LERR-NAV: LEARNING TO COORDINATE EXPLORATION, RECOVERY, AND REMINISCING FOR OBJECT NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Object Navigation in unseen environments remains a fundamental challenge in embodied AI, as it requires agents to generalize visual and spatial reasoning to novel layouts. While recent advances in large language model (LLM) and vision-language model (VLM) have enhanced semantic mapping, spatial memory, and waypoint selecting for navigation, successful navigation requires more than selecting the next promising waypoint. A robot must also recognize stalled progress, escape unproductive behaviors, and revisit overlooked target cues. Addressing these challenges requires both specialized navigation strategies and a high-level policy that coordinates their execution. Motivated by the analysis of failed navigation episodes, we propose LERR-Nav, an object navigation framework that coordinates exploration, recovery, and reminiscing through a learned high-level policy. The framework comprises four complementary navigation strategies: value-function-guided exploration, LLM-guided exploration, recovery from stalled or repetitive behavior, and memory-guided reminiscing. To support history-aware coordination, we maintain the execution memory that preserves textual summaries of all strategy executions alongside a sliding window of map sequences. To learn the coordination policy, we initialize a high-level vision-language agent from Qwen2.5-VL-7B-Instruct and train it through a two-stage curriculum. The first stage uses map-centric supervised fine-tuning (SFT) to establish spatial grounding. The second stage performs SFT on variable-length navigation segments to learn which navigation strategy to invoke next based on current maps and the hybrid execution memory. Experiments on HM3D and MP3D demonstrate state-of-the-art navigation performance, while ablation studies validate the contributions of individual components. We further deploy our LERR-Nav on a Unitree Go2 quadruped robot for real-world object navigation.

1 INTRODUCTION

Object Navigation is a fundamental and challenging task in Embodied AI, requiring agents to locate specific objects in an unknown, complex environment based on a natural language description of the target category and visual information Batra et al. (2020). Compared to conventional data-driven learning methodsChaplot et al. (2020); Ramakrishnan et al. (2022); Chen et al. (2023); Wasserman et al. (2024); Yoo et al. (2024); Ginting et al. (2024); Wang et al. (2025), the zero-shot paradigmKuang et al. (2024); Cai et al. (2025); Debnath et al. (2025); Yu et al. (2024); Guan et al. (2024); Wen et al. (2025); Zhou et al. (2023) can adapt to unseen environments without the need for additional training data. In practical scenarios, for example the open-world environments, this capability allows robots to navigate a completely new environment to retrieve specific items (e.g., a book) for patients. Furthermore, from the perspective of real-world deployment.

Early methods are primarily designed for single-floor environments. Initially, Gadre et al. (2023); Majumdar et al. (2022) leveraged Vision-Language Model (VLM) to extract semantic features from RGB observations, which were then integrated with exploration strategies to accomplish object-goal navigation tasks. However, these approaches proved highly inefficient. With the rapid advancement of Multimodal Large Language Model (LLM), recent works seek to leverage the powerful reasoning capabilities of LLM to improve navigation efficiency. Consequently, LLM are frequently employed

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

to select waypoints based on current RGB imageGu et al. (2026); Long et al. (2024); Zhao et al. (2024), value mapHan et al. (2024); Kuang et al. (2024). Nevertheless, these methods struggle to generalize to multi-floor navigation tasks. As the scale of the environment expands, the demands on the agent for complex geometric perception and path planning increase significantly.

Recent ZSON studies recognize this limitation and propose solutions for navigating across multiple floors. Spatially, methods such as Zhang et al. (2026); Zheng et al. (2026); Yang et al. (2025) utilize 3D Simultaneous Localization and Mapping (SLAM) to build maps and Zhang et al. (2025a); Yin et al. (2024); Loo et al. (2025) construct hierarchical topology maps from floors to rooms. To improve efficiency, approaches like Gong et al. (2026) prioritize searching nearest frontiers and rely on LLM for distant planning. Additionally, methods like Liu et al. (2025); Zhou et al. (2025) focus on finding the shortest exploration paths, method Cao et al. (2025) further decomposes the exploration strategy to emulate human cognition, it does so at the cost of making each decision step heavily dependent on an LLM, leading to substantial token consumption. Despite improving object-finding success rates by enhancing spatial representations and boosting exploration efficiency, these works overlook two critical issues. First, their frameworks lack a well-defined strategy representation and strategy transitions; they often overemphasize exploration, leaving robots less robust during navigation and unable to cope effectively with abnormal situations. For example, the robot may fail to locate the entrance to a staircase or get stuck while attempting to reach a frontier point. Second, their exploration strategies fail to adequately adapt to environmental variations in exploration coverage, semantic cues, and structural information and so on, making it difficult to balance exploration, exploitation, and high-level planning.

To enable the robot to simulate human-like thinking and problem-solving approaches, we propose LERR-Nav. In summary, the contributions of our method are as follows: (1) We propose a Coordinated Exploration–Recovery–Reminiscing Strategy that explicitly accounts for the diverse situations a robot may encounter during object search, and seamlessly adapts to navigation tasks by dynamically switching among these strategies. (2) We propose a Dual-Mode Exploration strategy featuring Fast- and Slow-Thinking modes. The Fast-Thinking mode explicitly incorporates priors on the environment's semantic value and uncertainty estimation to guide efficient exploration, while the Slow-Thinking mode integrates commonsense knowledge with current observations to perform high-level planning. This strategy helps robots better balance exploration, exploitation, and higher-level reasoning during the task. (3) We achieve state-of-the-art zero-shot performance on Object Goal Navigation (OGN) benchmarks, improving SR by 6.9% and SPL by 2.1% in HM3DRamakrishnan et al. (2021), and SR by 3.4% in MP3DChang et al. (2017). Ablation studies and comparative experiments further demonstrate the effectiveness of the strategy and modules.

2 Related Works

Object Navigation. Object navigation refers to the task of guiding a robot to search for a given target object within an unknown environment. The dominant strategies can generally be divided into two primary categories: the first group includes training-dependent methods, such as reinforcement learning and imitation learning, which necessitate extensive training using task-specific datasetsRamakrishnan et al. (2022); Ramrakhya et al. (2023); Fukushima et al. (2022); Maksymets et al. (2021); Ye & Yang (2021). The second category includes zero-shot methodsKuang et al. (2024); Cai et al. (2025); Debnath et al. (2025); Yu et al. (2024); Guan et al. (2024); Wen et al. (2025); Zhou et al. (2023), which leverage pre-trained models like VLMLi et al. (2023); Radford et al. (2021) or LLM to enable the robot to exhibit strong adaptability in unseen environments. These methods commonly utilize LLM as decision-maker. However, they often neglect to consider the robot's strategy. Therefore, our zero-shot approach incorporates an LLM role-playing strategy, allowing the robot to take different actions depending on its current strategy, thus improving the decision-making process in dynamic and varied contexts.

Exploration Strategy for Object Navigation. Under a strict step budget, balancing exploration and exploitation in complex multi-floor environments is crucial for robots. Most early methods rely heavily on greedy strategies. SSR-ZSMeng et al. (2025) constructs semantic maps and utilizes LLM to identify and select target-relevant frontiers; VLFMYokoyama et al. (2024) employs VLM to build value maps linking frontiers to the target; and SGNavYin et al. (2024) leverages 3D scene graphs

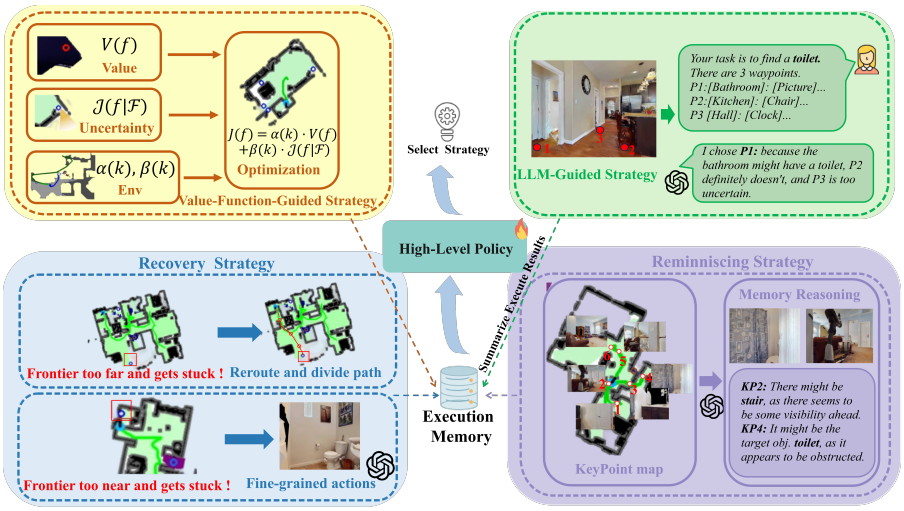

Figure 1: **LERR-Nav pipeline:** Conditioned on the target instruction and execution memory, a learned high-level policy invokes one of four complementary strategies: value-function-guided exploration, LLM-guided exploration, recovery, or reminiscing. The selected strategy interacts with the environment until termination, after which its textual execution summary and RGB–map records are used to update the execution memory. The updated memory then informs the next high-level strategy invocation, forming a closed-loop navigation process.

