H3D-LVR: HIERARCHICAL 3D LATENT VISUAL REASONING FOR MULTI-VIEW VLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language models have achieved strong progress in visual perception and instruction following, yet multi-view 3D spatial reasoning remains challenging. Although textual chain-of-thought can expose intermediate reasoning steps, it operates at the output level and does not directly constrain the model to form spatially correct intermediate hidden states. Existing 3D or latent visual reasoning methods provide spatial supervision, but typically rely on a single-level latent representation, which is insufficient to capture the multi-granular information required for multi-view reasoning. We propose H3D-LVR, a hierarchical 3D latent visual reasoning framework that organizes a VLM's reasoning trajectory around three generated latent states: scene layout, key objects, and input views. During SFT, VGGT-derived layout features and matched object/view features supervise the corresponding spans through reconstruction losses. The corresponding features also fill latent-pad inputs under teacher forcing, but no such features are provided at inference. We further adapt answer-level GRPO to this hierarchical latent format with hidden-state replay and full-span loss masking, allowing reinforcement learning to improve answer behavior without corrupting the learned latent-token semantics. During inference, H3D-LVR uses only the input images and prompt; it does not run VGGT, receive precomputed layout/object/view features or annotations, or invoke external 3D tools. Experiments on MindCube-Tiny, SPAR-Bench, and MMSI-Bench demonstrate that H3D-LVR achieves substantial performance gains, strongly validating its effectiveness.

1 INTRODUCTION

Supervised fine-tuning and reinforcement learning have enabled vision-language models (VLMs) to generate high-quality chains of thoughts (CoTs), making them strong general-purpose visual reasoners (Liu et al., 2023; Bai et al., 2025; Zhang et al., 2023). However, reasoning about complex 3D spatial relationships across limited varying 2D views remains a fundamental challenge. This requires models to identify key objects shared across views, infer camera-relative directions, and maintain a coherent scene representation throughout reasoning. However, current VLMs still struggle with such multi-view spatial reasoning, as shown in Figure 1, where the model incorrectly places the white-red cabinet opposite the black bag and produces the wrong answer.

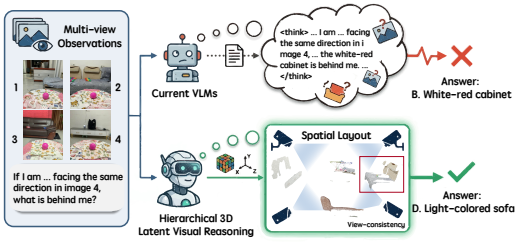


Figure 1: Hierarchical 3D latent visual reasoning. H3D-LVR asks a VLM to organize multi-view spatial reasoning around three generated latent states: scene layout, key objects, and input views.

One key limitation is that explicit CoT reasoning operates at the model's output layer and may not constrain the model to form correct intermediate hidden states. Recent works (Chen et al., 2024a; Wu et al., 2025; Chen et al., 2025b; Li et al., 2026) have explored 3D/latent visual reasoning for VLMs and provided initial evidence for the effectiveness of spatial supervision. However, existing methods rely on single-level spatial latent representations and fail to capture the multi-granular

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

information inherent in multi-view reasoning, including scene-level layouts for maintaining global consistency, object-level anchors for tracking key entities, and view-level anchors for encoding camera viewpoints and directions, thereby limiting their performance.

To this end, we propose H3D-LVR, a hierarchical 3D latent visual reasoning framework for multi-view VLMs. Instead of placing all reasoning in free-form text, H3D-LVR makes the spatial state explicit through three types of generated latent spans. Within this hierarchy, a layout span at the beginning of the ¡think¿ block initializes a scene-level state before textual reasoning, object spans are attached to key entities in the reasoning trace, and view spans are attached to referenced input views. To align the three types of latent spans with their intended spatial semantics, geometry-rich 3D features extracted by VGGT (Wang et al., 2025a) are decomposed into layout, object, and view-level supervision signals. This design gives the model explicit latent slots tied to the spatial entities that the reasoning trace discusses, while keeping the final inference path end-to-end and tool-free.

To support the training of these latent spans, we construct a feature-aligned SFT corpus from existing multi-view spatial and object-relation data (Yin et al., 2025; Zhang et al., 2026). The construction uses strong VLMs to produce structured CoT with descriptive object and view anchors, localizes tagged objects with an automated bbox pipeline, extracts VGGT layout/view/object features, and filters unreliable object features caused by small boxes or crop truncation. High-quality feature-aligned data are essential, as latent regression is highly sensitive to mismatched object-feature pairs, while noisy local targets can undermine object-level supervision.

We further study answer-level reinforcement learning after SFT. Directly applying answer reward to a response containing latent special tokens is risky, because pad tokens inside layout/object/view spans are not ordinary text tokens; they are meant to carry continuous hidden-state semantics learned through feature reconstruction. We therefore use hidden-state replay and full-span loss masking. Rollouts generate complete hierarchical responses; when GRPO recomputes current- and reference-policy log probabilities, it replays the same sequence through the hierarchical path, while the policy loss is masked over layout/object/view spans. This lets RL improve natural-language reasoning ability and final-answer accuracy while preserving the latent span semantics learned during SFT. Our contributions are summarized as follows:

- We introduce a hierarchical 3D latent reasoning format that interleaves textual CoT with layout, object, and view latent spans for multi-view spatial reasoning.
- We use VGGT-derived 3D features as SFT-only latent-pad inputs and layout, object, and view reconstruction targets, enabling multi-granularity 3D supervision inside the generated reasoning trajectory.
- We adapt answer-level GRPO to hierarchical latent VLMs with hidden-state replay and full-span loss masking, preventing answer reward from directly corrupting latent special-token semantics.
- We evaluate H3D-LVR on MindCube-Tiny, SPAR-Bench, and MMSI-Bench, where it shows strong performance and consistent improvements, confirming the effectiveness of our method.

2 Related Work

2.1 Vision-Language Models and Spatial Reasoning

In recent years, Vision-Language Models (VLMs) have achieved remarkable progress across various visual understanding tasks (Fu et al., 2024; Liu et al., 2023; Mathew et al., 2021; Li et al., 2024c;b;a; Bai et al., 2023; Wang et al., 2024; Bai et al., 2025; Chen et al., 2024c;b; Hurst et al., 2024; Team et al., 2024). As application scenarios extend into the physical world, spatial intelligence and 3D geometric perception have emerged as focal points of research (Ray et al., 2024; Zhang et al., 2026; Tang et al., 2025; Chen et al., 2024a; Zhang et al., 2025; Yang et al., 2025a; Yin et al., 2025). To break through the limitations of purely 2D visual cues, recent advanced works such as Spatial-MLLM (Wu et al., 2025) and 3DThinker (Chen et al., 2025b) have attempted to introduce pre-trained 3D foundation models (e.g., VGGT (Wang et al., 2025a)) to extract 3D spatial features, thereby enhancing the spatial reasoning capabilities of VLMs. However, these existing 3D reasoning models typically treat the 3D scene features as a single macroscopic global layout for

