GLEAM: Learning Generalizable Exploration Policy for Active Mapping in Complex 3D Indoor Scenes

Xiao Chen[1,2] Tai Wang[2] Quanyi Li[2] Tao Huang[2] Jiangmiao Pang[2] Tianfan Xue[1]
[1]The Chinese University of Hong Kong [2]Shanghai AI Laboratory

Project Website: xiao-chen.tech/gleam

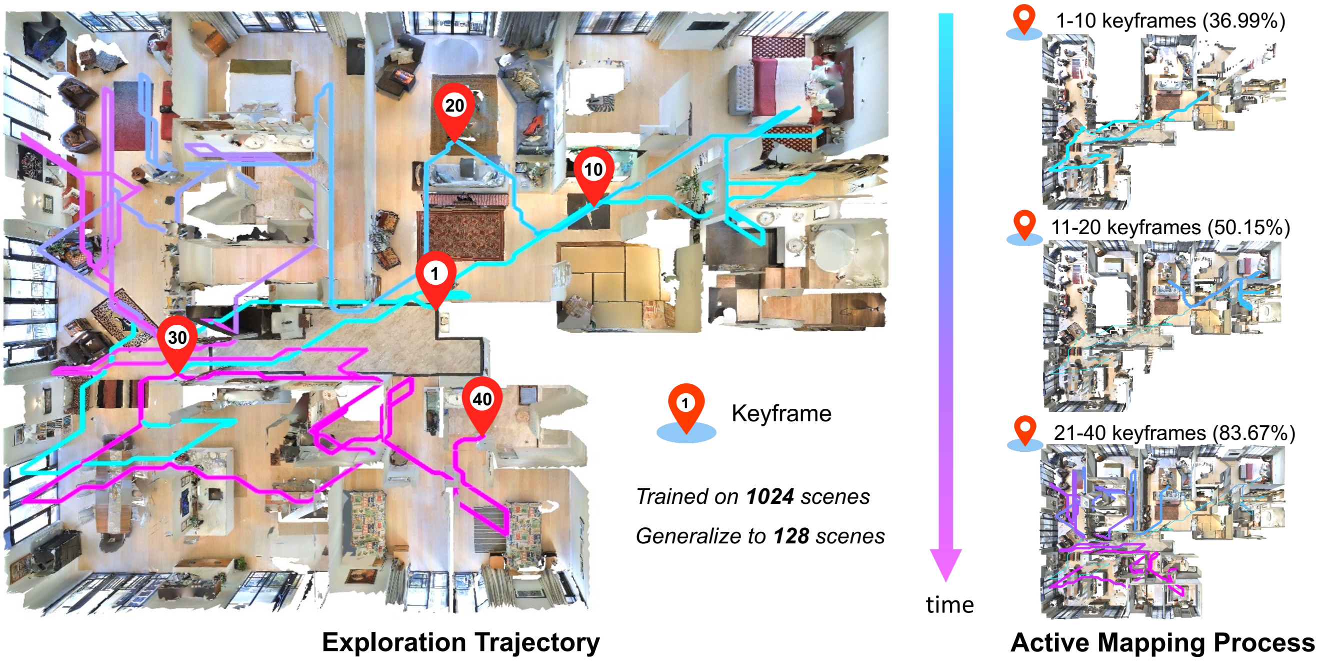

Figure 1. We introduce **GLEAM**, a unified generalizable exploration policy for active mapping in complex 3D indoor scenes, trained and evaluated on 1,152 diverse scenes from our benchmark **GLEAM-Bench**. Our cross-dataset generalization to an unseen real-scan scene from Matterport3D [6] achieves 83.67% coverage using 40 keyframes, without any fine-tuning and prior knowledge.

Abstract

*Generalizable active mapping in complex unknown environments remains a critical challenge for mobile robots. Existing methods, constrained by insufficient training data and conservative exploration strategies, exhibit limited generalizability across scenes with diverse layouts and complex connectivity. To enable scalable training and reliable evaluation, we introduce **GLEAM-Bench**, the first large-scale benchmark designed for generalizable active mapping with 1,152 diverse 3D scenes from synthetic and real-scan datasets. Building upon this foundation, we propose **GLEAM**, a unified generalizable exploration policy for active mapping. Its superior generalizability comes mainly from our semantic representations, long-term navigable goals, and randomized strategies. It significantly outperforms state-of-the-art methods, achieving 66.50% coverage (+9.49%) with efficient trajectories and improved mapping accuracy on 128 unseen complex scenes.*

1. Introduction

While existing methods have enabled robots to represent [21, 30] and reconstruct [31, 43] 3D environments through predefined trajectories or offline visual data, autonomous exploration and mapping in unknown 3D environments remain a cornerstone challenge for robots. In unknown environments with complex connectivity, robots must strategically prioritize unexplored areas while balancing exploration efficiency. Classic active SLAM [7, 8] and active mapping [15, 35, 49] have been investigated in the context of small-scale or simple scenarios, often assuming in-distribution environments with few rooms.

The generalization of existing active mapping methods to unknown scenes remains inadequately studied. Three primary challenges emerge across data, technical frameworks, and training strategies. First, most existing methods [7, 8, 15, 35] are trained on fewer than 100 homogeneous scenarios, failing to leverage data diversity for robust generalization, as shown in Tab. 1. Second, many existing approaches rely on empirically defined heuristics to guide the exploration, including information gain [49], gain or layout anticipations [15, 27, 35], and structured map [5, 8]. The heuristic metrics hinder their generalization in heterogeneous environments with diverse obstacle layouts and topological connectivity. Third, previous training settings such as centralizing starting positions [7] simplify the exploration pattern, thus diminishing the policy's capacity to explore complex interconnected spaces.

Therefore, to build an active mapping system that can generalize to different complex indoor scenes, we build **GLEAM-Bench**, a new training dataset and evaluation benchmark. It is the first large-scale exploration dataset encompassing over 1,000 complex 3D indoor scenes for training and more than 100 scenes for evaluation, with parallel simulation support. To show the importance of using a large-scale benchmark, we train the previous state-of-the-art active mapping algorithms, ANS [7] and OccAnt [35], on 32 scenes with 12 rooms. As shown in Fig. 2, the trained models can perform well in scenes with 10 rooms, but they fail in even simpler environments with only 5 rooms, demonstrating their poor generalization. Therefore, in order to ensure the generalization of new exploration policies, both the training and evaluation datasets contain indoor scenes from different datasets, including both synthetic ones (ProcTHOR [12], HSSD [22]) and real-scanned ones (Gibson [47], Matterport3D [6]), with 1,152 scenes in total, ensuring both diversity and complexity.

This new benchmark motivates a better active mapping algorithm that can generalize across diversified indoor scenes. Therefore, we subsequently propose **GLEAM**, a reinforcement learning (RL)-based exploration policy that can be generalized to complex unseen indoor scenes without any fine-tuning or prior knowledge. Our policy achieves better generalization through the following three key designs. First, it maintains a global probabilistic map that integrates historical observations and a semantic egocentric map with four task-related states, enabling environment-agnostic spatial reasoning—the lightweight LocoTransformer [50] distills these states into task-aware embeddings without scene-specific architectural constraints. Second, we replace classic motion primitives with long-horizon action spaces validated by heuristic planners, enabling agents to focus on high-level exploration while offloading low-level path safety at the early training stage. Third, the training strategies include randomized initial poses to enhance generalization by exposing agents to broader environmental variations, thereby mitigating scene-specific overfitting and improving policy robustness.

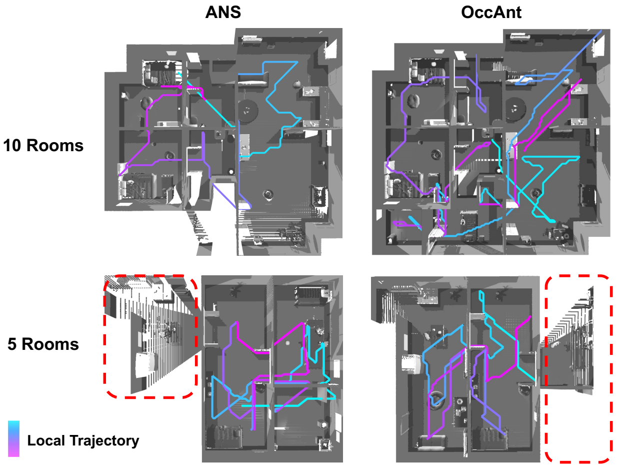

Figure 2. Despite being trained on complex indoor scenes with 10 rooms, classic RL-based methods, ANS [8] and OccAnt [35] exhibit limited generalizability when tested on simple but structurally distinct scenes with 5 rooms.

Table 1. The data sources of existing exploration policies for active mapping. †: The SUN-CG dataset is not available now. "*": The policies are based on neural representations and thus need per-scene optimization on evaluation scenes.

