GSMEM: 3D GAUSSIAN SPLATTING AS PERSISTENT SPATIAL MEMORY FOR ZERO-SHOT EMBODIED EXPLORATION AND REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Effective embodied exploration requires agents to accumulate and retain spatial knowledge over time. However, existing scene representations, such as discrete scene graphs or static view-based snapshots, lack *post-hoc re-observability*. If an initial observation misses a target, the resulting memory omission is often irrecoverable. To bridge this gap, we propose **GSMem**, a zero-shot embodied exploration and reasoning framework built upon 3D Gaussian Splatting (3DGS). By explicitly parameterizing continuous geometry and dense appearance, 3DGS serves as a persistent spatial memory that endows the agent with *Spatial Recollection*: the ability to render photorealistic novel views from optimal, previously unoccupied viewpoints. To operationalize this, GSMem employs a retrieval mechanism that simultaneously leverages parallel object-level scene graphs and semantic-level language fields. This complementary design robustly localizes target regions, enabling the agent to "hallucinate" optimal views for high-fidelity Vision-Language Model (VLM) reasoning. Furthermore, we introduce a hybrid exploration strategy that combines VLM-driven semantic scoring with a 3DGS-based coverage objective, balancing task-aware exploration with geometric coverage. Extensive experiments on embodied question answering and lifelong navigation demonstrate the robustness and effectiveness of our framework.

1 INTRODUCTION

Embodied navigation has recently emerged as a central problem in embodied AI, driving growing interest in agents that interact with complex 3D environments. Effective embodied exploration in such environments depends on an agent's ability to accumulate and retain spatial knowledge over time.

To support such capabilities, existing systems adopt different forms of scene representation. One line of work focuses on *object-centric abstractions*, such as 3D scene graphs (Gu et al., 2024; Yin et al., 2024), which model environments as a discrete set of nodes and edges representing objects and their relationships. Another line employs *view-based representation*s, such as 2D top-down maps (Yokoyama et al., 2024; Chang et al., 2023) or egocentric image snapshots (Yang et al., 2025b), to encode spatial observations for planning and querying.

Despite their effectiveness, both paradigms lack comprehensive coverage of previously explored regions required for robust reasoning and navigation. *Object-centric methods* rely highly on the accuracy of real-time perception modules. Since they discard raw visual data for discrete labels, a single detection failure results in an irrecoverable memory omission. Such errors propagate directly to downstream tasks, causing retrieval failures, particularly for open-vocabulary objects. Conversely, *view-based representations* remain inherently sparse and view-dependent. If a target is captured from a suboptimal angle or suffers from occlusion during the initial pass, the static memory lacks the geometric fidelity for VLMs to resolve ambiguity or for the agent to plan precise re-visitation.

We identify a fundamental gap underlying these limitations: the lack of *post-hoc re-observability*. Unlike humans, who can mentally re-visit a past scene from a new perspective to discover missed details, current agents are "locked" into the specific observations they made during their initial exploration process.

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: With GS-Mem, previously explored regions can be retrieved and re-observed directly from the 3DGS memory without physically navigating to them.

To bridge this gap, we propose GSMem, a zero-shot embodied exploration and reasoning framework built upon 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023). Specifically, 3DGS explicitly parameterizes both scene geometry and dense appearance, inherently supporting real-time, high-fidelity novel view synthesis. Leveraging this ability, we move beyond discrete abstractions and static snapshots toward a dense, continuous radiance field that serves as a persistent spatial memory. As shown in fig. 1, this allows the agent to perform spatial recollection: it can "re-visit" previously explored regions by rendering photorealistic views from arbitrary, optimal viewpoints, even those it never physically occupied.

To ensure robust localization, our framework introduces a multi-level retrieval-rendering mechanism. When tasked with a query, GSMem localizes the region of interest with two parallel representations: object-level cues from a 3D scene graph and semantic-level features from a language field. The two levels are complementary: even when the scene graph misses a target, the semantic-level features can still localize the candidate region. The agent then "hallucinate" this region from an optimal viewpoint through 3DGS rendering for VLM reasoning. Furthermore, we address the challenge of task-aware exploration by designing a hybrid strategy that combines VLM-driven semantic scoring with a 3DGS-based coverage objective. By quantifying the representational uncertainty of the Gaussian field via entropy, our agent can identify under-observed regions that offer the highest information gain, ensuring the constructed memory is both comprehensive and semantically rich.

In summary, our contributions are as follows:

- We propose **GSMem**, a zero-shot embodied exploration and reasoning framework built upon persistent 3D Gaussian Splatting (3DGS) memory, equipping agents with the capability of spatial recollection.
- We introduce a **multi-level retrieval-rendering mechanism** that integrates object-level scene graphs and semantic-level language fields to localize relevant regions, followed by optimal viewpoint selection for post-hoc re-observation to support VLM reasoning.
- We design a **hybrid exploration strategy** that combines VLM-driven semantic relevance with information gain-based coverage score to form a balanced and task-aware zero-shot exploration strategy.
- Extensive experiments on **embodied question answering** and **lifelong navigation** demonstrate the effectiveness and generalization capability of our framework.

2 Related Work

Scene Representations for Embodied Agents. Effective spatial memory is fundamental for embodied agents to reason about their environment. Early approaches largely relied on 2D top-down maps or occupancy grids, which support navigation but often discard the fine-grained visual details necessary for complex semantic reasoning. To address this, object-centric representations, such as ConceptGraphs (Gu et al., 2024) and Hydra (Hughes et al., 2024), were introduced to model environments as structured graphs of objects and their relationships. While these methods provide high-level semantic abstractions, they inherently discretize the world, potentially losing geometric precision and context outside of detected objects. Alternatively, view-based representations, such as 3D-Mem (Yang et al., 2025b) and GOAT (Chang et al., 2023), maintain memory as a collection of egocentric image snapshots or keyframes. While these preserve visual fidelity, they are view-dependent and sparse, limiting the agent's ability to extrapolate novel viewpoints or reason about 3D geometry continuously. In contrast, our GSMem utilizes 3DGS (Kerbl et al., 2023) to maintain a dense, continuous, and re-renderable memory. Unlike recent offline semantic 3DGS approaches (Qin et al., 2024; Zhou et al., 2024), we employ an optimization-free language field that updates in real-time, enabling immediate zero-shot retrieval without the artifacts of discretization or the latency of iterative training.

Embodied Exploration and Navigation. Embodied navigation has evolved from reinforcement learning (RL) baselines to zero-shot VLM agents. Traditional methods often rely on trained policies or modular skill chains, such as Modular GOAT (Chang et al., 2024) and CLIP on Wheels (CoW) (Gadre et al., 2023). The GOAT-Bench benchmark (Khanna et al., 2024) further established baselines utilizing monolithic and skill-chain policies. However, these methods often struggle to generalize to open-vocabulary queries in unseen environments without extensive training. The rise of VLMs has shifted the focus toward zero-shot exploration. Methods like Explore-EQA (Ren et al.), VLMNav (Goetting et al., 2025), and DyNaVLM (Ji et al., 2025) utilize VLM reasoning to select frontiers or generate subgoals directly from observations. More recent approaches, such as TANGO (Ziliotto et al., 2025) and MTU3D (Zhu et al., 2025), leverage powerful commercial models (e.g., GPT-4o (Hurst et al., 2024)) to enhance reasoning and planning capabilities. Despite their advancements, these methods often rely on discrete observations or sparse graphs. Our framework distinguishes itself by combining VLM-driven semantic scores with a rigorous information-theoretic coverage objective, ensuring robust zero-shot performance that outperforms both RL-based and VLM-based baselines.

