LSM-VLM: Long-Short Memory VLM with World-Model for 3D Spatial Reasoning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent vision-language models (VLMs) have demonstrated strong perceptual and reasoning capabilities, but their ability to answer 3D spatial questions remains limited, especially in long-horizon reasoning and cross-view correspondence. Existing approaches partially address this challenge by introducing structured spatial memory or world-model-based view synthesis; however, they either fail to capture both global and local spatial details, or introduce hallucinated evidence. To address these challenges, we present LSM-VLM, a unified framework that connects perception, imagination, and reasoning through structured cognitive memory. Specifically, we construct a long-short memory module that integrates a scene graph and a bird's-eye-view (BEV) map, enabling the VLM to capture both object-level relational information and global spatial layouts. To recover missing spatial evidence, we leverage a world model to synthesize question-relevant unseen views conditioned on the current memory state. Since synthesized views may contain unreliable content, we further introduce a confidence-aware memory update mechanism that selectively incorporates trustworthy evidence into memory. Experiments show that LSM-VLM achieves state-of-the-art performance, reaching 69.2% on VSI-Bench and 63.6% on SQA3D, with consistent gains on long-horizon and cross-view reasoning tasks such as relative direction reasoning and route planning. [1]

1 Introduction

3D spatial understanding is a fundamental capability for embodied agents to perceive, navigate, and make decisions in real-world environments [34, 23, 25, 53]. Given an egocentric video and a question grounded in an embodied scene, an agent must reason not only about objects visible from the current camera view, but also about the scene layout, object relationships, occluded regions, and correspondences across different viewpoints. Such reasoning naturally calls for a *cognitive map*: an internal representation that organizes spatial structures, object locations, and semantic relations into a form suitable for reasoning and decision-making [36, 53, 18].

In recent years, vision-language models (VLMs) have demonstrated strong capabilities in open-vocabulary perception and visual question answering. However, their spatial reasoning is still largely driven by image or video tokens, which introduces several potential limitations. First, when the input video is long, such implicit representations often become insufficient for retaining all task-relevant evidence and become fragile for 3D spatial VQA [53, 63, 62, 50]. Second, for questions that require reasoning beyond the current view, such as questions about route planning or unobserved regions, implicit representations struggle to support effective scene understanding and reasoning [11, 12, 8, 59].

To mitigate long-context limitations, many methods attempt to maintain and update a compact, structured spatial memory. Some approaches use bird's-eye-view (BEV) maps as memory [39, 6, 26],

[1] An anonymized demo is available at `https://lsm-vlm.github.io/demo/`.

Submitted to 40th Conference on Neural Information Processing Systems (NeurIPS 2026). Do not distribute.

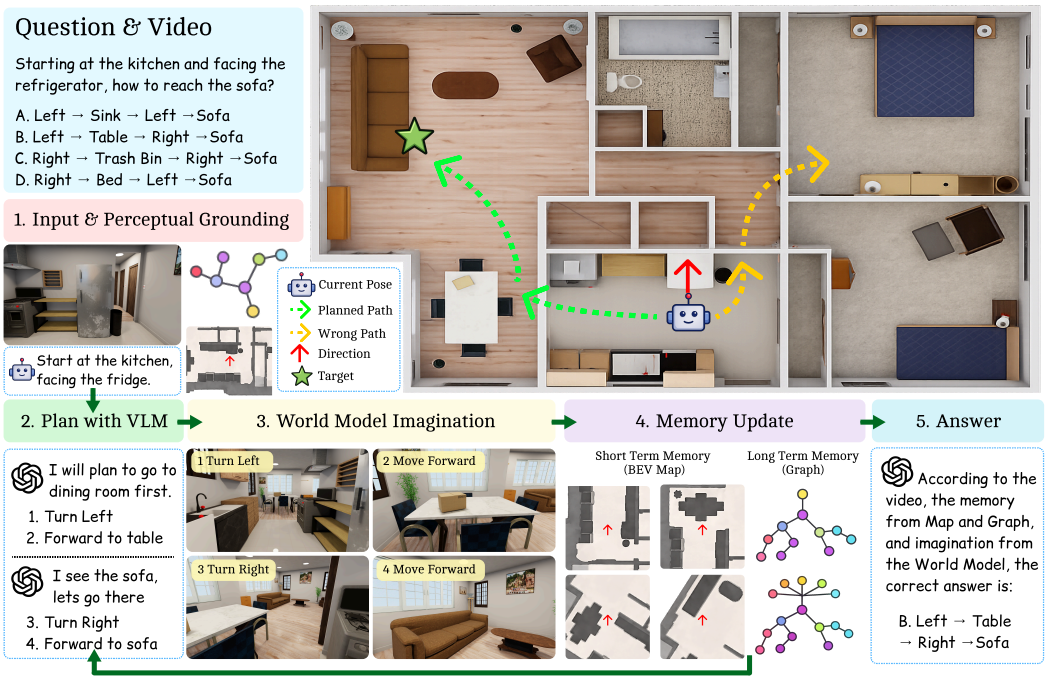


Figure 1: LSM-VLM. Given a video and a question, our framework first grounds the query in the observed scene, then predicts a question-relevant trajectory for exploration. A world model imagines observations along this trajectory, whose evidence is used to selectively update scene graph and BEV map. The enriched spatial memories are finally integrated into the VLM to produce the answer.

some use scene graphs [17, 58, 41], and others use latent or recurrent memory tokens [32, 56, 49]. However, large standalone BEV maps can be difficult for VLMs to interpret semantically, while graphs and latent memory tokens are often too abstract to preserve fine-grained local details. For the viewpoint-coverage limitation, world models offer a promising way to figure out incomplete visual evidence. By generating plausible observations from queried viewpoints, a world model provides a form of *imagination*, allowing an agent to inspect what it might see after changing pose [19, 20, 7, 38]. This capability is particularly useful for spatial VQA, where the answer often depends on evidence beyond the current frame. Recent camera-conditioned diffusion and novel-view synthesis models can provide such queried views [31, 40, 16, 47, 64], and test-time imagination methods have begun to use them for spatial reasoning [55, 27]. However, this strategy is computationally expensive, increases visual-context redundancy, and may introduce hallucinated objects or irrelevant details.

To address these challenges, we propose LSM-VLM, a unified framework for 3D spatial VQA that uses structured spatial memory as an interface between perception, imagination, and language reasoning. As shown in Fig. 1, the framework integrates perceptual alignment, dual cognitive memory, question-conditioned world-model imagination, and confidence-aware recurrent memory updating into a single closed loop. Given an input video, LSM-VLM progressively detects objects and constructs a dual cognitive memory, consisting of a long-term scene graph and a short-term BEV map with occupancy and semantic channels. The VLM uses this memory to identify informative target viewpoints and predict an action sequence to guide the world model. The world model then imagines the corresponding observations, providing additional visual evidence beyond the original trajectory. A lightweight perception module aligns these imagined observations and converts them into structured candidate updates, which are then integrated into memory through a recurrent update mechanism with LSTM-like gates [22]. Through this closed loop of planning, imagination, and memory updating, LSM-VLM enables the VLM to answer questions based on original observations, structured memory, and selectively integrated imagined evidence.

This design is well aligned with the requirements of modern spatial reasoning benchmarks. On VSI-Bench [53], questions typically require long-horizon evidence accumulation, cross-view correspondence, relative direction reasoning, and route planning. On SQA3D [34], models need to answer situated questions in reconstructed 3D scenes, which requires learning generalizable spatial representations. Built on the Qwen3-VL-8B backbone [4], LSM-VLM achieves 69.2% average accuracy

on VSI-Bench and 63.6% on SQA3D. Further ablation studies show that performance gains come from the synergy among dual cognitive memory, world-model imagination, and confidence-aware recurrent memory integration.

Our contributions are summarized as follows:

- We propose LSM-VLM, a closed-loop framework for 3D spatial VQA that connects perception, memory, imagination, and language reasoning through a structured spatial-memory interface, avoiding direct reasoning over long videos or raw imagined images.
- We design a dual cognitive memory with confidence-aware recurrent updating, which combines scene graphs for long-term semantic relations and local BEV maps for short-term spatial reasoning, and selectively integrates evidence imagined by the world model.
- Experiments on VSI-Bench and SQA3D show promising results, especially on long-horizon and cross-view reasoning tasks such as relative direction reasoning and route planning.

2 Related Work

2.1 VLMs for Spatial VQA

Spatial VQA has evolved from 3D-input benchmarks and 3D-aware models [3, 34, 65, 23] toward methods that inject 3D awareness into 2D VLMs through spatial QA synthesis, depth modules, or RGB-D conditioning [11, 12, 8]. Recent video-based studies have further exposed the difficulty of metric and configurational reasoning for modern MLLMs [53], leading to approaches based on position-aware tokens, implicit 3D priors, reconstruction-based tuning, and BEV representations [63, 62, 15, 39]. However, existing methods largely reason over the observed video alone, without actively completing missing evidence from unobserved viewpoints or maintaining a structured spatial memory, which limits their ability to handle layout, visibility, and cross-view reasoning.

2.2 World Models

World models learn predictive representations of environment dynamics [19, 20], with recent generative video models extending this idea to action- or pose-conditioned rollouts [7, 38, 2]. For spatial reasoning, a relevant branch is camera-controlled novel view synthesis, where diffusion models generate object- or scene-level views from specified camera poses [31, 40, 16, 21, 48]. We adopt Stable Virtual Camera [64], which produces 3D-consistent novel views from arbitrary input images and target cameras, as our goal-conditioned world model. Unlike recent methods that use imagined views or distilled world-model priors for spatial reasoning [55, 27, 60, 28], our approach performs question-conditioned, memory-grounded imagination and consolidates the synthesized evidence into a dual memory for subsequent reasoning.

2.3 Memory Mechanisms for Spatial Reasoning

Spatial reasoning benefits from memory mechanisms that aggregate evidence across time and viewpoints into a coherent representation. Prior work has explored cognitive maps for MLLMs [53, 18, 26, 42], structured 3D scene memories [17, 58, 41, 56], and long-horizon video memories such as working/episodic memory, memory folding, or BEV reconstruction [32, 49, 39]. However, most methods commit to a single memory modality and do not explicitly address how to absorb noisy evidence from generative world models without error accumulation. Our approach instead maintains a persistent dual memory that selectively consolidates simulated spatial evidence for robust cross-view reasoning.

3 Method

3.1 Overview

We propose LSM-VLM, a unified framework that integrates dual structured explicit spatial memory, question-conditioned world model imagination, and vision-language reasoning. Given a video sequence $\mathcal{V} = \{I_1, \ldots, I_T\}$ and a question q, the goal is to output an answer a that may depend on scene layout, relative direction, object visibility, and cross-view spatial relations. An overview of our

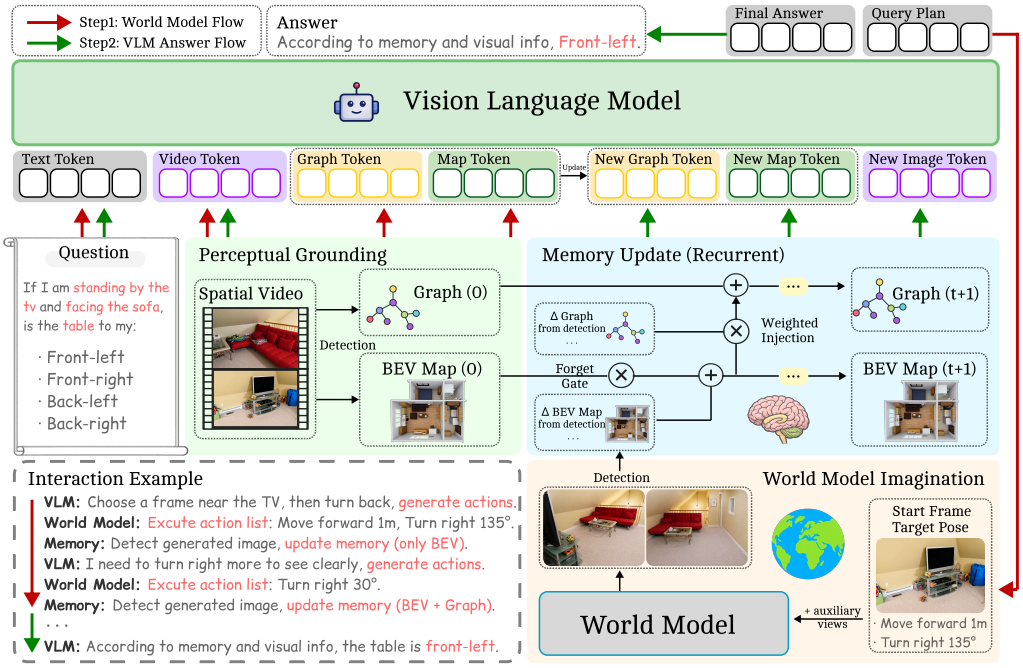


Figure 2: Overview of LSM-VLM: Given a question and an egocentric video, the *Perceptual Grounding* stage initializes a dual cognitive memory consisting of a scene graph G_0 and a BEV map B_0. The VLM then generates a target pose with an action list, which the *World Model* uses to imagine a new observation. A detection module converts the imagined view into candidate updates (ΔG, ΔB), which are integrated into memory through a forget gate and weighted injection. This updated information will be fed back to the VLM. The loop iterates until the VLM emits the final answer.

method is shown in Figure 2. The overall framework consists of two alternating closed loops: the World Model Flow for finding spatial evidence and updating memory, and the VLM Answer Flow for reading memory and answering. Our framework consists of three stages: *Perceptual Grounding*, *World Model Imagination*, and *Memory Update*, as illustrated in Figure 2. In *Perceptual Grounding*, we extract relevant information and build a dual cognitive map (3.2). VLM then processes these inputs and gives instructions to the world model. In *World Model Imagination*, the world model synthesizes additional observations, converts them into a candidate graph, and performs BEV updates. (3.3). In *Memory Update*, candidate updates are selectively integrated into the current memory via a forget gate and weighted injection, yielding the updated representation for the VLM. (3.4). This imagination-and-update loop can be repeated for multiple rounds until the VLM produces the final answer. To better align the VLM with the scene graph and BEV map, we further adopt a two-stage fine-tuning strategy. The corresponding training details are provided in the experiments section.

