BeliefMapNav: 3D Voxel-Based Belief Map for Zero-Shot Object Navigation

Zibo Zhou[1], **Yue Hu**[2], **Lingkai Zhang**[1], **Zonglin Li**[1], **Siheng Chen**[1*]
[1]Shanghai Jiao Tong University
[2]University of Michigan
{zb_zhou, zhanglingkai, channeler_xfya, sihengc}@sjtu.edu.cn
huyu@umich.edu

Abstract

Zero-shot object navigation (ZSON) allows robots to find target objects in unfamiliar environments using natural language instructions, without relying on pre-built maps or task-specific training. Recent general-purpose models, such as large language models (LLMs) and vision-language models (VLMs), equip agents with semantic reasoning abilities to estimate target object locations in a zero-shot manner. However, these models often greedily select the next goal without maintaining a global understanding of the environment and are fundamentally limited in the spatial reasoning necessary for effective navigation. To overcome these limitations, we propose a novel 3D voxel-based belief map that estimates the target's prior presence distribution within a voxelized 3D space. This approach enables agents to integrate semantic priors from LLMs and visual embeddings with hierarchical spatial structure, alongside real-time observations, to build a comprehensive 3D global posterior belief of the target's location. Building on this 3D voxel map, we introduce BeliefMapNav, an efficient navigation system with two key advantages: i) grounding LLM semantic reasoning within the 3D hierarchical semantics voxel space for precise target position estimation, and ii) integrating sequential path planning to enable efficient global navigation decisions. Experiments on HM3D and HSSD benchmarks show that BeliefMapNav achieves state-of-the-art (SOTA) Success Rate (SR) and Success weighted by Path Length (SPL), with a notable **9.7** SPL improvement over the previous best SR method, validating its effectiveness and efficiency. The source code is publicly available at: https://github.com/ZiboKNOW/BeliefMapNav

1 Introduction

Zero-shot object navigation (ZSON) enables robots to locate targets in novel environments through natural language instructions (e.g., "find the red sofa"), eliminating reliance on pre-mapped scenes or object-specific training [1, 2, 3, 4, 5]. In domestic settings, ZSON supports assistive tasks such as retrieving user-specified objects [6]. In industrial inspection, ZSON enables autonomous localization of malfunctioning components (e.g., detecting a leaking pipe) within complex facilities. In warehouse operations, ZSON enhances robotic picking and inventory management by allowing flexible retrieval of objects without pre-built maps. These real-world applications highlight the necessity of robust zero-shot navigation for scalable, adaptable robot deployment across diverse domains.

To enable ZSON, prior works have progressed along two main directions. The first approach constructs bird's-eye view (BEV) value maps [7, 8, 9] by leveraging pixel-level semantic cues to

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

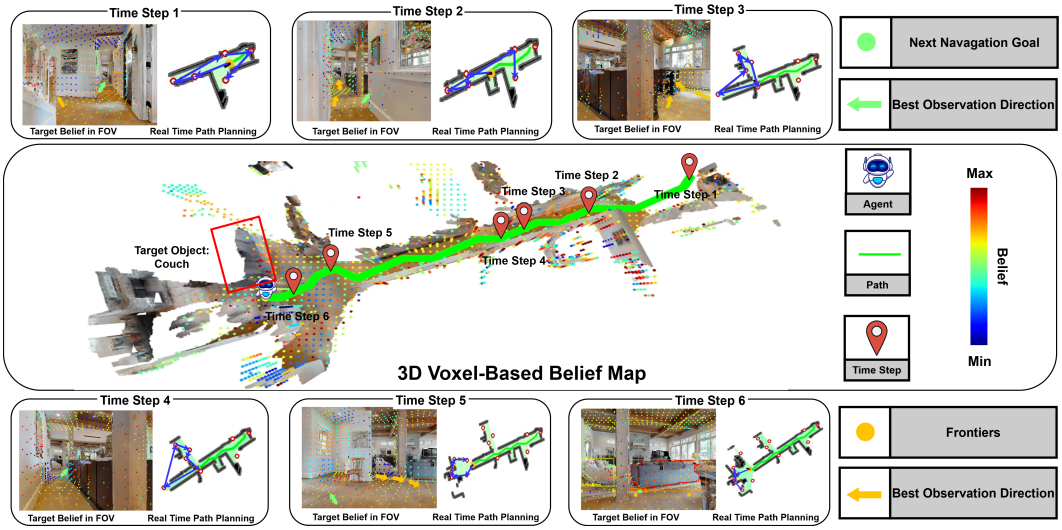


Figure 1: The search process: BeliefMapNav plans frontier paths by minimizing the expected search distance based on the 3D voxel-based belief map, ensuring efficient and stable exploration.

provide dense estimations of target locations in the BEV space. While dense BEV representations provide object position estimation at a fine-grained level, the actual numerical distinctions between positions remain ambiguous with limited discriminative semantics. This insufficient differentiation in positional values, combined with the lack of high-level semantic reasoning, ultimately leads to compromised accuracy in the target location prediction. More recently, a second approach has emerged that employs large language models (LLMs) or vision-language models (VLMs) to reason about the target locations [10, 11, 12, 13, 14]. However, both LLMs and VLMs face limitations in spatial understanding and reasoning [15], which significantly affect target location prediction accuracy. Additionally, VLMs are limited by their inability to effectively extract the most relevant and fine-grained semantic information from image observations, as well as by the lack of spatial information, resulting in imprecise target position predictions [10, 11, 12, 13, 14]. For LLMs, while leveraging spatial information, reliance on semantic content from environment language descriptions leads to significant loss of information and reduced prediction precision [16, 17, 12, 18]. Consequently, LLMs and VLMs both struggle with reliable and accurate target location inference. Furthermore, existing methods generally rely on greedy navigation strategies [12, 8], which cause frequent back-and-forth movements, significantly hindering search efficiency. Together, in existing works, the lack of semantic cues and spatial reasoning leads to inaccurate and imprecise target object position estimation. Meanwhile, the absence of efficiency optimization in the search behavior hinders robust searching for diverse, open-set targets in unbounded real-world 3D spaces.

To enable more precise and accurate predictions of the target object's location within 3D space, we propose a novel 3D voxel-based belief map that considers rich hierarchical spatial semantic cues and LLM-generated target-adaptive semantic cues to reason about prior belief of target presence in dense 3D voxel space. This structured representation enables spatially fine-grained estimation across the 3D voxel space, facilitating more precise and generalizable localization of target objects in complex, unbounded environments. Moreover, this fine-grained and accurate representation enables efficient guidance for high-degree-of-freedom mobile agents, facilitating precise, task-relevant, language-driven search within localized areas and enhancing the robustness of manipulation tasks.

To further enhance search efficiency, we introduce BeliefMapNav, an efficient zero-shot object navigation system based on path sequence optimization over the belief map. The system is composed of three tightly integrated modules and builds upon the belief map framework to enable efficient, goal-directed exploration. 1) The **3D voxel-based belief mapping module** encodes prior beliefs of object presence in 3D space by integrating hierarchical spatial semantics with commonsense knowledge from an LLM. 2) The **frontier observation belief estimation module** combines the belief map with a visibility map, which encodes real-time observation feedback likelihood, to produce posterior beliefs of the target object's position. Then, the module estimates the posterior observation belief of detecting the target in each frontier's field of view (FOV). 3) The **observation belief-based planning module** selects the next navigation goal by minimizing the expected distance cost based on the posterior observation beliefs to find the target. By explicitly modeling uncertainty and updating

spatial beliefs dynamically, BeliefMapNav enables more efficient, goal-directed exploration than methods using static priors or reactive policies.

The contributions of our method are mainly summarized as follows: 1) We propose BeliefMapNav, an efficient zero-shot object navigation system that accurately predicts target location through fine-grained belief estimation in a 3D voxel-based belief map and real-time feedback, enabling belief-driven sequential planning for efficient navigation. 2) At the core of our system is a 3D voxel-based belief map that integrates hierarchical spatial-visual features with LLM-derived commonsense, enabling accurate and fine-grained prior estimation of the target's location. Based on the prior estimation, we design a planner that optimizes navigation by minimizing the expected path distance cost for efficient navigation. 3) The proposed system achieves SOTA performance on the HM3D [19] and HSSD [20] benchmarks, surpassing all the previous zero-shot methods by 3.4% in SR and 9.7 in SPL on HM3D, 14.2% in SR and 7.2 in SPL on HSSD, thereby demonstrating the overall effectiveness and efficiency of our approach.

2 Related Works

Object Navigation Object navigation refers to the task of guiding a robot to search given target objects in an unknown environment. It can be divided into two categories: i) training-required methods, such as reinforcement learning (RL) and imitation learning, which require extensive training on task-specific data [21, 22, 23, 24, 25], and ii) zero-shot methods, which leverage pre-trained models, such as VLMs or LLMs, to perform navigation without additional training [14, 3, 9, 8, 12]. Training-based methods typically require large amounts of data and have difficulty generalizing due to limited environmental diversity [19, 26], while zero-shot methods offer flexibility and adaptability to novel environments, but are constrained by the spatial reasoning limitations of LLMs and VLMs [15]. Our method follows a zero-shot approach. We leverage LLMs and rich hierarchical spatial semantics to provide accurate and fine-grained estimations of the target's location, while employing probability-based optimization algorithms to ensure the efficiency of the search path.

