Scanning Bot: Efficient Scan Planning using Panoramic Cameras

Euijeong Lee*, Kyung Min Han*, and Young J. Kim

Abstract— **Panoramic RGB-D cameras enable high-quality 3D scene reconstruction but require manual viewpoint selection and physical camera transportation, making the process time-consuming and tedious—especially for novice users. Key challenges include ensuring sufficient feature overlap between camera views and planning collision-free paths. We propose a fully autonomous scan planner that generates efficient and collision-free tours with adequate viewpoint overlap to address these issues. Experiments in both synthetic and real-world environments show that our method achieves up to 99% scan coverage and is up to three times faster than state-of-the-art view planning approaches.**

I. Introduction

The increasing advancements in mobile robotics research have enhanced robots' ability to improve the efficiency and completeness of outcomes in various active trajectory planning tasks. As a result, modern mobile robots can navigate large-scale environments with minimal human intervention to complete their missions. These applications include autonomous mapping of unknown terrain [1], search and rescue missions [2], building information modeling (BIM) [3], plant phenotyping [4], and post-disaster analysis [5], among others. In particular, view planning is critical in efficiently achieving reliable results. Indeed, there has been a long history of extensive research regarding how to plan agents' view paths to effectively gather essential information to reconstruct maps or scenes of the environment they navigate.

Although many state-of-the-art view planning systems have been developed over the past few decades, it is difficult to consider view planning a solved problem due to the diversity of applications and problem conditions in the context of mobile robot navigation. Consequently, it remains an active research topic within the robotics community. For example, in applications such as space exploration or search-and-rescue missions, the quality of the reconstructed model may be less critical than the scanning efficiency. In contrast, for applications such as BIM and digital twins, precise and thorough modeling of the environment is a key component of the problem. In these scenarios, not only is efficient view planning essential, but ensuring sufficient feature overlap between sensor viewpoints is also critical to guarantee a complete reconstruction of the scanned environment.

This paper focuses on the latter challenge, addressing the problem of best-view planning for a panoramic RGB-D camera. Although this camera offers superior reconstruction quality compared to low-end RGB-D cameras, it also introduces constraints that must be satisfied to solve the underlying reconstruction problem, as outlined below.

1) The selected viewpoints should ensure complete scene coverage across the entire space.
2) The number of viewpoints must be minimized, and the planned trajectory must ensure collision-free navigation.
3) Two neighboring viewpoints in the viewpoint sequence must lie within a certain distance to ensure reliable feature matching and construct a high-quality 3D model.

The first and second constraints are standard, as similar rules are often applied in most view planning systems. However, the third constraint is unique to our problem of using a panoramic camera, such as Matterport Pro2, and introduces additional complexity. Unfortunately, this constraint hinders us from directly applying previous view planning systems to our problem, as most of these approaches prioritize minimizing trajectory length without accounting for the interactions among viewpoints.

Furthermore, in the absence of an automated method, a human expert must manually operate the panoramic RGB-D camera, typically selecting viewpoints based on a grid pattern or a similar approach. In addition, the camera must be physically transported to each location and fixed in place, making the process tedious and time-consuming. The challenge becomes even more daunting when large scene models need to be produced to scale up datasets [6] for deep learning-related research.

To address these challenges, this research aims to develop a fully autonomous indoor mobile planner, called a scanning bot, that is equipped with a panoramic RGB-D camera. The robot can explore an unknown environment to construct a 2D map using our previous work, Autoexplorer [7]. Once the map is completed, our novel view planner selects the optimal viewpoints and determines their visiting order. Finally, the scanning bot stitches image data from each viewpoint to reconstruct a 3D model of the observed scenes. To achieve these goals, we propose a novel view planner with the following contributions:

- Fully autonomous mobile scan planning system, *scanning bot*, for reconstructing 3D models of scenes using a panoramic RGB-D camera.
- A fast and efficient greedy coverage planner that guarantees comprehensive scanning coverage. Our approach minimizes the number of selected viewpoints while positioning them as closely as possible to ensure sufficient feature overlap among camera views.
- A new collision-free and visibility-aware path planner that maintains sufficient feature overlap between neigh-

*Equal contributions. The authors are with the Department of Computer Science and Engineering at Ewha Womans University in South Korea $\{euijeonglee|hankm|kimy\}@ewha.ac.kr$.

boring viewpoints, ensuring an optimal tour plan for a high-quality reconstruction.

- Extensive comparisons of our method against state-of-the-art view planning approaches. The results demonstrate that our method substantially outperforms existing approaches in terms of efficiency in most scenarios while maintaining over 99% coverage rate.

II. Related Work

A. Coverage Path Planning (CPP)

Coverage path planning (CPP) focuses on generating an optimal coverage trajectory that is both shortest and collision-free. Given its significance in real-world applications, research in this area remains highly active. It continues to evolve due to its wide range of modern applications, including home cleaning robots, farm-land robots, and mine detection tasks.

Boustrophedon cellular decomposition (BCD) [8] is one of the fundamental methods that has served as a foundation for many subsequent CPP studies. As such, one significant challenge in improving trajectory efficiency is minimizing the number of turns when the robot generates a zigzag pattern [9]. Additionally, differential-wheeled robots often waste considerable time finding paths in cluttered environments or navigating through tight spaces, such as narrow passages. To address these issues, [10] and [11] explored CPP with reconfigurable robots, enhancing coverage efficiency in such scenarios.

