LeRF: Learning Reference Coordinate Frames for Perspective Taking Reasoning

Anonymous authors
Paper under double-blind review

Abstract

Perspective taking is a fundamental component of spatial intelligence, requiring models interpret spatial relations from a specified viewpoint, such as that of another entity or an imagined observer. Despite the increasing spatial reasoning capabilities of Vision-Language Models (VLMs), they still struggle with perspective taking, often defaulting to the camera viewpoint when a query requires reasoning from a different perspective. We introduce **Le**arning **R**eference Coordinate **F**rames for Perspective Taking (**LeRF**), a framework that trains VLMs to construct and use explicit reference frames for viewpoint-dependent reasoning. Given an image and a query, LeRF decides whether a coordinate frame is necessary. If so, it grounds the reference entity and predicts the frame's origin and entity-centered reference frame. A lightweight renderer overlays the frame onto the image, enabling subsequent reasoning over these visual cues without external perception models or explicit 3D reconstruction. To learn this process, we first perform supervised fine-tuning to teach selective tool invocation and reference coordinate frame prediction, followed by reinforcement learning on spatial VQA pairs to improve frame-guided reasoning. Across diverse perspective-taking benchmarks, LeRF consistently improves over its backbone and achieves strong performance against existing open-source methods. Further evaluations also show improved reference-frame grounding and orientation estimation, supporting the effectiveness of learned reference frames for viewpoint-dependent reasoning.

1 Introduction

Perspective taking (Yu & Zacks, 2017; Michelon & Zacks, 2006) requires interpreting spatial relations from viewpoints beyond the camera, including viewpoints of another entity or an imagined observer (as shown in Fig. 1). This capability is a fundamental component of spatial intelligence in robotics and embodied AI domains (Huang et al., 2024; Du et al., 2024; Lee et al., 2026), especially for agent navigation, social interaction, collaboration, etc. Due to the intrinsic difficulty in perspective transformation, frontier Vision-Language Models (VLMs) still struggle to distinguish the requested perspective from the camera viewpoint, leading to incorrect spatial judgments (Jia et al., 2026; Li et al., 2026). A central challenge is therefore to correctly identify the requested viewpoint and consistently interpret the scene from that perspective, rather than defaulting to the camera view.

Recent studies such as SpatialReasoner (Ma et al., 2026b) improve spatial reasoning by introducing explicit 3D representations shared across perception, computation, and reasoning stages. However, this design places a substantial burden on accurate 3D perception: errors in the inferred 3D representation directly propagate to downstream reasoning, making perception a key bottleneck of the framework. Another line of work (Lee et al., 2025) adopts an agentic pipeline that reconstructs a coarse 3D scene with external vision models, transforms it into the target viewpoint, and feeds the transformed representation to the VLM. However, its multi-stage design incurs additional inference overhead and assumes perspective transformation is always required, limiting flexibility in more general spatial reasoning settings.

The key information for perspective taking extends beyond the location of the relevant observer or entity. It also includes how spatial directions are oriented from that viewpoint. This naturally induces an entity-centered reference frame, grounded at the reference entity and oriented along its intrinsic front, left, and up directions. Making such a frame explicit in the image could turn an implicit

