# LIFELONG SMALL-OBJECT NAVIGATION IN CHANGING OBJECT LAYOUTS: A BENCHMARK AND METHOD

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Household robots need to continually navigate to different objects in the same environment, many of which are small and portable, such as tools and toys. Their small visual footprint and frequent occlusion make reliable observation difficult, and they may be moved by people without the robot observing the changes. We formulate this challenging task as Lifelong Small-object Navigation in Changing Object Layouts (**LiSoNav-COL**). Agents must seek suitable viewpoints for reliable observation, accumulate and reuse scene knowledge to efficiently locate subsequent targets, and update outdated memory after object relocation. To eliminate the need for prior scene scanning, we also require agents to start navigation with empty scene memory. Although practical, this task still lacks benchmarks designed around its defining assumptions. To bridge this gap, we introduce **LiSoNav-Eval**, a dedicated benchmark spanning 28 indoor scenes with 45 small-object categories. Its lifelong navigation sequences include both unchanged and relocated targets to evaluate memory reuse and adaptation to object relocation. To address this challenging task, we propose a navigation method based on multi-view Inspection with Viewpoint-Anchored Memory, dubbed **IVAM-Nav**. IVAM-Nav actively observes supporting surfaces from complementary viewpoints for reliable small-object perception and anchors the resulting memory to their observation viewpoints, supporting relational memory reuse and revalidation under similar viewing conditions. Extensive experiments on LiSoNav-Eval demonstrate favorable performance of IVAM-Nav against representative methods. Benchmark analyses also show that smaller objects, larger environments, and longer relocation distances pose greater challenges. We will release the dataset and code.

## 1 INTRODUCTION

Object-goal navigation requires embodied agents to search for and navigate to target objects specified by images or language (Krantz et al., 2023; Zhu et al., 2021). In household environments, robots may repeatedly perform such tasks in the same scene, motivating lifelong navigation that retains and reuses experience across successive tasks (Khanna et al., 2024b). Scene memory supports this process by preserving previously acquired object information to reduce redundant exploration.

In everyday household activities, many objects that robots need to locate and manipulate are small, portable items, such as tools and toys. People may move objects unnoticed by the robot, leaving its scene memory outdated and requiring verification and updates. These objects also occupy only a small portion of the visual field and are often occluded, making them difficult to observe clearly (Wang & Soh, 2023). These real-world challenges motivate us to study **Li**felong **S**mall-**o**bject **Nav**igation in **C**hanging **O**bject **L**ayouts (**LiSoNav-COL**). In this task, an agent starts with no prior scene memory and navigates to successive small-object goals within the same environment, incrementally acquiring scene knowledge throughout the navigation. Given that unobserved object relocations may render its memory outdated, the agent must learn to reuse accumulated scene knowledge while adapting to these changes for accurate and efficient navigation. Starting with an empty memory eliminates the need for a separate scene-scanning phase in real-world applications.

Existing navigation benchmarks adopt task assumptions that differ from those of LiSoNav-COL. The mainstream lifelong navigation benchmarks (Wani et al., 2020; Khanna et al., 2024b; Wang et al., 2026) mainly evaluate experience reuse in static scenes, where target locations remain unchanged across subtasks. OpenIN (Tang et al., 2025) evaluates sequential navigation after object



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.

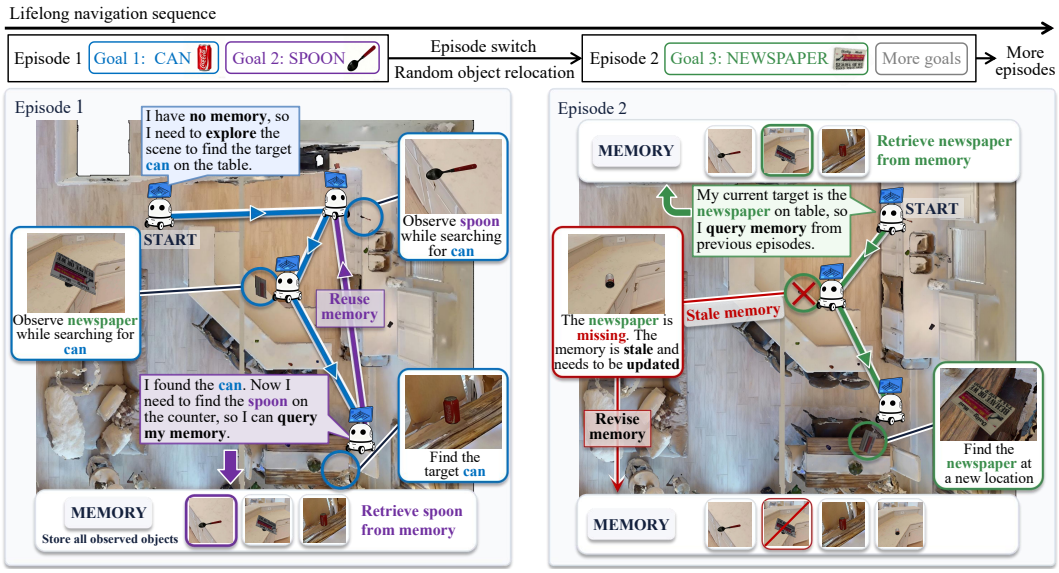



Figure 1: **An example navigation sequence in LiSoNav-Eval,** in which the agent is instructed to find the CAN on the table, the SPOON on the counter, the NEWSPAPER on the table, and other targets in sequence. Starting with no prior scene memory, the agent searches for the CAN while constructing memory from observed objects, which is later reused to find the SPOON. Between episodes, randomly selected objects are relocated without the agent observing the changes. When searching for the NEWSPAPER, the agent first revisits the remembered location, finds the newspaper missing, updates the scene memory, and then locates the newspaper elsewhere. This example illustrates how LiSoNav-Eval evaluates small-object navigation as agents accumulate scene knowledge, reuse it across tasks, and respond to changes in object locations.

relocation, but starts with a preconstructed scene graph and relocates objects before the task sequence begins. This setting assesses the use of prior scene knowledge after relocation, rather than its online acquisition and maintenance through repeated changes. Meanwhile, the small-object navigation benchmark (Wang & Soh, 2023) focuses on perception and target search in static scenes under a single-goal setting. These evaluation settings do not jointly capture the requirements of LiSoNav-COL: acquiring scene knowledge from an initially empty memory, reusing it across small-object navigation tasks, and adapting to repeated object relocation.

To support systematic evaluation of LiSoNav-COL, we introduce **LiSoNav-Eval**, a new benchmark comprising navigation sequences and an evaluation protocol for this task. Figure 1 presents an illustrative example of our benchmark, where the agent completes a sequence of small-object navigation subtasks (each corresponding to a specific goal) in the same environment, starting with no prior scene memory and retaining the knowledge acquired during task execution. Specifically, each navigation sequence comprises multiple episodes, each containing several navigation subtasks. Object locations remain unchanged within each episode, while randomly selected objects are relocated between episodes without the agent observing the changes. By including both unchanged and relocated targets, LiSoNav-Eval preserves opportunities to benefit from previously acquired knowledge while testing adaptation when remembered locations become outdated. Together, these settings support the evaluation of small-object navigation as agents accumulate scene knowledge, reuse it across tasks, and respond to changes in object layouts.

In LiSoNav-COL, incomplete or unreliable small-object observations can limit the effectiveness of existing scene-memory methods (Huang et al., 2026; Yang et al., 2025; Tang et al., 2025). Object relocation compounds these observation difficulties, making memory reuse and revalidation more challenging. In this paper, we propose **IVAM-Nav**, a lifelong small-object navigation method based on multi-view **I**nspection and **V**iewpoint-**A**nchored **M**emory, to address these challenges. Our key idea is to actively inspect the supporting surfaces where the target may be located from multiple viewpoints and anchor the memory to these viewpoints. Specifically, the agent acquires reliable and complementary observations around potential supporting surfaces and stores them with their camera poses and semantic associations. These records enable reobservation under similar viewing

conditions to effectively verify and update the memory. Extensive experiments on LiSoNav-Eval demonstrate the effectiveness of our algorithm. Our contributions can be summarized as:

- We formulate LiSoNav-COL, a new task for lifelong small-object navigation from empty scene memory under changing object layouts. We also introduce LiSoNav-Eval, a dedicated benchmark for evaluating agent performance on this task.
- We propose IVAM-Nav, which constructs a viewpoint-anchored graph memory through active multi-view inspection. This design supports relational memory reuse and memory revalidation under similar viewing conditions, enabling robust navigation.
- Extensive experiments on LiSoNav-Eval demonstrate favorable performance of IVAM-Nav against representative methods, and benchmark analyses identify smaller objects, larger environments, and longer relocation distances as persistent challenges.

## 2 Related Work

**Lifelong navigation in changing environments.** Existing navigation tasks and benchmarks span single-goal, sequential, and lifelong settings in largely static scenes (Chaplot et al., 2020; Krantz et al., 2023; Zhu et al., 2021; Wani et al., 2020; Khanna et al., 2024b; Miao et al., 2026; Yu et al., 2025; WANG et al., 2026), while dynamic settings consider moving objects and spatiotemporal retrieval (Kurenkov et al., 2023; Dorbala et al., 2026; Jiang et al., 2025; Bogenberger et al., 2026; Chen et al., 2025). Most closely related to our setting, OpenIN studies changing object layouts but requires a preconstructed scene representation before navigation, with its evaluation data and implementation currently unavailable (Tang et al., 2025). LiSoNav-Eval instead evaluates lifelong small-object navigation from empty memory under changing object layouts.

**Memory construction, reuse, and revalidation.** Existing navigation methods construct and query scene memory through object- or scene-centric graphs, visual observations, and spatial representations (Gu et al., 2024; Werby et al., 2024; Yang et al., 2025; Liu et al., 2025; Jiang et al., 2025; Zhu et al., 2025; Chang et al., 2024; Wang & Soh, 2023; Xu et al., 2026; Ge et al., 2026), while dynamic-memory methods further update or reassess stored information based on new observations (Tang et al., 2025; Jiang et al., 2025; Liu et al., 2025; Yan et al., 2025; Zhao et al., 2026; Ko et al., 2025). APRR and MSGNav actively adjust viewpoints after target detection or localization for manipulation or final-view selection (Qin et al., 2025; Huang et al., 2026). In contrast, IVAM-Nav builds reliable small-object memory via complementary multi-view inspection and uses acquisition viewpoints as persistent anchors for relational retrieval and revalidation.

## 3 LiSoNav-Eval Benchmark

### 3.1 Task Definition

The agent is required to complete a sequence of navigation subtasks (each corresponding to a small-object goal) in the same indoor environment with no prior scene memory. It needs to retain the knowledge acquired during subtask execution to facilitate the subsequent subtasks. The lifelong navigation sequence in LiSoNav-Eval spans three or four consecutive episodes, with each episode containing two or three navigation subtasks. Figure 1 illustrates such a sample.

The goal object of each subtask is specified by a natural language instruction. To clearly specify the target object, the instruction is designed to include coarse contextual cues, such as its *supporting surface* or *nearby surroundings*, while leaving its exact current location unspecified. This formulation also aligns with how people typically describe the location of a small object in a home environment. Herein, we show two instruction samples:

- **Instruction 1:** *Find the tissue box on the speckled countertop near the sink.*
- **Instruction 2:** *Find the turquoise Lego Duplo brick on a small wooden table next to a bed.*

We also provide an optional reference image to aid small-object recognition, which reveals no information about the current location of the target. In real-world applications, such images could be obtained through online searches or other external sources. Within an episode, object locations remain unchanged, and subtasks are executed sequentially from the terminal pose of the preceding subtask. Objects are relocated only between episodes to render memory outdated, not during navigation, to avoid turning the task into online tracking of moving objects.

Table 1: Comparison of LiSoNav-Eval with existing lifelong navigation benchmarks. † indicates that the corresponding evaluation data are not publicly released.

| | Scenes | Subtasks | Semantic target categories | Object relocation | Relocated subtasks | Relocated ratio | Prior scene knowledge | Subtasks per sequence |
|---|---|---|---|---|---|---|---|---|
| MultiON 3-ON val (Wani et al., 2020) | 11 | 3,150 | 1 | No | – | – | No | 3.00 |
| GOAT val-unseen (Khanna et al., 2024b) | 36 | 2,669 | 36 | No | – | – | No | 7.41 |
| OpenIN† (Tang et al., 2025) | 4 | – | – | Yes | – | 100% | Yes | 4–5 |
| LMEE-Bench test (Wang et al., 2026) | 36 | 828 | 38 | No | – | – | No | 4.99 |
| **LiSoNav-Eval** (Ours) | 28 | 1,199 | 45 | Yes | 545 | 45.45% | No | 8.27 |

A subtask is successful when the agent stops within 1.5 m of the valid instance matching the target description. Each subtask has a path-length budget of 10 times the shortest geodesic distance from the start to the target, and terminates immediately once this budget is exceeded. This budget allows exploration and recovery from outdated location information after object relocation while limiting excessive search. We report standard navigation metrics (SR and SPL), together with Historical Revisit (HR) and Relocated-target SR. Detailed definitions are provided in 5.1.