Moreover, map changes frequently occur during navigation due to human intervention or previously undetected static objects. To handle partially unknown maps, [12] proposed an efficient replanning method based on an integer programming formulation. Notably, CPP tasks can be completed more efficiently when multiple robots collaborate, as demonstrated in [13].

B. Informative Path Planning (IPP)

While the primary objective of CPP is to ensure complete coverage and trajectory efficiency, IPP focuses on maximizing information gain or minimizing uncertainty within the exploration area. Consequently, IPP solvers are particularly suited for tasks such as search and rescue missions [14], change detection [15], and large-scale surveillance and monitoring [16]. Gaussian process regression (GPR) has traditionally been the most widely used approach for IPP problems [17], owing to its probabilistic nature. Research has recently explored deep learning-based IPP solvers, including an attention-based neural network trained using a reinforcement learning (RL) framework [18].

C. Active SLAM

Note that an IPP solver can utilize a map configuration as prior knowledge. In such cases, the solver can generate a non-myopic path plan. However, if the exploration space is unknown, the planning must be adaptive, relying on previous observations. IPP solvers designed for unknown space mapping are also known as active SLAM [19]. In recent years, numerous new approaches have emerged in active SLAM, particularly driven by advancements in the DARPA Subterranean Challenge [20].

D. Autonomous Scene Reconstruction

Scene reconstruction using an RGB-D camera sensor has been a significant research topic in robotics. Among the proposed methods, [21] introduced an approach that reconstructs scenes by leveraging time-varying tensor fields. More recently, 3D model reconstruction has regained attention due to the success of 3D Gaussian Splatting (3DGS) [22]. One notable application of 3DGS is demonstrated by [23], who presented an efficient room-scale reconstruction using multiple indoor robots.

III. Problem Formulation

The main objective of our problem is determining the collision-free optimal view trajectory $\mathcal{T}^*$ that *covers* the entire scene. To achieve this, we decompose the problem into two phases: (1) optimal viewpoint $\mathcal{V}^*$ selection and (2) generating $\mathcal{T}^*$ from $\mathcal{V}^*$.

A. Optimal Viewpoint Selection

A 2D grid map $\mathcal{M}$ consists of two types of grid cell x classes: (i) occupied cells $\mathcal{O}$ corresponding to obstacles and (ii) unoccupied or free cells $\mathcal{F}$ corresponding to the collision-free space. Such a map can be given or autonomously constructed [7]. x_i is said to be *visible* to x_j, $x_i \mapsto x_j$ (or $x_j \mapsto x_i$) if

$$\mathbf{\Lambda} = tx_i + (1-t)x_j \in \mathcal{F},\ 0 \le \forall t \le 1 \text{ and } ||\mathbf{\Lambda}|| \le r, \quad \text{(1a)}$$

where r is a sensor-range constant. A cell $x_i \in \mathcal{F}$ *sees* a set of free cells $\mathcal{S}_i \in \mathcal{F}$, denoted as $x_i \mapsto \mathcal{S}_i$, when $\forall x \in \mathcal{S}_i, x_i \mapsto x$. Our first goal is to find a minimal set of "viewpoint" cells, $\mathcal{V}^* = \{x_i \in \mathcal{F}\}$ such that $\mathcal{F} \subset \bigcup_{x_i \mapsto \mathcal{S}_i} \mathcal{S}_i$. This optimization problem can be formulated as a set covering problem:

$$\min \quad \sum_{\forall i} \alpha_i \cdot \bar{x}_i, \quad \bar{x}_i \in \{0,1\}, \alpha_i \in \mathbb{R}^+ \quad \text{(2a)}$$

$$\text{s.t.} \quad \sum_{i} \bar{x}_i \ge 1, \quad 1 \le \forall i \le |\mathcal{F}| \quad \text{(2b)}$$

$$\bar{x}_i - \sum_{j \ne i} \bar{x}_j \le 0 \quad \text{for } \rho(x_i, x_j) \le r,\ \forall i, \quad \text{(2c)}$$

where $\bar{x}_i$ corresponds to a viewpoint $x_i \in \mathcal{F}$, α_i refers to the significance of x_i, and ρ is a distance function between two viewpoints, $x_i \mapsto x_j$. Eq. 2a and 2b form a typical set covering problem, but Eq. 2c complicates the combinatorial optimization problem with geometric constraints. Finally, $\mathcal{V}^* = \{\forall x_i | \bar{x}_i \ne 0\}$ is given from the solution of Eq. 2a.

