	607.1	92.58	30.42	532.1	**93.04**
VOL	12.94	246.3	81.20	32.16	627.5	88.48	27.24	484.1	91.73
CF	17.69	295.7	83.50	37.12	802.5	94.79	32.90	714.3	92.12

Fig. 8 The graphs in (**a**), (**b**), and (**c**) show unknown volumes and the surface coverage of the target over time in the infrastructure modeling scenarios. Note that the reduction rates of the unknown volume in INSP-COV are near constant throughout the explorations. The table shows the average performances when each algorithm had completed 95% of exploration. INSP-COV in general outperforms the other approaches. Especially, INSP-COV shows the highest average surface coverage, 94.01%

Table 2 Parameters used in infrastructure modeling scenarios

Parameter	Value	Parameter	Value
Resolution of $\mathcal{M}$	0.3 m	RRT edge length	2 m
Max sensing range	Scenario 4: 8 m	d_{sample}	2 m
	Scenario 5: 8 m	θ_{exp}	95%
	Scenario 6: 10 m	θ_{front}	100
v_{max}	0.4 m/s	$\theta^{sector}_{adj-res}$	1.2 m
$\dot{\psi}_{max}$	0.5 rad/s	θ^{insp}_{cover}	90%

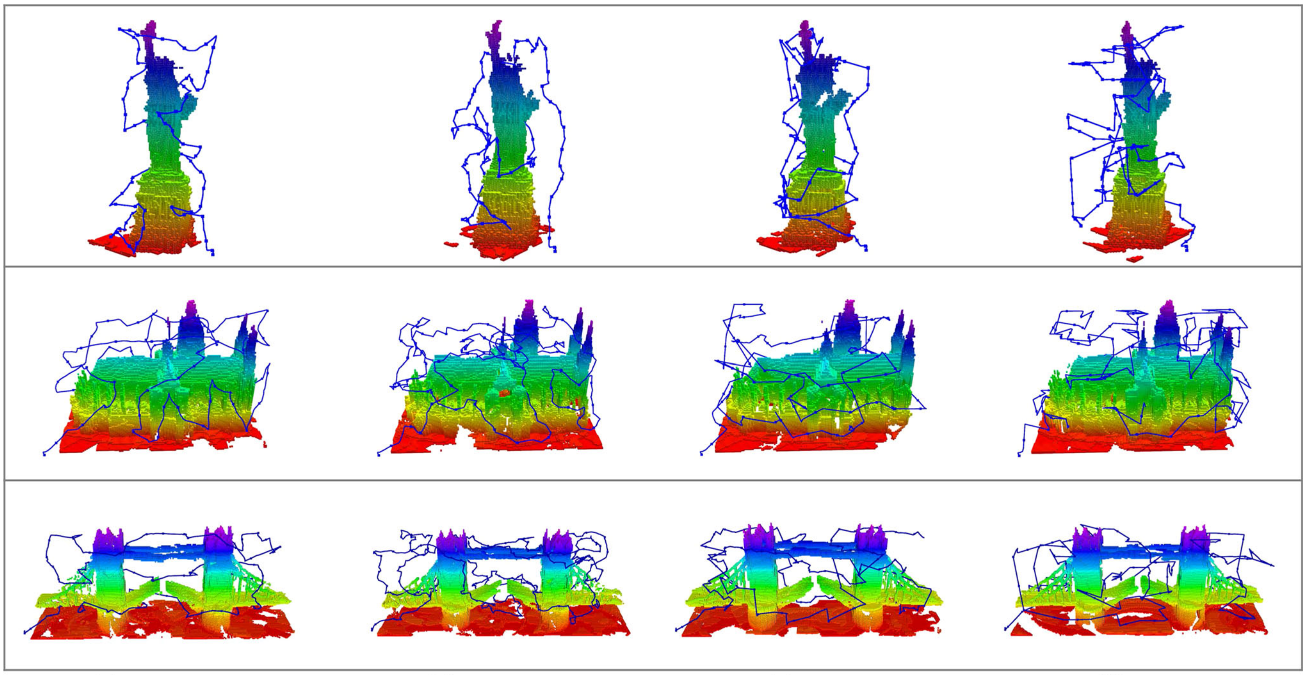
with other methods. In particular, as compared to INSP, it reduces the average exploration time by 9.45% and the average path length by 7.29%. In addition, it can be confirmed that the robustness of the global coverage method is increased because it has the minimum standard deviation values in all evaluation criteria.