Method	Data Source	Training	Evaluation
ExploreNav [8]	SUN-CG† [41]	20	20
ANS [7]	SUN-CG†	20	20
OccAnt [35]	Gibson [47], MP3D [6]	72	18
UPEN [15]	MP3D	61	18
ANM* [49]	Gibson, MP3D	N/A	14
NARUTO* [14]	Replica [42], MP3D	N/A	13
GLEAM (Ours)	ProcTHOR [12], MP3D, HSSD [22], Gibson	**1,024**	**128**

At last, to validate the generalizability of GLEAM, we conduct comprehensive evaluations on our benchmark comprising 128 challenging scenes from four distinct datasets. Without any fine-tuning, GLEAM achieves 66.50% average coverage ratio across all scenes, surpassing the previous state-of-the-art ANM [49] by 11.41% while concurrently improving path efficiency (+9.49% AUC of coverage) and reconstruction completeness (-0.22m nearest distance).

2. Related Work

Active Mapping. Active mapping is a promising field yet to be thoroughly benchmarked. The active mapping pipeline alternates between inferring the next optimal viewpoints, capturing new data, and updating the built 3D modeling. Classic active SLAM [7, 8, 28] and active

Dataset	**Type**	**#Scenes**	**NS(m^2)**	**FS(m^2)**	**NC**	**SC**
Replica [42]	synthetic	18	0.56k	2.19k	5.99	3.40
HSSD [22]	synthetic	211	53.21k	-	13.7	5.90
ProcTHOR-10K [12]	synthetic	10k	220k	-	-	-
Gibson (4+ only) [47]	real-scan	106	7.18k	17.74k	11.90	3.04
Matterport3D [6]	real-scan	90	30.22k	101.82k	17.09	2.99
GLEAM-Bench	mixed	1,152	91.16k	164.66k	11.35	3.44

Table 2. The description of the mentioned dataset in Table 1. Note that only a few high-quality scenes from these datasets are used by previous active mapping algorithms. **NS**: navigable space, **FS**: floor space, **NC**: navigation complexity, **SC**: scene clutter. "-": challenging to access without annotation or statistical documentation.

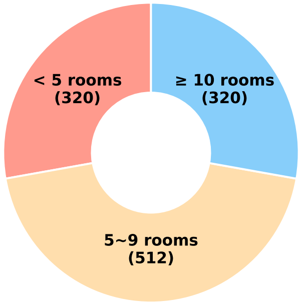

Figure 3. The distribution of 1,152 scenes by the number of rooms in our benchmark GLEAM-Bench.

mapping [15, 35, 49] have been investigated in the context of small-scale or simple scenarios, often assuming in-distribution environments with few rooms. As one of the most representative heuristic policies, frontier-based exploration (FBE) policy [2, 13, 48] recognizes the boundary between explored areas and unknown areas and then navigates agents to the frontier. However, these policies usually rely on impractical criteria such as always moving to the closest frontier and don't leverage the semantic priors like diverse indoor layouts, thus cannot effectively generalize to the complex unseen scenes in the real world.

Information gain-based policies are also known as uncertainty-driven or utility-driven policies. Classic works [3, 18] use information theory to quantify the information gain for a probabilistic volumetric representation. A recent group of works [19, 25, 36, 49, 53] leverages neural implicit representation, such as SDF [32] and NeRF [30], to model the uncertainty fields of target scenes.

Existing benchmark for Exploration. Existing benchmarks [6, 11, 42, 46, 47, 51] largely derived from 3D datasets tailored for perception, short-range navigation, or small-scale scene reconstruction. These benchmarks suffer from three critical constraints: (1) Restricted environmental scales with discontinuous room layouts, inadequate for long-horizon task requirements; (2) Oversimplify collision dynamics by neglecting dense obstacle arrangements, resulting in policies that struggle in cluttered real-world settings; (3) Fragmented geometric surfaces in real-scan environments that compromise exploration policies' ability to apply learned scene priors. While these benchmarks enable basic validation of exploration efficiency and mapping quality, they lack diversity and complexity in scene layouts, asset quality, and topological structures.

The Generalizability of Exploration Policies. The generalization of exploration policies for active mapping in unknown environments remains underexplored, with three key limitations in existing approaches. First, data scarcity and homogeneity hinder robust generalization: most methods [7, 8, 15, 35] are trained on fewer than 100 scenes, often with uniform room counts or layouts, leading to overfitting specific or simple environmental patterns. Second, classic technical frameworks rely on rigid heuristics, such as handcrafted information gain [49], layout anticipations [27, 35], or structured map priors [5, 8], which struggle to adapt to unseen obstacle configurations or topological variations. For example, methods using motion primitives or fixed action spaces [8, 35] fail to handle long-horizon exploration in interconnected spaces. Third, simplified training strategies—such as centralized agent starting positions [8, 35]—reduce exposure to environmental variability, weakening policies' adaptability to complex initial conditions. These limitations collectively constrain the deployment of exploration policies in real-world scenarios with heterogeneous layouts and dynamic connectivity.

3. GLEAM-Bench

We introduce GLEAM-Bench, a benchmark for generalizable exploration for active mapping in complex 3D indoor scenes. The statistical metrics can be found in Table 2 and Fig. 3, following [6, 46**?**]. These scene meshes are characterized by watertight geometry, diverse floorplan (≥10 types), and complex interconnectivity. We unify and refine multi-source datasets through manual filtering, geometric repair, and task-oriented preprocessing. To simulate the exploration process, we connect our dataset with NVIDIA Isaac Gym [29], enabling parallel sensory data simulation and online policy training, achieving 150 FPS on an RTX 3090 GPU, even trained on 512 complex scenes. Additional details about GLEAM-Bench are provided in Appendix A.

3.1. Datasets

Our benchmark features ***1,152* diverse complex 3D scenes** from different indoor datasets, including realistic synthetic datasets (ProcTHOR [12], HSSD [22]) and two real-scan datasets (Gibson [47], Matterport3D [6]). It is built to overcome the problem of data scarcity and homogeneity faced by most existing methods, as shown in Table 1. Concretely, we scale up, filter, and preprocess scenes using uni-

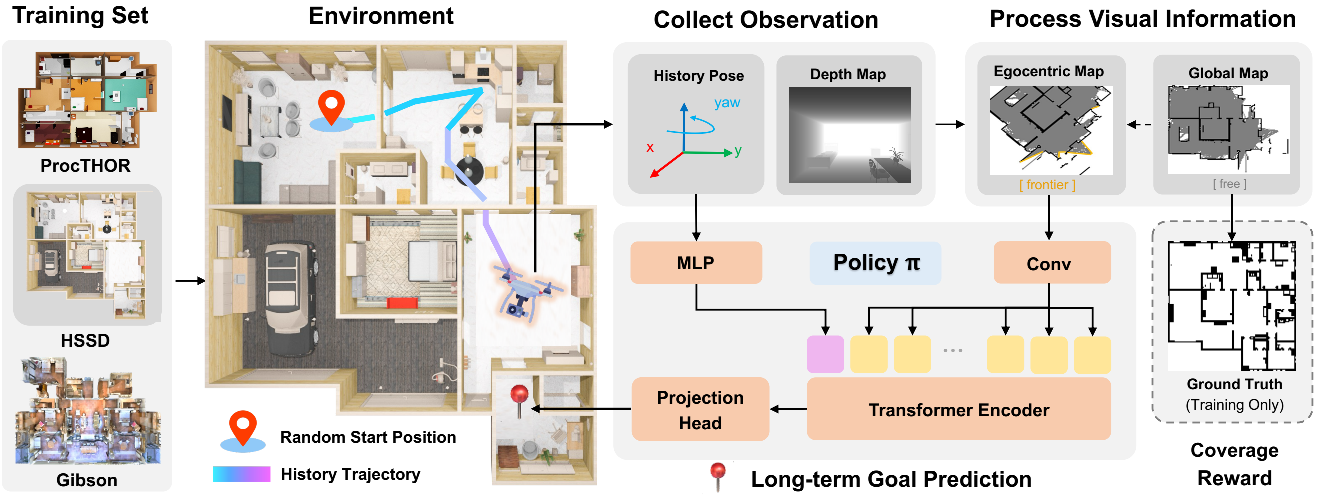

Figure 4. The overview of our framework. Trained on 1,024 diverse indoor scenes, GLEAM processes depth observations and agents' poses to iteratively update a global map. An egocentric map is extracted and augmented with exploration frontiers to capture semantic exploration cues. A lightweight Transformer encoder then analyzes the egocentric map and trajectory history to predict the long-term goals. The reward function of coverage is computed by the global map and ground-truth occupancy map.

fied criteria to build our dataset. For example, we only select enclosed scene meshes with a nearly watertight external surface and high-quality real-scan meshes with minimal floaters and artifacts. However, unlike other datasets providing mesh files directly, we can only access the digital assets of ProcTHOR through the Habitat platform [39] and Unity Engine [20], as they are stored and organized in a proprietary format. To address this and extend these valuable synthetic data usages, we developed an autonomous script that can launch the generation given configuration, i.e., number of rooms, and then batch export the generated scenes from Unity Editor to mesh files. The resulting scene meshes have multiple editable contents, including materials, floorplans, object placement, and controllable connectivity. We believe these large-scale synthetic datasets will benefit many data-driven applications and will release both the export script and the created assets.