### 3.2 LiSoNav-Eval Dataset

**Scene and object assets.** We use indoor scenes from HM3D (Ramakrishnan et al., 2021) and select 82 diverse small-object instances across 45 categories from YCB (Calli et al., 2015) and HSSD (Khanna et al., 2024a), covering portable household items such as cups, balls, tools, and fruits. Examples are shown in Appendix Figure B.1.

**Dataset construction.** To construct a lifelong navigation sequence, we first populate an HM3D scene with small objects. We use Qwen-VL-Max (Bai et al., 2023) to identify suitable supporting surfaces and define category-specific placement rules. These rules exclude implausible object-surface combinations, such as placing food on beds. We also limit local object density to avoid excessive clutter and occlusion that would make small objects unnecessarily difficult to observe. Based on these rules, we randomly generate an initial object configuration for each scene and sample the target objects episode by episode. Targets selected in the first episode remain at their initial locations. In each subsequent episode, a selected target has a probability of being relocated to another compatible supporting surface, preserving plausible object placements after relocation. The same target is not selected more than once within an episode, but may reappear in later episodes, allowing previously acquired knowledge to be useful for subsequent subtasks.

Since LiSoNav-COL starts from empty scene memory, relocating an object that was not previously observed may not invalidate any acquired memory. To increase the likelihood that relocation creates stale memory, we preferentially sample objects in the same semantic region as, and within $5\,\mathrm{m}$ of, a previously selected target, as they are more likely to have been observed during earlier navigation.

**Dataset verification.** To ensure each target is visually identifiable from a suitable viewpoint, we sample observations around every target with the target near the view center. We use Qwen-VL-Max to assess target recognizability and manually remove overly difficult cases.

**Dataset annotation.** We provide a textual description for each of the 82 small-object instances, emphasizing attributes that distinguish visually similar instances. For coarse contextual cues, we use a real observation around the target location and Qwen-VL-Max to generate coarse descriptions of the supporting surface and available surroundings using attributes such as category, color, and material, while excluding fine-grained spatial details that could reveal the target's precise location.

**Dataset statistics.** As shown in Table 1, LiSoNav-Eval contains 145 lifelong navigation sequences from 28 HM3D scenes, comprising 500 episodes and 1,199 subtasks across 45 categories and 82 small-object instances. 545 (45.45%) of these subtasks involve relocated targets. Each navigation sequence contains 8.27 subtasks on average. Compared with OpenIN (Tang et al., 2025), LiSoNav-Eval spans more scenes and includes more subtasks per sequence. Besides, LiSoNav-Eval also supports evaluation of both knowledge reuse and adaptation to object relocation.

## 4 IVAM-Nav

To handle this challenging task, we propose IVAM-Nav, which is designed to actively inspect supporting surfaces from multiple viewpoints for reliable small-object perception and anchor its memory to these viewpoints. For the first task in a new environment, the agent searches for and navigates to a supporting surface where the target may be located. It then samples nearby viewpoints and

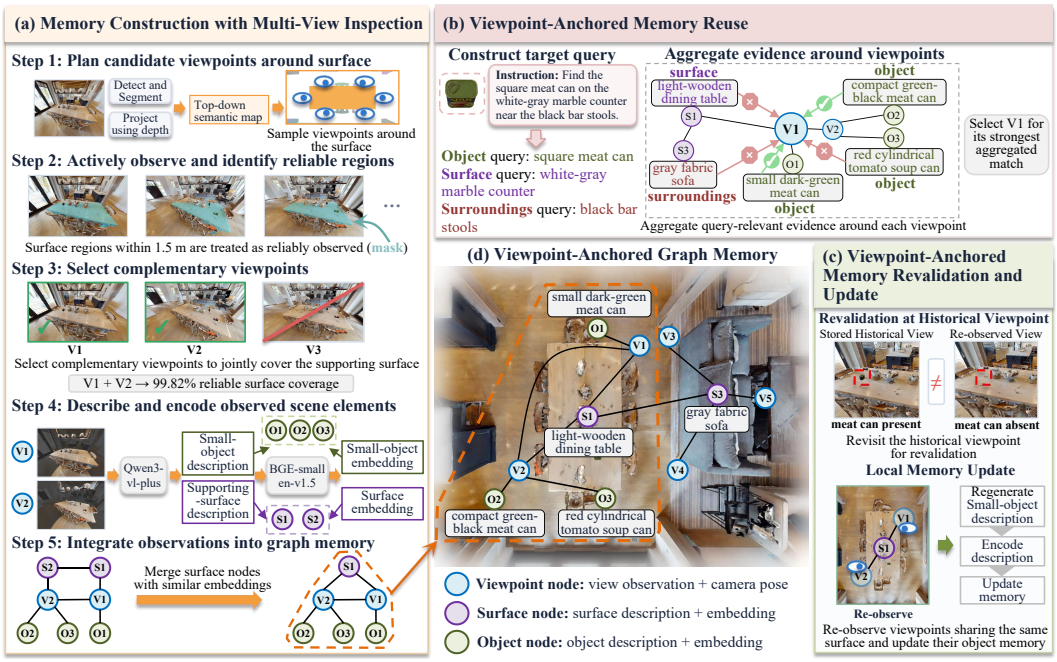



Figure 2: **Overview of IVAM-Nav.** IVAM-Nav maintains a viewpoint-anchored graph memory through three processes: (a) **Memory construction with multi-view inspection** actively observes supporting surfaces from complementary viewpoints to acquire reliable small-object evidence. (b) **Viewpoint-anchored memory reuse** aggregates relational evidence around historical viewpoints to retrieve those relevant to the current query. (c) **Viewpoint-anchored memory revalidation and update** revisits retrieved historical viewpoints to verify memory under similar viewing conditions and locally update inconsistent information.

retains a small complementary set to obtain clear observations of the supporting surface. These observations, together with their poses and related scene information, form a viewpoint-anchored graph memory. For subsequent subtasks, the agent aggregates the small-object and supporting-surface memories attached to each historical viewpoint and evaluates their relevance to the target query to retrieve the most relevant viewpoint. This design enables the agent to revisit previously recorded viewpoints and verify memory under similar viewing conditions.

### 4.1 Overall Framework

IVAM-Nav is a navigation framework built on a viewpoint-anchored graph memory. It comprises three core processes: (a) memory construction with multi-view inspection, (b) viewpoint-anchored memory reuse, and (c) viewpoint-anchored memory revalidation and update, as shown in Figure 2.

For each navigation subtask, the agent first queries the memory and revisits relevant historical viewpoints. At each revisited viewpoint, it checks the current observation for the target and compares it with the stored observation to assess memory validity. Detected inconsistencies trigger reobservation and updates at viewpoints associated with the same supporting surface. When useful historical candidates are unavailable or exhausted, the agent explores unobserved regions and actively inspects selected supporting surfaces, constructing new memories for subsequent tasks. The agent stops once the MLLM confirms the target in a current observation. Next we detail the three core processes.

### 4.2 Persistent Viewpoint-Anchored Graph Memory

The memory of IVAM-Nav is initialized empty and progressively accumulated throughout the entire lifelong navigation sequence. As shown in Figure 2(d), it is organized as a viewpoint-anchored graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with *viewpoint*, *surface*, and *object* nodes. Viewpoint nodes store observations and camera poses, while surface and object nodes store semantic descriptions and embeddings and are connected to their observation viewpoints. Viewpoints observing the same surface are connected, while nearby distinct surfaces are linked, enabling relational aggregation during retrieval. Historical viewpoints thus serve as anchors for memory retrieval and preserve the poses needed for revalidation under similar viewing conditions.

### 4.3 Memory Construction with Multi-View Inspection

To construct reliable memory, IVAM-Nav combines frontier-based exploration with active multi-view inspection of selected supporting surfaces, as illustrated in Figure 2(a). During exploration, RGB-D observations are used to detect and segment supporting surfaces, which are projected onto a top-down semantic map using depth and camera poses. Based on the target instruction, an MLLM selects either a frontier for further exploration or a nearby supporting surface for multi-view inspection. If a surface is selected, the agent samples viewpoints around it based on the semantic map (Step 1). The agent then visits the sampled viewpoints to acquire close-range observations (Step 2). For each viewpoint $v$, we define its reliable observation region $\mathcal{R}(v)$ as the visible surface region within a predefined observation range, visualized by the mask in Step 2 of Figure 2(a). We then select a compact complementary subset that jointly covers the supporting surface (Step 3) by greedily maximizing the additional reliable surface coverage:

$$v^* = \arg\max_{v \in \mathcal{C} \setminus \mathcal{S}} \frac{\left|\mathcal{R}(v) \setminus \bigcup_{u \in \mathcal{S}} \mathcal{R}(u)\right|}{|\mathcal{R}_{\text{surf}}|}, \tag{1}$$

where $\mathcal{C}$ and $\mathcal{S}$ denote the visited and selected viewpoint sets, respectively, $\mathcal{R}_{\text{surf}}$ is the supporting-surface region, and $|\cdot|$ denotes the area of a region. Selection stops once the required coverage is reached, reducing redundant observations while preserving reliable small-object information across the supporting surface. For each selected viewpoint, we mask regions beyond the reliable observation range in the corresponding view. We then use an MLLM to describe the visible small objects and supporting surface, and encode the resulting descriptions into semantic embeddings (Step 4). These scene elements are associated with their observation viewpoints and integrated into the graph (Step 5), with semantically consistent observations of the same supporting surface merged into a shared surface node.

### 4.4 Viewpoint-Anchored Memory Reuse

Memory reuse allows accumulated scene knowledge to guide subsequent navigation, but noisy or incomplete small-object information can make direct object-level retrieval unreliable. IVAM-Nav therefore treats historical viewpoints as retrieval anchors and evaluates them using the object, surface, and surroundings information associated with each viewpoint, as shown in Figure 2(b). The instruction is decomposed into object, surface, and surroundings queries and encoded into the same semantic space as the corresponding memory information.

For each viewpoint node, we aggregate query-relevant information within its two-hop graph neighborhood, capturing sufficient local context while limiting unrelated information. The object and surface queries are matched with directly connected object and surface nodes, respectively, such as O1 and S1 for V1 in the figure. The second hop further uses object information from neighboring viewpoints and surroundings information from nearby surfaces. The highest-scoring viewpoints are selected for revisit and memory revalidation.

### 4.5 Viewpoint-Anchored Memory Revalidation and Update

A retrieved viewpoint provides both a target-search location and an observation reference for memory revalidation. As shown in Figure 2(c), IVAM-Nav revisits the viewpoint and uses an MLLM to compare the new observation with the stored view under similar viewing conditions, focusing on small-object presence and spatial arrangement. This reduces viewpoint-induced variation, making inconsistencies more indicative of actual scene changes. When an inconsistency is detected, IVAM-Nav follows graph relations to re-observe related viewpoints sharing the same surface and updates their small-object descriptions and embeddings. This relation-guided revision removes stale local information without unnecessary global memory reconstruction.

## 5 Experiments

### 5.1 Experimental Setup

**Baselines.** We select representative baselines with different cross-task memory capabilities. VLFM (Yokoyama et al., 2024) provides a reference without cross-task memory. AstraNav (Hu et al., 2026) and 3D-Mem (Yang et al., 2025) retain and reuse scene knowledge across tasks but do not explicitly revise stale information, whereas DualMap (Jiang et al., 2025) and DynaMem (Liu

Table 2: Overall performance on LiSoNav-Eval. Oracle variants use ground-truth detection assistance as described in Sec. 5.1.

| Method | Cross-task Memory | Stale-memory Revision | Overall SR↑ | Overall SPL↑ | HR | Relocated-target SR↑ |
|---|---|---|---|---|---|---|
| VLFM (Yokoyama et al., 2024) | No | – | 12.3% | 8.8% | 27.6% | 11.7% |
| AstraNav (Hu et al., 2026) | Yes | No | 12.2% | 9.5% | 23.2% | 9.2% |
| DualMap (Jiang et al., 2025) | Yes | Yes | 13.9% | 10.2% | 35.2% | 10.3% |
| DynaMem (Liu et al., 2025) | Yes | Yes | 18.0% | 12.7% | 38.1% | 14.9% |
| DynaMem Oracle (Liu et al., 2025) | Yes | Yes | 20.1% | 14.4% | 39.6% | 15.8% |
| 3D-Mem (Yang et al., 2025) | Yes | No | 21.9% | 15.5% | 34.3% | 17.2% |
| 3D-Mem Oracle (Yang et al., 2025) | Yes | No | 27.5% | 18.3% | 49.2% | 19.3% |
| **IVAM-Nav** | Yes | Yes | **33.4%** | **18.4%** | **77.4%** | **30.5%** |
| IVAM-Nav (w/ reference image) | Yes | Yes | 36.3% | 21.3% | 79.6% | 30.1% |

et al., 2025) additionally incorporate mechanisms for adapting stored scene information to environmental changes. Together, these methods provide comparisons across navigation without cross-task memory, cross-task memory reuse, and memory adaptation under scene changes.

For a consistent task interface, all methods receive the same navigation instructions while retaining their core memory mechanisms. Since DualMap assumes a preconstructed scene map, we initialize it by traversing each environment along a predefined route before evaluation. We additionally report oracle variants of 3D-Mem and DynaMem with ground-truth small-object detections within 1 m. These variants reduce failures caused by small-object detection, allowing us to examine whether performance differences persist when this perceptual difficulty is partially alleviated. Details of baseline adaptations are provided in Appendix D.2.