to prompt LLM for spatial reasoning. However, strategies that depend so heavily on semantic or LLM-derived scoring often suffer from inefficient exploration. Subsequent approaches have sought to address this limitation. ApexNavZhang et al. (2025b) introduces an adaptive exploration method that dynamically switches operational modes based on the strength of semantic cues. However, the lack of continuous LLM reasoning during the exploration process often leads the robot to overlook areas where the target is hidden, particularly when the semantic cues are weak. Similarly, while ASCGong et al. (2026) proposes a coarse-to-fine exploration strategy that combines nearest-frontier exploration with LLM guidance, it fails to adapt dynamically to environmental variations, causing the robot to simply navigate to the nearest frontier most of the time. In short, we should use the LLM in moderation and integrate it into the exploration strategy. So, we propose an environment-adaptive exploration strategy integrated with a fast-and-slow thinking mechanism.

3 Method

3.1 Task Definition

We formulate Object Navigation task, the robot needs to locate target objects in unfamiliar environments it has never encountered before. The target object c is provided as a free-form linguistic description. At each discrete time step t, the agent perceives the environment through an RGB-D sensor. Concurrently, the agent tracks its relative strategy $s_t = (x_t, r_t)$ via on-board odometry, representing its position $x_t \in \mathbb{R}^3$ and orientation $r_t \in SO(3)$. The agent's decision-making framework must select a discrete action a_t from the action space $\mathcal{A}$: = {MOVE_FORWARD, TURN_LEFT, TURN_RIGHT, LOOK_UP, LOOK_DOWN, STOP}. The locomotion is executed in fixed increments (e.g., 0.25m for translation and 30° for rotation). An episode is deemed a success if the agent issues a STOP command within a proximity threshold of 0.1m from the target object, without exceeding a maximum budget of 500 steps.

3.2 Framework Overview

As illustrated in Fig. 1, LERR-Nav is an object navigation framework that coordinates exploration, recovery, and reminiscing through a learned high-level policy. During navigation, the robot maintains different maps to support reasoning and preserve informative observations (Sec. 3.3). Building

on these representations, the framework provides four complementary navigation strategies, as described in Sec. 3.4. The execution memory (Sec. 3.5) offer agent Long-Term Memory. A learned high-level policy coordinates strategy execution and is trained through two-stage SFT (Sec. 3.6).

3.3 MAP REPRESENTATION

Visibility Map $\mathcal{M}_{\text{vis}}$. We maintain a 2D occupancy grid map by projecting depth observations into 3D point cloud and subsequently performing Bird's Eye View projection. Based on the boundaries between explored free space and unexplored regions, we extract a set of candidate frontiers $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$. These frontiers are categorized into two subsets: intra-floor frontiers $\mathcal{F}_{\text{floor}}$ and staircase frontiers $\mathcal{F}_{\text{stair}}$, where staircases are treated as traversable transitions. The specific geometry-based staircase detection and frontier generation logic follow the Gong et al. (2026).

Exploration Value Map $\mathcal{M}_{\text{val}}$. We define a value function $V(f)$ for each frontier $f \in \mathcal{F}_{\text{floor}}$ as follows: $V(f) = \alpha \cdot S_{\text{sem}}(f) + \beta \cdot S_{\text{dist}}(f)$, where the $S_{\text{sem}}(f)$ evaluates the semantic relevance of a frontier. Using the BLIP-2Li et al. (2023), we can calculate the score by measuring the cosine similarity between an RGB observation I_f at the frontier and a set of context-aware text prompts T (e.g., bathtub or sink when the target is toilet) generated by the LLM's common-sense reasoning. $S_{\text{dist}}(f)$ favors proximal exploration by employing a linear decay function based on the geodesic distance $d(r, f)$ between the robot's current position r and the frontier f: $S_{\text{dist}}(f) = \max\left(0, 1 - \frac{d(r,f)}{d_{\text{max}}}\right)$, where d_{max} represents a predefined normalization constant.

KeyPoint Map $\mathcal{M}_{\text{key}}$. Inspired by the keyframes of Orb-SLAM2Mur-Artal & Tardós (2017), the difference is that these keyframes are fixed onto a 2D map as keypoints. The criteria for selecting keypoints are as follows: all room entrances and frontiers with open visibility. Frontiers with open visibility can be quantified by the area of the robot's field of view (FOV) at that point, which is not obstructed by obstacles.

We independently maintain these three distinct map layers for each floor, resulting in a global environmental representation structured as a set of floor-specific tuples: $\mathcal{M} = \{\langle F_i, \mathcal{M}_{\text{vis}(i)}, \mathcal{M}_{\text{val}(i)}, \mathcal{M}_{\text{key}(i)} \rangle\}_{i=1}^{n}$, where F_i denotes the i-th floor index.

3.4 COMPLEMENTARY NAVIGATION STRATEGIES

3.4.1 VALUE-FUNCTION-GUIDED EXPLORATION STRATEGY

We formulate the frontier selection as an optimization problem that balances current values priors with the expected reduction in environmental uncertainty. Let $\mathcal{Z}_k$ be the observations up to step k, and $\rho_{k|k} = p(\Theta|\mathcal{Z}_k)$ denote the current belief of the target Θ. When the robot evaluates a candidate frontier $f \in \mathcal{F}_{\text{floor}}$, it anticipates a future belief $\hat{\rho}_{k+1|k} = p(\Theta|\mathcal{Z}_k, \hat{z}_{k+1}) = \mathcal{I}(f|\mathcal{F})$, where $\hat{z}_{k+1}$ is a virtual estimate within the FOV at f: $J(f) = \alpha(k) \cdot V(f) + \beta(k) \cdot \mathcal{I}(f|\mathcal{F})$, where $\alpha(k)$ and $\beta(k)$ are factors that change with the environment, and will be explained later, $V(f)$ is the prior value, and $\mathcal{I}(f|\mathcal{F})$ quantifies the expected uncertainty reduction. To compute this, we first project the FOV at f to obtain the coverage area S_1. For the set of semantic scores $\mathcal{C}$ at the boundaries of S_1, we apply a Gaussian kernel $G(\cdot)$ to estimate the spatial information density $d(\mathbf{p}) = G(\mathcal{C})$.
To account for the redundancy and overlap between multiple frontiers, the total estimated uncertainty f_{1_u} for a frontier is adjusted by considering the mutual information between overlapping coverage areas. Specifically, the integrated uncertainty at f is formulated as: $\mathcal{I}(f|\mathcal{F}) = \int_{S_1(f)} d(\mathbf{p}) d\mathbf{p} + \sum_{f_j \in \mathcal{F}, j \neq f} \text{Area}\left(S_1(f) \cap S_1(f_j)\right)$, where $\text{Area}(S_1(f) \cap S_1(f_j))$ represents the spatial overlap between the estimated coverage of frontier f and its neighbors f_j, more details can be found in the Appendix A.2.

$\alpha(k)$ and $\beta(k)$ are adapted based on the Exploration Reward (ER), which characterizes the robot's exploration progress and the remaining environmental potential at time k. The exploration reward $\text{ER}(k)$ is defined as a multi-factor metric:

$$\text{ER}(k) = \sigma_1 \cdot \frac{|U(k)|}{|E|} + \sigma_2 \cdot \frac{N_{\text{frontier}}(k)}{N_{\text{total}}} + \sigma_3 \cdot \left(1 - \frac{k}{K_{\text{max}}}\right), \tag{1}$$

where $|U(k)|/|E|$ represents the ratio of the unexplored area to the total environment size at step k, $N_{\text{frontier}}(k)/N_{\text{total}}$ denotes the current density of discovered frontiers, and the final term introduces a temporal decay to prioritize rapid exploration in the early stages of the mission. The strategy-dependent coefficients $\alpha(k)$ and $\beta(k)$ are updated as follows: $\beta(k) = \beta_{\text{max}} \cdot \text{ER}(k), \alpha(k) = \alpha_{\text{min}} \cdot (1 - \text{ER}(k))$.