3.2 Perceptual Grounding with Dual Cognitive Memory

Structured spatial memory. To support long-horizon spatial reasoning, we equip the VLM with an effective memory mechanism. Existing methods typically rely on a single intermediate memory, such as a BEV map [39, 6, 26], a scene graph [17, 58, 41], or latent memory tokens [32, 56, 49], each of which only captures a subset of information needed for spatial reasoning. Graphs capture global semantic relations but are often too coarse for fine-grained spatial reasoning, whereas BEV maps encode geometry effectively but lack rich object semantics; latent memories are flexible but tend to be less interpretable and empirically less stable. Motivated by these observations, LSM-VLM maintains dual cognitive memory by using a scene graph as long-term semantic memory and a BEV map as short-term spatial memory. After reconstructing with VGGT [43] and semantically parsing the input video with SAM3 [10], the system initializes a dual cognitive map $\mathcal{M}_0 = \{G_0, B_0\}$, where G_0 denotes the graph and B_0 denotes the BEV map, and the subscript 0 denotes the memory update round in pipeline.

BEV map. VLMs often struggle to perform stable and accurate spatial reasoning over global BEV representations. Therefore, we intentionally keep the BEV representation as simple as possible. As shown in 2, the BEV map always uses a local coordinate frame centered on the agent, with the agent orientation aligned upward. The BEV map is defined as a two-channel local grid $B_0 \in \mathbb{R}^{H\times W\times 2}$, where $B_0^{\text{occ}} \in \{0,1\}^{H\times W}$ encodes occupancy and $B_0^{\text{sem}} \in \{1,\dots,C\}^{H\times W}$ stores semantic labels as category indices under the NYU40 [35] taxonomy. This localized design avoids the redundancy and reasoning burden introduced by a global map while preserving direction-sensitive geometric structure. In this sense, the BEV map mainly serves as a short-term, local, and viewpoint-sensitive geometric memory.

Scene graph. The scene graph G_0 is stored as a structured JSON memory and represents global object instances, the agent state, and their spatial relations. We represent the scene graph as $G_0 = (\mathcal{O}_0, \mathcal{R}_0)$, where $\mathcal{O}_0 = \{o_i\}_{i=1}^{N_0}$ denotes object nodes and $\mathcal{R}_0 = \{r_{ij}\}$ denotes spatial relations between nodes as defined in 3DSSG [51]. Each object node is represented as $o_i = (c_i, \mathbf{p}_i^w, \mathbf{e}_i^w, s_i, \mathbf{z}_i)$, where c_i is the object name, $\mathbf{p}_i^w = (x_i, y_i)$ is the object center in the world frame, $\mathbf{e}_i^w = (w_i, h_i, \psi_i)$ denotes its oriented spatial extent, s_i is the confidence score, and $\mathbf{z}_i$ is the semantic encoding under the NYU40 taxonomy. This representation provides a compact abstraction of the scene structure and facilitates high-level relational reasoning across long temporal horizons. Moreover, the shared semantic taxonomy and explicit coordinate attributes align the scene graph with the BEV map, forming a unified spatial memory for the VLM.

3.3 World Model for Spatial Evidence Completion

Target pose prediction. LSM-VLM augments perception with a question-conditioned world imagination module that actively completes missing spatial evidence. At reasoning round t, the VLM reads the question q, the original observed video $\mathcal{V}$, the previously imagined observations $\tilde{\mathcal{I}}_{<t}$, where the tilde notation denotes imagined content, and the current dual memory $\mathcal{M}_t = \{G_t, B_t\}$, and predicts a set of informative camera poses together with a target pose:

$$(\mathcal{P}_t, p_t^\star) = \text{VLM}_\theta^{\text{pose}}(q, \mathcal{V}, \tilde{\mathcal{I}}_{<t}, \mathcal{M}_t), \qquad \mathcal{P}_t = \{p_t^0, p_t^1, p_t^2, p_t^3\}.$$

Here, each candidate pose $p_t^j = (x_t^j, y_t^j, \phi_t^j)$ denotes an informative camera pose in the local BEV coordinate frame. The primary pose p_t^0 is used as the start pose of the imagined trajectory, while the remaining poses $\{p_t^1, p_t^2, p_t^3\}$ provide auxiliary spatial context, such as nearby or complementary viewpoints. The target pose $p_t^\star = (x_t^\star, y_t^\star, \phi_t^\star)$, also predicted by the VLM, specifies the endpoint of the trajectory and indicates where the imagination module should navigate to acquire question-relevant missing evidence. Given the current BEV map B_t, the start pose p_t^0, and the VLM-predicted target pose $p_t^\star$, we derive a discrete action list

$$\mathcal{A}_t = \{a_t^1, \dots, a_t^{L_t}\} = \text{Plan}(B_t, p_t^{\text{cur}}, p_t^\star),$$

where each action a_t^ℓ is selected from the navigation action space, and L_t denotes the number of planned steps required. $\text{Plan}(\cdot)$ computes a shortest feasible path on the BEV map and converts it into a discrete action sequence. The action list guides the world model to synthesize observations along the planned trajectory toward the target viewpoint.

World model imagination and detection. Conditioned on the observations associated with the queried viewpoints and the action list, the world model acts as a goal-conditioned simulator that synthesizes a short rollout toward the target pose:

$$\tilde{\mathcal{V}}_t = \mathcal{W}_\psi(\mathcal{I}_{\mathcal{P}_t}, \mathcal{A}_t),$$

where $\mathcal{I}_{\mathcal{P}_t} = \{I(p_t^0), I(p_t^1), I(p_t^2), I(p_t^3)\}$ denotes the observations corresponding to the queried poses $\mathcal{P}_t$, and $\tilde{\mathcal{V}}_t = \{\tilde{I}_t^1, \dots, \tilde{I}_t^L\}$ denotes the imagined visual observations along the planned trajectory. We then apply the same perception module used in perceptual grounding to the synthesized rollout. Specifically, VGGT and SAM3 extract object masks, geometry, camera-aware spatial cues, and semantic labels from $\tilde{\mathcal{V}}_t$. We denote this perception-and-reconstruction pipeline as Φ_{det}, where VGGT provides camera-aware geometric reconstruction and spatial cues, while SAM3 produces object-level masks and semantic detections. These predictions are then converted into candidate updates for the dual cognitive memory:

$$(\Delta B_t, \Delta G_t, \rho_t) = \Phi_{\text{det}}(\tilde{\mathcal{V}}_t),$$

where ΔB_t and ΔG_t are the proposed updates to the BEV memory and graph memory, respectively, and ρ_t denotes the confidence or reliability of the detected-and-reconstructed evidence. These candidate updates are not directly inserted into memory; They are passed to the confidence-aware memory consolidation module described below.

3.4 Confidence-Aware Memory Consolidation and Cross-View Reasoning

Although world-model generated observations provide useful complementary evidence, they may also introduce synthesis artifacts. Therefore, we draw inspiration from the gating mechanism in LSTM and design a dual confidence-aware memory consolidation module.

Memory update. Given the current memory $\mathcal{M}_t = \{G_t, B_t\}$ and candidate updates $(\Delta G_t, \Delta B_t, \rho_t)$ extracted from the imagined rollout, we update the BEV map and the scene graph asynchronously. For the BEV map, the forget operation is applied only to the previous BEV memory. It is independent of the current candidate update ΔB_t. Specifically, we update the BEV map as

$$B_{t+1} = f_B(B_t, \rho_{t-1}) + i_B(\Delta B_t, \rho_t),$$

where ρ_{t-1} denotes the historical confidence associated with the previous BEV memory B_t. The function $f_B(\cdot)$ selectively preserves reliable cells and forgets low-confidence cells from the last BEV map. The current update ΔB_t is then confidence-weighted by $i_B(\cdot)$ before being injected into the retained BEV memory. Before fusion, candidate BEV updates are transformed into the current local BEV frame using the estimated relative pose between the imagined camera and the memory frame.

For the scene graph, since it serves as long-term semantic memory, candidate graph updates are not directly committed. We compute a graph input gate only for candidates that have been confirmed by the BEV memory:

$$G_{t+1} = G_t + i_G(G_t, B_{t+1}, \Delta G_t, \rho_t).$$

Here, B_{t+1} is not a single-step observation but the consolidated BEV memory after recurrent updates. Thus, using B_{t+1} allows the weighted injection function $i_G(\cdot)$ to verify candidate graph updates against accumulated multi-step spatial evidence before committing them to the scene graph.

Cross-view reasoning. After memory consolidation, the VLM predicts the final answer by jointly attending to the original observation, imagined evidence, and the updated dual memory:

$$a = \mathrm{VLM}_\theta^{\mathrm{ans}}(q, \mathcal{V}, \tilde{\mathcal{I}}_{\le t}, \mathcal{M}_{t+1}).$$

This design allows LSM-VLM to use imagined observations for spatial evidence completion while reducing the risk of contaminating long-term memory with hallucinated content. As a result, the model can perform robust cross-view spatial reasoning under partial observation.

4 Experiments

4.1 Implementation Details

Two-Stage Training Strategy. We build our framework on Qwen3-VL-8B-Instruct and adopt Stable Virtual Camera as the world model for view generation. To incorporate BEV maps and graph, we employ a two-stage training strategy trained on ScanNet [13] and ScanNet++ [57] training splits.

In Stage 1, we introduce a map encoder and a graph encoder, together with their corresponding projectors, to align the newly introduced modalities with the pretrained language space. The map encoder is initialized from ViT-B/16 [14], while the graph encoder is initialized from MiniLM-L6-v2 [46]. During this stage, the original vision tower, language model, and multimodal merger are frozen, and only the newly introduced encoders and projectors are optimized.

In Stage 2, we continue training from the Stage 1 checkpoint and perform task-level adaptation with LoRA [24]. The map and graph encoders, together with their projectors, remain trainable, while LoRA is applied to the selected language modeling modules. The visual backbone and token embedding layers remain frozen for stable and efficient adaptation. More details are in Appendix B.

Table 1: Main results on VSI-Bench and SQA3D. All values are accuracies (%) for VSI-Bench and Avg. EM-1 for SQA3D. Higher is better, and - means unavailable.

Method	Avg.	Obj. Cnt.	Abs. Dist.	Obj. Size	Room Size	Rel. Dist.	Rel. Dir.	Route Plan	Appr. Order	SQA3D
Proprietary API Models										
GPT-4o [37, 39]	34.0	46.2	5.3	43.8	38.2	37.0	41.3	31.5	28.5	42.0
Claude-3.7-Sonnet [1, 45]	47.0	–	–	–	–	–	–	–	–	–
Gemini-2.5 Pro [59]	52.7	48.2	35.6	71.3	51.7	58.9	42.4	46.8	66.7	–
Open-source General VLMs										
Qwen3-VL-4B [4, 29]	59.3	–	–	–	–	–	–	–	–	49.9
Qwen3-VL-8B [9, 29]	57.9	67.6	47.0	76.3	61.9	58.0	51.0	35.1	66.3	50.2
Qwen3-VL-30B-A3B [44]	59.5	71.5	42.5	**77.4**	65.2	54.9	58.8	41.8	64.4	–
LLaVA-OneVision-72B [53]	40.2	43.5	23.9	57.6	37.5	42.5	39.9	32.5	44.6	–
LLaVA-NeXT-Video-72B [62]	40.9	48.9	22.8	57.4	35.3	42.4	36.7	35.0	48.6	–
InternVL3-8B [9]	42.1	66.1	34.9	43.6	47.5	48.0	39.3	26.3	31.4	–
InternVL3.5-8B [44]	49.4	61.3	34.4	61.2	55.4	50.4	44.2	34.0	54.4	–
Spatial-Enhanced Models										
VG-LLM-4B [62]	47.3	66.0	37.8	55.2	59.2	44.6	45.6	33.5	36.4	–
VG-LLM-8B [62]	50.7	67.9	37.7	58.6	62.0	46.6	40.7	32.4	59.2	–
Spatial-MLLM-4B [50]	48.4	65.3	34.8	63.1	45.1	41.3	46.2	33.5	46.3	55.9
VLM-3R-7B [15]	60.9	70.2	49.4	69.2	67.1	65.4	80.5	45.4	40.1	60.7
VST-7B [54]	61.2	71.6	43.8	75.5	69.2	60.0	55.6	44.3	69.2	–
VLM^2-7B [33]	68.8	72.5	**59.6**	70.8	69.9	**69.0**	**87.8**	52.6	68.3	60.4
LSM-VLM w/o Memory	67.1	72.8	52.3	75.6	68.8	64.2	74.1	51.0	**78.3**	48.9
LSM-VLM (Ours)	**69.2**	**73.1**	53.8	75.8	**70.0**	65.2	82.8	**54.9**	**78.3**	**63.6**

4.2 General Spatial Reasoning Benchmark

Benchmark & Metrics We evaluate general spatial reasoning on two complementary benchmarks, **VSI-Bench** [53] and **SQA3D** [34]. VSI-Bench is a video-based benchmark designed for long-horizon visual-spatial intelligence from egocentric indoor trajectories. It contains over 5K questions derived from ScanNet, ScanNet++ and ARKitScenes [5], and covers eight sub-tasks spanning measurement estimation and spatiotemporal reasoning, For multiple-choice or categorical questions, performance is measured by answer accuracy, while numerical estimation tasks are evaluated with the mean relative accuracy metric; the final *Avg.* score is the average over all eight sub-tasks.

We also report results on SQA3D [34], a standard 3D situated question answering benchmark built on ScanNet scenes. We follow the common SQA3D setting and use the average exact-match score on the test split. Overall, VSI-Bench evaluates long-horizon spatial reasoning from video observations, while SQA3D tests whether the learned spatial representations generalize to standard 3D QA.

