HIREX: HIERARCHICAL RETRIEVAL AND ACTIVE OBSERVATION OVER VISUAL MEMORY FOR EMBODIED EXPLORATION AND REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Embodied exploration and reasoning is the capability to explore an environment to answer questions or reach specified goals. Recent zero-shot methods build a scene representation from observations, over which a vision-language model (VLM) reasons to answer and to decide where to explore. However, existing representations fall short. 3D scene graphs abstract the scene into objects and relations, discarding visual evidence before the task is given. Image-set methods preserve the evidence but retrieve it through detected classes, inheriting their errors. Both leave exploration separate from retrieved evidence. To address these issues, we propose HiREx, a scene memory that organizes the image set under room, view, and object layers, on which both retrieval and exploration operate. Views are selected to preserve object co-visibility, rooms index the views as semantic anchors, and object crops supply fine-grained details. Given a task, retrieval narrows the evidence through the layers, and the level at which evidence is found activates exploration options at that level. Across question-answering and goal-navigation benchmarks, our method outperforms all baselines and shows that memory, retrieval, and exploration each contribute to embodied exploration and reasoning.

1 INTRODUCTION

Embodied exploration and reasoning refers to the capability to explore and understand an environment to answer questions or reach specified goals (Das et al., 2018; Majumdar et al., 2024; Khanna et al., 2024), and is fundamental for embodied agents. It requires the agent to reason over accumulated observations and to move for information it still lacks. With vision-language models (VLMs) (Hurst et al., 2024; Bai et al., 2025) handling reasoning over complex queries, recent work has focused on improving the scene representation that VLMs reason over, namely how observations are organized and maintained.

Since the task arrives only after the representation is built, the representation should preserve observed evidence in a task-agnostic manner, while supporting access to a compact, task-relevant subset for reasoning. Moreover, efficient exploration requires using what the representation already holds, so that the agent seeks the information it still lacks. From these insights, we consider three conditions for effective embodied exploration and reasoning. (1) The representation preserves visual evidence while allowing compact VLM input. (2) Given a task, it identifies task-relevant information from the representation. (3) It determines an efficient exploration option using the current scene representation as a cue.

3D scene graphs, a common representation for this task, abstract object properties, inter-object relations, or scene hierarchy into a graph (Armeni et al., 2019; Rosinol et al., 2021; Gu et al., 2024). This abstraction makes them more compact than dense representations such as point clouds (Peng et al., 2023) or neural fields (Kerr et al., 2023), suiting long-horizon exploration in complex environments. However, since the scene is simplified before the task is given, visual evidence needed for the task may be lost. Recent methods instead construct an informative image set from observations and reason directly over it (Yang et al., 2025), preserving visual evidence. Their retrieval, however, relies on the classes detected in each image, and inherits detection errors. Moreover, these methods

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.

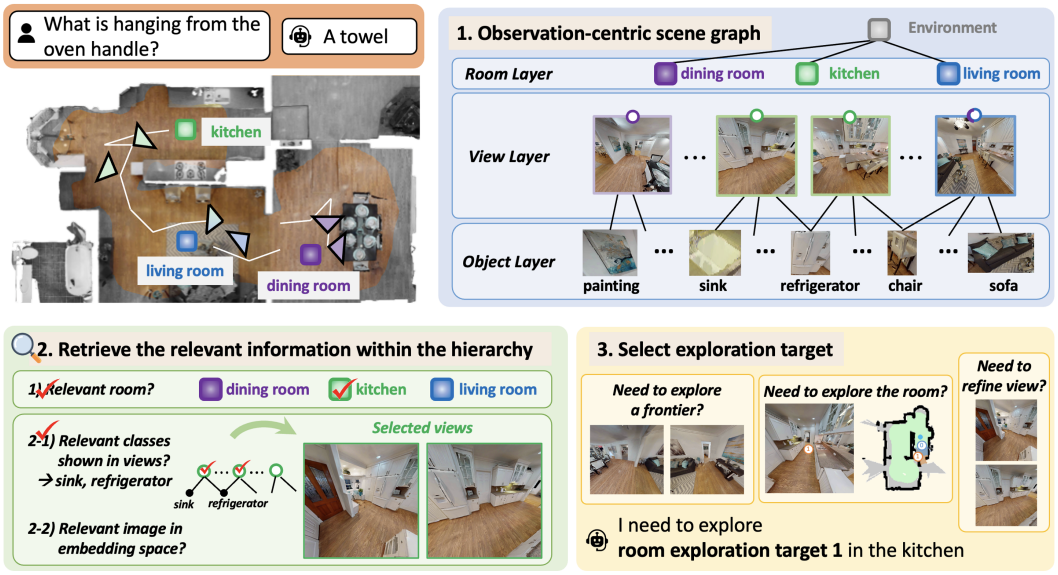


Figure 1: We propose HiREx, which organizes observed images into an observation-centric scene graph (room–view–object), providing room type as semantic anchors and object co-visibility as visual evidence. Given a task, relevant information is retrieved across hierarchy levels, and the retrieved results activate exploration candidates at the corresponding level. Exploration is structured by the representation itself.

treat exploration separately from the representation (Yang et al., 2025; Wang et al., 2026), offering limited support for reasoning about where to explore based on what is known.

To address these gaps, we organize the image-set memory under room, view, and object layers, on which both retrieval and exploration operate. We call this structure an observation-centric scene graph. Fig. 1 summarizes our method. The room layer organizes views by the room types they observe, providing semantic cues that disambiguate the lower layers. The view layer is a sampled image set. It preserves visual context, including object co-visibility (multiple objects visible in the same image) and the surrounding background, which the VLM can use to infer context. We especially aim to capture more diverse object co-visibility than a previous method (Yang et al., 2025). The object layer stores a crop from the view that best observes each object and is used when the task specifies a target object, as in goal-oriented navigation (Khanna et al., 2024).

As memory grows during exploration, we retrieve a compact subset for the VLM to reason over. Room retrieval restricts view candidates to task-relevant rooms, excluding views of similar objects in irrelevant rooms. Among these candidates, views observing task-relevant object classes are retrieved. Views whose embeddings match the query embedding are then added, reaching relevant views that class labels alone can miss. Exploration options are then activated at the levels where evidence is retrieved. From retrieved rooms, we construct options that explore unvisited positions within the room. From retrieved views, we let the agent refine its viewpoint to better observe task-relevant regions.

Experiments on embodied question answering (Majumdar et al., 2024), in both episodic-memory and active settings, and on multi-modal goal navigation (Khanna et al., 2024) show that the memory, retrieval, and exploration components each contribute to embodied exploration and reasoning. Our contributions are as follows:

- We propose HiREx, which organizes observed images into a layered scene memory, where retrieval and exploration operate as a single loop.
- We introduce a memorization strategy that selects views by the combinations of co-observed objects they capture, keeping visual evidence that later tasks may need.

- We design a retrieval scheme that narrows the search by room and complements class-based view retrieval with task–view matching, reaching views that class labels alone can miss.
- We propose an adaptive exploration strategy in which exploration options are activated at the levels where task-relevant evidence is retrieved.

2 Related Work

Scene Representation for Embodied Agents. To support embodied tasks, recent works have grounded 2D semantic features in 3D space (Peng et al., 2023; Takmaz et al., 2023; Zhang et al., 2023; Kerr et al., 2023), supporting open-vocabulary segmentation and grounding. However, exploration keeps expanding the scene, and maintaining such dense maps grows computationally heavy. 3D scene graph-based representations (Gu et al., 2024; Wu et al., 2021; Wald et al., 2020; Yin et al., 2024) instead abstract the scene into objects and relations, staying compact even as exploration updates the graph. Hierarchical variants further organize the graph under levels such as floors and rooms (Rosinol et al., 2021; Werby et al., 2024; Zhou et al., 2025), and their nodes hold abstracted features rather than the observations. Since the abstraction is fixed before the task is given, visual evidence the task needs may already be discarded. In OpenEQA (Majumdar et al., 2024), agents reasoning over scene-graph captions fall behind a VLM reading the frames directly. To preserve the evidence, recent methods keep the images themselves. 3D-Mem (Yang et al., 2025), built on the same object registration (Gu et al., 2024), maintains an informative subset of observed images, and SplatReasoner (Kim et al., 2026) extends the stored views with novel view synthesis. Images have also been attached to abstract scene graphs, as a few globally similar frames (Saxena et al., 2024) or as attributes of object-pair edges (Huang et al., 2026). In these designs, however, the stored images are accessed through detected objects at query time. Compared with these approaches, we build a scene memory that organizes the observed images in a hierarchy, where retrieval operates directly on the stored observations and reaches views that class labels alone can miss.

VLM-Based Reasoning and Exploration Policies. Vision-language models (VLMs) (Hurst et al., 2024; Bai et al., 2025; Hu et al., 2025) have become a common foundation for embodied exploration and reasoning. Existing methods are divided into end-to-end and modular designs. End-to-end methods (Ramrakhya et al., 2023; Wijmans et al., 2022; Zhu et al., 2025) directly predict low-level actions from egocentric observations, but their generalization remains limited under current data and model scales (Hong et al., 2021; Yu et al., 2023). Modular methods instead decouple high-level reasoning from low-level control through intermediate representations, drawing on pretrained VLM reasoning. Some methods (Ren et al., 2024; Yokoyama et al., 2024; Shah et al., 2023; Jiang et al., 2025) build task-conditioned semantic maps from observed targets to guide exploration, and such maps are rebuilt when the objective changes. Others (Yang et al., 2025; Wang et al., 2026; Ziliotto et al., 2025) maintain task-agnostic memory and delegate next-target selection to a VLM over candidate options at each step. HGR (Chen et al., 2026) further records semantic predictions about frontiers and revises them against later observations. What has been found for the task also indicates where further evidence may lie. In these methods, however, the exploration options stay the same no matter what has been found for the task. Fine-EQA (Jiang et al., 2025) partially addresses this via room-level triggering, but still depends on task-conditioned maps. In contrast, we retrieve task-relevant evidence from a task-agnostic scene memory, and the retrieved level activates exploration options at the corresponding level.