Semantic Mapping for Object Navigation. Semantic mapping is important in object navigation as it provides spatially structured representations that guide the robot in locating targets. Traditional methods, such as category-based approaches [14, 27, 17] and scene graph-based methods [28, 16, 18, 29, 30, 31], often rely on predefined categories or topological graphs, leading to semantic information loss and mapping errors due to detection failures [32, 33]. Although value map-based methods [8, 7, 9] aim for non-vocabulary representations, existing methods struggle to align spatial and semantic scales, resulting in incomplete or misaligned spatial semantics under varying scene extents. As a result, the generated maps lack the precision needed to accurately localize target objects. In contrast, our method constructs a multi-level, spatially-aligned semantic map that supports accurate target object localization estimation.

3 Method

In this section, we first define the object navigation task and then present the BeliefMapNav system, which consists of three main modules: 3D voxel-based belief mapping, frontier observation belief estimation, and observation belief-based planning, as shown in Fig. 2.

3.1 Task definition

We define the ZSON task, where an agent is required to locate a specified target object in an unknown environment without task-specific training, pre-built maps, or a fixed vocabulary. The target category c is specified in free-form text. At each timestep t, the agent receives RGB-D observations $I_t = (I_t^{\mathrm{rgb}}, I_t^{\mathrm{depth}})$, where $I_t^{\mathrm{rgb}} \in \mathbb{R}^{H \times W \times 3}$ and $I_t^{\mathrm{depth}} \in \mathbb{R}^{H \times W}$, and its pose $s_t = (x_t, r_t)$, with $x_t \in \mathbb{R}^3$ and $r_t \in \mathrm{SO}(3)$, from odometry. The action space $\mathcal{A}$ includes: MOVE FORWARD (0.25 m), TURN LEFT/RIGHT (30°), LOOK UP/DOWN (30°), and STOP. The task is successful if the agent issues a STOP within 0.1 m of the target object within 500 steps. At each timestep, the system takes as input the current RGB-D observation I_t, the agent's pose s_t, and the text-specified target c, and outputs a navigation action $a_t \in \mathcal{A}$ from the discrete action set.

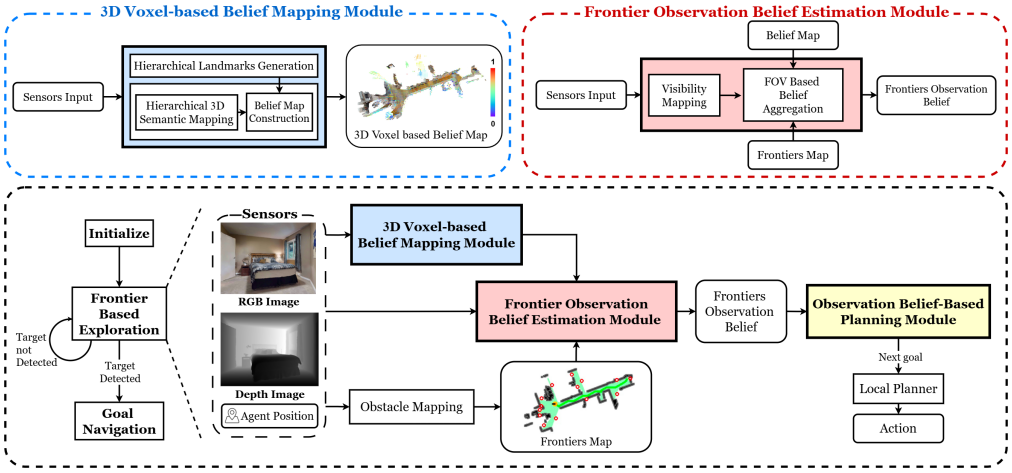


Figure 2: BeliefMapNav pipeline: The agent initializes with a 360° rotation. During exploration, the 3D voxel-based belief mapping module fuses sensor input, the 3D hierarchical semantic map, and landmarks to create a belief map. The frontier observation belief estimation module computes frontier observation belief from the belief, frontiers, and visibility maps via FOV-based aggregation. The observation belief-based planning module selects the next goal based on this belief and outputs navigation actions. Upon detecting the target, the agent navigates to it.

3.2 System overview

BeliefMapNav is a novel 3D voxel-based zero-shot open-vocabulary object navigation system; see the overview in Fig. 2. BeliefMapNav has three key modules: 1) 3D voxel-based belief mapping module in Sec. 3.3, which encodes a prior belief of the target's presence by combining hierarchical spatial semantics with LLM commonsense in a belief map; 2) frontier observation belief estimation module in Sec. 3.4, which fuses the prior belief map with real-time observation likelihood to estimate the posterior belief of detecting the target in each frontier's FOV; 3) observation belief-based planning module in Sec. 3.5, which selects the next navigation point by optimizing expected distance cost to detect the object.

3.3 3D voxel-based belief mapping

3D voxel-based belief mapping aims to represent the spatial prior belief of the target object's presence in a 3D voxel grid. The key intuition is that maintaining a fine-grained representation enables more spatially detailed and accurate prediction of the target's location. To achieve this, we construct a 3D voxel map where each voxel stores the belief of the target existing within its spatial region:

$$\mathcal{B} = \left\{ (u, b_u) \;\middle|\; u \in \mathbb{Z}^3 \right\}$$

where $\mathcal{B}$ denotes the belief map, $u \in \mathbb{Z}^3$ represents the discrete spatial coordinate of a voxel in 3D space and $b_u \in \mathbb{R}$ represents the estimated prior belief that the target object exists within voxel u. Unlike previous methods [7, 8], the 3D voxel-based belief map leverages hierarchical language and spatial semantics to provide more precise and nuanced estimation of the target belief in each voxel of 3D space. It involves three steps as shown in Fig. 3: 1) constructing a 3D hierarchical semantic voxel map based on visual observations in Sec. 3.3.1; 2) using an LLM to infer hierarchical landmarks with corresponding relevance scores in Sec. 3.3.2; and 3) mapping the inferred landmarks and relevance scores into the hierarchical semantic voxel map to form the belief map in Sec. 3.3.3.

3.3.1 3D Hierarchical semantic mapping

The 3D hierarchical semantic voxel map $\mathcal{M}_c$ represents the environment across three levels, $\mathcal{L}_s = \{\text{scene}, \text{region}, \text{object}\}$, with progressively finer semantics. This structure enables reasoning across spatial scales, from coarse layouts to fine object details, improving the accuracy of target object position estimation. Formally: $\mathcal{M}_c = \left\{ \left(u, \{\hat{v}_u^{l_s}, \hat{s}_u^{l_s}\}\right) \;\middle|\; u \in \mathbb{Z}^3, l_s \in \mathcal{L}_s \right\}$. Here, for each level $l_s \in \mathcal{L}_s$, voxel u stores an image CLIP [34] feature vector $\hat{v}_u^{l_s} \in \mathbb{R}^d$ and a feature confidence score

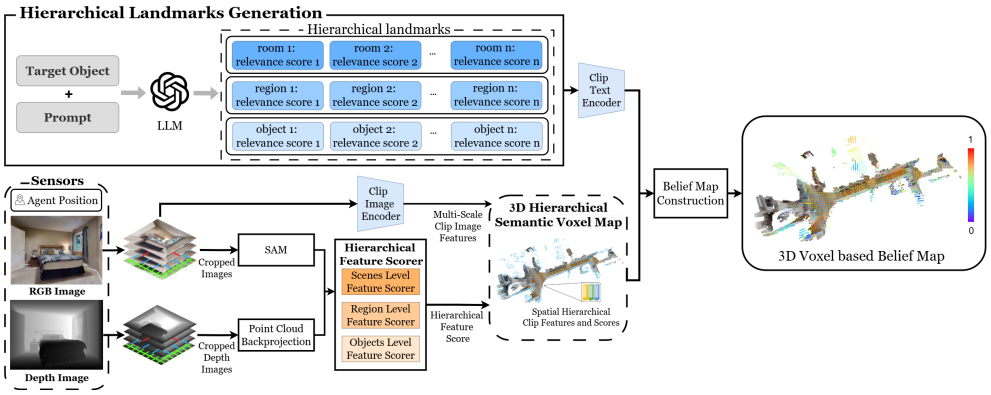


Figure 3: The pipeline begins by passing the target and prompt to the LLM to generate hierarchical landmarks with relevance scores. Meanwhile, RGB and depth images are cropped into multi-scale patches and processed via SAM and point cloud back-projection. Multi-scale CLIP image features are extracted, and the top features selected by hierarchical feature scorers update the hierarchical 3D semantic map. Finally, landmarks encoded by text CLIP and the semantic map are combined via the belief map construction to update the 3D voxel-based belief map.

$\hat{s}_u^{l_s} \in \mathbb{R}$, which quantifies the reliability of the semantic feature at each level and guides the selection of the most informative features for belief map updating.

The module operates in three stages: 1) **Multi-scale feature extraction**: Extract image CLIP features from multi-scale RGB images and spatial information from the depth images. 2) **Hierarchical feature scoring**: Assign confidence scores to features at different image scales, reflecting their relevance to specific semantic levels $\mathcal{L}_s$. These scores enable adaptive scale selection for each semantic level. 3) **Adaptive hierarchical feature selection**: At each hierarchical level, features from the most confident scale are selected and back-projected to update the 3D voxel map $\mathcal{M}_c$.

Multi-scale feature extraction: To better capture both global context and local details, the observed RGB image $I^{rgb} \in \mathbb{R}^{H \times W \times 3}$ is divided into equal-sized patches at multiple scales. At scale k, the image is partitioned into $2^{(k-1)} \times 2^{(k-1)}$ patches, with each patch denoted as $P_{h,w}^k$. We use CLIP [34] to extract visual features $v_{h,w}^k$ for each patch and each patch $P_{h,w}^k$ is processed by the Segment Anything Model (SAM) [35] to estimate the number of semantic instances $n_{h,w}^k$. In parallel, the corresponding depth image is divided in the same way. We back-project depth values of patches into 3D space to form a point cloud and compute two geometric properties: the volume $V_{h,w}^k$ and point density $\rho_{h,w}^k$. The detailed extraction process is in Appendix A.1.