Baselines We compare LSM-VLM against three groups of baselines in Table 1. The first group contains **proprietary API models**, including GPT-4o [37], Claude-3.7-Sonnet [1], and Gemini-2.5 Pro, which represent strong closed-source multimodal systems with large-scale video and image understanding capabilities. The second group contains **open-source general VLMs**, including the Qwen3-VL series [52], LLaVA-OneVision [30], LLaVA-NeXT-Video [61] and InternVL3/3.5 [45]. These models are not explicitly designed for spatial reasoning, but they provide strong and widely used general-purpose video-language baselines. The third group contains **spatial-enhanced models**, such as VG-LLM [62], Spatial-MLLM [50], VLM-3R [15], VST [54], and VLM^2 [33], which inject 3D priors, reconstruction signals or external memory mechanisms to improve spatial understanding. This comparison allows us to assess whether the proposed workflow of dual memory mechanism and world-model imagination remains competitive against recent methods.

Results Analysis Table 1 shows that LSM-VLM achieves the best overall performance on both benchmarks. On VSI-Bench, our full model reaches **69.2** average accuracy, outperforming the strongest previous result, VLM^2-7B, by **0.4** points and exceeding the strongest open-source general VLMs by a large margin. In particular, compared with the same Qwen3-VL family backbone, our method improves over Qwen3-VL-8B from 57.9 to 69.2 (**+11.3**), indicating that the gain does not come from model scale alone, but from explicitly introducing structured spatial representations. We also obtain the best result on SQA3D, reaching **63.6**, which is **+2.9** above VLM-3R and **+3.2** above VLM^2. This cross-benchmark improvement suggests that our approach learns transferable spatial reasoning abilities rather than overfitting to a single video benchmark.

Looking into the sub-task breakdown, the main advantage of LSM-VLM lies in **long-horizon relational and temporal reasoning**. Our model achieves the best performance on *route planning*

BEV	Graph	World Model	Mem.	Avg.	Obj Cnt	Abs Dist	Obj Size	Room Size	Rel Dist	Rel Dir	Route Plan	App. Order
×	×	×	×	61.5	68.8	51.4	76.3	66.2	63.8	55.4	36.1	74.3
✓	×	×	×	61.6	70.0	51.8	75.6	66.9	62.7	56.3	33.5	75.7
×	✓	×	×	61.4	70.1	51.7	75.8	66.5	62.5	55.5	32.9	75.8
✓	✓	×	×	64.3	72.8	52.3	75.5	68.7	64.7	64.7	37.1	**78.5**
×	×	✓	×	62.6	70.4	52.3	75.4	68.8	63.6	54.7	37.1	78.4
✓	✓	✓	×	67.1	72.8	52.3	75.6	68.8	64.2	74.1	51.0	78.3
✓	✓	✓	✓	**69.2**	**73.1**	**53.8**	**75.8**	**70.0**	**65.2**	**82.8**	**54.9**	78.3

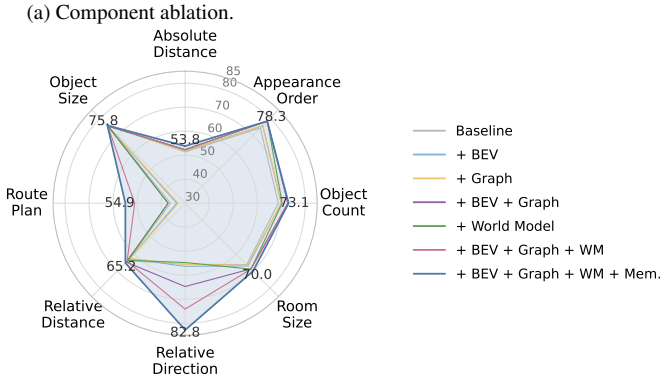
(a) Component ablation.

Stage-1	Stage-2	4B	8B
0	0	59.3	57.9
2K	0	57.7	56.2
2K	2K	59.1	59.8
2K	4K	60.9	61.3
2K	6K	59.2	61.5

(b) Training data scale.

BEV	Graph	8B
1	2	68.9
1	4	69.2
2	2	67.9
2	4	68.5

(c) Memory configuration.



(d) Qualitative visualization.

Figure 3: Ablation study and qualitative analysis on VSI-Bench. (a) Component ablation over BEV memory, graph memory, world model, and recursive memory update. (b) Training ablation under different Stage-1 and Stage-2 training steps. (c) Ablation of BEV and graph memory update intervals using the 8B model. (d) Qualitative visualization of the proposed memory representation.

(54.9) and *appearance order* (78.3). It also remains highly competitive on *object counting*, *room size*, *relative distance*, and *relative direction*, showing that the combination of BEV grounding, graph structure, and world-model completion helps the model maintain a globally coherent scene representation over time. At the same time, our method is not the best on every metric and we view this pattern as meaningful: our design brings the largest benefits to questions that require integrating evidence across viewpoints, preserving temporal order, and reasoning over scene structure, while fine-grained metric estimation remains a challenging direction for future improvement.

4.3 Ablation Studies

To better understand where the gains of LSM-VLM come from, we conduct ablation studies on VSI-Bench from four perspectives: component contribution, VLM training strategy, memory update intervals configuration, and computational efficiency. In particular, we examine the effect of different structured inputs and functional modules, analyze how the two-stage VLM adaptation process affects performance, compare different BEV and graph update settings, and finally quantify the runtime overhead introduced by world-model-based spatial evidence completion.

Component Ablation We first study the contribution of each major component in Figure 3(a). Starting from the fine-tuned VLM without any additional input, the model reaches 61.5 average accuracy on VSI-Bench, showing that fine-tuning itself does not simply bias the model toward the answer distribution. Adding only BEV memory or only graph memory brings almost no gain, whereas using the two structured memories together improves accuracy by 2.8 points over the VLM-only baseline, confirming that the two memories are complementary rather than redundant.

We then evaluate world-model imagination. Using the world model alone increases the average score from 61.5 to 62.6, suggesting that synthesized views already provide useful supplementary evidence even without explicit memory. However, the gain becomes much larger when imagination is grounded by structured memory: combining BEV, graph, and world model lifts the score to 67.1, with strong improvements on relative direction (74.1) and route planning (51.0). Finally, enabling recursive memory update yields the full model at 69.2, adding another 2.1 points and further improving relative distance, relative direction, and route planning. This result shows that imagined observations are most effective when converted into structured updates and consolidated back into persistent memory.

Training Strategy Ablation for VLM. Figure 3(b) analyzes the two-stage adaptation strategy for the VLM. Without any additional training, the 4B and 8B backbones achieve 59.3 and 57.9,

respectively. Applying only Stage 1 instead lowers the scores, since new tokens disrupt the original concatenation and interpretation. Once Stage 2 tuning is introduced, performance consistently recovers and then surpasses the original backbone. For the 8B model, extending Stage 2 to 6K steps gives the best result of 61.5. In contrast, the smaller 4B model saturates earlier and is more sensitive to overfitting. Overall, these results validate the necessity of the proposed two-stage training strategy.

Memory Update Ablation Because the BEV map and the scene graph play different short-term and long-term memory roles in our framework, their updates are not synchronized. Figure 3(c) compares four BEV&graph update interval configurations. Among them, the best result is obtained with the asymmetric setting of BEV=1 and Graph=4, which reaches 69.2. The overall pattern suggests that the two memories should not be treated identically. This finding supports our design choice of maintaining dual memory with asynchronous update behavior instead of collapsing both representations into a single shared memory process.

Table 2: Runtime breakdown by component.

Component	Time / Call	Call Times	Total Time
VLM (128 frames)	8.2s	3.4	27.88s
World Model	13.8s	1.7	23.46s
VGGT (1 frame)	0.1s	152.6	15.26s
SAM3 (1 frame)	0.1s	152.6	15.26s
Memory Update	0.1s	2.4	0.24s

Efficiency Analysis We finally analyze the computational overhead of each component in Table 2. When the target view is covered previously, we skip the world model and reuse the nearest observed view. The total inference time is about 82.1 seconds per sample on average. Among all modules, the VLM contributes the largest cumulative cost, requiring 8.2 seconds per call and 27.88 seconds in total because it is invoked 3.4 times on average. The world model is the most expensive single call at 13.8 seconds, but it is queried 1.7 times. VGGT and SAM3 are both lightweight on a per-call basis (0.1 second), but they are applied densely over many frames. The memory update module itself is lightweight. Importantly, the world model does not dominate the full runtime and it is affordable.

5 Conclusion

In this paper, we presented LSM-VLM, a closed-loop framework for 3D spatial visual question answering motivated by a central challenge raised in the introduction: spatial reasoning in embodied scenes cannot rely only on implicit video tokens, especially when evidence is distributed across long trajectories and missing viewpoints. Instead of asking the VLM to directly reason over long videos or raw generated images, LSM-VLM uses structured spatial memory as the interface between perception, imagination, and language reasoning. Specifically, it maintains a dual cognitive map with a scene graph for long-term semantic relations and a local BEV map for short-term geometric grounding, queries a question-conditioned world model to complete missing evidence, and selectively consolidates imagined observations through confidence-aware gated memory updates. This design enables the model to reason jointly over what exists in the scene, how entities are spatially arranged, and what may be observed from unvisited viewpoints.

Experiments on VSI-Bench and SQA3D show that this combination leads to strong and consistent gains, improving the Qwen3-VL-8B backbone from 57.9 to 69.2 on VSI-Bench and achieving 63.6 on SQA3D. Ablation studies further confirm that the improvements come from the synergy among dual memory, world-model imagination, and recurrent memory integration, with especially clear benefits on long-horizon and cross-view tasks such as relative direction reasoning, route planning, and appearance order.

Limitations and Future Work. Our current study is primarily built on the Qwen3-VL 4B and 8B backbones, and we have not yet systematically evaluated how well the proposed framework transfers to other VLM families. Extending the method to a broader range of backbones is therefore an important direction for future work. In particular, the overall pipeline is compatible with asynchronous execution, and future systems could further reduce latency by parallelizing parts of the VLM and world-model inference process. Finally, the current memory is used mainly for VLM-side reasoning and update, but it is not yet incorporated as an explicit condition for the world model itself. Allowing the world model to directly access structured memory may improve the quality, consistency, and task relevance of imagined views.

References

[1] Claude 3.7 sonnet system card. URL https://api.semanticscholar.org/CorpusID:276612236.

[2] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

[3] Daichi Azuma, Taiki Miyanishi, Shuhei Kurita, and Motoaki Kawanabe. Scanqa: 3d question answering for spatial scene understanding. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19107–19117, 2022. doi: 10.1109/CVPR52688.2022.01854.

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

[5] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data, 2022. URL https://arxiv.org/abs/2111.08897.

[6] Felix Brandstätter, Erik Schütz, Katharina Winter, and Fabian B. Flohr. Bev-llm: Leveraging multimodal bev maps for scene captioning in autonomous driving. In *2025 IEEE Intelligent Vehicles Symposium (IV)*, pages 345–350, 2025. doi: 10.1109/IV64158.2025.11097781.

[7] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. URL https://openai.com/research/video-generation-models-as-world-simulators.

[8] Wenxiao Cai, Iaroslav Ponomarenko, Jianhao Yuan, Xiaoqi Li, Wankou Yang, Hao Dong, and Bo Zhao. Spatialbot: Precise spatial understanding with vision language models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9490–9498, 2025. doi: 10.1109/ICRA55743.2025.11128671.

[9] Zhongang Cai, Ruisi Wang, Chenyang Gu, Fanyi Pu, Junxiang Xu, Yubo Wang, Wanqi Yin, Zhitao Yang, Chen Wei, Qingping Sun, Tongxi Zhou, Jiaqi Li, Hui En Pang, Oscar Qian, Yukun Wei, Zhiqian Lin, Xuanke Shi, Kewang Deng, Xiaoyang Han, Zukai Chen, Xiangyu Fan, Hanming Deng, Lewei Lu, Liang Pan, Bo Li, Ziwei Liu, Quan Wang, Dahua Lin, and Lei Yang. Scaling spatial intelligence with multimodal foundation models, 2026. URL https://arxiv.org/abs/2511.13719.

[10] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. Sam 3: Segment anything with concepts, 2026. URL https://arxiv.org/abs/2511.16719.

[11] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brain Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14455–14465, June 2024.

[12] An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. SpatialRGPT: Grounded spatial reasoning in vision-language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=JKEIYQUSUc.

[13] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes, 2017. URL https://arxiv.org/abs/1702.04405.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL https://arxiv.org/abs/2010.11929.

[15] Zhiwen Fan, Jian Zhang, Renjie Li, Junge Zhang, Runjin Chen, Hezhen Hu, Kevin Wang, Huaizhi Qu, Shijie Zhou, Dilin Wang, Zhicheng Yan, Hongyu Xu, Justin Theiss, Tianlong Chen, Jiachen Li, Zhengzhong Tu, Zhangyang Wang, and Rakesh Ranjan. Vlm-3r: Vision-language models augmented with instruction-aligned 3d reconstruction, 2026. URL https://arxiv.org/abs/2505.20279.

[16] Ruiqi Gao, Aleksander Hołyński, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T. Barron, and Ben Poole. Cat3d: create anything in 3d with multi-view diffusion models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

[17] Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, Chuang Gan, Celso Miguel de Melo, Joshua B. Tenenbaum, Antonio Torralba, Florian Shkurti, and Liam Paull. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5021–5028, 2024. doi: 10.1109/ICRA57147.2024.10610243.

[18] Chanyoung Gwak, Yoonwoo Jeong, Byungwoo Jeon, Hyunseok Lee, Jinwoo Shin, and Minsu Cho. Cog3dmap: Multi-view vision-language reasoning with 3d cognitive maps, 2026. URL https://arxiv.org/abs/2603.23023.

[19] David Ha and Jürgen Schmidhuber. World models. 2018. doi: 10.5281/ZENODO.1207631. URL https://zenodo.org/record/1207631.