3 Method

We build an observation-centric scene graph from the agent's observations, and both retrieval and exploration operate directly on it. Retrieval proceeds top-down through the layers, progressively narrowing the search space, and the level at which evidence is found activates the corresponding exploration options. Fig. 2 illustrates the overall pipeline.

3.1 Observation-Centric Scene Graph

Our scene graph is observation-centric, meaning that the stored content is the observations themselves (Fig. 1, top right). The view layer keeps selected images, the object layer keeps object crops from them, and the room layer indexes these observations by the room types they observe.

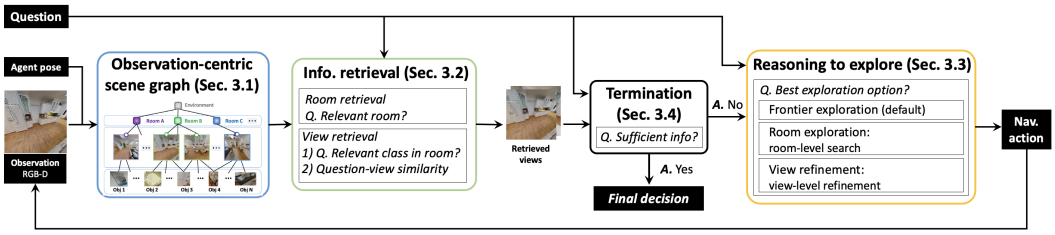


Figure 2: Pipeline of HiREx. At every step, the observation-centric scene graph (Sec. 3.1) is updated with new observations. Given a question or goal, relevant views are retrieved from the graph (Sec. 3.2), and a VLM judges whether they suffice to answer (Sec. 3.4). If not, exploration options are constructed at the level where evidence was retrieved—room exploration, view refinement, or the default frontier exploration—and the VLM selects among them (Sec. 3.3).

Object layer. Given a sequence of RGB-D images and camera poses, we first update the object set in the scene following (Gu et al., 2024). Each object node is defined as the object-centric crop from the image that best captures the object, preserving fine-grained visual details. The crops are used when the task specifies a target object, as in goal-oriented navigation. From the detection results, a visibility mask $\mathcal{M}$ is updated, where $\mathcal{M}_{i,j} = 1$ if the j-th object o_j is detected in the i-th image v_i, and 0 otherwise. Edges between object and view nodes are defined by $\mathcal{M}$ (Appendix A.1.1).

View layer. We treat object co-visibility, defined as multiple objects appearing in the same view, as information to preserve. Views observing the same object can carry different information depending on which objects appear together, and a combination discarded here cannot be recovered at retrieval. To keep such combinations, we group objects by agglomerative clustering (Johnson, 1967), which merges co-observed objects into progressively larger clusters. We merge clusters by the similarity $\mathrm{Sim}(C_i, C_j) = |V(C_i \cup C_j)|/|V(C_i) \cup V(C_j)|$, where $V(C)$ is the view set of a cluster, defined as:

$$V(C) = \begin{cases} \{v_i \mid \mathcal{M}_{i,j} = 1\} & \text{if } C = \{o_j\} \\ \bigcap_k \{v_i \mid \exists o_j \in C_k : \mathcal{M}_{i,j} = 1\} & \text{otherwise} \end{cases}, \tag{1}$$

where $|\cdot|$ is the number of elements of a set, and C_k is the k-th child of C. Eq. 1 associates a view with a cluster if it observes any object from each child. This condition is easily met even as clusters grow, compared to requiring every object in the cluster. Merging continues until no pair of clusters share a view, i.e., $\mathrm{Sim}(C_i, C_j) = 0$ for all pairs.

We assign views by traversing the merged clusters in inverse order. For each cluster C, we select the view $v_C \in V(C)$ least similar to the previously selected views, where the similarity is measured with CLIP features aggregated over the objects in C. If no sufficiently distinct view exists, the cluster is pruned, and its children are reattached to the parent. Selecting at most one view per cluster keeps the view layer compact, and the dissimilarity rule aims to preserve, within this budget, the object groups that the clusters' views show. Algorithm 1 summarizes the construction.

During exploration, the clustering and representative views are updated incrementally, re-clustering only newly and re-observed objects while keeping high-level clusters fixed (Appendix A.1.2).

Room layer. The room layer indexes views by the room types they observe. Every time a view v_j is added to the view layer, a VLM predicts the room types observed in it and the room affiliation of each detected object. The latter is accumulated across views via a Bayesian update, $P(r_i = c|v_j) = P(v_j|r_i = c)\,P(r_i = c) \,/\, \sum_{c'} \big(P(v_j \mid r_i = c')\,P(r_i = c')\big)$, where r_i is the room assignment of object o_i and c denotes the room index. A view is linked to a room if the VLM predicts the room from the view directly, or if the view contains objects assigned to that room. Combining the two sources links more views to rooms than either alone, aiming to keep candidate views reachable at retrieval (Appendix A.1.3).

3.2 Information Retrieval

While the scene graph retains information about the entire environment for a wide range of tasks, retrieval narrows it to task-relevant information. Given a question or goal, an LLM selects up to k

Algorithm 1 View Layer Construction

Require: Cluster set $\mathcal{C}$, child map Φ, co-visibility matrix $\mathcal{M}$, object-wise CLIP features $\mathcal{F}$, similarity threshold η
Ensure: View layer $\mathcal{V}$, updated $\mathcal{C}$ and Φ

$\mathcal{C}_{new} \leftarrow \emptyset$
while $\max_{C_a, C_b \in \mathcal{C}} \mathrm{Sim}(C_a, C_b) > 0$ **do** ▷ Sim with $V(C)$ of Eq. 1
  $(C_i, C_j) \leftarrow \arg\max_{C_a, C_b \in \mathcal{C}} \mathrm{Sim}(C_a, C_b)$
  $C_m \leftarrow C_i \cup C_j$; $\Phi(C_m) \leftarrow \{C_i, C_j\}$
  Replace C_i, C_j with C_m in $\mathcal{C}$; $\mathcal{C}_{new} \leftarrow \mathcal{C}_{new} \cup \{C_m\}$
for $C \in \mathcal{C}_{new}$, **in inverse merge order do**
  Compute $f(v)$ for $v \in V(C) \cup \mathcal{V}$ by Eq. 3 (Appendix A.1.2)
  $s(v) \leftarrow \max_{v' \in \mathcal{V}} \cos(f(v), f(v'))$ for $v \in V(C)$
  if $s(v_C) < \eta$ where $v_C \leftarrow \arg\min_{v \in V(C)} s(v)$ **then**
    $\mathcal{V} \leftarrow \mathcal{V} \cup \{v_C\}$
  else
    Replace C with $\Phi(C)$ in $\Phi(\mathrm{parent}(C))$ ▷ prune C
return $\mathcal{V}, \mathcal{C}, \Phi$

relevant rooms, matching the view budget. It receives room types in the room layer and the object classes assigned to each room, with per-class counts, so rooms can be identified by type or by contained objects. The candidate views are restricted to those linked to the selected rooms, excluding views of similar objects in irrelevant rooms. If no room is selected, the candidate set falls back to all stored views.

At the view layer, retrieval proceeds along two paths. The first path is class-based, as in prior image-set methods (Yang et al., 2025). The LLM selects task-relevant classes among those observed in the candidate views, ranking them by relevance. This ranking determines the views. Views are greedily selected to cover the ranked classes, taking at each step the view observing the most objects of the highest-ranked class not yet covered, breaking ties by the number of other relevant classes observed.

Class-based retrieval, however, matches views only through detected class labels, inheriting detection errors and missing what the labels do not capture. We therefore add a second, similarity-based path, matching the task and views directly in a joint embedding space. Task–view similarity is computed with Qwen3-VL-Embedding-2B (Li et al., 2026), an embedding model built on Qwen3-VL (Bai et al., 2025) and designed for retrieval with queries beyond short captions, such as questions and instructions. View embeddings are computed once when a view enters the memory and are cached, so retrieval compares the task embedding against the stored ones. The class-based path selects up to k views, and adds the most task-similar remaining view if its similarity exceeds a threshold and exceeds the best class-retrieved view by a margin, replacing the last one when the budget k is full. Appendix B.2 compares combinations of the two paths, and additional details are shown in Appendix A.2.

3.3 Reasoning for Exploration

When the retrieved evidence does not suffice, the agent proceeds to exploration. Each retrieved level then serves as a cue for where to observe further. This is because evidence at a level indicates the task relates to that room or view, and observing around it can add related evidence. Exploration options, each an active observation at its level, are thus activated at the levels where evidence is retrieved. We instantiate three option types, room exploration, view refinement, and frontier exploration, and the VLM selects among those activated.

Since the view layer keeps the observed images, room and view options can be presented to the VLM with the views observing them. Alongside the graph, two geometric maps support option construction. An occupancy map models navigable space, with explored and unexplored regions defined following prior frontier-based exploration (Yang et al., 2025; Chen et al., 2026). A room segmentation map is maintained by propagating SAM3 (Carion et al., 2025) predictions from views to the top-down map.