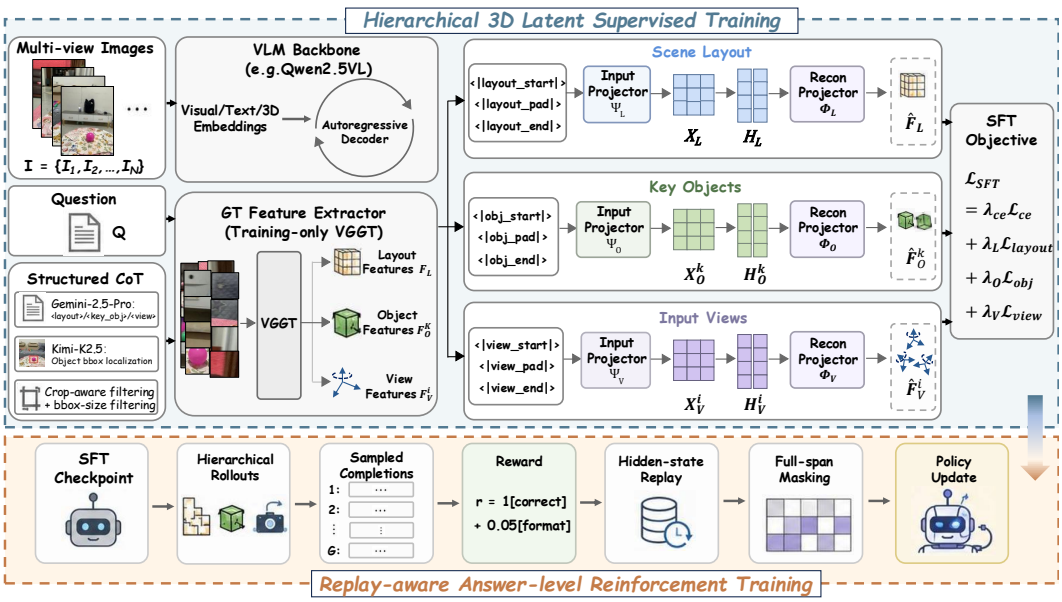


Figure 2: Overview of H3D-LVR. During SFT, VGGT-derived features fill latent-pad inputs and supervise layout, object, and view reconstruction. After SFT, replay-aware RL computes policy scores under the generated hierarchical latent context while masking layout, object, and view spans from answer-level policy gradients.

coarse alignment. They lack a precise depiction of the fine-grained 3D morphology of local objects and dynamic camera views. In contrast to these methods, our work proposes a multi-granularity, hierarchical reasoning framework. We are the first to finely decouple 3D latent spatial imagery into three dimensions: global layout, key objects, and camera views. This enables accurate reasoning over local spatial relationships and complex geometric transformations.

2.2 Visual and Latent Chain-of-Thought Reasoning

Chain-of-Thought (CoT) prompting has significantly enhanced the complex logical reasoning capabilities of large language models (Wei et al., 2022; Zhang et al., 2023). To overcome the representational bottleneck of discrete text sequences when modeling continuous geometric structures, recent studies have begun extending CoT into the multimodal latent space (Wang et al., 2025b; Chen et al., 2025a; Yang et al., 2025b). For instance, Chain-of-Visual-Thought (CoVT) (Qin et al., 2025) proposes interleaving continuous visual tokens containing perceptual cues (e.g., depth, segmentation) within the text reasoning trajectories. Furthermore, Latent Visual Reasoning (LVR) (Li et al., 2026)introduce a novel paradigm that guides models to autoregressively reconstruct visual semantics in the latent space. Although these latent reasoning methods effectively broaden the representational capacity of models, they predominantly rely on 2D visual encoders, leading to the limitation of representational redundancy. Our work inherits the elegant paradigm of continuous latent reasoning but fundamentally extends it into the 3D domain. By introducing an adaptive spatial feature purification pipeline based on multi-view redundancy dynamic filtering, we provide VLMs with high-purity, multi-granularity 3D supervision signals, effectively bridging the modality gap from 2D to 3D.

3 Method

3.1 Problem Formulation

Given multi-view images $\mathcal{I} = \{I_1, \ldots, I_N\}$ and a question Q, the model predicts a reasoning trace Y and a final answer A. Standard VLM fine-tuning optimizes the likelihood of answer text, or a textual CoT followed by the answer. Figure 2 gives an overview of our format. H3D-LVR asks the

assistant response to contain a hierarchy of latent spatial spans:

$$Y = [\langle \text{think} \rangle,\ S_L,\ T(S_O, S_V),\ \langle /\text{think} \rangle,\ \langle \text{answer} \rangle A], \tag{1}$$

where S_L is a layout span emitted at the beginning of the <think> block before textual reasoning, S_O denotes object spans attached to key-object anchors inside the reasoning trace, S_V denotes view spans attached to view anchors, and $T(\cdot)$ is the surrounding natural-language reasoning. The VLM input is unchanged at inference: it receives the same multi-image and question as a standard multi-view VLM. Training is the only place where 3D features enter: under teacher forcing, they fill latent-pad inputs and serve as reconstruction targets; inference uses only the images and prompt.

This response design follows the structure of multi-view spatial reasoning. The model first needs a scene-level state to summarize the observed layout; it then needs object-level anchors for the entities mentioned by the question; and it finally needs view-level anchors because many relations are expressed relative to an input image or camera direction. We make these levels explicit as generated spans rather than hiding them in an unstructured CoT. The span positions are not globally fixed in the sequence; instead, they create explicit latent slots tied to the layout prefix, key-object mentions, and view mentions. During SFT, the matched feature is placed at the slot input, and the resulting hidden state is trained to reconstruct the target feature for the same level.

3.2 Feature-Aligned Data Construction

We process multi-view spatial reasoning and object-relation data (Yin et al., 2025; Zhang et al., 2026) into a unified hierarchical response schema. First, Gemini-2.5-Pro (DeepMind, 2025) generates structured CoT responses with temporary markup tags: <layout> marks the scene-level description, <key_obj> marks descriptive object phrases, and <view> marks explicit view references. These tags are used only during data construction. They tell the collator where to insert the final model special tokens, and they are removed before the response is used for training.

Second, before converting the tags to model tokens, we align the marked text with 3D features. For every <key_obj> phrase extracted from the structured CoT, we query Kimi-K2.5 (Moonshot AI, 2026) on each input image to determine whether the referenced object is present and, if so, to return a tight bounding box. VGGT then extracts layout and view features, while object features are pooled from the valid boxes matched to the same text anchor. Object-level supervision is the fragile part: a correct noun phrase can still produce a wrong regression target if its box is tiny, heavily cropped, or not aligned with the VGGT feature grid. We retain object-latent supervision only when at least one per-view detection remains valid after the crop-aware filtering described below. Specifically, each normalized box is mapped through the same resize and center-crop transformation used for VGGT feature extraction; its cropped-area ratio must satisfy the view-count-dependent threshold in Equation 2, and its minimum side after resizing must be at least 50 pixels in the main setting. The generation prompt and feature-alignment details appear in Appendices A.1 and A.2, respectively. If no box passes, the phrase remains in the CoT as ordinary text, but it does not receive an object span or object reconstruction loss.

Third, the collator converts the cleaned markup into the final assistant response. The <layout> tag becomes one layout special-token span, each retained <key_obj> anchor is followed by an object span, and each <view> anchor is followed by a view span. The assistant response begins with <think>, immediately followed by the layout latent span before the textual layout description. Inside the reasoning text, the model first generates the readable object phrase or view reference and then emits the corresponding object or view latent span. Thus, a latent span does not replace the text explanation; it gives the following reasoning tokens a compact hidden state tied to the just-mentioned entity or view. At evaluation time, none of these markup tags, boxes, or feature targets are provided.