7.2 Modeling large infrastructures

In this experiment, we evaluated the exploration performance of each approach in terms of the modeling quality. We considered three target infrastructures[1] placed on the ground within four walls (Fig. 7). The first target was Scenario 4 : *Statue of Liberty* (Fig. 7a), which is a simple cylindrical-like structure. The second was Scenario 5: *St. Vitus Cathedral* (Fig. 7b), whose structural size is larger and more complex than that of the first model. The final target was Scenario 6: *Tower Bridge* (Fig. 7c), which is composed mostly of open free spaces.

Table 2 summarizes the parameters used in the scenarios. Because of the increased size of the maps, we set larger parameter values of the RRT edge length (2 m) and the

[1] http://3dwarehouse.sketchup.com/.



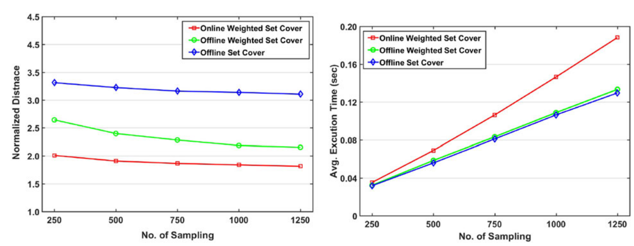
(a) INSP-COV **(b)** RH-NBV **(c)** VOL **(d)** CF

Fig. 9 Constructed volumetric models of each target structure (*top*: Scenario 4, *middle*: Scenario 5, and *bottom*: Scenario 6) with trajectories taken by the micro-aerial vehicle at the end of executions of proposed **a** INSP-COV and three alternative methods, **b** RH-NBV (Bircher et al. 2018), **c** VOL (Vasquez-Gomez et al. 2014), and **d** CF (Yamauchi 1997). Note that INSP-COV produces the shortest paths with the minimum path overlap while constructing the complete volumetric models

maximum sensing range (8 m or 10 m) than for the previous scenarios. The maximum velocity of the MAV was set to $v_{max} = 0.4$ m/s. We assumed that the bounded space containing an infrastructure was known and the remaining workspace of the robot was empty. The MAV explored the bounded space while simultaneously constructing a volumetric map of the structure until a certain percentage θ_{exp} of the map is explored. We performed experiments ten times to compare the results with different methods.

We present the results in Figs. 8 and 9. The plots in Fig. 8 show the unknown volume and surface coverage in the map over time. Figure 9 shows the constructed volumetric models and the MAV's trajectories after each execution of best case. The graphs of the unknown volume represent the time taken by each algorithm to explore an unknown area. The surface coverage is the percentage of the observed surface cells as compared to the total number of surface cells of the original model. It represents the completeness of the constructed volumetric model.

The performance of CF is the poorest in all scenarios. RH-NBV showed better performances than in the 2D exploration scenarios. The frequency of the back-and-forth reciprocating motions of RH-NBV is reduced, because the RRT samples are more uniformly distributed in the open 3D spaces. VOL can move at maximum speed in a large



(a) **(b)**

Fig. 10 Comparative results of different sampling algorithms for inspection path planning. **a** Normalized path distance and **b** averaged execution time over the number of samples for three different algorithms. Our approach, online weighted set cover, produces the shortest inspection paths even with a small sampling

free space, and therefore outperforms RH-NBV in terms of exploration in Scenario 6. In the graph of the unknown volume, the most important observation is the reduction rate of the unknown volume. This shows the speed at which unknown volumes are explored and this indicates the efficiency of the exploration. The unknown volume reduction rates of RH-NBV and VOL are similar to those of our methods until the half of the total unknown volume has been explored, but thereafter they are sharply reduced. The unknown volume curves of RH-NBV and VOL have long, flat tails. The reason is that the newly covered volumes are

Table 3 Average and maximum computation time (s) of the four sub-modules and total computation in three scenarios

Key submodules	Scenario 2		Scenario 4		Scenario 6	
	Avg.	Max.	Avg.	Max.	Avg.	Max.
Sector decomposition	0.04	0.07	0.23	0.29	0.81	0.94
Sector coverage planning	0.04	0.11	0.12	0.66	0.05	0.16
Global path planning	0.03	0.09	0.20	0.27	0.46	0.55
Local inspection planning	0.02	0.04	0.06	0.32	0.09	0.24
Total computation	0.09	0.24	0.43	1.20	1.14	1.70

relatively small when the MAV is exploring or overpassing an already explored area. As can be seen in Fig. 9, the paths of RH-NBV and VOL evenly cover the entire area of the structure, but they sometimes overlap. However, our methods fully cover the local regions using the inspection method, and therefore already covered regions are rarely revisited. Therefore, the reduction rates of the unknown volume in our approaches do not decrease significantly. The trajectories of the paths of INSP-COV are shorter than those of other methods and simultaneously cover the entire area of the structures.