[20] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2024. URL https://arxiv.org/abs/2301.04104.

[21] Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation, 2025. URL https://arxiv.org/abs/2404.02101.

[22] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9 (8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

[23] Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: injecting the 3d world into large language models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

[24] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

[25] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[26] Yibin Huang, Wang Xu, Wanyue Zhang, Helu Zhi, Jingjing Huang, Yangbin Xu, Yangang Sun, Conghui Zhu, and Tiejun Zhao. Video2layout: Recall and reconstruct metric-grounded cognitive map for spatial reasoning, 2026. URL https://arxiv.org/abs/2511.16160.

[27] Saurav Jha, M. Jehanzeb Mirza, Wei Lin, Shiqi Yang, and Sarath Chandar. Probing the effectiveness of world models for spatial reasoning through test-time scaling, 2025. URL https://arxiv.org/abs/2512.05809.

[28] Haoyi Jiang, Liu Liu, Xinjie Wang, Yonghao He, Wei Sui, Zhizhong Su, Wenyu Liu, and Xinggang Wang. Spa3r: Predictive spatial field modeling for 3d visual reasoning, 2026. URL https://arxiv.org/abs/2602.21186.

[29] Ruei-Chi Lai, Bolivar Enrique Solarte, Chin-Hsuan Wu, Yi-Hsuan Tsai, and Min Sun. Seeing once is enough? online geometry-aware token pruning for 3d question answering. In *The First Workshop on Efficient Spatial Reasoning*, 2026. URL https://openreview.net/forum?id=jnDbE6cV2D.

[30] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer, 2024. URL https://arxiv.org/abs/2408.03326.

[31] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9264–9275, 2023. doi: 10.1109/ICCV51070.2023.00853.

[32] Zuntao Liu, Yi Du, Taimeng Fu, Shaoshu Su, Cherie Ho, and Chen Wang. Vision-language memory for spatial reasoning, 2025. URL https://arxiv.org/abs/2511.20644.

[33] Zuntao Liu, Yi Du, Taimeng Fu, Shaoshu Su, Cherie Ho, and Chen Wang. Vision-language memory for spatial reasoning, 2025. URL https://arxiv.org/abs/2511.20644.

[34] Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes, 2023. URL https://arxiv.org/abs/2210.07474.

[35] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[36] J. O'Keefe and L. Nadel. *The Hippocampus as a Cognitive Map*. Clarendon Press, 1978. ISBN 9780198572060. URL https://books.google.com.hk/books?id=trR-AAAAIAAJ.

[37] OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, and et.al. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

[38] Jack Parker-Holder, Philip Ball, Jake Bruce, Vibhavari Dasagi, Kristian Holsheimer, Christos Kaplanis, Alexandre Moufarek, Guy Scully, Jeremy Shar, Jimmy Shi, Stephen Spencer, Jessica Yung, Michael Dennis, Sultan Kenjeyev, Shangbang Long, Vlad Mnih, Harris Chan, Maxime Gazeau, Bonnie Li, Fabio Pardo, Luyu Wang, Lei Zhang, Frederic Besse, Tim Harley, Anna Mitenkova, Jane Wang, Jeff Clune, Demis Hassabis, Raia Hadsell, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 2: A large-scale foundation world model. 2024. URL https://deepmind.google/discover/blog/genie-2-a-large-scale-foundation-world-model/.

[39] Zhangyang Qi, Zhixiong Zhang, Ye Fang, Jiaqi Wang, and Hengshuang Zhao. Gpt4scene: Understand 3d scenes from videos with vision-language models, 2025. URL https://arxiv.org/abs/2501.01428.

[40] Kyle Sargent, Zizhang Li, Tanmay Shah, Charles Herrmann, Hong-Xing Yu, Yunzhi Zhang, Eric Ryan Chan, Dmitry Lagun, Li Fei-Fei, Deqing Sun, and Jiajun Wu. Zeronvs: Zero-shot 360-degree view synthesis from a single image, 2024. URL https://arxiv.org/abs/2310.17994.

[41] 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, 2025. URL https://arxiv.org/abs/2412.14480.

[42] Jinzhou Tang, Jusheng zhang, Sidi Liu, Waikit Xiu, Qinhan Lv, and Xiying Li. Beyond pixels: Introducing geometric-semantic world priors for video-based embodied models via spatio-temporal alignment, 2025. URL https://arxiv.org/abs/2509.00210.

[43] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer, 2025. URL https://arxiv.org/abs/2503.11651.

[44] Pan Wang, Yang Liu, Guile Wu, Eduardo R. Corral-Soto, Chengjie Huang, Binbin Xu, Dongfeng Bai, Xu Yan, Yuan Ren, Xingxin Chen, Yizhe Wu, Tao Huang, Wenjun Wan, Xin Wu, Pei Zhou, Xuyang Dai, Kangbo Lv, Hongbo Zhang, Yosef Fried, Aixue Ye, Bailan Feng, Zhenyu Chen, Zhen Li, Yingcong Chen, Yiyi Liao, and Bingbing Liu. Spatial4d-bench: A versatile 4d spatial intelligence benchmark, 2026. URL https://arxiv.org/abs/2601.00092.

[45] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency, 2025. URL https://arxiv.org/abs/2508.18265.

[46] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020. URL https://arxiv.org/abs/2002.10957.

[47] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, SIGGRAPH '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400705250. doi: 10.1145/3641519.3657518. URL https://doi.org/10.1145/3641519.3657518.

[48] Daniel Watson, Saurabh Saxena, Lala Li, Andrea Tagliasacchi, and David J. Fleet. Controlling space and time with diffusion models, 2025. URL https://arxiv.org/abs/2407.07860.

[49] Yuxi Wei, Wei Huang, Qirui Chen, Lu Hou, and Xiaojuan Qi. See, remember, explore: A benchmark and baselines for streaming spatial reasoning, 2026. URL https://arxiv.org/abs/2603.23864.

[50] Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence, 2025. URL https://arxiv.org/abs/2505.23747.

[51] Shun-Cheng Wu, Keisuke Tateno, Nassir Navab, and Federico Tombari. Incremental 3d semantic scene graph prediction from rgb sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5064–5074, 2023.

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

[53] Jihan Yang, Shusheng Yang, Anjali W. Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces, 2025. URL https://arxiv.org/abs/2412.14171.

[54] Rui Yang, Ziyu Zhu, Yanwei Li, Jingjia Huang, Shen Yan, Siyuan Zhou, Zhe Liu, Xiangtai Li, Shuangye Li, Wenqian Wang, Yi Lin, and Hengshuang Zhao. Visual spatial tuning, 2025. URL https://arxiv.org/abs/2511.05491.

[55] Yuncong Yang, Jiageng Liu, Zheyuan Zhang, Siyuan Zhou, Reuben Tan, Jianwei Yang, Yilun Du, and Chuang Gan. Mindjourney: Test-time scaling with world models for spatial reasoning, 2025. URL https://arxiv.org/abs/2507.12508.

[56] 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, 2025. URL https://arxiv.org/abs/2411.17735.

[57] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes, 2023. URL https://arxiv.org/abs/2308.11417.

[58] Tatiana Zemskova and Dmitry Yudin. 3dgraphllm: Combining semantic graphs and large language models for 3d scene understanding, 2025. URL https://arxiv.org/abs/2412.18450.

[59] Gongjie Zhang, Wenhao Li, Quanhao Qian, Jiuniu Wang, Deli Zhao, Shijian Lu, and Ran Xu. On the generalization capacities of mllms for spatial intelligence, 2026. URL https://arxiv.org/abs/2603.06704.

[60] Kevin Zhang, Kuangzhi Ge, Xiaowei Chi, Renrui Zhang, Shaojun Shi, Zhen Dong, Sirui Han, and Shanghang Zhang. Can world models benefit vlms for world dynamics?, 2025. URL https://arxiv.org/abs/2510.00855.

[61] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Llava-video: Video instruction tuning with synthetic data, 2025. URL https://arxiv.org/abs/2410.02713.

[62] Duo Zheng, Shijia Huang, Yanyang Li, and Liwei Wang. Learning from videos for 3d world: Enhancing mllms with 3d vision geometry priors, 2025. URL https://arxiv.org/abs/2505.24625.

[63] Duo Zheng, Shijia Huang, and Liwei Wang. Video-3d llm: Learning position-aware video representation for 3d scene understanding. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8995–9006, 2025. doi: 10.1109/CVPR52734.2025.00841.

[64] Jensen Zhou, Hang Gao, Vikram Voleti, Aaryaman Vasishta, Chun-Han Yao, Mark Boss, Philip Torr, Christian Rupprecht, and Varun Jampani. Stable virtual camera: Generative view synthesis with diffusion models, 2025. URL https://arxiv.org/abs/2503.14489.

[65] Ziyu Zhu, Xiaojian Ma, Yixin Chen, Zhidong Deng, Siyuan Huang, and Qing Li. 3d-vista: Pre-trained transformer for 3d vision and text alignment. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2899–2909, 2023. doi: 10.1109/ICCV51070.2023.00272.

Appendix

A Pseudocode For Inference

We summarize the inference procedure of LSM-VLM in Algorithms 1–3. Algorithm 1 gives the overall closed-loop inference process. Given an egocentric video $\mathcal{V}$ and a question q, the model first constructs an initial dual memory $\mathcal{M}_0 = \{G_0, B_0\}$, where G_0 is the scene graph and B_0 is the local BEV map. The detector $\Phi_{\text{det}}(\cdot)$ extracts visual evidence and returns the initial memory together with confidence ρ_0. At each round, the model imagines missing spatial evidence, updates the memory, and then feeds the last imagined frame $\tilde{I}_t^{\text{last}}$ together with the updated memory $\mathcal{M}_{t+1}$ to the VLM for answering. The loop stops when the VLM outputs a final answer signal or when the maximum number of rounds is reached. In our implementation, we set the maximum number of rounds as 4.

Algorithm 2 describes the world-model imagination step. At round t, the current memory is $\mathcal{M}_t = \{G_t, B_t\}$. The pose predictor $\text{VLM}_\theta^{\text{pose}}$ predicts informative viewpoints $\mathcal{P}_t = \{p_t^0, p_t^1, p_t^2, p_t^3\}$ based on the question, observed video, imagined history, and current memory. The planner $\text{Plan}(\cdot)$ then computes an action sequence $\mathcal{A}_t$ from the current pose p_t^{cur} to the target pose $p_t^\star$. The world model $\mathcal{W}_\psi$ generates an imagined rollout $\tilde{\mathcal{V}}_t$, which is parsed into candidate memory updates $(\Delta B_t, \Delta G_t, \rho_t)$.

Algorithm 3 shows how candidate updates are fused into memory. For the BEV map, $f_B(B_t, \rho_{t-1})$ keeps reliable previous cells, while $i_B(\Delta B_t, \rho_t)$ injects new confidence-weighted BEV evidence. Since the newly generated BEV update ΔB_t and the previous BEV memory B_t are not necessarily defined in the same coordinate frame, we always use an agent-centric local frame, where the agent position is placed at the center and its heading direction is aligned upward. Before memory fusion, all spatial coordinates are transformed into this current local frame, including the local object coordinates stored in the scene graph. The updated BEV memory is denoted as B_{t+1}. For the scene graph, $i_G(G_t, B_{t+1}, \Delta G_t, \rho_t)$ verifies candidate graph updates using the consolidated BEV map before adding them to G_t. The final updated memory is $\mathcal{M}_{t+1} = \{G_{t+1}, B_{t+1}\}$.

Algorithm 1 Overall LSM-VLM Inference

Require: Egocentric video $\mathcal{V}$, question q, maximum rounds $T_{\max}$
Ensure: Final answer a

▷ **Perceptual Grounding**

Extract geometry, object masks, and semantic labels from $\mathcal{V}$ using VGGT and SAM3.

$$(B_0, G_0, \rho_0) \leftarrow \Phi_{\text{det}}(\mathcal{V}).$$

Build initial scene graph G_0 and local BEV map B_0.
Initialize dual memory $\mathcal{M}_0 \leftarrow \{G_0, B_0\}$.
Initialize imagined buffer $\tilde{\mathcal{I}}_{<0} \leftarrow \emptyset$ and confidence ρ_0.
for $t = 0$ **to** $T_{\max} - 1$ **do**

▷ **Evidence Completion Loop (World Model Flow)**

$(\tilde{\mathcal{V}}_t, \Delta B_t, \Delta G_t, \rho_t) \leftarrow \textsc{ImagineEvidence}(q, \mathcal{V}, \tilde{\mathcal{I}}_{<t}, \mathcal{M}_t)$.
Extract the last imagined frame & Update imagined evidence history:

$$\tilde{I}_t^{\text{last}} \leftarrow \text{LastFrame}(\tilde{\mathcal{V}}_t), \qquad \tilde{\mathcal{I}}_{<t+1} \leftarrow \tilde{\mathcal{I}}_{<t} \cup \{\tilde{I}_t^{\text{last}}\}.$$

$\mathcal{M}_{t+1} \leftarrow \textsc{UpdateMemory}(\mathcal{M}_t, \Delta B_t, \Delta G_t, \rho_{t-1}, \rho_t)$.