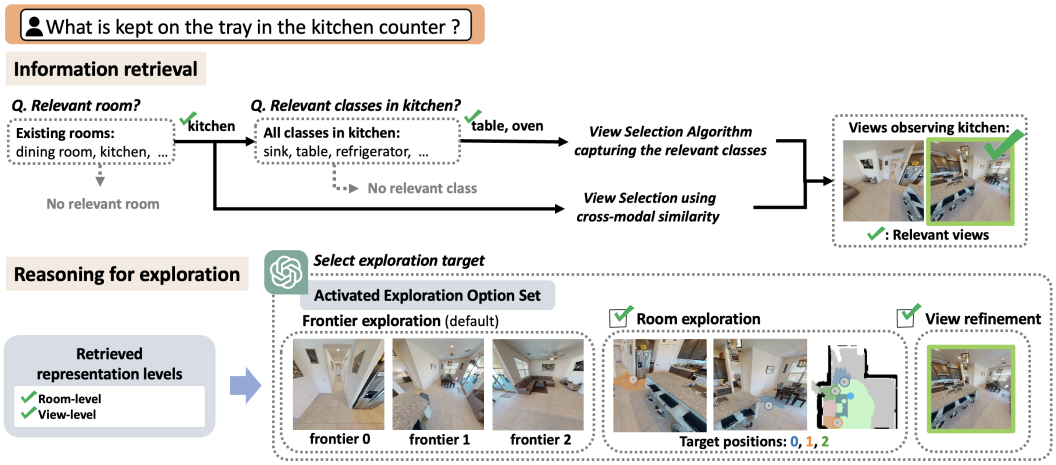


Figure 3: Retrieval and exploration on an example question. An LLM is asked which rooms, then which classes in them, are relevant to the question. Views covering the answered classes are selected, and a view with high query similarity is added when complementary. Each retrieved level activates its exploration option. Room-level activates room exploration, view-level activates view refinement, and frontier exploration is the default. The VLM selects among them.

Room exploration. This option is activated when rooms are retrieved, and the agent is located in one of them. We cluster unexplored regions within the room under a predefined radius r, and compute a target position for observing each cluster. We present the option as the views connected to the room, marked with the projected clusters and their target positions. The occupancy map is also provided, showing all clusters with the agent position under shared option indices.

View refinement. This option is activated when views are retrieved. A retrieved view may observe a task-relevant region yet remain insufficient, such as appearing too small, lacking surroundings, or occluded from the current viewpoint. We define three adjustment strategies accordingly: closer view, wider context, and different angle. If the VLM selects this option, it outputs the index of the view to refine, a description of the region to observe, and the strategy. We apply SAM3 to obtain a mask of the described region, and a preset function for the strategy computes the target position from it.

Frontier exploration. Frontier exploration is the default option, activated regardless of the retrieval result. We follow Yang et al. (2025) to update frontier regions and construct images observing them. In addition, we match current frontiers with previously selected ones by position and orientation. The prompt instructs the VLM to prefer the previously selected frontier unless another offers significantly greater potential to gather task-relevant information.

We restrict the VLM to selecting an exploration option, decoupling semantic decisions from the computation of target coordinates, as VLMs remain unreliable at numerical reasoning (Davoodi et al., 2025; Mirzadeh et al., 2024) (Appendix A.3).

The observations gathered by exploration are added to the memory, so retrieval and exploration repeat over the single growing structure.

3.4 Termination Check

We query the VLM to determine whether to stop or continue exploration, given the task and the retrieved views. For question answering, we provide only the views. For goal-oriented navigation, the object layer supplies the crops of objects observed in the retrieved views, and the VLM terminates by selecting the matching object among them. We query termination separately from the selection of exploration options, keeping the two decisions apart.

Table 1: **Experiments on EM-EQA.** We report LLM-Match for QA performance and the average number of frames for frame efficiency. With both closed (GPT-4o) and open (Qwen3-VL-8B-Instruct) VLMs, our method outperforms all baselines at a frame cost comparable to the frame-based baselines. Retrieval is run per VLM, so frame counts are closed / open. Baseline numbers are from SplatReasoner (Kim et al., 2026).

Method	LLM-Match (↑)		Avg. Frames
	Closed Model	Open Model	
BlindLLM	34.8	29.4	0
Frame Cap.	24.1	31.7	0
CG Cap.	36.5	-	0
SVM Cap.	38.9	-	0
Multi-Frame	49.1	48.2	3.0
3D-Mem (CVPR'25)	54.6	50.8	2.7
SplatReasoner (ECCV'26)	57.8	51.6	2.6
HiREx (Ours)	**65.1**	**59.0**	2.7 / 2.5
Human	86.8		Full

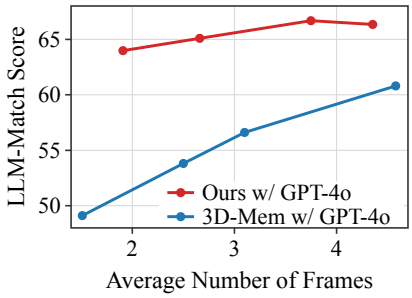


Figure 4: **Frame Efficiency of HiREx on EM-EQA.** Similar to 3D-Mem, we vary the number of retrieved entries k to obtain LLM-Match scores under different numbers of frames used for answering. HiREx consistently outperforms 3D-Mem at comparable frame budgets across the evaluated range.

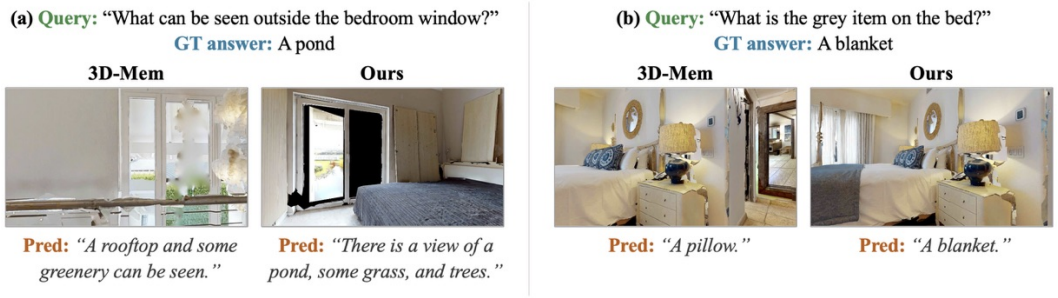


Figure 5: Qualitative comparison on EM-EQA episodes between 3D-Mem and our method. (a) The room layer excludes a window detected in another room, and the view retrieval selects the correct view despite the missed detection. (b) Our view layer keeps the wider view where the bed and its surroundings are co-visible, while 3D-Mem crops the item in question.

4 Experiments

4.1 Experimental Settings

Implementation Details. We use GPT-4o (2024-11-20) as the VLM throughout our system, except for room assignment, where we use GPT-4o-mini (2024-07-18). Once the target position is determined, we assume a collision-free trajectory planner navigates the agent to that location. In practice, we use the built-in pathfinder in Habitat Sim, following VLM-guided exploration approaches (Ren et al., 2024; Yang et al., 2025). The per-step runtime of each component is reported in Appendix B.3. Additional details are shown in Appendices A and C.

Benchmarks. We evaluate on three benchmarks covering different aspects of embodied exploration and reasoning. EM-EQA, a subset of OpenEQA (Majumdar et al., 2024), poses over 1600 questions over pre-recorded RGB-D trajectories with camera parameters, isolating representation and retrieval from exploration. A-EQA, another subset of OpenEQA, also requires active exploration; following the OpenEQA evaluation protocol and due to resource limitations, we evaluate the official 184-question subset. GOAT-Bench (Khanna et al., 2024) targets navigation toward goals specified by object category, language, or image, with multiple subtasks per episode; following 3D-Mem, we use a 1/10 subset of the Val Unseen split.

Table 2: **Experiments on A-EQA.** Methods marked with † are reproduced with their official code using our GPT-4o release.

Method	LLM-Match ↑	LLM-Match SPL ↑
CG Cap.	34.4	6.5
SVM Cap.	34.2	6.4
Frame Cap.	38.1	7.0
Multi-Frame	41.8	7.5
Explore-EQA (RSS'24)	46.9	23.4
TANGO (CVPR'25)	37.2	–
3D-Mem† (CVPR'25)	49.6	37.0
HGR† (ECCV'26)	52.0	39.8
HiREx (Ours)	**57.5**	**48.6**

Table 3: **Experiments on GOAT-Bench.** Methods marked with † are reproduced with their official code using our GPT-4o release.

Method	SR ↑	SPL ↑
Modular GOAT	24.9	17.2
Modular CoW	16.1	10.4
SenseAct-NN Skill Chain	29.5	11.3
SenseAct-NN Monolithic	12.3	6.8
Explore-EQA (RSS'24)	55.0	37.9
TANGO (CVPR'25)	32.1	16.5
3D-Mem† (CVPR'25)	69.4	50.7
MTU3D (ICCV'25)	47.7	27.7
HGR† (ECCV'26)	67.6	50.1
HiREx (Ours)	**76.6**	**53.1**

Metrics. For both EM-EQA and A-EQA, we adopt LLM-Match proposed in OpenEQA, where GPT-4 scores the predicted answer against the ground truth on a 0-100% scale, and for A-EQA we additionally report LLM-Match SPL, which weights LLM-Match by the ratio of the reference to the predicted path length. For GOAT-Bench, we report Success Rate (SR) and Success weighted by Path Length (SPL), where a subtask is successful if the agent stops within 1 meter of the goal.

Baselines. For EM-EQA, the baselines span diverse memory designs, including abstraction-based (ConceptGraph Captions, Gu et al., 2024), dense 3D (Sparse Voxel Maps Captions, Majumdar et al., 2024), image-set based (Multi-Frame and 3D-Mem, Yang et al., 2025), and novel view synthesis based (SplatReasoner, Kim et al., 2026) methods. For A-EQA and GOAT-Bench, they include end-to-end policies (MTU3D, Zhu et al., 2025) and VLM-guided methods, Explore-EQA (Ren et al., 2024), TANGO (Ziliotto et al., 2025), 3D-Mem, and HGR (Chen et al., 2026).