3.4.2 LLM-GUIDED EXPLORATION STRATEGY

A local room and object topology map is first constructed to enhance the accuracy of LLM predictions. First, the current RGB image is divided based on the regions of doorframes. Then, each RGB image is processed using RAMZhang et al. (2024) to classify the object categories, and Place365Zhou et al. (2017) is used to identify the room types, ultimately generating descriptions for different rooms. Following this, prompts are designed to allow LLM to evaluate, in conjunction with the visibility map and the current RGB image, whether a frontier point is suitable for exploration.

3.4.3 RECOVERY STRATEGY

As shown in Fig. 5(b), when dealing with distant frontier point, the A* algorithm is first applied on the 2D visibility map to compute a collision-free path from the current position to the target. By employing an interval sampling method, several intermediate points are generated along the path. The target, therefore, becomes to traverse all the intermediate points, and after testing, the robot successfully resumes normal movement.

As shown in Fig. 5(e), when handling nearby frontier points, LLM is used to generate fine-grained action commands by integrating the RGB image and the local map.

3.4.4 REMINISCING STRATEGY

Following the completion of single-floor exploration, a two-stage reminiscing process is initiated to ensure thoroughness and facilitate multi-floor transitions. This process relies on $\mathcal{M}_{\text{key}}$, where each point preserves a corresponding RGB observation on the global map. This integration allows the LLM to perform joint spatial-semantic reasoning by synthesizing the global geometric structure with localized visual data. As shown in Fig. 5, the reminiscing logic is executed in two hierarchical stages:

Target Verification. The LLM first reviews the aggregate keypoint collection to identify potential nodes where the target object may have been overlooked. Upon selecting candidate keypoints, the system extracts the RGB images for a secondary, fine-grained confirmation. If verified, the robot navigates to the specific coordinates and executes a terminal approach to the target.

Look for Staircase. If no potential target locations are identified, the LLM shifts focus to detecting potential staircase entrances to enable vertical mobility. The robot employs the PointNav Wijmans et al. (2022) policy to navigate toward the keypoint with the highest probability of containing a staircase. Final alignment with the staircase entrance is achieved through LLM-guided fine-grained action.

3.5 EXECUTION MEMORY WITH TIME SPARSIFICATION.

We maintain an execution memory that combines textual summaries of all completed strategy executions with a sliding window of RGB observations and map snapshots. Let t_k denote the environment timestep of the k-th high-level decision. Execution segment j begins when a strategy is invoked at t_j and ends when it terminates at t_{j+1}. For each completed segment, we record a textual summary e_j and a temporally ordered sequence of RGB–map pairs: $\mathbf{V}_j = \big((t, I_t, M_t)\big)_{t \in \mathcal{T}_j}$, where $\mathcal{T}_j$ contains the recorded timesteps within segment j, I_t is the RGB observation, and M_t is the corresponding rendered navigation map. The summary e_j describes the executed strategy, its observed outcome, the start and end locations, and the rooms or landmarks encountered during execution.

At decision k, the visual window contains at most the W most recent completed execution segments: $\mathcal{W}_k = \{j \mid \max(1, k - W) \leq j \leq k - 1\}$. We apply time sparsification to the visual records within this window. Specifically, we measure the age of segment j by its distance from the current decision,

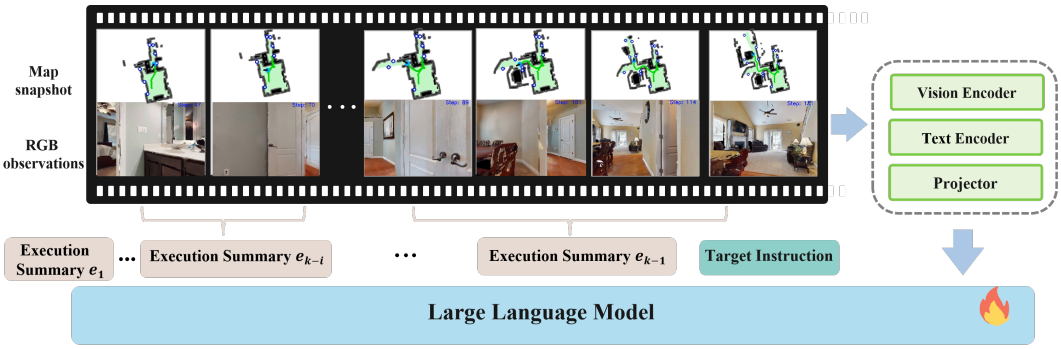

Figure 2: **Pipeline of the high-level coordination policy.** At the k-th decision, execution memory offers textual summaries of all completed strategy executions $(e_1, \ldots, e_{k-1})$, and historical RGB observations and map snapshots to the agent.

$a = k - j$, and assign it a non-increasing visual retention budget $b(a)$: $b(a+1) \le b(a)$, $\left|\widetilde{\mathbf{V}}_j^{(k)}\right| \le \min\left\{|\mathbf{V}_j|, b(k-j)\right\}$, where $\widetilde{\mathbf{V}}_j^{(k)}$ is the retained visual subsequence of segment j at decision k. The retained RGB observations and map snapshots remain paired, with their original timestamps and chronological order preserved. Time sparsification therefore reduces the number of historical snapshots rather than their spatial resolution.

The resulting execution memory is $\mathcal{E}_k = (\mathcal{H}_k, \mathcal{V}_k), \mathcal{H}_k = (e_1, \ldots, e_{k-1}), \mathcal{V}_k = \left(\left(j, \widetilde{\mathbf{V}}_j^{(k)}\right)\right)_{j \in \mathcal{W}_k}$. As the window advances, visual records outside $\mathcal{W}_k$ are removed from the coordinator's visual memory, while their textual summaries remain in $\mathcal{H}_k$.

3.6 High-Level Policy

We instantiate the high-level coordination policy with Qwen2.5-VL-7B-Instruct Bai et al. (2025). Rather than predicting low-level action, the agent determines which navigation strategy to invoke after the active strategy terminates, as shown in Fig. 2.

Stage I: Map-Centric Spatial and Trajectory Understanding. The first stage adapts the model to the map representation used by the navigation system through map-centric visual question answering (VQA). We construct a task-specific VQA dataset from rendered navigation maps and their associated execution records: $\mathcal{D}_{\text{map}} = \{(\mathbf{M}, q, y)\}$, where $\mathbf{M} = (M_{\tau_1}, \ldots, M_{\tau_L})$ is a temporally ordered sequence of map snapshots, q is a map-related question, and y is the reference answer.

The questions cover three complementary aspects: *spatial configuration*, including the number and relative distribution of marked frontier candidates; *map evolution*, including changes in observed regions across snapshots; and *trajectory interpretation*, including movement direction, displacement, and repeated traversal of previously visited areas.

We optimize a unified answer-generation objective: $\mathcal{L}_{\text{map}}(\theta) = -\mathbb{E}_{\mathcal{D}_{\text{map}}}\Big[|y|^{-1} \sum_{j=1}^{|y|} \log P_\theta(y_j \mid \mathbf{M}, q, y_{<j})\Big]$. where the expectation is taken over $(\mathbf{M}, q, y) \sim \mathcal{D}_{\text{map}}$, θ denotes the trainable model parameters and $y_{<j}$ denotes the preceding answer tokens. Details of dataset construction, question templates, answer annotation rules, and representative VQA examples are provided in Appendix **??**.

Stage II: Supervised Strategy Coordination. Starting from the Stage I checkpoint, we further adapt the model to coordinate navigation strategies through supervised demonstrations. The training inputs follow the same organization as policy inference.

We collect navigation rollouts under the high-level guidance of GPT-6 Astra OpenAI (2026), which determines which navigation strategy to invoke at each decision boundary. Rollouts that achieve task success are retained and subjected to manual quality screening. An LLM-based annotation tool then

organizes the retained demonstrations into strategy-execution segments, with segment boundaries aligned with the recorded strategy invocations and terminations.

For each recorded decision k, we reconstruct the execution memory using only the preceding completed segments. The input is $\mathbf{x}_k = \mathrm{Pack}(g, \mathcal{H}_k, \mathcal{V}_k)$, where $\mathcal{H}_k$ contains the preceding textual execution summaries and $\mathcal{V}_k$ is constructed using the same sliding-window and time-sparsification mechanism as inference. Different decision points provide execution histories of varying lengths.

The reference strategy s_k^{ref} is the strategy invoked by the teacher at decision k. Let y_k denote its textual encoding. The resulting supervised dataset is $\mathcal{D}_{\mathrm{coord}} = \{(\mathbf{x}_k, y_k)\}$. Observations and execution outcomes resulting from s_k^{ref} are excluded from $\mathbf{x}_k$. Likewise, each historical summary is restricted to information available when its corresponding execution ended.