▷ **VLM Answer Flow**

Feed the last imagined frame and updated memory to the VLM:

$$(y_t, s_t) \leftarrow \text{VLM}_\theta^{\text{ans}}(q, \mathcal{V}, \tilde{I}_{<t+1}, \mathcal{M}_{t+1}).$$

if $s_t = \textsc{FinalAnswer}$ **or** $t = T_{\max} - 1$ **then**
return y_t.
end if
end for

Algorithm 2 IMAGINEEVIDENCE: Question-Conditioned World Imagination

Require: Question q, video $\mathcal{V}$, imagined buffer $\tilde{\mathcal{I}}_{<t}$, memory $\mathcal{M}_t = \{G_t, B_t\}$
Ensure: Imagined rollout $\tilde{\mathcal{V}}_t$, candidate updates $\Delta B_t, \Delta G_t$, confidence ρ_t

▷ **Target Pose Prediction**

Predict informative viewpoints:

$$\mathcal{P}_t = \{p_t^0, p_t^1, p_t^2, p_t^3\} \leftarrow \text{VLM}_\theta^{\text{pose}}(q, \mathcal{V}, \tilde{\mathcal{I}}_{<t}, \mathcal{M}_t).$$

Select target pose $p_t^\star$ according to q and $\mathcal{M}_t$.
Plan actions on the current BEV map:

$$\mathcal{A}_t \leftarrow \text{Plan}(B_t, p_t^{\text{cur}}, p_t^\star).$$

▷ **World Model Rollout**

Retrieve observations at queried viewpoints:

$$\mathcal{I}_{\mathcal{P}_t} \leftarrow \{I(p_t^0), I(p_t^1), I(p_t^2), I(p_t^3)\}.$$

Generate imagined rollout:

$$\tilde{\mathcal{V}}_t \leftarrow \mathcal{W}_\psi(\mathcal{I}_{\mathcal{P}_t}, \mathcal{A}_t).$$

▷ **Candidate Evidence Extraction**

Apply VGGT and SAM3 to $\tilde{\mathcal{V}}_t$.
Convert detections into candidate memory updates:

$$(\Delta B_t, \Delta G_t, \rho_t) \leftarrow \Phi_{\text{det}}(\tilde{\mathcal{V}}_t).$$

Transform ΔB_t into the current local BEV frame using the estimated relative pose.
return $(\tilde{\mathcal{V}}_t, \Delta B_t, \Delta G_t, \rho_t)$.

Algorithm 3 UPDATEMEMORY: Confidence-Aware Memory Consolidation

Require: Current memory $\mathcal{M}_t = \{G_t, B_t\}$, candidate updates $\Delta B_t, \Delta G_t$, historical confidence ρ_{t-1}, current confidence ρ_t
Ensure: Updated memory $\mathcal{M}_{t+1}$

▷ **Coordinate Alignment**

Align BEV cells and graph coordinates to the current agent-centric local frame:

$$(B_t^{\text{loc}}, G_t^{\text{loc}}, \Delta B_t^{\text{loc}}, \Delta G_t^{\text{loc}}) \leftarrow \text{AlignToAgentFrame}(B_t, G_t, \Delta B_t, \Delta G_t).$$

▷ **BEV Memory Consolidation**

Preserve reliable cells and inject new confidence-weighted BEV evidence:

$$B_t^{\text{keep}} \leftarrow f_B(B_t^{\text{loc}}, \rho_{t-1}), \qquad B_t^{\text{new}} \leftarrow i_B(\Delta B_t^{\text{loc}}, \rho_t).$$

Update BEV memory:

$$B_{t+1} \leftarrow B_t^{\text{keep}} + B_t^{\text{new}}.$$

▷ **Scene Graph Consolidation**

Verify candidate graph updates and update long-term semantic memory:

$$\Delta G_t^{\text{valid}} \leftarrow i_G(G_t^{\text{loc}}, B_{t+1}, \Delta G_t^{\text{loc}}, \rho_t), \qquad G_{t+1} \leftarrow G_t^{\text{loc}} + \Delta G_t^{\text{valid}}.$$

▷ **Dual Memory Output**

$\mathcal{M}_{t+1} \leftarrow \{G_{t+1}, B_{t+1}\}$.
return $\mathcal{M}_{t+1}$.

B Training Setup Details

Our training pipeline is implemented in a Hugging Face and DeepSpeed codebase built around `transformers`, `peft`, and `trl`, with custom modules under `src/train`, `src/dataset`, and `src/trainer`. The default backbone used in both stages is `Qwen/Qwen3-VL-8B-Instruct`. The implementation extends the standard Qwen-VL supervised fine-tuning pipeline by adding two extra modality branches, namely a *map branch* and a *graph branch*, and by monkey-patching the Qwen forward function so that these additional modality features can be injected into the language token stream during training.

Additional modality branches. The map branch uses `google/vit-base-patch16-224-in21k` as the map encoder, while the graph branch uses `sentence-transformers/all-MiniLM-L6-v2` as the graph encoder. Two learnable linear projectors are newly introduced to align the map and graph features to the hidden size of the Qwen language model. Concretely, the model adds `map_encoder`, `graph_encoder`, `map_projector`, and `graph_projector` on top of the original Qwen-VL architecture. Two new special tokens, `<|map_pad|>` and `<|graph_pad|>`, are added to the tokenizer vocabulary and are used as placeholders for the injected map and graph features.

Map representation. Each training sample may contain a video, a top-down occupancy map sequence, and a graph file. For maps, the code first collects the map frames associated with the sample. When training on video data, the map frames are aligned to the sampled video frame indices; in the default implementation, this is done by selecting files such as `occ_{frame_idx}.png` from the occupancy-map directory. The selected map frames are tiled into a single 2D canvas arranged on a near-square grid, and the resulting canvas is processed by the map image processor. When semantic supervision is enabled (`semantic=True` in both training scripts), an additional semantic tensor is loaded from the corresponding `.npz` files and concatenated channel-wise with the RGB map canvas. As a result, the input to the map encoder has 43 channels (3 RGB channels plus 40 semantic channels in the default setup). Since the pretrained ViT map encoder expects 3-channel input, its patch embedding convolution is explicitly adapted to 43 input channels by copying the original RGB weights and initializing the extra channels with the mean of the original filters.

Graph representation. For graph inputs, the code loads the graph file (typically `graph/graph.json`) as raw text and tokenizes it with the graph encoder tokenizer. Graph se-

quences are truncated or padded to a maximum length of 256 tokens. If a graph exists but the human prompt does not explicitly contain a graph placeholder, the loader automatically inserts <graph> into the prompt so that graph features are always aligned with an explicit placeholder position in the language sequence.

Feature injection into Qwen. The Qwen forward pass is modified to support mixed-modality inputs. During training, the visual encoder processes the image or video tokens as usual, while the new map and graph encoders produce separate hidden-state sequences. These hidden states are then compressed into a fixed number of placeholder-aligned tokens by chunk-wise average pooling after dropping the first encoder token. In the default configuration, each map input is compressed into 16 tokens and each graph input is compressed into 8 tokens. Accordingly, in the text preprocessing stage, <map> is expanded into 16 repeated <|map_pad|> tokens and <graph> is expanded into 8 repeated <|graph_pad|> tokens. After projection to the Qwen hidden dimension, the resulting map and graph features replace the embeddings at those placeholder positions, so the language model receives aligned map-aware and graph-aware token embeddings inside the normal autoregressive sequence.

Data formatting and supervision. The training data are stored in JSON format and contain conversation pairs together with multimodal references, e.g., `video`, `map`, and `graph`. A typical prompt contains <video>, <map>, and <graph> markers followed by the task instruction. The data loader converts the original LLaVA-style placeholders into Qwen-compatible tokens, assembles the multimodal tensors, and constructs next-token prediction targets in the standard supervised fine-tuning format. Only assistant responses contribute to the loss; all user-side prompt tokens are masked with `IGNORE_INDEX`. Since the default backbone in our scripts is Qwen3-VL, no extra system prompt is prepended by the data loader.

Optimization framework. Both stages use DeepSpeed ZeRO-2, `adamw_torch`, bfloat16 training, TF32 matmul, gradient checkpointing, cosine learning-rate decay, and a warmup ratio of 0.03. The scripts enable FlashAttention-2 (`disable_flash_attn2=False`) and disable Liger kernels (`use_liger_kernel=False`). Weight decay is 0.1. Training logs are written every step and checkpoints are saved every 2000 steps with a maximum of 5 retained checkpoints. The scripts use on-the-fly preprocessing (`lazy_preprocess=True`) with `dataloader_num_workers=0`. Images are constrained to a pixel budget between 512×28^2 and 1280×28^2, and videos are sampled with 128 frames by default.

Stage 1: modality-branch pretraining. Stage 1 is designed to learn the newly added map and graph components while keeping the original Qwen-VL backbone fixed. Specifically, LoRA is disabled, the Qwen language model is frozen, the original visual tower is frozen, and the visual merger is frozen. In contrast, both the map encoder and graph encoder are trainable, and both projectors are trainable. The default learning rates are 2×10^{-6} for the map encoder, 2×10^{-6} for the graph encoder, 1×10^{-5} for the map projector, and 1×10^{-5} for the graph projector. The generic learning rate argument is 5×10^{-5}, but in practice the trainable parameters in this stage are dominated by the modality-specific encoders and projectors because the base Qwen-VL modules remain frozen. The default script runs for 10 epochs on 8 GPUs with per-device batch size 1, giving a default global batch size of 8 and gradient accumulation of 1.

Stage 2: LoRA-based alignment. Stage 2 starts from a Stage-1 checkpoint (by default `output/4_09_stage1_map_graph_only/checkpoint-2000`) and performs a weights-only continuation rather than resuming the optimizer or scheduler states. LoRA is enabled with rank 16, scaling factor 32, and dropout 0.05. The code applies LoRA to all eligible linear or embedding modules except those explicitly excluded by name. By default, LoRA is excluded from the visual backbone, the map encoder, the graph encoder, the two projectors, the language-model embedding layer, and the LM head. Therefore, Stage 2 mainly aligns the language-model internals to the already learned map/graph representations while preserving the newly introduced modality branches as separate trainable components. The Qwen language model, vision tower, and visual merger remain frozen in the standard full-parameter sense, but LoRA adapters on the allowed language-model modules are trainable. In the provided script, both the map encoder and graph encoder also remain unfrozen by default (`FREEZE_MAP_ENCODER=False`, `FREEZE_GRAPH_ENCODER=False`), so Stage 2 jointly updates the LoRA adapters, the map encoder, the graph encoder, and both projectors. The

default script runs for 10 epochs on 8 GPUs with per-device batch size 1, corresponding to a default global batch size of 8 and gradient accumulation of 1.

Learning-rate grouping. The trainer uses explicit parameter groups with separate learning rates for the visual tower, merger, map encoder, graph encoder, map projector, and graph projector. In our two-stage setting, this design is important because it allows the newly introduced modality branches to be trained conservatively while LoRA parameters in the language model use the base learning rate. This decoupled optimization is one of the key implementation choices that stabilizes training after adding heterogeneous map and graph inputs to a large pretrained vision-language backbone.

Checkpointing and reproducibility details. When loading from a local checkpoint, the code first instantiates the backbone, then attaches the custom map/graph modules, and finally reloads the checkpoint weights with non-strict matching so that the extra modality parameters are correctly restored. If LoRA is enabled, checkpoints additionally save the non-LoRA trainable parameters separately, which is necessary because the map/graph modules are not part of the LoRA adapters themselves. The training code also logs trainable-parameter summaries and gradient norms for the map encoder, graph encoder, and both projectors, which we used as a sanity check to ensure that gradients propagated correctly through the newly added branches.

C Inference setup details

We implement a closed-loop inference pipeline that explicitly separates perceptual grounding, planning, future-view imagination, memory consolidation, and final answering. The lightweight unit test in `tests/test_pipeline.py` validates the contract of this pipeline, while the full implementation in `origin_code/inference` instantiates the heavy-weight vision-language model (VLM), map/graph-conditioned prompting, and Stable Virtual Camera (SVC) world-model generation.

Pipeline contract. The pipeline interface is defined by `LSMVLMPipeline` with configuration `PipelineConfig(rounds, bev_height, bev_width, bev_resolution, graph_update_interval, manual_actions)`. Given a question, a sequence of observations, and optional initial graph/BEV files, the pipeline executes the following stages:

1. load the initial cognitive memory from a scene graph and a BEV map;
2. ground the input observations into an initial graph and BEV update;
3. initialize the recurrent memory;
4. predict a target pose or action sequence conditioned on the current memory and the question;
5. imagine future observations along the planned trajectory using a world model;
6. detect candidate objects and BEV occupancy from the imagined views;
7. update the memory using a confidence-aware recurrent rule;
8. answer the question from the updated memory.

The unit test additionally verifies that the pipeline serializes intermediate memory states to disk (e.g., `memory_round_000.json`, `memory_round_001.json`, and `result.json`) and that action parsing and execution are deterministic.

Dual-memory representation. The recurrent memory is represented as a pair of structured states:

- a *scene graph* containing object nodes, relations, and the agent pose;
- a *BEV map* containing occupancy, semantic labels, confidence, origin, and metric resolution.

Each object node stores a node ID, category name, planar position, extent, confidence, semantic ID, and auxiliary attributes. The BEV memory stores three aligned grids: occupancy, semantic labels, and confidence. Coordinates are converted between world and grid frames using the map origin and resolution. This design matches the role split used throughout the codebase: the graph provides sparse symbolic structure, while the BEV map provides dense spatial support.

Confidence-aware recurrent update. Memory updates are consolidated by `ConfidenceAwareMemoryUpdater`. Let the previous BEV confidence be decayed by a factor of 0.92, and let cells whose decayed confidence falls below 0.25 be forgotten. New candidate occupancy and semantic evidence are injected wherever the new confidence exceeds the retained confidence. For graph updates, candidate nodes are added only if they are confirmed by the BEV map, i.e., the confidence at the node's projected BEV location exceeds the graph confirmation threshold (0.35 by default). Graph nodes with the same class name are merged when they fall within a Euclidean distance of 0.75 m. After each update, pairwise spatial relations such as `left_of`, `right_of`, `in_front_of`, and `behind` are recomputed for nearby objects.

Trajectory parameterization. Planned actions are represented as short symbolic commands such as `move-forward {meters}`, `turn-left {degrees}`, and `turn-right {degrees}`. The action parser accepts either a JSON array or a comma-separated textual list. Actions are deterministically applied to a planar pose (x, y, θ), and the resulting waypoints are converted to camera extrinsics for the world model. The unit test explicitly checks that parsing `["turn-left 90", "move-forward 1"]` from the default origin pose produces the expected translated pose.