After text-feature matching, the main quality-control step is crop-aware validation. VGGT preprocessing resizes and crops images before feature extraction, so objects near image boundaries can lose part of their visual evidence. Let $\gamma_{i,k}$ denote the fraction of the detected box for object anchor k that is removed by VGGT preprocessing in view I_i, and let N_k be the number of views in which object k has a valid matched box before crop-aware filtering. We define the object-specific threshold

as

$$\tau(N_k) = \begin{cases} 0.35, & N_k \geq 3, \\ 0.25, & N_k = 2, \\ 0.20, & N_k = 1. \end{cases} \tag{2}$$

A view-specific object feature is discarded when $\gamma_{i,k} > \tau(N_k)$, because a heavily cropped box no longer provides a faithful local target for object-level latent regression. If at least one view remains valid, the remaining local features are averaged across views to form the object target. If all views are invalid, the object anchor is not used for feature regression. We further apply bbox-size quality control in the main setting, removing boxes whose minimum side length is below 50 pixels after resizing. This threshold is a quality/coverage trade-off: weak filtering retains more samples but introduces tiny-box noise, whereas overly strict filtering reduces the coverage of relevant relations.

3.3 Hierarchical 3D Latent Supervision

Hierarchical latent spans. H3D-LVR adds three special-token families to the VLM vocabulary. The layout family is <|layout_start|>, <|layout_pad|>, <|layout_end|>; the object family is <|obj_start|>, <|obj_pad|>, <|obj_end|>; and the view family is <|view_start|>, <|view_pad|>, <|view_end|>. The layout span contains 12 pad tokens and appears once at the beginning of the <think> block. Each key-object anchor is followed by an object span with two pad tokens, and each view anchor is followed by a view span with one pad token:

$$S_L = [s_L, p_L^{\times 12}, e_L], \quad S_O = [s_O, p_O^{\times 2}, e_O], \quad S_V = [s_V, p_V, e_V]. \tag{3}$$

Here s, p, and e denote the start, pad, and end tokens of each family. Start and end tokens are generated like ordinary response tokens so the model learns when to open and close each structure. Pad tokens are not trained as ordinary next-token labels: their CE labels are ignored, and their hidden states are supervised by feature reconstruction instead. A strict rollout processor enforces the fixed number of pad tokens after each valid span start, prevents pad/end tokens from appearing in normal text regions, and caps the number of object/view spans.

VGGT feature targets. For each training sample, VGGT produces per-image feature sequences that are converted into three target types. Let $F_{\text{vggt}} \in \mathbb{R}^{N \times T \times C}$ denote the VGGT feature sequence for N views. The current implementation uses $C = 2048$. The layout target uses dense spatial tokens after removing prefix tokens, the view target uses the matched view feature, and the object target pools local layout features inside aligned boxes:

$$F_L = F_{\text{vggt}}[:, \mathcal{P}, :], \quad F_O^k = \text{Pool}(F_L, \mathcal{B}_k), \quad F_V^i \in \mathbb{R}^C. \tag{4}$$

Here $\mathcal{P}$ indexes the dense spatial-token region and $\mathcal{B}_k$ denotes the validated boxes for object anchor k. Object features are averaged only across views that pass the crop-aware filtering above. The three feature types therefore carry different granularity: layout is dense and scene-level, object is local and entity-level, and view is camera/image-level.

Input conditioning and contextualized latent states. Let $Y = (y_1, \ldots, y_T)$ denote the teacher-forced assistant response, and let $e(y_t)$ be the standard token embedding at position t. During SFT, the input projectors map the matched 3D features to model-hidden-size vectors:

$$X_L = \Psi_L(F_L), \quad X_O^k = \Psi_O(F_O^k), \quad X_V^i = \Psi_V(F_V^i), \tag{5}$$

where X_L, X_O^k, and X_V^i contain 12, 2, and 1 vectors, respectively. These vectors replace the standard embeddings $e(y_t)$ at the corresponding layout, object, and view pad positions; all natural-language tokens and span delimiters retain their ordinary token embeddings. The VLM decoder processes this feature-conditioned sequence together with the input images and produces a next-token distribution $p_\theta(\cdot \mid \mathcal{I}, y_{<t}, X_{<t})$ and a contextualized hidden state h_t at each position. We collect the decoder outputs at the same pad positions as H_L, H_O^k, and H_V^i.

Feature reconstruction and SFT objective. The reconstruction projectors map the contextualized pad states back to their corresponding feature spaces:

$$\widehat{F}_L = \Phi_L(H_L, E_{\text{img}}), \ \widehat{F}_O^k = \Phi_O(H_O^k), \ \widehat{F}_V^i = \Phi_V(H_V^i). \tag{6}$$

Here E_{img} denotes the original VLM image embeddings. The layout projector combines H_L with E_{img} to reconstruct the dense layout target F_L, while the object and view projectors reconstruct F_O^k and F_V^i from their corresponding pad states.

Let $\mathcal{M}_{\text{ce}}$ denote the supervised assistant-token positions, including natural-language tokens and span start/end delimiters but excluding all layout/object/view pad positions. The four SFT losses are

$$\mathcal{L}_{\text{ce}} = -\frac{1}{|\mathcal{M}_{\text{ce}}|} \sum_{t \in \mathcal{M}_{\text{ce}}} \log p_\theta(y_t \mid \mathcal{I}, y_{<t}, X_{<t}), \qquad \mathcal{L}_L = \text{MSE}(\widehat{F}_L, F_L),$$
$$\mathcal{L}_O = \text{MSE}\Big(\text{norm}(\widehat{F}_O), \text{norm}(F_O)\Big), \qquad \mathcal{L}_V = \text{MSE}\Big(\text{norm}(\widehat{F}_V), \text{norm}(F_V)\Big). \tag{7}$$

The full objective is

$$\mathcal{L}_{\text{SFT}} = \lambda_{\text{ce}}\mathcal{L}_{\text{ce}} + \lambda_L\mathcal{L}_L + \lambda_O\mathcal{L}_O + \lambda_V\mathcal{L}_V. \tag{8}$$

We use $\lambda_{\text{ce}} = 0.1$, $\lambda_L = 0.0005$, and $\lambda_O = \lambda_V = 0.5$. The smaller layout weight accounts for its dense raw-MSE target, whereas object and view losses operate on sparse L2-normalized 2048-dimensional features.

Replay-aware answer-level RL. After SFT, we run answer-level GRPO on verifiable targets. The reward is intentionally simple:

$$r = \mathbf{1}[\widehat{A} = A] + 0.05\,\mathbf{1}[\text{valid format}], \tag{9}$$

where the format term checks for valid `<think>` and `<answer>` structure, and latent/object/view special tokens are stripped before answer extraction. For each prompt x_i, GRPO samples G completions and computes group-normalized advantages $\hat{A}_i$ from their rewards. Let $c_{i,t} = (x_i, y_{i,<t})$ denote the context for completion token $y_{i,t}$. We define the policy importance ratio as $\omega_{i,t} = \pi_\theta(y_{i,t} \mid c_{i,t})/\pi_{\theta_{\text{old}}}(y_{i,t} \mid c_{i,t})$ and its clipped form as $\omega_{i,t}^{\text{clip}} = \text{clip}(\omega_{i,t}, 1-\epsilon, 1+\epsilon)$. The token-level objective is

$$\mathcal{J}_{\text{GRPO}} = \frac{1}{G}\sum_{i=1}^{G} \frac{1}{\sum_t m_{i,t}} \sum_t m_{i,t}\Big[\min\big(\omega_{i,t}\hat{A}_i, \omega_{i,t}^{\text{clip}}\hat{A}_i\big) - \beta D_{i,t}\Big]. \tag{10}$$

where π_θ and $\pi_{\theta_{\text{old}}}$ denote the current and rollout policies, ϵ is the clipping range, $D_{i,t}$ is the per-token KL term, and $m_{i,t}$ is our loss mask. A rollout first samples a complete hierarchical response. Scoring this response with an ordinary teacher-forced forward pass would treat each layout/object/view pad as a discrete token embedding and would therefore fail to reproduce the continuous latent states that conditioned subsequent tokens during hierarchical generation. We instead replay the same prompt–completion pair without gradients under each policy being evaluated. During replay, hierarchical pad positions are processed sequentially, and the continuous state produced for each pad replaces its ordinary token embedding. These replayed pad embeddings are then reinjected when computing token log probabilities under the current and reference policies, so each policy scores the sampled completion under its own hierarchical latent context. In the main setting, $m_{i,t} = 0$ for every token inside a layout, object, or view span, and $m_{i,t} = 1$ elsewhere. Thus answer reward updates the natural-language reasoning and final answer, but not the special spans themselves.