3 Methodology

3.1 3DGS Mapping

In this section, we go through the 3DGS mapping and online language field generation processes during active exploration. Preliminaries of 3DGS can be found in section A.1.

Keyframe Selection & Optimization. Our mapping system takes RGB-D images from sensors as input and maintains a keyframe set $\mathcal{K}$ to incrementally update the 3DGS map. At each time step, the agent captures three surrounding RGB-D views. For each incoming frame, we compute the average flow magnitude between the current frame and the last keyframe using a pretrained optical flow network (Teed & Deng, 2020). If the average flow exceeds a predefined threshold, the frame is selected as a new keyframe and added to $\mathcal{K}$.

As exploration progresses, the number of keyframes grows. To ensure efficient and stable map updates, we adopt a sliding window $\mathcal{W} \subset \mathcal{K}$ with a fixed size of 10. During each optimization step, we update the Gaussian map using keyframes within the current window $\mathcal{W}$, together with two additional frames randomly sampled from $\mathcal{K}$. Let $\mathcal{S}$ denote this sampled subset, and define the optimization set as: $\mathcal{T} = \mathcal{W} \cup \mathcal{S}$. The optimization targets are defined as

$$\mathcal{L}_{\text{rgb}} = \sum_{k \in \mathcal{T}} \left\| \hat{\mathbf{I}}_k - \mathbf{I}_k \right\|_1, \quad \mathcal{L}_{\text{depth}} = \sum_{k \in \mathcal{T}} \left\| \hat{\mathbf{D}}_k - \mathbf{D}_k \right\|_1, \tag{1}$$

predicted RGB-D images $\hat{\mathbf{I}}_k$ and $\hat{\mathbf{D}}_k$ are rendered via eq. (12) and supervised against the observed images $\mathbf{I}_k$ and $\mathbf{D}_k$ to update Gaussian parameters.

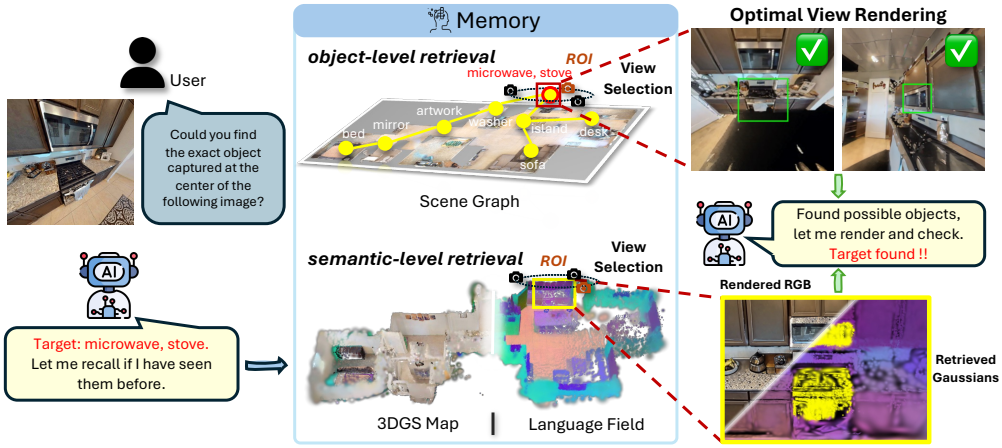


Figure 2: **Demonstration of Multi-level Retrieval-Rendering.** Given a task-related target, the agent retrieves ROIs based on object-level and semantic-level cues. Subsequent viewpoint selection and rendering enable the agent to re-observe these regions for further reasoning.

Online Language Field Generation. To enable semantic-level grounding, we equip each 3D Gaussian g_i with a language embedding $\mathbf{f}_i$. Existing methods typically achieve this by either optimizing high-dimensional semantic features (Zhou et al., 2024; Qin et al., 2024) or assigning features after scene reconstruction (Marrie et al., 2025; Cheng et al., 2024a). To enable efficient online mapping, we propose an optimization-free approach that directly lifts dense 2D semantics onto 3D Gaussians.

We first extract dense 2D features $\mathbf{f}_{p,k}^{2D}$ at pixel p in keyframe k using a pixel-wise CLIP encoder (Xie et al., 2024) combined with a lightweight super-resolution decoder (Katragadda et al., 2025). To minimize memory overhead, these features are compressed from 768 to 32 dimensions via a pre-trained autoencoder (Katragadda et al., 2025). Instead of relying on multi-pass rendering or additional optimization, we perform a weight-consistent reverse aggregation. In standard forward rendering, a 2D pixel's feature is generated by alpha-blending 3D Gaussian features. Recognizing this structural symmetry, we invert the process: we distribute the 2D pixel features back to the 3D Gaussians using the exact same blending weights. Let $w_{i,p,k}^t$ denote the blending weight of Gaussian g_i at pixel p in keyframe k during time step t. Then the online update of the Gaussian feature $\mathbf{f}_i^t$ at time t is defined as:

$$\mathbf{f}_i^t = \frac{W_i^{t-1}\mathbf{f}_i^{t-1} + \sum_{k\in\mathcal{T}_t}\sum_p w_{i,p,k}^t \mathbf{f}_{p,k}^{2D}}{W_i^t}, \quad W_i^t = W_i^{t-1} + \sum_{k\in\mathcal{T}_t}\sum_p w_{i,p,k}^t. \tag{2}$$

Since the blending weight is inherently computed during the forward RGB-D rendering process, this formulation generates a dense 3D language field without introducing any optimization overhead.

In addition to the 3DGS map and language field, we maintain an object-level scene graph for structured region retrieval. Following ConceptGraphs (Gu et al., 2024), we perform object detection, matching, and merging on the incoming RGB-D stream to obtain a consolidated set of objects with their 3D locations and semantic labels. For each object, we store the highest-confidence detection pose for subsequent region retrieval. Additionally, we maintain a frontier map and a TSDF (Truncated Signed Distance Function) map to facilitate region retrieval and exploration.

3.2 Multi-level Retrieval-Rendering

In this section, we describe the proposed multi-level retrieval-rendering mechanism, illustrated in fig. 2. We first extract multi-level cues from memory to retrieve task-relevant regions, and then select suitable viewpoints that allow the agent to "revisit" these regions for further reasoning.

Object-level Region Retrieval. At each step, we provide the VLM with the input question together with an object list maintained in the scene graph. The VLM then ranks the objects based on their semantic relevance to the question and selects the top-K_{obj} candidates. Each selected object defines an object-level region of interest (ROI) for subsequent viewpoint selection and rendering.

Semantic-level Region Retrieval. At the beginning of each episode, we provide the VLM with the input question and prompt it to identify potential relevant target objects or semantic entities associated with the task. The VLM returns a set of target descriptions, which are subsequently encoded into CLIP embeddings. During each exploration step, we query the 3D language field with these embeddings and retrieve Gaussians whose feature cosine similarity exceeds a threshold τ_{clip}. Since the retrieved Gaussians may span multiple regions, we further cluster them into spatially coherent groups. We use a KD-Tree (Maneewongvatana & Mount, 1999) to identify neighbors within distance τ_d and construct an adjacency graph. Connected components are treated as candidate clusters, from which we discard small clusters and retain the top-K_{cluster} ranked by mean cosine similarity as semantic-level ROIs.

Optimal Viewpoint Selection for Visual Recall. For each retrieved ROI, the agent selects an optimal viewpoint, and renders it for VLM reasoning. We adopt a sample-then-score strategy. Specifically, for each ROI, we sample viewpoints on a horizontal circle centered at its 3D bounding box, using 36 azimuth angles at 10° intervals and three elevations $(-10^\circ, 0^\circ, 15^\circ)$, resulting in 108 candidates. These candidates are then evaluated and scored as follows:

Phase 1: As a preliminary step, we perform coarse feasibility filtering by discarding any camera poses that lie inside obstacle regions of the TSDF map. We then evaluate the remaining candidates based on a ray visibility score S_{vis} and a projected area score S_A.