4.2 Results

EM-EQA. We evaluate our representation and retrieval on the EM-EQA benchmark. As shown in Table 1, our method outperforms all baselines with both closed and open VLMs (+7.3 and +7.4 LLM-Match over the best baseline). Fig. 4 varies the number of retrieved entries to compare LLM-Match under different numbers of input frames. Our method attains higher LLM-Match than 3D-Mem across the evaluated range, showing higher frame efficiency.

To better understand where these gains come from, Fig. 5 presents two cases, one where retrieval excludes a distractor and one where the stored views provide evidence that 3D-Mem discards. (a) The target object, a bedroom window, is not detected, so 3D-Mem does not retrieve the view showing it and answers from another window detected in a different room. Our method removes that distractor at the room level, and the view retrieval selects the correct view. (b) 3D-Mem keeps only views capturing the objects around the headboard, cropping the grey item. Our view layer keeps the wider view where the bed and its surroundings are co-visible, which our construction preserves.

A-EQA. We evaluate the full system on A-EQA, where the agent explores to answer the question. As shown in Table 2, our method achieves the best results in both QA accuracy (LLM-Match) and exploration efficiency (LLM-Match SPL). The margins over the best baseline are +5.5 and +8.8. This is achieved while querying fewer views per episode, averaging 2.34 against 3.26 for 3D-Mem. A breakdown by question category is provided in Appendix B.1.

GOAT-Bench. We also evaluate the full system on GOAT-Bench, a navigation task where later subtasks reuse the memory accumulated over earlier ones. As shown in Table 3, our method performs best in both navigation accuracy (SR) and efficiency (SPL). The margins over the best baseline are +7.2 and +2.4. This is achieved while querying fewer views per episode, averaging 2.98 against 4.66 for 3D-Mem. A breakdown by goal modality is provided in Appendix B.1.

Table 4: **Ablation on representation and retrieval.** EM-EQA 300-question subset sampled preserving the question category proportions.

Exp #	View Constr.	Room Retr.	Sim. View Retr.	LLM-Match ↑	Avg. Frames
1	✓	✗	✗	58.1	3.17
2	✗	✓	✗	54.2	2.64
3	✓	✓	✗	63.5	3.22
4	✓	✓	✓	**66.1**	2.66

Table 5: **Ablation on exploration strategy.** The same A-EQA 184-question subset as the main evaluation.

Method	LLM-Match ↑	LLM-Match SPL ↑
w/o room exploration	56.5	45.7
w/o view refinement	55.0	41.7
HiREx (Ours)	**57.5**	**48.6**

Table 6: **Robustness to weaker upstream models.** We replace the LLM for one retrieval stage from GPT-4o with an open model (Qwen3-VL-4B-Instruct), keeping all other stages unchanged, on a 46-question A-EQA subset. 3D-Mem uses GPT-4o for all stages.

Method	LLM-Match ↑	LLM-Match SPL ↑
HiREx (Ours)	**57.1**	**51.0**
w/ open-model room retrieval	57.1	50.9
w/ open-model view retrieval	53.3	50.5
3D-Mem	50.0	43.0

4.3 Analysis

Ablation study. We quantify the contributions of memory and retrieval on EM-EQA, where all methods observe the same trajectories and exploration does not affect the results. Table 4 covers three components. Without the view layer construction, views are selected by a prior strategy (Yang et al., 2025). Without similarity-based view retrieval, class-based view retrieval uses the full budget k. Removing similarity-based view retrieval lowers LLM-Match (Exps. 4 and 3), and removing room retrieval lowers it again (Exp. 1). Replacing our view construction also lowers LLM-Match (Exps. 3 and 2). The two differ in the co-visibility they keep. Ours covers 75% of co-visible class pairs while keeping only 15% of observations as views, against 50% with 11% for the prior.

We also examine the contributions of exploration options in the full system on the A-EQA 184-question subset. As shown in Table 5, removing either room exploration or view refinement lowers both QA accuracy (LLM-Match) and exploration efficiency (LLM-Match SPL).

Robustness to weaker upstream models. Our retrieval proceeds from rooms to views, and its result activates the exploration options, so errors in an earlier stage can propagate to later decisions. We evaluate this risk by replacing the LLM for each retrieval decision from GPT-4o with an open model (Qwen3-VL-4B-Instruct), letting errors arise naturally rather than injecting them. For view retrieval, this concerns the class-based path, and the similarity-based path involves no such decision. The evaluation uses a 46-question subset of A-EQA, preserving the question category proportions. As shown in Table 6, the open model on room retrieval leaves both QA accuracy (LLM-Match) and exploration efficiency (LLM-Match SPL) close to the original system. Room retrieval thus tolerates the weaker model. With the swap in view retrieval, QA accuracy decreases. Even then, both metrics remain above 3D-Mem using the same closed model for all stages.

5 Conclusion

We presented HiREx, in which a layered scene memory, task-relevant retrieval, and adaptive exploration repeat as a single loop. The memory organizes the image set under room, view, and object layers, keeping views to preserve diverse combinations of co-observed objects. Retrieval narrows the evidence through the layers, selecting views along two complementary paths, and the level at which evidence is found activates exploration options at that level. Across question answering and goal navigation benchmarks, the system outperforms all baselines, with each component contributing to the results. As future work, we plan to extend the loop to interaction with the environment, beyond the question answering and goal navigation addressed here.

AI USE STATEMENT

Research ideation and execution. LLMs were used for feedback on the methodology and for interpreting results. The authors designed the method and experiments, and made all decisions.

Writing. LLMs were used to draft and edit parts of the authors' text. We have reviewed all AI-assisted work. The authors checked each interpretation against the measured results and revised all LLM-edited text, including descriptions of prior works. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5664–5673, 2019.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025.

Peixin Chen, Guoxi Zhang, Jianwei Ma, and Qing Li. Hypothesis graph refinement: Hypothesis-driven exploration with cascade error correction for embodied navigation. In *Proceedings of the European Conference on Computer Vision*, 2026. arXiv:2604.04108.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–10, 2018.

Arash Gholami Davoodi, Seyed Pouyan Mousavi Davoudi, and Pouya Pezeshkpour. Llms are not intelligent thinkers: Introducing mathematical topic tree benchmark for comprehensive evaluation of llms. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3127–3140, 2025.

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.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. Vln bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 1643–1653, 2021.

Wenbo Hu, Yining Hong, Yanjun Wang, Leison Gao, Zibu Wei, Xingcheng Yao, Nanyun Peng, Yonatan Bitton, Idan Szpektor, and Kai-Wei Chang. 3dllm-mem: Long-term spatial-temporal memory for embodied 3d large language model. *arXiv preprint arXiv:2505.22657*, 2025.

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*, 2026. arXiv:2511.10376.

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

Kaixuan Jiang, Yang Liu, Weixing Chen, Jingzhou Luo, Ziliang Chen, Ling Pan, Guanbin Li, and Liang Lin. Beyond the destination: A novel benchmark for exploration-aware embodied question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9091–9101, 2025.

Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 19729–19739, 2023.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16373–16383, 2024.

Yu-Ji Kim, Dahye Lee, Jun-Seong Kim, Hyeon-Woo Nam, GeonU Kim, Yongjin Kwon, Yu-Chiang Frank Wang, Jaesung Choe, and Tae-Hyun Oh. Splatreasoner: Enhancing embodied reasoning and grounding by novel view synthesis. In *Proceedings of the European Conference on Computer Vision*, 2026. arXiv:2601.13132.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Qwen3-VL-Embedding and Qwen3-VL-Reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*, 2026.

Arjun Majumdar, Anurag Ajay, Xiaohan Zhang, Pranav Putta, Sriram Yenamandra, Mikael Henaff, Sneha Silwal, Paul Mcvay, Oleksandr Maksymets, Sergio Arnaud, et al. Openeqa: Embodied question answering in the era of foundation models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16488–16498, 2024.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*, 2024.

Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 815–824, 2023.

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

Allen Z Ren, Jaden Clark, Anushri Dixit, Masha Itkina, Anirudha Majumdar, and Dorsa Sadigh. Explore until confident: Efficient exploration for embodied question answering. *arXiv preprint arXiv:2403.15941*, 2024.

Antoni Rosinol, Andrew Violette, Marcus Abate, Nathan Hughes, Yun Chang, Jingnan Shi, Arjun Gupta, and Luca Carlone. Kimera: From slam to spatial perception with 3d dynamic scene graphs. *The International Journal of Robotics Research*, 40(12-14):1510–1546, 2021.

Saumya Saxena, Blake Buchanan, Chris Paxton, Peiqi Liu, Bingqing Chen, Narunas Vaskevicius, Luigi Palmieri, Jonathan Francis, and Oliver Kroemer. Grapheqa: Using 3d semantic scene graphs for real-time embodied question answering. *arXiv preprint arXiv:2412.14480*, 2024.

Dhruv Shah, Michael Robert Equi, Błażej Osiński, Fei Xia, Brian Ichter, and Sergey Levine. Navigation with large language models: Semantic guesswork as a heuristic for planning. In *Conference on Robot Learning*, pp. 2683–2699. PMLR, 2023.

Ayça Takmaz, Elisabetta Fedele, Robert W Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. *arXiv preprint arXiv:2306.13631*, 2023.

Johanna Wald, Helisa Dhamo, Nassir Navab, and Federico Tombari. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3961–3970, 2020.

Ningnan Wang, Weihuang Chen, Liming Chen, Haoxuan Ji, Zhongyu Guo, Xuchong Zhang, and Hongbin Sun. Expand your scope: Semantic cognition over potential-based exploration for embodied visual navigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18620–18628, 2026.