Training supervision vs. inference. During SFT, VGGT-derived layout, object, and view features have two roles: they provide latent-pad inputs and they define reconstruction targets. After training, both roles disappear. At test time, H3D-LVR receives only the VLM, input images, and prompt; it autoregressively generates the layout/object/view spans by itself. We do not run VGGT, provide precomputed layout/object/view features or bounding-box annotations, or use any external 3D tool during evaluation.

4 Experiments

4.1 Setup

Evaluation setup. We evaluate Qwen2.5-VL-7B-Instruct (Bai et al., 2025) on MindCube-Tiny (Yin et al., 2025), SPAR-Bench (Zhang et al., 2026), and MMSI-Bench (Yang et al., 2025a) against the baselines in Table 1. We use accuracy as the evaluation metric, and each H3D-LVR result in the main table is averaged over three runs. SFT jointly optimizes text generation and layout/object/view reconstruction with the learning rate of 10^{-4}, while RL uses hidden-state replay and full-span masking

Table 1: Benchmark comparison on MindCube-Tiny, SPAR-Bench, and MMSI-Bench.

Method	Params	MindCube-Tiny				SPAR-Bench			MMSI-Bench
		Rotation	Among	Around	Overall ↑	ObjRel OC-MV ↑	ObjRel OO-MV ↑	Overall ↑	Acc. ↑
Closed-source Models									
o3 (OpenAI, 2025)	–	–	–	–	–	–	–	–	41.0
Claude-4-Sonnet (Anthropic, 2025)	–	48.4	44.2	47.6	45.8	–	–	–	–
GPT-4o (Hurst et al., 2024)	–	32.7	40.2	29.2	36.2	58.0	46.0	52.3	30.3
Open-source and 3D Reasoning Models									
Qwen2-VL (Wang et al., 2024)	2B	28.0	21.1	15.7	21.1	34.0	25.2	29.8	21.4
Qwen2-VL (Wang et al., 2024)	7B	–	–	–	–	30.3	41.0	35.4	–
Qwen2.5-VL (Bai et al., 2025)	3B	37.4	33.3	30.3	33.4	–	–	–	26.5
Qwen2.5-VL (Bai et al., 2025)	7B	38.8	29.5	21.4	29.3	44.8	32.1	38.8	25.9
Qwen2.5-VL (Bai et al., 2025)	32B	38.5	39.5	37.0	38.7	49.2	44.6	47.0	27.7
InternVL2.5 (Team, 2024)	8B	–	–	–	–	51.8	41.6	47.0	28.7
InternVL3 (Zhu et al., 2025)	8B	26.0	37.8	32.2	34.2	–	–	–	25.7
LLaVA-OneVision (Li et al., 2024a)	7B	36.5	48.4	44.0	45.1	37.3	38.2	37.7	24.5
HATCH (Oi et al., 2026)	3B	31.0	50.2	65.6	50.2	–	–	–	27.0
SpaceR-7B (Ouyang et al., 2025)	7B	31.5	28.7	31.6	29.9	–	–	–	26.7
Video-R1 (Feng et al., 2025)	7B	30.0	38.5	44.0	38.2	–	–	–	28.0
3DThinker (Chen et al., 2025b)	7B	43.5	66.3	76.4	64.4	29.3	23.3	26.5	28.0
H3D-LVR-SFT	7B	**76.5**	67.3	79.2	71.9	75.8	69.8	72.9	27.7
H3D-LVR-RL	7B	76.0	**73.0**	**82.0**	**75.7**	**79.8**	**74.8**	**77.4**	**29.1**

at 5×10^{-6}. H3D-LVR checkpoints use greedy decoding and a common answer parser, except for the inference span ablation. At inference, we use neither VGGT nor external 3D tools and provide neither precomputed features nor bounding boxes. Further settings are provided in Appendix A.3.

4.2 Main Results

Table 1 compares H3D-LVR with representative closed-source models, open-source VLMs, and spatial-reasoning baselines. H3D-LVR-SFT improves the Qwen2.5-VL-7B backbone from 29.3% to 71.9% on MindCube-Tiny and from 38.8% to 72.9% on SPAR. The improvements cover viewpoint transformation and egocentric relations on MindCube, as well as object–camera and object–object relations on SPAR. At the same 7B scale, H3D-LVR-SFT also outperforms LLaVA-OneVision, SpaceR-7B, Video-R1, and the 3D-aware 3DThinker. Moreover, Qwen2.5-VL-32B remains below H3D-LVR-SFT on both benchmarks despite its larger capacity. Overall, these results support organizing 3D supervision into explicit layout, object, and view states for multi-view spatial reasoning.

Replay-aware RL further improves MindCube to 75.7%, SPAR to 77.4%, and MMSI to 29.1%. The MindCube gains are concentrated on among and around, suggesting improved relation selection and local spatial comparison, while the gains on both SPAR OC-MV and OO-MV show that RL is not restricted to one relation type. Although MMSI remains challenging, RL still improves over SFT, indicating that answer-level optimization complements the hierarchical latent states learned during supervised training.

4.3 SFT Component Ablation

Table 2 isolates the three hierarchical SFT components on MindCube-Tiny. All variants use the same general response schema and data pipeline, but one feature-supervision branch is removed. Removing layout supervision drops overall accuracy to 66.86%, showing that the scene-level latent state contributes directly to spatial reasoning. The drop is broad across all three subsets, which is consistent with the role of layout as a global scaffold that supports later object and view grounding.

Removing object supervision drops to 67.90%, with the largest drop on rotation. Rotation questions often require the model to track local entities while mentally changing viewpoint; without object-level feature reconstruction, the generated object spans become weaker anchors for those entities. Removing view supervision drops overall accuracy to 68.38%, with declines across all three subsets. Questions throughout MindCube may define spatial relations relative to a specified viewpoint or a change in camera orientation, so view-level supervision helps the model maintain the appropriate reference frame when interpreting these relations. The full model is best overall. The three levels therefore behave as complementary supervision signals: layout stabilizes the scene state, object spans bind local entities, and view spans keep camera references available to the decoder.

Table 2: SFT component ablation studies.

Setting	L	O	V	Rotation	Among	Around	Overall
w/o Layout	–	✓	✓	70.50	63.17	72.80	66.86
w/o Object	✓	–	✓	64.50	66.00	75.20	67.90
w/o View	✓	✓	–	63.50	65.83	78.40	68.38
Full H3D-LVR	✓	✓	✓	**76.50**	**67.33**	**79.20**	**71.90**

Table 3: RL ablation on MindCube-Tiny. All rows are initialized from the same SFT model.