For S_{vis}, we sample 14 points on the ROI bounding box (8 vertices and 6 face centers) and use TSDF ray marching to compute the visible ratio S_{vis} from candidate viewpoints. For S_A, we project the 8 vertices to the candidate view and compute the 2D area A. To encourage an appropriate viewing scale, avoiding views that are too close or too far, we define S_A using a Gaussian penalty:

$$S_{\text{vis}} = \frac{N_{\text{visible}}}{14}, \quad S_A = \exp\left(-\frac{(A - A^*)^2}{2{\sigma_a}^2}\right), \tag{3}$$

where A^* is the target projected area and σ_a controls tolerance to scale deviation. Candidate poses are ranked by the combined score $S_{\text{vis}} + S_A$, and the top-10 poses are advanced to the final phase.

Phase 2: For the top-10 candidate viewpoints, we further assess the rendering quality using the 3DGS opacity map. We render an opacity map for each pose and compute the average opacity within the projected 2D ROI. Due to the compositional nature of 3DGS, a higher accumulated opacity indicates a stronger surface presence and better visual observability. The opacity score S_{opa} is defined as:

$$S_{\text{opa}} = \frac{1}{|\Omega|} \sum_{p \in \Omega} \alpha_p, \tag{4}$$

where Ω is the set of pixels within the projected ROI , and α_p is the accumulated opacity value at pixel p. Finally, we select the candidate with the highest combined score as the optimal rendering viewpoint for visual recall:

$$S_{\text{final}} = S_{\text{vis}} + S_A + S_{\text{opa}}. \tag{5}$$

3.3 ROI RENDERING & VISION-LANGUAGE REASONING

Given the selected optimal viewpoint, we render the ROI for visual analysis. In addition, we also render the highest-confidence viewpoint of the object-level ROI during detection to provide complementary visual evidence. We further incorporate a single-step diffusion model (Wu et al., 2025) to enhance the rendering quality. The rendered views are then fed into the VLM to determine whether the question can be answered with the current observations. The detailed prompting strategy is provided in section A.12. If the VLM determines that the available visual evidence is insufficient, the agent proceeds to further explore the environment following our Hybrid Exploration Strategy.

3.4 HYBRID EXPLORATION STRATEGY

Our exploration module follows the frontier-based paradigm (Yamauchi, 1997), where frontiers are defined as the boundaries between known free space and unexplored regions in the occupancy map. At each exploration step, we extract a set of candidate frontiers from the occupancy map and evaluate them using a hybrid criterion that balances semantic relevance and geometric coverage.

Semantic Relevance. For each candidate frontier, we feed the corresponding observation to a VLM conditioned on the task query. The VLM outputs a normalized relevance score $s_i^{\text{sem}} \in [0, 1]$, reflecting how likely the frontier is to reveal information related to the task.

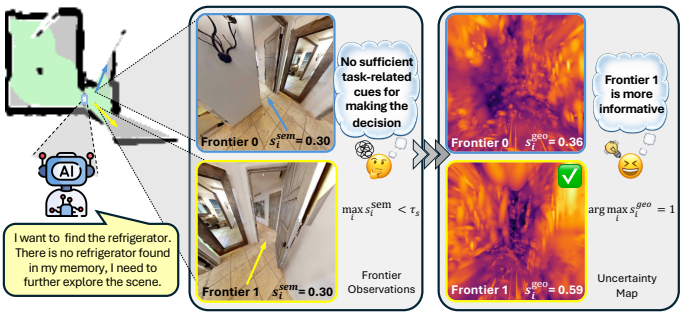


Figure 3: **Hybrid Exploration Strategy.** When frontier observations lack sufficient task-related cues, we use an information gain-based score to select the most informative frontier for further exploration.

Geometric Coverage. To promote exploration toward geometrically informative regions, we score each frontier ξ_i (with an associated camera pose $\mathbf{T}_i$) based on the expected information gain of acquiring new observations $\mathcal{Y}_i$. Following information-theoretic formulations (Lindley, 1956), this corresponds to the reduction in the differential entropy $\mathcal{H}$ of the 3DGS parameters θ given the active window $\mathcal{W}_t$:

$$s_i^{\text{geo}} = \mathcal{H}(\theta \mid \mathcal{W}_t) - \mathcal{H}(\theta \mid \mathcal{W}_t, \mathcal{Y}_i). \tag{6}$$

Under the Laplace approximation, this information gain can be estimated by the D-optimality criterion, *i.e.*, the log-determinant increase of the Fisher Information Matrix (FIM) (Houlsby et al., 2011; Kirsch et al., 2019; Jiang et al., 2023):

$$\mathcal{I}(\theta; \mathcal{Y}_i \mid \mathcal{W}_t) \approx \frac{1}{2} \log\det(\mathbf{I}_t + \mathbf{I}_i) - \frac{1}{2} \log\det(\mathbf{I}_t)\,, \tag{7}$$

where $\mathbf{I}_t$ and $\mathbf{I}_i$ denote the current and incremental FIMs, approximated by rendering Jacobians. Since computing the global log-determinant is intractable for 3DGS, we adopt an efficient T-optimality surrogate and use the trace of the incremental FIM as the geometric score:

$$s_i^{\text{geo}} \approx \text{Tr}(\mathbf{I}_i). \tag{8}$$

This score can be computed from 3DGS rendering gradients without ground-truth supervision.

Exploration Strategy. Based on the two scores defined above, we design the following hybrid exploration strategy, as shown in fig. 3. At each step, if $\max_i s_i^{\text{sem}} > \tau_s$, we select the frontier with the highest semantic score. Otherwise, when no frontier provides sufficient task-relevant information for the VLM, we select the frontier that maximizes the geometric exploration score:

$$i^* = \begin{cases} \arg\max_i s_i^{\text{sem}}, & \text{if } \max_i s_i^{\text{sem}} > \tau_s, \\ \arg\max_i s_i^{\text{geo}}, & \text{otherwise.} \end{cases} \tag{9}$$

4 Experiments

To systematically evaluate GSMem, we first evaluate Active Embodied Question Answering (A-EQA) using the OpenEQA benchmark (Majumdar et al., 2024), where the agent starts in an unseen environment and must actively explore to gather evidence for answering task queries, testing the general exploration and reasoning ability. Second, we consider a Multimodal Lifelong Navigation setting to assess memory reliability over long-horizon interactions, using GOAT-Bench (Khanna et al., 2024), where agents sequentially locate multiple targets in unseen scenes. In this setting, the agent continuously accumulates observations across episodes, and targets may appear in previously explored regions, requiring the system to retrieve and utilize stored memory effectively.

4.1 Active Embodied Question Answering

Metrics. Following (Yang et al., 2025b), we use the 184-question evaluation set of OpenEQA and report LLM-Match and LLM-Match SPL. LLM-Match is scored by GPT-4 (Achiam et al., 2023) on a 0–100 scale, while LLM-Match SPL additionally accounts for exploration efficiency.

Baselines. Following prior work (Majumdar et al., 2024; Zhu et al., 2025; Yang et al., 2025b), we compare GS-Mem with four categories of baselines: (1) **Blind LLMs**: LLMs (Hurst et al., 2024; Grattafiori et al., 2024) answer the question using only textual input. (2) **LLM + Captions**: Frame captions generated by LLaVA-1.5 (Liu et al., 2024) are provided to the LLM. (3) **VLMs**: Observations are directly fed into VLMs (Hurst et al., 2024; Yang et al., 2025a). (4) **VLM Exploration**: Explore-EQA (Ren et al.), ConceptGraphs (Gu et al., 2024), and 3D-Mem (Yang et al., 2025b). For fairness, all methods in VLM exploration use GPT-4o (Hurst et al., 2024) as the VLM backbone.