**Evaluation metrics.** We report Success Rate (SR) and Success weighted by Path Length (SPL) for overall navigation performance. To evaluate historical-memory retrieval, we additionally introduce *Historical Revisit* (HR) as a behavioral measure. Among subtasks with valid historical target observations, HR is the fraction of such subtasks for which the agent revisits within 1.5 m of a location where the target was previously observed. Whether a target was previously observed is determined using ground-truth simulator information, with detailed criteria provided in Appendix D.5. To evaluate adaptation to object relocation, we further report *Relocated-target SR*, computed over subtasks whose targets have been relocated before the current task.

**Implementation details.** IVAM-Nav uses YOLO-World (Cheng et al., 2024) and SAM-L (Kirillov et al., 2023) for supporting-surface perception, Qwen3-VL (Bai et al., 2025) for MLLM-based operations, and BGE-small-en-v1.5 (Xiao et al., 2024) for textual semantic embeddings. We use `qwen3-vl-plus-2025-12-19` for all MLLM calls. For a controlled comparison, baseline components requiring MLLM inference also use the same model. Hyperparameters are fixed across all experiments, with further implementation details provided in Appendix C.

### 5.2 Overall Performance

As shown in Table 2, IVAM-Nav achieves an overall SR of 33.4% and SPL of 18.4%, outperforming all baselines in SR and remaining competitive in path efficiency. Notably, 3D-Mem Oracle and DynaMem Oracle achieve 27.5% and 20.1% SR despite receiving ground-truth small-object detection assistance, suggesting that alleviating small-object perception difficulty alone does not fully address LiSoNav-COL, where accumulated scene knowledge must also be effectively reused and maintained as object locations change. With an additional reference image, IVAM-Nav further reaches 36.3% SR and 21.3% SPL, indicating that improved target appearance information can provide additional benefits. In addition, IVAM-Nav achieves an HR of 77.4%, substantially higher than the baselines, indicating that its memory frequently guides the agent back to previously observed target locations. However, high HR alone does not imply success in LiSoNav-COL, as historical locations may become outdated after object relocation and require revalidation. On relocated targets, IVAM-Nav achieves 30.5% SR, compared with 19.3% for 3D-Mem Oracle and 15.8% for DynaMem Oracle. Together, these results show that IVAM-Nav effectively reuses accumulated scene knowledge while remaining effective under object relocation.

**Performance across episodes.** Figure 3 reports the first three episodes shared by all sequences, excluding Episode 4 to ensure comparison over the same sequence set. Episode 1 contains no object relocation, while later episodes provide more accumulated memory for reuse but may also introduce stale memory through relocation. IVAM-Nav, with or without a reference image, maintains

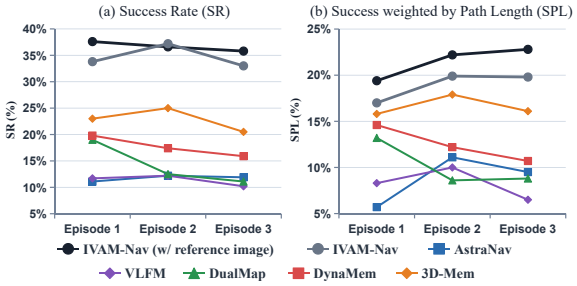



Figure 3: Performance across the first three episodes.

Table 3: Ablation study of IVAM-Nav on the LiSoNav-Eval subset.

| Setting | SR↑ | SPL↑ |
|---|---|---|
| w/o cross-task memory | 33.06% | 9.18% |
| w/o complementary viewpoints | 26.12% | 13.75% |
| w/o graph aggregation | 26.53% | 13.66% |
| w/o historical-viewpoint revisit | 28.98% | 14.73% |
| w/o memory update | 24.90% | 14.22% |
| w/o contextual cues | 30.20% | 15.38% |
| **Full** | **36.73%** | **21.38%** |

Table 4: Performance under static and changing-object-layout settings on the LiSoNav-Eval analysis subset. Relative drops are computed from the static to the changing setting.

| Method | Cross-task Memory | Stale-memory Revision | SR | | | SPL | | |
|---|---|---|---|---|---|---|---|---|
| | | | Static | Changing | Drop (%) | Static | Changing | Drop (%) |
| AstraNav (Hu et al., 2026) | Yes | No | 14.7% | 11.0% | 25.2% | 12.9% | 8.5% | 34.1% |
| DynaMem (Liu et al., 2025) | Yes | Yes | 20.8% | 17.6% | 15.7% | 15.7% | 12.6% | 19.7% |
| **IVAM-Nav** | Yes | Yes | **40.8%** | **36.7%** | **10.0%** | **25.6%** | **21.4%** | **16.4%** |

strong SR across episodes, while its SPL increases after Episode 1 and remains consistently higher thereafter, indicating sustained efficiency benefits from accumulated memory despite potential staleness. In contrast, other cross-task memory methods generally show larger SR degradation or less consistent SPL benefits in later episodes.

### 5.3 Method Analysis

To enable efficient controlled analysis, we conduct the following experiments on a representative LiSoNav-Eval subset containing 245 subtasks, using the reference-image variant of IVAM-Nav to reduce target-recognition ambiguity and focus the analysis on its memory mechanisms. The representativeness of the subset is analyzed in Appendix D.1, and its composition will be released.

**Ablation study.** We consider six variants. *w/o memory* removes cross-task memory entirely. *w/o complementary viewpoints* retains only one viewpoint near each supporting surface during memory construction. *w/o graph aggregation* matches memory candidates independently with the target object without aggregating graph information around historical viewpoints. *w/o historical-viewpoint revisit* navigates to the associated supporting surface for revalidation but does not return to the stored viewpoint. *w/o memory update* leaves inconsistent historical memory unchanged and continues to use it for subsequent navigation decisions. Finally, *w/o contextual cues* removes the surface and surroundings cues from the instruction, using only the small-object description for memory retrieval, to evaluate IVAM-Nav when such contextual information is unavailable.

As shown in Table 3, removing cross-task memory reduces SPL from 21.38% to 9.18%, showing the benefit of accumulated memory for navigation efficiency. Retaining only a single viewpoint for each surface reduces SR from 36.73% to 26.12%, supporting the use of complementary viewpoints to obtain reliable small-object information. Without graph aggregation, SR decreases to 26.53%, showing that aggregating scene information around historical viewpoints improves memory retrieval over independently matching memory candidates. Without historical-viewpoint revisit for revalidation, SR drops to 28.98%, showing the benefit of revisiting the original observation viewpoint under similar viewing conditions (Table A.1 further shows improved revalidation accuracy, especially for inconsistent observations). Disabling memory update yields the largest SR drop, to 24.90%, highlighting the importance of revising stale memory rather than continuing to use it for subsequent decisions.

Finally, even without contextual cues, IVAM-Nav retains an SR of 30.20%, indicating that the method remains effective when it relies only on the small-object description (Table A.2 further shows that this advantage persists over representative baselines under target-only input).

**Robustness to object relocation.** To assess how robustly different methods handle object relocation and the resulting stale memory, we compare representative cross-task memory methods under static and changing-object-layout settings. The static setting removes cross-episode object relocation, with details provided in Appendix D.3. As shown in Table 4, IVAM-Nav exhibits the smallest degradation under scene changes, with relative SR and SPL drops of 10.0% and 16.4%, respectively.

Table 5: Results by object size. Small and Large correspond to bounding-box volumes of $< 1\,\mathrm{L}$ and $\geq 1\,\mathrm{L}$, respectively.

| Method | Small | | Large | |
|---|---|---|---|---|
| | SR↑ | SPL↑ | SR↑ | SPL↑ |
| VLFM (Yokoyama et al., 2024) | 11.2% | 7.4% | 14.8% | 12.0% |
| AstraNav (Hu et al., 2026) | 12.5% | 9.6% | 11.5% | 9.1% |
| DualMap (Jiang et al., 2025) | 13.2% | 9.5% | 15.6% | 11.8% |
| DynaMem (Liu et al., 2025) | 17.9% | 12.7% | 18.3% | 12.8% |
| 3D-Mem (Yang et al., 2025) | 20.9% | 14.4% | 24.3% | 18.1% |
| IVAM-Nav | 29.2% | 16.4% | 42.9% | 23.0% |
| IVAM-Nav (w/ reference image) | 33.3% | 19.5% | 43.2% | 25.4% |

Table 6: Results by relocation distance. Short and Long correspond to geodesic displacements of $< 7.5\,\mathrm{m}$ and $\geq 7.5\,\mathrm{m}$, respectively.

| Method | Short | | Long | |
|---|---|---|---|---|
| | SR↑ | SPL↑ | SR↑ | SPL↑ |
| VLFM (Yokoyama et al., 2024) | 13.9% | 10.1% | 9.7% | 7.2% |
| AstraNav (Hu et al., 2026) | 11.3% | 9.1% | 7.2% | 7.2% |
| DualMap (Jiang et al., 2025) | 10.5% | 7.7% | 10.1% | 7.6% |
| DynaMem (Liu et al., 2025) | 17.6% | 11.9% | 12.2% | 7.7% |
| 3D-Mem (Yang et al., 2025) | 21.0% | 17.4% | 13.6% | 10.0% |
| IVAM-Nav | 35.5% | 18.4% | 25.5% | 13.4% |
| IVAM-Nav (w/ reference image) | 34.5% | 18.0% | 25.9% | 13.2% |

Table 7: Results by environment scale. Small, Medium, and Large correspond to navigable areas of $< 32\,\mathrm{m}^2$, $32$–$< 64\,\mathrm{m}^2$, and $\geq 64\,\mathrm{m}^2$, respectively.

| Method | Small | | Medium | | Large | |
|---|---|---|---|---|---|---|
| | SR↑ | SPL↑ | SR↑ | SPL↑ | SR↑ | SPL↑ |
| VLFM (Yokoyama et al., 2024) | 16.7% | 12.3% | 11.7% | 8.0% | 8.9% | 6.7% |
| AstraNav (Hu et al., 2026) | 13.3% | 11.6% | 12.8% | 10.0% | 10.1% | 6.6% |
| DualMap (Jiang et al., 2025) | 18.3% | 12.1% | 13.2% | 10.2% | 11.0% | 8.4% |
| DynaMem (Liu et al., 2025) | 20.7% | 15.6% | 18.5% | 12.9% | 14.6% | 9.6% |
| 3D-Mem (Yang et al., 2025) | 31.6% | 26.1% | 19.6% | 12.8% | 16.4% | 9.9% |
| IVAM-Nav | 41.5% | 22.4% | 33.7% | 19.1% | 25.0% | 13.5% |
| IVAM-Nav (w/ reference image) | 50.5% | 30.3% | 34.3% | 20.3% | 25.9% | 14.2% |

Table 8: Prior-observation rate of relocated targets.

| Method | Rate |
|---|---|
| VLFM (Yokoyama et al., 2024) | 42.4% |
| AstraNav (Hu et al., 2026) | 55.1% |
| DynaMem (Liu et al., 2025) | 49.7% |
| DynaMem Oracle (Liu et al., 2025) | 49.4% |
| 3D-Mem (Yang et al., 2025) | 72.7% |
| 3D-Mem Oracle (Yang et al., 2025) | 73.8% |
| IVAM-Nav | 71.0% |
| IVAM-Nav (w/ reference image) | 74.1% |

This indicates that IVAM-Nav is less affected by object relocation and the potentially stale memory it introduces. See Appendix A.1 for further analysis.

### 5.4 Benchmark Analysis

We analyze LiSoNav-Eval along factors that affect small-object perception, target search, and adaptation to scene changes. As shown in Table 5, most methods perform worse on smaller objects, reflecting the increased difficulty of acquiring reliable visual information when targets occupy a limited image area. Table 7 shows a clear performance decline as environment scale increases, indicating the greater exploration and search difficulty in larger spaces. For relocated targets, Table 6 shows that most methods perform worse under longer relocation distances, suggesting that larger location changes make recovery from outdated target memory more challenging.

**Prior target observation.** To assess whether relocation may invalidate acquired target information, we measure the fraction of relocated targets observed before relocation using ground-truth simulator information and the HR observation criterion (see Appendix D.5). As shown in Table 8, prior-observation rates range from 42.4% to 74.1% across methods, showing that relocation can frequently make historical target information outdated. Moreover, the SR gap between unchanged and relocated targets is larger for previously observed targets than for unobserved targets for both IVAM-Nav (12.08 vs. 3.11 percentage points) and DynaMem (9.00 vs. 5.84 percentage points). See Appendix A.2 for details.

## 6 Limitations and Future Directions

IVAM-Nav leaves several directions for future work. Active multi-view observation introduces additional navigation cost, motivating more cost-aware acquisition strategies. Memory revalidation is mainly triggered by retrieved historical viewpoints, motivating broader and opportunistic memory maintenance. Memory reuse and exploration are currently sequential and could be more tightly integrated. Finally, LiSoNav-Eval relocates small objects while keeping background scenes stable; future benchmarks could consider broader scene changes. See Appendix E for further discussion.

