PIPE Planner: Pathwise Information Gain with Map Predictions for Indoor Robot Exploration

Seungjae Baek*[1,2], Brady Moon*[2], Seungchan Kim*[2], Muqing Cao[2], Cherie Ho[2], Sebastian Scherer†[2], Jeong hwan Jeon†[1]

***Abstract*— Autonomous exploration in unknown environments requires estimating the information gain of an action to guide planning decisions. While prior approaches often compute information gain at discrete waypoints, pathwise integration offers a more comprehensive estimation but is often computationally challenging or infeasible and prone to overestimation. In this work, we propose the Pathwise Information Gain with Map Prediction for Exploration (*PIPE*) planner, which integrates cumulative sensor coverage along planned trajectories while leveraging map prediction to mitigate overestimation. To enable efficient pathwise coverage computation, we introduce a method to efficiently calculate the expected observation mask along the planned path, significantly reducing computational overhead. We validate *PIPE* on real-world floorplan datasets, demonstrating its superior performance over state-of-the-art baselines. Our results highlight the benefits of integrating predictive mapping with pathwise information gain for efficient and informed exploration. Website: pipe-planner.github.io**

I. Introduction

Autonomous robots are often deployed to explore in unknown environments, where they must make decisions and take actions without knowing what future observations will reveal [1]–[3]. A key challenge of exploration is estimating the potential information gain from each action [4], which quantifies the value of the resulting sensor observations using metrics such as coverage [5], [6], uncertainty reduction [7], or map quality. This challenge is particularly important in indoor exploration, where limited sensing range and complex layouts can lead to inefficient navigation without informed decision-making [8].

A common approach to computing predicted information gain for an exploration action is to estimate the expected gain at possible next waypoints. For example, Bircher *et al.* [5] iteratively selects viewpoints that maximize the volume of unmapped, visible voxels within the sensor's field of view. These methods, which evaluate information gain at specific waypoints, can be referred to as *pointwise* information gain-based approaches. In contrast, a more comprehensive and accurate strategy is to adopt a *pathwise* information gain metric, which optimizes planned paths based on total information gain accumulated along the robot's trajectory. This better matches reality, where most robotic sensors continuously gather information, not merely at waypoints. Prior works have explored different aspects of pathwise information gain. In informative path planning, Binney *et al.* [9] proposed a broader formulation that accounts for information gain not only at individual graph nodes but also sensor measurements along the edges, such as temperature or depth, rather than modeling broader sensor coverage along the path. More specialized approaches, such as [10], focus on modeling sensor coverage along a path.

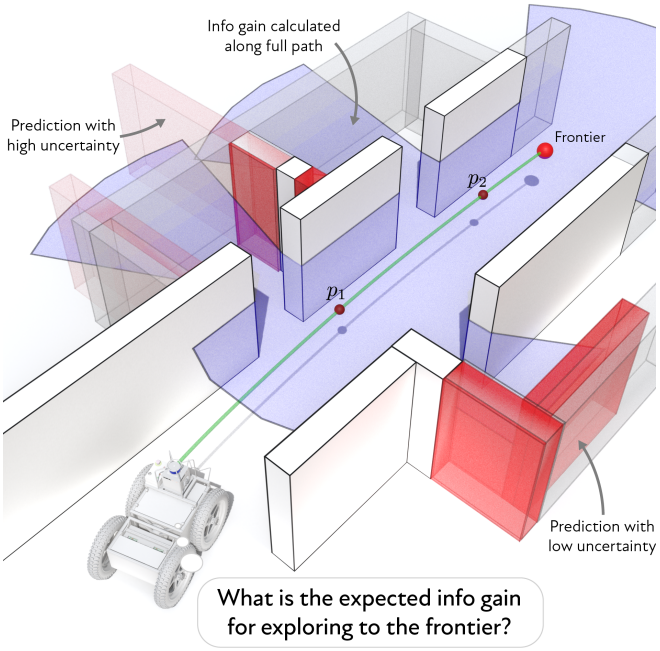

Fig. 1. Pathwise exploration often suffers from overestimating information gain. Our planner, ***PIPE***, solves this by using a map predictor to accurately calculate the expected information gain over the entire path, reasoning about both visibility and map quality.

In this work, we focus on pathwise sensor coverage, a subset of pathwise information gain that considers the sensor's ability to observe and map the environment along a planned trajectory. Pathwise integration of sensor coverage presents several challenges. First, it is often computationally expensive; for example, estimating camera or LiDAR coverage at every point along a path requires significant computation. Finding an efficient integration method remains a non-trivial problem. Second, information gain can be overestimated,

*: Equal Contributions †: Equal Advising

This work was supported by the National Institute of Advanced Industrial Science and Technology (AIST), DSTA contract No. DST000EC124000205, NSF GRFP under Grant No. DGE1745016, a hardware grant from Nvidia, and by Institute of Information & communications Technology Planning & Evaluation (IITP) grants by the Korean government (MSIT) (No.RS-2022-00143911, AI Excellence Global Innovative Leader Education Program; No.RS-2020-II201336, Artificial Intelligence graduate school support (UNIST); and No.RS-2024-00341055).

[1] Authors are with Ulsan National Institute of Science and Technology, Ulsan, Republic of Korea. {bsj970, jhjeon}@unist.ac.kr

[2] Authors are with the Robotics Institute, School of Computer Science at Carnegie Mellon University, Pittsburgh, PA, USA. {bradym, seungch2, muqingc, cherieh, basti}@andrew.cmu.edu

especially when undiscovered obstacles or occlusions exist beyond the robot's current field of view. Third, these estimation errors can accumulate over the trajectory, leading to compounding inaccuracies and suboptimal planning.

To address these challenges, we first focus on efficiently computing integrated information gain along a planned trajectory. We propose an approach that effectively estimates sensor coverage along the path by insightfully leveraging computational geometry, significantly reducing computational overhead. Second, to mitigate overestimation and error accumulation, we integrate map prediction into the robot's raycasting for sensor coverage estimation. Prior research [11]–[14] has shown that deep learning models can predict maps beyond observed areas, enhancing exploration in various ways. We specifically use such predictive models to refine sensor coverage estimates by generating plausible map predictions ahead of the robot. To the best of our knowledge, our work is the first to integrate map prediction-based exploration with a pathwise information gain metric, enabling more efficient exploration planning.