Results. We report the quantitative results of active EQA in table 1 (a). GSMem achieves state-of-the-art performance on this benchmark. Compared to high-level graph-based methods, GSMem maintains dense geometric and visual representations, providing richer visual evidence for downstream reasoning. Compared with keyframe-based 3D-Mem, GSMem selects better viewpoints for retrieved regions, yielding improved visual coverage and higher-quality renderings. These more informative visual inputs further enhance VLM reasoning performance. At the same time, our hybrid exploration strategy also improves efficiency during active exploration.

Table 1: **Experiments on A-EQA and GOAT-Bench.** "CG" denotes ConceptGraphs.

Method	LLM-Match ↑	LLM-Match SPL ↑
Blind LLMs		
GPT-4o	35.9	-
LLaMA-3.1-8B-Instruct	32.2	-
LLM with Frame Captions		
GPT-4o w/ LLaVA-1.5-13B	33.11	-
LLaMA-3.1 w/ LLaVA-1.5-13B	30.84	-
VLMs		
GPT-4o	47.6	-
Qwen3-VL-8B-Instruct	44.0	-
VLM Exploration		
Explore-EQA	46.9	23.4
CG w/ Frontier Snapshots	47.2	33.3
3D-Mem	52.6	42.0
GSMem (Ours)	**55.4**	**43.8**

(a) A-EQA.

Method	Success Rate ↑	SPL ↑
GOAT-Bench Baselines		
Modular GOAT	24.9	17.2
Modular CLIP on Wheels	16.1	10.4
SenseAct-NN Monolithic	12.3	6.8
SenseAct-NN Skill Chain	29.5	11.3
VLM Exploration		
VLMnav	20.1	9.6
DyNaVLM	25.5	10.2
TANGO	32.1	16.5
MTU3D	47.2	27.7
3D-Mem	62.9	44.7
GSMem (Ours)	**67.2**	**46.9**

(b) GOAT-Bench.

4.2 Multi-modal Lifelong Navigation

Metrics. We evaluate on the **valid unseen** split of GOAT-bench, which contains 36 scenes and over 2,600 subtasks. Following GOAT-Bench, we report Success Rate (SR) and Success weighted by Path Length (SPL). A subtask is successful if the agent stops within 1 meter of the target.

Baselines. We include four reinforcement learning-based baselines from the GOAT-Bench paper. In addition, we compare against several VLM-based exploration methods. VLMNav (Goetting et al., 2025) and DyNaVLM (Ji et al., 2025) employ open-source VLM backbones, while recent methods such as TANGO (Ziliotto et al., 2025), MTU3D (Zhu et al., 2025), and 3D-Mem (Yang et al., 2025b) utilize commercial models (*i.e.*, GPT-4o) as their VLM backbone. To ensure a fair comparison, we also adopt GPT-4o as the underlying VLM.

Results. The quantitative results on GOAT-Bench are presented in table 1 (b). Under the lifelong navigation setting, GSMem achieves a larger performance gain than in the A-EQA benchmark, suggesting that our persistent memory representation is particularly beneficial for long-horizon scenarios. Moreover, the hybrid exploration strategy enables more efficient exploration, as reflected in the improved SPL metric. We also provide detailed comparison with 3DMem under different task modalities in appendix section A.4.

4.3 Case Analysis

We present several case studies in fig. 4 to evaluate the robustness of GSMem against the inherent limitations of discrete and view-dependent memories.

Missing and Incorrect Detections. A primary failure mode of current methods is their heavy reliance on object detectors. In fig. 4a, the target ("white robe") is missed during mapping, which is an irrecoverable omission for 3D-Mem since it is never explicitly labeled. In fig. 4b, 3D-Mem finds a relevant view via contextual cues, but the undetected target ("ficus tree") still cannot be precisely grounded for navigation. In fig. 4c, the baseline misidentifies a "white door" as a "refrigerator", misleading the VLM's decision. In all three cases, GSMem retrieves the target from its continuous



Figure 4: **Case analysis.** We analyze several cases where scene-graph and view-based representations fail, and demonstrate the advantages of 3DGS-based memory. The images shown are the views selected by VLM to answer the questions. Examples (a-c) illustrate failures of the scene-graph detector, while example (d) highlights how optimal viewpoint rendering benefits the VLM's reasoning.

language field via semantic similarity, enabling correct localization and navigation despite missing or erroneous detections.

View-Dependency and Resolution Limits. The fourth example (fig. 4d) demonstrates the necessity of optimal viewpoint selection and rendering. Although the baseline captures the correct region through its object detector, its static egocentric snapshots suffer from limited resolution and suboptimal perspective, preventing the VLM from recognizing the hanging clothes. By leveraging the re-observability nature of 3DGS, GSMem re-visits the region with an optimal viewpoint. This provides the VLMs with better visual evidence for reasoning and decision-making.

4.4 Ablation Study

We conduct module ablations on 10% of the GOAT-Bench "valid unseen" split (left part of table 2). Removing object-level or semantic-level retrieval causes the largest drops, especially object-level retrieval (71.1 $\rightarrow$ 50.1 SR), confirming that the two levels are complementary. Viewpoint selection and diffusion enhancement both improve SR through better-aligned and higher-quality renderings, while hybrid exploration mainly improves SPL (47.8 $\rightarrow$ 51.9) via coverage-aware exploration.

We further analyze viewpoint selection and memory representation in the right part of table 2. Due to API cost, these experiments adopt the open-source Qwen3-VL-8B backbone, so the absolute numbers are not directly comparable with the left part. More ablations can be found in the appendix.

Viewpoint Selection. We compare our viewpoint selection with rendering multiple random views of each ROI, while keeping ROI retrieval, the VLM backbone, and the reasoning pipeline fixed. Although more random views improve performance, they require additional rendering and VLM tokens and may introduce redundant evidence. In contrast, our method renders a single view selected by visibility, projected coverage, and accumulated opacity. It outperforms random 3-view rendering by 3.6 SR and 4.8 SPL, and even slightly exceeds random 6-view rendering at a substantially lower cost, offering an effective accuracy-efficiency trade-off.

Memory Representation. To isolate the effect of the memory representation, we use only object-level retrieval and keep the exploration and reasoning pipeline fixed. We compare an image-snapshot memory, which returns the highest-confidence detection frame following 3D-Mem, with our 3DGS memory, which renders an optimal view of the same target. Since only the memory representation changes, the improvement from 53.7 to 59.7 SR directly reflects the benefit of free-viewpoint recollection enabled by 3DGS. The full GSMem further adds the language field and hybrid exploration based on the property of 3DGS, reaching 63.0 SR. This shows that 3DGS contributes not

Table 2: **Ablation studies on GOAT-Bench.** *Left*: effect of removing individual components of GSMem. *Right*: effect of the view selection strategy and the memory representation.

Method	Success Rate ↑	SPL ↑
w/o object-level	50.1	29.9
w/o semantic-level	66.6	45.3
w/o viewpoint selection	68.4	46.3
w/o diffusion	69.8	48.2
w/o hybrid explore	69.9	47.8
GSMem (Ours)	**71.1**	**51.9**

Method	Success Rate ↑	SPL ↑
View Selection		
Random 3	59.4	40.8
Random 6	62.9	43.7
Optimal (Ours)	**63.0**	**45.6**
Memory Representation		
Image Snapshots	53.7	36.7
3DGS Memory	59.7	42.1
GSMem (Full)	**63.0**	**45.6**

only through free-viewpoint recollection, but also as the shared representation underlying the full retrieval and exploration framework.