## 7 Conclusion

We introduced LiSoNav-COL, which studies lifelong small-object navigation from empty memory under changing object layouts, together with the LiSoNav-Eval benchmark. We further proposed IVAM-Nav, which combines multi-view inspection for reliable memory construction with viewpoint-anchored memory reuse and revalidation. Experiments show consistent improvements over existing navigation and scene-memory methods and validate the key designs of IVAM-Nav.

## AI USE STATEMENT

Generative AI tools were used to assist with the preparation of this paper. Specifically, they were used to help draft portions of the manuscript and to polish the authors' writing for clarity, grammar, and readability. The authors reviewed and revised all AI-assisted text and take full responsibility for the final manuscript.

We also used a multimodal large language model (MLLM) during dataset construction to generate textual annotations from collected visual observations. These AI-generated annotations constitute part of the dataset rather than the underlying navigation environments or object configurations. Details of the annotation procedure are provided in the paper. The generated annotations were reviewed as part of the dataset preparation process. We additionally use multimodal large language models as components of IVAM-Nav and applicable baseline methods. These uses include information extraction and decision-making, with method-specific details provided in the main paper and appendix.

The authors reviewed all AI-assisted work and take responsibility for the final content of this work, including text and dataset annotations produced with the aid of generative AI.

## ETHICS STATEMENT

This work does not involve human subjects or personally identifiable information. The proposed benchmark is constructed from existing simulated 3D environments and is intended for research on embodied navigation and lifelong scene memory. We do not identify specific ethical risks beyond those generally associated with embodied AI research. We will document the benchmark construction and evaluation protocols to support responsible use and reproducible research.

## REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the task formulation, benchmark construction, method, and evaluation protocol in the main paper and appendix. The appendix further documents implementation details, baseline adaptations, dataset construction and annotation procedures, evaluation metrics, and additional experimental settings and analyses. These materials are intended to provide the information necessary to reproduce the benchmark and experimental results. Code and benchmark data will be made available upon publication to facilitate future research and reproducibility.

## REFERENCES

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Benjamin Bogenberger, Oliver Harrison, Orrin Dahanaggamaarachchi, Lukas Brunke, Jingxing Qian, Siqi Zhou, and Angela P. Schoellig. Where did i leave my glasses? open-vocabulary semantic exploration in real-world semi-static environments. *IEEE Robotics and Automation Letters*, 11(3):3342–3349, 2026. doi: 10.1109/LRA.2026.3656790.

Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015*

*International Conference on Advanced Robotics (ICAR)*, pp. 510–517, 2015. doi: 10.1109/ICAR.2015.7251504.

Matthew Chang, Theophile Gervet, Mukul Khanna, Sriram Yenamandra, Dhruv Shah, So Yeon Min, Kavit Shah, Chris Paxton, Saurabh Gupta, Dhruv Batra, Roozbeh Mottaghi, Jitendra Malik, and Devendra Singh Chaplot. GOAT: GO to Any Thing. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.073.

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

Taijing Chen, Sateesh Kumar, Junhong Xu, Georgios Pavlakos, Joydeep Biswas, and Roberto Martín-Martín. Searching in space and time: Unified memory-action loops for open-world object retrieval, 2025. URL https://arxiv.org/abs/2511.14004.

Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16901–16911, 2024. doi: 10.1109/CVPR52733.2024.01599.

Vishnu Sashank Dorbala, Bhrij Patel, Amrit Singh Bedi, and Dinesh Manocha. Personalized embodied navigation for portable object finding, 2026. URL https://arxiv.org/abs/2403.09905.

Zuhao Ge, Xiaosong Jia, Chao Wu, Yuchen Zhou, Zuxuan Wu, and Yu-Gang Jiang. Evomemnav: Efficient self-evolving fine-grained memory for zero-shot embodied navigation, 2026. URL https://arxiv.org/abs/2606.03509.

Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5021–5028. IEEE, 2024.

Junjun Hu, Xinda Xue, Botao Ren, Minghua Luo, Jintao Chen, Haochen Bai, Liangliang You, and Mu Xu. Astranav-memory: Contexts compression for long memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8097–8109, June 2026.

Xun Huang, Shijia Zhao, Yunxiang Wang, Xin Lu, Wanfa Zhang, Rongsheng Qu, Weixin Li, Yunhong Wang, and Chenglu Wen. Msgnav: Unleashing the power of multi-modal 3d scene graph for zero-shot embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 37154–37163, June 2026.

Jiajun Jiang, Yiming Zhu, Zirui Wu, and Jie Song. Dualmap: Online open-vocabulary semantic mapping for natural language navigation in dynamic changing scenes. *IEEE Robotics and Automation Letters*, 10(12):12612–12619, 2025. doi: 10.1109/LRA.2025.3621942.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16384–16393, 2024a. doi: 10.1109/CVPR52733.2024.01550.

Mukul Khanna, Ram Ramrakhya, Gunjan Chhablani, Sriram Yenamandra, Theophile Gervet, Matthew Chang, Zsolt Kira, Devendra Singh Chaplot, Dhruv Batra, and Roozbeh Mottaghi. Goat-bench: A benchmark for multi-modal lifelong navigation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16373–16383, 2024b. doi: 10.1109/CVPR52733.2024.01549.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3992–4003, 2023. doi: 10.1109/ICCV51070.2023.00371.

Po-Chen Ko, Hung-Ting Su, CY Chen, Jia-Fong Yeh, Min Sun, and Winston H. Hsu. Context-aware replanning with pre-explored semantic map for object navigation. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4253–4267. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/ko25b.html.

Jacob Krantz, Theophile Gervet, Karmesh Yadav, Austin Wang, Chris Paxton, Roozbeh Mottaghi, Dhruv Batra, Jitendra Malik, Stefan Lee, and Devendra Singh Chaplot. Navigating to objects specified by images. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10882–10891, 2023. doi: 10.1109/ICCV51070.2023.01002.

Andrey Kurenkov, Michael Lingelbach, Tanmay Agarwal, Emily Jin, Chengshu Li, Ruohan Zhang, Li Fei-Fei, Jiajun Wu, Silvio Savarese, and Roberto Martín-Martín. Modeling dynamic environments with scene graph memory, 2023. URL https://arxiv.org/abs/2305.17537.

Peiqi Liu, Zhanqiu Guo, Mohit Warke, Soumith Chintala, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Dynamem: Online dynamic spatio-semantic memory for open world mobile manipulation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13346–13355, 2025. doi: 10.1109/ICRA55743.2025.11127619.

Bo Miao, Weijia Liu, Jun Luo, Lachlan Shinnick, Jian Liu, Thomas Hamilton-Smith, Yuhe Yang, Zijie Wu, Vanja Videnovic, Feras Dayoub, and Anton van den Hengel. Langmap: A human-verified benchmark for hierarchical open-vocabulary goal navigation. *arXiv preprint arXiv:2602.02220*, 2026.

Liang Qin, Min Wang, Peiwei Li, Wengang Zhou, and Houqiang Li. Active perception meets rule-guided rl: A two-phase approach for precise object navigation in complex environments. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7603–7612, 2025. doi: 10.1109/ICCV51701.2025.00713.

Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. In J. Vanschoren and S. Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/file/34173cb38f07f89ddbebc2ac9128303f-Paper-round2.pdf.

Yujie Tang, Meiling Wang, Yinan Deng, Zibo Zheng, Jingchuan Deng, Sibo Zuo, and Yufeng Yue. Openin: Open-vocabulary instance-oriented navigation in dynamic domestic environments. *IEEE Robotics and Automation Letters*, 10(9):9256–9263, 2025. doi: 10.1109/LRA.2025.3592071.

Jia-Ming Wang and Harold Soh. Small object navigation with context information. In *AAAI Summer Symposia*, 2023. URL https://api.semanticscholar.org/CorpusID:265120186.

Sen Wang, Bangwei Liu, Zhenkun Gao, Lizhuang Ma, Xuhong Wang, Yuan Xie, and Xin Tan. Explore with long-term memory: A benchmark and multimodal llm-based reinforcement learning framework for embodied exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 37098–37108, June 2026.

XUDONG WANG, Jiahua Dong, Baichen Liu, Qi Lyu, Lianqing Liu, and Zhi Han. Lifelong embodied navigation learning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 115492–115515, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/bbd7d8bd780fcf7143add2317ba04638-Paper-Conference.pdf.

Saim Wani, Shivansh Patel, Unnat Jain, Angel X. Chang, and Manolis Savva. Multi-on: Benchmarking semantic map memory using multi-object navigation. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram Burgard. Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation. *Robotics: Science and Systems*, 2024.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian yun Nie. C-pack: Packed resources for general chinese embeddings. *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024.

Yunzhe Xu, Yiyuan Pan, and Zhe Liu. Dream to recall: Imagination-guided experience retrieval for memory-persistent vision-and-language navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(8):9035–9049, 2026. doi: 10.1109/TPAMI.2026.3679426.

Zhijie Yan, Shufei Li, Zuoxu Wang, Lixiu Wu, Han Wang, Jun Zhu, Lijiang Chen, and Jihong Liu. Dynamic open-vocabulary 3d scene graphs for long-term language-guided mobile manipulation. *IEEE Robotics and Automation Letters*, 10(5):4252–4259, 2025. doi: 10.1109/LRA.2025.3547643.

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 Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 17294–17303, June 2025.

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

Ming-Ming Yu, Fei Zhu, Wenzhuo Liu, Yirong Yang, Qunbo Wang, Wenjun Wu, and Jing Liu. C-nav: Towards self-evolving continual object navigation in open world. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Shibo Zhao, Guofei Chen, Honghao Zhu, Zhiheng Li, Changwei Yao, Nader Zantout, Seungchan Kim, Wenshan Wang, Ji Zhang, and Sebastian Scherer. SuperMap: A Spatio-Temporal SLAM System for Visual-Language Navigation. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2026. doi: 10.15607/RSS.2026.XXII.052.

Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12684–12694, 2021. doi: 10.1109/CVPR46437.2021.01250.

Ziyu Zhu, Xilin Wang, Yixuan Li, Zhuofan Zhang, Xiaojian Ma, Yixin Chen, Baoxiong Jia, Wei Liang, Qian Yu, Zhidong Deng, Siyuan Huang, and Qing Li. Move to understand a 3d scene: Bridging visual grounding and exploration for efficient and versatile embodied navigation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8120–8132, 2025. doi: 10.1109/ICCV51701.2025.00761.

**Appendix Overview.** For convenient reference, the appendix is organized as follows:


- **Appendix A: Additional Experimental Results**
  - Additional Method Analysis
  - Additional Benchmark Analysis
- **Appendix B: Additional Details of LiSoNav-Eval**
  - Initial Scene and Object Population
  - Lifelong Sequence and Target Sampling
  - Object Relocation and Scene Evolution
  - Target Observability and Dataset Verification
  - Target and Context Annotation
  - Examples of Object Relocation
- **Appendix C: Additional Details of IVAM-Nav**
  - End-to-End Navigation Pipeline
  - Persistent Viewpoint-Anchored Graph Memory
  - Memory Construction with Multi-View Inspection
  - Viewpoint-Anchored Memory Retrieval Details
  - Viewpoint-Anchored Memory Revalidation and Local Update
  - Implementation Hyperparameters
  - MLLM Prompts
- **Appendix D: Additional Experimental Details**
  - Method-Analysis Subset
  - Baseline Adaptations
  - Static vs. Changing-Layout Evaluation Setting
  - Benchmark Factor Definitions
  - Historical Revisit and Prior-Observation Criteria
- **Appendix E: Further Discussion on Limitations and Future Directions**


# A Additional Experimental Results

## A.1 Additional Method Analysis

**Historical-viewpoint revalidation analysis.** We further analyze whether revisiting historical viewpoints provides more reliable information for memory revalidation. We compare the full IVAM-Nav with the *w/o historical-viewpoint revisit* variant from Table 3 on the same method-analysis subset. In the full method, the agent returns to the stored historical viewpoint before comparing the current observation with the historical one. In the ablated variant, the agent instead navigates to the associated supporting surface for revalidation without returning to the stored viewpoint. We evaluate whether the MLLM correctly determines if the historical and current observations are consistent in terms of their local small-object information. Consistent and Inconsistent cases correspond to unchanged and changed local small-object configurations, respectively. Since the two settings produce different navigation and memory retrieval behaviors, they trigger different numbers of revalidation cases on the same evaluation subset.

Table A.1: Accuracy of historical-memory revalidation on the method-analysis subset. Consistent and Inconsistent report accuracy separately for historical-current observations with unchanged and changed local small-object configurations, respectively.

| Setting | # Cases | Overall | Consistent | Inconsistent |
|---|---|---|---|---|
| w/o historical-viewpoint revisit | 1,067 | 70.29% | 78.11% | 66.67% |
| Full | 1,436 | **80.50%** | **79.74%** | **94.59%** |

As shown in Table A.1, the full method achieves an overall revalidation accuracy of 80.50%, compared with 70.29% without historical-viewpoint revisit. The accuracy on consistent cases is similar

(79.74% vs. 78.11%), whereas a substantially larger difference is observed on inconsistent cases (94.59% vs. 66.67%). This result is consistent with the motivation for historical-viewpoint revisit: returning to the stored observation viewpoint provides more comparable visual information when revalidating potentially changed small-object information. Together with the end-to-end results in Table 3, where removing historical-viewpoint revisit also reduces navigation performance, this analysis provides additional information for using historical viewpoints as anchors for memory revalidation.