We optimize the reference strategy outputs using $\mathcal{L}_{\mathrm{coord}}(\theta) = -\mathbb{E}_{\mathcal{D}_{\mathrm{coord}}}\Big[|y|^{-1}\sum_{j=1}^{|y|}\log P_\theta(y_j \mid \mathbf{x}, y_{<j})\Big]$, where the expectation is taken over $(\mathbf{x}, y) \sim \mathcal{D}_{\mathrm{coord}}$.

This stage trains the agent to associate execution history with subsequent strategy invocations. Dataset statistics are reported in the experimental setup.

4 Experimental Results

4.1 Experimental Setup

Datasets. We utilize the HM3DRamakrishnan et al. (2021) and MP3DChang et al. (2017) datasets under the standard Object Goal Navigation (OGN) benchmark. The HM3D validation split consists of 2000 episodes across 20 scenes, encompassing 6 object categories. Similarly, the MP3D validation set contains 2195 episodes spanning 11 scenes and 21 categories. Notably, a statistical analysis of these environments reveals a strong prevalence of multi-story layouts: approximately 65% of the scenes in HM3D and 73% in MP3D feature multiple floors.

Training Datasets. For SFT Stage I, we construct a map-centric VQA dataset using 21,029 trajectories from VLN-CE Krantz et al. (2020) and the HM3D and MP3D environments. For each trajectory, we sample an average of 20 map subsequences, each spanning 3–8 consecutive navigation steps, and construct corresponding question–answer pairs. This yields 420,780 VQA training samples. For SFT Stage II, we use GPT-4o as the teacher model to collect navigation demonstrations in HM3D and MP3D. We perform approximately 100 rollout attempts for each navigation segment, obtaining an average of 32.3 valid successful rollouts and approximately 135,498 valid trajectories in total. An automated annotation tool then partitions these trajectories into strategy-execution segments. Each trajectory yields an average of 8 supervised decision samples, resulting in 1,083,984 training samples for strategy coordination.

Evaluation Metrics. To comprehensively evaluate all methods, we employ two standard metrics: Success Rate (SR) and Success weighted by inverse Path Length (SPL). While SR captures the binary outcome of task completion, SPL quantifies the agent's navigational efficiency by comparing its trajectory against the shortest optimal path, assigning a score of zero to unsuccessful episodes.

Experimental Details. We employ RAMZhang et al. (2024) to generate category prompts, while Grounding-DINOLiu et al. (2024) is utilized for open-vocabulary detection. Object contours are extracted via SAMKirillov et al. (2023) segmentation. To facilitate boundary reasoning, we leverage BLIP-2Li et al. (2023) for semantic similarity representation, Places365Zhou et al. (2017) for scene classification, and GPT-4o for LLM-based inference. For all experiments, the system was deployed using two NVIDIA RTX 4090 GPUs, with one GPU dedicated to running the fine-tuned Qwen2.5-VL-7B-Instruct model. We use the AdamW optimizer with a learning rate of 2×10^{-5} and keep all parameters of the vision encoder frozen throughout both SFT stages. Each stage is trained for 1,000 optimizer update steps. Training across both stages takes approximately 150 hours on eight NVIDIA A100 GPUs, corresponding to approximately 1,200 GPU-hours.

Table 1: Comparison of LERR-Nav and other methods on the HM3D and MP3D datasets.

Method	Venue	HM3D		MP3D	
		SR↑	SPL↑	SR↑	SPL↑
ZSON Majumdar et al. (2022)	NeurIPS'22	25.5	12.6	15.3	4.8
ESC Zhou et al. (2023)	ICML'23	39.2	22.3	28.7	14.2
L3MVN Yu et al. (2023)	IROS'23	50.4	23.1	34.9	14.5
PSL Sun et al. (2024)	ECCV'24	42.4	19.2	18.9	6.9
VLFM Yokoyama et al. (2024)	ICRA'24	52.5	30.4	36.4	17.5
OpenFMNav Kuang et al. (2024)	NAACL'24	52.5	24.1	37.2	15.7
SG-NAV Yin et al. (2024)	NeurIPS'24	54.0	24.9	40.2	16.0
BeliefMapNav Zhou et al. (2025)	NeurIPS'25	61.4	30.6	37.3	17.6
ASCENT Gong et al. (2026)	RAL'26	65.4	33.5	44.5	15.5
LERR-Nav	**ours**	**72.3**	**35.6**	**47.9**	**18.7**

4.2 Comparison with SOTA

As shown in Tab. 1, we evaluate our proposed LERR-Nav methods on the HM3D and MP3D datasets. Our method consistently outperforms all existing baselines, establishing a new SOTA performance across both benchmarks in zero-shot settings.

Specifically, on the HM3D dataset, LERR-Nav achieves an impressive 72.3% in SR and 35.6% in SPL. This represents a significant absolute improvement of $+6.9\%$ in SR and $+2.1\%$ in SPL over the previously best-performing method, ASCENT. Furthermore, on the MP3D dataset, our approach reaches 47.9% SR and 18.7% SPL. Compared to the strong baseline ASCENT, we observe a $+3.4\%$ increase in SR. Additionally, LERR-Nav surpasses BeliefMapNav, the prior best in path efficiency on MP3D, yielding a $+1.1\%$ gain in SPL.

To validate cross-floor capabilities, we divide the HM3D episodes into intra-floor (The target object and the robot are initialized on the same floor) and inter-floor (The target object and the robot are initialized on the different floor) scenarios. As shown in Tab. 2, LERR-Nav's overall enhancement is predominantly driven by its inter-floor breakthrough. Alongside steady intra-floor improvements, our method achieves remarkable cross-level gains, outperforming the strongest baseline, ASCENT, by $+29.2\%$ in SR and $+12.4\%$ in SPL.

Table 2: Performance comparison between intra and inter-floor navigation scenarios.

Method	**Intra-Floor Scenarios**		**Inter-Floor Scenarios**		**All Episodes**	
	SR↑	**SPL↑**	**SR↑**	**SPL↑**	**SR↑**	**SPL↑**
VLFMYokoyama et al. (2024)	64.6	37.3	0.4	0.1	52.8	30.5
MFNPZhang et al. (2025a)	68.4	30.5	13.4	9.8	58.3	26.7
ASCENTGong et al. (2026)	72.6	37.7	33.3	14.9	65.4	33.5
LERR-Nav	**75.8**	**38.1**	**62.5**	**27.3**	**72.3**	**35.6**

4.3 Ablative Study

To comprehensively validate the efficacy of the proposed components in LERR-Nav, we conduct extensive ablation study. Additional ablation studies are provided in the Appendix A.4.2.

Impact of Our High-Level Policy. We evaluate different variants of the high-level policy on HM3D in Table 3. Stage-I map-centric SFT improves the base model from 43.3/19.6 to 58.1/27.3 in SR/SPL, outperforming GPT-5. Stage-II coordination SFT further increases performance to 72.3/35.6. Overall, the two-stage training yields gains of 29 SR and 16 SPL points over the base model, validating the benefits of both map-centric grounding and strategy-coordination training.

Table 3: Impact of the high-level coordination policy on HM3D.

High-Level Policy	SR↑	SPL↑
GPT-5	52.2	23.2
LERR-Nav (Base)	43.3	19.6
LERR-Nav (Stage-I SFT)	58.1	27.3
LERR-Nav (Stage-I+II SFT)	**72.3**	**35.6**

Table 4: Ablation of the navigation strategies on HM3D.

Variant	VF-E	LLM-E	Rec.	Rem.	SR↑	SPL↑
w/o VF-E	–	✓	✓	✓	63.5	32.6
w/o LLM-E	✓	–	✓	✓	51.8	30.2
w/o Rec.	✓	✓	–	✓	65.2	31.9
w/o Rem.	✓	✓	✓	–	66.3	32.5
Full	✓	✓	✓	✓	**72.3**	**35.6**

Impact of Navigation Strategies. Table 4 shows that removing any strategy degrades navigation performance. LLM-guided exploration contributes the most, as its removal reduces SR and SPL by 20.5 and 5.4 points, respectively. Removing value-function-guided exploration decreases SR by 8.8 points, while removing recovery causes a 7.1-point SR drop and a 3.7-point SPL drop. Reminiscing also provides consistent gains, improving SR and SPL by 6.0 and 3.1 points. These results demonstrate that the four strategies are complementary, with the complete model achieving the best performance of 72.3 SR and 35.6 SPL.

4.4 Qualitative Analysis

To further demonstrate the effectiveness of our LERR-Nav, we compare it against a strong recent baseline ASCENTGong et al. (2026) on the same navigation episode (See in Appendix A.4.3). Real-world deployment on the Unitree Go2 (Fig. 3) further demonstrates the effectiveness of our method in sim-to-real transfer.