B. Optimal Scan Ordering

This subsection formulates our second goal: to generate an optimal scan tour plan from $\mathcal{V}^*$. Formally, our tour optimization problem is:

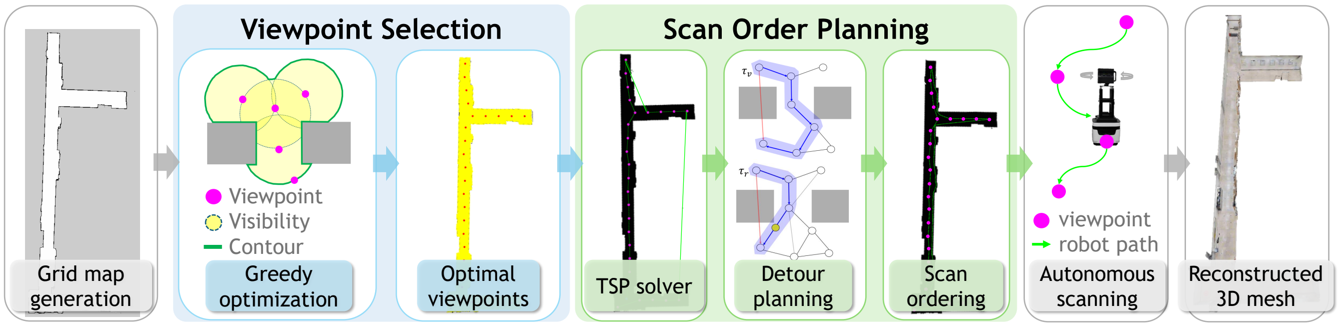


Fig. 1: **Scanning Bot Pipeline.** The set-covering procedure first identifies the optimal viewpoints. This is followed by TSP-based, collision-free view planning to autonomously scan a scene.

$$\min \quad \sum_{i} \sum_{j \neq i} \beta_{ij} \bar{e}_{ij} \tag{3a}$$

$$\text{s.t.} \quad \sum_{j \neq i}^{n} \bar{e}_{ij} = 1 \text{ and } \sum_{i \neq j}^{n} \bar{e}_{ij} = 1 \tag{3b}$$

$$\forall i, \exists j < i, \ \rho(x_i, x_j) \leq r \tag{3c}$$

Here, $\bar{e}_{ij} \in \{0, 1\}$ represents a tour sequence from $x_i \in \mathcal{V}^*$ to $x_j \in \mathcal{V}^*$, and β_{ij} represents the cost for choosing e_{ij}. For instance, $\beta_{ij} = \infty$ if x_i is not visible from x_j. We call such an edge e_{ij} with $\beta_{ij} = \infty$ *infeasible*. Eq. 3a and 3b constitute a typical TSP problem, respectively, but our problem requires the additional precedence constraint (Eq. 3c) where each viewpoint in the final tour plan must have a visible viewpoint among its predecessors. A solution to Eq. 3a is the final tour $\mathcal{T}^* = \{\bar{e}_{12}, \cdots, \bar{e}_{n-1,n}\}, \forall \bar{e}_{i,i+1} \neq 0$.

IV. Autonomous View Planning

Fig. 1 shows the full pipeline of our method, consisting of viewpoint selection and scan order planning to be described in the following Section IV-A and IV-B, respectively.

A. Viewpoint Selection

Due to the additional complexity introduced by Eq. 2c, the standard set-cover formulation is unsuitable for our problem. Furthermore, the size of $\mathcal{F}$ in our problem also hinders efficient optimization. To address these challenges, we propose a greedy set covering-based approach to compute $\mathcal{V}^*$ as explained below (also illustrated in Fig. 2):

1) Initially, using [24], we compute a universe of free cell sets $\mathcal{U} = \{\mathcal{S}_i | \forall x_i \in \mathcal{F}, x_i \mapsto \mathcal{S}_i\}$. Pick x_0 where $\mathcal{S}_0$ is the largest in $\mathcal{U}$, and set $\mathcal{V}^* = \{x_0\}$.
2) Identify the contour cells $\partial\mathcal{S}_i$ of $\mathcal{S}_i$ utilizing DFFP [25].
3) For each x in $\partial\mathcal{S}_i$, evaluate $\phi(\cdot)$, which depends on the size of the set and the distance to the nearby obstacles:

$$\phi(x) = |\mathcal{S}_i| / \max_i |\mathcal{S}_i| - e^{-\|x - \mathbf{o}\|_2}, \tag{4}$$

where $\mathbf{o}$ represents the position of the obstacle cell closest to x. Choose $x_j = \text{argmax}_x \, \phi(x)$ as the next best viewpoint x_j and add x_j to $\mathcal{V}^*$.
4) We repeat 2)∼4) until $\mathcal{F} \subset \{\mathcal{S}_i | \forall x_i \in \mathcal{V}^*\}$.

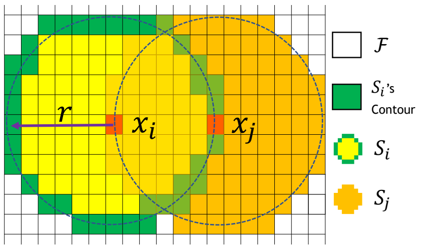
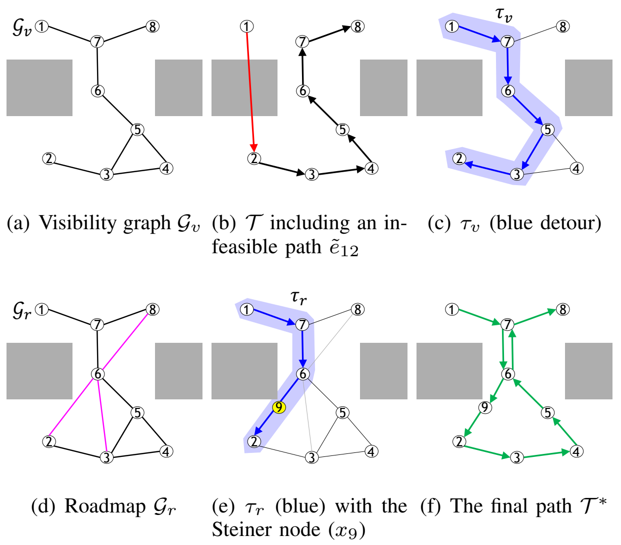


Fig. 2: **Greedy set-covering.** x_i covers the map with $\mathcal{S}_i$. The green cells represent $\mathcal{S}_i$'s contour cells. The next viewpoint x_j is selected from these contour cells based on $\phi(\cdot)$.