**Performance without contextual cues.** LiSoNav-Eval provides coarse contextual cues about the target's supporting surface and, when available, nearby surroundings. To examine whether the performance advantage of IVAM-Nav depends on these auxiliary cues, we conduct an additional controlled comparison on the same method-analysis subset used in Sec. 5.3. For all methods, we remove the supporting-surface and surroundings cues and provide only the target-object description. We also exclude the optional reference image, so that all methods receive the same target-only input.

Table A.2: Performance on the method-analysis subset using only target-object information. No supporting-surface cues, surroundings cues, or reference images are provided.

| Method | SR↑ | SPL↑ |
|---|---|---|
| DynaMem (Liu et al., 2025) | 16.33% | 10.58% |
| 3D-Mem (Yang et al., 2025) | 17.55% | 11.14% |
| **IVAM-Nav** | **26.94%** | **14.16%** |

As shown in Table A.2, IVAM-Nav achieves 26.94% SR and 14.16% SPL when only target-object information is available, compared with 17.55%/11.14% for 3D-Mem and 16.33%/10.58% for DynaMem. The performance advantage therefore persists when the auxiliary contextual cues are removed, indicating that IVAM-Nav does not rely on these cues to outperform the representative scene-memory baselines in this controlled setting. These results complement the contextual-cue ablation in Table 3 and suggest that the gains of IVAM-Nav are not solely attributable to the additional supporting-surface and surroundings cues provided by LiSoNav-Eval.

**Historical retrieval scale across episodes.** During historical memory retrieval, IVAM-Nav identifies historical viewpoints with retrieval scores above the threshold of 0.65 as primary candidates for subsequent revisit and verification, as detailed in Appendix C.3. We analyze how the average number of these candidates changes across episodes. The statistics are computed only over subtasks that actually enter the historical retrieval stage, excluding those for which historical retrieval is not performed.

Table A.3: Average number of historical retrieval candidates across episodes, computed over subtasks that perform historical retrieval.

| Setting | Episode 1 | Episode 2 | Episode 3 | Episode 4 |
|---|---|---|---|---|
| IVAM-Nav | 6.08 | 8.92 | 10.97 | 11.82 |
| IVAM-Nav w/ reference image | 6.71 | 9.44 | 11.31 | 12.20 |

As shown in Table A.3, the average number of historical retrieval candidates increases across episodes, while the growth gradually slows as the sequence progresses. Without a reference image, the average increases from 6.08 to 8.92, 10.97, and 11.82, with successive increases of 2.84, 2.05, and 0.85 viewpoints. With a reference image, the corresponding averages are 6.71, 9.44, 11.31, and 12.20, with successive increases of 2.73, 1.87, and 0.89 viewpoints. Thus, in both settings, the additional number of retrieval candidates becomes progressively smaller in later episodes. These results show that, although the number of candidates increases as scene memory accumulates, its growth progressively slows over the episodes evaluated in LiSoNav-Eval.

### A.2 Additional Benchmark Analysis

**Interaction between prior observation and object relocation.** To further characterize the changing-layout challenge in LiSoNav-Eval, we analyze how navigation performance varies jointly

with prior target observation and object relocation. We conduct this analysis on IVAM-Nav without the optional reference image and DynaMem, both of which maintain cross-task memory and explicitly revise stale information. For each method, we partition all 1,199 subtasks according to two factors: whether the target had been observed before the beginning of the current episode and whether the target had been relocated. Prior observation is determined separately from each method's actual navigation history using the ground-truth observation criterion defined in Appendix D.5.

This partition distinguishes whether object relocation can make previously acquired target information outdated. For a previously observed target that remains unchanged, information acquired in earlier episodes can remain valid for subsequent navigation. In contrast, when a previously observed target is relocated, such historical target information can become outdated. If the target has not been observed before the current episode, its relocation does not invalidate any prior target observation of the agent. This analysis therefore helps distinguish the challenge associated with potentially outdated historical information from the general difficulty of navigating to relocated targets.

Table A.4: Performance conditioned on prior target observation and object relocation. Prior observation indicates whether the target had been observed before the beginning of the current episode. IVAM-Nav is evaluated without the optional reference image.

| Method | Observed before current episode | Target status | # Subtasks | SR↑ | SPL↑ |
|---|---|---|---|---|---|
| IVAM-Nav | No | Unchanged | 421 | 31.59% | 15.02% |
| | No | Relocated | 158 | 28.48% | 12.75% |
| | Yes | Unchanged | 233 | 43.35% | 30.62% |
| | Yes | Relocated | 387 | 31.27% | 17.09% |
| DynaMem | No | Unchanged | 490 | 18.98% | 13.96% |
| | No | Relocated | 274 | 13.14% | 7.52% |
| | Yes | Unchanged | 164 | 25.61% | 18.79% |
| | Yes | Relocated | 271 | 16.61% | 12.02% |

As shown in Table A.4, both methods exhibit a consistent pattern in SR. Previously observed targets achieve higher SR than targets without prior observations, with a larger difference when the target remains unchanged. For IVAM-Nav, the SR difference between previously observed and unobserved targets is 11.76 percentage points for unchanged targets, compared with 2.79 points for relocated targets. DynaMem exhibits the same pattern, with corresponding differences of 6.63 and 3.47 points. This pattern is consistent with the lifelong setting of LiSoNav-Eval, where target information acquired in earlier episodes can remain useful when the object layout is unchanged, while relocation can make such information outdated.

The same interaction can be viewed from the effect associated with relocation. For IVAM-Nav, the SR difference between unchanged and relocated targets is 12.08 percentage points among previously observed targets, compared with 3.11 points among targets without prior observations. For DynaMem, the corresponding differences are 9.00 and 5.84 points. Thus, for both methods, the SR difference associated with relocation is larger when the target has already been observed in an earlier episode. These results indicate that the changing-layout challenge is particularly pronounced when previously acquired target information can become outdated, rather than reflecting only the general difficulty of navigating to relocated targets.

# B Additional Details of LiSoNav-Eval

This section provides additional details of the construction of LiSoNav-Eval. During dataset generation, we introduce a set of constraints and validation steps to encourage plausible object arrangements, maintain valid navigation tasks, and reduce unintended artifacts introduced by procedural construction. These design choices aim to preserve the intended challenges of lifelong small-object navigation while keeping the generated sequences sufficiently diverse. We next describe the initial scene population, target sampling, object relocation, and dataset verification procedures in detail.

### B.1 Initial Scene and Object Population

**Scene and placement constraints.** We first determine the navigable region in which a lifelong navigation sequence is constructed, and then populate this region with small objects under semantic placement constraints. Specifically, we sample the initial agent pose on a navigable island with an area of at least $10\,\mathrm{m}^2$, excluding small isolated components that provide little meaningful navigation space. Following GOAT-Bench (Khanna et al., 2024b), we further restrict each sequence to a single floor and consider only supporting surfaces and object placements on the same floor as the initial agent pose.

Within this navigation region, we place small objects on semantically compatible supporting surfaces. As described in the main paper, we use Qwen-VL-Max offline to establish object–room and object–surface compatibility rules. A candidate placement is retained only when the room is plausible for the object and the supporting-surface category is not incompatible with it. These constraints reduce implausible object placements while still allowing diverse configurations to be randomly generated within the selected navigation region.

**Object population constraints.** Purely random placement can easily create unrealistically crowded supporting surfaces, which would make small-object visibility an artifact of synthetic clutter rather than the intended navigation and memory problem. We therefore limit the number of small objects placed on each supporting surface according to its horizontal top area. Specifically, for a supporting surface with top area $A$, we define its maximum object capacity $C(A)$ as

$$C(A) = \begin{cases} 1, & A < 0.3\,\mathrm{m}^2, \\ 2, & 0.3 \le A < 0.6\,\mathrm{m}^2, \\ 3, & 0.6 \le A < 1.2\,\mathrm{m}^2, \\ 5, & A \ge 1.2\,\mathrm{m}^2. \end{cases} \tag{B.1}$$

We additionally allow at most one instance of the same object type on a supporting surface, reducing excessive local repetition. This area-based capacity serves as a simple constraint to reduce overcrowded configurations rather than an exact physical packing model. It allows larger surfaces to contain more objects while avoiding excessive clutter that could unnecessarily increase the perceptual difficulty of small-object navigation.

We place at most 45 small objects in each initial scene configuration. This upper bound limits the overall object density while providing a sufficiently large pool of candidate targets for constructing multi-episode navigation sequences. We require at least six successfully placed objects before constructing a sequence, ensuring that the scene can support the minimum sequence configuration of three episodes with two subtasks each. Our object-placement pipeline is adapted from the Habitat-Data-Collector used in DualMap (Jiang et al., 2025), with modifications to support our small-object assets, placement constraints, and multi-episode sequence construction.

### B.2 Lifelong Sequence and Target Sampling

**Temporal scene structure.** A lifelong navigation sequence contains three or four episodes, each with two or three navigation subtasks. The first episode uses the initial object configuration. Each subsequent episode inherits the object configuration resulting from the preceding episode. Consequently, relocations persist over time rather than being independently regenerated from the original scene. This produces a genuinely evolving scene state in which accumulated memory can remain useful for some objects while becoming outdated for others.

**Episode start poses.** The first episode starts from the initial pose sampled before scene population. For each subsequent episode, we independently resample an agent start pose rather than using the terminal pose of the previous episode. The new pose is required to lie on the same navigable island as the initial sequence start and to have a height within $0.4\,\mathrm{m}$ of the initial height.

Independent resampling prevents the outcome of one episode from directly determining the starting condition of the next. At the same time, restricting the pose to the same navigable component and a similar height prevents this randomization from inadvertently moving the agent to a disconnected region or a different floor level. Thus, episode transitions change the starting condition without changing the underlying navigation domain.

**Candidate target filtering.** For every subtask, we first construct a basic candidate pool from the small objects successfully placed in the scene. A candidate object must have a valid NavMesh point within $1.5\,\mathrm{m}$ in the horizontal plane and admit a finite geodesic path from the current construction pose. We additionally restrict the geodesic distance to $1$–$30\,\mathrm{m}$. The lower bound excludes nearly trivial cases in which the target is already next to the agent, while the upper bound avoids extreme distance outliers that could disproportionately increase the search scale of individual subtasks.

When several feasible instances of the same object type are available, we retain the instance with the shortest geodesic distance and then randomly sample among the remaining object types. Sampling directly from all instances would favor object types with more copies, whereas always selecting the globally nearest object would bias the benchmark toward short-range targets. This two-stage procedure reduces both effects while retaining diversity in target selection.

The same target is not selected more than once within an episode. Since the object configuration remains unchanged within an episode, repeatedly querying the same target would provide limited additional value for evaluating lifelong navigation. The target may reappear in later episodes, where previously acquired knowledge can be reused if the object remains unchanged or become outdated if it is relocated.

### B.3 Object Relocation and Scene Evolution

**Relocation and target sampling.** The first episode uses the initial object configuration without relocation, providing an initial period for the agent to acquire scene knowledge from empty memory. For each subtask in subsequent episodes, we first determine whether its target will be relocated, with probability $0.7$. If relocation is not triggered, the target is sampled directly from the basic candidate pool described above.

When relocation is triggered, simply sampling a target from the same pool may select an object that was unlikely to have been observed previously. Moving such an object would change the scene without necessarily making acquired memory outdated. We therefore further decide, with probability $0.6$, whether to apply a history-based constraint when sampling the target to be relocated. Under this constraint, candidates are restricted to objects located in the same semantic region as, and within $5\,\mathrm{m}$ of, a previously selected target. Objects in these regions are more likely to have been encountered during earlier navigation, increasing the chance that relocating them makes previously acquired information stale. If the constraint is not activated, the relocation target is sampled from the basic candidate pool.

The history-based constraint is used probabilistically rather than for every relocation. This increases the occurrence of relocations that can affect previously acquired memory without concentrating all relocated targets around earlier navigation goals. If no valid candidate satisfies the constraint, we fall back to the basic candidate pool, avoiding failed sequence generation in scenes where suitable nearby objects are unavailable.

The $0.7$ relocation probability also leaves a substantial fraction of targets unchanged across episodes. This mixture is important for LiSoNav-COL: retained memory should sometimes remain useful and sometimes become outdated, rather than allowing an agent to assume that all historical target locations become invalid after every episode transition.

**Relocation procedure.** Once a target has been selected for relocation, we move that selected object instance to a different compatible supporting surface. Candidate destinations follow the same room and object–surface compatibility rules used for initial scene population. We additionally require a horizontal displacement of at least $1\,\mathrm{m}$ from the original location, ensuring a meaningful change in the target location rather than a minor local displacement.

After a candidate relocation, we rebuild the NavMesh and check that the target remains reachable from navigable space. Invalid or unreachable placements are discarded and another compatible supporting surface is considered. If a valid relocation cannot be obtained after repeated attempts, the target remains at its original location. We prefer retaining an unchanged but valid target over forcing a relocation that would introduce an implausible or unreachable navigation task.

The object configuration resulting from an episode is retained when constructing the next episode, so relocations accumulate over the lifelong navigation sequence. Once an episode begins, however,

its object configuration remains fixed throughout all of its subtasks. Thus, LiSoNav-Eval introduces unobserved changes between episodes without turning the task into navigation among continuously moving targets.