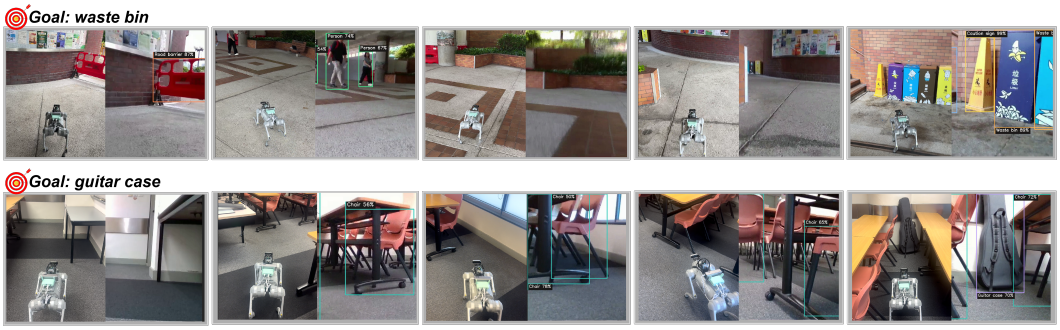

Figure 3: Real-world deployment of LERR-Nav on a Unitree Go2 quadruped robot.

5 Conclusion

We presented LERR-Nav, which learns to coordinate four complementary navigation strategies using execution memory and a high-level vision-language policy. A two-stage SFT curriculum progressively develops spatial understanding and strategy coordination. Experiments on HM3D and MP3D validate its effectiveness, while deployment on a Unitree Go2 demonstrates its potential for sim-to-real object navigation.

AI USE STRATEGYMENT

In this work, we used generative AI tools (e.g., large language models) for manuscript language editing, grammar refinement, and code debugging. We have not used generative AI tools for generating core research ideas, designing methodologies, or creating experimental data. We have thoroughly reviewed all AI-assisted text and code to ensure their accuracy, correctness, and originality. We take full responsibility for the final content of this work, including all text, claims, and artifacts.

REFERENCES

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. doi: 10.48550/arXiv.2502.13923. URL https://arxiv.org/abs/2502.13923.

Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*, 2020.

Yuxin Cai, Xiangkun He, Maonan Wang, Hongliang Guo, Wei-Yun Yau, and Chen Lv. Cl-cotnav: Closed-loop hierarchical chain-of-thought for zero-shot object-goal navigation with vision-language models. *arXiv preprint arXiv:2504.09000*, 2025.

Yihan Cao, Jiazhao Zhang, Zhinan Yu, Shuzhen Liu, Zheng Qin, Qin Zou, Bo Du, and Kai Xu. Cognav: Cognitive process modeling for object goal navigation with llms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9550–9560, 2025.

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.

Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Shizhe Chen, Thomas Chabal, Ivan Laptev, and Cordelia Schmid. Object goal navigation with recursive implicit maps. *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7089–7096, 2023.

Arnab Debnath, Gregory J Stein, and Jana Kosecka. Semnav: A model-based planner for zero-shot object goal navigation using vision-foundation models. *arXiv preprint arXiv:2506.03516*, 2025.

Rui Fukushima, Kei Ota, Asako Kanezaki, Yoko Sasaki, and Yusuke Yoshiyasu. Object memory transformer for object goal navigation. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 11288–11294. IEEE, 2022.

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 (CVPR)*, pp. 23171–23181, 2023.

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. In *Proc. of Robotics: Science and Systems (RSS)*, 2024.

Zeying Gong, Rong Li, Tianshuai Hu, Ronghe Qiu, Lingdong Kong, Lingfeng Zhang, Guoyang Zhao, Yiyi Ding, and Junwei Liang. Stairway to success: An online floor-aware zero-shot object-goal navigation framework via llm-driven coarse-to-fine exploration. *IEEE Robotics and Automation Letters*, 2026.

Zhengyi Gu, Yuheng Zhou, Yuquan Xue, and Hanzhang Wang. Curiosity-driven zero-shot object navigation with vision-language models. *IEEE Robotics and Automation Letters*, 2026.

Tianrui Guan, Yurou Yang, Harry Cheng, Muyuan Lin, Richard Kim, Rajasimman Madhivanan, Arnie Sen, and Dinesh Manocha. Loc-zson: Language-driven object-centric zero-shot object retrieval and navigation. *arXiv preprint arXiv:2405.05363*, 2024.

Peilong Han, Min Zhang, Hongyao Tang, YAN ZHENG, et al. Hule-nav: Human-like exploration for zero-shot object navigation via vision-language models. In *NeurIPS 2024 Workshop on Behavioral Machine Learning*, 2024.

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 (ICCV)*, pp. 4015–4026, 2023.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision (ECCV)*, 2020.

Yuxuan Kuang, Hai Lin, and Meng Jiang. Openfmnav: Towards open-set zero-shot object navigation via vision-language foundation models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 338–351, 2024.

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*, pp. 12888–12900. PMLR, 2022.

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*, pp. 19730–19742. PMLR, 2023.

Ruimeng Liu, Xinhang Xu, Shenghai Yuan, and Lihua Xie. Handle object navigation as weighted traveling repairman problem. *arXiv preprint arXiv:2503.06937*, 2025.

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 (ECCV)*, pp. 38–55. Springer, 2024.

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.

Joel Loo, Zhanxin Wu, and David Hsu. Open scene graphs for open-world object-goal navigation. *The International Journal of Robotics Research*, 2025. URL https://doi.org/10.1177/02783649251369549.

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 (NeurIPS)*, 35:32340–32352, 2022.

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 (ICCV)*, pp. 15374–15383, 2021.

Xiangyi Meng, Delun Li, Zihao Mao, Yi Yang, and Wenjie Song. Ssr-zson: Zero-shot object navigation via spatial-semantic relations within a hierarchical exploration framework. *arXiv preprint arXiv:2509.24763*, 2025.

Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics (TRO)*, 33(5):1255–1262, 2017.

OpenAI. GPT-6 Astra Model. https://developers.openai.com/api/docs/models/gpt-6-astra, 2026. OpenAI API documentation. Accessed: September 26, 2026.

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 (ICML)*, pp. 8748–8763. PMLR, 2021.

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.

Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18890–18900, 2022.

Ram Ramrakhya, Dhruv Batra, Erik Wijmans, and Abhishek Das. Pirlnav: Pretraining with imitation and rl finetuning for objectnav. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17896–17906, 2023.

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 (ECCV)*, pp. 161–178. Springer, 2024.

Hongze Wang, Boyang Sun, Jiaxu Xing, Fan Yang, Marco Hutter, Dhruv Shah, Davide Scaramuzza, and Marc Pollefeys. What matters in rl-based methods for object-goal navigation? an empirical study and a unified framework. *arXiv preprint arXiv:2510.01830*, 2025.

Justin Wasserman, Girish Chowdhary, Abhinav Gupta, and Unnat Jain. Exploitation-Guided Exploration for Semantic Embodied Navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2901–2908, 2024.

Congcong Wen, Yisiyuan Huang, Hao Huang, Yanjia Huang, Shuaihang Yuan, Yu Hao, Hui Lin, Yu-Shen Liu, and Yi Fang. Zero-shot object navigation with vision-language models reasoning. In *International Conference on Pattern Recognition (ICPR)*, pp. 389–404. Springer, 2025.

Erik Wijmans, Irfan A. Essa, and Dhruv Batra. VER: Scaling on-policy RL leads to the emergence of navigation in embodied rearrangement. In *Advances in Neural Information Processing Systems*, volume 35, pp. 7727–7740, 2022.

Yuncong Yang, Han Yang, Jiachen Zhou, Peihao Chen, Hongxin Zhang, Yilun Du, and Chuang Gan. 3d-mem: 3d scene memory for embodied exploration and reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17294–17303, 2025.

Xin Ye and Yezhou Yang. Efficient robotic object search via hiem: Hierarchical policy learning with intrinsic-extrinsic modeling. *IEEE Robotics and Automation Letters*, 6(3):4425–4432, 2021.

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 (NeurIPS)*, 37:5285–5307, 2024.

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)*, pp. 42–48, 2024.

Hwiyeon Yoo, Yunho Choi, Jeongho Park, and Songhwai Oh. Commonsense-aware object value graph for object goal navigation. *IEEE Robotics and Automation Letters*, 9(5):4423–4430, 2024.

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)*, pp. 3554–3560. IEEE, 2023.

Bangguo Yu, Yuzhen Liu, Lei Han, Hamidreza Kasaei, Tingguang Li, and Ming Cao. Vln-game: Vision-language equilibrium search for zero-shot semantic navigation. *arXiv preprint arXiv:2411.11609*, 2024.