3.2. Complex Scenes for Long-Horizon Exploration

Largely derived from 3D datasets tailored for perception or short-range navigation, previous long-horizon exploration benchmarks [42, 47] exhibit limitations such as fragmented surfaces and simple layouts, detailed in Section 2. As a part of GLEAM-Bench, approximately 400 complex indoor scenes aim to address these shortcomings. They are meticulously curated to include large-scale, diverse but architecturally realistic layouts, such as 2-bed-2-bath and multi-family apartments that mirror the complexity of real buildings. We incorporate high-density clutter (e.g., furniture, appliances) and geometrically intricate surfaces to simulate realistic obstacle distributions and contact dynamics.

4. Methodology

4.1. Task Formulation

We formulate the active mapping problem as learning an optimal exploration policy π that controls an agent to capture enough information of unseen target scenes for 3D reconstruction, with limited decision-making budgets and safety constraints. Following [8, 9, 35], we adopt a reinforcement learning (RL)-based framework to model these long-horizon exploration tasks for active mapping.

As shown in Fig. 4, our simulated agent is embodied as CrazyFlie [16], a type of unmanned aerial vehicle equipped with an onboard depth camera and an IMU, to execute data collection for reconstruction. Our agent's observation o_t includes the depth map and its poses at each time step t. After capturing novel observations, we update the maintained egocentric map M_t (Sec. 4.2) as the agent's current state s_t. Based on the map, our RL-based policy predicts the action a_t (Sec. 4.3) including movement and orientation to the navigable long-term goal. To learn a generalizable unified exploration policy, we propose effective training strategies (Sec. 4.4) and optimization settings (Sec. 4.5).

4.2. Semantic Map Representations

As Fig. 4 shows, we maintain two maps during exploration: 1) an egocentric semantic map M_t as the agent's state s_t, and 2) a probabilistic global map G_t in the world coordinate, designed to integrate historical observations. Each cell of the semantic egocentric map M_t is categorized as task-related states among occupied, free, unknown, frontier, enabling environment-agnostic spatial reasoning.

We begin by detailing how to construct and update G_t.

At the initial time step $t = 0$, the depth map D_0 is back-projected into a 3D point cloud in the world coordinate using the camera's intrinsic and extrinsic parameters. The raw point cloud is filtered to retain only key points within a predefined height range. These filtered points are then projected into a temporary binary occupancy map via top-down projection. Inspired by [9], we extend the binary occupancy map to the probabilistic occupancy map G_t that distinguishes the unexplored area and free area to guide the exploration process. To construct the probabilistic map, we adopt Bresenham's Line Algorithm [4] to cast the ray path in this temporary binary map from the agent's position to the endpoints among occupied cells. Following the classical occupancy grid mapping algorithm [9, 44], we have the log-odds formulation of occupancy probability:

$$\log \mathrm{Odd}(m_i|z_j) = \log \mathrm{Odd}(m_i) + C, \tag{1}$$

where m_i is the occupancy probability of i^{th} cell in the map M_t, z_j is the measurement event that j^{th} camera ray passes through this cell, and $C = \log \frac{p(z_j|m_i=1)}{p(z_j|m_i=0)}$ can be regarded as an empirical constant. The derivation can be found in the Appendix. At each step $t + 1$, we update the probabilistic global map G_{t+1} based on the preceding map G_t and newly observed points. Each cell of the global map can be discretized into three categories among {occupied, free, unknown}.

We proceed to elaborate on the construction of the egocentric semantic map M_t. At each time step t, we extract an egocentric map centered at the agent's current position from the global map G_t. To enhance the exploration-oriented semantics, we implement a frontier detection module inspired by [48] to extend the category of frontiers. Specifically, we define a convolutional kernel to recognize boundaries between free and unknown areas. The resulting egocentric map M_t preserves geometric structures while encoding exploration progress, enabling effective decision-making through differentiable spatial reasoning.

After each round of map updating, the global map G_t is compared with the ground-truth occupancy map to compute the coverage ratio reward introduced in Sec. 4.5. The egocentric map M_t and historical poses are encoded by LocoTransformer [50], and taken as the input of our policy network.

4.3. Long-term Action Space

Prior methods suffer from critical limitations in action space design for long-horizon exploration tasks. First, motion primitives (e.g., move forward 10cm) [7, 8, 35, 49] force reliance on a two-stage global-local pipeline, which introduces excessively costly simulation trials, severely degrading training efficiency. Furthermore, such fragmented trajectories inevitably compromise smoothness in real-world deployments. Second, short-term action spaces [14, 15] constrained to immediate neighborhoods demand simultaneous learning of collision avoidance and exploration under conservative behaviors, resulting in myopic policies that fail to reason about long-term environmental structures.

To address these limitations, we propose an action space within the navigable area that allows distant but reachable long-term goals as atomic actions. The action space defines relative movement in the agent's local $SE(2)$ frame, parameterized as vectors (Δx, Δy, $\Delta\theta$). Compared to the local planner that outputs the motion primitives as action, a heuristic A* planner [17] is used to plan a local trajectory to determine whether the goal is reachable. Note that we only capture observations at the long-term goals (i.e., keyframes) during training to avoid excess rounds of simulation. This design disentangles global exploration intent from local navigation, enabling agents to focus on high-level decision-making while offloading low-level path safety to an A*-based verifier. Specifically, for each predicted long-term goal, our framework employs the dynamically updating global map G_t that indicates states among occupied, free, unknown, to verify its connectivity to the agent's current position via lightweight A* planning. Only goals with collision-free and navigable paths are considered safe, ensuring reachability without sacrificing exploration diversity. This innovation bridges the gap between reactive RL policies and deliberative planning, yielding trajectories that are both globally coherent and locally smooth.

4.4. Training Strategy

To enhance generalizability, we propose the following training strategies to diversify the decision process during policy learning, inspired by classic RL-based implementation [10, 26, 45]. Also, we present additional termination conditions in our framework to accelerate policy learning.

Scene Updating Strategy. We leverage diverse training scenes from ProcTHOR dataset [12] and AI2-THOR [24] platform. However, we cannot launch such large numbers of parallel training environments in the simulator due to the limitations of computational efficiency and memory. On the other hand, Isaac Gym doesn't support users in replacing the loaded asset with a novel one. Thus, we cannot directly update the loaded scene using its provided API. In practice, we allocate $N//32$ distinct scenes per environment across 32 parallel Isaac Gym environments due to memory constraints, when training on $N(\geq 512)$ scenes. During training, each environment may activate a scene from its inactive area with probability p to replace the original one that would be moved to the inactive area. This design preserves scene diversity while maintaining memory efficiency.

Random Initialization Strategy. Previous work [7] initializes the agent's position in the center of the scene, which implicitly introduces impractical prior to training, and thus cannot well generalize to unknown environments in the real

Table 3. The generalization results of exploration policies for active mapping on 128 unseen indoor scenes from our GLEAM-Bench, including synthetic datasets (**ProcTHOR**, **HSSD**) and real-scan datasets (**Gibson**, **Matterport3D**). All learnable methods are trained on 1024 scenes. We evaluate 10 episodes per scene and report the average results. Different methods initialize with the shared random poses of agents. "*": ANM is based on per-scene optimized neural representation and thus is directly trained on each testing scene.

Exploration Policy		**Overall**			**ProcTHOR & HSSD**			**Gibson & Matterport3D**		
		Cov. ↑	**AUC** ↑	**CD** ↓	**Cov.** ↑	**AUC** ↑	**CD** ↓	**Cov.** ↑	**AUC** ↑	**CD** ↓
Heuristic	Random	31.41%	27.58%	2.15m	34.88%	30.85%	1.85m	20.89%	24.30%	2.76m
	Vacuum [8]	42.96%	35.89%	1.67m	45.81%	37.56%	1.41m	37.13%	32.48%	2.20m
	FBE [48]	56.80%	45.07%	0.94m	61.56%	50.99%	0.56m	46.17%	37.60%	1.72m
Info	UPEN [15]	49.65%	42.98%	1.38m	54.84%	46.65%	1.01m	39.02%	35.47%	2.14m
	ANM* [49]	57.01%	49.56%	1.02m	63.98%	56.37%	0.66m	42.73%	35.63%	1.77m
RL-based	ANS [7]	48.86%	41.98%	1.39m	54.31%	46.93%	1.01m	37.71%	31.86%	2.17m
	OccAnt [35]	53.61%	46.03%	1.16m	60.30%	51.99%	0.82m	39.91%	33.84%	1.85m
	GLEAM	**66.50%**	**57.63%**	**0.80m**	**76.01%**	**66.13%**	**0.38m**	**47.04%**	**40.23%**	**1.67m**

world. To narrow this sim-to-real gap, we randomly set the initial poses of our agent in any non-collision area during training and evaluation. In particular, we extract the navigatable maps from preprocessed point clouds of scenes, and then slightly narrow the initial areas by max pooling to discard easy-to-collide corners. It turns out that the random initialization strategy significantly enriches the diversity of the decision process during training, and enhances the generalizability during evaluation.