4.5 Analysis of Real-world Practicality

We analyze the runtime of each component in our framework on a subset of GOAT-Bench, and the results are shown in fig. 5. We account for modules that can operate concurrently and estimate the effective runtime of the system in a practical setting. For this estimation, we replace the OpenAI API with an offline-deployed Qwen3-VL-8B model and remove the single-step diffusion process during rendering. The navigation framework runs on an RTX 4090, while the Qwen3 model is deployed on an H200 GPU. Under this multi-process setup, the average processing time per step is around 1.2 seconds. We also provide a real robot demo in supplementary materials and appendix section A.9.

To evaluate robustness to localization errors, we simulate realistic odometry drift by injecting Gaussian noise into consecutive relative poses and accumulating it over time. The noise scales with traveled distance and rotation, and is calibrated to produce a median final drift above 1 m, exceeding the success threshold. Similar to the ablation study, this experiment adopts the Qwen3-VL-8B backbone due to API cost. As shown in table 3, such drift reduces the success rate of GSMem from 63.0 to 50.5. Applying bundle adjustment (BA) during mapping reduces the drift from 1.49 m / 14.28° to 0.59 m / 5.89° and recovers the performance to 57.4 SR and 40.3 SPL, demonstrating that GSMem remains effective under realistic localization errors when combined with standard pose correction.

Table 3: Effect of bundle adjustment (BA) on navigation performance and trajectory drift.

Method	Success Rate	SPL	Drift Before BA	Drift After BA
GSMem	50.5	31.7	1.536 m / 13.96°	–
GSMem + BA	57.4	40.3	1.493 m / 14.28°	0.59 m / 5.89°
GSMem (w/o noise)	63.0	45.6	–	–

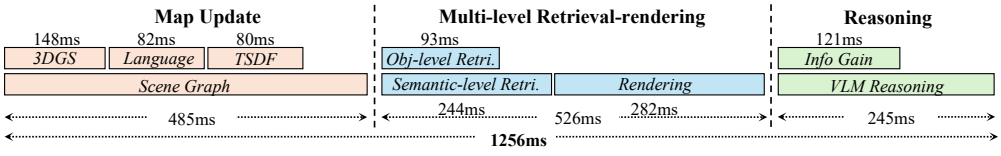


Figure 5: **Runtime analysis for real-world multi-process setup.**

5 Conclusion

In this work, we present **GSMem**, a zero-shot embodied exploration and reasoning framework built upon 3D Gaussian Splatting that equips embodied agents with *post-hoc re-observability*. This capability allows the agent to revisit previously explored regions from arbitrary viewpoints without physically navigating to them, enabling more efficient memory retrieval and precise recall in lifelong navigation scenarios. As a result, GSMem achieves higher success rates and more efficient navigation compared to existing methods.

AI USE STATEMENT

In this work, we did not use generative AI tools for research tasks requiring disclosure, including developing or refining research hypotheses, designing the methodology or experiments, implementing the proposed method, analyzing data, or interpreting experimental results. Other required-disclosure tasks, such as generating synthetic datasets, developing theoretical models, formulating or proving mathematical claims, translation, and qualitative or thematic data analysis, were not applicable to this work.

Additionally, we used generative AI tools to improve the readability and presentation of the manuscript and to organize author-provided experimental results into tables. The AI tools were used only to assist with the presentation of existing results and were not used to generate, compute, modify, or interpret the reported experimental results. We reviewed all AI-assisted work: all AI-assisted text was proofread and revised by the authors, and all values in AI-assisted tables were manually verified against the original experimental results. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the implementation details, model configurations, training settings, and hyperparameters necessary to reproduce our experiments in the main paper and the appendix. To further facilitate reproducibility, we plan to publicly release the source code after the review process, including the training and evaluation scripts as well as model checkpoints corresponding to the results reported in the paper. Together, these materials are intended to enable reproduction of both the proposed method and the reported experimental results.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Matthew Chang, Theophile Gervet, Mukul Khanna, Sriram Yenamandra, Dhruv Shah, So Yeon Min, Kavit Shah, Chris Paxton, Saurabh Gupta, Dhruv Batra, et al. Goat: Go to any thing. *arXiv preprint arXiv:2311.06430*, 2023.

Matthew Chang, Theophile Gervet, Mukul Khanna, Sriram Yenamandra, Dhruv Shah, So Min, Kavit Shah, Chris Paxton, Saurabh Gupta, Dhruv Batra, et al. Goat: Go to any thing. 2024.

Jiahuan Cheng, Jan-Nico Zaech, Luc Van Gool, and Danda Pani Paudel. Occam's lgs: An efficient approach for language gaussian splatting. *arXiv preprint arXiv:2412.01807*, 2024a.

Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16901–16911, 2024b.

Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

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

Dylan Goetting, Himanshu Gaurav Singh, and Antonio Loquercio. End-to-end navigation with vision-language models: Transforming spatial reasoning into question-answering. In *International Conference on Neuro-symbolic Systems*, pp. 22–35. PMLR, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

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.

Dongjiao He, Wei Xu, Nan Chen, Fanze Kong, Chongjian Yuan, and Fu Zhang. Point-lio: robust high-bandwidth light detection and ranging inertial odometry. *Advanced Intelligent Systems*, 5 (7):2200459, 2023.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

Nathan Hughes, Yun Chang, Siyi Hu, Rajat Talak, Rumaia Abdulhai, Jared Strader, and Luca Carlone. Foundations of spatial perception for robotics: Hierarchical representations and real-time systems. *The International Journal of Robotics Research*, 43(10):1457–1505, 2024.

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.

Zihe Ji, Huangxuan Lin, and Yue Gao. Dynavlm: Zero-shot vision-language navigation system with dynamic viewpoints and self-refining graph memory. *arXiv preprint arXiv:2506.15096*, 2025.

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

Saimouli Katragadda, Cho-Ying Wu, Yuliang Guo, Xinyu Huang, Guoquan Huang, and Liu Ren. Online language splatting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 25882–25892, 2025.

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

Mukul Khanna, Ram Ramrakhya, Gunjan Chhablani, Sriram Yenamandra, Theophile Gervet, Matthew Chang, Zsolt Kira, Devendra Singh Chaplot, Dhruv Batra, and Roozbeh Mottaghi. Goatbench: A benchmark for multi-modal lifelong navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16373–16383, 2024.

Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32, 2019.

Dennis V Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 26296–26306, 2024.

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.

Songrit Maneewongvatana and David M Mount. Analysis of approximate nearest neighbor searching with clustered point sets. *arXiv preprint cs/9901013*, 1999.

Juliette Marrie, Romain Ménégaux, Michael Arbel, Diane Larlus, and Julien Mairal. Ludvig: Learning-free uplifting of 2d visual features to gaussian splatting scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7440–7450, 2025.

Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20051–20060, 2024.

Allen Z Ren, Jaden Clark, Anushri Dixit, Masha Itkina, Anirudha Majumdar, and Dorsa Sadigh. Explore until confident: Efficient exploration for embodied question answering. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*.

Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pp. 402–419. Springer, 2020.

Jay Zhangjie Wu, Yuxuan Zhang, Haithem Turki, Xuanchi Ren, Jun Gao, Mike Zheng Shou, Sanja Fidler, Zan Gojcic, and Huan Ling. Difix3d+: Improving 3d reconstructions with single-step diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26024–26035, 2025.

Bin Xie, Jiale Cao, Jin Xie, Fahad Shahbaz Khan, and Yanwei Pang. Sed: A simple encoder-decoder for open-vocabulary semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3426–3436, 2024.