Hierarchical feature scoring: To select features for different spatial semantic levels, we design hierarchical feature scorers for L_s, which assign confidence scores to each image patch. A higher score indicates a better alignment of the feature with the corresponding semantic level. The hierarchical feature scorers are defined as: **(1) Scene Scorer:** This scorer favors patches covering larger spatial extents with more semantic instances, score defined as $S_{h,w}^{\text{scene},k} = w_1 \cdot V_{h,w}^k + w_2 \cdot n_{h,w}^k$. **(2) Region Scorer:** This scorer favors patches with more densely packed instances and concentrated point clouds, score defined as $S_{h,w}^{\text{region},k} = w_3 \cdot \left(\frac{n_{h,w}^k}{V_{h,w}^k} \right) + w_4 \cdot \rho_{h,w}^k$. **(3) Object Scorer:** This scorer favors patches with a high average point density per instance, score defined as $S_{h,w}^{\text{object},k} = \frac{\rho_{h,w}^k}{n_{h,w}^k}$

Adaptive hierarchical feature selection: Each image pixel position has k scale candidate features, and in each pixel position, we select the image CLIP features for hierarchical semantic levels from multi-scale patches that achieve the highest score under the corresponding semantic-level feature scorer. After scoring, pixels are back-projected into the 3D semantic map, where for each semantic level in every voxel, only the feature with the highest confidence score is retained in the map. The detailed selection method is provided in the Appendix A.2.

3.3.2 Hierarchical landmarks generation

Landmarks are semantic cues inferred by a language model from the target object description, indicating where the object is likely to appear. In our method, we focus on three levels of landmarks $\mathcal{L}_t = \{\text{room}, \text{region}, \text{object}\}$ to assist in locating the target object. To extract these landmarks, we prompt an LLM (GPT-4o [36]) with the target object description, asking it to generate two outputs: (1) a set of landmark strings $\mathcal{R} = [\mathcal{R}^{\text{room}}, \mathcal{R}^{\text{region}}, \mathcal{R}^{\text{object}}]$, where $\mathcal{R}^{l_t} = [r_1^{l_t}, r_2^{l_t}, \ldots, r_{n_{l_t}}^{l_t}]$ denotes the landmarks at level $l_t \in \mathcal{L}_t$; and (2) the corresponding relevance scores $\alpha = [\alpha^{\text{room}}, \alpha^{\text{region}}, \alpha^{\text{object}}]$, where $\alpha^{l_t} = [\alpha_1^{l_t}, \alpha_2^{l_t}, \ldots, \alpha_{n_{l_t}}^{l_t}]$. Where n_{l_t} is the number of landmarks in level l_t. Each score $\alpha_i^{l_t}$ indicates the likelihood that the target object appears near the corresponding landmark $r_i^{l_t}$. Details about the prompt design and generation process are provided in the Appendix A.3.

3.3.3 Belief map construction

After obtaining hierarchical textual landmarks with associated relevance scores, we project both the landmarks and the target object name into the 3D hierarchical semantic voxel map to generate a 3D voxel-based belief map, which represents a prior belief over the target object's presence in space. Each landmark $r_i^{l_t} \in \mathcal{R}^{l_t}$ and the target object name r_{target} are encoded using the CLIP text encoder, resulting in embeddings $\mathcal{E}_i^{l_t}$ for landmarks and $\mathcal{E}_{\text{target}}$ for the target object. For each of these textual inputs, we compute the maximum cosine similarity scores with stored spatial semantic features at corresponding levels: $p_{u,i}^{l_t} = \max_{l_s} \text{cosine}(\mathcal{E}_i^{l_t}, \hat{v}_u^{l_s})$ and $p_{u,\text{target}} = \max_{l_s} \text{cosine}(\mathcal{E}_{\text{target}}, \hat{v}_u^{l_s})$. Each similarity score is weighted by its associated relevance score $\alpha_i^{l_t}$ for landmarks and $\alpha_{\text{target}} = 1$ for the target object. The final belief score at voxel u is computed as: $b_u = \sum_{l_t \in \mathcal{L}_t} \sum_{i=1}^{n_{l_t}} \alpha_i^{l_t} \cdot p_{u,i}^{l_t} + p_{u,\text{target}}$. The values b_u in the 3D voxel-based belief map represent the prior belief of the target object's presence in each voxel.

3.4 Frontier observation belief estimation

The 3D voxel-based belief map serves as the prior, through which the frontier observation belief module dynamically integrates visibility map likelihoods to calculate the belief of detecting the target object within each frontier's FOV.

3.4.1 Visibility map

To capture the impact of real-time detection feedback on the target belief distribution, we introduce a visibility map. The visibility map is inspired by [37] and is defined as $\mathcal{P}^v = \left\{(u, \hat{p}_u^v) \;\middle|\; u \in \mathbb{Z}^3,\ \hat{p}_u^v \in [0,1]\right\}$, $\hat{p}_u^v$ is the likelihood of detecting the target at voxel u based on visual observations during the search. The key intuition is that if a voxel u has high detection confidence in the FOV but no detection, the belief that the target object exists in that region is very low ($\hat{p}_u^v \to 0$). By estimating the object absence likelihood, the visibility map can refine prior beliefs to prevent revisiting areas that have been well observed. This correction improves navigation efficiency. The detection confidence is calculated in a way that it decreases near image boundaries and at longer distances from the camera pose. Details of the construction of the visibility map are provided in the Appendix A.4.

3.4.2 FOV-based belief aggregation

After constructing the visibility map $\mathcal{P}^v$ and belief map $\mathcal{B}$, we fuse them to obtain the **posterior belief map** $\mathcal{B}^{\text{post}} = \left\{(u, \hat{b}_u^{\text{post}}) \;\middle|\; u \in \mathbb{Z}^3,\ \hat{b}_u^{\text{post}} = \hat{p}_u^v \cdot b_u\right\}$. This fusion enables more dynamic and accurate estimation of the belief of detecting the target from each frontier's FOV by combining spatial priors belief map with the observation feedback visibility map. It improves search efficiency and reduces exploration of well-observed regions. For each candidate frontier position x_{f_i}, we evaluate four viewing directions $\theta \in \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$. For each θ, we perform **ray casting** from x_{f_i} to identify voxels within the FOV, separating them into $\mathcal{O}_{\text{map}}^\theta(x_{f_i})$ (voxels in the belief map) and $\mathcal{O}_{\text{unk}}^\theta(x_{f_i})$ (voxels out of belief map). The observation belief for direction θ is computed as: $P_{\text{obs}}^\theta(x_{f_i}) = \sum_{u \in \mathcal{O}_{\text{map}}^\theta(x_{f_i})} \hat{b}_u^{\text{post}} + |\mathcal{O}_{\text{unk}}^\theta(x_{f_i})| \cdot w_{\text{unobserved}}$. Where $|\mathcal{O}_{\text{unk}}^\theta(x_{f_i})|$ is the number of

voxels not in map and $w_{\text{unobserved}}$ is a constant weight. The final observation belief at x_{f_i} is defined as: $P_{\text{obs}}(x_{f_i}) = \max_\theta P^\theta_{\text{obs}}(x_{f_i})$. Aggregating over the FOV allows us to account for obscure vision and the agent's limited FOV, leading to a more accurate estimation of the likelihood of observing the target in each frontier's FOV.

3.5 Observation belief-based planning module

Using the estimated observation belief at each frontier based on the posterior belief map, we prioritize frontiers to minimize expected search distance for a more stable and efficient target search. Unlike previous greedy approaches that myopically prioritize immediate gain, our distance-based optimization ensures smoother early-stage movement under uncertain belief distributions and gradually converges to the optimal path as the belief becomes more reliable, resulting in more efficient exploration. The optimal exploration strategy seeks a permutation of frontier visiting sequence $\pi = [f_{\pi_1}, f_{\pi_2}, \ldots, f_{\pi_n}]$ that minimizes the expected search cost, where $f_{\pi_i} \in \{x_{f_1}, \ldots, x_{f_n}\}$ denotes the i-th frontier position. The objective is formulated as:

$$\pi^* = \underset{\pi \in S_n}{\operatorname{argmin}} \sum_{i=1}^{n} \left(\sum_{k=1}^{i} d_{A^*}(f_{\pi_{k-1}}, f_{\pi_k}) \right) P_{\text{obs}}(f_{\pi_i})$$

where S_n denotes the permutation group over n frontier points, f_{π_k} represents the k-th visited frontier in permutation π (with the initial point $f_{\pi_0} \equiv x_0$ defined as the agent's current position), $d_{A^*}(f_{\pi_{k-1}}, f_{\pi_k})$ denotes the path distance between adjacent frontiers computed via the A* [38] algorithm, $\sum_{k=1}^{i} d_{A^*}(\cdot)$ calculates the cumulative path cost to the i-th frontier, and $P_{\text{obs}}(f_{\pi_i})$ is the observation belief of frontier f_{π_i}.

The proposed objective improves search efficiency by minimizing exploration cost with A*-optimized paths and prioritizing high-belief frontiers via observation-weighted costs. Combining geometric path costs and belief weights, it reduces noise impact on navigation stability. Integrating shortest-path guarantees with probabilistic reasoning, it enables dynamic, real-time replanning, adapting to evolving beliefs for flexible, efficient navigation. At the same time, the precise and detailed belief map provides a solid foundation for effective optimization. We solve this optimization via GPU-accelerated simulated annealing [39]. The detailed optimization process is in the Appendix A.5. Before each action, the agent selects the first frontier in the optimized sequence π^* as the next navigation target and replans at every step with the updated belief map. We adopt the local point navigation planner from VLFM [7] to generate actions toward the given goal. An open-vocabulary detector [40] and GPT-4o [36] verify detected objects; if confirmed, the system localizes the target using the bounding box, SAM [35], and depth, then sets it as the final navigation goal.