B. Scan Order Planning



(a) Visibility graph $\mathcal{G}_v$ (b) $\mathcal{T}$ including an infeasible path $\tilde{e}_{12}$ (c) τ_v (blue detour)

(d) Roadmap $\mathcal{G}_r$ (e) τ_r (blue) with the Steiner node (x_9) (f) The final path $\mathcal{T}^*$

Fig. 3: **Detour plan using the roadmap.** (a) $\mathcal{G}_v$ (b) a TSP plan $\mathcal{T}$ with an infeasible edge between x_1 and x_2. (c) presents the τ_v found in $\mathcal{G}_v$. (d) and (e) show the roadmap $\mathcal{G}_r$ and τ_r found on $\mathcal{G}_r$, respectively. τ_r has a new Steiner node x_9. (f) shows the final path $\mathcal{T}^*$.

We use a TSP solver [26] to initialize the scan planning $\mathcal{T}$ with $\mathcal{V}^*$ as input. However, $\mathcal{T}$ may violate the constraints in Eq. 3c, yielding infeasible paths. To address this issue, our approach searches for a detour plan τ to replace any infeasible edge $\tilde{e} \in \mathcal{T}$. For instance, Fig. 3(a) and 3(b) show an instance of $\mathcal{G}_v$ and the initial plan $\mathcal{T} = \{e_{12}, \cdots, e_{78}\}$ from the TSP solver starting from x_1. The edge $\tilde{e}_{12}$ in $\mathcal{T}$ is infeasible and, requires a detour.

One approach is to search for a detour path τ_v in the visibility graph $\mathcal{G}_v = (\mathcal{V}^*, \mathcal{E}_v)$, where $\mathcal{E}_v = \{e_{ij} | \beta_{ij} \neq \infty, \forall ij\}$. However, restricting the plan to $\mathcal{G}_v$ may lead to an inefficient path due to the constraint defined in Eq. 2c. For instance, as illustrated in Fig. 3(c), replacing $\tilde{e}_{12}$ with $\tau_v = \{x_1, x_7, \cdots, x_2\}$ is suboptimal.

Alternatively, we expand $\mathcal{G}_v$ to the roadmap graph $\mathcal{G}_r = (\mathcal{V}^*, \mathcal{E}_r)$ such that $\mathcal{G}_v \subset \mathcal{G}_r$. The edge set $\mathcal{E}_r$ satisfies $x_i \not\mapsto x_j$ for $\forall e_{ij} \in \mathcal{E}_r$, but the sensor-range constant r relaxes to explore more free space. $\mathcal{G}_r$ is efficiently computed using Delaunay triangulation [27] with edge cutting. Fig. 3(d) shows $\mathcal{G}_r$ with newly added edges (e_{68}, e_{26}, e_{36}). Since these may violate the distance constraint r, a Steiner node is inserted in the edges, for instance, the yellow node x_9 in Fig. 3(e), between x_2 and x_6.

Given two route choices τ_r and τ_v, the next challenge is to decide which one is more optimal. This is not simply choosing the shorter path, since the robot must physically visit nodes and spend time capturing images at viewpoints. To reflect this cost, we define the following cost function:

$$\psi(\tau) = (1 - \eta) \sum_{i}^{m-1} \rho(x_{i+1}, x_i) + \eta \cdot n, \tag{5}$$

where n represents the number of newly added viewpoints, while η is a penalty factor balancing the total path length and the number of additional viewpoints. We observed that a new viewpoint adds approximately 50 more seconds to our panoramic scanning. Lastly, we substitute all $\tilde{e}_{ij} \in \mathcal{T}$ with $\min(\psi(\tau_v), \psi(\tau_r))$ to obtain the final scan sequence $\mathcal{T}^*$, as shown in Fig. 3(f) as $\mathcal{T}^* = \{e_{17}, e_{76}, \cdots, e_{78}\}$.

After $\mathcal{T}^*$ is obtained, a standard motion planner plans the continuous motion. In our case, we use the A* algorithm for global planning between successive viewpoints x_i and x_{i+1}, and the TEB planner [28] for collision-free local planning.

V. Experiments

A. *Performance Analysis in Synthetic Environments*

This section compares the performance of our method with state-of-the-art approaches in synthetic environments. All experiments were conducted on the same machine with an AMD Ryzen 9 CPU and an NVIDIA RTX 3090 GPU, running Ubuntu 20.04. For the synthetic world experiments, we utilized Gazebo worlds from Explore-Bench [29], originally designed for the TurtleBot3 robot. However, for this experiment, we slightly modified the "room" world to accommodate a larger robot, as shown in Fig. 4(a). This change was necessary because one of the baseline approaches, 3DMR, required the use of the Jackal robot for the experiment. Additionally, we created a custom world, called "rooms", as shown in Fig. 4(b).