Jiwen Zhang, Zejun Li, Siyuan Wang, Xiangyu Shi, Zhongyu Wei, and Qi Wu. Spatialnav: Leveraging spatial scene graphs for zero-shot vision-and-language navigation. *arXiv preprint arXiv:2601.06806*, 2026.

Lingfeng Zhang, Hao Wang, Erjia Xiao, Xinyao Zhang, Qiang Zhang, Zixuan Jiang, and Renjing Xu. Multi-floor zero-shot object navigation policy. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6416–6422. IEEE, 2025a.

Mingjie Zhang, Yuheng Du, Chengkai Wu, Jinni Zhou, Zhenchao Qi, Jun Ma, and Boyu Zhou. Apexnav: An adaptive exploration strategy for zero-shot object navigation with target-centric semantic fusion. *IEEE Robotics and Automation Letters*, 2025b.

Youcai Zhang, Xinyu Huang, Jinyu Ma, Zhaoyang Li, Zhaochuan Luo, Yanchun Xie, Yuzhuo Qin, Tong Luo, Yaqian Li, Shilong Liu, et al. Recognize anything: A strong image tagging model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1724–1732, 2024.

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.

Wancai Zheng, Hao Chen, Xianlong Lu, Linlin Ou, and Xinyi Yu. 3dgsnav: Enhancing vision-language model reasoning for object navigation via active 3d gaussian splatting. *arXiv preprint arXiv:2602.12159*, 2026.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(6):1452–1464, 2017.

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 (ICML)*, pp. 42829–42842. PMLR, 2023.

Zibo Zhou, Yue Hu, Lingkai Zhang, Zonglin Li, and Siheng Chen. Beliefmapnav: 3d voxel-based belief map for zero-shot object navigation. *arXiv preprint arXiv:2506.06487*, 2025.

A APPENDIX

A.1 EXPLORATION VALUE MAP DETAILS

We define a value function $V(f)$ for each frontier $f \in \mathcal{F}_{\text{floor}}$ as follows:

$$V(f) = \alpha \cdot S_{\text{sem}}(f) + \beta \cdot S_{\text{dist}}(f), \tag{2}$$

The semantic relevance score of a frontier f, denoted as $S_{\text{sem}}(f)$, can be computed using BLIP-2 Li et al. (2023) by measuring the cosine similarity between the RGB observation I_f at the frontier and a set of context-aware text prompts T (e.g., bathtub or sink when the target is toilet) generated by the LLM's common-sense reasoning:

$$S_{\text{sem}}(f) = \max_{t \in T} \cos_\text{sim}\big(\phi(I_f), \psi(t)\big), \tag{3}$$

where $\phi(\cdot)$ and $\psi(\cdot)$ denote the visual and textual encoders of BLIP-2, respectively, and $\cos_\text{sim}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \, \|\mathbf{b}\|}$ is the cosine similarity between feature vectors.

A.2 VALUE-FUNCTION-GUIDED EXPLORATION STRATEGY DETAILS

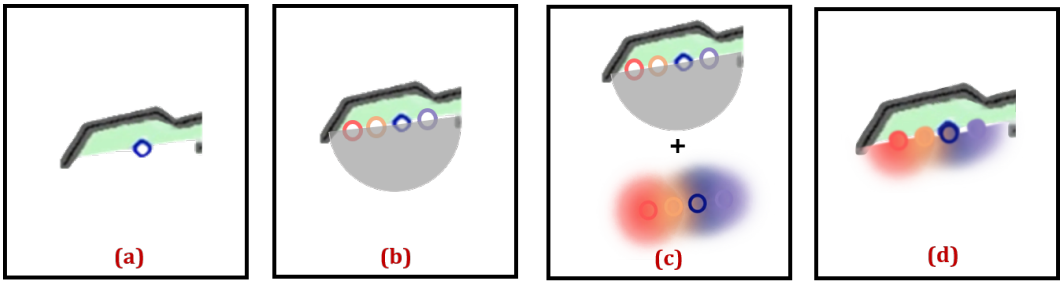

Figure 4: The uncertainty computation process. (a) Initial frontier point at the exploration boundary. (b) FOV projection defining the observable area S_1 and uniformly sampled boundary points. (c) Fusion process with the Gaussian kernel function and the assigned value scores for each sampled boundary point. (d) Final Estimated Uncertainty, restricted to the region within S_1.

The computation of uncertainty is illustrated in a visual manner. As shown in Fig. 4(a), a frontier point emerges at the boundary between explored and unexplored regions. Subsequently, as illustrated in Fig. 4(b), a field-of-view (FOV) rotation projection is applied to the frontier point, resulting in the maximum observable area S_1 associated with this point. For the boundary region L, additional boundary points are sampled uniformly. As depicted in Fig. 4(c), each boundary point is assigned a value score, represented with different colors, then, a Gaussian kernel function is applied to each boundary point. The scores are accumulated for points falling within the region S_1, while the scores for points outside S_1 are set to zero. Fig. 4(d) demonstrates the final result. It should also be noted that, the Gaussian kernel function is defined as:

$$G(x) = \sum_i v_i \exp\left(- \frac{\|x - x_i\|^2}{2\sigma^2} \right), \tag{4}$$

where x_i denotes the location of the i-th boundary point (serving as the center of the Gaussian), v_i is the value score of that boundary point, x is a location within the region S_1, and σ controls the spread of the Gaussian.

Finally, the pseudo-code of Fast-Thinking is shown in Algorithm 1.

A.3 LLM PROMPTS

In this section, we will provide detailed LLM prompts. These prompts are specifically designed around the robot's three strategys: Exploration strategy, Recovery strategy, and Reminiscing strategy.

Algorithm 1 Fast-Thinking Mode for Frontier Selection

Require: Observations $\mathcal{Z}_k$ up to step k, Set of candidate floor frontiers $\mathcal{F}_{\text{floor}}$, Total environment size $|E|$, Max steps K_{max}, Total frontiers N_{total}, Gaussian spread parameter σ
Ensure: Optimal frontier f^* for the next step

Compute Exploration Reward $\text{ER}(k)$:
  $\text{ER}(k) \leftarrow \sigma_1 \cdot \frac{|U(k)|}{|E|} + \sigma_2 \cdot \frac{N_{\text{frontier}}(k)}{N_{\text{total}}} + \sigma_3 \cdot \left(1 - \frac{k}{K_{\text{max}}}\right)$
Update balancing factors:
  $\beta(k) \leftarrow \beta_{\text{max}} \cdot \text{ER}(k)$
  $\alpha(k) \leftarrow \alpha_{\text{min}} \cdot (1 - \text{ER}(k))$
Initialize maximum objective value $J_{\text{max}} \leftarrow -\infty$ and $f^* \leftarrow \emptyset$
for each candidate frontier $f \in \mathcal{F}_{\text{floor}}$ **do**
    Obtain prior value $V(f)$
    Apply FOV rotation projection to f to obtain maximum observable area $S_1(f)$
    Extract the boundary region L of $S_1(f)$
    Sample boundary points $\{x_i\}$ uniformly from L
    Assign a value score v_i to each sampled boundary point x_i
    Estimate spatial information density $d(\mathbf{p})$ for points $\mathbf{p} \in S_1(f)$:
      $d(\mathbf{p}) \leftarrow \sum_i v_i \exp\left(-\frac{\|\mathbf{p}-x_i\|^2}{2\sigma^2}\right)$
      Note: $d(\mathbf{p}) \leftarrow 0$ *for points* $\mathbf{p} \notin S_1(f)$
    Compute expected uncertainty reduction:
      $\mathcal{I}(f|\mathcal{F}) \leftarrow \int_{S_1(f)} d(\mathbf{p})d\mathbf{p} + \sum_{f_j \in \mathcal{F}, j \neq f} \text{Area}\left(S_1(f) \cap S_1(f_j)\right)$
    Evaluate objective function:
      $J(f) \leftarrow \alpha(k) \cdot V(f) + \beta(k) \cdot \mathcal{I}(f|\mathcal{F})$
    if $J(f) > J_{\text{max}}$ **then**
      $J_{\text{max}} \leftarrow J(f)$
      $f^* \leftarrow f$
    end if
end for
return f^*

A.3.1 Exploration strategy Prompts

Exploration strategy Prompts

Task: You are a Exploration Planner Assistant. Analyze the image and select the most likely waypoint where the target object (e.g., TV, toy, or other household object) may be located. Provide a clear reasoning for your choice, considering visual cues such as doors, furniture, and activity indicators.