Combining these two aspects, we propose **P**athwise **I**nformation Gain with Map **P**rediction for **E**xploration (***PIPE***) planner. ***PIPE*** generates predictions of map beyond observed areas, computes cumulative sensor coverage along planned trajectories, and leverages predictive uncertainties as an information gain metric for planning. We validate ***PIPE*** on a real-world floorplan dataset, and show that it outperforms SOTA baselines across multiple evaluations.

The contributions of this work are as follows:

- We propose an indoor robot exploration planner, ***PIPE***, that estimates information gain using path-integrated cumulative sensor coverage and map predictions.
- We propose a fast method for estimating expected pathwise info gain by leveraging computational geometry.
- We show that ***PIPE*** outperforms state-of-the-art indoor exploration baselines on a real-world floorplan dataset in both budget-constrained and full exploration.

II. Related Work

A. Robot Exploration

Robot exploration is the process of sensing an environment and finding feasible paths to construct a map. Early approaches, such as frontier-based exploration [1], incrementally explore the environment by sequentially visiting frontiers—the boundary edges between observed and unobserved spaces. Other methods, known as information gain-based approaches, estimate information gain and select actions that maximize it. Various information gain metrics have been proposed, including maximizing 2D sensor coverage [15], [16], optimizing for 3D volumetric information gain [5], [6], [17], and reducing entropy or uncertainty [7]. In this work, our information gain metric is a combination of sensor coverage and uncertainty reduction.

B. Path Planning with Pathwise Information Gain

In path planning, reasoning about a robot's viewpoint is a common approach. Estimating the amount of information a robot can observe within its sensor's field of view at a given point has been widely studied in various areas, such as data gathering [18], [19], reconstruction [20], inspection [21], and exploration [5], [15], [22], [23]. Beyond viewpoint-based reasoning, some works have explored more densely sampled points along planned trajectories in the context of path planning. Binney *et al.* [9] proposed variants of informative path planning algorithms that leverage edge-based samples, taking dense points along edges into account. In contrast with sampling points along a path, Moon *et al.* [24] approximates the information gain along the path using an efficient lower bound. The closest work to ours is Liu *et al.* [10], which aims to estimate path information gain for exploration by measuring sensor coverage along the trajectory from the current frontier to the next. However, their approach approximates cumulative sensor coverage using a parallelogram-based estimation, which doesn't capture accurate information gain within complex layouts like indoor environments. Our work differs from previous approaches by providing a precise estimation of the information gain a robot can obtain along its path using raycast modeling. Additionally, we introduce a technique for efficiently computing the sensor coverage area, enabling more accurate and computationally feasible path information gain estimates for exploration planning.

C. Map Prediction for Exploration

Recent studies in exploration go beyond using only the observed area for path planning; instead, they leverage broader map predictions to improve planning. This technique, commonly used in indoor robot exploration, involves training deep learning models on offline datasets to generate plausible map predictions derived from the observed portion of a top-down occupancy map. These approaches guide robots to maximize information gain by leveraging the generated map predictions. For instance, [13] directs the robot toward more uncertain areas by summing variances along the trajectory, but it does not account for sensor coverage. In contrast, [12] guides the robot to regions where predicted sensor coverage is expected to be higher. A recent work [14] enables the robot to probabilistically reason about both sensor coverage and uncertainty simultaneously, choosing frontiers with a combination of high uncertainty and high coverage.

Our work also utilizes map prediction as a crucial component for exploration, helping to avoid overestimation when computing pathwise information gain. Without predicted maps, a robot planning a path beyond its observed area risks overestimating the amount of information its sensors could acquire. By assuming a plausible predicted map, we establish a better estimate of the environment and provide a reasonable bound on the expected information gain along the path.

III. Problem Statement

We define the indoor robot exploration problem as a robot navigating a 2D indoor environment $\mathcal{E} \subset \mathcal{R}^2$ to build a map of the environment. The robot's state at time t is given by $\mathbf{x}_t = [x_t, y_t]$, where $x_t, y_t \in \mathbb{Z}$. It perceives its surroundings using a noise-free 2D LiDAR sensor with a range of λ and

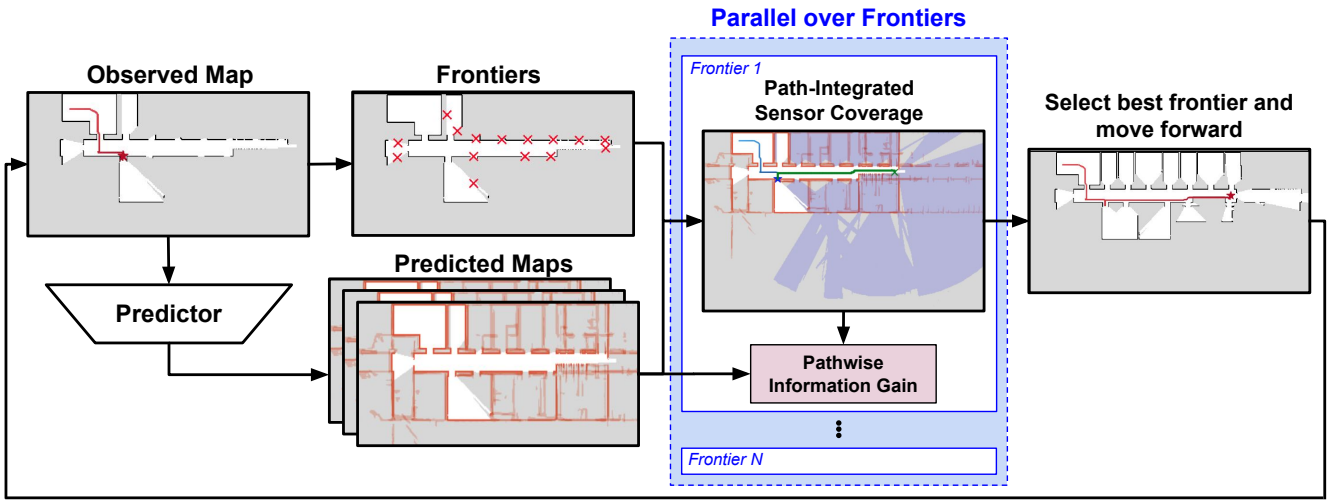

Fig. 2. Overview of ***PIPE*** planner pipeline. The robot extracts frontiers from its observed occupancy grid map and generates predicted maps. For each frontier, ***PIPE*** estimates path-integrated sensor coverage and generates a visibility mask for the full path. It computes pathwise information gain by summing the pixelwise variance of predictions within the visibility mask. The robot then selects the frontier with the highest information gain for exploration planning.