The table in Fig. 8 summarizes the performances when each algorithm has completed 95% of the exploration. The results show that INSP-COV in general outperforms the other approaches. As compared with CF, INSP-COV reduces the average exploration time by 44.60% and the average path length by 46.23%. Furthermore, INSP-COV performs better as the scene size increases. As compared to RH-NBV, INSP-COV reduces the exploration time by 18.39% and the path length by 17.99% in Scenario 4. However, in Scenarios 5 and 6, INSP-COV reduces the average exploration time by 37.93% and the average path length by 33.82% as compared to RH-NBV. When the MAV needs to revisit an already explored area, the larger the scene size, the longer the distance traveled for the return to revisit. INSP-COV reduces the number of revisits by considering a global visitation sequence and by completely scanning a local area. Thus, INSP-COV can get a more improved performance in an increased size scene.

The performances of INSP-COV and INSP are similar in Scenario 4, but different in the remaining scenarios. As compared to INSP, INSP-COV reduces the average exploration time by 16.50% and the average path length by 14.50% in Scenarios 5 and 6. This suggests that the global coverage planning is ineffective for modeling a simple cylindrical structure, but is effective for modeling complex structures. In particular, our methods show the highest average surface coverage, 94.01%, indicating that our approach is very suitable for autonomous modeling systems.

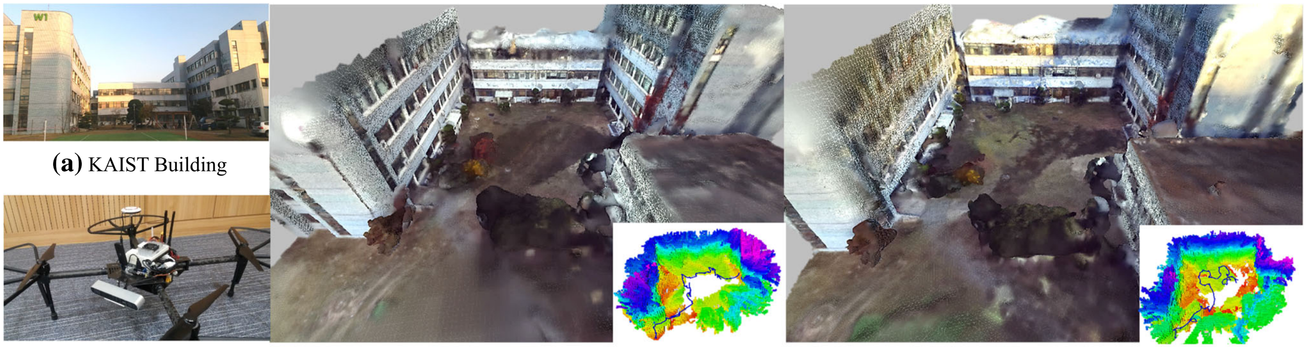
7.3 Effect of informative sampling

In order to verify the advantage of informative sampling (Sect. 6.1) over the existing offline method (Englot and Hover 2012), we conducted a comparative experiment. The offline method consists of a batch algorithm that solves the set cover problem after extracting all samples. The proposed method that solves online weighted set cover was compared with two offline methods. The first solves the original set cover problem that finds a small number of coverage configurations. Similarly to our method, the second solves the weighted set cover problem that finds a set of coverage samples with the minimum distances to the shortest path.

In Fig. 10a, we plot the average length of the inspection path generated by each method without path smoothing. Each path length is normalized by dividing by the shortest path length such that the value represents the increase in its length as compared to that of the shortest path. The offline set cover method provides the longest path lengths, which means that the configurations close to the shortest path in general generate a shorter inspection path in a local region. The inspection path of the weighted set cover method becomes shorter as the sampling number increases. However, our method generates a short inspection path even with a small sampling, and always shows a higher performance than the weighted set cover method. This indicates that the method of reducing the sampling range in the online set cover can efficiently improve the sampling performance and the inspection path quality.