4 Experimental Results

In this section, we outline datasets and key implementation details, then compare BeliefMapNav's performance against SOTA baselines on HM3D [19], MP3D [26], and HSSD [20]. Ablation studies assess each component's contribution. Qualitative analysis visualizes maps indicating target presence probability. Search process visualizations highlighting our approach are in Appendix A.9. Baseline summaries and HM3D failure analyses appear in Appendix A.6 and A.7, respectively.

4.1 Benchmarks and Implementation details

Dataset: We evaluate our method on three standard benchmarks: HM3D [19], MP3D [26] and HSSD [20]. HM3D, the official dataset of the Habitat 2022 ObjectNav Challenge, includes 2,000 validation episodes across 20 environments and 6 object categories. MP3D, a large-scale indoor 3D scene dataset, is commonly used in Habitat-based ObjectNav evaluations. We conduct experiments on its validation set, consisting of 11 environments, 21 object categories, and 2,195 object-goal navigation episodes. HSSD, a synthetic dataset with scenes based on real house layouts, contains 40 validation scenes, 1,248 navigation episodes, and 6 object categories.

Evaluation Metrics: We use two standard metrics: Success Rate (SR) and Success weighted by Path Length (SPL). SR measures the proportion of episodes where the agent reaches the target within a

Table 1: Zero-shot object navigation results on MP3D, HM3D and HSSD. We compare the SR and SPL of state-of-the-art methods in different settings.

Method	Unsupervised	Zero-shot	HM3D		MP3D		HSSD	
			SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑
Habitat-Web [41]	✗	✗	41.5	16.0	31.6	8.5	-	-
OVRL [42]	✗	✗	-	-	28.6	7.4	-	-
ProcTHOR [43]	✗	✗	54.4	31.8	-	-	-	-
SGM [44]	✗	✗	60.2	30.8	37.7	14.7	-	-
ZSON [24]	✗	✓	25.5	12.6	15.3	4.8	-	-
PSL [45]	✗	✓	42.4	19.2	18.9	6.4	-	-
PixNav [11]	✗	✓	37.9	20.5	-	-	-	-
VLFM [7]	✓	✓	52.5	30.4	36.4	17.5	-	-
ESC [3]	✓	✓	39.2	22.3	28.7	14.2	38.1	22.2
CoWs [13]	✓	✓	-	-	9.2	4.9		
L3MVN [14]	✓	✓	50.4	23.1	34.9	14.5	41.2	22.5
ImagineNav [46]	✓	✓	53.0	23.8	-	-	51.0	24.9
VoroNav [28]	✓	✓	42.0	26.0	-	-	41.0	23.2
GAMap [8]	✓	✓	53.1	26.0	-	-	-	-
OpenFMNav [47]	✓	✓	52.5	24.1	37.2	15.7	-	-
SG-Nav [12]	✓	✓	54.0	24.9	40.2	16.0	-	-
UniGoal [48]	✓	✓	54.5	25.1	**41.0**	16.4	-	-
InstructNav [9]	✓	✓	58.0	20.9	-	-	-	-
BeliefMapNav	✓	✓	**61.4**	**30.6**	37.3	**17.6**	**65.2**	**32.1**

preset distance. SPL evaluates path efficiency by considering both success and trajectory optimality: if successful, SPL $= \frac{\text{Optimal path length}}{\text{path length}}$, otherwise SPL $= 0$. Higher values indicate better performance.

Implementation details: We limit navigation to 500 steps, defining success as stopping within 0.1 m of the target. Each step moves the agent 0.25 m forward or rotates it by 30°. The RGB-D camera, mounted 0.88 m high, captures 640 × 480 images. The 3D voxel map has 45,000 voxels at 0.25 m resolution. We set $w_{\text{unobserved}} = 0.01$ (Sec. 3.4.2). CLIP-ViT-B-32 encodes visual/text features with image crop scale $k = 3$. GPT-4o generates three landmarks per level (nine total). Hierarchical scorer weights are $w_1 = 0.05$, $w_2 = 0.1$, $w_3 = 2$, $w_4 = 0.01$. The system runs on a single RTX 4090 (24 GB VRAM) and uses approximately 13 GB of VRAM. Local planner parameters vary slightly by dataset, while the exploration module remains unchanged.

4.2 Comparison with SOTA methods

In this section, we compare our proposed BeliefMapNav with SOTA object navigation approaches in different settings, including unsupervised, supervised, and zero-shot methods on the MP3D [26], HM3D [19], and HSSD [20] benchmarks. As shown in Table 1, our method outperforms all existing zero-shot baselines except the SR of UniGoal and SG-Nav, achieving significant improvements across multiple benchmarks. Specifically, on **HM3D**, we observe a gain of +3.4% in SR and +0.2 in SPL. On **MP3D**, we achieve +0.1 in SPL. Finally, on **HSSD**, our method delivers remarkable gains of +14.2% in SR and +7.2 in SPL. These results highlight the effectiveness of our approach in enhancing both SR and SPL across diverse datasets. While our SR on the MP3D dataset is marginally lower than that of UniGoal and SG-Nav, this is primarily due to their utilization of an object verification mechanism. This feature specifically targets the reduction of detector false positives (FP) caused by poor mesh quality inherent in MP3D. Crucially, while such object verification is an important technique, it remains orthogonal to our main contribution, which focuses squarely on efficient target-oriented exploration. This distinction is further evidenced by our performance on the high-quality HM3D dataset, where our method significantly outperforms UniGoal in both SR (+6.9%) and SPL (+5.5).

On the HM3D dataset, our method improves SPL by 9.7 compared to the zero-shot method InstructNav [9], which achieves the highest SR. While InstructNav prioritizes SR with a dense search strategy, our approach maintains high success rates and boosts search efficiency by generating more accurate

target position estimates and optimizing the search path with a distance cost-aware planner. On the MP3D benchmark, improvements are less pronounced due to two factors: first, the lower data quality of MP3D, which makes target recognition more challenging. Second, there are a lot of mesh "holes" in MP3D, which allow the agent to see through obstacles, causing it to mistakenly prioritize these holes as targets, leading to navigation failures when it gets stuck near non-traversable areas. However, on the HSSD dataset, performance significantly improves because the synthetic scenes avoid the issues present in MP3D and HM3D. Across all datasets, the performance limitations of the local planner in [7] lead to significant degradation, especially in narrow areas.

4.3 Ablation study

To evaluate the effectiveness of each module in our system, we conduct an ablation study on 400 randomly sampled episodes from the HM3D validation set, using a fixed random seed. The ablation study of the effectiveness of LLMs and image scale k are in Appendix A.8.

Table 2: Impact of the planner and visibility map.

Method	SR↑	SPL↑
BeliefMapNav w/o Planner	56.0	29.3
BeliefMapNav w/o Visibility Map	57.2	28.0
BeliefMapNav	**62.5**	**31.6**

Table 3: Impact of vision-language encoders.

Encoder	SR↑	SPL↑
BLIP [49]	59.3	31.0
BLIP2 [50]	62.0	31.1
CLIP [34]	**62.5**	**31.6**

Effectiveness of visibility map and belief-based planning: In Table 2, we compare the effectiveness of the Visibility Map and Belief-based planning against basic exploration. Without the Visibility Map, relying on spatial priors alone leads to a $5.3\% \downarrow$ drop in SR and $3.6 \downarrow$ in SPL, as the agent revisits previously observed regions. Without the planner, navigation is based solely on the highest posterior belief, resulting in a $6.5\% \downarrow$ drop in SR and $2.3 \downarrow$ in SPL due to frequent navigation goal switching and inefficient back-and-forth movement.

Effectiveness of different levels of hierarchical 3D semantics: As shown in Table 5, we evaluate the impact of different levels of the Hierarchical 3D Semantic Map on performance, comparing four settings: no semantics (random walk), scene-level only, scene + region levels, and the full hierarchy with object-level semantics. Results indicate that incorporating more semantic levels generally improves SR. However, omitting object-level semantics enhances efficiency (32.0), as fine-grained searches with object-level cues increase success rates but often result in slower, localized exploration, leading to longer paths and slightly reduced efficiency.

Table 4: Impact of different levels of landmarks.

Landmarks	SR↑	SPL↑
w/o	60.0	30.9
Room	61.0	31.1
Room+Region	61.5	31.2
Room + Region + Object	**62.5**	**31.6**

Table 5: Impact of different semantic levels.

Semantics	SR↑	SPL↑
Random Walking	21.5	10.8
Scene	59.0	30.4
Scene + Region	61.5	**32.0**
Scene + Region + Object	**62.5**	31.6

Effectiveness of different vision-language encoders: as shown in Table 3, CLIP- and BLIP-2-based systems achieve comparable performance (SR: 62.5 vs. 62.0; SPL: 31.6 vs. 31.1), both outperforming BLIP. Prior work [51] similarly shows that BLIP-2 slightly surpasses CLIP in zero-shot text-to-image retrieval accuracy, while both are significantly better than BLIP. However, CLIP demonstrates stronger generalization to out-of-distribution data and supports efficient inference via independent encoders and pre-computed features. These properties make CLIP a more robust and practical encoder choice for retrieval-based navigation tasks.