C.1 Instantiation with the Trained VLM and World Model

The abstract pipeline above is instantiated in `origin_code/inference` using the trained map/graph-conditioned Qwen-VL model and a future-view generator. The code supports both a direct-answer baseline and a closed-loop world-model setting.

VLM loading and modality restoration. At inference time, the code reconstructs the multimodal model in the same way as during training. It first reads `training_args.bin` from the checkpoint directory to recover the base model ID, map encoder ID, graph encoder ID, map token count, graph token count, semantic-map flag, and numerical precision. It then:

1. loads the Qwen-VL base model;
2. monkey-patches the forward function to accept `map_pixel_values` and `graph_input_ids`;
3. re-attaches `map_encoder`, `graph_encoder`, `map_projector`, and `graph_projector`;
4. restores either a full checkpoint or a LoRA checkpoint;
5. if LoRA is used, separately loads `non_lora_state_dict.bin` before merging the adapter into the base model.

This is important because the map and graph branches are not native parts of the original Qwen-VL architecture and therefore must be explicitly re-created before checkpoint restoration.

Prompt and modality preparation. The inference code preserves the same placeholder mechanism used in training. Human prompts are normalized so that `<map>` and `<graph>` are expanded to repeated `<|map_pad|>` and `<|graph_pad|>` spans according to the checkpoint-specific token counts. When the graph is available but the prompt omits an explicit graph placeholder, the code automatically inserts one. For video inputs, the model samples frames using the same Qwen video preprocessing utilities as in training; by default, the scripts use up to 128 frames. The map input is constructed by aligning occupancy-map frames with the sampled video frame indices and then building a tiled map canvas. When semantic maps are enabled, semantic one-hot channels are concatenated to the RGB map channels before being fed to the adapted ViT map encoder. Graph inputs are read from `graph/graph.json` and tokenized to a maximum length of 256.

Two-stage closed-loop reasoning. The full inference pipeline uses the VLM twice. In the first stage, the VLM acts as a planner. In the second stage, after the world model synthesizes future observations, the VLM acts as an observer-answerer.

There are two planner modes:

- `actions`: the model directly predicts a short action sequence in JSON form;

- `navigation_target`: the model predicts a graph node ID and object name, and the code converts this symbolic target into a metric goal pose and a derived action sequence.

When graph input is unavailable for the current sample, the code automatically falls back from `navigation_target` mode to `actions` mode.

Question-aware planning prompts. Planner prompts are specialized by task type. The code contains separate prompt templates for generic action planning, navigation-target selection, SQA3D-style situation-grounded navigation, and route-planning tasks. For object-relative-direction questions, the planner prompt is augmented with an explicit explanation that the "standing by" object defines the starting position and the "facing" object defines the initial heading. For route-planning questions, the planner is encouraged to select the final destination object node rather than an intermediate landmark. For SQA3D, the planner is instructed to infer the start pose from the situation description before choosing a target that is useful for answering the question.

Source-view selection before world-model rollout. A distinctive implementation detail is that the planner is not always anchored to the last video frame. The code resolves ground-truth camera poses for all video frames and, when the question text provides a reliable start anchor, it selects a source frame whose pose best matches that textual anchor. This is used for route-planning, SQA3D, and object-relative-direction questions. If the selected planner-start pose differs too much from the current visual anchor (more than 1.0 m position error or more than 45° yaw error), the code inserts an explicit relocation segment before the task-specific actions. This makes the generated trajectory consistent with the semantics of the question rather than merely with the temporal end of the source video.

World-model conditioning and rollout. The world model is instantiated through `generate_video_from_image_camera_poses` and is configured in the provided scripts with `stabilityai/stable-virtual-camera` as the default backend. The default configuration in `inference_world_model_plan.sh` uses:

- model version 1.1;
- spatial resolution 576×576;
- shortest side 576;
- context frames `21,21`;
- classifier-free guidance `4.0,2.0`;
- guider types `1,2`;
- minimum CFG 1.2;
- camera scale 2.0;
- 20 denoising steps;
- trajectory prior enabled;
- crop-based transform target with scale 1.0;
- frame interval 3;
- field of view 90°;
- camera pitch 20°;
- output frame rate 8 FPS.

The model is conditioned on multiple source views extracted from the original video. By default, three matched conditioning frames plus the anchor frame are selected so that the conditioning set is geometrically close to the planned future path.

Observer-view construction. After planning, the code converts the symbolic actions into dense camera poses and writes them to `camera_poses.json`. It then concatenates: (i) the source conditioning poses, (ii) the planned forward trajectory, and, for some question types, (iii) additional orbit views around the target pose. For object-relative-direction tasks, the code renders a dense 15° orbit around the final pose and feeds a sparse 45° subset of up to seven views back into the VLM, so the observer receives multiple imagined perspectives from the same target location.

Follow-up answering. Once the imagined video is generated, the code either extracts the final predicted frame or a selected subset of observer views and asks the same VLM to answer the original question. The follow-up prompt explicitly states that the image(s) are predicted future views after executing the planned actions or after navigating to the selected target node. In navigation-target mode, the follow-up prompt also injects the metric goal pose (x, y, yaw) in map coordinates. If both map and graph are disabled, the code skips the world-model stage entirely and directly applies the trained VLM to the original video.

C.2 Inference Configurations Used in Practice

The provided shell scripts define the main evaluation settings.

Direct trained-model inference. The script `inference_trained.sh` evaluates the trained checkpoint from `output/4_10_stage2_lora_align/checkpoint-6000` by default. It enables semantic maps by default, resumes unfinished runs, and launches three modality ablations:

- map + graph,
- map only,
- graph only.

The default launcher uses GPUs `0,1,2,3` with 3 workers per GPU.

Closed-loop world-model inference. The script `inference_world_model_plan.sh` evaluates the full planner–world-model–observer pipeline. By default it uses:

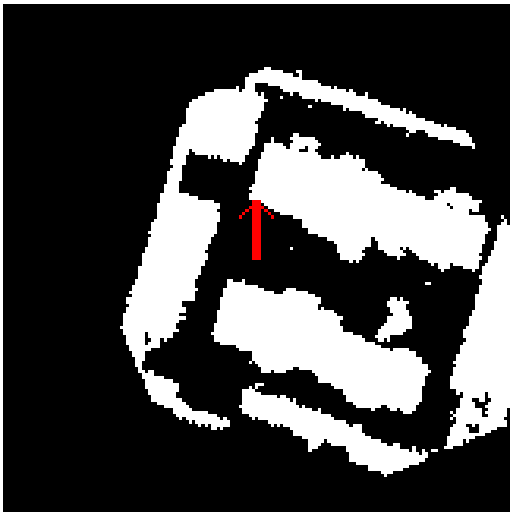
- the same stage-2 checkpoint at step 6000;
- 2 workers per GPU;
- route-planning validation data from `dataset/data_val_route_planning_scannet_scannetpp.json`;
- planner output mode `actions`;
- map directory `obs_bbox_id/occupancy`;
- graph file `graph/graph.json`;
- maximum VLM generation length 256, temperature 0, and top-p 0.9;
- VLM video sampling with up to 128 frames.

Outputs for reproducibility. For each sample, the code writes a detailed result package containing the resolved input paths, selected source frames, source and target poses, planner prompt, planner raw output, parsed actions or navigation target, world-model camera poses, rendered future view(s), observer prompt, and final answer. In the heavy-weight world-model setting, the output directory also contains `world_model.mp4`, `world_model.json`, `camera_poses.json`, `world_model_camera_poses.json`, and per-sample result summaries such as `vlm_world_model_result.json`. This serialization mirrors the test-pipeline design goal: every major intermediate state is materialized to disk for debugging and analysis.

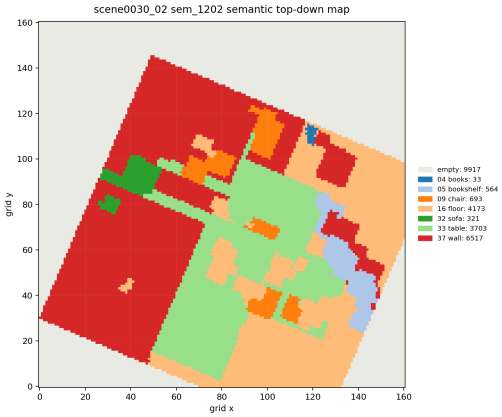
Interpretation. Conceptually, `tests/test_pipeline.py` specifies the algorithmic skeleton of LSM-VLM, namely recurrent memory initialization, plan generation, imagination, update, and answer. The code under `origin_code/inference` provides the concrete realization used in experiments: a map/graph-augmented Qwen-VL planner-observer, question-aware source-pose selection, Stable Virtual Camera rollout, and confidence-aware closed-loop reasoning over future predicted views.

D BEV Map and Graph Examples

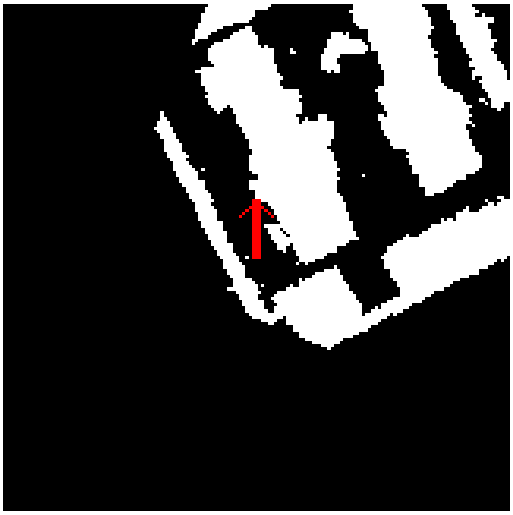
Here, the short-term memory represented by our BEV map does not refer to a memory that is immediately forgotten after observation. Instead, similar to the hidden state in an LSTM, this short-term memory is continuously maintained and updated over time. It is termed "short-term" because it is updated at a higher frequency and may gradually forget outdated information. In contrast, long-term memory is more stable and persistent, and is not subject to forgetting.



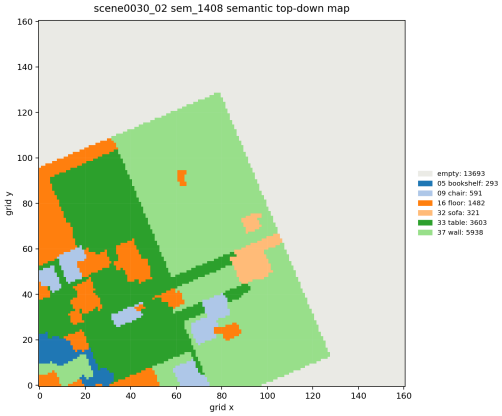
(a) Occupancy Channel from the first viewpoint.



(b) Semantic Channel from the first viewpoint.



(c) Occupancy Channel from another viewpoint.



(d) Semantic Channel from another viewpoint.

Figure 4: Examples of the BEV map representation from two viewpoints. Each viewpoint includes the Occupancy Channel and the Semantic Channel. The BEV map is agent-centric, with the agent located at the map center and its heading aligned upward.

In this section, we provide examples of the Occupancy Channel and Semantic Channel in our BEV map. The BEV map has a resolution of 161×161 grid cells, where each cell corresponds to 0.05 meters in the real world. The agent is always placed at the center of the map and is indicated by an upward-facing arrow. When the agent rotates, the entire map is rotated accordingly around the agent as the anchor, ensuring that the map remains expressed in the current agent-centric coordinate frame.

We also visualize the Semantic Channel of the BEV map. In our implementation, each grid cell in the Semantic Channel stores a 40-dimensional one-hot semantic encoding. For visualization purposes, we assign a distinct color to each semantic code and render the resulting semantic map as a colored grid.

The other two images below show the Occupancy Channel and Semantic Channel of the BEV map from another viewpoint.

We also present an example of the graph representation and provide a detailed explanation of each field. The schema below follows the generated `graph/graph.json` files and the fields consumed

by CognitiveMap for object lookup, viewpoint planning, memory update, and VLM-readable graph serialization.

```
{
  "schema_version": "1.1",          // Graph format version.
  "frame_id": 0,                    // Source frame used to initialize/update the graph.
  "timestamp": 0,                   // Optional time index; kept for temporal extensions.

  "frames": {
    "world": "map",                 // World frame used by pose_w and rel_world.
    "agent": "agent"                // Agent-local frame used by rel_agent.
  },

  "conventions": {
    "units": {
      "distance": "m",              // Distances are measured in meters.
      "angle": "deg"                // Angles are measured in degrees.
    },
    "agent_axes": "x_fwd, y_left",  // Agent-relative x is forward, y is left.
    "bearing_deg": "positive_to_right_of_agent_heading",
                                     // Positive bearing is to the agent's right.
    "yaw_deg": "counter_clockwise_from_world_x"
                                     // Agent/object yaw convention in the map frame.
  },

  "nodes": [
    {
      "node_id": "A0",              // Unique agent node id; CognitiveMap updates this pose.
      "pose_w": {
        "x": 2.51,                  // Agent x coordinate in the map/world frame.
        "y": 4.87,                  // Agent y coordinate in the map/world frame.
        "yaw": -30.79               // Agent heading; used for bearing and FOV tests.
      },
      "fov": {
        "half_angle_deg": 45.0,     // Half field-of-view angle for visibility labels.
        "range_m": 4.0              // FOV range for in_fov computation.
      }
    },
    {
      "node_id": "O00",             // Unique object node id used by graph edges/planning.
      "name": "chair",              // Detector/raw object name; indexed for text matching.
      "type": "chair",              // Canonical semantic class; also indexed for lookup.
      "pose_w": {
        "x": 1.06,                  // Object center x in the map/world frame.
        "y": 4.60                   // Object center y; used for navigation and merging.
      },
      "extent_w": {
        "w": 0.95,                  // Object footprint width in the world/map frame.
        "h": 0.85,                  // Object footprint height/depth in the world/map frame.
        "yaw": 74.78                // Object footprint orientation.
      },
      "score": 0.80,                // Detection/memory confidence for this object.
      "semantic_one_hot": [0, 0, 0] // NYU40-style semantic one-hot vector, shortened here.
    }
  ],

  "edges": [
    {
      "src": "A0",                  // Source node id.
```