7.4 Computation time analysis

In this sub-section, we analyze the computational performance of our method. Table 3 lists the average and maximum computational time for the key components of our method for three different scenarios. The total computation time in Table 3 is the average and maximum of the total time taken in each planning step. The planning step includes all cases where global and local planning are performed together or inspection path re-planning is performed alone. All components were processed on a standard desktop PC with an Intel Core i7 CPU without a graphics processing unit.



(a) KAIST Building

(b) MAV Platform

(c) INSP-Offline (completion time: 4.64 min, path length: 115.10 m)

(d) INSP-COV (completion time: 5.07 min, path length: 126.09 m)

Fig. 11 Results of real-world experiments. **a** Experimental environment of part of the campus building ($40 \times 50 \times 25\,m^3$). **b** Micro-aerial vehicle (MAV) platform with stereo camera employed in the experiment. Reconstructed 3D models and volumetric models with trajectories taken by the MAV at the end of executions of **c** INSP-Offline (Englot and Hover 2012) and **d** INSP-COV. Notice that INSP-COV performs as good as INSP-Offline in completion time and path length, considering that INSP-COV planned complex paths for the free space exploration. The 3D models were reconstructed from each set of recorded images by using the multi-view stereo program, COLMAP (Schönberger et al. 2016)

In Scenario 2 and 4, the results show that our method can be operated online. Each element does not significantly delay the entire system, especially in the worst case. In Scenario 6, most components have the longest computation time, because the map size and the maximum sensing range are large. The major part of the computation time is spent on the sector decomposition and global path planning step. In particular, sector decomposition takes an average time of $0.81 sec$, which is not much time as compared to the performance gains from global coverage planning.

8 Real-world experiments

We conducted a real-world experiment to demonstrate the feasibility of proposed approach in real-world applications. The MAV platform used in the experiment is shown in Fig. 11b. A DJI Matrice-100 drone was used and a ZED stereo camera and Jetson TX2 board were mounted on the drone. We employed a ZED SDK[2] for depth sensing and ORB-SLAM (Mur-Artal and Tardós 2017) for pose estimation. The Jetson TX board computed the depth map and camera pose and transmitted them to a laptop at the ground station. The laptop constructed the volumetric map and planned the exploration path. In reality, the MAV cannot follow a planned path exactly because of the internal sensor noise and wind gusts. For safety reasons, we manually controlled the MAV to follow the path instead of using autonomous navigation. To achieve precise manual control in a constant velocity, we set v_{max} and $\dot{\psi}_{max}$ to the minimum speed values, 0.45 m/s and 0.4 rad/s, respectively. Table 4 summarizes the parameters used in this experiment.

Table 4 Parameters used in real-world experiments

Parameter	Value	Parameter	Value
Resolution of $\mathcal{M}$	0.5 m	d_{sample}	2 m
Max sensing range	15 m	θ_{exp}	90%
v_{max}	0.45 m/s	θ_{front}	200
$\dot{\psi}_{max}$	0.4 rad/s	$\theta^{sector}_{adj-res}$	2 m
RRT edge length	2 m	θ^{insp}_{cover}	90%

A 3D model of a part of the campus building was constructed by exploring the environment depicted in Fig. 11a. The performance of the proposed method (INSP-COV) was compared with that of the offline inspection method (INSP-Offline) (Englot and Hover 2012), which precomputes a scanning path for 3D modeling using prior geometric information. The offline method can provide an optimal solution for given geometric information of a target; therefore, its performance can be an appropriate baseline for feasibility demonstration. If the performance of INSP-COV is as good as INSP-Offline, the proposed method can be considered as effective and feasible in a real-world experiment. To acquire the prior information for INSP-Offline, we manually collected the images of the target and constructed the 3D model by using the multi-view stereo program, COLMAP (Schönberger et al. 2016). Localization uncertainty is a major issue in INSP-Offline, because localization errors are incrementally accumulated; however, the MAV must follow the planned path without online path refinement (Galceran et al. 2015). To obtain globally consistent localization of INSP-Offline, we generated the feature map of the entire

[2] https://www.stereolabs.com/.

environment in advance and estimated the camera pose from the feature map in the localization mode of ORB-SLAM. The shape, scale, and position of the constructed prior model were refined to match the feature map. To compare the online exploration performances with the performances of INSP-Offline, INSP-COV also used the same feature map for localization. This allowed an exact performance evaluation of the planned paths of INSP-COV and INSP-Offline, excluding the localization inconsistency.