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

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

Shun-Cheng Wu, Johanna Wald, Keisuke Tateno, Nassir Navab, and Federico Tombari. Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7515–7525, 2021.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*, 2023.

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*, pp. 17294–17303, 2025.

Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. Sg-nav: Online 3d scene graph prompting for llm-based zero-shot object navigation. *Advances in neural information processing systems*, 37:5285–5307, 2024.

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

Bangguo Yu, Hamidreza Kasaei, and Ming Cao. Frontier semantic exploration for visual target navigation. *arXiv preprint arXiv:2304.05506*, 2023.

Junbo Zhang, Runpei Dong, and Kaisheng Ma. Clip-fo3d: Learning free open-world 3d scene representations from 2d dense clip. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2048–2059, 2023.

Xiaolin Zhou, Tingyang Xiao, Liu Liu, Yucheng Wang, Maiyue Chen, Xinrui Meng, Xinjie Wang, Wei Feng, Wei Sui, and Zhizhong Su. Fsr-vln: Fast and slow reasoning for vision-language navigation with hierarchical multi-modal scene graph. *arXiv preprint arXiv:2509.13733*, 2025.

Ziyu Zhu, Xilin Wang, Yixuan Li, Zhuofan Zhang, Xiaojian Ma, Yixin Chen, Baoxiong Jia, Wei Liang, Qian Yu, Zhidong Deng, et al. Move to understand a 3d scene: Bridging visual grounding and exploration for efficient and versatile embodied navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8120–8132, 2025.

Filippo Ziliotto, Tommaso Campari, Luciano Serafini, and Lamberto Ballan. Tango: training-free embodied ai agents for open-world tasks. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 24603–24613, 2025.

HiREx: Hierarchical Retrieval and Active Observation over Visual Memory for Embodied Exploration and Reasoning

Appendix

A Additional Details of the Method

A.1 Details about Observation-Centric Scene Graph

A.1.1 Object layer.

Each object node in the object layer maintains an object-centric crop image that best represents the corresponding object. We next describe how this representative crop is updated as new image observations are obtained. To update the best-observing image for each object, we define the observation score of object o_j for image v_i using its point cloud p_j and mask in the image m_{ij}:

$$s(v_i, o_j) = \alpha \cdot s_{\text{vis}}(v_i, o_j) + (1 - \alpha) \cdot s_{\text{area}}(v_i, o_j). \tag{2}$$

Here, s_{vis} is the rate of points in p_j that are projected in m_{ij} without occlusion. We consider that a point is occluded in an image if its projected depth is more than 0.2 m larger than the image's depth at that position. s_{area} is the rate of mask area with respect to a fixed size: 360×360. α denotes the weight between the two scores, and is set to 0.5 in all experiments. When images are newly observed, we calculate the observation score of each object in each image that observes it. Then, we update the best image and the best score for an object if one of the newly calculated scores is higher than the previous best score.

A.1.2 View layer.

The view layer consists of a subset of images selected to capture co-visibility among objects. This subset is constructed by first performing bottom-up clustering over the object-layer nodes and then assigning a representative view to each resulting cluster node in a top-down manner. We next describe in detail how the view layer is constructed.

The bottom-up clustering procedure, referred to as Child-aware Co-visibility Agglomerative Clustering (CCAC), corresponds to lines 1–5 of Alg. 1. Clusters are merged based on the co-visibility similarity (Sim) defined in Sec. 3.1, which depends on how the co-visible view set of each cluster C is defined. A naive approach is to regard an image as observing a cluster only if it observes all objects in the cluster. However, as clusters grow, it becomes unlikely that a single image observes all objects, so merging under this definition stops at small clusters. Instead, we define the cluster view set using child-aware views $V(C)$ (Eq. 1), which allows clusters to grow even when no single image observes all objects.

The subsequent loop (lines 6–12), referred to as Top-down View Assigning (TVA), traverses the newly merged clusters $\mathcal{C}_{\text{new}}$ in inverse merge order and assigns to each cluster C the view $v_C \in V(C)$ least similar to the views already in $\mathcal{V}$. To compute CLIP similarity $s(v_i)$, we define a cluster-aware feature $f(v_i)$ for each image by aggregating the features of objects in C observed in the image. Focusing on objects in C makes the similarity reflect cluster-specific content.

$$f(v_i) = \frac{\sum_j \mathcal{F}_{ij} \mathcal{M}_{ij} \delta_{j \in C}}{\sum_j \mathcal{M}_{ij} \delta_{j \in C}}, \tag{3}$$

where $\delta_{j \in C}$ is an indicator (1 if $o_j \in C$, 0 otherwise), and $\mathcal{F}_{ij}$ is the CLIP visual feature of the object–image pair (o_j, v_i) obtained during the object set update (Gu et al., 2024). We set the threshold η that determines whether to select v_C for C as 0.85. If $s(v_C) \geq \eta$, no view is added, and C is pruned with its children reattached to its parent (line 12).

Finally, Alg. 2 describes how the view layer is efficiently updated in an incremental manner as new images arrive. CCAC and TVA in Alg. 2 denote lines 1–5 and lines 6–12 of Alg. 1, respectively.

Algorithm 2 Incremental Update of View Layer

Input: Co-visibility matrix $\mathcal{M}$, cluster set $\mathcal{C}$, child map Φ, object-wise CLIP visual features $\mathcal{F}$, view layer $\mathcal{V}$, similarity threshold η, height threshold H
Output: Updated cluster set $\mathcal{C}$, child map Φ, and view layer $\mathcal{V}$

Identify newly observed objects $\mathcal{O}_{new}$ and re-observed objects $\mathcal{O}_{re}$ from $\mathcal{M}$
$\mathcal{C} \leftarrow \mathcal{C} \cup \{\{o_j\} \text{ for } o_j \in \mathcal{O}_{new}\}$
$\mathcal{C}_{det}, \mathcal{C}_{fix} \leftarrow \varnothing$
for $o_j \in \mathcal{O}_{re}$ **do**
    $C \leftarrow \{o_j\}$
    while $h(C) < H$ **do**
        $p \leftarrow$ parent of C
        if $p = \varnothing$ **then**
            break
        delete C from Φ
        $\mathcal{C}_{det} \leftarrow \mathcal{C}_{det} \cup \Phi(p)$
        $C \leftarrow p$
    $\mathcal{C}_{fix} \leftarrow \mathcal{C}_{fix} \cup \{C\}$
$\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}_{det}$
$\mathcal{C}_{new}, \mathcal{C}, \Phi = \mathrm{CCAC}(\mathcal{C}, \Phi, \mathcal{M})$ with ConstrainMerge($\mathcal{C}_{det}$; $\mathcal{C}_{fix}$) ▷ Alg. 1, lines 1–5
$\Phi, \mathcal{V} = \mathrm{TVA}(\mathcal{C}_{new}, \Phi, \mathcal{M}, \mathcal{F}, \mathcal{V}, \eta)$ ▷ Alg. 1, lines 6–12
return $\mathcal{C}, \Phi, \mathcal{V}$



Figure 6: Example of a view image used to query the VLM for view–room assignment. Detected objects are annotated using Set-of-Mark with numbered markers: “vase”, “bed”, and “tv” correspond to marks 0, 1, and 2, respectively.

During an incremental update, the hierarchy is edited for re-observed objects $\mathcal{O}_{re}$ while newly observed objects $\mathcal{O}_{new}$ are inserted as new leaves. For each $o_j \in \mathcal{O}_{re}$, we traverse its ancestor chain until reaching the first cluster C whose height satisfies $h(C) \geq H$ ($H = 3$). Clusters below this level are removed from the hierarchy, and their remaining siblings are collected into the detached cluster set $\mathcal{C}_{det}$. The surviving ancestor C is added to the fixed cluster set $\mathcal{C}_{fix}$, and re-clustering is performed with $\mathrm{ConstrainMerge}(\mathcal{C}_{det}; \mathcal{C}_{fix})$, which ensures that detached clusters $C \in \mathcal{C}_{det}$, even when merged with newly observed objects, remain within the subtree of their corresponding fixed ancestor.

A.1.3 ROOM LAYER.

We next describe how newly added views are processed with the VLM to infer room labels for the detected objects. Before querying each newly added view with the VLM, the detected objects are marked using Set-of-Mark (Yang et al., 2023), as illustrated in Fig. 6. The marked image v_j is then provided in the prompt shown in Fig. 10. We enforce a structured output format (e.g., "Rooms: ...\n0:A\n1:B\n...(n-1):A") so that the model explicitly assigns a room label to each object. The token probability at the position where the room alphabet r_i is generated for object i is used as a confidence score for assigning room label c to object i in view v_j. This score is then used as an unnormalized likelihood in the room–instance assignment described in Sec. 3.1. The rooms listed for a view link it to those room types directly, and the room assignments of its objects link it to the rooms of the objects it contains.

A.2 DETAILS ABOUT INFORMATION RETRIEVAL

The reasoning process for task-relevant information retrieval is illustrated in Fig. 3. We first use the LLM to identify up to k task-relevant rooms from the room layer of the scene graph. When no room is selected, the candidates for view retrieval are all stored views.

Then, view retrieval proceeds along two paths. For class-based view retrieval, the LLM first selects the object classes relevant to the task among those observed in the candidate views and ranks them by relevance. Views are then greedily selected to cover the ranked classes. At each greedy step, we select the view observing the most uncovered objects of the highest-ranked class, breaking ties by the number of other selected classes observed in the view. Objects observed in the selected view are then marked as covered. This process continues until either $k(= 5)$ views have been selected or all objects from the selected classes are covered.