updates an (observed) occupancy grid map O_t, where each cell is classified as free, occupied, or unknown. The sensor collects l evenly distributed samples per scan over a full 360° field of view. At each time step, the robot moves Δ by selecting an action $a \in \mathcal{A}$, where $\mathcal{A}$ is a movement on an 8-connected grid. The objective of exploration is to determine a feasible path $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_T\}$ that constructs an accurate map representation within a fixed time T.

IV. Approach

In this section, we introduce the approach for ***PIPE*** planner, as summarized in Fig. 2. We begin by discussing exploration frameworks in Sec. IV-A and introducing the concept of sensor coverage and visibility masks in Sec. IV-B. In Sec. IV-C, we propose a path-integrated sensor coverage approach for computing information gain, followed by our methods to accelerate computation and optimize performance in Sec. IV-D. Finally, in Sec. IV-E, we explain how the pathwise information gain metric is integrated into the exploration planner and propose the ***PIPE*** planner.

A. Preliminary I: Exploration Framework

In this work, we use classical frontier-based exploration [1] as our planning framework. In frontier-based approaches, the robot identifies boundaries between known and unknown regions and selects the nearest frontier to explore. Building on this, we incorporate an information gain metric to select the next frontier to travel that maximizes expected information gain, following the "next-best-view" paradigm [5], [6], [15], [25], [26].

To quantify the expected information gain, we first generate a predicted map M_t from the observed occupancy grid O_t using a pretrained image inpainting network [27], similar to other prior map prediction-based exploration methods [12]–[14]. While frontiers are extracted from O_t, the information gain for traveling to the frontier is evaluated using both O_t and M_t by choosing actions that will reduce the uncertainty in M_t. Following [13], [14], we estimate prediction uncertainty map U_t by computing the pixelwise variance across an ensemble of predicted maps. The robot then selects the frontier that will result in observing the highest total amount of uncertainty relative to the path cost. In contrast with previous works, our work selects not the next best frontier observation but rather the holistic next best frontier by including the path to the frontier.

B. Preliminary II: Sensor Coverage and Visibility Masks

To estimate the LiDAR sensor coverage that a robot can obtain at a state $\mathbf{x}$, it must perform a raycast by emitting virtual rays from $\mathbf{x}$. For example, when calculating the sensor coverage at a frontier f in the 2D NBV-based methods [15], [25], we perform a raycast from f with a range of λ. The ray stops when it hits an occupied cell in O_t, while in free space, it extends until it reaches the range limit λ.

$$\{v_{1:l}\} \leftarrow \textsc{Raycast}(f, \lambda, O_t) \tag{1}$$

Here, $\{v_{1:l}\}$ are the vertices that each ray hits at the map, and l is the number of laser samples.

As we leverage map predictions, we can perform raycast on predicted map M_t instead of O_t,

$$\{v_{1:l}\} \leftarrow \textsc{Raycast}(f, \lambda, M_t) \tag{2}$$

While we can apply the same raycasting method to the predicted map M_t, where the ray stops upon hitting a predicted occupied cell, prior work [14] has shown that this approach can underestimate sensor coverage, particularly when erroneous predictions falsely classify free space as occupied. To address this, we adopt the probabilistic raycasting method proposed in [14]. Instead of terminating immediately upon encountering a predicted occupied cell, the ray starts with an initial value $\delta = 0$, and incrementally accumulates the values of predicted cells it traverses. Once δ reaches a predefined threshold ϵ, the raycasting stops. This probabilistic approach allows for tuning of the sensor coverage estimation by adjusting ϵ, balancing between overestimation and underestimation. Following prior work, we set $\epsilon = 0.8$.

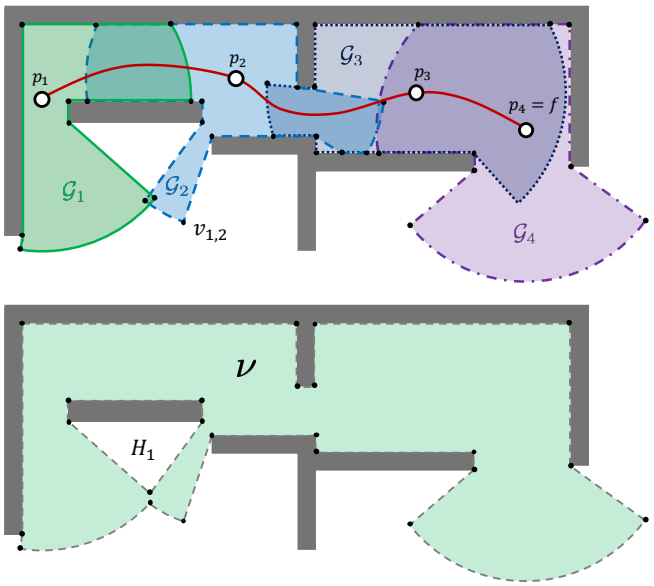

Fig. 3. A visualization of the path visibility mask computation. Individual polygons are first generated using raycasting. These polygons are then merged into a single polygon, including any holes, and a flood-fill algorithm is applied to determine the final visibility mask.

From the set of l vertices generated from the raycast, we can draw a polygon $\mathcal{G}$, and use a flood fill algorithm to generate a visibility mask ν, which is an estimate sensor coverage at the frontier point f.

$$\begin{aligned} \mathcal{G} &\leftarrow \textsc{DrawPolygon}(\{v_{1:l}\}) \\ \nu &\leftarrow \textsc{FloodFill}(\mathcal{G}) \end{aligned} \tag{3}$$

C. Path-Integrated Sensor Coverage for Information Gain

The core component of our ***PIPE*** planner is to evaluate the information gain of each frontier point $f \in \mathcal{F}$ (where $\mathcal{F}$ is a set of frontiers) by integrating the cumulative information gain along the path from the robot's current pose $\mathbf{x}_t$ to f_i, rather than estimating sensor coverage solely at f_i.