Figure 11c, d show the experimental results. In the two experiments, the MAV moved in a globally similar direction. INSP-Offline focuses only on the scanning of the target surfaces, while INSP-COV additionally considers the paths for securing free spaces. INSP-Offline showed better exploration performances than INSP-COV in terms of completion time and path length. However, considering that INSP-COV planned complex local paths for the free space exploration, the performance gap is not significant. Although an accurate 3D model could not be constructed because of the error in localization and depth estimation, the volumetric models maintained the structural shape of the building. Figure 11c, d show the 3D models reconstructed from each set of recorded images by using COLMAP. As can be seen in the figures, the two models are similar and of high quality, which indicates that the proposed online inspection method could cover all the target surfaces, similarly to the offline method. This result demonstrates the practical feasibility of the proposed online inspection method in real-world environments.

9 Limitations and discussion

The proposed approach has four major limitations. First, we restricted the maximum limit of the motion speed to be small to achieve stable sensing accuracy and exact path followings. All the experiments were conducted under this restriction. In the real-world experiments, the localization error was not sufficiently large to affect the exploration performance. However, if an MAV is in fast motion or a challenging environment (e.g., a dynamic or textureless environment), the uncertainty of the motion and state of the MAV should be taken into account for path planning. The application of active SLAM approaches (Chaves et al. 2016; Costante et al. 2018) to our method would be a good direction for future work.

Second, although the proposed sector decomposition method enforces the convexity of each cluster shape, this method, unlike of the greedy cut (Das et al. 2014) or trapezoidal decomposition (Oksanen and Visala 2009), does not perform exact convex partitioning. The objective of our decomposition method is to divide the entire map into sub-regions that can be sufficiently covered by the local inspection planning algorithm. To successfully perform this, two major factors need to be considered: 1) the decomposed sectors should be as convex as possible, and 2) the decomposed sectors should be evenly distributed (which is similar to the objective of distributed environment partitioning problem for coverage control (Durham et al. 2011)). In order to satisfy both factors at the same time, the convexity is not always guaranteed while conventional 2D coverage algorithms (Choset et al. 2000; Oksanen and Visala 2009) require to partition sectors as completely convex. Therefore, the proposed method may not be as effective in the conventional 2D coverage problem.

Third, the inspection path computed by our algorithm frequently changed in terms of the yaw angle to fully cover a local region. This made it difficult for the MAV to maintain its maximum velocity, although it could improve the exploration and modeling performances. VOL, similarly to the method in Cieslewski et al. (2017), maintains the maximum velocity most frequently. However, as verified in Sects. 7.1 and 7.2, the proposed method showed a higher exploration performance than VOL in every scenario. This suggests that inspection with a frequent yaw change is effective in exploration and modeling.

Finally, we used the sampling-based approach (Englot and Hover 2012), which separately solves the coverage sampling and multi-goal planning problems. Although the separation approach may provide a feasible coverage solution, it does not guarantee an optimal solution because of the decoupled two-step optimization. Even if we obtain an optimal set cover solution and find the shortest path solution, the computed inspection path may not be the shortest path. To alleviate this limitation, some approaches (Papadopoulos et al. 2013; Kafka et al. 2016) were aimed at solving the sampling and multi-goal planning problems simultaneously. However, these approaches are computationally expensive, and therefore are not suitable for online operation. We have no choice but to employ the separation approach for real-time computation and should accept its limitations.

10 Conclusion

In this paper, we proposed online coverage and inspection methods. In our study, we applied them to exploration tasks for constructing 3D models of unknown environments. The online coverage method computes the global coverage of unexplored sectors, which guides an MAV to complete the model sequentially. The online inspection method plans a local inspection path that provides comprehensive scanning of local frontiers. The local inspection path is continually refined according to the updated local model. Our simulation results show that the proposed method performs better than other state-of-the-art methods and in particular improves the completeness of the constructed 3D models. In real-world experiments, without prior structural information the proposed method showed a performance similar to that of the offline inspection method (Englot and Hover 2012). To the best of our knowledge, this is the first study in which an online coverage and inspection approach was implemented to explore and model 3D environments using an MAV.

Acknowledgements This work was supported by Basic Science Research Program through the National Research Foundation of Korea funded by the Ministry of Education (Grant No. NRF-2016R1D1A1B01013573).