Variant	Rotation	Among	Around	Overall
Answer only (10^{-5})	**79.00**	68.17	78.80	72.76
Replay, no mask (10^{-5})	77.00	71.00	76.80	73.52
Replay + mask (10^{-5})	78.50	71.00	81.20	74.86
Replay + mask (5×10^{-6})	76.00	**73.00**	**82.00**	**75.71**

4.4 Replay-Aware RL Ablation

Table 3 compares variants with the same SFT initialization and answer/format reward; their rollout, replay, span-masking, and learning-rate settings are summarized in the table. Answer-only RL reaches 72.76% overall, but its training trajectory bypasses the latent-state path used by the full model at inference; answer reward can refine textual decisions, yet it cannot optimize those decisions in the context created by generated layout/object/view states. Replay restores that context and raises overall accuracy to 73.52%.

Without span masking, however, the same sequence-level answer advantage is also assigned to special start/pad/end token probabilities. A discrete pad token identifies a latent slot, while its sample-specific meaning resides in a continuous state; answer reward provides no target for what that state should encode. Direct policy gradients on these positions can therefore alter span emission and induce drift from the feature-aligned semantics learned during SFT. Full-span masking removes this ill-posed credit assignment and improves overall accuracy to 74.86%. Lowering the learning rate further limits parameter drift under sparse answer reward, yielding the best overall result of 75.71%, with the largest additional gains on among and around. Thus, answer-level RL produces its strongest gains when log-probability computation follows the hierarchical latent path and the latent spans themselves are isolated from semantics-agnostic reward gradients.

4.5 Data Quality: BBox Threshold

Table 4 studies the object-box filtering threshold while keeping hierarchical supervision fixed. No filtering retains more samples but also includes small or visually unstable boxes, reducing the among score and overall accuracy to 66.86%. This behavior reflects the sensitivity of feature regression to localization quality.

BBox30 removes part of this noise and recovers overall accuracy to 70.67%, but remains below bbox50. BBox70 removes more small boxes, yet the resulting loss of relation coverage reduces accuracy to 69.43%. The main bbox50 setting is therefore not simply the strictest filter; it provides the best balance between target quality and relation coverage, reaching 71.90%.

4.6 Object Latent Semantics

We compare two checkpoints with the same architecture and response schema. Full includes object-feature reconstruction, whereas w/o Object disables the object reconstruction loss while retaining object spans, the object projector, and layout/view supervision. For each object span, the two projected slot outputs are averaged and ranked by cosine similarity against valid features from the same sample. Figure 3 shows that Full reaches 75.26% overall top-1 accuracy compared with 39.76% for w/o Object, with consistent improvements across rotation, among, and around.

Although retrieval is not perfectly accurate, the Full result indicates that the compact object states retain instance-specific continuous features. The w/o Object model still carries some identity information through the object phrase, visual tokens, and layout/view context. However, the 35.50-point gap shows that explicit object reconstruction substantially improves feature alignment rather than merely teaching the decoder to emit the object-span syntax.

Table 4: BBox-size filtering ablation studies.

Data Setting	Rotation	Among	Around	Overall
No filter	73.50	60.33	77.20	66.86
BBox30	72.50	66.67	78.80	70.67
BBox70	66.50	66.67	78.40	69.43
BBox50	**76.50**	**67.33**	**79.20**	**71.90**

Table 5: Inference-time span ablations.

Inference	L	O	V	Rotation	Among	Around	Overall
No layout	–	✓	✓	64.00	62.67	71.20	64.95
No object	✓	–	✓	70.00	64.17	78.00	68.57
No view	✓	✓	–	66.00	**67.50**	78.40	69.81
Full rollout	✓	✓	✓	**76.50**	67.33	**79.20**	**71.90**

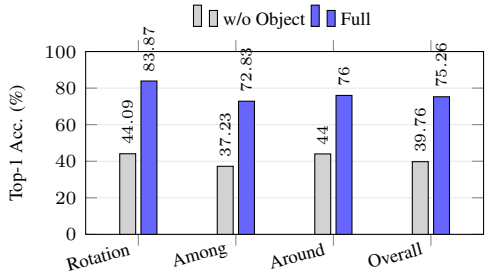


Figure 3: Sample-local object feature retrieval from object latent states. Object supervision improves top-1 retrieval by 39.78, 35.60, 32.00, and 35.50 points on rotation, among, around, and overall, respectively.

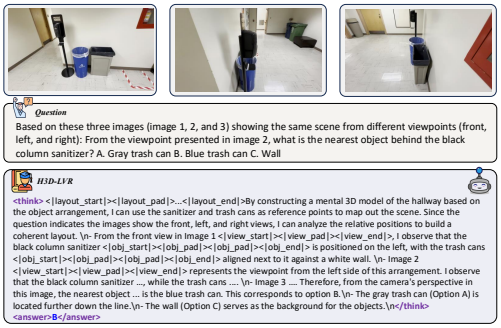


Figure 4: Example structured reasoning trace produced by H3D-LVR on a multi-view spatial question.

4.7 Inference Span Ablation

Table 5 evaluates the same full SFT checkpoint while making one generated span type unavailable at a time. For each ablation, the selected span-start logit is set to $-\infty$, while the remaining span types are generated normally and retain the fixed pad/end constraints. This setup isolates the inference-time contribution of each learned span type.

The full rollout achieves 71.90% overall accuracy. Suppressing layout causes the largest drop to 64.95%, while suppressing object or view spans yields 68.57% and 69.81%, respectively. No-object mainly affects rotation and among, whereas no-view produces its largest decrease on rotation. These results show that all three generated states contribute to reasoning: layout provides the global scene scaffold, while object and view spans preserve local entity and camera-frame information.

4.8 Qualitative Cases

Figure 4 presents a three-view hallway example that asks which object is immediately behind the black column sanitizer from the viewpoint of Image 2. Within the `<think>` block, the model first emits a layout span, then attaches an object span to the sanitizer and view spans to the referenced images. Its textual trace uses these states to align the front and side observations into a single spatial ordering, distinguish the blue bin from the farther gray trash. It consequently selects the blue trash can (B) while rejecting the gray trash can and wall. This case and those in Appendix A.4 illustrate how scene-, object-, and view-level latent states interleave with readable reasoning to resolve viewpoint-conditioned relations.

5 Conclusion

We presented H3D-LVR, a hierarchical 3D latent visual reasoning framework that represents scene layout, key objects, and input views as generated latent states aligned with training-only VGGT features. By combining feature-aligned SFT data with replay-aware, span-masked RL, the framework learns and preserves a multi-granular spatial state while requiring only the input images and prompt at inference. Across MindCube-Tiny, SPAR-Bench, and MMSI-Bench, H3D-LVR substantially improves the 7B backbone, while the ablations show that reliable feature targets, complementary layout/object/view supervision, and latent-state-preserving RL jointly account for the gains. These findings support hierarchical latent state tracking as a direct mechanism for integrating limited 2D observations into coherent multi-view 3D reasoning.

AI USE STATEMENT

LLM-based generative AI tools were used during manuscript preparation to revise prose, assist with LaTeX formatting, and review the clarity of technical explanations. We also used them to discuss possible experimental controls and assess whether the wording of our claims was supported by the reported evidence. All AI-assisted material was reviewed by the authors. All decisions about experimental design, implementation, analysis, interpretation, and reporting were made by the authors, who take responsibility for the final content of this paper.

REPRODUCIBILITY STATEMENT

We describe H3D-LVR's model architecture, hierarchical latent format, training stages, and evaluation procedures in the main text and appendix. Implementation details include the pretrained backbone, latent-span construction, feature-aligned data pipeline, crop-aware filtering, loss weights, optimization settings, computing environment, benchmark splits, decoding procedure, and controlled ablation protocols. The training-data construction process is detailed in Appendix A.2, and the training and evaluation configurations are reported in Appendix A.3. The authors take responsibility for all reported results and conclusions.