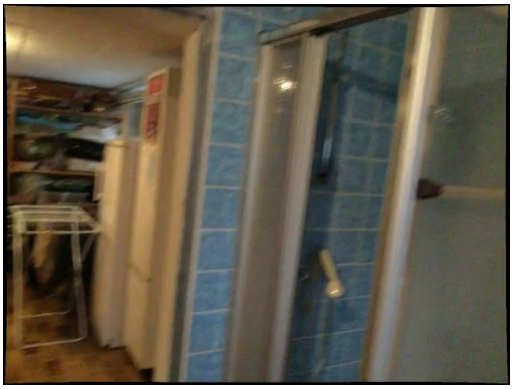
```
      "dst": "O00",                 // Destination object node id.
      "type": "agent_object",       // Agent-to-object relation.
      "rel_agent": {
        "range_m": 1.48,            // Agent-object distance.
        "bearing_deg": 138.99,      // Bearing in the agent frame.
        "rel_xy": {
          "x_fwd": -1.12,           // Object x in agent frame; forward is positive.
          "y_left": -0.97           // Object y in agent frame; left is positive.
        },
        "in_fov": false             // Whether object lies inside the agent FOV.
      },
      "qualitative": {
        "sector": "behind",         // Discrete bearing/FOV bucket for text serialization.
        "proximity": "near"         // Discrete range bucket.
      },
      "confidence": 0.80,           // Relation confidence.
      "last_updated_frame": 0       // Last frame that produced/updated this relation.
    },
    {
      "src": "O00",
      "dst": "O01",
      "type": "oo_spatial",         // Object-object spatial relation.
      "rel_world": {
        "delta_xy": {
          "dx": 2.23,               // dst.x - src.x in the world/map frame.
          "dy": -0.84               // dst.y - src.y in the world/map frame.
        },
        "distance_m": 2.38          // Euclidean object-object distance.
      },
      "relation": {
        "primary": "right_of"       // Main spatial predicate used by graph text/reasoning.
      },
      "qualitative": {
        "left_right_world": "right",// Coarse world-frame lateral relation.
        "near_level": "far"         // Coarse distance bucket.
      },
      "confidence": 0.52,
      "last_updated_frame": 0
    }
  ],

  "views_for_vlm": {
    "node_table": [
      "A0 agent pose_w=(2.51,4.87,yaw=-31) fov=+/-45deg range=4m",
      "O00 chair type=chair pose_w=(1.06,4.60) score=0.80"
    ],
    "edge_table": [
      "A0 -> O00 sector=behind range=1.48 bearing=+139 in_fov=0",
      "O00 <-> O01 far d=2.38 right_world"
    ]
                                    // Compact textual graph view tokenized by the VLM.
  }
}
```

E Case Study

In this section, we provide qualitative case studies to better illustrate how the proposed cognitive-map-based interaction pattern works in practice. Specifically, we include one successful example and one

(a) Case1-View1.

(b) Case1-View2.



(c) Case1-View3.

(d) Case1-View4.

Figure 5: Input Video Views.

failure example. The successful examples demonstrate how the world model and memory module contribute to spatial reasoning and long-horizon embodied understanding. The failure examples highlight current limitations, including unverifiable hallucinations in the world model and cases where our interaction pattern does not provide sufficient benefit.

E.1 Positive Case 1: Relative Direction Reasoning

The first positive example focuses on relative direction reasoning. This type of question requires the agent not only to recognize objects in the scene, but also to reason about their spatial relationships from an egocentric point of view. In particular, the question asks the model to infer the relative quadrant of one object with respect to the user's position and facing direction.

Question.

> If I am standing by the chair and facing the refrigerator, is the washing machine to my front-left, front-right, back-left, or back-right?
>
> The directions refer to the quadrants of a Cartesian plane, where I am standing at the origin and facing along the positive y-axis.

Input Video. Here we show four images in the input video. The object chair(left-down side of Case1-View1), washing machine(left-down side of Case1-View2) and refrigerator(left side of Case1-View3) have been detected during the detection video part. So right now the input memory include these three objects. But the problem is that VLM is not good at direction change, so right now if you let it answer directly, you will get a wrong answer.

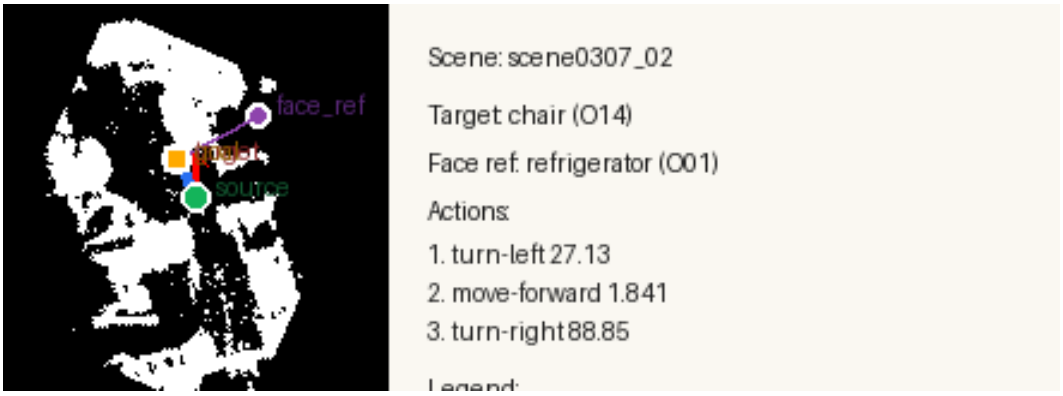


Figure 6: Action list plan visualized on the BEV map.

VLM-generated action list. As shown in Fig 6, given the question and the scene graph, the VLM first identifies the three objects explicitly mentioned in the query: the chair, the refrigerator, and the washing machine. These objects are treated as the primary reasoning anchors. In the scene graph, the chair is grounded to node O14, the refrigerator is grounded to node O01, and the washing machine is grounded to node O17. The chair defines the agent's target position, because the question asks the agent to imagine standing by the chair. The refrigerator defines the orientation reference, because the agent should face the refrigerator after reaching the chair. The washing machine is not selected as the navigation target; instead, it is reserved as the query object whose relative direction should be evaluated after the agent reaches the correct egocentric pose.

The VLM then converts this spatial reasoning objective into an executable action list for the world model. Starting from the last observed camera pose, the system plans a short navigation trajectory toward the chair and then rotates the camera to face the refrigerator. The resulting action list is:

```
turn-left 27.13;
move-forward 1.841;
turn-right 88.85.
```

World model imagination. Following the VLM-generated action list, the world model simulates the agent's future observations step by step. As shown in Fig. **??**, the agent first turns left from the last observed pose, then moves forward toward the chair, and finally turns right to align its view with the refrigerator. The imagined observations indicate that the agent successfully reaches the neighborhood of the chair and obtains the intended egocentric pose facing the refrigerator. Under this simulated viewpoint, the washing machine becomes the query object to be localized relative to the agent's final orientation, enabling the system to infer its direction from the imagined scene rather than from the original observation alone. From the World Model View4, we know the world model now have already generate a good view point.

Memory Update and VLM answering. After the world model completes the imagined action sequence, the system updates both the BEV map and the scene graph according to the agent's simulated final pose. At this stage, the agent's position and orientation are aligned with the condition specified in the question: the agent is standing near the chair and facing the refrigerator. This pose normalization is important because the original video observations are collected from different viewpoints, whereas the question requires reasoning in a specific egocentric coordinate frame.

In the updated BEV map, the chair is treated as the origin of the local coordinate system, and the direction from the chair to the refrigerator defines the positive y-axis. The relative position of the washing machine is then evaluated under this newly established egocentric frame. Since the washing machine lies behind the agent and to the agent's right with respect to the refrigerator-facing orientation, it falls into the back-right quadrant of the Cartesian plane described in the question.

Therefore, after incorporating the world-model imagination result into memory, the VLM determines that the available spatial information is sufficient to answer the question reliably. The final answer is:

(a) World-Model View1.

(b) World-Model View2.

(c) World-Model View3.

(d) World-Model View4.

Figure 7: Input Video Views.

Back-Right.

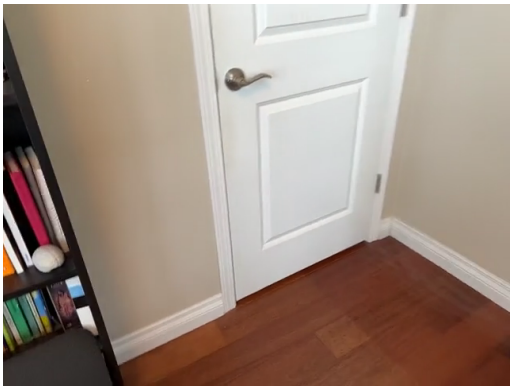
E.2 Negative Case 1: Hallucinated Spatial Evidence

This negative example focuses on hallucinated spatial evidence produced during question-conditioned imagination. The question requires the agent to reason about an object or relation that is only partially visible in the input video. The system therefore asks the world model to imagine a missing viewpoint. However, the imagined rollout introduces an unsupported object appearance or an incorrect object-object relation, and this hallucinated evidence propagates into the final VLM answer.

Question.

> You are a robot beginning at the black desk chair and facing the bookshelf. You want to navigate to the red desk chair. You will perform the following actions (Note: for each [please fill in], choose either 'turn back,' 'turn left,' or 'turn right.'): 1. [please fill in] 2. Go forward until the red desk chair. You have reached the final destination.

Input Video. The input video contains the reliable visual evidence used to initialize the memory. In this case, the relevant anchors are *black desk chair*, *bookshelf*, and *red desk chair*. These objects are either directly detected in the observed frames or inferred from high-confidence memory entries. Importantly, the hallucinated entity or relation that later appears in the imagined rollout is not supported by these observed frames.



(a) Case2-View1.

(b) Case2-View2.

(c) Case2-View3.

(d) Case2-View4.

Figure 8: Input Video Views.

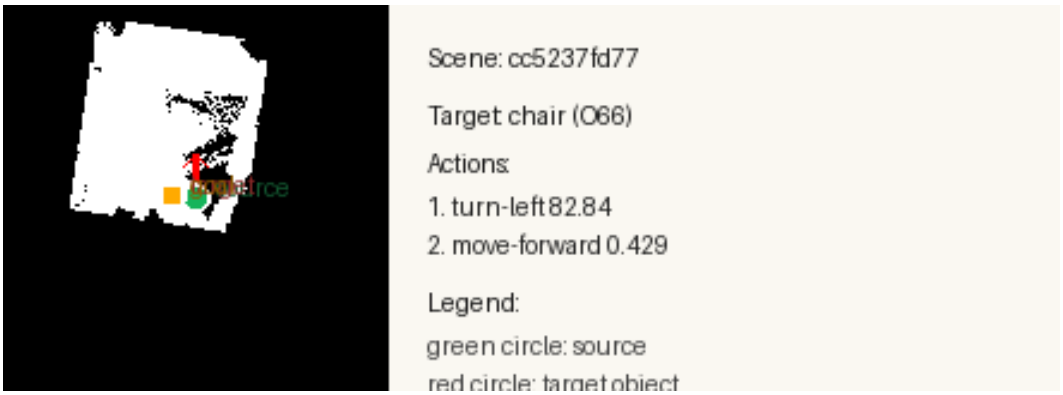


Figure 9: Action list plan visualized on the BEV map.

VLM-generated action list. Given the question and the current scene graph, the VLM identifies the objects and spatial relations that are necessary for answering. It then selects an informative target pose intended to reveal the missing evidence. The resulting action list can be summarized as:

```
turn-left 82.84;
move-forward 0.429;
```

The planned trajectory is reasonable under the current memory: it moves the agent toward *red desk chair* and rotates the view toward the region where the relative position of red desk chair with respect

(a) World-Model View1.

(b) World-Model View2.

(c) World-Model View3.

(d) World-Model View4.

Figure 10: Negative Case World Model Imagine Results.

to the bookshelf is expected. The failure does not come from the action list itself, but from the imagined observation generated after executing this plan.

World model imagination. Following the action list, the world model generates a future observation sequence. In the early imagined frames, the scene remains consistent with the observed video and the memory. However, after the viewpoint changes to the unobserved region, the world model introduces hallucinated object. This evidence is visually plausible, but it is not grounded in the input video or in the high-confidence scene graph.

Memory Update and VLM answering. After the imagined rollout, the candidate update is passed to the memory module. Ideally, confidence-aware memory update should suppress hallucinated content when it conflicts with reliable prior evidence. In this case, however, the hallucinated evidence is difficult to reject because it appears in an unobserved region and does not directly contradict a high-confidence existing entry. As a result, the VLM treats the imagined evidence as if it were valid scene evidence.

The final answer is therefore biased toward the hallucinated observation:

Turn Right

The correct answer should be:

Turn Back

Failure analysis. This case illustrates a hallucination pattern with three stages. First, the question requires evidence from a viewpoint that is weakly covered or not covered by the input video. Second, the world model fills this missing region with plausible but unsupported content. Third, because the

Table 3: Repeated-seed results on VSI-Bench. We report mean $\pm$ standard deviation over $N = 3$ world-model seeds. The dataset split, model checkpoint, prompts, and evaluation protocol are fixed across all runs.

Method	Avg.	Rel. Dir.	Route Plan	App. Order
LSM-VLM w/o Mem.	67.1 ± 0.3	74.1 ± 0.3	51.0 ± 1.4	78.3 ± 0.1
LSM-VLM	69.2 ± 0.4	82.8 ± 0.3	54.9 ± 1.7	78.3 ± 0.1

hallucinated content is not directly contradicted by existing memory, it can pass through the update stage and influence the VLM's final reasoning. This failure suggests that future systems should add stronger verification for imagined evidence, for example by requiring multi-view consistency, checking object persistence across imagined frames, or explicitly marking low-support imagined content as uncertain rather than treating it as observed evidence.