Input: [RGB Image with Waypoint Marking]

Constraints:

- *Evaluate each waypoint marked in the image.*
- *Explicitly explain why the selected waypoint has the highest probability.*
- *Keep reasoning concise (2-3 sentences max).*
- *Do not suggest waypoints not marked in the image.*

Model Output: JSON List
[
"Selected waypoint": "P2",
"Reasoning": "P2 has a door leading to a room where children's activity is visible, making it the most likely location for the target object. P1 leads to a bathroom with minimal activity, and P3 is the kitchen area where children are unlikely to play."
]

A.3.2 RECOVERY STRATEGY PROMPTS

Recovery strategy Prompts

Task: You are the helper for fine-tuning the robot's movements. The robot has already reached the selected KeyPoint. Generate a sequence of fine-grained navigation actions to approach the target area (e.g., stairs or object) from this KeyPoint. Provide reasoning for each action based on visual cues and local map layout, considering obstacles, doorways, and alignment with the target.
Inputs:
1. Visual Input: [Single RGB Image at current KeyPoint]
2. Local Map: [Top-down view showing robot position, obstacles, and target area]

Constraints:

- *Select from the actions list {Move Forward, Turn Left, Turn Right, Look Up, Look Down, Stop}.*
- *Provide reasoning for every action in 1-2 sentences.*
- *Combine RGB observation and map information for decision-making.*
- *Ensure the robot avoids obstacles and aligns efficiently toward the stairs or target area.*

Model Output: JSON List
[
"Actions": "[Go Forward, Go Forward, Turn Right, Turn Right]",
"Reasoning": " S1: Go Forward – The current view faces the window with stairs to the right, moving forward leaves the window area and approaches the staircase base;
S2: Go Forward – Continue forward to position within the reachable staircase area;
S3: Turn Right – Adjust orientation toward the staircase entrance so steps begin to appear;
S4: Turn Right – Fine-tune the view, fully revealing the staircase and allowing direct approach to the entrance. "
]

A.3.3 REMINISCING STRATEGY PROMPTS

Reminiscing strategy Prompts (Before reaching the selected KeyPoint)

Task: You are a Reminiscing Assistant. Analyze a set of RGB images and a top-down map with waypoint annotations. Your goal is to identify: 1. Which image(s) may contain a forgotten or overlooked target object. 2. Which image(s) may indicate a potential but previously unobserved stair entrance.
For each identification, provide a concise reasoning based on visual cues (furniture, doors, objects) and map context (edges of rooms, hallway junctions, corners).

Inputs:
1. RGB Image Set: [Rgb Images 1–n]
2. KeyPoint Map: [Top-down view with Keypoints, robot trajectory, and explored areas]

Constraints:

- *Only select from the provided RGB images.*
- *Provide reasoning for each selection in 1-2 sentences.*
- *Use map context to infer unexplored or partially observed areas.*
- *Distinguish between probable target object locations and potential stair entrances.*

Model Output: JSON List
[
"Potential Target Object": "Image 2",
"Target Reasoning": "Image 2 shows a doorway to a bedroom with visible toys or activity, making it the most likely location for a forgotten target object.",
"Potential Stair Entrance": "Image 1",
"Stair Reasoning": "Image 1 is at the edge of the house and at a junction between floors, which is typical for a stair location. No other images indicate vertical access points."
]

Reminiscing strategy Prompts (After reaching the selected KeyPoint)

Task: You are the helper for fine-tuning the robot's movements. The robot has already reached the selected KeyPoint. Generate a sequence of fine-grained navigation actions to approach the target area (e.g., stairs or object) from this KeyPoint. Provide reasoning for each action based on visual cues and local map layout, considering obstacles, doorways, and alignment with the target.
Inputs:
1. Visual Input: [Single RGB Image at current KeyPoint]
2. Local Map: [Top-down view showing robot position, obstacles, and target area]

Constraints:

- *Select from the actions list {Move Forward, Turn Left, Turn Right, Look Up, Look Down, Stop}.*
- *Provide reasoning for every action in 1-2 sentences.*
- *Combine RGB observation and map information for decision-making.*

Table 5: Ablation of value-function-guided exploration on HM3D and MP3D, comparing semantic and uncertainty components and static versus dynamic weighting.

Exploration Strategy	HM3D		MP3D	
	SR↑	SPL↑	SR↑	SPL↑
(a) Semantic ($\alpha = 1, \beta = 0$)	67.2	29.3	41.1	14.8
(b) Uncertainty ($\alpha = 0, \beta = 1$)	66.8	31.1	41.8	15.5
(c) Semantic + Uncertainty ($\alpha = 0.5, \beta = 0.5$)	68.5	33.4	43.3	16.3
(d) Semantic + Uncertainty ($\alpha = 0.5, \beta = 0.5$)	69.8	34.2	45.4	17.1
(e) Semantic + Uncertainty (Dynamic α, β)	69.3	33.8	45.9	17.3
(f) Semantic + Uncertainty (Dynamic α, β)	**72.3**	**35.6**	**47.9**	**18.7**

- *Ensure the robot avoids obstacles and aligns efficiently toward the stairs or target area.*

Model Output: JSON List
[
"Actions": "[Go Forward, Go Forward, Turn Right, Turn Right]",
"Reasoning": " S1: Go Forward – The current view faces the window with stairs to the right, moving forward leaves the window area and approaches the staircase base;
S2: Go Forward – Continue forward to position within the reachable staircase area;
S3: Turn Right – Adjust orientation toward the staircase entrance so steps begin to appear;
S4: Turn Right – Fine-tune the view, fully revealing the staircase and allowing direct approach to the entrance. "
]

A.4 Experimental Details

A.4.1 Experimental Parameters.

We will provide some specific parameters for the initialization of the experiment. In adaptive exploration strategy, the fixed weights α and β in the value function $V(f)$ are initialized to 0.5, d_{max} in $S_{\text{dist}}(f)$ is initialized to 20, which is set to the maximum distance the robot can travel, $\alpha(k)$ and $\beta(k)$, which are factors in the optimization function $J(f)$, are initialized to 0.5 and vary with the step index k. σ_1, σ_2, and σ_3, which are coefficients in the exploration reward $ER(k)$, are initialized to 2, 1 and 1. K_{max} and N_{total} are set to 500 and 50. In strategy transition, we set N_{rec}, SN_{rec} and D_{rec} to 10, 5 and 0.5.

A.4.2 Experimental Ablative Study Details

Impact of Parameters in Value-Function-Guided Exploration Strategy. As shown in Table 5, combining semantic and uncertainty cues with equal weights (c) outperforms either component alone (a, b) on both datasets. Compared with semantic-only exploration, the combined configuration improves SR and SPL by 1.3 and 4.1 percentage points on HM3D, and by 2.2 and 1.5 points on MP3D, respectively. These results suggest that the two cues provide complementary information for exploration. Replacing equal static weights with dynamic weighting (c $\rightarrow$ e) further increases SR/SPL by 0.8/0.4 points on HM3D and 2.6/1.0 points on MP3D, indicating that adjusting their relative contributions benefits both navigation success and path efficiency. Configuration (f) achieves the highest SR and SPL on both benchmarks.

Effectiveness of VLM and LLM for Navigation Strategy. As shown in Tab. 6, using Gemini results in a baseline SR of 70.8% and SPL of 33.8%. The Qwen2.5-VL model performs slightly better, achieving an SR of 71.4% and an SPL of 34.5%. However, GPT-4o demonstrates the highest efficacy. This indicates that GPT-4o provides superior spatial reasoning and zero-shot generalization capabilities. As shown in Tab. 7, CLIPRadford et al. (2021) and BLIP-2Li et al. (2023) achieve comparable performance—yielding SRs of 72.1% and 72.3%, and SPLs of 35.5% and 35.6%, respectively—both of which outperform the BLIP baseline. Notably, BLIP2 maintains a slight edge over CLIP in zero-shot text-to-image retrieval accuracy.

Table 6: Effect of the LLM for Navigation Strategy on HM3D.

Model	SR↑	SPL↑
Gemini	70.8	33.8
Qwen2.5-VL	71.4	34.5
GPT-4o	**72.3**	**35.6**

Table 7: Effect of the VLM for Navigation Strategy on HM3D.

Model	SR↑	SPL↑
BlipLi et al. (2022)	70.9	35.1
ClipRadford et al. (2021)	72.1	35.5
Blip-2 Li et al. (2023)	**72.3**	**35.6**

A.4.3 Experimental Demo Details

Please refer to our supplementary video material for more details. The video is divided into third sections: the first section provides a full walkthrough of two complete navigation episodes using our LERR-Nav; the second section presents additional comparison experiments with ASCENTGong et al. (2026), highlighting the advantages of our approach in terms of exploration speed and robustness. The third stage demonstrates our method's ability to generalize to real-world environments.