References

Atkar, P. N., Choset, H., Rizzi, A. A., & Acar, E. U. (2001). Exact cellular decomposition of closed orientable surfaces embedded in/spl rfr//sup 3. In *IEEE international conference on robotics and automation (ICRA)*, *1* (pp. 699–704).

Atkar, P. N., Greenfield, A., Conner, D. C., Choset, H., & Rizzi, A. A. (2005). Uniform coverage of automotive surface patches. *International Journal of Robotics Research*, *24*(11), 883–898.

Bircher, A., Kamel, M., Alexis, K., Burri, M., Oettershagen, P., Omari, S., et al. (2016). Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Autonomous Robots*, *40*(6), 1059–1078.

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., & Siegwart, R. (2018). Receding horizon path planning for 3d exploration and surface inspection. *Autonomous Robots*, *42*(2), 291–306.

Blaer, P. S., & Allen, P. K. (2009). View planning and automated data acquisition for three-dimensional modeling of complex sites. *Journal of Field Robotics*, *26*(11–12), 865–891.

Blochliger, F., Fehr, M., Dymczyk, M., Schneider, T., & Siegwart, R. (2018). Topomap: Topological mapping and navigation based on visual slam maps. In *IEEE international conference on robotics and automation (ICRA)* (pp. 1–9).

Brown, S., & Waslander, S. L. (2016). The constriction decomposition method for coverage path planning. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3233–3238).

Charrow, B., Kahn, G., Patil, S., Liu, S., Goldberg, K., Abbeel, P., Michael, N., & Kumar, V. (2015). Information-theoretic planning with trajectory optimization for dense 3d mapping. In *Robotics: Science and systems (RSS)*, vol. 6.

Chaves, S. M., Kim, A., Galceran, E., & Eustice, R. M. (2016). Opportunistic sampling-based active visual slam for underwater inspection. *Autonomous Robots*, *40*(7), 1245–1265.

Cheng, P., Keller, J., & Kumar, V. (2008). Time-optimal uav trajectory planning for 3d urban structure coverage. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 2750–2757).

Choset, H. (2000). Coverage of known spaces: The boustrophedon cellular decomposition. *Autonomous Robots*, *9*(3), 247–253.

Choset, H., Acar, E., Rizzi, A. A., & Luntz, J. (2000). Exact cellular decompositions in terms of critical points of morse functions. In *IEEE international conference on robotics and automation (ICRA)*, *3* (pp. 2270–2277).

Cieslewski, T., Kaufmann, E., & Scaramuzza, D. (2017). Rapid exploration with multi-rotors: A frontier selection method for high speed flight. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 2135–2142).

Connolly, C. (1985). The determination of next best views. In *IEEE international conference on robotics and automation (ICRA)*, *2* (pp. 432–435).

Costante, G., Delmerico, J., Werlberger, M., Valigi, P., & Scaramuzza, D. (2018). Exploiting photometric information for planning under uncertainty. In *Robotics research* (pp. 107–124). Springer.

Das, A., Diu, M., Mathew, N., Scharfenberger, C., Servos, J., Wong, A., et al. (2014). Mapping, planning, and sample detection strategies for autonomous exploration. *Journal of Field Robotics*, *31*(1), 75–106.

Durham, J. W., Carli, R., Frasca, P., & Bullo, F. (2011). Discrete partitioning and coverage control for gossiping robots. *IEEE Transactions on Robotics*, *28*(2), 364–378.

Emek, Y., & Rosén, A. (2016). Semi-streaming set cover. *ACM Transactions on Algorithms*, *13*(1), 6.

Englot, B., & Hover, F. S. (2012). Sampling-based coverage path planning for inspection of complex structures. In *International conference on automated planning and scheduling (ICAPS)*.

Englot, B., & Hover, F. S. (2013). Three-dimensional coverage planning for an underwater inspection robot. *International Journal of Robotics Research*, *32*(9–10), 1048–1073.

Feigenbaum, J., Kannan, S., McGregor, A., Suri, S., & Zhang, J. (2005). On graph problems in a semi-streaming model. *Theoretical Computer Science*, *348*(2–3), 207–216.

Furrer, F., Burri, M., Achtelik, M., & Siegwart, R. (2016). Rotors—A modular gazebo mav simulator framework. In *Robot operating system (ROS)* (pp. 595–625). Springer.

Gabriely, Y., & Rimon, E. (2002). Spiral-STC: An on-line coverage algorithm of grid environments by a mobile robot. In *IEEE international conference on robotics and automation (ICRA)*, *1* (pp. 954–960).