### B.4 Target Observability and Dataset Verification

**Viewpoint search.** For every selected target, we explicitly verify that reasonable visual observations exist from navigable space. Candidate viewing directions are sampled at $15^\circ$ intervals around the target, yielding 24 angular directions. For each direction, we consider target distances of $0.5$, $1.0$, $1.5$, and $2.0\,\mathrm{m}$. Candidate positions must admit a sufficiently close NavMesh projection and remain reachable. The camera is oriented toward the target, and simulator semantic annotations are used to confirm that the target is visible.

The purpose of this procedure is not to guarantee that a navigation method will observe the target during evaluation. Instead, it removes pathological cases in which the target is effectively unobservable even from nearby navigable viewpoints. Without such a check, failures caused by invalid asset placement or severe occlusion could be incorrectly attributed to navigation or memory.

**Target-centered observations.** Each candidate view is first rendered at $2560 \times 2560$ resolution and then center-cropped to $640 \times 640$. A view is considered valid only if the target remains visible in the crop and at least $90\%$ of the target pixels visible in the full image are retained inside it. This criterion ensures that the collected verification observations actually provide clear target appearance information rather than relying on a few peripheral pixels.

We require at least one valid angular viewpoint for every target and retain at most three. The first retained view maximizes the number of visible target pixels. Additional views favor angular diversity while maintaining target visibility; in particular, angularly complementary candidates are preferred among views whose target-pixel count remains at least $80\%$ of the strongest candidate. Multiple viewpoints reduce the chance that dataset acceptance is determined by an accidental single favorable view, while limiting the number of retained views keeps offline verification cost manageable.

**MLLM-assisted quality control.** The retained views are evaluated with Qwen-VL-Max as an offline quality-control step. The model jointly considers multiple observations of the same target and checks whether the target is visually identifiable and whether its placement appears physically valid. Samples in which the target cannot be identified or is visibly floating or abnormally embedded in the environment are rejected. Ambiguous cases are further inspected manually, as described in the main paper.

Importantly, Qwen-VL-Max is used only during dataset construction and annotation. It does not determine navigation success during evaluation, which is computed from the simulator-based stopping criterion defined in Sec. 3.1.

### B.5 Target and Context Annotation

The real observations collected around each target during dataset verification are also used for contextual annotation. For relocated targets, these observations are collected after relocation and therefore reflect the target's current placement. We use Qwen-VL-Max to generate the supporting-surface and surroundings descriptions from these observations.

In household object search, users may remember coarse contextual information, such as the general supporting surface or nearby surroundings of an object, without knowing its exact location. Accordingly, the contextual descriptions are consistent with the target's current placement while excluding fine-grained spatial information that would directly reveal its location. Using pre-relocation context for a relocated target would make the task instruction itself outdated, conflating stale internal scene memory with an incorrect goal specification. With current context, the instruction remains valid while the agent's accumulated memory may still contain outdated object-location information that needs to be revalidated. These cues provide coarse semantic guidance rather than uniquely specifying the target location, as similar supporting surfaces or surroundings may occur at multiple locations within a scene; the ablation in Table 3 in the main paper further shows that IVAM-Nav remains effective without these contextual cues.

The remaining annotation procedure follows the description in Sec. 3.2.

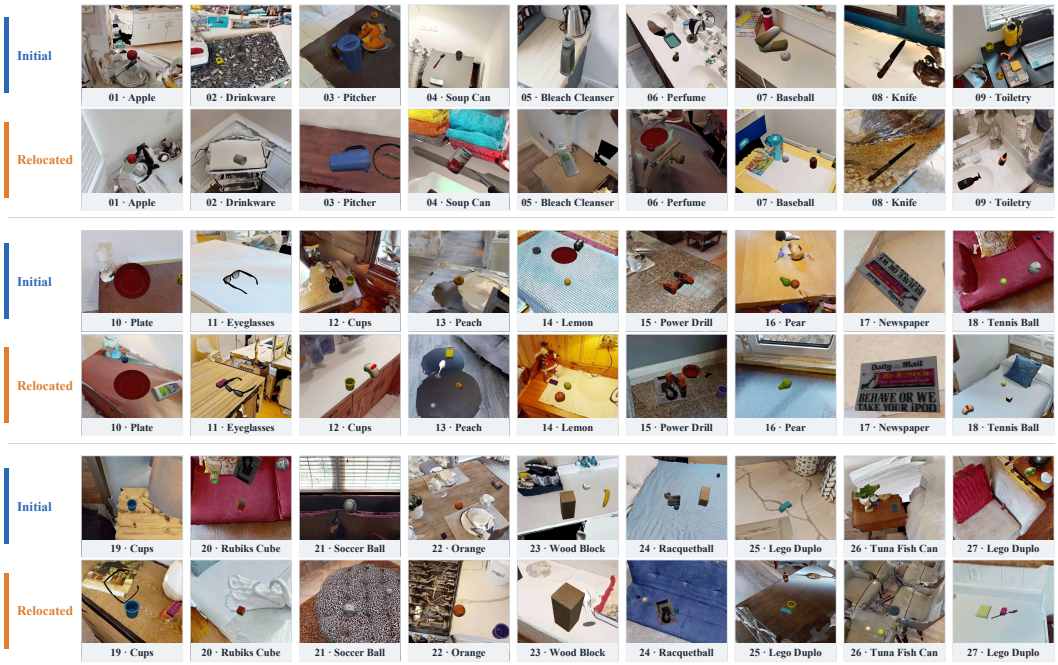



Figure B.1: **Examples of object relocation in LiSoNav-Eval.** Each column shows the same target instance at its initial (top) and relocated (bottom) locations using actual scene observations.

### B.6 Examples of Object Relocation

Figure B.1 provides examples of target instances before and after relocation in LiSoNav-Eval. Each column shows the same object instance at its initial and relocated locations, using actual observations rendered from the corresponding scene configurations. The examples cover diverse object categories, supporting surfaces, and relocation surroundings, illustrating the visual changes encountered by an agent when previously observed small objects are moved to new locations.

## C Additional Details of IVAM-Nav

This section provides additional implementation details of IVAM-Nav and clarifies how reliable memory construction, relational memory reuse, and memory revalidation and local revision are integrated into the complete navigation process. We focus on the navigation and memory-maintenance logic here. The detailed prompts used for MLLM-based description, decision, comparison, and target verification are provided in Appendix C.7.

### C.1 End-to-End Navigation Pipeline

IVAM-Nav maintains a viewpoint-anchored graph memory that is initialized empty at the beginning of each LiSoNav-Eval lifelong navigation sequence and retained across all subtasks and episodes within the sequence. Consequently, observations acquired during earlier subtasks can be reused later, while unobserved object relocation between episodes may make part of the accumulated small-object memory stale.

At the beginning of each subtask, the agent constructs a target query from the target-object description, supporting-surface description, and optional surroundings. The optional reference image provides additional visual information about the target appearance but does not reveal its location. The resulting target information is subsequently used for historical-memory retrieval, exploration decisions, and target verification.

**History-first retrieval.** IVAM-Nav first attempts to locate the target using previously accumulated memory. It performs relational retrieval over the graph, where object, supporting-surface, and neighboring contextual information are aggregated within the two-hop graph neighborhood of each historical viewpoint to compute a retrieval score. Promising viewpoints are then selected and visited in a greedy verification order.

For each visited historical viewpoint, the agent returns to its stored position and orientation and reacquires the current observation. An MLLM first compares the historical and current views to determine whether the local small-object configuration remains consistent. If the two views are consistent, the target is verified directly from the current observation. If an inconsistency is detected, the affected local small-object memory is revised before historical retrieval continues. The corresponding MLLM comparison and target-verification prompts are described in Appendix C.7.

Importantly, historical semantic similarity only determines which locations are worth inspecting. A high retrieval score never directly triggers STOP; the target must be confirmed from a current observation.

**Switching to active exploration.** If no useful historical candidate is available, or the retrieved candidates are exhausted without target confirmation, the agent switches to active exploration.

The agent first performs a multi-directional scan to initialize or update the explored-area map, frontiers, and visible supporting-surface candidates. At each subsequent exploration step, an MLLM chooses between visiting a frontier and inspecting a nearby supporting surface. Frontier navigation expands the currently observed region and updates the online map, whereas selecting a supporting surface invokes the dedicated close-range multi-view observation procedure introduced in the main paper.

The resulting observations are used to construct or extend the graph. Selected viewpoints store their observations and camera poses and are connected to the corresponding small-object and supporting-surface nodes. These newly constructed memories can immediately support the current search and persist for reuse in subsequent subtasks.

Overall, the navigation process integrates the three core processes described in the main paper. IVAM-Nav first attempts to reuse accumulated memory by retrieving and revisiting relevant historical viewpoints, where the stored information is revalidated against the current scene and locally updated when necessary. When useful historical candidates are unavailable or exhausted, the agent switches to active exploration and multi-view inspection, with newly acquired observations incorporated into the graph for subsequent reuse.

### C.2 Persistent Viewpoint-Anchored Graph Memory

The graph contains *viewpoint*, *surface*, and *object* nodes, connected according to their observation and spatial relations. Each viewpoint is connected to the small objects and supporting surface observed from that viewpoint through viewpoint–object and viewpoint–surface edges. For multiple viewpoints associated with the same supporting surface, neighboring viewpoints are connected by viewpoint–viewpoint edges, preserving relations across observations of different parts of the surface. When a new viewpoint is added to memory, its spatial proximity to existing viewpoints is also examined. If an existing viewpoint is sufficiently close and the two viewpoints correspond to distinct surfaces, their associated surface nodes are connected by a surface–surface edge. These relations provide the graph neighborhoods used to aggregate object, surface, and cross-view contextual information during memory retrieval.

### C.3 Viewpoint-Anchored Memory Retrieval Details

The main paper describes relational memory reuse using two-hop information aggregation. Here, we provide the detailed scoring and candidate-selection procedure.

**Two-hop object information.** During memory construction, complementary viewpoints are selected to reliably observe different parts of the same supporting surface. Their observed regions can still partially overlap, so the same small objects may be observed from neighboring viewpoints. Such consistent observations provide additional support for the target match beyond a single viewpoint.

Specifically, for a historical viewpoint $v$, let $O_1(v)$ and $O_2(v)$ denote the maximum target-object similarity among the small-object descriptions in the first- and second-hop neighborhoods of $v$, respectively. When the second-hop similarity exceeds a predefined threshold, it provides an additional bonus to the first-hop object score:

$$B_{\mathrm{obj}}(v) = \begin{cases} \min(0.2O_2(v),\, 0.08), & O_2(v) > 0.8, \\ 0, & \text{otherwise.} \end{cases} \tag{C.1}$$

The resulting object score is

$$S_{\mathrm{obj}}(v) = \min\left(1, O_1(v) + B_{\mathrm{obj}}(v)\right). \tag{C.2}$$

This allows matching small-object information from neighboring viewpoints to reinforce the object score while limiting its maximum contribution.

**Supporting-surface and surroundings information.** Let $S_1(v)$ denote the maximum similarity between the target supporting-surface query and the first-hop supporting-surface information of viewpoint $v$. When optional surroundings information is available, let $N_2(v)$ denote the maximum similarity between the surroundings query and supporting-surface information in the second-hop neighborhood. The surroundings bonus is

$$B_{\mathrm{ctx}}(v) = \begin{cases} 0.2N_2(v), & N_2(v) > 0.8, \\ 0, & \text{otherwise,} \end{cases} \tag{C.3}$$

and the supporting-surface score is

$$S_{\mathrm{surf}}(v) = \min\left(1, S_1(v) + B_{\mathrm{ctx}}(v)\right). \tag{C.4}$$

The final retrieval score combines the object and supporting-surface scores:

$$S_{\mathrm{view}}(v) = 0.6S_{\mathrm{obj}}(v) + 0.4S_{\mathrm{surf}}(v). \tag{C.5}$$

**Candidate selection and verification.** Historical viewpoints with $S_{\mathrm{view}}(v) \geq 0.65$ are first identified as promising candidates and ranked according to their retrieval scores. IVAM-Nav then greedily selects the highest-scoring candidates from the remaining viewpoints one by one for verification. For each selected candidate, historical viewpoints within $1\,\mathrm{m}$ can also be included for verification even if their individual scores are below the retrieval threshold. This allows IVAM-Nav to inspect additional nearby observations while prioritizing viewpoints with stronger retrieval information, improving retrieval efficiency without substantially increasing navigation cost.

The resulting viewpoints are visited sequentially for verification. Each viewpoint is verified independently: the agent returns to its stored position and orientation and reacquires the current observation before target verification.

### C.4 Viewpoint-Anchored Memory Revalidation and Local Update

Historical retrieval identifies locations that are semantically relevant to the current target, but their stored small-object information may have become outdated. IVAM-Nav therefore revalidates each retrieved viewpoint before using its historical content as current information.

For each candidate viewpoint, the agent navigates back to the stored position and orientation and acquires a new observation. The historical and current observations are then compared by an MLLM under matched viewing conditions. This comparison focuses on changes in the presence, number, and spatial arrangement of small objects rather than using background appearance as the primary signal. The detailed prompt is given in Appendix C.7.

If the historical and current observations are consistent, IVAM-Nav performs target-presence verification using the current observation. The stored historical image is not used as information that the target is currently present.