Termination Conditions. Classic termination criteria include the maximum episode lengths and success thresholds. To enhance exploration safety and navigation reliability in diverse environments, we propose three supplementary termination criteria: 1) *Collision detection*, triggering immediate termination upon physical contact with obstacles; 2) *Progress stagnation*, activated when the cumulative coverage reward over last ten steps falls below 1%; and 3) *Goal viability assessment*, terminating episodes where long-term objectives become unnavigable in the current observation space or exceed predefined trajectory length. These enhanced termination mechanisms collectively optimize risk mitigation while maintaining navigation effectiveness through adaptive response to the environment.

4.5. Reward Functions and Optimization

We highlight the optimization objective of our exploration policy lies in the following key aspects: exploration completeness, path efficiency, safety, and navigability. We design the following optimization settings, such as collision penalty, to encourage our exploration policy to predict reliable and safe target poses. The details of implementation can be found in Appendix B.2 and C.1.

The Setup of Reinforcement Learning. Our end-to-end exploration policy is optimized with proximal policy optimization [40] (PPO) for parallelizing sampling. Hence we design the following reward functions to reflect the task objective of exploration for active mapping.

Reward Functions. With the occupancy probability F_t^G at time step t, we can threshold each voxel with an empirical bound to determine if it is occupied. This discrimination process outputs a binary occupancy map with $\tilde{N}_t$ voxels being occupied, which is used to calculate the coverage ratio:

$$\mathrm{CR}_t = \frac{\tilde{N}_t}{N^*} \cdot 100\%, \tag{2}$$

where N^* is the number of ground-truth occupied cells representing the surface of scenes. To encourage our exploration policy to cover as many unseen scene areas as possible, we use the difference of coverage ratio (CR) between two consecutive steps as the main reward function r^{CR}:

$$r_{t+1}^{\mathrm{CR}} = \mathrm{CR}_{t+1} - \mathrm{CR}_t. \tag{3}$$

To mitigate hazardous exploration behaviors, a negative reward is proposed to penalize collisions at long-term target viewpoints. When the predicted goal lies within the observed area and would cause a collision, the agent will remain stationary, re-plan its next goal, but still incur the collision penalty. Re-planning avoids unnecessary risky behavior and improves sample efficiency during training. In addition, the policy learning benefits from a termination reward triggered at 90% coverage ratio, addressing the challenge of sparse successful samples in long-horizon exploration tasks. In particular, our collision reward $r_t^{\mathrm{Col}} = -1$ if a collision occurs at step t, else 0. The termination reward $r_t^{\mathrm{Term}} = +1$ if the episode terminates with the final coverage exceeding 75%, else 0.

5. Experiments

5.1. Experimental Setup

Dataset. We conduct comprehensive training and evaluation of exploration policies on the proposed GLEAM-Bench

Table 4. Ablation studies of the diversity and complexity of training scenes. We train our exploration policies on different dataset sources to show the advances of cross-dataset and complex training data. The **bold** lines indicate the optimal designs, which are proposed by us and adopted in our framework.

# of Scenes	Data Source	Cov. ↑	AUC ↑	CD ↓
96	Gibson	50.32%	43.08%	1.09m
192	ProcTHOR (≥ 10 rooms)	63.23%	53.74%	0.90m
416	ProcTHOR (< 6 rooms)	61.66%	53.07%	0.91m
896	ProcTHOR	65.50%	55.46%	0.83m
928	ProcTHOR, HSSD	66.09%	57.00%	0.81m
1024	ProcTHOR, HSSD, Gibson	**66.50%**	**57.63%**	**0.80m**

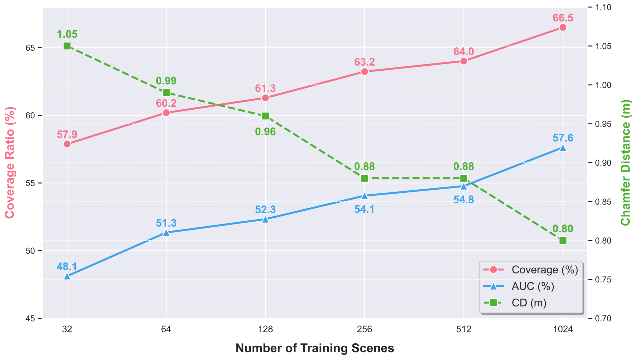

Figure 5. Ablation studies of the number of training scenes. We sample a proportionally reduced number of scenes from the complete 1024 training scenes to demonstrate the significance of the quantity and diversity of training scenes.

benchmark. Specifically, the policy is trained using 1,024 indoor scenes spanning three datasets: two synthetic environments (ProcTHOR-10K [12], HSSD [22]) and one real-scan dataset (Gibson [47]). To rigorously validate cross-domain generalizability, we generalize the unified policy to 128 unseen scenes from the mentioned three datasets and a challenging real-scan dataset, Matterport3D [6], for zero-shot evaluation. The details of data sources, data preprocessing, and dataset split can be found in Appendix A.

Simulation Environment. We conduct all experiments in NVIDIA Isaac Gym [29], a GPU-accelerated physics simulation platform designed for embodied AI.

Evaluation Metrics. We adopt the following metrics to evaluate the exploration completion, trajectory efficiency, and reconstruction accuracy for the active mapping task: (1) *Coverage (%)* quantifies the percentage of the environment successfully mapped by the end of the exploration process. It is calculated as the ratio of the number of explored points to the total number of ground-truth points. (2) *AUC (%)* measures the cumulative coverage ratio over the entire exploration duration, providing insight into both the path efficiency and thoroughness of exploration. (3) *Chamfer Distance (CD, unit: meter)* [43, 55] is the mono-directional Chamfer distance between each ground-truth point and the nearest captured points. The upward arrow symbol (↑) indicates that higher values correspond to superior performance in the evaluated metric, whereas the downward arrow (↓) signifies scenarios where diminished magnitudes are preferable for optimal outcomes.

Table 5. Ablation studies of the scene representations and training strategies. The **bold** lines indicate the optimal designs, which are proposed by us and adopted in our framework.

Settings	Cov. ↑	AUC ↑	CD ↓
Scene Representation			
Binary Occupancy Map	62.92%	53.98%	0.96m
Probabilistic Occupancy Map	64.37%	55.37%	0.83m
Semantic Occupancy Map	**66.50%**	**57.63%**	**0.80m**
Feature Encoder			
UNet [35, 37]	64.01%	54.77%	0.89m
LocoTransformer [50]	**66.50%**	**57.63%**	**0.80m**
Action Space			
Short-term Goal	48.85%	40.24%	1.46m
Long-term Goal	**66.50%**	**57.63%**	**0.80m**
Initialization Strategy			
Fixedly Initialize	56.05%	47.95%	0.92m
Randomly Initialize	**66.50%**	**57.63%**	**0.80m**
Scene Updating Strategy			
Updating Probability $p = 0.05$	63.17%	53.90%	0.86m
Updating Probability $p = 1$	**66.50%**	**57.63%**	**0.80m**

Implementation Details. All experiments are conducted based on legged gym [**?**] in NVIDIA Isaac Gym [29], using CrazyFlie [16] as our agent equipped with an onboard depth camera. The depth maps are rendered at a resolution of 256 × 256 with a 90°field of view (FOV). The resolution of our egocentric semantic map M_t and probabilistic global map G_t is both 128 × 128. The cell size of M_t is $10cm$ × $10cm$. The evaluation is under the keyframe budget of $T = 50$. The ground-truth point clouds on the surfaces are generated using the Poisson Disk sampling method [52] through the Open3D API [54]. The exploration policy is optimized through $5k$ iterations and uses approximately 96 hours of training time on a single GeForce RTX 4090 GPU. All networks are randomly initialized. All networks are frozen during evaluation. Please refer to the Appendix B for further results and details.

5.2. Performance Comparison

To conduct a comprehensive evaluation of GLEAM's effectiveness and generalizability, we systematically assess its performance across both synthetic and real-world test scenarios, as detailed in Table 3. We implement and evaluate these works in our benchmark to demonstrate the superiority of our proposed method. The key details include: 1) **Random Policy** randomly samples actions from a Gaussian distribution within the action space. 2) **Vacuum Policy** simulates a heuristic exploration policy for robot vacuums.