For similarity-based view retrieval, one view is added from the room-restricted candidates not selected by the class-based path, replacing the last class-retrieved view. The candidate whose embedding is most similar to the task embedding is added if the cosine similarity exceeds a threshold of 0.23 and exceeds the most similar class-retrieved view by a margin of 0.01, fixed across all benchmarks. Otherwise, the k class-retrieved views are kept. View embeddings are computed once when a view enters the memory and cached, and the task embedding is computed once per query.

A.3 DETAILS ABOUT REASONING FOR EXPLORATION

In this section, we describe how target positions are determined for exploration options activated by retrieval. For frontiers, we follow (Yang et al., 2025).

A.3.1 ROOM EXPLORATION.

We update a top-down room segmentation map at every step. After establishing connections between the newly added view and room types, we input the view and connected room types to SAM3 to obtain a mask for each room. Using the depth map, camera pose, and camera intrinsics, the masked regions are projected onto a top-down map with the same resolution as the occupancy map. We then update each pixel's room-type probability using a Bayesian update similar to the room–instance assignment described in Sec. 3.1.

When the room exploration option is activated, we cluster unexplored regions within the retrieved room under a radius of $r(= 1.5m)$. For each cluster, the navigation target is selected from explored points located $0.75m$ from the cluster center. Among these candidates, we choose the point whose direction from the cluster center is closest to the principal axis with the smallest eigenvalue obtained by applying PCA to the cluster region. This favors approaching the unexplored region from a direction approximately normal to its elongated boundary, aiming at a broader first view into the region.

A.3.2 VIEW REFINEMENT.

We describe how each refinement strategy determines the target position. The task-relevant region is obtained by projecting the intersection between the SAM3 prediction mask and detected-object

Table 7: Performance on A-EQA by Question Categories.

Method	object recognition		object localization		attribute recognition		spatial understanding		object state recognition		functional reasoning		world knowledge		overall	
Metric	Match	SPL	Match	SPL	Match	SPL	Match	SPL	Match	SPL	Match	SPL	Match	SPL	Match	SPL
CG Cap.	25.3	-	16.5	-	29.2	-	37.0	-	52.2	-	46.8	-	37.8	-	34.4	6.5
SVM Cap.	29.0	-	17.2	-	31.5	-	31.5	-	54.2	-	39.8	-	38.9	-	34.2	6.4
Frame Cap.	25.0	-	24.0	-	34.1	-	34.4	-	56.9	-	53.5	-	40.6	-	38.1	7.0
Multi-Frame	34.0	-	34.3	-	51.5	-	39.5	-	51.9	-	45.6	-	36.6	-	41.8	7.5
Explore-EQA (RSS'24)	44.0	19.6	37.1	29.6	55.3	36.0	**42.1**	6.6	46.3	9.2	**63.2**	35.7	45.5	22.0	46.9	23.4
3D-Mem† (CVPR'25)	39.0	35.9	52.1	42.3	55.3	41.3	35.5	18.9	58.3	43.3	52.9	39.4	48.2	30.7	49.6	37.0
HGR† (ECCV'26)	38.0	36.4	**55.0**	**46.2**	59.9	44.8	35.5	27.2	62.0	48.1	57.4	33.8	50.0	33.0	52.0	39.8
HiREx (Ours)	**54.0**	**47.1**	**55.0**	43.0	**62.9**	**49.8**	40.8	**33.6**	**66.7**	**59.9**	58.8	**41.6**	**58.9**	**58.8**	**57.5**	**48.6**

Table 8: Performance on GOAT-Bench by Question Modalities.

Method	Object Category		Language		Image		Overall	
Metric	Success Rate	SPL	Success Rate	SPL	Success Rate	SPL	Success Rate	SPL
Explore-EQA (RSS'24)	64.7	48.4	42.9	22.7	56.8	41.8	55.0	37.9
3D-Mem† (CVPR'25)	71.7	45.6	62.6	**45.2**	73.9	62.2	69.4	50.7
HGR† (ECCV'26)	73.7	50.4	57.1	39.9	71.6	60.3	67.6	50.1
HiREx (Ours)	**83.8**	**55.0**	**68.1**	41.8	**77.3**	**62.7**	**76.6**	**53.1**

masks onto the occupancy map, and the focus point is computed as the average position of this region. For *closer view*, the target position is placed in the same direction from the focus point at $0.6\times$ the observation distance. If occupied, it is moved toward the focus point until an unoccupied location is found. For *wider context*, the target position is placed in the same direction at $1.3\times$ the observation distance, and moved farther from the focus point if occupied. For *different angle*, we keep the distance from the focus point unchanged to avoid introducing an additional scale change, and instead change the direction to the principal axis with the smallest eigenvalue obtained from PCA on the task-relevant region.

B Additional Experimental Results and Analysis

B.1 Detailed Experiment Results

In this section, we provide a detailed breakdown of results by question categories and modalities.

Table 7 presents the breakdown of A-EQA across the seven OpenEQA question categories, evaluated on the 184-question subset described in the manuscript. For each category, the first and second columns denote LLM-Match and LLM-Match SPL, respectively. Our method outperforms existing baselines in most categories.

Table 8 shows the breakdown on GOAT-Bench across the three query modalities, evaluated on the $1/10$ subset of the Val Unseen split. Our method outperforms the baselines in seven of the eight modality-metric combinations, including both overall metrics.

B.2 Comparison of View-Retrieval Strategies

Fig. 7 compares allocations of the view budget k between the two retrieval paths. Allocating the budget across both paths outperforms spending it on either path alone, at every allocation tested. We adopt class $(k-1)$ + similarity (1), which attains higher LLM-Match at comparable frames. Although the models differ, the gain of the adopted allocation over class-based retrieval alone is consistent with the GPT-4o ablation in Table 4.

B.3 Runtime Analysis

Table 9 reports the per-step runtime of each component on the A-EQA 184-question subset. The total is 8.07 seconds per step, against 6.93 reported for 3D-Mem (Yang et al., 2025), a 16% overhead.

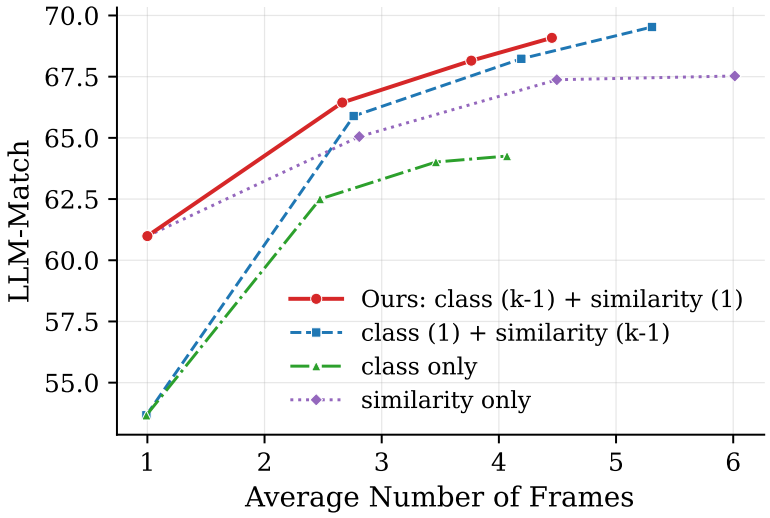


Figure 7: **Comparison of view-retrieval strategies** on the 1,636 EM-EQA questions. Retrieval decisions and answering use GPT-5.6-luna for cost, and answers are scored with GPT-4 following the benchmark protocol (Majumdar et al., 2024). Absolute scores are thus not comparable to the main results, and the comparison concerns the relative ordering of strategies.

Table 9: **Per-step runtime of each component** on the A-EQA 184-question subset, in seconds. Retrieval is split into its three stages, where Q–V denotes the query–view similarity over cached view embeddings. The total of 8.07 seconds is a 16% overhead over 3D-Mem, below the 24% that HGR reports, while outperforming HGR by +5.5 LLM-Match and +8.8 LLM-Match SPL (Table 2).

2D–3D Lift	View Layer Update	Retrieval			Termination	Exploration
		Room	Class	Sim. (Q–V)		
2.50	0.04	0.84	0.71	0.38	1.39	2.21

For comparison, HGR (Chen et al., 2026) reports approximately 24% overhead over 3D-Mem, while our method attains +5.5 LLM-Match and +8.8 LLM-Match SPL over HGR on the same subset (Table 2). The 2D–3D lifting runs concurrently with navigation, as in 3D-Mem. The view embeddings dominate the cost of the similarity stage, and they are computed during navigation when views enter the memory.

B.4 Visualization of Exploration

Fig. 8 illustrates an example sequence of the VLM's exploration decisions when answering an open-ended question in an unknown environment: "If I sit in the armchair in the living room, what animal can I touch with my right hand?" The figure shows the top-down map and the selected exploration option at each step. The agent starts from a bedroom and initially selects frontier exploration to move toward unexplored areas. After reaching the living room, room exploration is activated, and the agent navigates toward unexplored regions of the room. Once candidate regions are explored, the agent switches to view refinement to verify whether the observations contain the information required to answer the question. At step 6, the agent refines the view by moving closer to an armchair to inspect nearby objects, but determines that it is not the armchair referenced in the question. The agent then refines another candidate view retrieved from memory (step 7). This view shows an animal statue near the armchair, and the agent moves closer to identify which animal it represents. After the closer observation, the VLM produces the correct answer, "an elephant." This example shows how the selected option changes with the level at which evidence is retrieved. A real-robot demonstration of the system is provided in the supplementary video.