To ensure a diverse set of baselines, we selected four state-of-the-art methods from Section II. Specifically, we included two CPP approaches: BCD [8] and CLCPP [30], a global Kriging variance minimization (GKVM) based IPP solver [17], and one active SLAM method: 3DMR [31]. Notably, [31] is one of the latest implementations of the next-best-view (NBV) planning approach [32]. Since the NBV approach is an online adaptive planner, we first allowed the module to explore the test environment into 2*m* sensor range satisfying the overlapping constraint (Eq. 2c). The viewpoints used for comparison were selected based on those chosen during the online NBV process. For the other three baselines, we first utilized Autoexplorer [7] to construct the base map. Using this map, we applied each method to generate a view plan. The final set of viewpoints was obtained by segmenting these trajectories into 2*m* segments, satisfying Eq. 2c and 3c.

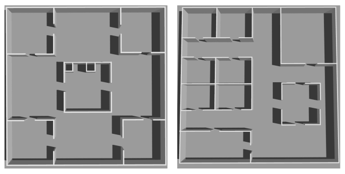

(a) Room (b) Rooms

Fig. 4: **Examples of synthetic worlds.** The modified "room" and the newly created "rooms" world.

Table I summarizes the total coverage achieved and the number of viewpoints for each method. Table II reports both exploration and planning times. Fig. 6 shows an example of the "rooms" environment where we qualitatively compare our method against baseline approaches. GKVM achieved the shortest path length and the lowest number of viewpoints; however, it only covered approximately 74% of the space. The two CPP methods attained full 100% coverage but required a significantly higher number of viewpoints, leading to inefficient path lengths. In contrast, our method achieved over 99% coverage with only 133 viewpoints, demonstrating a balance between efficiency and completeness.

According to our study, CPP approaches, such as the BCD and CLCPP method, exhibit the fastest planning time and the highest coverage among the five methods. However, BCD and CLCPP require nearly five times as many viewpoints as our method. On the other hand, GKVM achieves the lowest number of viewpoints among all approaches. However, its coverage is significantly lower than other methods, and in some cases, it fails to complete the input map reliably. Lastly, 3DMR finds a balance between coverage and the number of viewpoints. However, it is about five times slower than other methods regarding total planning time, including view planning time and space exploration time. This is primarily because the next-best-view type planners are better suited for unknown space exploration rather than for selecting viewpoints in a post-processing scenario.

Overall, our method generates the fewest viewpoints while achieving over 99% scanning coverage. This is because our method incorporates the visibility constraint in view plan optimization, whereas other approaches do not consider such a constraint. The two CPP-based methods achieved faster planning times than our approach. However, the planning time constitutes a relatively small portion of the total scanning time, which includes planning, navigation, and image capturing. This is because the number of viewpoints primarily influences the total scanning time. In Section V-B, we will further discuss the comparison of total scanning time between our approach and other baseline methods.

B. Real World Experiments

In this section, we conducted real-robot experiments to assess the total scanning time, encompassing the planning, the robot's motion, and the image capturing time. Since the total scanning time is closely associated with planning efficiency, the real-world experiment provides a comprehensive analysis of the proposed approach compared to baseline methods.

Fig. 5 introduces our scanning-bot system utilized for real-world experiments. The system consists of a view planning PC and a Matterport Pro2 panoramic camera mounted on top of a differential-wheeled mobile robot. This robot is equipped with basic sensors for 2D LiDAR SLAM. It is powered by an Intel i5 processor with 8GB of RAM, enabling on-board navigation processes such as SLAM and motion planning. The view planning PC has an Intel Core i9 CPU and an NVIDIA RTX 3080 GPU, running Ubuntu 18.04. The captured images are transmitted to Matterport's Cortex AI platform to reconstruct a textured 3D mesh by [6]. A crucial constraint for Matterport reconstruction is that each scanning point should be within the $2m$ range of neighboring points.

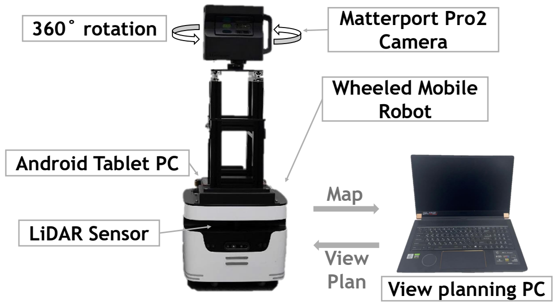


Fig. 5: **Robot platform for real-world experiments.**

Similar to Section V-A, we conducted several experiments to assess the baseline approaches except 3DMR owing to its strict dependency on a 3D LiDAR sensor. Consequently, we report the BCD, CLCPP, and GKVM results as the baselines against our method for these experiments. The experiments were carried out using the scanning-bot system presented in Fig. 5, including the ROS navigation SW packages [24][28][33].

The experiments were conducted in the ASAN Engineering Building at Ewha Womans University, which consists of a long corridor; see Fig. 7. Table III presents the quantitative results of these experiments. We observed that our method and CLCPP achieved over 99% coverage of the scanning space. However, our method was approximately three times faster than CLCPP in terms of total scanning time. The total scanning time encompasses visiting all planned viewpoints and capturing images at each point.