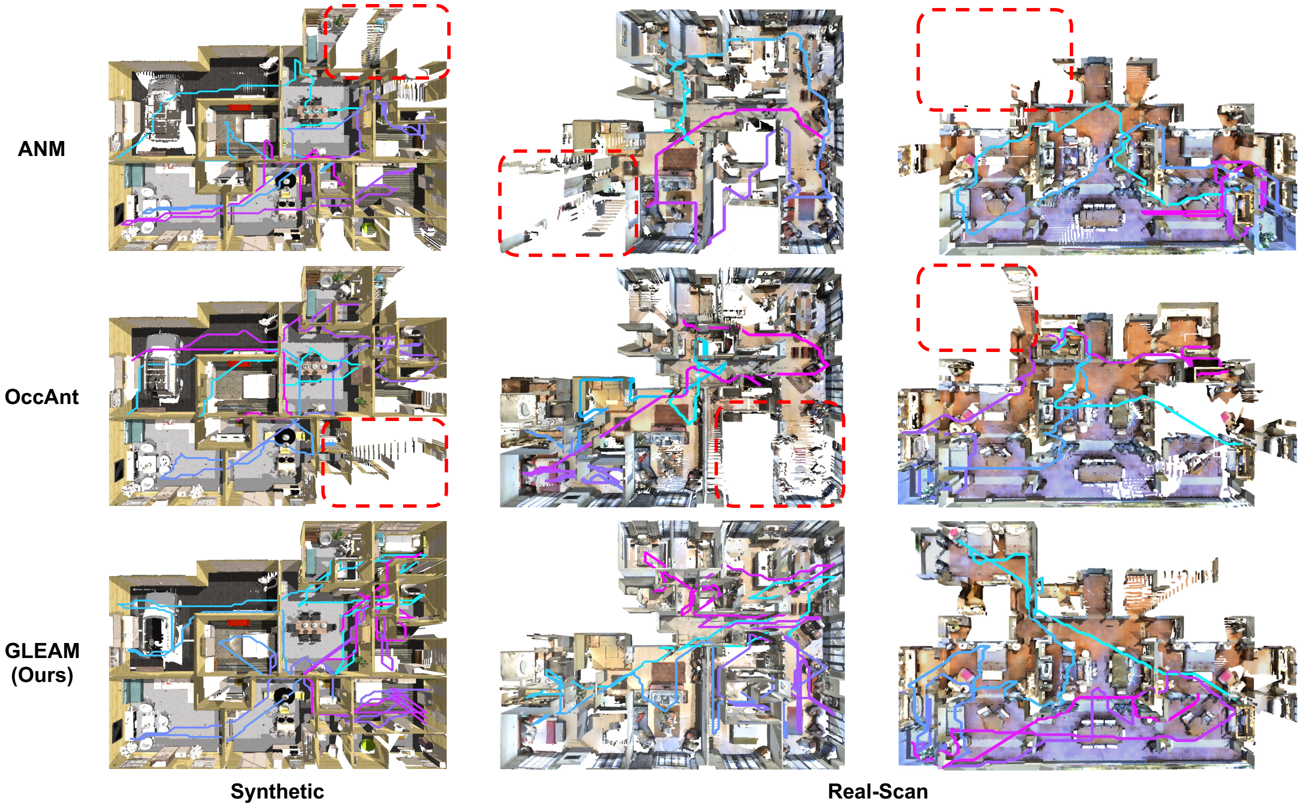

Figure 6. The visualization results of ANM, OccAnt, and GLEAM on three unseen complex indoor scenes from the test set of GLEAM-Bench. The methods share the same random initial poses for each scene.

We follow [8] to let the policy move straight when safe and execute a random number of 9°turns when a collision occurs. 3) **FBE** always moves the agent towards the navigable nearest boundary between observed and unknown areas. 4) **UPEN** [15] estimates the information gain of candidate trajectories sampled by RRT [1], where the gain is estimated by model ensembles. We reduce the number of ensembling models and the number of layers due to the limited memory. 5) We replace **ANM** [49]'s learned RL-based local planner with our A-star planner. 6) **ANS** [7] and **OccAnt** [35] are adapted to active mapping systems, given ground-truth poses. More details can be found in Appendix B.

The experimental results in Table 3 reveal GLEAM's superior generalizability across all evaluation metrics (Table 3). Our method achieves more than 9.49% overall coverage improvement over existing baselines. In particular, our policy attains 76.01% coverage in unseen synthetic indoor scenes that have more than five rooms on average. Notably, even for extremely challenging and cross-dataset real-scan scenes, our policy maintains around 50% final coverage.

To evaluate the deployability in real-world scenarios, we evaluate the exploration trajectory (unit: meter) and safety in Table 6. It shows that GLEAM effectively explore unseen obstacle-dense scenarios while ensuring exploration safety.

Table 6. The average number of keyframes during the generalization evaluation on 128 unseen indoor scenes from GLEAM-Bench.

Policy	**Random**	**FBE**	**UPEN**	**ANM**	**OccAnt**	**GLEAM**
Coverage	31.41%	56.80%	49.65%	57.01%	53.61%	66.50%
#Keyframes	37.01	26.21	24.84	28.31	18.58	29.57
Traj. Length	9.56m	52.34m	30.45m	46.51m	38.08m	54.51m

5.3. Ablation Study

The Effect of Training Scenes. As shown in Table 4 and Fig. 5, we demonstrate that training scene quantity, diversity, and complexity jointly enhance the generalization of exploration policy for active mapping. The policy trained on 96 Gibson scenes underperforms across all metrics, achieving only 50.32% coverage. An inspiring result is that the policy trained on fewer but more complex scenes (with $\geq$ 10 rooms) matches the performance of those using twice as many simple scenes, emphasizing the critical role of scene complexity. Compared to the ProcTHOR-only baseline, our best policy integrating multi-domain datasets improves AUC and coverage by 2.17% and 1.00%, respectively, highlighting cross-dataset heterogeneity as critical for generalizability.

Scaling training scenes from 32 to 1,024 monotonically improves coverage (+8.6%), AUC (+9.5%), and completion (-0.25m). This trend reveals that larger datasets force models to learn generalized exploration strategies by exposing them to rare spatial patterns (e.g., narrow corridors and multi-room connections). In summary, the shown results establish that sufficient, diverse, and complex training scenes are essential for robust active mapping in unseen environments, offering practical guidelines for learning generalizable robotic systems for active mapping.

The Effect of Scene Representations and Encoders. We ablate the scene representations in Table 5. "Binary Occupancy Map" (occupied, others) and "Probabilistic Occupancy Map" (occupied, free, unknown) denote the egocentric maps extracted from the corresponding intermediate representations introduced in Sec. 4.2. Our semantic occupancy map that extends binary occupancy with navigability and frontier outperforms classic binary occupancy maps by 3.65% AUC and 3.58% coverage. This demonstrates that incorporating semantic task-specific information (e.g., frontier categories) enables more informed exploration decisions compared to geometric-only representations. For feature encoding, the lightweight LocoTransformer encoder substantially surpasses conventional UNet architectures, leveraging cross-layer attention to capture long-range spatial dependencies.

The Effect of Long-Term Action Space. Our experiments confirm that long-term goal planning achieves superior performance over short-horizon counterparts (see Table 5). This validates two critical advantages of our design: 1) The integration of A*-verified path connectivity inherently resolves the safety-reachability dilemma faced by traditional two-stage planners, with effective exploration (+17.39% AUC); 2) Decoupling global exploration decisions from local obstacle avoidance preserves action space diversity, enabling the policy to discover non-myopic trajectories that cover more unknown regions (+17.65%). Crucially, these gains do not sacrifice training efficiency.

The Effect of Training Strategies. We ablate the key training strategies, including random initialization and scene updating, in Table 5. Strategic training configurations prove vital for balancing exploration efficiency and robustness. Notably, the random initialization strategy yields a 9.68% AUC and 10.45% coverage improvement over fixed initialization, highlighting that diverse starting conditions prevent overfitting to specific layouts. We define p as the probability of replacing the original active training scenes with another training scene after each episode. The frequent update of the scene ($p = 1$) surpasses the occasional update ($p = 0.05$) with coverage of 3.33% and AUC of 3.73%. These results collectively establish that goal horizon, initialization diversity, and update frequency must be jointly optimized to achieve robust generalization.

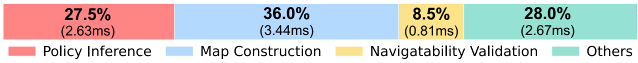

Figure 7. The one-step inference time of our key components.

5.4. Qualitative Results

We visualize the mapping results and the predicted scanning trajectories of ANM, OccAnt, and GLEAM within a single episode in Fig. 6. It demonstrates that our GLEAM generalizes well, even in complex indoor scenes with ≥ 10 rooms and a large number of obstacles. More visualization results can be found on our project website.

5.5. Discussion and Future Directions

Realistic settings & Real-world deployment. 1) *Noisy observation.* Real-world deployments inevitably confront imperfect sensor inputs such as camera noise and depth ambiguity. Our framework employs probabilistic occupancy maps to mitigate noisy raw inputs of sensors, yet persistent noise patterns still propagate geometric errors during active mapping. 2) *Pose estimation.* While our system obtains accurate poses through the simulator, the pose estimation in practical scenarios with fast camera motions or textureless regions induces pose drift. This spatial uncertainty manifests as misaligned geometry fragments, particularly when scanning thin structures like chair legs or lamp arms. 3) *Open environments.* Unlike bounded scanning domains in simulation, real-world scenes often contain dynamically expanding areas (e.g., newly opened doors). Existing frameworks struggle to build memory-efficient representations for unbounded and dynamic scenes, which shows that the sim-to-real gap still has a lot of potential to be explored.