Effectiveness of hierarchical landmarks: As shown in Table 4, without landmarks, we retrieve directly using the object name in the hierarchical 3D semantic map. With incremental landmark introduction, we observe a gradual improvement in SR (60 to 62.5) and SPL (30.9 to 31.6). However, this improvement is less significant than the gain from incorporating spatial semantics at different hierarchical levels in Table 5, as object names already show inherent relevance to scene, region, and object semantics. In this context, hierarchical landmarks mainly reinforce these semantic associations.

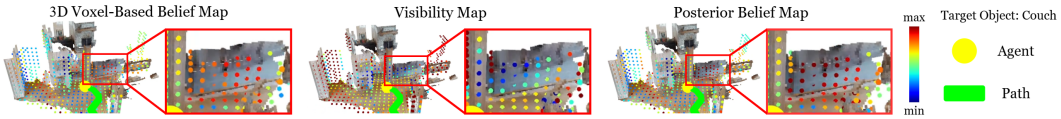


Figure 4: Visualization of the prior belief map, visibility map, and the posterior belief map, with an enlarged section highlighting the target object.

4.4 Qualitative Analysis

Fig. 4 shows the 3D voxel-based belief map, visibility map, and posterior belief map, and the complete visualization is in Fig. 12 of Appendix. The posterior belief map assigns high belief to the target object's (couch) location. While the visibility map indicates low likelihood, the prior belief map, guided by vision-and-language cues, strongly suggests the target's presence, effectively guiding the agent's search (Fig. 1). Additional visualizations are provided in Appendix A.9.

5 Conclusion

In this paper, we have proposed BeliefMapNav, a zero-shot object navigation system that integrates hierarchical spatial semantics, commonsense reasoning from LLMs, and real-time feedback through a 3D voxel-based belief representation. Compared to prior approaches that rely on planar value maps or local greedy goal selection, BeliefMapNav maintains a global 3D belief posterior, performs visibility-aware updates, and plans over informative frontiers, improving robustness to environmental complexity, reducing backtracking, and enabling more efficient exploration. Experiments on three benchmarks show that it outperforms previous methods. Ablation studies highlight the effectiveness of our belief map and belief-based planning for efficient exploration, emphasizing the system's effectiveness and efficiency.

Limitation and future work. We validate the effectiveness of the 3D voxel-based belief map solely on object navigation tasks. This high-resolution 3D map can be further extended to enable robot interaction for mobile manipulation tasks, with future work focusing on real-world implementation.

Acknowledgements

The authors gratefully acknowledge the computational support provided by the CMIC of Shanghai Jiao Tong University.

References

[1] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. *arXiv preprint arXiv:2004.05155*, 2020.

[2] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33:4247–4258, 2020.

[3] Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. In *International Conference on Machine Learning*, pages 42829–42842. PMLR, 2023.

[4] Gengze Zhou, Yicong Hong, and Qi Wu. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7641–7649, 2024.

[5] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. *URL https://arxiv.org/abs/2210.03629*, 2023.

[6] Peiqi Liu, Yaswanth Orru, Jay Vakil, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Ok-robot: What really matters in integrating open-knowledge models for robotics. *arXiv preprint arXiv:2401.12202*, 2024.

[7] Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 42–48. IEEE, 2024.

[8] Hao Huang, Yu Hao, Congcong Wen, Anthony Tzes, Yi Fang, et al. Gamap: Zero-shot object goal navigation with multi-scale geometric-affordance guidance. *Advances in Neural Information Processing Systems*, 37:39386–39408, 2024.

[9] Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment. *arXiv preprint arXiv:2406.04882*, 2024.

[10] Dylan Goetting, Himanshu Gaurav Singh, and Antonio Loquercio. End-to-end navigation with vision language models: Transforming spatial reasoning into question-answering. *arXiv preprint arXiv:2411.05755*, 2024.

[11] Wenzhe Cai, Siyuan Huang, Guangran Cheng, Yuxing Long, Peng Gao, Changyin Sun, and Hao Dong. Bridging zero-shot object navigation and foundation models through pixel-guided navigation skill. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5228–5234. IEEE, 2024.

[12] Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. Sg-nav: Online 3d scene graph prompting for llm-based zero-shot object navigation. *Advances in Neural Information Processing Systems*, 37:5285–5307, 2024.

[13] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23171–23181, 2023.

[14] Bangguo Yu, Hamidreza Kasaei, and Ming Cao. L3mvn: Leveraging large language models for visual target navigation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3554–3560. IEEE, 2023.

[15] Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. *arXiv preprint arXiv:2412.14171*, 2024.

[16] Kunal Pratap Singh, Jordi Salvador, Luca Weihs, and Aniruddha Kembhavi. Scene graph contrastive learning for embodied navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10884–10894, 2023.

[17] Junting Chen, Guohao Li, Suryansh Kumar, Bernard Ghanem, and Fisher Yu. How to not train your dragon: Training-free embodied object goal navigation with semantic frontiers. *arXiv preprint arXiv:2305.16925*, 2023.

[18] Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12689–12699, 2021.

[19] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*, 2021.

[20] 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.

[21] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. *arXiv preprint arXiv:1911.00357*, 2019.

[22] Arsalan Mousavian, Alexander Toshev, Marek Fišer, Jana Košecká, Ayzaan Wahid, and James Davidson. Visual representations for semantic target driven navigation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8846–8852. IEEE, 2019.

[23] Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. *arXiv preprint arXiv:1810.06543*, 2018.

[24] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *Advances in Neural Information Processing Systems*, 35:32340–32352, 2022.

[25] Oleksandr Maksymets, Vincent Cartillier, Aaron Gokaslan, Erik Wijmans, Wojciech Galuba, Stefan Lee, and Dhruv Batra. Thda: Treasure hunt data augmentation for semantic navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15374–15383, 2021.

[26] 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.

[27] Qianfan Zhao, Lu Zhang, Bin He, Hong Qiao, and Zhiyong Liu. Zero-shot object goal visual navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2025–2031. IEEE, 2023.

[28] Pengying Wu, Yao Mu, Bingxian Wu, Yi Hou, Ji Ma, Shanghang Zhang, and Chang Liu. Voronav: Voronoi-based zero-shot object navigation with large language model. *arXiv preprint arXiv:2401.02695*, 2024.

[29] Rui Liu, Xiaohan Wang, Wenguan Wang, and Yi Yang. Bird's-eye-view scene graph for vision-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10968–10980, 2023.

[30] Zachary Ravichandran, Lisa Peng, Nathan Hughes, J Daniel Griffith, and Luca Carlone. Hierarchical representations and explicit memory: Learning effective navigation policies on 3d scene graphs using graph neural networks. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 9272–9279. IEEE, 2022.

[31] Hayato Suzuki, Kota Shimomura, Tsubasa Hirakawa, Takayoshi Yamashita, Hironobu Fujiyoshi, Shota Okubo, Nanri Takuya, and Wang Siyuan. Human-like guidance by generating navigation using spatial-temporal scene graph. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 1988–1995. IEEE, 2024.

[32] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615. IEEE, 2023.

[33] Hao Huang, Shuaihang Yuan, CongCong Wen, Yu Hao, and Yi Fang. Noisy few-shot 3d point cloud scene segmentation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11070–11077. IEEE, 2024.

[34] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[35] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[36] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[37] Muhammad Fadhil Ginting, Sung-Kyun Kim, David D Fan, Matteo Palieri, Mykel J Kochenderfer, and Ali-akbar Agha-Mohammadi. Seek: Semantic reasoning for object goal navigation in real world inspection tasks. *arXiv preprint arXiv:2405.09822*, 2024.

[38] 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.

[39] Dimitris Bertsimas and John Tsitsiklis. Simulated annealing. *Statistical science*, 8(1):10–15, 1993.

[40] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pages 38–55. Springer, 2024.

[41] Ram Ramrakhya, Eric Undersander, Dhruv Batra, and Abhishek Das. Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5173–5183, 2022.

[42] Karmesh Yadav, Ram Ramrakhya, Arjun Majumdar, Vincent-Pierre Berges, Sachit Kuhar, Dhruv Batra, Alexei Baevski, and Oleksandr Maksymets. Offline visual representation learning for embodied navigation. In *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*, 2023.

[43] 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.

[44] Sixian Zhang, Xinyao Yu, Xinhang Song, Xiaohan Wang, and Shuqiang Jiang. Imagine before go: Self-supervised generative map for object goal navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16414–16425, 2024.

[45] Xinyu Sun, Lizhao Liu, Hongyan Zhi, Ronghe Qiu, and Junwei Liang. Prioritized semantic learning for zero-shot instance navigation. In *European Conference on Computer Vision*, pages 161–178. Springer, 2024.

[46] Xinxin Zhao, Wenzhe Cai, Likun Tang, and Teng Wang. Imaginenav: Prompting vision-language models as embodied navigator through scene imagination. *arXiv preprint arXiv:2410.09874*, 2024.

[47] Yuxuan Kuang, Hai Lin, and Meng Jiang. Openfmnav: Towards open-set zero-shot object navigation via vision-language foundation models. *arXiv preprint arXiv:2402.10670*, 2024.

[48] Hang Yin, Xiuwei Xu, Lingqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. Unigoal: Towards universal zero-shot goal-oriented navigation. *arXiv preprint arXiv:2503.10630*, 2025.

[49] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pages 12888–12900. PMLR, 2022.

[50] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[51] Pavan Kalyan Tankala, Piyush Pasi, Sahil Dharod, Azeem Motiwala, Preethi Jyothi, Aditi Chaudhary, and Krishna Srinivasan. Wikido: A new benchmark evaluating cross-modal retrieval for vision-language models. *Advances in Neural Information Processing Systems*, 37:140812–140827, 2024.