Galceran, E., Campos, R., Palomeras, N., Ribas, D., Carreras, M., & Ridao, P. (2015). Coverage path planning with real-time replanning and surface reconstruction for inspection of three-dimensional underwater structures using autonomous underwater vehicles. *Journal of Field Robotics*, *32*(7), 952–983.

Helsgaun, K. (2000). An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research*, *126*(1), 106–130.

Heng, L., Gotovos, A., Krause, A., & Pollefeys, M. (2015). Efficient visual exploration and coverage with a micro aerial vehicle in unknown environments. In *IEEE international conference on robotics and automation (ICRA)*, *3* (pp. 3–5).

Hepp, B., Dey, D., Sinha, S. N., Kapoor, A., Joshi, N., & Hilliges, O. (2018). Learn-to-score: Efficient 3D scene exploration by predicting view utility. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 437–452).

Hess, J., Tipaldi, G. D., & Burgard, W. (2012). Null space optimization for effective coverage of 3d surfaces using redundant manipulators. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1923–1928).

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, *34*(3), 189–206.

Jadidi, M. G., Miro, J. V., & Dissanayake, G. (2015). Mutual information-based exploration on continuous occupancy maps. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 6086–6092).

Julian, B. J., Karaman, S., & Rus, D. (2014). On mutual information-based control of range sensing robots for mapping applications. *International Journal of Robotics Research*, *33*(10), 1375–1392.

Juliá, M., Gil, A., & Reinoso, O. (2012). A comparison of path planning strategies for autonomous exploration and mapping of unknown environments. *Autonomous Robots*, *33*(4), 427–444.

Kafka, P., Faigl, J., & Váňa, P. (2016). Random inspection tree algorithm in visual inspection with a realistic sensing model and differential constraints. In *IEEE international conference on robotics and automation (ICRA)* (pp. 2782–2787).

Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, *30*(7), 846–894.

Kuffner, J. J., & LaValle, S. M. (2000). RRT-connect: An efficient approach to single-query path planning. In *IEEE international conference on robotics and automation (ICRA)*, *2* (pp. 995–1001).

Li, X., Yu, W., Lin, X., & Iyengar, S. (2012). On optimizing autonomous pipeline inspection. *IEEE Transactions on Robotics*, *28*(1), 223–233.

Liu, M., Colas, F., Oth, L., & Siegwart, R. (2015). Incremental topological segmentation for semi-structured environments using discretized GVG. *Autonomous Robots*, *38*(2), 143–160.

Moorehead, S. J. (2001). *Autonomous surface exploration for mobile robots*. Pittsburgh: Carnegie Mellon University.

Mur-Artal, R., & Tardós, J. D. (2017). ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras. *IEEE Transactions on Robotics*, *33*(5), 1255–1262.

Oksanen, T., & Visala, A. (2009). Coverage path planning algorithms for agricultural field machines. *Journal of Field Robotics*, *26*(8), 651–668.

Oßwald, S., Bennewitz, M., Burgard, W., & Stachniss, C. (2016). Speeding-up robot exploration by exploiting background information. *IEEE Robotics and Automation Letters*, *1*(2), 716–723.

Papadopoulos, G., Kurniawati, H., & Patrikalakis, N. M. (2013). Asymptotically optimal inspection planning using systems with differential constraints. In *IEEE international conference on robotics and automation (ICRA)*.

Papon, J., Abramov, A., Schoeler, M., & Worgotter, F. (2013). Voxel cloud connectivity segmentation-supervoxels for point clouds. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 2027–2034).

Ramanagopal, M. S., Phu-Van Nguyen, A., & Le Ny, J. (2018). A motion planning strategy for the active vision-based mapping of ground-level structures. *IEEE Transactions on Automation Science and Engineering*, *15*(1), 356–368.

Roberts, M., Dey, D., Truong, A., Sinha, S., Shah, S., Kapoor, A., Hanrahan, P., & Joshi, N. (2017). Submodular trajectory optimization for aerial 3d scanning. In *International conference on computer vision (ICCV)*.

Schönberger, J. L., Zheng, E., Frahm, J. M., & Pollefeys, M. (2016). Pixelwise view selection for unstructured multi-view stereo. In *European conference on computer vision (ECCV)* (pp. 501–518). Springer.

Shade, R., & Newman, P. (2011). Choosing where to go: Complete 3d exploration with stereo. In: IEEE international conference on robotics and automation (ICRA), pp 2806–2811.