For each frontier $f_i \in \mathcal{F}$, we generate an A* path P_i connecting the robot's current pose $\mathbf{x}_t$ to f_i. Along P_i, we sample each point p_j to perform a raycast with a range of λ on the predicted maps M_t. These sampled points along P_i are every Δ, which is the same distance for each robot time step and update to the observed occupancy grid map O_t.

A simple approach to calculating the total visibility mask along the path is to first generate the visibility masks ν_j for each p_j:

$$\begin{aligned} \{v_{1:l}\}_j &\leftarrow \textsc{Raycast}(p_j, \lambda, M_t) \\ \mathcal{G}_j &\leftarrow \textsc{DrawPolygon}(\{v_{1:l}\}_j) \\ \nu_j &\leftarrow \textsc{FloodFill}(\mathcal{G}_j) \end{aligned} \tag{4}$$

The mask ν_j represents the sensor coverage the robot can obtain at each point p_j along the path to the frontier f_i. The path-integrated sensor coverage, ν, is then obtained by computing the union of each ν_j along the path. However, this straightforward approach to calculate pathwise information gain introduces significant computational overhead.

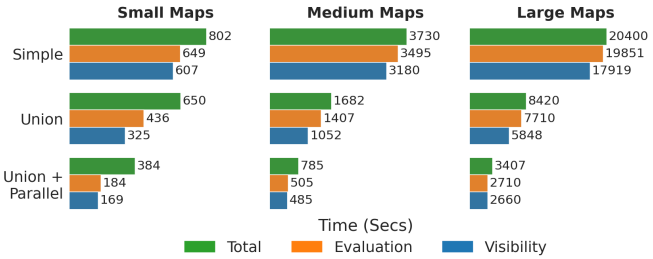

Fig. 4. Computation time comparison of our optimized approach (union + parallel) across map sizes, with the greatest reduction for large maps.

Algorithm 1 PathVisibilityMask

Input: Path P_i (from the robot's current pose $\mathbf{x}_t$ to the candidate frontier f_i), map M, LiDAR range λ

$\mathbb{G} = \{\}$ ▷ Initialize
for each point $p_j \in P_i$:
  ▷ **Probabilistic Raycast and Polygon Construction:**
  $\{v_{1:l}\}_j \leftarrow \textsc{Raycast}(p_j, \lambda, M)$ ▷ Vertices
  $\mathcal{G}_j \leftarrow \textsc{DrawPolygon}(\{v_{1:l}\}_j)$ ▷ Polygon
  $\mathbb{G} \leftarrow \mathbb{G} \cup \{\mathcal{G}_j\}$
▷ **Merge Polygons and Generate Visibility Mask:**
$\mathcal{G}_{union} \leftarrow \bigcup_{\forall j} \mathcal{G}_j$
$S \leftarrow \bigcup_{\forall j} (\mathcal{G}_j \triangle \mathcal{G}_{union})$
$H \leftarrow S \setminus \mathcal{G}_{union}$
$\nu \leftarrow \textsc{FloodFill}\ (\mathcal{G}_{union}, H)$

Output: Visibility Mask ν

D. Optimization of Calculating Information Gain

To identify the most computationally demanding components of our naive planner implementation, we conducted a simple experiment. We ran our planner on two small, two medium, and two large maps from the KTH dataset [28] (which will be further detailed in Sec. V-A), each with two different starting positions (one from a corner and one from the center), measuring the computation time. Computation time comparisons were conducted on a desktop computer equipped with an *Intel Core i5-10400F* CPU (2.90GHz), an *NVIDIA GeForce RTX 3060* GPU, and 16GB of RAM. The results are shown in Fig. 4. As expected, we found that in the simple approach, frontier evaluations accounted for 80.9%, 93.7%, and 97.3% of the total computation time in small, medium, and large maps, respectively. The increasing proportion in larger maps is likely due to the greater number of frontiers and longer path lengths. Among frontier evaluations, visibility mask generation was the most time-consuming step (blue bars in the figure), with repeated flood fill operations being the primary bottleneck.

To mitigate the prohibitive computational overhead, our implementation avoids applying a separate flood-fill to each raycast polygon $\mathcal{G}_j$. Instead, we first compute the union of all polygons to form a single composite shape, $\mathcal{G}_{union}$. By deferring the expensive flood-fill to a single, final step on this merged polygon, the overall computational cost is substantially reduced.

Since most paths contain multiple, typically nonconvex

Algorithm 2 ***PIPE*** Planner

Input: Robot's start pose $\mathbf{x}_0$, Time budget T

while $t \leq T$:
  ▷ **Sense and Update Map:**
  $O_t \leftarrow \textsc{Raycast}(\mathbf{x}_t, \lambda, O_t)$
  if robot needs to select a new waypoint:
    ▷ **Extract Frontiers and Predictions:**
    $\mathcal{F} \leftarrow \textsc{ExtractFrontiers}(O_t)$
    $M_t \leftarrow \textsc{GeneratePredictions}(O_t)$ ▷ Ensemble
    $U_t \leftarrow \textsc{ComputeVariance}(M_t)$
    ▷ **Evaluate Frontiers:**
    for frontier f in $\mathcal{F}$:
      $P \leftarrow \textsc{AStarPath}(\mathbf{x}_t, f)$
      $\nu \leftarrow \textsc{PathVisibilityMask}(P, M_t, \lambda)$ ▷ Alg. 1
      $I \leftarrow \sum_{(x_k, y_k) \in \nu} U_t(x_k, y_k)$
      $f.\text{score} \leftarrow I/|P|$ ▷ Normalize by path length
    $f_{\max} \leftarrow \arg\max_{f \in \mathcal{F}} f.\text{score}$
  ▷ **Plan Local Actions:**
  $a_t \leftarrow \textsc{AStarPath}(\mathbf{x}_t, f_{\max})$
  $\mathbf{x}_{t+1} \leftarrow \textsc{Transition}(\mathbf{x}_t, a_t)$
  $t \leftarrow t + 1$