Fig. 7 highlights the superiority of our method in terms of coverage rate and the number of viewpoints generated. Our method produces fewer viewpoints while ensuring the selected viewpoints thoroughly cover the entire space.

In a real-world scenario, the robot must rely on estimated SLAM, which is prone to errors during long-term navigation. Additionally, our method selects viewpoints away from obstacles using our evaluation function (Eq. 4), whereas other approaches ignore such factors when generating trajectories. Note that visiting viewpoints near obstacles can lead to unintended collisions or unnecessary motion behaviors, increasing the likelihood of SLAM errors. Moreover, this negatively impacts the final 3D model quality, as shown in the second row of Fig. 7. Lastly, Fig. 8 shows three more instances of the 3D models obtained from our system.

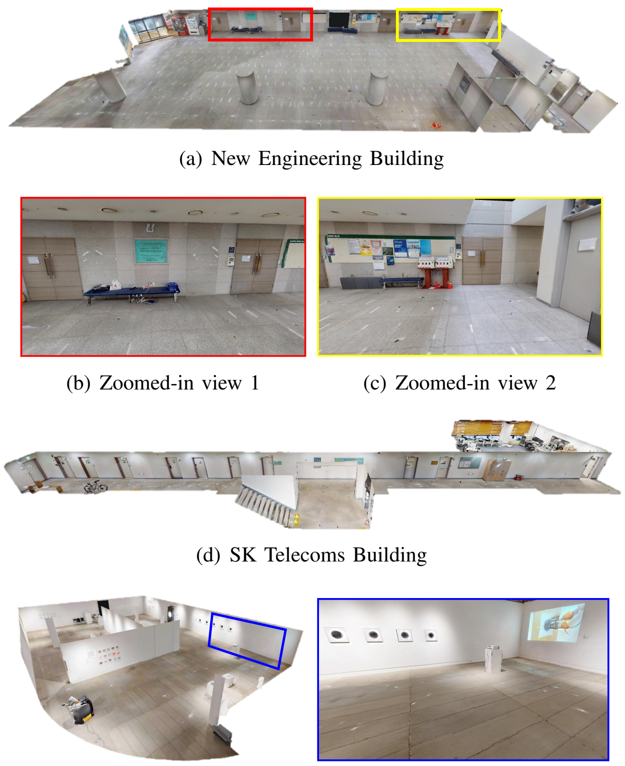


Fig. 8: **The 3D models reconstructed from various environments.** (a), (d), and (e) show the 3D models; (b) and (c) present zoom-in of (a); (f) is zoom-in of (e)

VI. Conclusion

We introduced a novel autonomous view planning system for panoramic RGB-D cameras. Our visibility-based set-covering algorithm selects the minimum set of viewpoints, which are used to generate efficient tour plans. The system achieved nearly 99% coverage while outperforming state-of-the-art approaches in view planning efficiency.

Methods	Coverage (%) / Number of viewpoints						
	corner	corridor	loop	loop_ with_ corridor	room	rooms	room_ with_ corner
BCD	99.8/259	**100**/304	**100**/219	**100**/404	**100**/594	**100**/525	**100**/342
CLCPP	**100**/296	**100**/256	**100**/191	**100**/277	**100**/480	**100**/516	99.9/386
GKVM	– / –	85.7/**48**	– / –	87.4/**60**	72.3/**72**	74.2/**94**	72.2/**52**
3DMR	99.3/86	99.3/75	99.9/61	99.3/93	99.4/182	99.1/170	99.4/117
Our method	99.0/**58**	99.4/70	99.6/**36**	99.9/70	99.6/127	99.5/133	98.9/89

TABLE I: **The quantitative comparisons of coverage and number of viewpoints for different methods in various simulation worlds.** "–" indicates that no feasible plan was generated within a finite time.

Methods	Exploration time (s) / Planning time (s)						
	corner	corridor	loop	loop_ with_ corridor	room	rooms	room_ with_ corner
BCD	356.12/**0.01**	207.44/**0.01**	258.98/**0.01**	661.62/**0.01**	360.58/**0.02**	314.39/**0.01**	347.47/**0.01**
CLCPP	356.12/**0.01**	207.44/**0.01**	258.98/**0.01**	661.62/**0.01**	360.58/0.11	314.39/0.04	347.47/0.07
GKVM	–	207.44/17.56	–	661.62/15.27	360.58/12.50	314.39/12.23	347.47/10.39
3DMR	2843.07	2246.83	1366.92	3572.30	9180.33	6939.96	5256.27
Our method	356.12/7.82	207.44/7.57	258.98/5.62	661.62/8.60	360.58/65.68	314.39/63.02	347.47/10.37

TABLE II: **Comparison of exploration and planning times with baseline methods in sim environments.** The exploration time is identical for BCD, CLCPP, GKVM, and our method, as they all use the map generated by AutoExplorer [7].