Brian Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*, pp. 146–151. IEEE, 1997.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Fan Yang, Chao Cao, Hongbiao Zhu, Jean Oh, and Ji Zhang. Far planner: Fast, attemptable route planner using dynamic visibility update. In *2022 ieee/rsj international conference on intelligent robots and systems (iros)*, pp. 9–16. IEEE, 2022.

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

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.

Yuheng Yuan, Qiuhong Shen, Xingyi Yang, and Xinchao Wang. 1000+ fps 4d gaussian splatting for dynamic scene rendering. *Advances in Neural Information Processing Systems*, 38:74665–74683, 2026.

Lin Zeng, Boming Zhao, Jiarui Hu, Xujie Shen, Ziqiang Dang, Hujun Bao, and Zhaopeng Cui. Gaussianupdate: Continual 3d gaussian splatting update for changing environments. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 25800–25809. IEEE, 2025.

Shijie Zhou, Haoran Chang, Sicheng Jiang, Zhiwen Fan, Zehao Zhu, Dejia Xu, Pradyumna Chari, Suya You, Zhangyang Wang, and Achuta Kadambi. Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21676–21685, 2024.

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.

A APPENDIX

A.1 PRELIMINARY OF 3DGS

3D Gaussian Splatting (3DGS) represents a scene as a set of anisotropic Gaussians $\mathcal{G} = \{g_i\}_{i=1}^{N}$, where each Gaussian is parameterized by its mean position $\mathbf{x}_i$, covariance $\mathbf{\Sigma}_i$, opacity σ_i, and color $\mathbf{c}_i$. For efficiency, we set the SH degree to 0 and directly use RGB colors.

Given a scene represented by 3D Gaussians $\mathcal{G}$, novel views are rendered by projecting the Gaussians onto the target image plane and differentiably compositing them. Given camera intrinsics $\mathbf{K}$ and world-to-camera pose $\mathbf{T}_{cw} = [\mathbf{R}_{cw}|\mathbf{t}_{cw}]$, each 3D Gaussian is projected to a 2D Gaussian as:

$$\mathbf{x}_i' = \pi(\mathbf{K}\mathbf{T}_{cw}\tilde{\mathbf{x}}_i), \quad \mathbf{\Sigma}_i' = \mathbf{J}_i\mathbf{R}_{cw}\mathbf{\Sigma}_i\mathbf{R}_{cw}^{\top}\mathbf{J}_i^{\top}, \tag{10}$$

where $\pi(\cdot)$ is homogeneous normalization and $\mathbf{J}_i$ is the Jacobian of the projection. For each pixel $\mathbf{p}$, overlapping Gaussians are depth-sorted and alpha-composited. The contribution of Gaussian g_i is

$$w_{i,\mathbf{p}} = \alpha_i(\mathbf{p}) \prod_{j=1}^{i-1} (1 - \alpha_j(\mathbf{p})), \tag{11}$$

where $\alpha_i(\mathbf{p})$ is determined by its learned opacity and the projected 2D covariance. The rendered color and depth are computed as:

$$\mathbf{C}(\mathbf{p}) = \sum_{i\in\mathcal{N}} w_{i,\mathbf{p}}\mathbf{c}_i, \quad D(\mathbf{p}) = \sum_{i\in\mathcal{N}} w_{i,\mathbf{p}} z_i, \tag{12}$$

where $\mathcal{N}$ denotes the Gaussians overlapping $\mathbf{p}$ and z_i is the camera-space depth of g_i.

A.2 IMPLEMENTATION DETAILS & HYPERPARAMETER SELECTION

We present the implementation details of GSMem in this section.

Multi-level retrieval. The number of retrieved object-level regions K_{obj} is set to 10 based on the ablation study in table 4. The semantic similarity threshold τ_{clip} in semantic-level is set to 0.8, according to emperical results. The number of accepted clusters $K_{cluster}$ is set to 3. The spatial radius τ_d of KD-Tree is set to 0.2.

Viewpoint selection. The optimal coverage A^* for calculating the coverage score S_A is set to 0.15. To determine A^*, we evaluate VLM object recognition under different object coverage ratios. We select 20 objects from HM3D, center-crop each object to occupy varying image areas, and ask the VLM to identify the centered object. We find that $A^* = 0.15$ best balances environmental context and object details, yielding the highest detection accuracy.

Hybrid exploration. When semantic score $s_I^{sem} \leq \tau_s$, the agent adopts an information gain-based exploration strategy. The threshold τ_s is set to 0.4 according to the ablation study in table 4.

Detection model. For a fair comparison, we use YOLOv8x-World (Cheng et al., 2024b) from Ultralytics as the detection model and adopt the 200-class set from ScanNet (Dai et al., 2017) as the detection categories, following 3DMem (Yang et al., 2025b). Detailed prompts for prompting the VLM are provided in section A.12.

Stop condition. The agent terminates when it answers the question, reaches the target, or exceeds the maximum step limit of 50.

A.3 ABLATION STUDY ON HYPERPARAMETERS

We analyze the sensitivity to two key hyperparameters: the semantic threshold τ_s for hybrid exploration and the number of rendered object regions K_{obj} in table 4. A large τ_s overemphasizes geometric coverage and delays target discovery, while a small τ_s may prematurely bias the agent toward semantic cues. For K_{obj}, too few regions provide insufficient context, whereas too many introduce redundant visual evidence. We set $\tau_s = 0.4$ and $K_{obj} = 10$ in our experiments.

To examine whether the historical replay budget affects long-horizon memory fidelity (table 5), we vary the number of historical views while keeping the active window and all other components fixed.

Table 4: **Ablation study on τ_s and K_{obj}.**

Variable	Success Rate ↑	SPL ↑
Exploration τ_s		
$\tau_s = 0.2$	69.3	50.8
$\tau_s = 0.4$	71.1	51.9
$\tau_s = 0.6$	70.6	49.6
Region retrieval K_{obj}		
$K_{obj} = 4$	68.8	48.6
$K_{obj} = 6$	67.6	48.9
$K_{obj} = 10$	71.1	51.9
$K_{obj} = 12$	70.3	49.7

The experiment is conducted on 36 scenes using Qwen3-VL-8B, and rendering quality is evaluated on held-out non-keyframes never used for optimization. From the results in table 5, we can see that the rendering metrics are already strong with only two historical frames. Increasing the number of historical views further improves held-out rendering fidelity, while SR and SPL remain relatively stable and show no monotonic trend. This suggests that two replay views already provide sufficient memory quality for retrieval and recognition, while downstream performance is also governed by exploration, retrieval, and VLM reasoning. Since additional replay frames increase optimization cost without consistently improving navigation, we use two historical views as an efficiency-performance trade-off.

Table 5: Ablation study on the historical reply budget and its effect on reconstruction quality and GOAT-bench performance.

# Views	PSNR	SSIM	LPIPS	Success Rate	SPL
2	28.569	0.8659	0.1647	61.81	39.82
4	28.718	0.8793	0.1525	60.19	37.50
6	29.423	0.8883	0.1364	62.96	39.87

A.4 Extended results on GOAT-Bench

Table 6: **Extended results on GOAT-Bench (Khanna et al., 2024) by task modalities**, evaluated a subset of the "Val Unseen" split. "CG w/ FS" denotes ConceptGraphs (Gu et al., 2024) with frontier snapshots. **Bold** indicates the best, underlined values denote the second-best.