If an inconsistency is detected, IVAM-Nav treats the associated small-object memory as potentially stale rather than simply discarding the retrieved viewpoint. The inconsistent viewpoint is used as an anchor to identify other historical viewpoints sharing the same supporting-surface node. The

agent re-observes these viewpoints under the current scene state, regenerates their small-object descriptions, encodes the new descriptions, and replaces the corresponding small-object information in memory.

The supporting-surface node and its semantic description are kept unchanged during this revision. The supporting surface instead serves as a stable local anchor for determining which historical viewpoints belong to the affected memory region. Thus, an observed change in a small object does not cause unnecessary modification of the surrounding structural memory.

Target presence is evaluated from the newly acquired observations during this local refresh. Viewpoints that have actually been re-observed and judged not to contain the target are excluded from the remaining candidate set. IVAM-Nav then replans the remaining historical verification route from the agent's current pose using the updated memory, rather than continuing the original verification order.

This produces a closed-loop interaction between retrieval and memory maintenance: retrieval exposes potentially stale historical information, revalidation identifies whether it remains valid, local revision replaces outdated small-object information, and the revised memory immediately affects subsequent navigation decisions.

### C.5 Memory Construction with Multi-View Inspection

When historical retrieval does not locate the target, IVAM-Nav switches to active exploration, during which new observations are continuously acquired and incorporated into the memory.

**Exploration and online perception.** Before the exploration loop, the agent performs an initial scan over up to seven viewing directions. The resulting RGB-D observations initialize or update the explored-area map, frontier candidates, and currently observed supporting surfaces. Repeated observations of the same supporting surface are fused into 3D instances and projected onto the top-down semantic map used for exploration and viewpoint generation.

At each exploration step, the agent considers frontier candidates together with up to three nearby supporting-surface candidates. When possible, the supporting-surface candidates are selected from different directions to reduce redundant alternatives. The MLLM receives the target-object, supporting-surface, and optional surroundings information together with the candidate observations and selects the next exploration action. When a reference image is available, it is used only to specify the target appearance. The corresponding decision prompt is provided in Appendix C.7.

If a frontier is selected, the agent navigates toward it and acquires observations along the trajectory and at the destination. These observations update the explored region, frontiers, and supporting-surface instances, after which the agent performs another exploration decision. If a supporting surface is selected, the agent approaches it and performs the dedicated multi-view observation procedure described below.

**Reliable and complementary viewpoint acquisition.** For a selected supporting surface, IVAM-Nav samples candidate viewpoints around its extent in the top-down semantic map and visits them sequentially. For each visited viewpoint, the reliable observation region is defined as the visible portion of the supporting surface observed within $1.5\,\mathrm{m}$ of the camera. Candidate viewpoints are sampled along the supporting-surface boundary with a $0.5\,\mathrm{m}$ step, and additional viewpoints are visited until the union of their reliable observation regions reaches a coverage ratio of $0.8$.

Because the visited viewpoints can contain substantial overlap, IVAM-Nav then applies the greedy complementary-viewpoint selection described in the main paper. At each step, the viewpoint providing the largest additional reliable surface coverage is selected until the required coverage is reached. This retains a compact set of complementary observations for subsequent semantic memory construction.

**Semantic description and graph update.** Before generating semantic descriptions from a selected viewpoint, IVAM-Nav uses its depth observation to suppress distant visual content. Pixels whose corresponding 3D points are farther than $1.5\,\mathrm{m}$ from the camera, as well as pixels with invalid depth, are overlaid with a dark-gray mask. The resulting full observation is resized to $640\times640$

Table C.1: Additional implementation settings of IVAM-Nav.

| Setting | Value |
|---|---|
| Historical retrieval threshold | $0.65$ |
| Nearby historical-view radius | $1.0\,\mathrm{m}$ |
| Candidate supporting surfaces per decision | up to 3 |
| Reliable observation range | $1.5\,\mathrm{m}$ |
| Reliable surface-coverage threshold | $0.8$ |
| Supporting-surface boundary sampling step | $0.5\,\mathrm{m}$ |
| Surface-node merge similarity threshold | $0.85$ |
| Opportunistic-observation radius | $2.5\,\mathrm{m}$ |
| Maximum additional path for opportunistic observation | $0.3\,\mathrm{m}$ |

and provided to the MLLM. The MLLM is instructed to ignore small objects entirely contained in the masked regions, while objects with clearly visible unmasked parts can still be described from those visible parts. The corresponding prompt is provided in Appendix C.7.

From each processed observation, the MLLM describes the visible movable small objects and their supporting surface. The distance mask focuses semantic memory construction on nearby visual information and reduces interference from distant visual content.

The resulting small-object and supporting-surface descriptions are encoded into semantic embeddings and used to update the corresponding viewpoint, small-object, and supporting-surface nodes in the graph. Supporting-surface descriptions obtained from different selected viewpoints are merged when their embedding similarity is at least $0.85$. Otherwise, they remain as separate surface nodes and their local relation is preserved in the graph.

**Opportunistic observation.** While acquiring multi-view observations of the selected supporting surface, IVAM-Nav can also observe nearby supporting surfaces when doing so requires little additional navigation. Specifically, supporting surfaces within $2.5\,\mathrm{m}$ are considered, and their observations are incorporated only when doing so adds at most $0.3\,\mathrm{m}$ to the current observation trajectory. These observations are also incorporated into the graph and can support memory retrieval in subsequent subtasks. This allows IVAM-Nav to accumulate useful information about nearby surfaces at little additional navigation cost, reducing repeated exploration and improving the efficiency of future navigation.

**Target verification.** After the selected views have been acquired, IVAM-Nav also checks them for the current navigation target. Since these observations are acquired under the current scene state, they can be used directly for target verification without historical-current consistency comparison. The normalized target description is used for verification, while an optional reference image, when available, is used only to specify the target appearance and not as information about the current scene. If the target is confirmed in any selected view, the agent stops; otherwise, it returns to frontier/supporting-surface action selection and continues exploration. The corresponding prompt is provided in Appendix C.7.

### C.6 Implementation Hyperparameters

Table C.1 summarizes the main implementation parameters introduced above. These parameters are fixed across all reported experiments.

### C.7 MLLM Prompts

We use Qwen3-VL (Bai et al., 2025) for all MLLM-based operations in IVAM-Nav. Below we provide the prompts used by the current full configuration at the main stages of the navigation pipeline. Runtime-dependent fields are enclosed by $\{\{\cdot\}\}$. Image placeholders indicate the corresponding multimodal inputs. A target reference image is optional for target-related MLLM operations; when-

ever it is provided, it is used only to specify the target appearance and never as information about the current scene.

**Target information preprocessing.** Before navigation, the specified target instruction is parsed into two parts: a target-object description and a placement description consisting of the supporting surface and optional nearby surroundings. We normalize the target object using the optional reference image when available, and separately normalize the supporting-surface and surroundings descriptions. The target-object normalization prompt is:

```
Normalize the target object description for retrieval. Return only strict JSON.
Use the same style as the scene small-object descriptions:
shape + color + distinctive appearance features + likely category if confident.
Distinctive features are visible cues such as a raised rim, handle, glossy
surface, animal-like silhouette, or bottle-like neck.
Describe only the target object itself. Do not mention support surface,
location, room, brand, or use.
Use the reference image if provided. If a visual attribute is not visible or
not stated, omit it; do not invent it.
JSON schema: {"object_description": string}.

Target label/description: {{RAW_TARGET_OBJECT_DESCRIPTION}}

[OPTIONAL TARGET REFERENCE IMAGE]
```

The supporting-surface and surroundings query is constructed using a separate text-only prompt:

```
Normalize the target support-surface description for retrieval. Return only
strict JSON.
Use the same style as the scene main support-surface descriptions:
color + texture/material + flatness + surface category if confident, for
example "dark polished flat wooden tabletop".
Extract only the real main support surface from the raw surface text. Do not
include nearby objects in surface_description.
Separately extract surrounding/nearby objects only when the raw surface text
explicitly says the target object is near, next to, beside, adjacent to, by,
or on the side of a medium or large object such as a sofa, couch, chair,
table, desk, cabinet, bed, shelf, counter, island, sideboard, nightstand,
wall, window, or door.
Do not put small movable objects in nearby_object_descriptions, such as
plates, cups, bowls, bottles, tools, books, papers, remotes, plants, baskets,
or lamps.
Do not extract objects that are only mentioned as surface appearance details,
examples, background context, or non-spatial cues.
Do not invent surface attributes or nearby objects. If there is no clear
support surface, use "no surface". If there are no explicit medium/large
nearby objects, return an empty nearby_object_descriptions list.
JSON schema: {"surface_description": string,
"nearby_object_descriptions": [string, ...]}.

Raw surface description: {{RAW_TARGET_SUPPORT_SURFACE_DESCRIPTION}}
```

**Active exploration and memory construction.** During active exploration, the MLLM selects between red-boxed nearby supporting-surface candidates and frontiers using the normalized target information and current visual observations. The current full configuration uses the surface-aware bounding-box prompt and does not request per-object numeric scores:

```
You are an indoor exploration decision assistant. The current task is to find
a target small object in the scene. Based on the target object, Target
support-surface description, red-boxed candidate object views, and frontier
views, choose the next action: move near a candidate object to observe it, or
explore a frontier.

Task question: {{TARGET_QUESTION}}
Target category/answer: {{TARGET_LABEL}}
Target text description: {{NORMALIZED_TARGET_DESCRIPTION}}
Target support-surface description:
{{NORMALIZED_TARGET_SUPPORT_SURFACE_DESCRIPTION}}
{{OPTIONAL_NEARBY_OBJECTS_LINE}}

[If provided:]
Target reference image: use it only to understand the appearance of the target
small object itself. Ignore the background, room type, support surface, and
surrounding objects in the reference image; do not treat them as evidence for
the current scene.
[OPTIONAL TARGET REFERENCE IMAGE]
```

```
"Target support-surface description" is the main text cue for the target's
placement in this scene; the reference-image background or surface does not
define that placement.
"Nearby objects around the target location" are auxiliary placement cues
only; they do not replace evidence from the target small object or its direct
support surface.
The reference image is only for target appearance, not evidence that the
target is visible here. Say target visible only if it appears in a candidate
object/frontier image.
For object views, the red box marks the candidate object that may be
approached, orbited, and checked; judge the object option mainly from the
red-boxed object/surface and its directly supported area.
Do not choose mainly by room type, furniture function, or common object
location; prioritize visible small items and whether visible surfaces match
"Target support-surface description" above.

Decision criteria:
1. Highest priority: if a target small object, or a highly suspicious target
small object, is already visible in an object/frontier view, choose that
option. For object views, the target-like item should preferably be on, above,
at the edge of, directly supported by, or very near the red-boxed
object/surface. If the target-like item is clearly elsewhere in the image, do
not use it as the main evidence for selecting that red-boxed object.
2. Next, prefer object views whose red-boxed object's visible surface matches
"Target support-surface description" above. Judge surface color, tone,
material, reflectiveness, and flatness. Even if the small object is unclear
because of distance, angle, or resolution, a visible surface on the red-boxed
object/surface that clearly matches this description should still be
considered worth approaching.
3. If object views do not show a visible red-boxed object/surface matching
"Target support-surface description", choose a frontier that may lead to such
a surface.
4. If there is no clear cue matching "Target support-surface description",
choose a frontier such as a doorway, corridor, opening, or room boundary that
can expand into a new area. Don't choose to go towards a closed door.
5. The target object is generally not on the floor.

{{OBJECT_OPTIONS_TEXT}}
{{FRONTIER_OPTIONS_TEXT}}

Do not output any per-object numeric scores; only choose the next action and
give a one-sentence reason.
The reason must briefly explain in one sentence the main basis for selecting
the choice_id, and it must be consistent with the final choice_id. The reason
must not use the reference image as evidence that the target is visible or
that a surface matches.
Output only strict JSON, no Markdown:
{"choice_id":"object_0/object_1/object_2 or frontier id",
"reason":"one-sentence reason"}

For every candidate object, the following text and image pair is appended:
The following image is labeled {{OBJECT_CHOICE_ID}} at the top; it is the
candidate object view for {{OBJECT_CHOICE_ID}}. The red box marks the candidate
object that may be approached, orbited, and checked.
[RED-BOXED OBJECT VIEW FOR {{OBJECT_CHOICE_ID}}]

For every frontier, the following text and image pair is appended:
The following image is labeled {{FRONTIER_CHOICE_ID}} at the top; it is the
frontier view for {{FRONTIER_CHOICE_ID}}.
[FRONTIER VIEW FOR {{FRONTIER_CHOICE_ID}}]
```