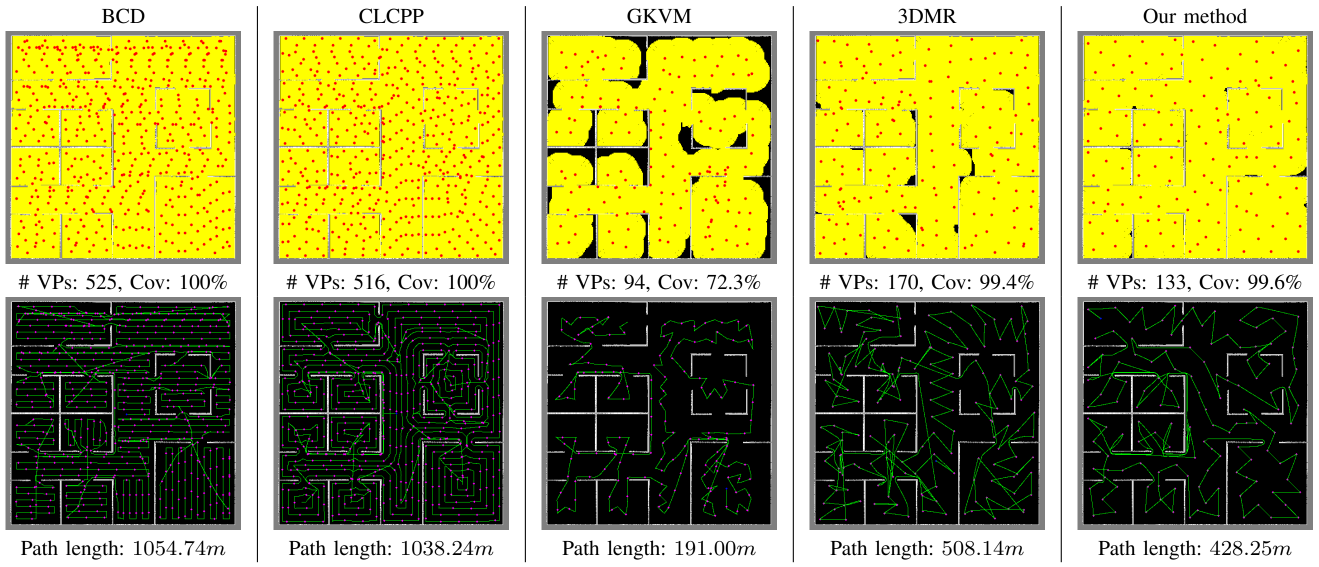


Fig. 6: **Qualitative comparisons of the sim results** Top row shows the selected viewpoints (red dots) and the covered area (yellow pixels). The bottom row shows the planned path (green lines).

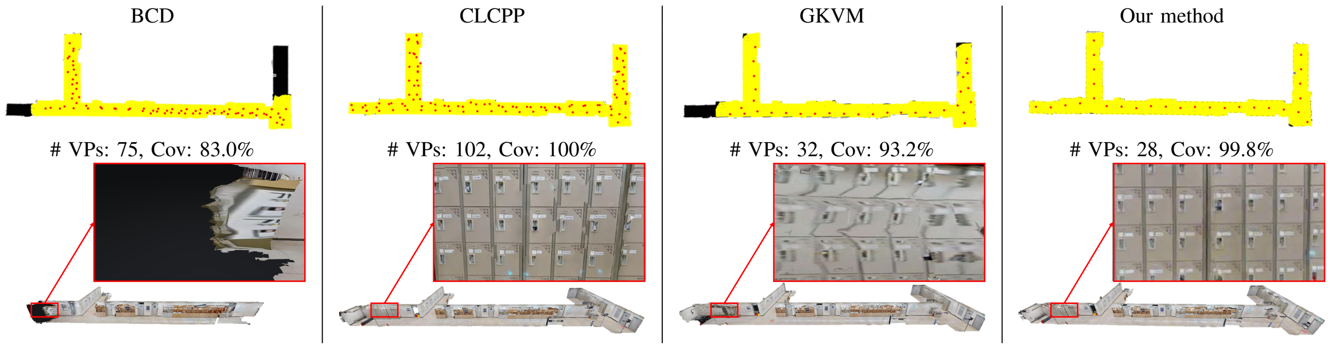


Fig. 7: **Qualitative comparisons of real-world experiments.** The top row shows viewpoints (red) and covered areas (yellow). The bottom row shows 3D models with detailed textures, highlighting our method's better-textured mesh quality.

Methods	BCD	CLCPP	GKVM	Our method
Path length (m)	151.56	205.73	**65.20**	112.08
Coverage (%)	82.98	**100**	93.21	99.81
Number of Viewpoints	75	102	32	**28**
Planning time (s)	**0.02**	**0.02**	65.24	12.85
Scanning time (m)	41	95	40	**33**

TABLE III: **Comparisons of real-world results.** Our method achieved over 99% coverage with the fewest viewpoints and shortest scanning time.

In the future, we plan to unify the exploration and scanning phases into a single process, enabling more efficient scanning in unknown environments. Moreover, we aim to extend our system to outdoor robots, developing a more generalized approach for scanning indoor and outdoor environments.

VII. Acknowledgments

This project was supported in part by ITRC/IITP Program (IITP-2025-RS-2020-II201460) and the NRF (NRF-2022R1A2B5B03001385 and RS-2024-00456464) in South Korea.

References

[1] C. Cao *et al.*, "Representation granularity enables time-efficient autonomous exploration in large, complex worlds," *Science Robotics*, vol. 8, no. 80, p. eadf0970, 2023.