	Object Category		Language		Image		Overall	
Method	**SR**	**SPL**	**SR**	**SPL**	**SR**	**SPL**	**SR**	**SPL**
Explore-EQA	64.7	48.4	42.9	22.7	56.8	41.8	55.0	37.9
CG w/ FS	65.3	44.7	55.0	38.9	64.0	52.8	61.5	45.3
3D-Mem	66.7	49.1	59.3	**45.0**	69.3	54.0	65.1	49.3
GSMem	**74.6**	**50.6**	**61.5**	43.5	**76.1**	**62.2**	**71.1**	**51.9**

We provide extended results on GOAT-Bench (Khanna et al., 2024) across different task modalities, as shown in table 6. "Object Categories" means that the agent is directly given an object class label; "Language" means that the agent is provided with a textual description of the target object; and "Image" means that the agent is given an image of the target. GSMem achieves the highest success rate (SR) across all modalities and also obtains the highest overall Success weighted by Path Length (SPL).

A.5 Performance on open-source VLM

GSMem is VLM-agnostic: the backbone can be replaced without modifying the 3DGS memory, retrieval, viewpoint selection, or rendering modules. We use GPT-4o in the main experiments because baseline methods in our comparison also adopt GPT-4o, ensuring a fair comparison. To evaluate

Table 7: Performance on GOAT-bench across different VLM backbones.

Method	Success Rate	SPL
GPT-4o		
3D-Mem	65.1	49.3
Ours	**71.1**	**51.9**
Qwen3-VL-8B		
3D-Mem	54.9	39.1
Ours	**63.0**	**45.6**

backbone dependence, we additionally test GSMem and 3D-Mem with the open-source Qwen3-VL-8B model (table 7), the experiments are conducted on the same subset as table 2 (left) in the main paper. Although Qwen3-VL-8B lowers the absolute performance of both methods, GSMem consistently outperforms 3D-Mem. The gains are 6.0 SR and 2.6 SPL points with GPT-4o, and 8.1 SR and 6.5 SPL points with Qwen3-VL-8B. This shows that GSMem's advantage is not specific to GPT-4o and is preserved with an open-source VLM.

A.6 Memory and scalability

Although GSMem maintains a dense and re-renderable 3D Gaussian representation, the number of Gaussians is deliberately constrained through multiple mechanisms (e.g. sparse initialization and aggressive pruning), resulting in a compact practical memory footprint.

Across the evaluated scenes in table 8 , the Gaussian map only contains an average of 209,015 Gaussians and occupies only 62 MB of memory. The mean map contains 209,015 Gaussians and occupies 62 MB, while even the largest scene requires only 109 MB. Under an approximately linear scaling assumption, a scene 100 times larger would require roughly 6-11 GB, which would still fit within the memory capacity of a single RTX 4090

This modest memory cost enables a capability unavailable to snapshot-based memories: post-hoc re-observation from previously unoccupied viewpoints, allowing the agent to recover visual evidence that was occluded, low-resolution, or absent from the originally stored frames.

Table 8: Statistics of trajectories and Gaussian memory size.

Statistic	Trajectory Length (m)	Number of Gaussians	Gaussian Map Memory (MB)
Mean	56	209,015	62
Minimum	11	110,130	33
Maximum	134	366,672	109

A.7 Performance analysis under different scene size

GOAT-Bench includes many large-scale indoor environments, with an average projected floor area of 231.4 m² and a maximum of 537.4 m². To further analyze scalability on scene size, we group the scenes by area and report the corresponding navigation performance in table 9.

Table 9: Life-long navigation performance across different scene sizes.

Scene Size (m^2)	Success Rate	SPL
63–153	76.8	55.0
153–223	74.2	50.3
223–537	67.9	46.5

We observe a more noticeable performance drop once the scene size exceeds 223 m². Our analysis suggests that this is not solely due to longer exploration trajectories. Ultra-large scenes often contain visually similar regions, which can cause the VLM to make an overconfident goal prediction and terminate exploration prematurely before the true target region is sufficiently covered. Nevertheless, GSMem still achieves 67.9% success rate and 46.5 SPL in the largest scenes, indicating that the persistent memory remains effective, while large-scale performance is mainly limited by premature VLM decisions and insufficient exploration coverage.

A.8 Analysis of cascading failure

We supplement the module-removal ablation with a cascading-failure analysis (table 10) in multi-level retrieval module. In our pipeline, 3DGS mapping and viewpoint selection are deterministic geometric operations, and they do not produce an explicit binary success/failure output. We therefore focus the analysis on the two retrieval branches, object-level retrieval (obj) and semantic-level retrieval (sem), where success can be objectively determined by whether the target is included in the retrieved candidates. Importantly, when object-level retrieval fails, the semantic branch still retrieves the correct target in 68.9% of cases. Conversely, when semantic-level retrieval fails, the object-level branch remains successful in 73.9% of cases. This directly demonstrates that the two retrieval mechanisms provide complementary coverage rather than forming a strictly sequential dependency.

Table 10: Conditional ROI retrieval success rates.

Conditional Retrieval Metric	**ROI Retrieval Success Rate**
$P(\text{sem} \mid \text{obj} = Y)$	86.0%
$P(\text{sem} \mid \text{obj} = N)$	68.9%
$P(\text{obj} \mid \text{sem} = Y)$	88.7%
$P(\text{obj} \mid \text{sem} = N)$	73.9%

We further analyze how their joint outcomes propagate to final task success, results are shown in table 11. When both branches succeed, the final success rate reaches 87.0%. When only one branch succeeds, the system still completes 50.0% or 37.5% of subtasks, whereas simultaneous retrieval failure leads to 0% success. The remaining failures after successful retrieval capture errors in subsequent stages, including VLM reasoning, and navigation execution. Overall, these results validate the claimed complementarity between object- and semantic-level retrieval and explicitly characterize how retrieval failures propagate to end-to-end performance.

Table 11: Effect of object-level and semantic-level retrieval on final task success rate.

Object-Level Retrieval (obj)	**Semantic-Level Retrieval (sem)**	**Success Rate**
✓	✓	**87.0%**
✓	×	50.0%
×	✓	37.5%
×	×	0.0%

A.9 Real Robot Demo

We provide a real-world demonstration video in the supplementary material to validate the practicality of our system in a physical environment. We deploy our framework on a Unitree Go2 quadruped robot platform equipped with onboard sensors. For state estimation, we adopt Point-LIO (He et al., 2023) to obtain accurate real-time pose estimation from LiDAR measurements. For motion planning and exploration, we integrate FAR Planner (Yang et al., 2022) as the path planning module to generate collision-free navigation trajectories.

For visual reasoning, we invoke GPT-4o (Hurst et al., 2024) as the VLM through the OpenAI API to interpret task instructions and guide the agent's decision making. All other components of our sys-

Frame captioning prompt

```
Write a dense, factual caption for downstream text-only reasoning.
Output exactly ONE paragraph and begin directly with concrete nouns
(do NOT start with phrases such as 'The image shows', 'This image
depicts', or 'There is/are'). Include: the room or setting; the
floor, only if visible (e.g., tile, wood, carpet, and any visible
rugs, cords, clutter, spills, or wet areas); as many visible
objects as possible, with counts when obvious; explicit spatial and
interaction relations (left/right, above/below, in front of/behind,
next to, on/under, inside, blocking, doorway open/closed); and any
visible action-relevant constraints (e.g., tight spaces, thresholds or
steps, water near electronics). Do not infer hidden areas or include
details that are not visible.
```

Figure 6: **Frame captioning prompt.** The prompt sent to the VLM to generate a dense, factual caption for each observed frame. The model is instructed to produce a single paragraph beginning with concrete nouns, covering room context, floor conditions, object inventory with counts, explicit spatial relations, and action-relevant constraints, without hallucinating unobserved content.

tem, including the 3DGS mapping, scene graph update, object-level retrieval rendering and hybrid exploration, are executed locally on a workstation equipped with a single RTX 4090 GPU.