Output: Final observed map O_T, final predicted map M_T

raycasted polygons, their union often creates holes H, or trapped regions. To identify these holes and account for them in the flood fill operation, we follow these steps: first, compute the symmetric difference S_j by performing symmetric difference operation $\triangle$ between each polygon $\mathcal{G}_j$ and the polygon union ($S \leftarrow \bigcup_{\forall j} (\mathcal{G}_j \triangle \mathcal{G}_{union})$). Then extract the parts of S that are not contained in the polygon union ($H \leftarrow S \setminus \mathcal{G}_{union}$). We then compute the final visibility mask ν using the polygon union and holes ($\nu \leftarrow \textsc{FloodFill}(\mathcal{G}_{union}, H)$). The full pseudocode for this process is shown in Alg. 1, with a visualization in Fig. 3.

Additionally, to take advantage of computational parallelism, we used Python's multiprocessing package to distribute the computation across multiple CPU cores, with each process handling a separate frontier. In an ideal scenario, this approach can reduce computation time by a factor approximately equal to the number of available CPU cores, assuming efficient implementation and minimal overhead.

The impact of these optimizations, shown in Fig. 4, is most pronounced in the large maps, with a 83.3% reduction in computation time.

E. *PIPE* Planner

Lastly, we outline the overall procedure of ***PIPE*** planner. When the robot needs to select a new waypoint, it first extracts a set of candidate frontier points, $\mathcal{F}$, from the current occupancy grid O_t, and generates predictions M_t and uncertainty map U_t. For each frontier $f \in \mathcal{F}$, it generates an A* path, P, from current pose $\mathbf{x}_t$ to the frontier f, and then applies Alg. 1 to compute the pathwise sensor coverage, yielding the visibility mask ν. The information gain, I, is then calculated by summing the uncertainties within the visibility mask and normalizing by the length of the path P. The frontier with the highest score is selected as the next waypoint until it has been reached, at which a new waypoint is selected through the same process described as above. The complete procedure of ***PIPE*** planner is presented in Alg. 2.

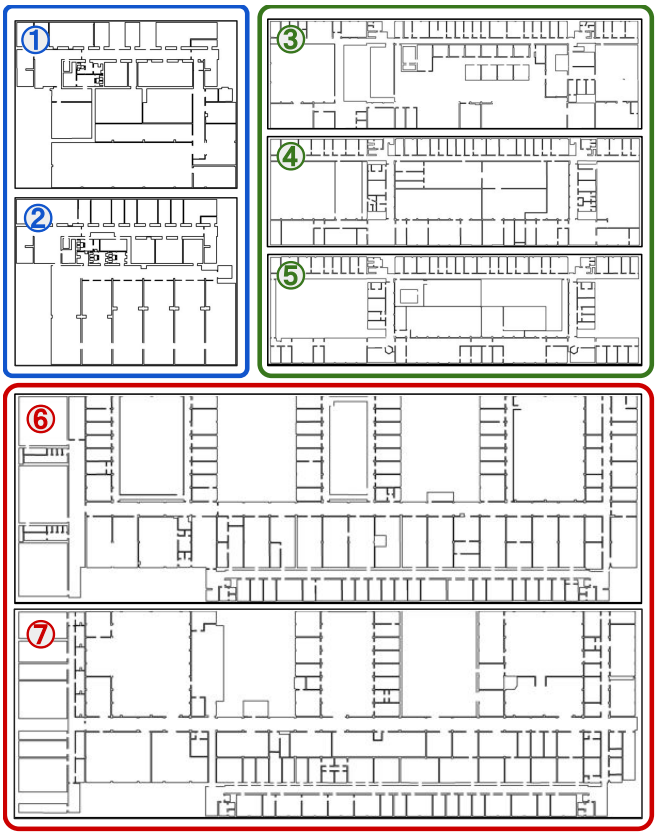

Fig. 5. The seven indoor maps used to evaluate our method. Small, medium, and large maps are shown in blue, green, and red, respectively. The maps have various topologies and are not shown to scale relative to each other.

V. Experiments

In this section, we describe the experimental setup, baselines, and evaluation metrics, followed by the experimental results. We provide both qualitative and quantitative analyses comparing our ***PIPE*** planner against the baselines.

A. Experimental Setup

We use a custom-built simulator based on the KTH floorplan dataset [28], which includes over 100 campus floorplans represented as XML files containing wall and door locations for each room. To ensure accurate raycasting, we corrected any inaccuracies in the maps that could lead to false projections into invalid spaces. These corrected maps serve as the simulator's ground truth, while the planner itself utilizes the LaMa model [27] to generate map predictions from online observations. The maps were downsampled to a resolution of 10 pixels per meter.

From the dataset, we selected seven floorplans, categorized into three sizes: two small (65 m × 85 m), three medium (60 m × 205 m), and two large maps (88 m × 265 m). Fig. 5 visualizes these environments. For each floorplan, the robot

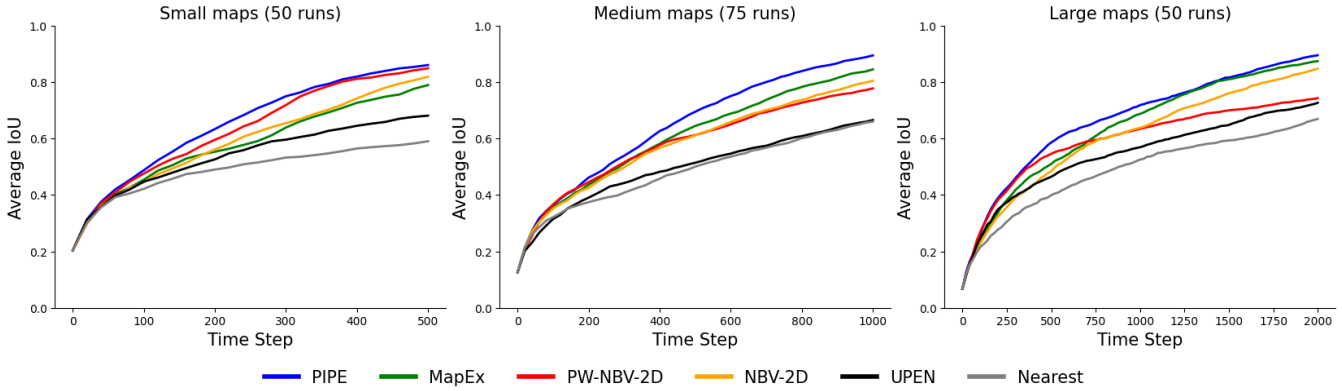

Fig. 6. Quantitative comparison of methods for each map size shows ***PIPE*** outperforms all others in most time steps, with the second-best method varying by map size. Prediction-based methods excel on larger maps, while pathwise NBV-2D performance declines as map size increases.