F Statistical Significance

Sources of randomness. We inspect the sources of randomness in our inference pipeline. The VLM evaluation uses greedy decoding with temperature set to 0 by default, and therefore does not introduce active sampling randomness. The world-model video generation is the main stochastic component and uses a fixed random seed by default, set to 42 in our standard evaluation. Therefore, once the dataset split, model checkpoint, prompts, preprocessing pipeline, and evaluation protocol are fixed, the remaining evaluation variability mainly comes from the random seed used by the world model. To quantify this variability, we repeat evaluation by varying only the world-model seed.

Repeated-seed evaluation. We evaluate the full LSM-VLM model and the strongest internal ablation, LSM-VLM without confidence-aware recurrent memory update, under multiple world-model seeds. Specifically, we use $N = 3$ seeds, $\{0, 42, 3407\}$, for the world-model generation module while keeping all other factors unchanged. The main paper reports the mean performance over these three seeds, and this appendix provides the corresponding standard deviations for the key metrics supporting our main claims.

For each metric, given scores $\{x_i\}_{i=1}^{N}$, we compute the mean and sample standard deviation as

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i, \qquad s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}.$$

Unless otherwise stated, all error bars denote one standard deviation across world-model seeds. These error bars capture inference-time variability induced by world-model stochasticity, rather than training-time variability.

Results and uncertainty analysis. As shown in Table 3, LSM-VLM consistently outperforms the strongest internal ablation under repeated world-model seeds. The full model improves the overall VSI-Bench average from 67.1 ± 0.3 to 69.2 ± 0.4, corresponding to a gain of $+2.1$ points. The improvement is especially large on relative direction reasoning, where LSM-VLM improves from 74.1 ± 0.3 to 82.8 ± 0.3, and on route planning, where it improves from 51.0 ± 1.4 to 54.9 ± 1.7. Route planning exhibits larger variance than the other metrics, which is expected because it depends more directly on the stochastic imagined rollouts generated by the world model. In contrast, appearance order remains nearly unchanged across seeds, suggesting that this task is mainly determined by observed temporal evidence rather than world-model stochasticity.

G Broader Impact

This work studies 3D spatial reasoning for vision-language models in embodied scenes. Potential positive impacts include improving the spatial understanding capabilities of embodied AI systems, such as assistive robots, navigation agents, household service robots, and systems that help users understand complex indoor environments. By explicitly maintaining structured spatial memories and

completing missing visual evidence through question-conditioned imagination, the proposed framework may make embodied agents more reliable in long-horizon reasoning, cross-view correspondence, route planning, and situated question answering.

At the same time, the proposed techniques also have potential negative societal impacts. First, stronger spatial reasoning and scene-memory capabilities could be misused for surveillance, unauthorized monitoring, or privacy-invasive reconstruction of indoor environments, especially when applied to videos of private spaces. Second, because the method uses a generative world model to imagine unobserved viewpoints, it may produce hallucinated objects, incorrect spatial relations, or misleading scene layouts. If such outputs are used in downstream embodied systems, incorrect reasoning could lead to unsafe navigation or inappropriate actions. Third, deployment in real-world assistive or robotic systems may amplify dataset biases or perception errors, for example if the underlying detectors, VLM backbone, or training data perform unevenly across environments, object categories, or user groups.

We discuss several technical design choices that partially mitigate these risks. In particular, LSM-VLM does not directly rely on raw imagined images as final evidence; instead, imagined observations are converted into structured candidate updates and selectively consolidated through confidence-aware memory updates. This design is intended to reduce the risk of contaminating long-term memory with hallucinated or low-confidence content. For real-world deployment, additional safeguards would be necessary, including privacy-preserving data collection, user consent for indoor video use, restricted or gated release in sensitive settings, monitoring for hallucinated spatial evidence, and human oversight when the system is used for safety-critical navigation or assistive decision-making.

H Assets and License

Our work builds on several publicly available research assets, including pretrained models, perception modules, datasets, and benchmarks. We use Qwen3-VL-8B-Instruct as the vision-language backbone, Stable Virtual Camera as the camera-conditioned world model for novel-view generation, and VGGT together with SAM3 for geometry estimation, segmentation, and perceptual grounding. We cite the original papers for all these components in the main paper.

For training and evaluation, we use existing 3D indoor-scene datasets and spatial reasoning benchmarks. Specifically, we train the proposed map and graph encoders using the training splits of ScanNet and ScanNet++, and evaluate the method on VSI-Bench and SQA3D. ScanNet and ScanNet++ provide annotated indoor 3D scenes, while VSI-Bench and SQA3D are used only for benchmarking spatial reasoning and situated question answering. We do not introduce a new dataset containing human subjects, private user data, or newly collected indoor videos.

All existing assets are used for research purposes and are credited through citations to their original papers. We follow the intended research-use setting of these assets and do not redistribute the original datasets, pretrained model weights, or third-party code as part of this submission. Users who wish to reproduce our experiments should obtain each asset from its official source and comply with the corresponding license, terms of use, and data-access requirements. Any released code from our side will contain instructions for installing dependencies and preparing these external assets, but will not repackage third-party data or model weights unless their licenses explicitly permit redistribution.

The main external assets used in this work are:

- **Qwen3-VL-8B-Instruct**: used as the pretrained VLM backbone. We cite the original Qwen3-VL technical report and follow its model usage terms.
- **Stable Virtual Camera**: used as the camera-conditioned world model for generating imagined observations from target viewpoints. We cite the original work and use it under its released research license and terms.
- **VGGT**: used for camera-aware geometry and visual grounding. We cite the original work and follow its release terms.
- **SAM3**: used for object segmentation and mask extraction. We cite the original work and follow its release terms.

- **ScanNet and ScanNet++**: used for training the newly introduced map and graph encoders. We do not redistribute the datasets and require users to obtain them from the official dataset sources under their respective licenses and data-use agreements.
- **VSI-Bench and SQA3D**: used as evaluation benchmarks for video-based spatial reasoning and 3D situated question answering. We cite the original benchmark papers and follow their official usage terms.

To the best of our knowledge, our use of these assets is consistent with their intended academic research purposes. We will include license and access information in the released code documentation where applicable, and we will ask users to follow the licenses and terms of all upstream assets when reproducing or extending our work.

I Prompt Examples and Token Examples

I.1 Prompt Examples

Our inference pipeline uses a two-stage prompting strategy. In the first stage, the VLM receives an egocentric video together with an occupancy map and a scene graph, and predicts an informative navigation target for world-model imagination. In the second stage, after the world model renders the target view and the memory module updates the scene representation, the VLM answers the question using both the imagined observation and the structured memory.

Stage 1: World Model Planning Prompt.

```
You are given an egocentric video, an occupancy map, and a scene
graph from the same indoor scene.
Choose one target object node from the graph that defines where the
agent should navigate in order to answer the question.
Do not answer the question directly.  Prefer the object that
provides the most informative viewpoint for reasoning.

Question:
If I am standing by the backpack and facing the door, is the trash
bin to my front-left, front-right, back-left, or back-right?

Options:
A. front-right
B. back-left
C. back-right
D. front-left

Question-grounded anchors:
- standing object:  backpack (start-position anchor)
- facing object:  door (heading anchor)
- queried object:  trash bin

Return JSON only:
{"target_node_id":"<graph node id>", "target_name":"<object name>",
"reason":"<short reason>"}
```

Stage 2: Answer Generation Prompt.

```
This image is the predicted future view after navigating to target
trash bin (O17).
The navigation goal pose in map coordinates is x=2.14, y=1.36,
yaw=1.57 rad.
Answer the question below using this image together with the updated
occupancy map and scene graph when helpful.

Question:
If I am standing by the backpack and facing the door, is the trash
bin to my front-left, front-right, back-left, or back-right?
```

```
Options:
A. front-right
B. back-left
C. back-right
D. front-left

Return only the answer.
```

I.2 Token Examples

To make the multimodal input explicit, we show the Stage 1 planning prompt in both human-readable form and its serialized token form before being fed into the VLM.

Human-readable Stage 1 prompt.

```
<video>
<map>
<graph>
You are given an egocentric video, an occupancy map, and a scene
graph from the same indoor scene.
Choose one target object node from the graph that defines where the
agent should navigate in order to answer the question.
Do not answer the question directly.  Prefer the object that
provides the most informative viewpoint for reasoning.

Question:
If I am standing by the backpack and facing the door, is the trash
bin to my front-left, front-right, back-left, or back-right?

Options:
A. front-right
B. back-left
C. back-right
D. front-left

Question-grounded anchors:
- standing object:  backpack (start-position anchor)
- facing object:  door (heading anchor)
- queried object:  trash bin

Return JSON only:
{"target_node_id":"<graph node id>", "target_name":"<object name>",
"reason":"<short reason>"}
```

Serialized multimodal prompt. Let K_m and K_g denote the numbers of map tokens and graph tokens, respectively. After preprocessing, the prompt is transformed into:

```
<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
<|vision_start|><|video_pad|><|vision_end|>
<|map_pad|> ...  <|map_pad|>                                  (K_m times)
<|graph_pad|> ...  <|graph_pad|>                              (K_g times)
You are given an egocentric video, an occupancy map, and a scene
graph from the same indoor scene.
Choose one target object node from the graph that defines where the
agent should navigate in order to answer the question.
Do not answer the question directly.  Prefer the object that
provides the most informative viewpoint for reasoning.
Question:
If I am standing by the backpack and facing the door, is the trash
bin to my front-left, front-right, back-left, or back-right?
Options:
```

```
A. front-right
B. back-left
C. back-right
D. front-left
Question-grounded anchors:
- standing object: backpack (start-position anchor)
- facing object: door (heading anchor)
- queried object: trash bin
Return JSON only: {"target_node_id":"<graph node id>",
"target_name":"<object name>", "reason":"<short reason>"}<|im_end|>
<|im_start|>assistant
```

Compact notation. The final VLM input can be summarized as

$$\texttt{X} = \texttt{[SYS]} \oplus \texttt{[VID]} \oplus \texttt{[MAP]}^{K_m} \oplus \texttt{[GRAPH]}^{K_g} \oplus \texttt{[TEXT\_plan]},$$

where $\oplus$ denotes concatenation,

$$\texttt{[VID]} = \texttt{<|vision\_start|><|video\_pad|><|vision\_end|>},$$

$$\texttt{[MAP]} = \texttt{<|map\_pad|>}, \qquad \texttt{[GRAPH]} = \texttt{<|graph\_pad|>}.$$

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes].

   Justification: The abstract and Introduction accurately summarize the scope and contributions of LSM-VLM, including the dual-memory design, the world-model-based reasoning component, and the empirical improvements reported on VSI-Bench and SQA3D. The claims are aligned with the experimental results and are limited to the evaluated embodied/spatial reasoning settings.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes].

   Justification: The Conclusion includes a dedicated "Limitations and Future Work" paragraph covering backbone coverage, inference efficiency, and the current lack of explicit memory conditioning in the world model.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A].

   Justification: The paper does not present theoretical theorems, formal claims, or proofs; it is an empirical method paper.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes].

   Justification: The Method section specifies the model components and memory update procedure, while the Experiments section describes datasets, benchmarks, training stages, optimization details, baselines, ablations, and runtime analysis needed to understand and reproduce the main claims.

   Guidelines:

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

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes].

   Justification: We will provide anonymized code with instructions for reproducing the main experimental results, including environment setup, data preparation, and evaluation scripts, in the supplemental material.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes].

   Justification: The Experiments section specifies the training data sources and splits, two-stage training protocol, trainable modules, LoRA adaptation, optimizer, learning rates, batch size, training steps, benchmarks, metrics, and baseline categories. More training/evaluation details will be in appendix.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes].

Justification: The paper reports benchmark scores and ablation results in the main text, and the supplementary material provides additional repeated-run analyses with uncertainty estimates for the key experiments supporting the main claims. Specifically, the supplement reports error bars computed over repeated runs under the same evaluation protocols, and describes the source of variability and the calculation procedure used for these estimates.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

Answer: [Yes].

Justification: The Experiments section states that training is performed on 8 H100 GPUs and Table 2 reports per-component inference time, call counts, and total inference time. The paper also describes efficiency considerations in the ablation section. More details are in appendix.

Guidelines:

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

9. **Code of ethics**

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

Answer: [Yes].

Justification: The research uses public benchmarks and existing model components for spatial VQA research and does not involve human-subject experiments, private data collection, or deception. The authors have reviewed the NeurIPS Code of Ethics and are not aware of deviations.

Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes].

    Justification: The Appendix includes a "Broader Impact" section discussing potential benefits for embodied and assistive systems, as well as privacy, surveillance, hallucination, and safety risks with suggested mitigations.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A].

    Justification: The paper does not release a new high-risk pretrained language model, image generator, scraped dataset, or deployment-ready system. The work uses existing model components in a research pipeline, and no new broadly deployable asset with elevated misuse risk is released in this submission.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes].

    Justification: The paper properly credits the creators of the existing assets used in the work, including VSI-Bench, SQA3D, the underlying model components, and implementation libraries. All assets are used in accordance with their stated licenses and terms, and we do not repackage or redistribute existing datasets under modified licenses. We cited their location in the paper.

    Guidelines:

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

13. **New assets**

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

    Answer: [Yes].

    Justification: The submission will include anonymized code as a new asset, together with documentation for installation, data preparation, training/evaluation, and reproduction of the main results. Release of model checkpoints is still under consideration and will be documented if included.

    Guidelines:

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

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

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

    Answer: [N/A].

Justification: The research does not involve crowdsourcing experiments or research with human subjects.

Guidelines:

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

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

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

Answer: [N/A].

Justification: The research does not involve crowdsourcing or human-subject studies, so IRB or equivalent review is not applicable.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes].

Justification: The paper explicitly describes the use of VLMs as a core methodological component, including Qwen3-VL-8B as the backbone and the VLM roles in pose prediction, memory reading, and answer generation.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.