While real-world deployment remains an open challenge, we advance the sim-to-real validation across *sensor-noise tolerance* and *computation cost* to demonstrate concrete progress toward deployability. To evaluate the sensor-noise tolerance of GLEAM, we inject hardware-aligned Gaussian noise to observations during inference, deriving

Table 7. The robustness of GLEAM under Gaussian noise $N(0, \sigma^2)$ (unit: meter) during inference. $\boldsymbol{\sigma_P^2}$: variance of cumulative pose noise. $\boldsymbol{\sigma_D^2}$: variance of depth noise. **KF**: keyframes. **TL**: trajectory length. **"*"**: non-cumulative per-step noise

Settings	σ_P^2	σ_D^2	Cov.	AUC	CD	KF	TL
GLEAM (no noise)			66.50%	57.63%	0.80m	29.57	54.51m
Pose-only	0.1	0	63.27%	54.41%	0.86m	28.27	41.63m
	0.3	0	60.73%	52.11%	0.92m	24.95	35.71m
	0.5	0	55.59%	48.24%	0.99m	20.31	22.79m
	0.1*	0	66.18%	56.62%	0.76m	30.69	49.36m
	0.5*	0	58.54%	50.53%	0.95m	23.61	38.51m
Depth-only	0	0.05	64.94%	55.31%	0.82m	30.44	45.05m
	0	0.1	60.21%	51.09%	0.96m	29.21	36.78m
	0	0.2	54.77%	46.30%	1.13m	27.51	27.21m
Depth+Pose	0.1	0.05	60.44%	51.76%	0.91m	28.20	36.62m
	0.3	0.1	54.03%	47.21%	1.09m	24.74	24.33m
	0.5	0.1	51.36%	44.32%	1.17m	20.94	20.56m

from real-world sensors like Intel®RealSense D455 depth camera ($< 2\%$ error at $4m$) and TDK InvenSense ICM-42688-P IMU. As shown in Table 7, GLEAM maintains strong robustness despite training on ideal observations, which stems from our probabilistic map that inherently suppresses transient noise by Bayesian updating.

Our system achieves real-time inference (**104.7Hz**) on a PC with an RTX 3090 GPU, with latency analysis in Figure 7 demonstrating the efficiency of our lightweight policy network and CUDA-accelerated map updating/A* planning, ensuring seamless high-frequency perception and decision-making in the real world.

Challenging 3D benchmark & 3D action space. We've explored the potential of our framework for challenging 3D benchmark, including 3D action space $(x, y, z, pitch, yaw)$ and 3D optimization objectives. In this setting, our agent is encouraged to capture all details, such as the surface under the underside of a table, in complex 3D scenes. While our agent demonstrates promising effectiveness and generalizability in scanning most of unobstructed surfaces, we found it quite difficult to capture the geometrically complex surfaces (e.g., the undersides of tables and self-occlusion surfaces of decorations) in cluttered 3D environments. Also, the heavy computational burden and limited memory prevent us from optimizing the components like 3D representations and scaling the number of training scenes.

Challenging multi-floor complex scenes. Although we've been exploring active mapping for complex single-floor indoor scenes, the tough cases in the real world are the multi-floor indoor scenes. These environments introduce unique cross-floor topological dependencies and vertical navigation constraints that existing frameworks fail to adequately model. Moreover, the inherent geometric discontinuities between floors exacerbate memory fragmentation when using conventional spatial representations, leading to increasing memory overhead.

Multi-agent collaboration. Multi-agent collaboration is one of the solutions for complex scenarios such as multi-floor scenes. However, scaling to collaborative active mapping introduces novel fundamental challenges in distributed strategy optimization, dynamic role allocation, and communications. These challenges demand the rethinking of existing frameworks, particularly in developing scalable and memory-efficient representations and decentralized decision-making architectures.

6. Conclusion

This work presents GLEAM-Bench, the first large-scale benchmark with 1,152 diverse 3D scenes to enable scalable training and reliable evaluation. Furthermore, we propose GLEAM, an RL-based generalizable exploration policy for active mapping in complex unknown environments. It significantly outperforms previous methods, achieving 66.50% coverage (+9.49%) with efficient trajectories, and improved mapping accuracy on 128 unseen complex scenes. We show the effectiveness of our proposed modules, including semantic map representations and the randomized initialization strategy.

Acknowledgements

This work is supported in part by the Centre for Perceptual and Interactive Intelligence (CPII) Ltd., a CUHK-led InnoCentre under the InnoHK initiative of the Innovation and Technology Commission of the Hong Kong Special Administrative Region Government, the National Key R&D Program of China (2022ZD0160201), and Shanghai Artificial Intelligence Laboratory. We thank all anonymous reviewers for their constructive feedback and valuable suggestions. We also extend our sincere gratitude to Yaokun Li, Chenjian Gao, and Ruikang Li for their advice on visualization, as well as to Gang Chen for his insightful guidance regarding drone control.

References

[1] Rapidly-exploring random trees: A new tool for path planning. *Research Report 9811*, 1998. 8, 16

[2] Ana Batinovic, Tamara Petrovic, Antun Ivanovic, Frano Petric, and Stjepan Bogdan. A multi-resolution frontier-based planner for autonomous 3d exploration. *IEEE Robotics and Automation Letters*, 6(3):4528–4535, 2021. 3

[3] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. Receding horizon" next-best-view" planner for 3d exploration. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1462–1468. IEEE, 2016. 3

[4] Jack E Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems journal*, 4(1):25–30, 1965. 5, 15

[5] Yuhong Cao, Rui Zhao, Yizhuo Wang, Bairan Xiang, and Guillaume Sartoretti. Deep reinforcement learning-based large-scale robot exploration. *IEEE Robotics and Automation Letters*, 2024. 2, 3

[6] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017. 1, 2, 3, 7, 14

[7] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations*, 2020. 1, 2, 3, 5, 6, 8, 16

[8] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *International Conference on Learning Representations*, 2019. 1, 2, 3, 4, 5, 6, 8, 16, 17

[9] Xiao Chen, Quanyi Li, Tai Wang, Tianfan Xue, and Jiangmiao Pang. Gennbv: Generalizable next-best-view policy for active 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16436–16445, 2024. 4, 5, 15, 17

[10] Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pages 2048–2056. PMLR, 2020. 5

[11] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 3, 14

[12] Matt Deitke, Eli VanderBilt, Alvaro Herrasti, Luca Weihs, Kiana Ehsani, Jordi Salvador, Winson Han, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35:5982–5994, 2022. 2, 3, 5, 7, 14

[13] Christian Dornhege and Alexander Kleiner. A frontier-void-based approach for autonomous exploration in 3d. *Advanced Robotics*, 27(6):459–468, 2013. 3

[14] Ziyue Feng, Huangying Zhan, Zheng Chen, Qingan Yan, Xiangyu Xu, Changjiang Cai, Bing Li, Qilun Zhu, and Yi Xu. Naruto: Neural active reconstruction from uncertain target observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21572–21583, 2024. 2, 5

[15] Georgios Georgakis, Bernadette Bucher, Anton Arapin, Karl Schmeckpeper, Nikolai Matni, and Kostas Daniilidis. Uncertainty-driven planner for exploration and navigation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 11295–11302. IEEE, 2022. 1, 2, 3, 5, 6, 8, 16

[16] Wojciech Giernacki, Mateusz Skwierczyński, Wojciech Witwicki, Paweł Wroński, and Piotr Kozierski. Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering. In *2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR)*, pages 37–42. IEEE, 2017. 4, 7

[17] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4 (2):100–107, 1968. 5, 16

[18] Stefan Isler, Reza Sabzevari, Jeffrey Delmerico, and Davide Scaramuzza. An information gain formulation for active volumetric 3d reconstruction. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3477–3484. IEEE, 2016. 3

[19] Wen Jiang, Boshu Lei, and Kostas Daniilidis. Fisherrf: Active view selection and uncertainty quantification for radiance fields using fisher information. *arXiv preprint arXiv:2311.17874*, 2023. 3

[20] Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*, 2018. 4, 14

[21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1

[22] Mukul Khanna, Yongsen Mao, Hanxiao Jiang, Sanjay Haresh, Brennan Shacklett, Dhruv Batra, Alexander Clegg, Eric Undersander, Angel X Chang, and Manolis Savva. Habitat synthetic scenes dataset (hssd-200): An analysis of 3d scene scale and realism tradeoffs for objectgoal navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16384–16393, 2024. 2, 3, 7, 14

[23] Andreas Klöckner, Nicolas Pinto, Yunsup Lee, Bryan Catanzaro, Paul Ivanov, and Ahmed Fasih. Pycuda and pyopencl: A scripting-based approach to gpu run-time code generation. *Parallel Computing*, 38(3):157–174, 2012. 15

[24] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017. 5, 14

[25] Soomin Lee, Le Chen, Jiahao Wang, Alexander Liniger, Suryansh Kumar, and Fisher Yu. Uncertainty guided policy for active robotic 3d reconstruction using neural radiance fields. *IEEE Robotics and Automation Letters*, 7(4):12070–12077, 2022. 3

[26] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE transactions on pattern analysis and machine intelligence*, 45(3):3461–3475, 2022. 5

[27] Shiyao Li, Antoine Guédon, Clémentin Boittiaux, Shizhe Chen, and Vincent Lepetit. Nextbestpath: Efficient 3d mapping of unseen environments. *arXiv preprint arXiv:2502.05378*, 2025. 2, 3

[28] Iker Lluvia, Elena Lazkano, and Ander Ansuategi. Active mapping and robot exploration: A survey. *Sensors*, 21(7): 2445, 2021. 2

[29] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 3, 7

[30] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 405–421. Springer, 2020. 1, 3