Method	Small Maps	Medium Maps	Large Maps
PIPE (ours)	**327.00**	**648.35**	**1338.16**
MapEx [14]	290.63	607.05	1278.39
PW-NBV-2D	316.81	587.59	1184.69
NBV-2D [25]	295.68	582.70	1209.37
UPEN [13]	270.04	494.08	1084.57
Nearest [1]	244.80	482.79	971.30

TABLE I. Area Under the Curve (AUC) of IoU for each method. Higher is better. ***PIPE*** outperforms other methods in all maps.

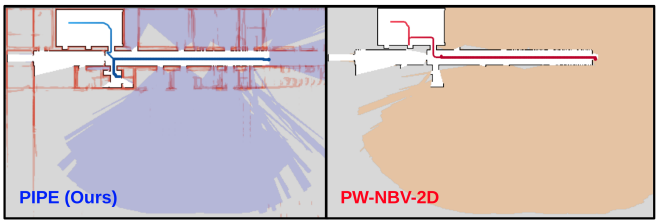

Fig. 7. Qualitative comparison between sensor coverages estimated by ***PIPE*** (left) and PW-NBV-2D (right). PW-NBV-2D overestimates cumulative sensor coverage, while ***PIPE*** uses predictions to mitigate it.

starts from 25 distinct initial locations. In total, we conducted 175 experiments to evaluate performance.

B. Baselines

We select five baselines for comparison, a conventional nearest frontier-based method, two for observed map based method, and other two for map prediction based exploration methods.

- Nearest [1]: A classical exploration method that selects frontiers based on the shortest Euclidean distance.
- NBV-2D [15], [25]: A "Next-best-view" strategy where, for each frontier, the robot performs virtual raycasting on the observed map O_t, computes sensor coverage, and normalize it by the Euclidean distance. *(Pointwise, No Prediction, Sensor Coverage-Based)*
- PW-NBV-2D: An extension of NBV-2D from a pointwise to a pathwise approach. The robot performs raycasting from multiple points along the path on O_t, aggregates the sensor coverage as information gain, and normalizes it by the A* path distance. *(Pathwise, No Prediction, Sensor Coverage-Based)*
- UPEN [13]: A map-prediction-based approach that leverages an ensemble of predicted maps to estimate uncertainty. Information gain is computed as the cumulative sum of variances (uncertainty) along the path, normalized by path distance. *(Pathwise, Prediction-Based, No Sensor Coverage, but Uncertainty-Only)*
- MapEx [14]: A map-prediction-driven exploration frontiers are scored using pointwise probabilistic raycasting on predicted maps, combined with uncertainty map, and normalized by Euclidean distance. *(Pointwise, Prediction-Based, Sensor Coverage and Uncertainty)*

C. Metrics

We assess the performance of the planners using the IoU (Intersection over Union) of the occupied cells in the predicted and ground truth maps. IoU measures the quality of the predicted 2D occupancy map, by comparing the map's predicted occupied cells with the ground truth occupancy, calculated as IoU $= \frac{\text{TP}}{\text{FP+FN+TP}}$. Higher IoU means the algorithm can create a more accurate predicted map. We also add a small buffer in the IoU calculations to not penalize small pixel-wise errors in the map, but rather targeting general map accuracy. Methods that leverage prediction (***PIPE***, MapEx, UPEN) have a clear definition of predicted maps, whereas methods without prediction (NBV-2D, PW-NBV-2D, Nearest) lack them. For these non-prediction-based methods, we collect the observed maps at every time step and generate predicted maps afterward to define predicted IoU.

D. Experimental Results

In this section, we first present the quantitative evaluation of our experiments. We analyze how much each method explores within a fixed time budget in Sec. V-D.1 and examine the time required to complete map exploration, along with their later-stage behaviors, in Sec. V-D.2.

1) IoU Comparison with time budgets: Fig. 6 compares exploration progress over time across three map scales for each method. Overall, we observe that ***PIPE*** consistently outperforms all other methods, demonstrating stable and

	Small Maps (50 runs)				Medium Maps (75 runs)				Large Maps (50 runs)			
Method	**90% IoU**	**FR (%)**	**95% IoU**	**FR (%)**	**90% IoU**	**FR (%)**	**95% IoU**	**FR (%)**	**90% IoU**	**FR (%)**	**95% IoU**	**FR(%)**
PIPE (ours)	**505±41**	0	**718±63**	0	**1039±56**	0	**1497±100**	0	**1979±115**	0	**2658±103**	0
MapEx [14]	610±38	0	752±58	0	1165±72	0	1602±101	3	2266±164	0	3399±217	2
PW–NBV–2D	519±40	0	784±94	12	1643±146	15	1956±178	51	3732±243	4	4825±280	32
NBV–2D [25]	577±34	0	852±70	0	1321±65	0	1694±104	4	2336±89	0	3037±116	0
UPEN [13]	833±85	34	1009±107	64	2171±168	40	2337±175	63	4085±301	18	4875±394	62
Nearest [1]	1050±57	8	1160±51	12	2273±77	11	2524±81	33	4109±249	2	4634±199	4

TABLE II. Comparison of the time–steps required to reach 90 % and 95 % IoU on ground–truth maps. Values are reported as *mean* ± *95 % confidence interval*; failed runs are excluded when computing the means and CIs. The failure rate (FR) column shows the percentage of runs that did not attain the target IoU within the maximum time-steps (1,500 for **small**, 3,000 for **medium**, and 6,000 for **large** maps). Lower numbers indicate better performance.