As shown in Fig. 7, our LERR-Nav completes the task with fewer time steps, highlighting the efficiency of our exploration strategy. The underlying reason is as follows. Comparing Fig. 7(c) and Fig. 7(i), the left half of the environment is dominated by a corridor, whereas the right half mainly contains a bathroom and a dressing room. Our LERR-Nav chooses to explore the hall, while ASCENT explores the bathroom and dressing room, which results in a substantial waste of steps for ASCENT. ASCENT made this choice because its coarse-grained exploration selects the nearest frontier point, in other words, LLM reasoning is only triggered when no frontier points exist near the robot. In contrast, our LERR-Nav triggers LLM reasoning upon detecting a door. LLM infers that sofa is unlikely in dressing room or bathroom, thus guiding the robot to explore the hallway as shown in Fig. 7(d).

As shown in Fig. 8, our LERR-Nav enables the robot to recall and localize the staircase entrance, allowing it to continue and complete the cross-floor task, which demonstrates the importance of the reminiscing strategy. As illustrated in Fig. 8(a), when no other frontier points are available, LERR-Nav enters a recall mode. As shown in Fig. 8(b) and 8(c), the LLM then guides the agent to hypothesize a likely location of the staircase entrance and successfully discovers the staircase. This ultimately enables the agent to accomplish the cross-floor task. In contrast, ASCENT fails to identify the staircase entrance and keeps wandering repeatedly, resulting in task failure.

A.4.4 Supplementary Figures

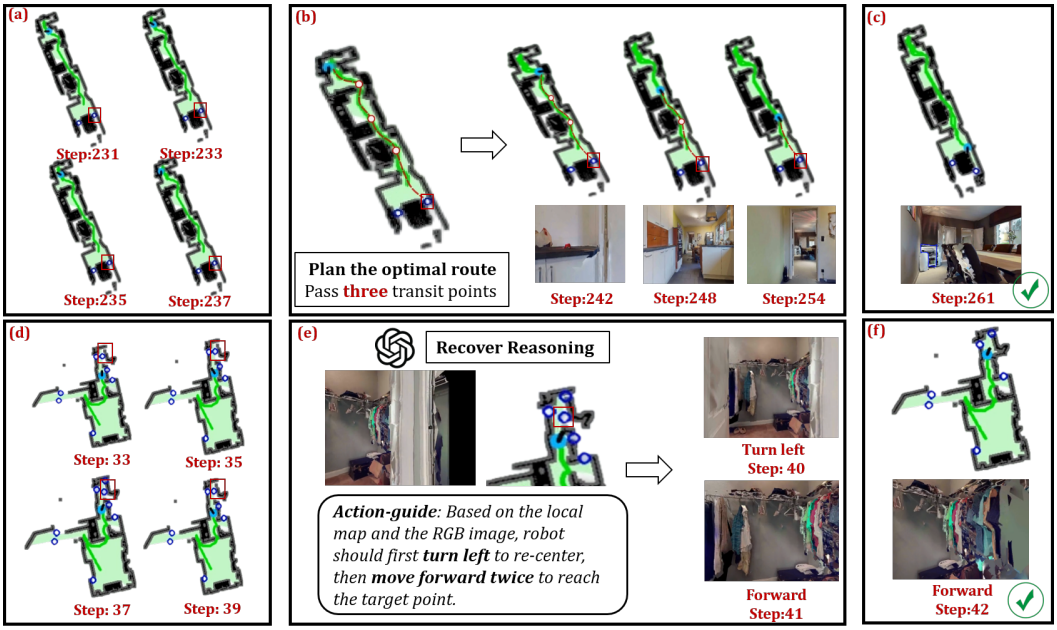

Figure 5: Explanation of the recovery strategy workflow: (a) and (d) illustrate abnormal behaviors caused by frontier points distance. (b)→(c) demonstrate how to handle overly distant frontier points using a segmented method. (e)→(f) show how LLM progressively approaches the target frontier point via fine-grained action adjustments.

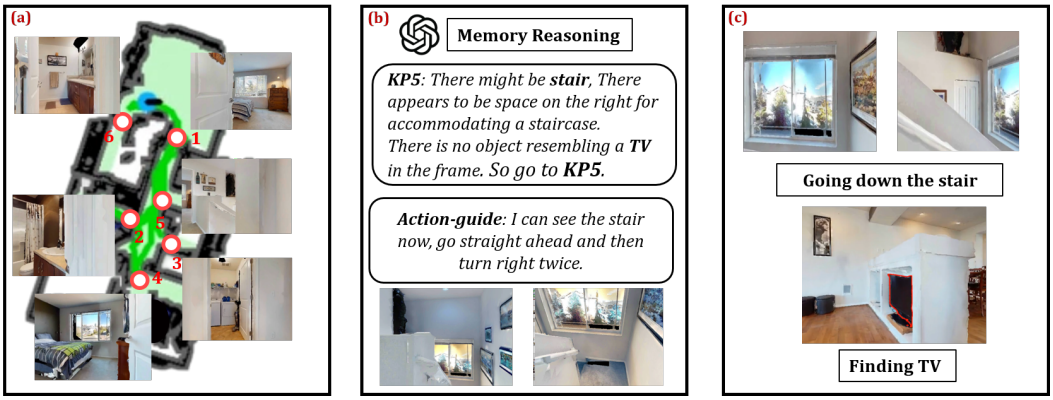

Figure 6: Explanation of how the reminiscing strategy facilitates progress in task: (a) visualizes the representation of the KeyPoint Map. (b) illustrates how the LLM chose to keypoints and locate the staircase entrance. (c) shows the robot descending the stairs after locating the stairway entrance and then continuing to find the target.

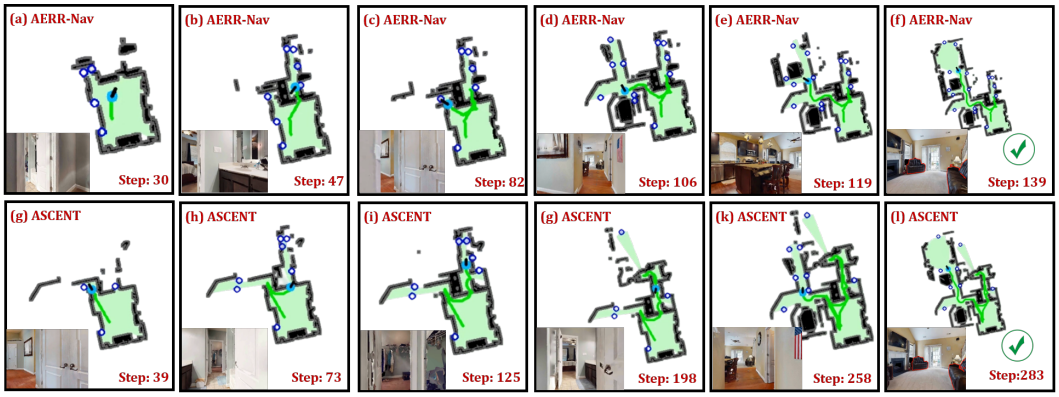

Figure 7: Case Study1: (a)–(f)/(g)–(l) illustrate the complete process of LERR-Nav/ASCENTGong et al. (2026) searching for the sofa.

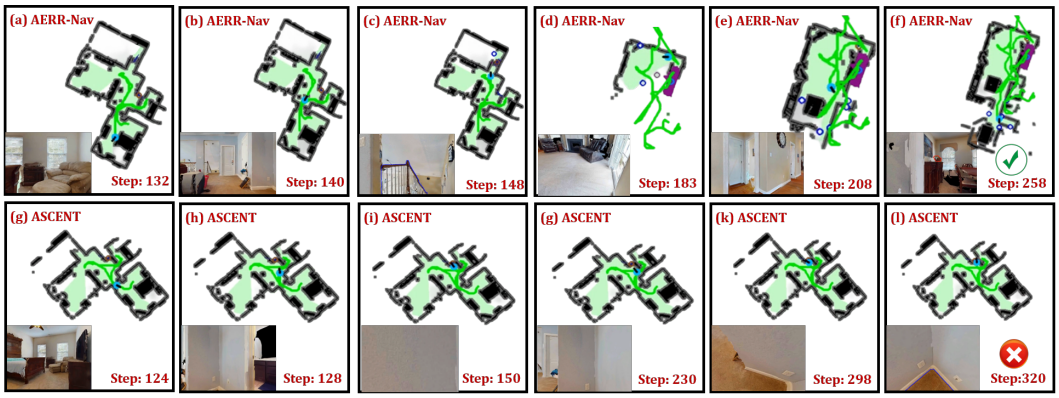

Figure 8: Case Study2: The visualization starts after each method has finished exploring the single-floor. (a)–(f)/(g)–(l) show the process of our LERR-Nav/ASCENTGong et al. (2026) searching for the potted plant.