[31] Joseph Ortiz, Alexander Clegg, Jing Dong, Edgar Sucar, David Novotny, Michael Zollhoefer, and Mustafa Mukadam. isdf: Real-time neural signed distance fields for robot perception. *Robotics: Science and Systems*, 2022. 1

[32] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 3

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 16

[34] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *The Journal of Machine Learning Research*, 22(1): 12348–12355, 2021. 16

[35] Santhosh K Ramakrishnan, Ziad Al-Halah, and Kristen Grauman. Occupancy anticipation for efficient exploration and navigation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 400–418. Springer, 2020. 1, 2, 3, 4, 5, 6, 7, 8, 17

[36] Yunlong Ran, Jing Zeng, Shibo He, Jiming Chen, Lincheng Li, Yingfeng Chen, Gimhee Lee, and Qi Ye. Neurar: Neural uncertainty for autonomous 3d reconstruction with implicit neural representations. *IEEE Robotics and Automation Letters*, 2023. 3

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 7

[38] Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In *5th Annual Conference on Robot Learning*, 2021. 16

[39] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019. 4, 14

[40] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 6

[41] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1746–1754, 2017. 2

[42] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 2, 3, 4, 14

[43] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021. 1, 7

[44] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002. 5, 15

[45] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017. 5

[46] Tai Wang, Xiaohan Mao, Chenming Zhu, Runsen Xu, Ruiyuan Lyu, Peisen Li, Xiao Chen, Wenwei Zhang, Kai Chen, Tianfan Xue, et al. Embodiedscan: A holistic multi-modal 3d perception suite towards embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19757–19767, 2024. 3

[47] Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9068–9079, 2018. 2, 3, 4, 7, 14

[48] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, pages 146–151, 1997. 3, 5, 6

[49] Zike Yan, Haoxiang Yang, and Hongbin Zha. Active neural mapping. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10981–10992, 2023. 1, 2, 3, 5, 6, 8, 16

[50] Ruihan Yang, Minghao Zhang, Nicklas Hansen, Huazhe Xu, and Xiaolong Wang. Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers. *arXiv preprint arXiv:2107.03996*, 2021. 2, 5, 7

[51] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023. 3

[52] Cem Yuksel. Sample elimination for generating poisson disk sample sets. *Computer Graphics Forum*, 34(2):25–32, 2015. 7, 15

[53] Huangying Zhan, Jiyang Zheng, Yi Xu, Ian Reid, and Hamid Rezatofighi. Activermap: Radiance field for active mapping and planning. *arXiv preprint arXiv:2211.12656*, 2022. 3

[54] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. 7, 15

[55] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. 7

GLEAM: Learning Generalizable Exploration Policy for Active Mapping in Complex 3D Indoor Scenes

Supplementary Material

A. Datasets

A.1. Data Creation & Data Filtering

There are several well-known datasets of indoor scenes, including ScanNet [11], Replica [42], Gibson [47], and Matterport3D (MP3D) [6]. However, the embodied AI community faces several challenges when working with these datasets: 1) there is a limited number of high-quality real-scan datasets, where "high-quality" refers to watertight surfaces, well-organized layouts, and unified reconstruction quality; 2) synthetic scenes often lack realistic features such as collision-rich layouts and high-fidelity furniture assets; 3) public datasets of indoor scenes employ different organizational structures, making it difficult to collect scene meshes with unified formats and scales, which impedes their effective use in simulation and policy learning.

In this work, we collect and preprocess 1152 high-quality meshes of complex indoor scenes from ProcTHOR-10K [12], HSSD [22], Gibson, and MP3D to learn a generalizable exploration policy for active 3D mapping. To leverage the digital assets in the general format originally coupling in AI2THOR [24] and Unity [20] backend, we created 972 meshes of complex scenes from ProcTHOR-10K using our export script. After manually filtering out the high-quality meshes of complex indoor scenes from these datasets, we preprocess them into a unified format and scale. Finally, these meshes are split into 1024 training scenes and 128 test scenes in our benchmark. The details of the data source are shown in Table 8. The details of preprocessing each dataset are as follows:

ProcTHOR-10K. ProcTHOR [12] and AI2-THOR [24] have empowered the research community to procedurally generate fully interactive, high-fidelity indoor scenes with diverse layouts for robotic training at scale. They introduce the ProcTHOR-10K dataset as templates of generated layouts, which includes 10,000 diverse indoor scenes. However, the original AI2-THOR platform and ProcTHOR dataset are limited by their reliance on the Habitat platform [39] and Unity Engine [20] for asset simulation and management, which constrains the extensibility of these valuable digital assets. To address this limitation, we developed an autonomous script that batch exports the generated scenes from Unity Editor to mesh files. This approach enables the procedural generation of scene meshes with editable content, including materials, floorplans, object placement, and controllable connectivity. Notably, users can generate an arbitrary number of scenes and then export them into mesh files using our exporting script. These infinitely generated 3D assets can be utilized for both policy training and digital content creation for AR/VR. We will release both the export script and the created assets.

Habitat Synthetic Scenes Dataset (HSSD). The HSSD dataset comprises 211 meticulously crafted 3D environments specifically designed to facilitate generalization ca-

Table 8. The sources of our collected and processed scene meshes. We created the meshes of complex scenes from ProcTHOR-10K using our export script. After manually filtering out the meshes of complex indoor scenes from **ProcTHOR-10K**, **HSSD**, **Gibson**, and **MP3D**, we preprocess them and split them into **1024 training scenes** and **128 test scenes**.

Dataset	Total Amount	Type / Mode	Amount	Training	Test
ProcTHOR-10K	896 (train) 76 (test)	4-room	284	256	28
		5-room	164	164	0
		2-bed-2-bath	280	256	24
		7-room-3-bed	114	96	18
		8-room-3-bed	28	28	0
		12-room	64	64	0
		12-room-3-bed	38	32	6
HSSD	32 (train), 10 (test)	easy	10	32	10
		medium	8		
		hard	24		
Gibson	96 (train), 24 (test)	real-scan	120	96	24
MP3D	18 (test)	real-scan	18	0	18
Total (Ours)	**1152**	**mixed**	**1152**	**1024**	**128**

pabilities within realistic 3D environments. This collection is characterized by its professionally curated digital assets and intricate spatial arrangements. We have selected 42 exemplary indoor scenes from this dataset, which serve as valuable photorealistic synthetic sources for exploration tasks. While decorative elements improve scene realism, they are unnecessary for policy learning and introduce excessive computational costs that hinder large-scale simulation training. Consequently, the scenes underwent systematic preprocessing through geometric simplification, particularly focusing on decorative elements and doors that might impede cross-room navigability.

Gibson & Matterport3D. Gibson and Matterport3D are public real-scan datasets providing hundreds of complex indoor scenes. However, the styles of these scene meshes exhibit significant variation, and the mesh quality is too inconsistent for direct use. Therefore, we filter these two datasets by the following criteria: 1) accurate reconstruction with minimal floaters and artifacts, 2) enclosed scene mesh with a nearly watertight external surface, and 3) one-floor structure. As a result, we obtain 120 diverse high-quality scene meshes from Gibson for training and evaluation. Also, we split all 18 selected meshes from MP3D for cross-dataset and out-of-domain evaluation.

A.2. Data Preprocessing

Mesh Preprocessing. To standardize the coordinate systems across scene meshes from different datasets, we transform all meshes such that their origin points lie at the geometric center of the floors, with the height direction isotropic to the +Z-axis. The transformation scripts are implemented in Python using Open3D library [54]

Ground-Truth Point Cloud. We generate ground-truth point clouds using the Poisson Disk sampling method [52], implemented in Open3D [54], to sample 100,000 points from the 3D scene meshes. To simplify visibility determination, we voxelize these point clouds at a specified resolution (grid size = 128 in this work) and filter out obviously invisible points, such as internal points enclosed within surfaces. These voxelized points serve as the ground-truth point clouds for the meshes and are used to compute key metrics like coverage ratio.

A.3. Dataset Split & Training Stages

Due to memory constraints and computational efficiency, we distributed the 1,024 training scenarios across two sequential training stages (i.e., stage 1 & stage 2). The final checkpoint from the first training stage served as parameter initialization for the subsequent stage. The one-stage exploration policy is optimized through 2.5k iterations and uses approximately 48 hours of training time on a single GeForce RTX 4090 GPU.

The dataset split of the two training stages is as follows. **Stage 1**: “procthor-4-room (256)”, “procthor-5-room (164)”, “procthor-8-room-3-bed (28)”, “procthor-12-room-3-bed (32)”, “hssd (32)”. **Stage 2**: “procthor-2-bed-2-bath (256)”, “procthor-7-room-3-bed (96)”, “procthor-12-room (64)”, “gibson (96)”.

B. Implementation Details

B.1. Occupancy Grid Mapping Algorithm

The goal of an occupancy mapping algorithm [44] is to estimate the posterior probability of occupancy over voxels given the current probabilistic grid and the novel measurement event of camera ray casting. In particular, the more frequently a voxel is passed through by camera rays, the more confident the agent regards it as navigable free space.