[52] ROYUD Nishino and Shohei Hido Crissman Loomis. Cupy: A numpy-compatible library for nvidia gpu calculations. *31st confernce on neural information processing systems*, 151(7), 2017.

[53] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[54] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[55] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly outline the main contributions of the paper, including the introduction of BeliefMapNav, an efficient zero-shot object navigation system. And all the modules in the system, including a 3D-voxel based belief map, a frontiers observation belief map, and an optimization planner.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation of our work in Sec. 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We do not have any assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have some implementation details in Sec. 4.1. The results in our paper can be reproduced by the readers.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We have open-sourced our code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We follow the standard testing details which are stated in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have some case studies in Appendix A.7.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Our system can run on a RTX 4090 GPU, as shown in Sec. 4.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We reviewed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: It is discussed in Sec. 1.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have cited all the code and dataset that we use.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: Our paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We have discussed the LLM usage in landmarks generation in Sec. 3.3.2.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Appendix

A.1 Multi-scale feature extraction

To better capture both global context and local details, the observed images are divided into equal-sized patches at different scales. Specifically, given an RGB observation image I^{rgb} of height H and width W, at image scale k, the image is divided into $2^{(k-1)} \times 2^{(k-1)}$ equally-sized patches, resulting in a total of $4^{(k-1)}$ patches. Each patch at level k has image size: $\frac{H}{2^{(k-1)}} \times \frac{W}{2^{(k-1)}}$, and we denote the set of patches at level k as: $I_k^{rgb} = \{P_{h,w}^k \mid 1 \leq h \leq 2^{(k-1)},\ 1 \leq w \leq 2^{(k-1)}\}$, where $P_{h,w}^k$ represents the patch located at the h-th row and w-th column of the partitioned image at scale k. We employ the CLIP model [34] to compute visual features for each patch. Specifically, for a given patch $P_{h,w}^k$, the corresponding visual feature is $v_{h,w}^k$. This process yields k candidate features for each pixel, and then each patch $P_{h,w}^k \in I_k^{rgb}$ is independently processed using the Segment Anything Model (SAM) [35] to estimate the number of semantic instances it contains $n_{h,w}^k$, where $n_{h,w}^k$ denotes the number of instances detected within the patch $P_{h,w}^k$ at scale k.

In parallel, the depth image is divided into patches using the same scheme as the RGB image. For each depth patch $D_{h,w}^k$ that corresponds to the position of RGB patch $P_{h,w}^k$. We back-project the depth values into 3D space to generate a point cloud: $\mathcal{C}_{h,w}^k$. We then compute two geometric properties of each point cloud: the volume $V_{h,w}^k$ of the point cloud, density $\rho_{h,w}^k$, defined as the number of points per unit volume: $\rho_{h,w}^k = \frac{|\mathcal{C}_{h,w}^k|}{V_{h,w}^k}$, where $|\mathcal{C}_{h,w}^k|$ denotes the total number of 3D points in the patch.

A.2 Adaptive hierarchical feature selection

For each image pixel p at hierarchical spatial level l_s, we select the CLIP feature from all candidate patches across scales that achieves the highest score under the corresponding scorer: $v_p^{l_s} = v_{h^*,w^*}^{k^*}, s_p^{l_s} = S_{h^*,w^*}^{l_s,k^*}$, and $(k^*, h^*, w^*) = \arg\max_{k,h,w:p \in P_{h,w}^k} S_{h,w}^{l_s,k}$. Where,$v_p^{l_s}$ denotes the image CLIP feature of pixel p at level l_s, $s_{h,w}^{l_s,k}$ denotes the score assigned to patch $P_{h,w}^k$ at level l_s, and the constraint $p \in P_{h,w}^k$ ensures that only p in patches are considered. After scoring, each pixel is back-projected into 3D and mapped to a voxel. For each semantic level, we keep only the feature with the highest confidence score in each voxel.

After scoring, each image pixel p is back-projected into 3D space global position x_p using the depth image and mapped to the corresponding voxel in the spatial map:

$$x_p = \text{BackProject}(p, I_t^{\text{depth}}(p), s_t) \in \mathbb{R}^3, \quad u_p = \left\lfloor \frac{x_p}{r} \right\rfloor \in \mathbb{Z}^3$$

where r denotes the voxel resolution, and $\lfloor \cdot \rfloor$ indicates the element-wise floor operation used to discretize the 3D coordinate into voxel space. For each semantic level, if the voxel does not contain an existing feature, we directly store the current feature and its associated confidence score. If a feature already exists, we compare the new score with the stored score and retain the feature with the higher confidence. The voxel map is then updated according to the following rule:

$$\hat{v}_{u_p}^{l_s} = \begin{cases} v_p^{l_s}, & \text{if } u_p \notin \mathcal{M}_c \text{ or } s_p^{l_s} > \hat{s}_{u_p}^{l_s} \\ \hat{v}_{u_p}^{l_s}, & \text{otherwise} \end{cases}, \quad \hat{s}_{u_p}^{l_s} = \begin{cases} s_p^{l_s}, & \text{if } u_p \notin \mathcal{M}_c \text{ or } s_p^{l_s} > \hat{s}_{u_p}^{l_s} \\ \hat{s}_{u_p}^{l_s}, & \text{otherwise} \end{cases}$$

A.3 Prompting

In the hierarchical landmarks generation, the complete prompt is as follows:

System: You are a helpful robot to find an object in an unknown environment.
User: Now that we need to find a/an {target}, please provide information about how it might be found in a private house. This information will be embedded with CLIP and must be useful for the robot to recognize the object.

- Provide which room it is likely to be found in.
- Provide where it is likely to be located within specific rooms. Please add the room type in front of the location.
- Provide what other items it is likely to be near.
- What is the probability of finding the {target} respectively around these landmarks? The sum of the probabilities of each level is equal to one.

At the same time, the output must meet the following requirements:

1. Output three related strings for each level. Each string should contain only one piece of information and avoid using "or" constructions.
2. Each piece of information should consist of only the most relevant phrases, not complete sentences. Keep the phrases simple and common; avoid uncertain words like "maybe".
3. Output two two-dimensional lists. For the landmarks string: The first dimension represents the information level, and the second dimension contains the three related strings for each level. For the Probability: The first dimension represents the information level, and the second dimension contains the three related probabilities for each level.
4. Output only the string and Probability, do not include any additional text.

In the generation of parameters d_{min} and d_{max} in the visibility map, the complete prompt is as follows:

System: You are a helpful robot to find an object in an unknown environment.
User: When we use `YOLOv7` and `GroundingDINO` to detect a/an {target} in a simulated mesh private house environment, the quality is poor. What is the best distance from the camera pose to the {target} to detect the {target}? The input images are 640×480 resolution.

1. Please output the distance in meters.
2. Please only output the distance range as a list: `[distance_min, distance_max]`, without any other text.

A.4 Visibility map

As shown in Fig. 5, detection confidence depends on the pixel locations of the voxels in the image and the distance to the camera. Pixels near the image center and at moderate distances to the camera yield higher confidence, while peripheral or extreme-distance regions tend to have lower confidence due to reduced detectability. For each pixel p in the RGBD image, we compute three components: the horizontal angle-based confidence $C_{\text{horizontal}}$, the vertical angle-based confidence C_{vertical}, and the distance-based confidence C_d. The detection confidence of pixel position p is defined as:

$$C_{\text{horizontal}} = \cos^2\left(\frac{\theta_p}{\theta_{\text{hfov}}} \cdot \pi\right),$$

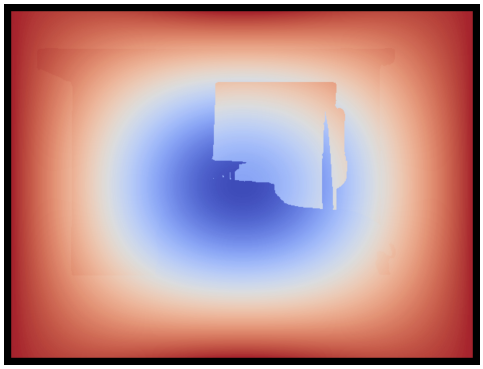
$$C_{\text{vertical}} = \cos^2\left(\frac{\phi_p}{\phi_{\text{vfov}}} \cdot \pi\right),$$

$$C_d = \begin{cases} 1, & \text{if } d_{\text{min}} \le d_p \le d_{\text{max}} \\ \exp\left(-\alpha \cdot \min\left((d_p - d_{\text{min}})^2,\ (d_p - d_{\text{max}})^2\right)\right), & \text{otherwise} \end{cases}$$

$$C_p = C_d \cdot C_{\text{horizontal}} \cdot C_{\text{vertical}}$$

Here, θ_p and ϕ_p denote the horizontal and vertical angles of pixel p, and θ_{hfov}, ϕ_{vfov} are the horizontal and vertical FOV angles in radians. C_d is computed from the pixel depth d_p based on the optimal

(a) Input depth image



(b) Confidence value in FOV

Figure 5: (a) is the input depth image. (b) shows the confidence computed from the depth image. Bluer regions indicate higher confidence, meaning the likelihood of an object being present is low if not detected there. Redder regions indicate lower confidence, implying that even if no object is detected in those areas, the probability of object presence remains relatively high.

range $[d_{\text{min}}, d_{\text{max}}]$ given by LLM A.3. The final confidence score C_p means the confidence to detect the object at pixel position p.