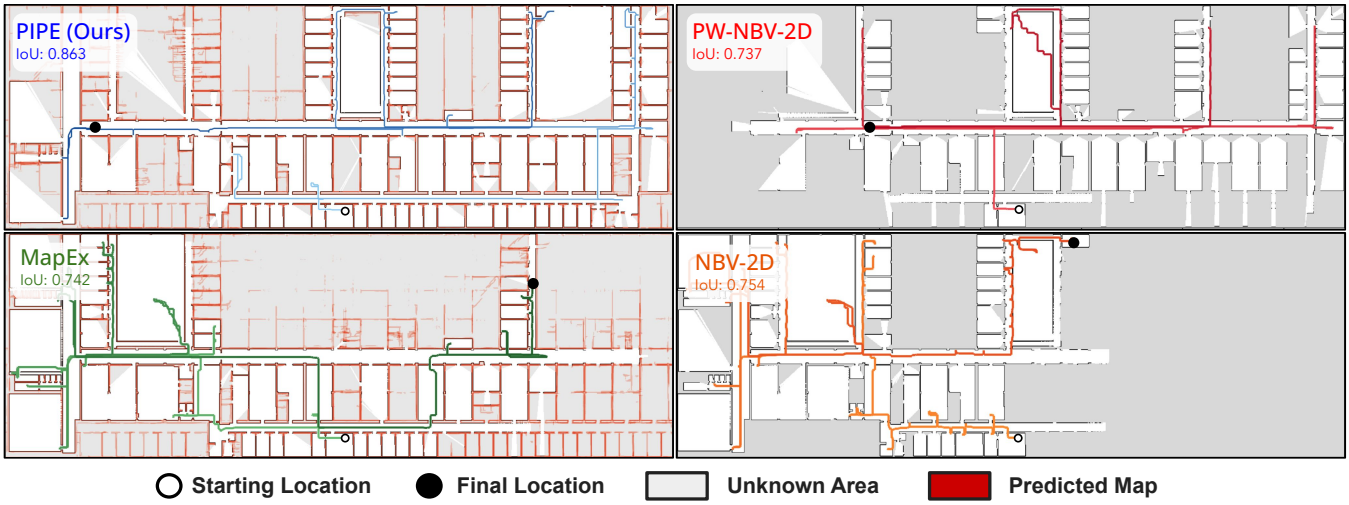

Fig. 8. Qualitative comparison of methods at time step $T = 2000$ on Map 6, with the path transitioning from light to dark to indicate the passage of time. ***PIPE*** performs the best, with more even exploration across the space and less backtracking compared to other methods.

high performance across all map sizes. In small maps, ***PIPE*** achieves the highest final IoU, surpassing PW-NBV-2D by 1.15%, MapEx by 7.08%, NBV-2D by 4.13%, UPEN by 17.91%, and Nearest by 26.99%.

In the medium-sized maps, ***PIPE*** maintains its lead, though the ranking among the baselines shifts. Here, MapEx emerges as the second-best method, outperforming the rest. At $T = 1000$, ***PIPE*** exceeds PW-NBV-2D by 11.69%, MapEx by 4.93%, NBV-2D by 8.99%, UPEN by 22.89%, and Nearest by 23.45%.

In large maps, ***PIPE***, MapEx, and NBV-2D perform at comparable levels, but PW-NBV-2D experiences a significant drop in performance. At $T = 2000$, ***PIPE*** outperforms MapEx by 2.1%, NBV-2D by 4.73%, UPEN by 16.85%, and Nearest by 22.57%. Although the final IoU differences between the top three methods appear moderate, the trajectory suggests that MapEx is reaching saturation, while ***PIPE***'s IoU continues to increase. This trend is further analyzed in Sec. V-D.2. Additionally, ***PIPE*** surpasses PW-NBV-2D by 15.20% in large maps—a notably larger gap than in small and medium maps—indicating that pathwise integration of information gain alone is insufficient and can even degrade performance when combined with certain estimation methods.

The performance of each method can also be measured with Area Under the Curve (AUC) of each method's plot. According to Table I, ***PIPE*** surpassed all other methods for every map size, verifying its efficient map exploration.

2) Complete Map Exploration: In addition to evaluating each method's performance under fixed time budgets, we also analyzed their ability to achieve complete map exploration. Here, we define this as reaching at least 95% IoU with the ground truth map. To better understand the long-tail behavior of the methods, we also report results for 90% IoU, highlighting the difficulty of achieving the additional 5% increase. For both IoU thresholds, we measure the number of time steps required by each method.

The results are shown in the Table II. Overall, ***PIPE*** outperformed all other methods in both efficiency and stability. It required the fewest time steps to reach both 90% and 95% IoU across all maps and was the only method to achieve a zero failure rate for complete exploration at both thresholds.

3) Qualitative Discussion: As shown in Fig. 6 and Table I, pathwise methods (***PIPE***, PW-NBV-2D) outperform pointwise methods (MapEx, NBV-2D) in small maps. However, in medium and large maps, ***PIPE*** continues to outperform MapEx, while PW-NBV-2D performs notably worse. This suggests that simply integrating cumulative sensor coverage can degrade performance as map size increases. The qualitative comparison in Fig. 7 provides insights into why PW-NBV-2D struggles in larger maps. ***PIPE*** computes sensor coverage using probabilistic raycasting over predicted maps, providing a more conservative and accurate estimation of unobserved areas. In contrast, PW-NBV-2D significantly

overestimates sensor coverage in unobserved regions. While the accumulation of errors along the pathwise raycasts may have little noticeable impact in small maps, it becomes increasingly significant as map size and complexity increase.

Lastly, Fig. 8 presents an example of the exploration progress of ***PIPE***, MapEx, PW-NBV-2D, and NBV-2D. All methods start from the same location. The two pointwise methods, NBV-2D and MapEx, exhibit similar behaviors, primarily focusing on the left half of the map. This highlights a key limitation of pointwise methods—their tendency to fixate on short-sighted local information gain. Meanwhile, PW-NBV-2D initially explores effectively but later gravitates toward traversing long corridors, skipping rooms. In contrast, ***PIPE*** leverages both predictive uncertainties and pathwise coverage, balancing exploration between rooms and corridors, ultimately achieving the best performance.