[2] F. Niroui *et al.*, "Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 610–617, 2019.

[3] Z. Chen *et al.*, "Improved coverage path planning for indoor robots based on bim and robotic configurations," *Automation in Construction*, vol. 158, p. 105160, 2024.

[4] F. Esser *et al.*, "Field robot for high-throughput and high-resolution 3d plant phenotyping: Towards efficient and sustainable crop production," *IEEE Robotics & Automation Magazine*, vol. 30, no. 4, pp. 20–29, 2023.

[5] R. Nagasawa *et al.*, "Model-based analysis of multi-uav path planning for surveying postdisaster building damage," *Scientific Reports*, vol. 11, no. 1, Dec. 2021, publisher Copyright: © 2021, The Author(s).

[6] A. Chang *et al.*, "Matterport3d: Learning from rgb-d data in indoor environments," *International Conference on 3D Vision (3DV)*, 2017.

[7] K. M. Han and Y. J. Kim, "Autoexplorer: Autonomous exploration of unknown environments using fast frontier-region detection and parallel path planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022, pp. 10 536–10 541.

[8] H. Choset and P. Pignon, "Coverage path planning: The boustrophedon cellular decomposition," in *Field and Service Robotics*, A. Zelinsky, Ed. London: Springer London, 1998, pp. 203–209.

[9] I. Vandermeulen *et al.*, "Turn-minimizing multirobot coverage," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 1014–1020.

[10] M. A. V. J. Muthugala *et al.*, "Online coverage path planning scheme for a size-variable robot," in *IEEE International Conference on Robotics and Automation*, 2023, pp. 5688–5694.

[11] P. T. Kyaw *et al.*, "Energy-efficient path planning of reconfigurable robots in complex environments," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2481–2494, 2022.

[12] M. Ramesh *et al.*, "Anytime replanning of robot coverage paths for partially unknown environments," *IEEE Transactions on Robotics*, vol. 40, pp. 4190–4206, 2024.

[13] R. Mitra and I. Saha, "Online on-demand multi-robot coverage path planning," in *IEEE International Conference on Robotics and Automation*, 2024, pp. 14 583–14 589.

[14] Y. Luo *et al.*, "Star-searcher: A complete and efficient aerial system for autonomous target search in complex unknown environments," *IEEE Robotics and Automation Letters*, vol. 9, no. 5, pp. 4329–4336, 2024.

[15] A. Blanchard and T. Sapsis, "Informative path planning for anomaly detection in environment exploration and monitoring," *Ocean Engineering*, vol. 243, p. 110242, 2022.

[16] M. Popović *et al.*, "An informative path planning framework for uav-based terrain monitoring," *Autonomous Robots*, vol. 44, no. 6, pp. 889–911, 2020.

[17] C. Xiao and J. Wachs, "Nonmyopic informative path planning based on global kriging variance minimization," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1768–1775, 2022.

[18] Y. Cao *et al.*, "Catnipp: Context-aware attention-based network for informative path planning," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 1928–1937.

[19] J. A. Placed *et al.*, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 1686–1705, 2023.

[20] E. Ackerman, "Robots conquer the underground: What darpa's subterranean challenge means for the future of autonomous robots," *IEEE Spectrum*, vol. 59, no. 5, pp. 30–37, 2022.

[21] K. Xu *et al.*, "Autonomous reconstruction of unknown indoor scenes guided by time-varying tensor fields," *ACM Transactions on Graphics*, vol. 36, no. 6, pp. 1–15, 2017.

[22] B. Kerbl *et al.*, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023.

[23] J. Yu *et al.*, "Language-embedded gaussian splats (legs): Incrementally building room-scale representations with a mobile robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024, pp. 13 326–13 332.

[24] E. Marder-Eppstein *et al.*, "The office marathon: Robust navigation in an indoor office environment," in *International Conference on Robotics and Automation*, 2010.

[25] K. M. Han and Y. J. Kim, "Neuro-explorer: Efficient and scalable exploration planning via learned frontier regions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024, pp. 3240–3245.

[26] D. L. Applegate, *The traveling salesman problem: a computational study*. Princeton university press, 2006, vol. 17.

[27] D.-T. Lee and B. J. Schachter, "Two algorithms for constructing a delaunay triangulation," *International Journal of Computer & Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.

[28] C. Rösmann *et al.*, "Kinodynamic trajectory optimization and control for car-like robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 5681–5686.

[29] Y. Xu *et al.*, "Explore-bench: Data sets, metrics and evaluations for frontier-based and deep-reinforcement-learning-based autonomous exploration," in *IEEE International Conference on Robotics and Automation*, 2022, pp. 6225–6231.

[30] R. Bormann *et al.*, "Indoor coverage path planning: Survey, implementation, analysis," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 1718–1725.

[31] L. Freda *et al.*, "3d multi-robot exploration with a two-level coordination strategy and prioritization," *arXiv preprint arXiv:2307.02417*, 2023.

[32] A. Bircher *et al.*, "Receding horizon "next-best-view" planner for 3d exploration," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 1462–1468.

[33] S. Macenski and I. Jambrecic, "Slam toolbox: Slam for the dynamic world," *Journal of Open Source Software*, vol. 6, no. 61, p. 2783, 2021.