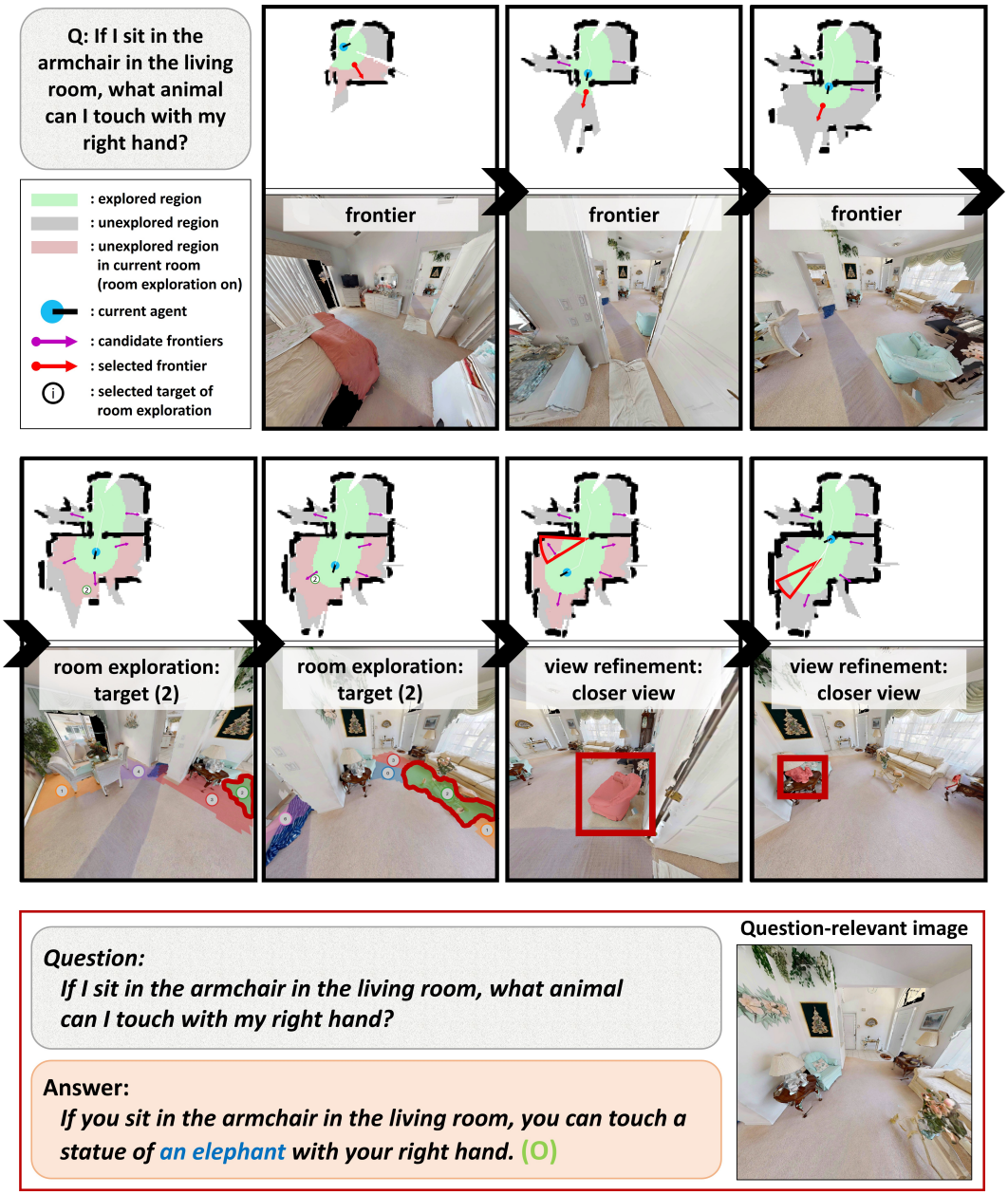


Figure 8: Example exploration sequence for answering an open-ended question in A-EQA. The agent progressively transitions from frontier exploration to room exploration and view refinement to gather task-relevant information, ultimately leading to the correct answer ("an elephant").

B.5 Failure Case Analysis

We compare failure cases of our system with 3D-Mem, a VLM-guided embodied exploration method for A-EQA that follows a comparable high-level workflow (exploration → task-agnostic memory → task-relevant memory → final answer). The analysis is conducted on the 184-episode subset of A-EQA used in the main comparison. Failure cases are defined as episodes where the final answer score is below 3 or the agent fails to produce an answer. The authors manually categorize all failures into five mutually exclusive types: *out-of-scope*, *exploration failure*, *memory failure*, *retrieval failure*, and *alternative answer*.

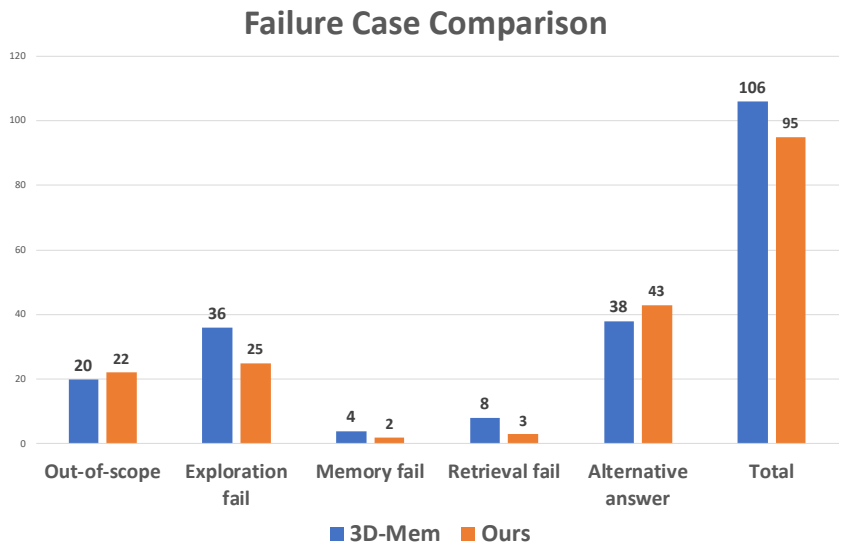


Figure 9: Distribution of failure cases of our method and 3D-Mem on A-EQA. Failures are grouped into five categories: out-of-scope cases, exploration failures, memory failures, retrieval failures, and alternative answers.

An *out-of-scope* case refers to questions that require navigating across multiple floors. Although these cases are counted as failures under the benchmark metric, we separate them in the analysis because both systems assume a single-floor navigation setup. An *exploration failure* occurs when the agent never observes the relevant target on the same floor before reaching the exploration limit. A *memory failure* occurs when the target is observed, but the corresponding image is not retained in the scene representation. A *retrieval failure* occurs when the relevant observation exists in the task-agnostic memory but is excluded during task-relevant memory construction. Finally, an *alternative answer* case occurs when the agent produces a plausible answer grounded in the environment, but the benchmark annotation does not include it as a valid reference.

Fig. 9 presents an analysis of the failure cases of our method and 3D-Mem. At first glance, the proposed method appears to exhibit more failures than 3D-Mem in the out-of-scope and alternative-answer categories. However, these two categories are less indicative of the system's actual capability: out-of-scope failures primarily arise from limitations of the agent's action space, while alternative-answer cases often correspond to semantically valid answers that are counted as failures due to limitations of the evaluation protocol. Excluding these categories, 3D-Mem exhibits more failures than our method in the remaining categories of exploration, memory, and retrieval. Since each of these categories corresponds to one stage of the workflow, the lower counts are consistent with the component contributions in Tables 4 and 5.

C Complete Prompts for VLMs

We provide the full prompts used for all LLM/VLM queries in our system: room–view and room–instance assignment (Fig. 10), room retrieval (Fig. 11), class-based view retrieval (Fig. 12), exploration (Fig. 13), and termination with the final decision for EQA (Fig. 14) and GOAT-Bench (Fig. 15). All structured LLM/VLM queries use fixed prompt templates. If the output format is invalid, we perform a retry with the same prompt.

System Prompt:
You are an AI agent analyzing indoor scenes.

Content Prompt:
Identify the room type(s) observed in this image and assign each numbered instance (from 0 to {n_instances-1}) to the appropriate room, one line per instance.

Possible room types: A. bathroom, B. bedroom, C. dining room, D. garage, E. kitchen, F. laundry room, G. living room, H. office, I. rec room, J. study, K. hallway, L. entryway, M. laboratory, N. workout room, O. warehouse, P. lounge, Q. balcony, R. staircase, S. cloakroom, T. unknown

These are the rules for the task.

1. Group instances that appear to be in the same room.
2. Decide room type for each group.
3. Assign the room type to each instance in the group, and output only the final format.

Consider not only the objects in the same group but also their nearby context when deciding the room type.
Use "unknown" only if the image does not provide enough evidence to reasonably support any of the possible room types.

[img]

Numbered instances in the image:

0: {object_0}
1: {object_1}
…

Note: Assign ALL marked instance indices to exactly one room using its alphabet.
Some rooms may appear in the image but contain no marked instances.
Do not include any indices not marked in the image.

Output format:

```
Rooms: A, B, ...
0: A
1: B
…
{n-1}: A
```

Figure 10: Prompt used to query the VLM for the room assignment of objects and a view. {object_i} denotes the label of the i-th marked object, {n_instances} and n denote the number of marked objects, and [img] represents the input image.

System Prompt:
You are an AI agent in a 3D indoor scene.

Content Prompt:
Your goal is to answer questions about the scene through exploration.
To efficiently solve the problem, you should first rank rooms in the scene based on their importance.

Note: Current room list shows only the rooms and objects observed so far, so keep in mind the possibility of finding helpful objects later.
Note: When ranking, consider both the room type and its contained objects together.

These are the rules for the task.

1. Read through the whole room list with their contained objects.
2. Rank rooms in the list based on how well they can help your exploration given the question.
3. Reprint the name of all rooms that may help your exploration given the question.
4. Do not print any room not included in the list or include any additional information in your response.