VI. Conclusion

In this work, we presented ***PIPE***, a planner based on Pathwise Information Gain with Map Prediction for Exploration. Our approach addresses the need to compute information gain cumulatively along a path rather than merely at a point. To achieve this efficiently, we proposed techniques to reduce computational overhead and incorporated map predictions to mitigate the overestimation of cumulative sensor coverage. Experiments on a real-world floorplan dataset simulator demonstrated that ***PIPE*** outperforms baselines, including map-prediction-based exploration method relying solely on pointwise gain and pathwise exploration methods without map prediction. For future work, we aim to extend our research to real-world robot deployment.

References

[1] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. of IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pp. 146–151, IEEE, 1997.

[2] H. Umari and S. Mukhopadhyay, "Autonomous robotic exploration based on multiple rapidly-exploring randomized trees," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1396–1402, IEEE, 2017.

[3] S. Scherer, V. Agrawal, G. Best, C. Cao, K. Cujic, R. Darnley, R. DeBortoli, E. Dexheimer, B. Drozd, R. Garg, *et al.*, "Resilient and modular subterranean exploration with a team of roving and flying robots," *Field Robotics*, 2022.

[4] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using rao-blackwellized particle filters.," in *Robotics: Science and systems*, vol. 2, pp. 65–72, 2005.

[5] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon" next-best-view" planner for 3d exploration," in *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 1462–1468, IEEE, 2016.

[6] J. Delmerico, S. Isler, R. Sabzevari, and D. Scaramuzza, "A comparison of volumetric information gain metrics for active 3d object reconstruction," *Autonomous Robots*, vol. 42, no. 2, pp. 197–208, 2018.

[7] J. Vallvé and J. Andrade-Cetto, "Dense entropy decrease estimation for mobile robot exploration," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6083–6089, IEEE, 2014.

[8] S. Kim, M. Corah, J. Keller, G. Best, and S. Scherer, "Multi-robot multi-room exploration with geometric cue extraction and circular decomposition," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1190–1197, 2023.

[9] J. Binney, A. Krause, and G. S. Sukhatme, "Optimizing waypoints for monitoring spatiotemporal phenomena," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 873–888, 2013.

[10] J. Liu, C. Wang, W. Chi, G. Chen, and L. Sun, "Estimated path information gain-based robot exploration under perceptual uncertainty," *Robotica*, vol. 40, no. 8, pp. 2748–2764, 2022.

[11] K. Katyal, K. Popek, C. Paxton, P. Burlina, and G. D. Hager, "Uncertainty-aware occupancy map prediction using generative networks for robot navigation," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5453–5459, 2019.

[12] R. Shrestha, F.-P. Tian, W. Feng, P. Tan, and R. Vaughan, "Learned map prediction for enhanced mobile robot exploration," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 1197–1204, IEEE, 2019.

[13] G. Georgakis, B. Bucher, A. Arapin, K. Schmeckpeper, N. Matni, and K. Daniilidis, "Uncertainty-driven planner for exploration and navigation," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 11295–11302, IEEE, 2022.

[14] C. Ho, S. Kim, B. Moon, A. Parandekar, N. Harutyunyan, C. Wang, K. Sycara, G. Best, and S. Scherer, "Mapex: Indoor structure exploration with probabilistic information gain from global map predictions," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[15] H. H. González-Banos and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *The International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 829–848, 2002.

[16] D. Deng, R. Duan, J. Liu, K. Sheng, and K. Shimada, "Robotic exploration of unknown 2d environment using a frontier-based automatic-differentiable information gain measure," in *2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 1497–1503, IEEE, 2020.

[17] M. Corah and N. Michael, "Volumetric objectives for multi-robot exploration of three-dimensional environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9043–9050, IEEE, 2021.

[18] M. Popović, G. Hitz, J. Nieto, I. Sa, R. Siegwart, and E. Galceran, "Online informative path planning for active classification using uavs," in *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 5753–5758, IEEE, 2017.

[19] B. Moon, N. Suvarna, A. Jong, S. Chatterjee, J. Yuan, and S. Scherer, "Ia-tigris: An incremental and adaptive sampling-based planner for online informative path planning," 2025.

[20] M. Maboudi, M. Homaei, S. Song, S. Malihi, M. Saadatseresht, and M. Gerke, "A review on viewpoints and path planning for uav-based 3-d reconstruction," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 16, pp. 5026–5048, 2023.

[21] A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, and R. Siegwart, "Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6423–6430, IEEE, 2015.

[22] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. Kumar, "Information-theoretic planning with trajectory optimization for dense 3d mapping.," in *Robotics: Science and Systems*, vol. 11, pp. 3–12, Rome, 2015.

[23] Y. Zhang, X. Chen, C. Feng, B. Zhou, and S. Shen, "Falcon: Fast autonomous aerial exploration using coverage path guidance," *IEEE Transactions on Robotics*, 2024.

[24] B. Moon, S. Chatterjee, and S. Scherer, "Tigris: An informed sampling-based algorithm for informative path planning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5760–5766, 2022.

[25] A. Visser, M. Van Ittersum, L. A. González Jaime, and L. A. Stancu, "Beyond frontier exploration," in *RoboCup 2007: Robot Soccer World Cup XI 11*, pp. 113–123, Springer, 2008.

[26] J. I. Vasquez-Gomez, L. E. Sucar, and R. Murrieta-Cid, "View/state planning for three-dimensional object reconstruction under uncertainty," *Autonomous Robots*, vol. 41, pp. 89–109, 2017.

[27] R. Suvorov, E. Logacheva, A. Mashikhin, A. Remizova, A. Ashukha, A. Silvestrov, N. Kong, H. Goka, K. Park, and V. Lempitsky, "Resolution-robust large mask inpainting with fourier convolutions," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2149–2159, 2022.

[28] A. Aydemir, P. Jensfelt, and J. Folkesson, "What can we learn from 38,000 rooms? reasoning about unexplored space in indoor environments," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4675–4682, IEEE, 2012.