REFERENCES

Anthropic. Introducing claude 4. https://www.anthropic.com/news/claude-4, 2025.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *arXiv preprint arXiv:2308.12966*, 1(2):3, 2023.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

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*, pp. 14455–14465, 2024a.

Chao Chen, Zhixin Ma, Yongqi Li, Yupeng Hu, Yinwei Wei, Wenjie Li, and Liqiang Nie. Reasoning in the dark: Interleaved vision-text reasoning in latent space. *arXiv preprint arXiv:2510.12603*, 2025a.

Zhangquan Chen, Manyuan Zhang, Xinlei Yu, Xufang Luo, Mingze Sun, Zihao Pan, Yan Feng, Peng Pei, Xunliang Cai, and Ruqi Huang. Think with 3d: Geometric imagination grounded spatial reasoning from limited views. *arXiv preprint arXiv:2510.18632*, 2025b.

Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *Science China Information Sciences*, 67 (12):220101, 2024b.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 24185–24198, 2024c.

Google DeepMind. Gemini 2.5: Our most intelligent ai model. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/, 2025.

Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Benyou Wang, and Xiangyu Yue. Video-r1: Reinforcing video reasoning in mllms. *arXiv preprint arXiv:2503.21776*, 2025.

Chaoyou Fu, Yi-Fan Zhang, Shukang Yin, Bo Li, Xinyu Fang, Sirui Zhao, Haodong Duan, Xing Sun, Ziwei Liu, Liang Wang, et al. Mme-survey: A comprehensive survey on evaluation of multimodal llms. *arXiv preprint arXiv:2411.15296*, 2024.

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

Bangzheng Li, Ximeng Sun, Jiang Liu, Ze Wang, Jialian Wu, Xiaodong Yu, Hao Chen, Emad Barsoum, Muhao Chen, and Zicheng Liu. Latent visual reasoning. *ICLR*, 2026.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024a.

Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. *arXiv preprint arXiv:2407.07895*, 2024b.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22195–22206, 2024c.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2200–2209, 2021.

Moonshot AI. Kimi k2.5: Visual agentic intelligence. `https://www.kimi.com/blog/kimi-k2-5`, 2026.

Masanari Oi, Koki Maeda, Ryuto Koike, Daisuke Oba, Nakamasa Inoue, and Naoaki Okazaki. From correspondence to actions: Human-like multi-image spatial reasoning in multi-modal large language models. In *Forty-third International Conference on Machine Learning*, 2026.

OpenAI. Introducing o3 and o4 mini. `https://openai.com/index/introducing-o3-and-o4-mini/`, 2025.

Kun Ouyang, Yuanxin Liu, Haoning Wu, Yi Liu, Hao Zhou, Jie Zhou, Fandong Meng, and Xu Sun. Spacer: Reinforcing mllms in video spatial reasoning. *arXiv preprint arXiv:2504.01805*, 2025.

Yiming Qin, Bomin Wei, Jiaxin Ge, Konstantinos Kallidromitis, Stephanie Fu, Trevor Darrell, and XuDong Wang. Chain-of-visual-thought: Teaching vlms to see and think better with continuous visual tokens. *arXiv preprint arXiv:2511.19418*, 2025.

Arijit Ray, Jiafei Duan, Reuben Tan, Dina Bashkirova, Rose Hendrix, Kiana Ehsani, Aniruddha Kembhavi, Bryan A Plummer, Ranjay Krishna, Kuo-Hao Zeng, et al. Sat: Spatial aptitude training for multimodal language models. *arXiv e-prints*, pp. arXiv–2412, 2024.

Kexian Tang, Junyao Gao, Yanhong Zeng, Haodong Duan, Yanan Sun, Zhening Xing, Wenran Liu, Kaifeng Lyu, and Kai Chen. Lego-puzzles: How good are mllms at multi-step spatial reasoning? *arXiv preprint arXiv:2503.19990*, 2025.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

OpenGVLab Team. Internvl2: Better than the best—expanding performance boundaries of open-source multimodal models with the progressive scaling strategy, 2024.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5294–5306, 2025a.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Qixun Wang, Yang Shi, Yifei Wang, Yuanxing Zhang, Pengfei Wan, Kun Gai, Xianghua Ying, and Yisen Wang. Monet: Reasoning in latent visual space beyond images and language. *arXiv preprint arXiv:2511.21395*, 2025b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence. *arXiv preprint arXiv:2505.23747*, 2025.

Sihan Yang, Runsen Xu, Yiman Xie, Sizhe Yang, Mo Li, Jingli Lin, Chenming Zhu, Xiaochen Chen, Haodong Duan, Xiangyu Yue, et al. Mmsi-bench: A benchmark for multi-image spatial intelligence. *arXiv preprint arXiv:2505.23764*, 2025a.

Zeyuan Yang, Xueyang Yu, Delin Chen, Maohao Shen, and Chuang Gan. Machine mental imagery: Empower multimodal reasoning with latent visual tokens. *arXiv preprint arXiv:2506.17218*, 2025b.

Baiqiao Yin, Qineng Wang, Pingyue Zhang, Jianshu Zhang, Kangrui Wang, Zihan Wang, Jieyu Zhang, Keshigeyan Chandrasegaran, Han Liu, Ranjay Krishna, et al. Spatial mental modeling from limited views. In *Structural Priors for Vision Workshop at ICCV'25*, 2025.

Jiahui Zhang, Yurui Chen, Yueming Xu, Ze Huang, Jilin Mei, Chunhui Chen, Yanpeng Zhou, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, et al. From flatland to space: Teaching vision-language models to perceive and reason in 3d. *Advances in Neural Information Processing Systems*, 38, 2026.

Weichen Zhang, Zile Zhou, Xin Zeng, Xuchen Liu, Jianjie Fang, Chen Gao, Yong Li, Jinqiang Cui, Xinlei Chen, and Xiao-Ping Zhang. Open3d-vqa: A benchmark for comprehensive spatial reasoning with multimodal large language model in open space. *arXiv preprint arXiv:2503.11094*, 2025.

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*, 2023.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Yuchen Duan, Hao Tian, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

A APPENDIX

This appendix provides the complete data-generation prompt, the full version of the dynamic object-feature filtering algorithm, and further data-construction, training, evaluation, and computing-infrastructure details. It concludes with additional qualitative case studies.

A.1 STRUCTURED COT GENERATION PROMPT

This section presents the system prompt used to instruct Gemini-2.5-Pro to generate structured CoT responses for constructing the SFT corpus.

System Prompt

▷ **Role, Input and Goal.**

```
**Role:**
You are an advanced AI Spatial Reasoning Expert. Your task is to
    generate high-quality "Chain-of-Thought" (CoT) reasoning paths
    for constructing a training dataset.

**Input:**
1.  **Images:** Multi-view images of a scene.
2.  **Question:** A spatial reasoning question.
3.  **Ground Truth Answer:** The correct answer.

**Goal:**
Generate a reasoning trace wrapped in <think>...</think> tags
    followed by <answer>...</answer>. Please note that the content
    within the <think>...</think> tags must not exceed 1,000 tokens.
```