After an observation is acquired for memory construction, its depth image is used to darken pixels whose ray-Euclidean distance from the camera exceeds 1.5 meters, as well as pixels with invalid depth. Qwen3-VL receives one uncropped full observation at a time and extracts the visible small-object and supporting-surface semantics using:

```
You are an indoor scene object recognition assistant.
I will provide one uncropped full observation image, with image index 0.
Dark gray masked regions are farther than 1.5 meters or invalid depth; ignore
objects entirely inside them, but describe clearly visible unmasked parts.
Identify visible movable small objects only; do not list furniture, walls,
floors, shadows, textures, or background as items.
Do not miss small target-like objects such as fruit, balls, toys, toy bricks,
sponges, small food items, or brightly colored small block-like objects; if
uncertain, describe visible shape, color, material, and texture.
For support_surface_description, still use the actual visible support surface
directly touching those small objects; beds, sofas, cushions, bedding, tables,
or similar supports can all count.
```

```
For each object, write one short sentence in this order: shape + color +
distinctive appearance features + likely category if confident.
Distinctive features are visible cues such as a raised rim, handle, glossy
surface, animal-like silhouette, or bottle-like neck.
If category is uncertain, omit it. Do not mention location/support relations
or guess brand/use.
Also describe the actual surface directly touching the visible small objects
in this order when visible: color + texture/material + flatness + surface
category if confident, e.g. "dark polished flat wooden tabletop". Tables,
beds, sofas, cushions, bedding, or similar visible supports can all be support
surfaces. Do not infer it from nearby or background surfaces. Omit uncertain
attributes rather than guessing; if the surface category is unclear, describe
only visible attributes. Return "no surface" only if none of color,
texture/material, flatness, or surface category is clear.
Output only strict JSON. Do not output Markdown or extra explanation.
JSON format:
{
  "support_surface_description":
    "color + texture/material + flatness + confident surface category",
  "items": [
    {
      "name": "object category guess only if confident, otherwise generic shape name",
      "description":
        "shape + color + distinctive appearance features + confident category if any.",
      "image_indices": [0]
    }
  ]
}
Requirements:
1. Do not include duplicate objects in items.
2. description must be one sentence. Do not list multiple different objects
   separated by semicolons.
3. image_indices must be [0] for objects visible in the input image.
4. Always include support_surface_description.

[CURRENT FULL OBSERVATION WITH 1.5 METER DEPTH MASK]
```

**Historical memory retrieval.** Historical memory retrieval itself does not require an additional MLLM call in the current full configuration. The normalized target-object, supporting-surface, and surroundings queries above are embedded and matched against their corresponding graph information, followed by the relational information aggregation described in Sec. 4. The resulting high-ranked historical viewpoints are selected for revisit.

**Pose-matched memory revalidation.** After the agent returns to a retrieved historical viewpoint at its stored pose, the MLLM first compares the saved and currently reacquired observations. This is a dedicated consistency call: it receives neither the target description nor the optional target reference image and does not decide whether the target is present.

```
You are an indoor historical-view consistency checker. Compare only the
historical observation view and the currently reacquired view:
Judge whether the distribution of visible small items / movable items in the
current view is consistent with the historical view, focusing on changes in
item count, relative positions, and obvious appearance.
Do not miss small items, such as cups, plates, small blocks, toys, fruit, or
small tools.
If the current image has obvious new objects, or if an obvious historical
object is missing, moved clearly, or changed appearance clearly, set
view_consistent=false.
Focus on visible items and their distribution. Do not judge true merely
because room structure, furniture layout, or camera angle is similar.
Do not judge whether the target exists; this step only checks view consistency.
Keep reason concise: at most 25 English words, stating only the key evidence.
Output only strict JSON:
{"view_consistent":true,"confidence":0.0,"reason":"..."}.

Image 1: historical observation view, used to compare consistency with the
currently reacquired view.
[SAVED HISTORICAL OBSERVATION]

Image 2: currently reacquired view. Use it only to judge consistency with the
historical view.
[CURRENTLY REACQUIRED OBSERVATION]
```

**Target confirmation.** Target confirmation is performed separately from the consistency check. During a successful consistent historical revisit, after a refreshed inconsistent revisit, or after newly acquired multi-view inspection observations, the MLLM determines whether the target is present in

the current observation. The normalized target description is always provided. When an optional reference image is available, it is used only to specify the target appearance.

```
You are an indoor target confirmation assistant. Judge only whether the target
small object, or a similar target object, is visible in the current view.
Use the target reference image and text only to understand the target
appearance; target_present must be judged only from the current view.
Ignore the reference image background, room type, support surface, and
surrounding objects; learn only the target object itself.
Consider category, color, overall shape, and key structure; do not reject a
similar target lightly due to viewpoint, occlusion, or distance.
Keep reason concise: at most 25 English words, stating only the key evidence.
Output only strict JSON:
{"target_present":false,"confidence":0.0,"reason":"..."}.

Task question: {{TARGET_QUESTION}}
Target category/answer: {{TARGET_LABEL}}
Target text description: {{NORMALIZED_TARGET_DESCRIPTION}}

[If provided:]
Target small-object reference image. Use it only to understand the target
object itself. Ignore the background, room type, support surface, and
surrounding objects; do not use it to judge whether the target is currently
visible.
[OPTIONAL TARGET REFERENCE IMAGE]

Current view: target_present must be judged only from this image.
[CURRENT OBSERVATION]
```

# D Additional Experimental Details

## D.1 Analysis Subset

**Representativeness of the analysis subset.** The analysis subset contains 29 lifelong navigation sequences and 245 subtasks, approximately one fifth of the full LiSoNav-Eval validation set. The proportions of subtasks across four successive episodes are 28.16%, 28.57%, 29.39%, and 13.88%, closely matching 28.61%, 28.69%, 29.36%, and 13.34% in the full benchmark, with a total variation distance of only 0.56%. The object-size composition is also nearly identical, with the proportions below and above 1 L differing by only 0.09 percentage points. Relocated targets account for 47.76% of the subset, compared with 45.45% in the full benchmark. These statistics indicate that the analysis subset preserves the key characteristics of the full benchmark.

## D.2 Baseline Adaptation Details

**Common protocol.** Since the baselines were originally developed for different task interfaces and execution platforms, we adapt them to the same Habitat-Sim evaluation protocol. All methods use the navigation instructions defined in LiSoNav-Eval and are evaluated with the same object configurations, starting poses, success criterion, and movement budget. For methods coupled to platform-specific execution stacks, we execute their selected navigation goals through the common Habitat-Sim interface while leaving their high-level goal selection unchanged. Additional method-specific details are provided below.

**VLFM.** VLFM is evaluated directly under the common protocol above without additional method-specific adaptation.

**AstraNav.** We directly use the released AstraNav-Memory checkpoint in a zero-shot manner without additional training. Its memory is retained throughout each lifelong navigation sequence so that historical observations can be reused across successive subtasks and episodes.

**DynaMem.** DynaMem is originally designed for open-world mobile manipulation, including object search, navigation, and manipulation. Since LiSoNav-Eval evaluates navigation, we use its search and navigation pipeline, with its voxel memory retained throughout each lifelong navigation sequence. Once DynaMem selects a navigation goal, we execute the complete path returned by its planner before the next high-level decision. This follows the same execution protocol as our method, where a selected frontier or supporting surface is navigated to before the next high-level decision. The navigation goal and planned path remain determined by DynaMem.

**3D-Mem.** We retain its memory throughout each lifelong navigation sequence and append the LiSoNav-Eval small-object categories to its open-vocabulary detection vocabulary, since otherwise some benchmark targets cannot be queried by its perception module. After 3D-Mem selects a snapshot or frontier, we execute the complete planned path to the selected destination before the next high-level decision. This follows the same execution protocol as our method, where a selected frontier or supporting surface is navigated to before the next high-level decision. The snapshot/frontier selection and planned path remain determined by 3D-Mem.

**DualMap.** DualMap assumes a preconstructed scene map, so evaluating it from empty memory would remove a prerequisite of its original setting. We therefore initialize its map before evaluation by traversing each environment along a predefined route. The route is independent of the navigation instructions and evaluated trajectories, is traversed once before the first subtask, and its travel distance is excluded from subtask SR and SPL. The resulting map is retained and updated throughout the lifelong navigation sequence. DualMap still determines the navigation destination, while the common Habitat-Sim navigation interface is used only to execute the selected endpoint.

**Oracle variants.** For the Oracle variants of DynaMem and 3D-Mem, ground-truth semantic detections within 1 m of the agent replace only the corresponding small-object detection calls. The subsequent memory operations, candidate selection, planning, and navigation remain the same as in their non-Oracle counterparts, and the target location is never directly provided to the agent.

### D.3 Static-scene Evaluation Setting

**Static-scene setting.** To isolate the effect of object relocation, we construct a static counterpart for each evaluated LiSoNav-Eval sequence using its final object configuration. Specifically, the final object layout is loaded at the beginning of the first episode and kept unchanged throughout the entire sequence, so objects do not change locations between episodes. The target information for each subtask, including the contextual cues in its navigation instruction, is correspondingly taken from the final object configuration, ensuring that the instruction remains consistent with the target's actual placement. The episode and subtask structure, subtask order, episode starting poses, model configurations, memory persistence, success criterion, and other evaluation settings remain unchanged. Importantly, the static setting changes only the external scene configuration and its corresponding target information; agents continue to navigate, acquire observations, and update their internal memories normally. This controlled comparison therefore removes cross-episode object relocation while preserving the remaining navigation and memory processes.

### D.4 Benchmark Factor Definitions

We use simple and interpretable physical thresholds to characterize benchmark difficulty while maintaining sufficiently large subsets for reliable comparison. Table D.1 summarizes the resulting groups.

Table D.1: Statistics of the subsets used for benchmark analysis.

| Factor | Subset | # Subtasks |
|---|---|---|
| Object size | Small ($< 1\,\mathrm{L}$) | 833 |
| | Large ($\geq 1\,\mathrm{L}$) | 366 |
| Environment scale | Small ($< 32\,\mathrm{m}^2$) | 323 |
| | Medium ($32$–$< 64\,\mathrm{m}^2$) | 540 |
| | Large ($\geq 64\,\mathrm{m}^2$) | 336 |
| Relocation distance | Short ($< 7.5\,\mathrm{m}$) | 267 |
| | Long ($\geq 7.5\,\mathrm{m}$) | 278 |

Object size is measured by bounding-box volume, with 1 L used as an interpretable threshold separating smaller and larger household objects. Environment scale is measured by the navigable area on the starting floor; 32 and $64\,\mathrm{m}^2$ provide three interpretable ranges while retaining substantial

numbers of subtasks in each group. Relocation distance is measured by geodesic displacement, with 7.5 m yielding two comparably sized groups.

### D.5 Historical Revisit and Prior-Observation Criteria

We use a common ground-truth observation criterion for Historical Revisit (HR) and the prior-observation analysis. Whether a target is observed is determined from simulator camera observations using ground-truth semantic segmentation, depth, and camera poses, independently of the perception modules of the evaluated methods.

A target is considered observed when it occupies at least $2.5 \times 10^{-4}$ of the image area and at least 50% of its visible pixels lie within the central 80% of the image. For HR, a subtask is eligible if the target has at least one valid observation from an earlier subtask. We collect the corresponding camera positions and measure whether the current trajectory enters a 1.5 m neighborhood of any such historical observation position. Observations acquired during the current subtask are excluded.

**Prior target observation.** We further examine whether relocated targets were observed before relocation. Prior observations depend on each method's navigation history and are determined from simulator camera observations using ground-truth semantic information, following the same criteria as HR. As shown in Table 8, the resulting probabilities remain substantial across methods despite differences in exploration behavior. This indicates that object relocation frequently has the potential to make previously acquired target information stale.

## E Further Discussion on Limitations and Future Directions

We further discuss the limitations of IVAM-Nav and LiSoNav-Eval described in the main paper and possible directions for future research.

**Balancing active observation and navigation cost.** IVAM-Nav actively visits multiple viewpoints around selected supporting surfaces to acquire reliable and complementary small-object observations, which introduces additional navigation cost. Future methods could more selectively determine when additional observations are necessary and how many viewpoints are sufficient. For example, an agent could estimate whether the current observations already provide sufficient small-object information and acquire additional views only when needed. Viewpoint selection could also jointly consider observation gain and travel cost, favoring complementary views without large detours. More generally, memory construction could be integrated with navigation planning so that observations useful for both the current task and future memory are acquired along efficient trajectories.

**Broader and more efficient memory revalidation.** IVAM-Nav primarily revalidates historical viewpoints retrieved for the current target, so outdated information outside these candidates may not be updated. Future methods could maintain a broader portion of accumulated memory without requiring frequent dedicated revisits. Promising directions include exploiting observations naturally acquired during navigation, selectively identifying memories that require revalidation, and jointly considering the benefit and navigation cost of memory maintenance. Such mechanisms could enable more comprehensive memory updates while limiting additional navigation overhead.

**Integrating memory reuse and exploration.** IVAM-Nav follows a history-first pipeline, revisiting relevant historical viewpoints before switching to exploration when useful candidates are unavailable or exhausted. Future methods could instead allow memory reuse and exploration to interact throughout navigation. New observations could update the relevance of historical viewpoints, while partial historical information, such as matching surfaces or surroundings, could guide exploration toward promising regions. A unified policy could therefore dynamically balance historical revisit and new exploration based on the available information rather than making a one-way transition between them.

**Extending to broader scene changes.** LiSoNav-Eval relocates small objects while keeping supporting surfaces and the background scene stable. Future benchmarks could introduce changes to supporting surfaces, furniture, or other scene elements, which may invalidate not only stored small-object information but also relations in the viewpoint-anchored graph memory. Such settings would require agents to identify which parts of the scene have changed and selectively revise the corre-

sponding memory, motivating more flexible memory representations and maintenance mechanisms for complex dynamic environments.