Shen, S., Michael, N., & Kumar, V. (2012). Stochastic differential equation-based exploration algorithm for autonomous indoor 3d exploration with a micro-aerial vehicle. *International Journal of Robotics Research*, *31*(12), 1431–1444.

Shnaps, I., & Rimon, E. (2014). Online coverage by a tethered autonomous mobile robot in planar unknown environments. *IEEE Transactions on Robotics*, *30*(4), 966–974.

Song, S., & Jo, S. (2017). Online inspection path planning for autonomous 3d modeling using a micro-aerial vehicle. In *IEEE international conference on robotics and automation (ICRA)* (pp. 6217–6224).

Song, S., & Jo, S. (2018). Surface-based exploration for autonomous 3D modeling. In: IEEE international conference on robotics and automation (ICRA) (pp. 1–8).

Song, S., Kim, D., & Jo, S. (2020). Active 3D modeling via online multi-view stereo. In *IEEE international conference on robotics and automation (ICRA)*.

Vasquez-Gomez, J. I., Sucar, L. E., & Murrieta-Cid, R. (2014). View planning for 3d object reconstruction with a mobile manipulator robot. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4227–4233).

Vasquez-Gomez, J. I., Sucar, L. E., & Murrieta-Cid, R. (2017). View/state planning for three-dimensional object reconstruction under uncertainty. *Autonomous Robots*, *41*(1), 89–109.

Vasquez-Gomez, J. I., Sucar, L. E., Murrieta-Cid, R., & Herrera-Lozada, J. C. (2018). Tree-based search of the next best view/state for three-dimensional object reconstruction. *International Journal of Advanced Robotic Systems*, *15*(1), 1729881418754575.

Vidal, E., Hernández, J. D., Istenic, K., & Carreras, M. (2017). Online view planning for inspecting unexplored underwater structures. *IEEE Robotics and Automation Letters*, *2*(3), 1436–1443.

Wang, Y., James, S., Stathopoulou, E. K., Beltrán-González, C., Konishi, Y., & Del Bue, A. (2019). Autonomous 3-d reconstruction, mapping, and exploration of indoor environments with a robotic arm. *IEEE Robotics and Automation Letters*, *4*(4), 3340–3347.

Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *IEEE international symposium on computational intelligence in robotics and automation* (pp. 146–151).

Yang, S. X., & Luo, C. (2004). A neural network approach to complete coverage path planning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, *34*(1), 718–724.

Zelinsky, A., Jarvis, R. A., Byrne, J., & Yuta, S. (1993). Planning paths of complete coverage of an unstructured environment by a mobile robot. In *International conference on advanced robotics*, *13* (pp. 533–538).

Zhu, D., Li, T., Ho, D., Wang, C., & Meng, M. Q. H. (2018). Deep reinforcement learning supervised autonomous exploration in office environments. In *2018 IEEE international conference on robotics and automation (ICRA)* (pp. 7548–7555). IEEE.

Soohwan Song received the B.S. degree in Information and Communication Engineering from Dongguk University, Seoul, Korea, and the M.S. and Ph.D. degrees in Computer Science from KAIST, Daejeon, Korea, in 2013, 2015, and 2020, respectively. Since March in 2020, he has been with the Information & Electronics Research Institute, KAIST, as a postdoctoral researcher. His research interests include robotics, motion planning, and computer vision.

Daekyum Kim received the B.S. degree in Mechanical Engineering from University of California, Los Angeles (UCLA) in 2015. He is currently a Ph.D. candidate in School of Computing at KAIST. His research interests are in the areas of machine learning, computer vision, and robotics.

Sungho Jo received the B.S. degree in the School of Mechanical & Aerospace Engineering from the Seoul National University, Seoul, Korea, the S.M. degree in Mechanical Engineering, and the Ph.D. degree in Electrical Engineering and Computer Science from the Massachusetts Institute of Technology (MIT), Cambridge, MA, in 1999, 2001, and 2006 respectively. While pursuing the Ph.D., he was associated with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Laboratory for Information Decision and Systems (LIDS), and Harvard-MIT HST NeuroEngineering Collaborative. Before joining the faculty with KAIST, he worked as a postdoctoral researcher with MIT media laboratory. Since December in 2007, he has been with the School of Computing at KAIST, where he is currently Professor. His research interests include robotic intelligence, augmented intelligence, neurohybrid intelligence.