Here is an example of selecting helpful rooms:

Question: What can I use to watch my favorite shows and movies?
Following is a list of rooms with their contained objects, each room one line:
bedroom (contains: 1 bed, 1 curtains, 2 book rack)
living room (contains: 1 sofa, 2 speakers, 1 tv)
kitchen (contains: 3 cabinet, 1 lamps, 1 oven)
hallway (contains: 2 boxes, 1 painting)
Answer:
living room
bedroom

Following is the concrete content of the task and you should retrieve helpful rooms in order:
Question: {question}
Following is a list of rooms with their contained objects, each room one line:
{room type 0} (contains: {n_0} {class_0}, {n_1} {class_1}, ...)
{room type 1} (contains: {n_0} {class_0}, {n_1} {class_1}, ...) ...
Answer:

Figure 11: Prompt used for room-level retrieval. {question} denotes the question, and {room type i}, {n_j}, and {$class$_j} denote the room type, object count, and object class in each room entry. The first note is included for A-EQA and GOAT-Bench.

System Prompt:
You are an AI agent in a 3D indoor scene.

Content Prompt:
Your goal is to answer questions about the scene through exploration.
To efficiently solve the problem, you should first rank objects in the scene based on their importance.

These are the rules for the task.

1. Read through the whole object list.
2. Rank objects in the list based on how well they can help your exploration given the question.
3. Reprint the name of all objects that may help your exploration given the question.
4. Do not print any object not included in the list or include any additional information in your response.

Here is an example of selecting helpful objects:

> Question: What can I use to watch my favorite shows and movies?
> Following is a list of objects that you can choose:
> painting / speaker / box / cabinet / lamp / tv / book rack / sofa / oven / bed / curtain
> Answer:
> tv / speaker / sofa / bed

Following is the concrete content of the task and you should retrieve helpful objects in order:
Question: {question}
Following is a list of objects that you can choose:
{class_0} / {class_1} / …
Answer:

Figure 12: Prompt used for object-class retrieval in the view layer. {question} denotes the question and {class_i} denotes candidate object classes.

System Prompt:
Task: You are an agent in an indoor scene tasked with exploring the environment to gather information for answering questions.

Definitions:

View: A focused observation of several objects that have been collected during exploration. Selecting a View is appropriate when it contains relevant content that seems to be helpful for answering the question if the view is refined. This allows you to obtain missing information needed to better interpret or verify the relevant content. When selecting a View, you should select the refinement strategy that is most likely to increase the amount of relevant information from the following options: (1) Closer view - to closely observe relevant content that appears too small in the current view. (2) Different angle - to observe relevant content from a new viewpoint when it is not clearly visible from the current perspective. (3) Wider context - to step back and observe more of the surroundings to help interpret the relevant content.

Room exploration: Target positions within the current room where you can gather additional information beyond the provided Views. They are marked with numbers in Views and additional room views, and are also marked with the same numbers in the room image. Selecting a Room exploration means you will move to that position and observe the associated area to gather more information in the current room.

Frontier: An observation of an unexplored region that could potentially lead to new information for answering the question. Selecting a frontier means you will explore in that direction. Note: If a frontier was previously selected (marked as 'Previously selected'), you should prefer continuing with it unless another frontier is significantly more helpful. When selecting a different frontier than the previously selected one, you must explicitly explain why it is significantly more helpful than continuing with the previous frontier.

Content Prompt:
Question: {question}

Select the exploration option that would help find the answer to the question.

The followings are all the Views that have been collected:
View 0 [img]
…
View n (current egocentric view in forward direction) [img]

The following describes room exploration targets, which indicate unexplored spatial locations within the current room. Each target is associated with a region, visualized in the same color.

The following is the room exploration map:
Room map: [img]

It is a top-down 2D occupancy map which provides spatial context to help selecting exploration targets. Explored free space is shown in green, unexplored free space is shown in gray, the agent's current position is marked in blue circle, and room exploration target locations are marked with numbers. Red regions are room areas that require further exploration, but are not yet assigned to any room exploration target.

The following are additional images showing room exploration targets unobserved in provided Views:
Additional image 0: [img]
Additional image 1: [img]
…

Room exploration target positions:
0 (observed in View i, View j, … and Additional image i, Additional image j, …)
1 (observed in View k, View l … and Additional image k, Additional image l, …)
2 (unobserved)
…

The followings are all the Frontiers:
Frontier 0 (Previously selected) [img]
Frontier 1 [img]
…

Please provide your answer in the following format: 'View i\n[Relevant content] - [Refinement strategy]', 'Room exploration i\n[Reason]', or 'Frontier i\n[Reason]', where i is the index you choose. Note that 'Relevant content' is a noun phrase describing the relevant content. Do not add any prefix or header in front of this in your response.

For example:
If you choose the first view, you can return 'View 0\nA refrigerator-like object that appears small in the current view - Closer view'. If you choose the second view, you can return 'View 1\nA refrigerator-like object observed from the side instead of the front - Different angle'. If you choose the third view, you can return 'View 2\nA refrigerator filling most of the frame with limited surrounding context - Wider context'.
If you choose the second room exploration target, you can return 'Room exploration 1\nI need to explore near the table area to see the objects on it more clearly.'.
If the first frontier was previously selected and you choose it again, you can return 'Frontier 0\nContinue toward the door that may lead to the living room, as other frontiers show only hallways or entry to irrelevant rooms without clear advantage.'. If the first frontier was previously selected but you choose the third one, you can return 'Frontier 2\nThis frontier is more helpful than the previously selected one because it reveals dining chairs near tables, whereas the previous frontier only leads to a hallway without relevant surfaces.'.

Figure 13: Prompt used for exploration. {question} denotes the question, View n is the current egocentric view, and additional images show room exploration targets not observed in the views. [img] represents the corresponding images.

System Prompt:

Task: You are an agent in an indoor scene tasked with exploring the environment to answer questions. You have collected multiple observations during exploration and must now decide whether to terminate exploration and answer the question, or continue exploring.

Termination Criteria:
If you find enough information to give an answer based on the available observations, you should terminate exploration and specify the view index that is most relevant for answering the question.

Content Prompt:

Question: {question}
The following are all the observations you have collected so far:
View 0 [img]
View 1 [img]
…
View n (current egocentric view in forward direction) [img]

Based on the collected observations, decide whether to terminate exploration.
Provide your decision in the following format: 'yes\nView i' or 'no\n[Missing Info]'.
For example:
If you decide to terminate and the first view is the most relevant, you can return 'yes\nView 0'
If you decide not to terminate, you can return 'no\nI cannot see the objects on the table clearly enough to determine the answer.'

(a) Termination

System Prompt:

Task: You are an agent in an indoor scene tasked with answering questions by observing the surroundings. Based on the available observations, you must provide a direct answer to the question.

Definitions:
View: A focused observation of several objects.

Content Prompt:

Question: {question}

The following is the egocentric view of the agent in forward direction: [img]

The followings are all the views available:
View 0 [img]
View 1 [img]
…

You should give a direct answer that can be understood by others. Don't mention words like 'view', 'on the left of the image', etc. Also, do not mention words like 'mark', 'A', 'B', ... (etc) to indicate an object.

(b) Final answer

Figure 14: Prompts used for (a) termination checking and (b) final answer generation in EQA. (b) is queried when (a) returns yes. {question} denotes the question; View n in (a) is the current egocentric view, and [img] represents the corresponding input images.

System Prompt:
Task: You are an agent in an indoor scene tasked with indoor navigation to find the target object required in the question. You have collected multiple observations during exploration and must now decide whether to terminate exploration and navigate to the target object, or continue exploring.

Definitions:
Object: An image crop of a single detected object instance, showing the best observed appearance of that object.
View: A wider observation image providing spatial and contextual information. Objects visible in each view are marked with their index.

Terminate exploration only if an object from the provided Object instances can be confidently identified as the target by verifying that the attributes and surrounding context of the target object described in the question are observable for that object in the current observations. Otherwise, if none of the provided Object instances can be confidently verified as the target based on the current observations, continue exploring.

Content Prompt:
Question: {question}

The following is your current egocentric view in forward direction: [img]

The following are image crops of candidate object instances:
Please note that the class name may not be accurate due to the limitation of the object detection model. So you still need to utilize the images to make the decision.
Object 0: {class_0} (observed in {view_i}, {view_j}, ...): [img]
Object 1: {class_1} (observed in {view_k}, {view_l}, ...): [img]
...

The following are all the Views you have collected (candidate objects are marked with their Object number):
View 0 [img]
View 1 [img]
...

Based on the collected observations, decide whether to terminate exploration.
If you decide to terminate, return 'yes' only. If you decide not to terminate, return 'no' only. Do not write anything before the decision line.

(a) Termination

System Prompt:
Task: You are an agent in an indoor scene tasked with indoor navigation to find the target object required in the question. You are given image crops of candidate object instances and supporting view images with the candidate objects marked.

Definitions:
[Object: as in (a)]
View: A wider observation image providing spatial and contextual information. Candidate objects visible in each view are marked with their index.

Content Prompt:
[Question, egocentric view, and object crops as in (a)]

The following are all collected Views (candidate objects are marked with their Object number):
View 0 [img]
View 1 [img]
...

Select the Object that best matches the target in the question. You can use both the Object crops and the Views (where objects are marked with their indices) to support your decision.
Provide your answer in the format: 'Object i' where i is the index of the object you choose. You can explain the reason for your choice, but put it in a new line after the choice.

(b) Final decision

Figure 15: Prompts used for (a) termination checking and (b) final decision in GOAT-Bench. (b) is queried when (a) returns yes, and bracketed lines in (b) abbreviate content identical to (a). {question} denotes the goal, and {class_i} denotes the class of each candidate object, whose crop comes from the object layer for the objects observed in the retrieved views. {view_i} denotes the views where the object is observed, and [img] represents the corresponding input images.