A.10 Limitations

Our method requires more GPU memory than keyframe-based methods (Yang et al., 2025b) or graph-based methods (Gu et al., 2024), as it maintains an explicit 3D Gaussian representation as persistent spatial memory.

GSMem currently targets static or predominantly static environments, consistent with the setting of the previous methods (e.g. 3D-Mem, MTU360, etc.) as well as standard benchmarks such as HM3D and GOAT-Bench, where scene geometry and object configurations remain fixed within an episode. This differs from manipulation settings, where persistent scene changes are intrinsic. Accordingly, the current system does not explicitly detect moving objects or remove stale geometry, which we acknowledge as a limitation.

This limitation is not fundamental to the GSMem framework. Recent continual 3DGS methods (Zeng et al., 2025) support change detection, stale-content removal, and localized Gaussian updates in evolving scenes, while 4DGS methods (Yuan et al., 2026) can model dynamic objects. These approaches could replace the current mapping backend, with GSMem's multi-level retrieval and free-viewpoint recollection modules remaining largely unchanged. We will clarify this limitation and discuss integrating continual or dynamic Gaussian mapping as a natural extension toward evolving and manipulation-centric environments.

A.11 Frame Captioning Setting

In section 4.1, we employ frame captioning to evaluate LLM performance in a plain-text reasoning setting. Specifically, we use LLaVA-v1.5-13B to generate a caption for each frame based on a predefined prompt, as shown in fig. 6. These frame-level captions are then concatenated and provided as textual input to the LLM for downstream reasoning, following the setting introduced in (Majumdar et al., 2024).

A.12 Prompt Design

Our VLM-based navigation pipeline relies on three structured prompts, each targeting a distinct stage of the decision process. The full prompt templates are provided in fig. 7, fig. 8 and fig. 9-fig. 10.

Prompt 1: Task-relevant Object Category Generation. When the task instruction is received, we first prompt the VLM to enumerate a set of object categories that are semantically relevant to

System prompt

```
You are an expert at identifying objects in indoor environments.
Given a description or image of a target object, list the most likely
concrete object class names that a robot should look for using an
object detector.  Return ONLY a comma-separated list of object class
names (nouns), with no extra explanation.  For example:  'chair,
armchair, desk chair'.  Keep each name short (1-3 words).  List 1-5
most relevant classes.
```

Input prompt (description goal)

```
I need to find an object described as:  '{description}'.  The object
category is '{goal_category}'.  What concrete object class names should
I look for with an object detector?  Include '{goal_category}' and any
closely related or synonymous object names.
```

Input prompt (image goal)

```
I need to find the exact object shown in the center of the following
image.  The object category is '{goal_category}'.  What concrete
object class names should I look for with an object detector?  Include
'{goal_category}' and any closely related or synonymous object names.
This is the image of the target object:  [target_image]
```

Figure 7: **Target class inference prompt.** The prompt sent to the VLM to infer concrete object class names for the detector. The system prompt constrains the output to a short comma-separated list of 1–5 class names. Two input variants are shown: the *description goal* variant provides a textual description of the target, while the *image goal* variant additionally supplies the target object image. In both cases, {goal_category} is always included in the output. For *object goal* tasks, the goal category is used directly without querying the VLM.

the query. These categories are then used to query the CLIP field, allowing the system to retrieve potentially relevant regions from the 3D memory.

Prompt 2: Object-level Prefiltering for Retrieval Rendering. At each exploration step, we provide the VLM with the list of objects detected in the current scene graph and ask it to select the objects that are most relevant to the task. Based on the selected objects, the system retrieves the corresponding regions from the 3DGS memory and determines the optimal viewpoints for object renderings.

Prompt 3: Exploration Decision Making. After rendering candidate object views, the rendered object observations together with frontier observations are provided to the VLM. The VLM determines whether the currently available information is sufficient to answer the task. If the information is insufficient, it selects the most promising frontier for further exploration.

System prompt

```
You are an AI agent in a 3D indoor scene.
```

Input 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, each object one
line
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} [image_goal]
Following is a list of objects that you can choose, each object one
line
{class_list, one per line}
Answer:
```

Figure 8: **Object pre-filtering prompt.** The prompt sent to the VLM to rank and select task-relevant objects before exploration. The model receives a navigation question (with an optional goal image), a complete list of detected object classes in the scene, and a one-shot example. It outputs a ranked subset of objects most likely to assist exploration. Placeholders: {question} is the navigation query, [image_goal] is the optional goal image, and {class_list} is the full set of detected object class names.

System prompt

```
You are an agent in an indoor scene that is able to observe the
surroundings and explore the environment.  You are tasked with indoor
navigation, and you are required to choose either a Snapshot or a
Frontier image to explore and find the target object required in the
question.
```

Input prompt

```
Definitions:
Snapshot:  A rendered observation of a detected object from its best
observation angle.  Choosing a snapshot means you believe the target
object is visible in this observation and you want to navigate to it.
Frontier:  An unexplored region that could potentially lead to new
information for finding the target object.  Selecting a frontier means
that you will further explore that direction.
Question:  {question} [image_goal]
Select the Frontier/Snapshot that would help find the answer of the
question.
IMPORTANT: Do NOT choose a Snapshot solely because the objects around
it match the description of the target object's surroundings.  Similar
environments (e.g., multiple desks with monitors, multiple beds with
nightstands) may exist in the scene.  Only choose a Snapshot when you
are confident that the target object ITSELF is clearly visible in the
rendered image.  If you are uncertain, prefer exploring a Frontier to
gather more information.
The following is the egocentric view of the agent in forward
direction: [egocentric_image]
The followings are all the snapshots that you can choose.  Each
snapshot shows a detected object from two viewpoints:  the
optimally-selected observation angle and the highest-confidence
detection angle.  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.
Snapshot 0 ({class_label}) - optimal view [img] | detection view [img]
Snapshot 1 ({class_label}) - optimal view [img] | detection view [img]
...
```

Figure 9: **Exploration decision prompt (Part 1/2).** The system prompt and input prompt sent to the VLM at each navigation step, including role definition, snapshot/frontier definitions, the navigation question with optional goal image, decision constraint, egocentric view, and multimodal snapshot/frontier candidates. Placeholders: {question} is the navigation query, [image_goal] is the optional goal image, [egocentric_image] is the agent's current forward view, {class_label} is the predicted object class, and [img] denotes rendered views. Continued in Fig. 10.

Input prompt (continued)

```
The followings are all the Frontiers that you should score:
Frontier 0 [img]   Frontier 1 [img]   ...
Please provide your answer in one of the following two formats.
Do NOT include any preamble or label before your answer --- start
directly with 'Snapshot' or 'Frontier'.
If a Snapshot contains the target object, reply with:
Snapshot i
[Reason]
If no Snapshot contains the target object, score ALL frontiers:
Frontier 0:  <score>
Frontier 1:  <score>
...
Reason:  <brief explanation>
Each <score> must be a decimal between 0.0 and 1.0.  You MUST score
every frontier listed above.  A score of 1.0 means you are very
confident exploring this frontier will lead to the target object.
A score of 0.0 means the frontier is very unlikely to help find the
target.
Important notes:
(1) Only choose a Snapshot if you are confident the target object
ITSELF is clearly visible in it.
(2) When scoring frontiers, consider all visual cues (doors, hallways,
objects in view) to estimate the likelihood of finding the target.
(3) Your very first word MUST be either 'Snapshot' or 'Frontier'.  Do
not write anything else before it.
```

Figure 10: **Exploration decision prompt (Part 2/2).** Continued from Fig. 9. The constrained output format requires the model to either select a single snapshot (when the target object is clearly visible) or score all frontier candidates with continuous confidence values in $[0.0, 1.0]$.