Each pixel p is back-projected into 3D space to obtain the corresponding voxel $u_p \in \mathbb{Z}^3$. The visibility map is then updated as:

$$\hat{p}^v_{u_p} = \begin{cases} 1 - C_p, & \text{if } u_p \notin \mathcal{P}^v \text{ or } 1 - C_p < \hat{p}^v_{u_p} \\ \hat{p}^v_{u_p}, & \text{otherwise} \end{cases}$$

A.5 Path optimization

The optimization objective of the path planning problem is to find a frontier visiting order $\pi = [f_{\pi_1}, f_{\pi_2}, \ldots, f_{\pi_n}]$ that minimizes the expected search distance:

$$\pi^* = \underset{\pi \in S_n}{\operatorname{argmin}} \sum_{i=1}^{n} \left(\sum_{k=1}^{i} d_{A^*}(f_{\pi_{k-1}}, f_{\pi_k}) \right) P_{\text{obs}}(f_{\pi_i})$$

Therefore the cost function $W(\pi)$ to evaluate the quality of a path π can be defined as:

$$W(\pi) = \sum_{i=1}^{n} \left(\sum_{k=1}^{i} d_{A^*}(f_{\pi_{k-1}}, f_{\pi_k}) \right) P_{\text{obs}}(f_{\pi_i})$$

The simulated annealing algorithm is a probabilistic optimization algorithm that can be used to find an approximate solution to the path planning problem. The algorithm is inspired by the annealing process in metallurgy, where a material is heated and then cooled to remove defects and improve its properties. The algorithm works by iteratively exploring the solution space and accepting or rejecting new solutions based on their cost and a temperature parameter. Our implementation is based on the following steps:

1. **Initialization**: Set the initial and terminal temperature T_0 and T_f, the cooling rate α, and the number of samples to simulate N. When the current temperature T is greater than the terminal temperature T_f, the algorithm will continue to run.
2. **Iterative Process**: While the termination criterion is not met, for each sample, the algorithm will perform the following steps:
   (a) **Generating Neighbor Solution**: Generate a neighbor solution π' from the current solution π by applying three kinds of operations: swap, shift, or reverse.
   (1) swap: swap two points in the path. (2) shift: move a segment of the path to a different position. (3) reverse: reverse a segment of the path.

The repetition times of the operations are controlled by the temperature T, which decreases with time. Due to different operations having different degrees of impact on π, the probability of selecting each operation is different. And because the first point in the path is the starting point, it does not participate in this transformation.

(b) **Evaluation and Acceptance**: Evaluate the cost of the new path $W(\pi')$ and apply the Metropolis Criterion: Compare it with the cost of the current path $W(\pi)$. If $W(\pi') < W(\pi)$, accept the new path. Otherwise, accept it with a probability of $\exp\left(\frac{W(\pi)-W(\pi')}{T}\right)$.

(c) **Cooling**: Update the temperature T by multiplying it with the cooling rate α.

3. **Termination**: The algorithm ends when the temperature T is less than the terminal temperature T_f. The final output is the sample with the lowest cost.

The algorithm is implemented in Python and uses the CuPy [52] library to accelerate the process. All the parameters are tuned to achieve a balance between the quality of the solution and the time taken to find it, for the path planning problem. On a laptop with Intel Core i5-12500H CPU, 16GB RAM, and NVIDIA GeForce RTX 2050 Laptop GPU, for a task of 10 frontiers, the algorithm takes about 0.2 seconds to solve the problem. A visualized example problem and the solution are shown in Fig. 7.

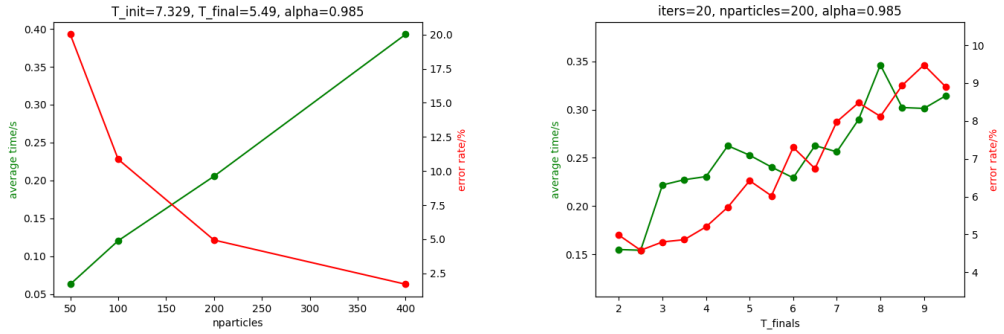


Figure 6: Error analysis of the algorithm for parameter tuning. All the results are based on 10 frontiers, over 50 different scenes, and each scene for 100 times. We take the solution obtained when the algorithm converges as the optimal solution. A solution is considered as an error if the cost is greater than 110% of which of the optimal solution. Between the performance of the algorithm and the time taken to find the solution, we take a balanced approach.

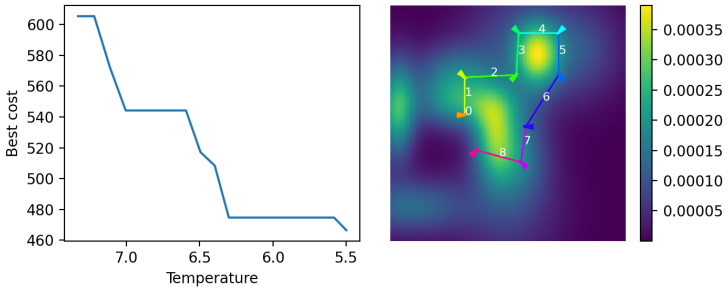


Figure 7: Result demo of the algorithm. The chart on the left shows the relationship between the cost and the temperature under a run. The chart on the right shows the path generated by the algorithm. The number on the lines indicates the order of visiting the frontiers. The arrow at each frontier is the orientation. The color of the background represents the distribution of the occurrence probability. The observation probability of a frontier is the sum of the weights of all points within a sector in front of the frontier, representing points within the FOV of the robot.

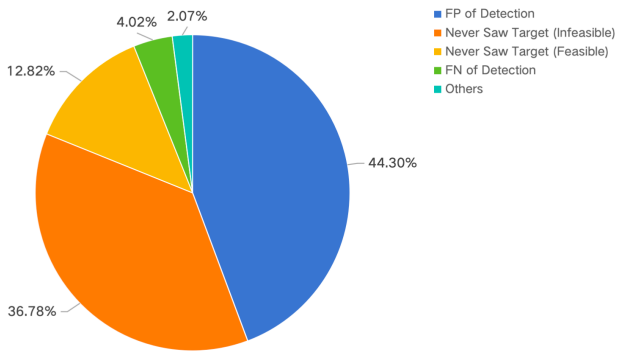


Figure 8: The proportion of different causes of failure in the HM3D dataset.

A.6 Baselines

We evaluate our approach in comparison with a range of ZSON baselines, including several SOTA methods. ZSON [24] incorporates object category cues to enable object-aware navigation. CoWs [13] leverages CLIP features to extract semantic object information and directs exploration toward nearby frontiers. ESC [3] combines a semantic scene representation with commonsense reasoning to guide object search. L3MVN [14] utilizes large language models to infer exploration goals from a semantic map constructed by a pretrained detector. VLFM [7] adopts BLIP-2 [50] for vision-language alignment, using target object descriptions to prioritize exploration frontiers. VoroNav [28] introduces a navigation method based on Voronoi partitioning. InstructNav [9] supports agent navigation by converting the output instructions from the VLM into various value maps. However, this method directly relies on the VLM to reason in spatial contexts, which can lead to hallucinations and reduced navigation efficiency. GAMap [8] guides navigation by leveraging object parts and affordance attributes through an image-based, multi-scale scoring approach that effectively captures both geometric components and functional affordances. SG-Nav [12] guides navigation by leveraging an online 3D scene graph and LLM-based hierarchical chain-of-thought reasoning, enabling explainable, robust zero-shot object navigation with a graph-based re-perception mechanism to correct false positives. UniGoal [48] guides universal zero-shot navigation by leveraging graph-based scene–goal matching and LLM-driven multi-stage exploration, effectively unifying object, image, and text goals with coordinate projection, graph correction, and blacklist-aware reasoning.

A.7 Error Analysis of HM3D

As shown in Fig. 8, the causes of failure cases can be primarily attributed to two factors. First, the current performance limitations of the open vocabulary detector, particularly due to false positives (FP) and false negatives (FN), account for a significant proportion, with these detection errors contributing to 48.32% of the failure cases. These errors result in the system either missing or incorrectly identifying the target object, which ultimately leads to failure in object navigation.

The second major factor stems from the existence of target objects located on a different floor than the agent's starting point in the HM3D dataset, which accounts for 36.78% of the failure cases. Although our 3D voxel map naturally supports modeling of spaces with varying heights, the limitations in stair recognition and local planner performance prevent successful traversal between floors, resulting in the inability to reach the target object on another floor.

Finally, only 12.82% of the failure cases are due to the target object being on the same floor but not found, which demonstrates the effectiveness and efficiency of our exploration module. The "other" category primarily involves cases where the target object was detected but the local planner was unable to navigate towards it, or where the number of steps exceeded the predefined limit.

A.8 Additional ablation study

Effectiveness of different LLMs As shown in Table 6, the choice of LLM exerts only a marginal effect on BeliefMapNav: SR varies by only 1.0% (61.5–62.5) and SPL by 0.8 (30.9–31.7) across four architectures ranging from 8B to trillion-parameter scales. This stability arises because the framework

Table 6: Impact of Different LLMs

LLMs	SR↑	SPL↑
Qwen3-32B [53]	62.0	**31.7**
gemini-2.5-flash [54]	61.8	30.9
Llama-3.3-8B-Instruct [55]	61.5	31.4
GPT-4o [36]	**62.5**	31.6

Table 7: Impact of the different image scales k.