▷ **The "Global Layout & Single Registration" Protocol.**

```
1. The <layout> Section (The ONLY place for Tags):
* Must be First: The reasoning MUST start with the <layout> tag
    immediately inside <think>.
* Function: This section acts as a "Global Registry". You must
    identify and tag ALL relevant views and ALL key objects here.
* Content & Directions:
    * Describe the global 3D scene structure (room type, landmarks).
    * Direction Rule:
        * If cardinal directions (North/South) are clear or inferred
     from the context, use them.
        * If directions are unclear, use relative landmarks (e.g., "
    The wall with the whiteboard", "The opposite side containing the
     sofa").
    * Define the camera views relative to the scene.
    * List key objects and their locations.

2. Input Views (<view>):
* Definition: Only tag references to the actual input images
    provided.
* Insertion Rule: Wrap the phrase referring to a specific image with
     <view>...</view>.
    * Example: "...from the perspective of <view>Image 1</view>..."
* Constraint: NEVER use <view> for hypothetical/virtual views (e.g.,
     "if I turn right").

3. Key Objects (<key_obj>):
* Definition: Identify critical objects relevant to the question.
* Insertion Rule:
    * Wrap the key object with <key_obj>...</key_obj> immediately
    when it is first introduced.
* Constraint: Identify the truly critical objects, and constrain the
     number of key objects to no more than six.
```

```
4. The "Once-Only" Constraint (Crucial):
* Constraint: Every specific Key Object and Input View must be
    tagged EXACTLY ONCE in the entire output.
* Location: This single tag must occur inside the <layout>
    description.
* Forbidden: Do NOT use tags (<key_obj>, <view>) in the subsequent
    reasoning paragraphs. Once defined in the layout, refer to them
    as normal text.

5. Wrapping Rules (Descriptive & Precise):
* Objects (<key_obj>):
    * Wrap the Full Phrase: Do not just wrap the noun. Wrap the full
     descriptive noun phrase including colors, materials, sizes,
    shapes, states, positions, etc.
    * Bad: huge white <key_obj>curtain</key_obj>
    * Good: <key_obj>huge white curtain</key_obj>
    * Good: <key_obj>broken wooden chair</key_obj>
* Views (<view>):
    * Wrap the image reference.
    * Example: "...captured in <view>Image 1</view>..."
```

▷ **Few-Shot Examples.**

```
<think>
<layout> I have constructed the 3D layout. This is a small office
    room. One wall features a <key_obj>whiteboard</key_obj>.
    Rotating 90 degrees clockwise, the next wall has a <key_obj>
    black leather sofa</key_obj> and a <key_obj>large glass window</
    key_obj>. Opposite the whiteboard is a desk area. <view>Image
    1</view> captures the whiteboard. <view>Image 2</view> captures
    the sofa and window. </layout>

The question starts at the viewpoint of Image 2.
In this view, I am facing the black leather sofa and the window.
The instruction is to turn 90 degrees to the left.
Geometrically, turning 90 degrees left from the sofa wall directs me
     to face the whiteboard wall, which corresponds to Image 1.
The question asks: "what is to my right?"
From this new orientation (facing the whiteboard), my "Right" side
    would be the direction I just turned from.
Looking back at the global layout, the wall to the right of the
    whiteboard contains the black leather sofa and the window.
</think>
<answer>A</answer>
```

A.2 Feature-Aligned Data Construction Details

Training-data sampling. We construct the SFT corpus by randomly sampling examples from the training sets of MindCube and SPAR.

Text and box alignment. Gemini-2.5-Pro produces a structured response in which the global layout, key-object phrases, and input-view references are marked with `<layout>`, `<key_obj>`, and `<view>`. For every key-object phrase, Kimi-K2.5 is queried independently on every input image. Each query returns whether the target is visible and, when present, a tight normalized box. The object phrase, image index, box record, and VGGT feature file are retained under the same sample identifier so that the collator can match every generated object/view span to its intended feature.

Coordinate mapping and crop-aware filtering. The normalized box is first mapped to the resized image coordinates used by the data pipeline and then through the same width-resize and center-crop transformation used by VGGT. Let the resulting box lose a fraction $\gamma_{i,k}$ of its area in view i. The main text defines the view-count-dependent threshold $\tau(N_k)$. A view-specific object feature is

retained only when $\gamma_{i,k} \leq \tau(N_k)$. Valid boxes are mapped to the VGGT patch grid, and the covered spatial features are average-pooled. Features from the surviving views are then averaged to form the target for object anchor k. The main data setting additionally requires a minimum resized box side of 50 pixels. If no view survives, the phrase remains in the textual CoT but is not assigned an object latent span or reconstruction target.

Complete filtering algorithm. The following pseudocode gives the complete dynamic filtering procedure described in the main paper.

Algorithm 1 Dynamic View-Cropping Filtering and Cross-View Feature Aggregation

Require: Multi-view images $\mathcal{I} = \{I_1, \ldots, I_N\}$, VGGT layout feature maps $\mathbf{F}_L$, and key-object anchors $\mathcal{O}$
Ensure: Aggregated local 3D targets $\{\mathbf{F}_O^k\}$ for valid objects

for each object anchor $o_k \in \mathcal{O}$ **do**
  $N_k \leftarrow$ number of views with a matched box for o_k
  if $N_k \geq 3$ **then**
    $\tau_k \leftarrow 0.35$
  else if $N_k = 2$ **then**
    $\tau_k \leftarrow 0.25$
  else
    $\tau_k \leftarrow 0.20$
  end if
  $\mathcal{V}_k \leftarrow \emptyset$ {valid-view set}
  $\mathbf{f}_k^{\text{sum}} \leftarrow \mathbf{0} \in \mathbb{R}^C$
  for each view I_i with a matched box $B_{i,k}$ **do**
    Map $B_{i,k}$ through the VGGT resize and center crop
    Compute the cropped-area ratio $\gamma_{i,k}$
    if $\gamma_{i,k} \leq \tau_k$ and the resized minimum box side is at least 50 pixels **then**
      $\mathbf{f}_{i,k} \leftarrow \text{AveragePooling}(\text{RoIAlign}(\mathbf{F}_L[i], B_{i,k}))$
      $\mathbf{f}_k^{\text{sum}} \leftarrow \mathbf{f}_k^{\text{sum}} + \mathbf{f}_{i,k}$
      $\mathcal{V}_k \leftarrow \mathcal{V}_k \cup \{i\}$
    end if
  end for
  if $|\mathcal{V}_k| > 0$ **then**
    $\mathbf{F}_O^k \leftarrow |\mathcal{V}_k|^{-1} \mathbf{f}_k^{\text{sum}}$
    Retain the <key_obj> anchor and its object target
  else
    Strip the <key_obj> tag and retain the phrase as text only
  end if
end for
return $\{\mathbf{F}_O^k\}$

Final feature contract. Each training row stores the ordered input images, structured response, answer, and paths to three aligned feature files: dense VGGT layout features, object features indexed by descriptive key-object phrases, and view features indexed by explicit image references. The collator verifies the number and ordering of object/view spans before constructing the feature-conditioned inputs and reconstruction targets.

A.3 Training and Evaluation Details

Supervised fine-tuning. The backbone is Qwen2.5-VL-7B-Instruct. The main SFT model is trained for 10 epochs in bf16 with full-parameter DeepSpeed ZeRO-2, per-device batch size 1, gradient accumulation 1, weight decay 0.01, and seed 42. The response uses 12 layout pads, two pads per object span, and one pad per view span. Object/view spans are placed after their readable text anchors, object/view pad CE labels are masked, and the main loss weights are $\lambda_{\text{ce}} = 0.1$, $\lambda_L = 0.0005$, and $\lambda_O = \lambda_V = 0.5$.