PyCUDA-based Bresenham’s Line Algorithm. Before updating the probabilistic occupancy grid, Bresenham’s line algorithm [4] is implemented to cast the ray path in 3D space between the camera viewpoint and the endpoints among the point cloud back-projected from captured depth maps. To accelerate the computing efficiency, we use PyCUDA [23] to implement Bresenham’s line algorithm.

Derivation of Map Updating. In practice, we adhere to the algorithm implementation outlined in GenNBV [9]. A comprehensive explanation of the methodology, along with the experimental results, is provided in the appendix of GenNBV. We provide the key derivation of the log-odds formulation of occupancy probability as follows:

Before updating the probabilistic occupancy map G_t, Bresenham’s line algorithm is implemented to cast the ray path in 3D space between the camera viewpoint and the endpoints among the point cloud back-projected from D_{t+1}. According to the classical occupancy grid mapping algorithm [44], we have the log-odds formulation of occupancy probability:

$$\log Odd(m_i|z_j) = \log Odd(m_i) + \log \frac{p(z_j|m_i=1)}{p(z_j|m_i=0)}, \quad (4)$$

where m_i denotes the occupancy probability of i^{th} voxel in the map G_t, z_j is the measurement event that j^{th} camera ray passes through this voxel.

For the item $C = \log \frac{p(z_j|m_i=1)}{p(z_j|m_i=0)}$, there are only two cases for the measurement event in fact: $z_j = 0$ or $z_j = 1$. Thus, if the measurement event z_j (i.e., the voxel is passed through by the j^{th} camera ray) happens, we’ll update the occupancy by adding the value of $C_1 = \log \frac{p(z_j=1|m_i=1)}{p(z_j=1|m_i=0)}$. If it’s not passed, we’ll add the value $C_2 = \log \frac{p(z_j=0|m_i=1)}{p(z_j=0|m_i=0)}$. The values of C_1 and C_2 can be set as empirical constants, depending on factors such as the accuracy of ray casting and the confidence of each ray. Actually, $C' = |\frac{C_1}{C_2}|$. We set a high value for C' (i.e., high confidence) because our experiments are based on the realistic simulator and accurate observations like depth maps.

Table 9. The key hyperparameters for our policy learning.

Term	Value
Optimizer	Adam
Optimization batch size	128
Learning rate	0.0001
Training Iterations	2500
Training Environments	32
N steps	512
N epochs	4
Buffer size	30
Value coefficient	0.8
Entropy coefficient	0.01
Discount factor γ	0.99
GAE τ	0.99
PPO clipping	0.2

Therefore, we can update the occupancy status of each voxel in the map G_t by adding a constant for each ray casting process. Note that the probabilistic occupancy map F^G is continuously updated within an episode. Finally, the occupancy status of voxels can be classified into three categories: unknown, occupied, and free, by setting an empirical threshold.

B.2. A* Path-Finding Algorithm

To evaluate the navigability between the agent's current position and predicted 3D target position, we implement a classic A* path-finding algorithm [17] in 3D space. We developed a CUDA-based implementation of the algorithm, increasing the computational efficiency. The system classifies a target position as unnavigable if the computed path length exceeds a predefined threshold, ensuring that our NBV policy predicts reliable and safe target poses.

Most previous works regard path-finding algorithms as a local policy and define a few movement commands (e.g., move forward $10cm$, turn left 30°) as their action space. However, we don't follow this paradigm in our work for the following main reasons: 1) The key challenge of exploration policy is to determine the next best viewpoint, instead of the next neighbor step. Classic and learning-based planning and control methods both are capable of handling the control process toward the target viewpoint. 2) Popular action space, which consists of move forward $10cm$, turn left 30°, turn right 30°, makes redundant waypoints that produce inefficient trajectory, non-smooth control, and costly frequency of map updating, planning, and control.

B.3. Key Hyperparameters and Details

The key hyperparameters of our policy learning are shown in Table 9. Our implementation builds upon the codebase of Legged Gym [38] and utilizes the PPO implementation from Stable-Baselines3 [34], which is developed in PyTorch [33].

PPO Implementation. Specifically, given our parameterized policy π_θ, the objective of PPO is to maximize the following function:

$$L(\theta) = \mathbb{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} A^{\pi_{\theta_{\text{old}}}}(s_t, a_t) \right], \tag{5}$$

where $A^{\pi_{\theta_{\text{old}}}}(s_t, a_t)$ is the advantage function that measures the value of taking action a_t at state s_t under the current policy $\pi_{\theta_{\text{old}}}$. To prevent significant deviation of the new policy from the old policy, PPO incorporates a clipped surrogate objective function:

$$\begin{aligned} L^{\text{CLIP}}(\theta) =& \mathbb{E}_t[\min(\eta_t(\theta) A^{\pi_{\theta_{\text{old}}}}(s_t, a_t), \\ & \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon) A^{\pi_{\theta_{\text{old}}}}(s_t, a_t))], \end{aligned} \tag{6}$$

where $\eta_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ and ϵ is a hyper-parameter that controls the size of the trust region.

Onboard Cameras. We assume that upon reaching the target pose, the agent performs four sequential 90-degree rotations and captures an observation at each orientation. To mitigate the significant computational overhead associated with repeated rendering during rotation, we implemented a simulation of four cameras mounted on the agent, with their headings oriented at 90-degree intervals.

Keyframe Budget During Inference. The budget $T = 50$ during inference was set based on the average exploration keyframes $\overline{T} = 31.78$ across methods. This value balances policy completeness and computational efficiency while not compromising the generalizability.

B.4. Implementation of Baseline Methods

We implement and evaluate the following works in our benchmark to demonstrate the superiority of our proposed method: 1) **Random Policy** randomly samples actions from Gaussian distribution within the action space. 2) **Vacuum** simulates a heuristic exploration policy for robot vacuums. We follow [8] to let policy move straight when safe and execute a random number of 9°turns when a collision occurs. 3) **FBE** always moves the agent towards the navigable nearest boundary between observed and unknown areas. 4) **UPEN [15]** estimates the information gain of candidate trajectories sampled by RRT [1], where the gain is estimated by model ensembles. We reduce the number of ensembling models and the number of layers due to the limited memory. 5) **ANM [49]** learns exploration in a neural implicit representation optimization framework. It estimates the information gain of candidate poses by three empirical criteria. We replace its RL-based local planner with our A-star planner. 6) **ANS [7]**: The original implementation of this policy relies on a global normalized map with unified resolution as input instead of an egocentric observed map, thus it cannot be directly generalized to unknown environments. We adapt this policy to a generalizable pipeline that takes a global egocentric map as input and also augment the policy learning with our random initialization strategy to make it generalizable. Given ground-truth poses, we adapt the original active SLAM system to an active mapping system. 7) **OccAnt [35]**: Similar to ANS, we also provide ground-truth poses to adapt it to an active mapping system. Due to the limited storage, we reduced its map resolution and consequently increased the voxel size to ensure a similar perceptual range.

B.5. Visualization Implementation

All trajectories in Figure 1, 2, 6 are actual results. We record waypoints/keyframes and reconstruct the active mapping process using Open3D for offline visualization.

C. Training Strategy

C.1. Scene Updating Strategy

To enhance the generalizability, we create a training set including 512 diverse indoor scenes in each training stage from our GLEAM-Bench. However, we cannot launch such a large number of parallel training environments in simulation due to the limitations of computational efficiency and memory. As introduced in Sec. 4.4, we adopt a workaround to update the active scene in the limited training environments. We launch 32 training environments in Isaac Gym, and load 16 different scenes as a sampling set for each environment. During training, there is a predefined probability of p to randomly activate a scene in each environment's inactive sampling set. In particular, we move the replaced scenes to the inactive area (i.e., out of the agents' movement space) in the simulator and move the sampled scenes to the active areas of corresponding environments. As shown in Table 5, we found that frequently updating the active scenes utilizes the diversity of training scenes, and improves the generalizability of policies.

C.2. Capturing at Long-Term Target Positions

Previous work [8] typically employs discrete single-step actions, such as moving forward $10cm$ or turning left 10. However, this single-step planning and control approach is inconsistent with real-world robotic systems and significantly increases the computational cost of simulation for RL-based policy training. Moreover, real-world inference of this setup requires numerous policy network iterations, making it prohibitively time-consuming. Therefore, we optimize our approach to predict navigable next-best viewpoints in free space rather than relying on classical single-step actions.

To enhance practical effectiveness, we capture four surrounding views at each predicted position, simulating the scanning process. This multi-view setup provides a broader spatial context and enables more effective long-term planning during policy training.

D. Additional Results

D.1. The Reward of Trajectory Efficiency

Table 10. The effect of path efficiency reward. †: trained on 128 scenes and half-standard 2.5k iterations.

Settings	**Cov.**	**AUC**	**Comp.**	**KF**	**TL**
GLEAM †	60.23%	51.69%	0.89m	23.20	47.32m
GLEAM † with effi. rew.	56.61%	47.91%	0.96m	17.91	34.41m

As shown in Table 10, while implementing a generic efficiency reward term $r_t^{\text{Effi}} = -1$ [9] indeed reduce the number of keyframes (-5.29) and trajectory length (-12.91m), it penalizes exploratory actions like detouring around obstacles, leading to conservative policies (-3.62% Cov.).