Scale(k)	SR↑	SPL↑
1	57.8	29.1
2	61.3	29.3
3	62.5	**31.6**
4	**62.8**	29.8

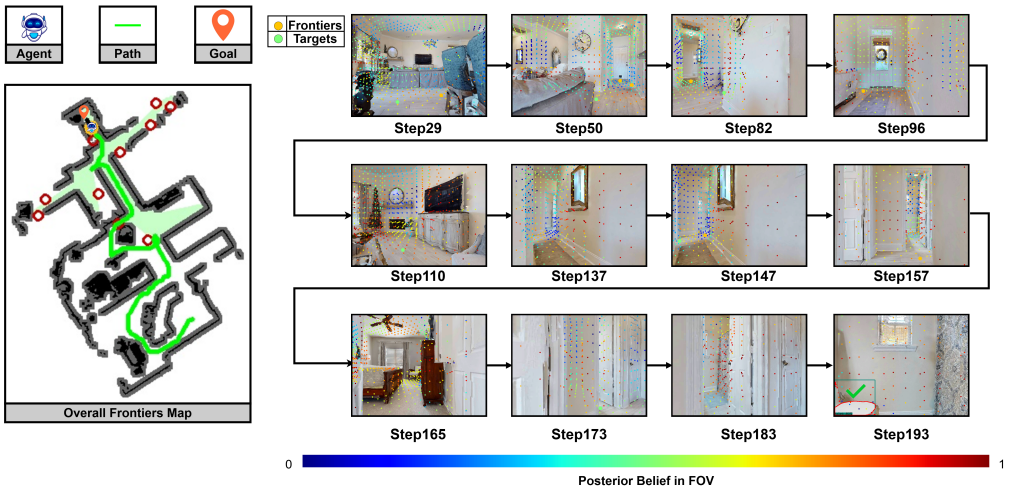


Figure 9: Visualization of the search process. The color of each point in the image represents the belief of object presence: redder points indicate higher belief, while bluer points indicate lower belief.

delegates metric spatial reasoning to the belief map, while the LLM is restricted to inferring landmark plausibilities from textual descriptions—an inference task that lies well within the capability ceiling of contemporary language models. Consequently, the pipeline exhibits negligible sensitivity to LLM specification, alleviating deployment constraints.

Effectiveness of image scale As shown in Table 7, increasing the Image Scales from 1 to 3 consistently improves both success metrics, with SR rising from 57.8 to a peak of 62.5 and SPL reaching its maximum value of 31.6. However, increasing the scale further to 4 yields only minimal gain in SR (from 62.5 to 62.8) but causes a notable reduction in SPL (from 31.6 to 29.8). This degradation in path efficiency is likely attributable to the introduction of noisy, fine-grained details at the small image scale, which negatively impacts target-oriented exploration.

A.9 Visualization

As shown in Fig. 9, 10, and 11, we provide qualitative visualizations to highlight the behavior and strengths of BeliefMapNav. Fig. 12 shows the complete 3D voxel-based belief map, visibility map, and posterior belief map.

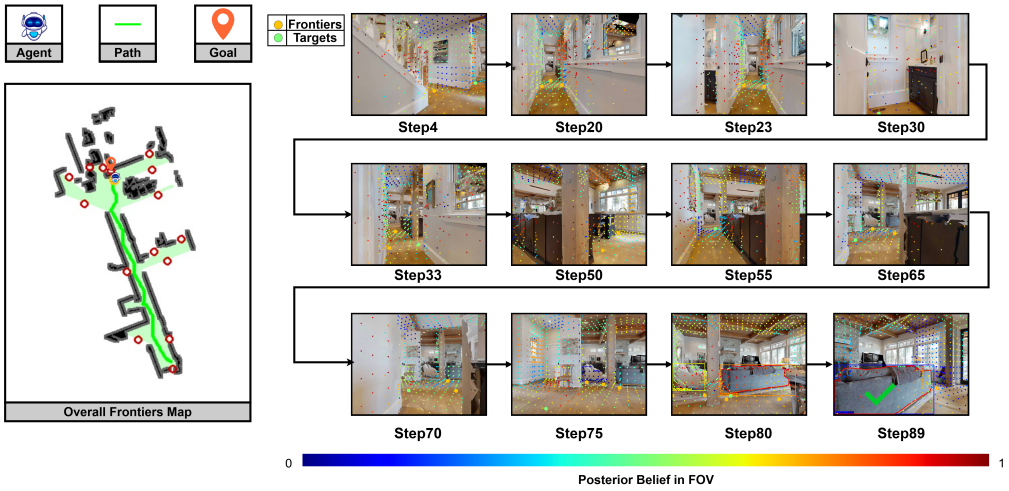


Figure 10: Visualization of the search process. The color of each point in the image represents the belief of object presence: redder points indicate higher belief, while bluer points indicate lower belief.

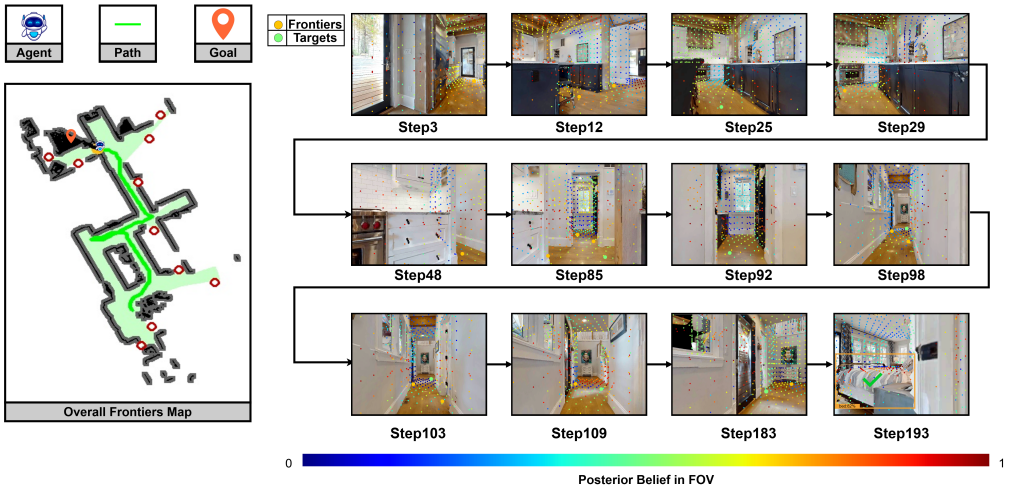


Figure 11: Visualization of the search process. The color of each point in the image represents the belief of object presence: redder points indicate higher belief, while bluer points indicate lower belief.

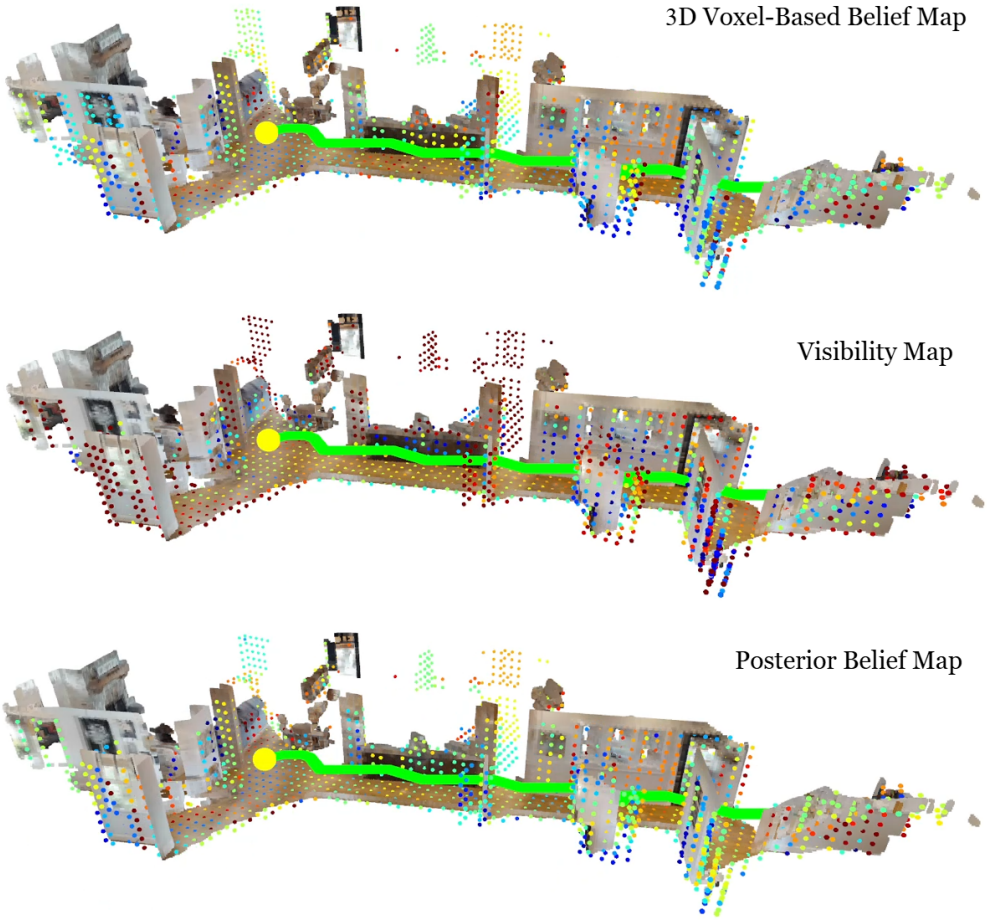


Figure 12: Visualization of the prior belief map, visibility map, and the posterior belief map, with an enlarged section highlighting the target object.