Answer-level reinforcement learning. RL is initialized from the main SFT checkpoint and uses one training epoch, 16 GPUs, per-device batch size 4, eight generations per prompt, maximum completion length 1024, $\beta = 0.04$, temperature 0.9, top-$p = 1.0$, top-$k = 50$, and full-parameter optimization without LoRA. The main configuration uses strict hierarchical rollout, hidden-state

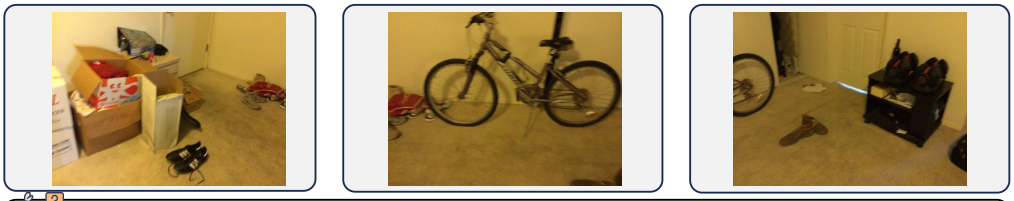
replay for current/reference log-probability computation, full-span policy/KL masking, and learning rate 5×10^{-6}.

Computing infrastructure. Training uses eight NVIDIA H20 GPUs for SFT and sixteen NVIDIA H20 GPUs for RL, with 96 GB of memory per GPU. The software environment uses Python 3.10 and CUDA 12.8, with PyTorch 2.7.0, DeepSpeed 0.15.3, Accelerate 1.10.1, Datasets 4.0.0, TRL 0.15.2, qwen-vl-utils 0.0.11, and FlashAttention 2 for RL.

Benchmarks and decoding. MindCube-Tiny contains 1,050 questions: 200 rotation, 600 among, and 250 around examples. The reported SPAR object-relation split contains 400 OC-MV and 361 OO-MV questions, and MMSI-Bench contains 1,000 multiple-choice questions. H3D-LVR evaluation uses greedy decoding, the strict hierarchical rollout processor, and a shared answer parser. No VGGT forward pass, precomputed layout/object/view feature, bounding box, or external 3D tool is used during evaluation. The inference-span ablation keeps the same checkpoint and decoding pipeline while setting the selected span-start logit to $-\infty$.

A.4 Additional Case Studies

This section presents additional case studies from MindCube-Bench. Specifically, visualizations of the 3D mental imagery constructed by our H3D-LVR model are shown in Figures 5, 6, and 7 for the Rotation, Among, and Around subtasks, respectively.



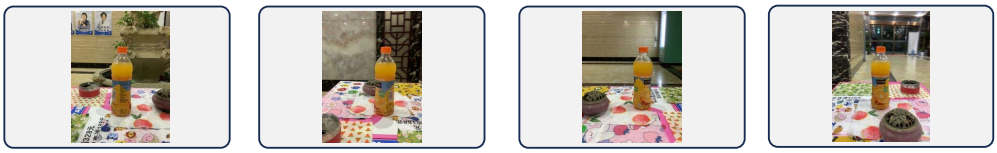
Question

These three images (image 1, 2, and 3) show the same scene from three different viewpoints. The image 2 was taken after turning the camera 90 degrees to the right (clockwise) from the position of image 1. For image 3, the camera was turned another 90 degrees right, so it's basically facing the opposite direction of image 1. Based on these three images: If I am standing at the same spot and facing the same direction as shown in image 1, what is to my behind? A. Black shoes shelf B. Bicycle C. A stack of boxes

H3D-LVR

<think> <|layout_start|><|layout_pad|>...<|layout_end|> By constructing a 3D model of the room using the camera's stationary position as a central reference point, I mapped out the surrounding space based on the specific rotational cues provided. Since the question describes a sequence of 90-degree turns, I analyzed the views to build a coherent layout of the environment. As seen in Image 1<|view_start|><|view_pad|><|view_end|>, I observed a stack of boxes <|obj_start|><|obj_pad|><|obj_pad|><|obj_end|> directly ahead. Turning 90 degrees to the right from the starting position reveals a bicycle <|obj_start|><|obj_pad|><|obj_pad|><|obj_end|>, which corresponds to the view in Image 2 <|view_start|><|view_pad|><|view_end|> . Turning another 90 degrees to the right results in a total rotation .... According to my mental layout, facing the direction of Image 1 means I am looking straight .... . The area directly behind me is a 180-degree rotation and Image 3 represents this view. Therefore, the object located in the view is the black shoes shelf. This matches option A. </think><answer>A</answer>

Figure 5: A representative case from the Rotation subtask of MindCube-Bench.



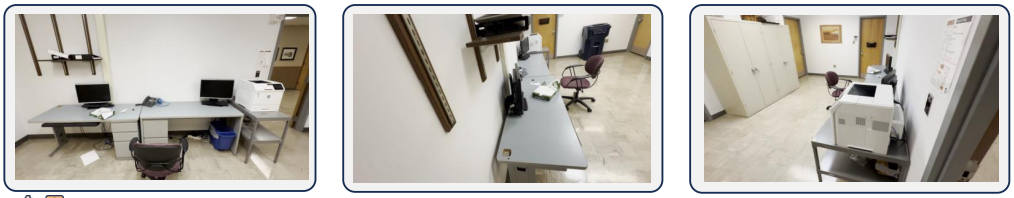
Question

Based on these four images (image 1, 2, 3, and 4) showing the drink bottle from different viewpoints (front, left, back, and right), with each camera aligned with room walls and partially capturing the surroundings: From the viewpoint presented in image 4, what is to the left of the drink bottle? A. Decorated wall B. Gate C. Light brown wall D. Stone fountain

H3D-LVR

<think> <|layout_start|><|layout_pad|>...<|layout_end|> By constructing a mental 3D model of the scene based on the object arrangement, I can use the central drink bottle <|obj_start|><|obj_pad|><|obj_pad|><|obj_end|> as reference points to map out the scene. Since the question indicates the images show the front, left, back, and right views, I can analyze the relative positions to build a coherent layout. – In Image 1 <|view_start|><|view_pad|><|view_end|> , I observed that the camera is facing a specific direction where the background features a stone fountain <|obj_start|><|obj_pad|><|obj_pad|><|obj_end|> . The red ashtray<|obj_start|><|obj_pad|><|obj_pad|><|obj_end|> is positioned to the left of the bottle.\n- In Image 4 <|view_start|><|view_pad|><|view_end|> , I observed .... According to my mental layout, the object located in that position corresponds to the background seen in the Image 3, which is the light brown wall. Therefore, the object to the left of the drink bottle is the light brown wall. This matches option C. </think><answer>C</answer>

Figure 6: A representative case from the Among subtask of MindCube-Bench.



Question

Based on these three images (image 1, 2, and 3) showing the same scene from different viewpoints (front, left, and right): In image 3, if I was positioned where the monitor, facing the same direction as it, what would be the nearest object to my right? A. Monitor B. Telephone

H3D-LVR

<think> <|layout_start|><|layout_pad|>...<|layout_end|> By constructing a 3D model of the office workspace using the desks as a reference point to map out the scene. Since the question indicates the images show the front, left, and right views, I can analyze the relative positions to build a coherent layout. In Image 1 <|view_start|><|view_pad|><|view_end|>, the arrangement from left to right consists of a left monitor<|obj_start|><|obj_pad|><|obj_pad|><|obj_end|> , a telephone<|obj_start|><|obj_pad|><|obj_pad|><|obj_end|> , a right monitor, and a printer on a stand at the far end. The left view looks down the line of desks from the left side, starting near the left monitor. - Image 3 <|view_start|><|view_pad|><|view_end|> depicts the perspective from the right side of the room, near the printer. Therefore, the monitor in Image 3 corresponds to the right-most monitor .... Tracing the linear arrangement along the desk, the object situated to my right is the telephone, followed further down by the second monitor. This matches option B.</think><answer>B</answer>

Figure 7: A representative case from the Around subtask of MindCube-Bench.