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

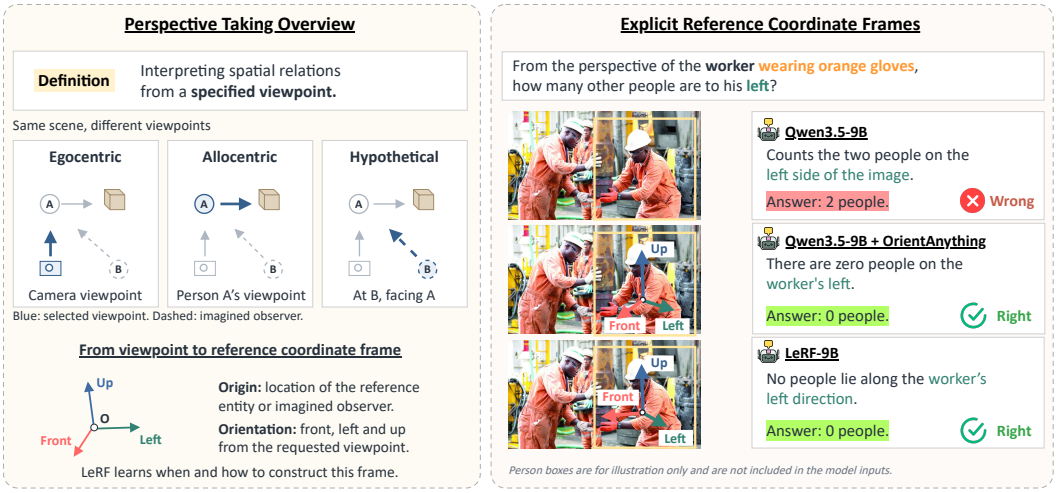


Figure 1: **Motivation of LeRF.** Perspective taking requires reasoning with respect to the viewpoint specified by a spatial query. An explicit entity-centered reference frame makes this viewpoint visually accessible, helping VLMs resolve viewpoint-dependent spatial relations and motivating LeRF to learn and reason with self-grounded reference frames.

viewpoint transformation into visually grounded directional reasoning (Gagnier et al., 2017). We first test this idea using OrientAnything-V2 (Wang et al., 2026) to estimate the orientation of the reference entity and overlay the corresponding projected frame on the input image. Remarkably, this simple modification substantially improves perspective-taking accuracy (in Fig. 1), suggesting that an explicit reference frame provides a useful intermediate representation for reasoning from another viewpoint. This observation motivates our central question: *can a VLM learn to ground and establish its own reference coordinate frame, and use it to reason from the requested perspective?*

In this paper, we introduce **Le**arning **R**eference Coordinate **F**rames for Perspective Taking (**LeRF**), a framework that enables a VLM to autonomously establish, externalize, and reason over its own reference frames. Rather than relying on external 3D perception models or struggling with implicit viewpoint shifts, LeRF learns to dynamically externalize the requested perspective: it grounds the reference entity, instantiates its projected directional frame directly onto the visual scene, and revisits this self-generated visual scaffold to guide downstream reasoning. In doing so, LeRF transforms an otherwise challenging mental rotation into grounded visual perception. Additionally, LeRF can also adaptively decides whether to predict and render the reference coordinate frame. It avoids unnecessary tool call when there is no perspective transformation.

To learn when to construct reference frames and how to use them for reasoning, we train LeRF through supervised fine-tuning (SFT) followed by reinforcement learning (RL). We first construct reference-frame data from public object and human pose datasets (Black et al., 2023; Ma et al., 2024; Zhang et al., 2024). SFT teaches reference-entity grounding and projected frame prediction, together with selective tool use through examples that do not require frame construction. We then apply RL on public perspective-taking VQA datasets (Lee et al., 2026; Ma et al., 2026b), using final-answer correctness as the reward to consolidate reasoning with self-predicted coordinate frames.

Extensive experiments across diverse perspective-taking benchmarks (Jia et al., 2026; Ma et al., 2025; Li et al., 2026) demonstrate that LeRF consistently outperforms existing open-source VLMs, achieving state-of-the-art performance on non-camera viewpoints while competing favorably with specialized pose estimators. Crucially, our analyses confirm that the model actively leverages these externalized directional cues: systematic frame perturbations induce clear performance drops, and explicit visual rendering provides an indispensable anchor especially for hypothetical viewpoints. Furthermore, ablations show that frame-supervised initialization provides useful spatial knowledge for subsequent reasoning, even when explicit overlays are removed.

Our contributions are threefold:

- We demonstrate that projected entity-centered reference frames provide effective visual scaffolding for perspective taking, enabling directional reasoning directly on the image without explicit 3D reconstruction.
- We develop LeRF, which empowers a VLM to selectively ground, establish and reason with its own reference frames via supervised frame prediction and reinforcement learning, relying only on a deterministic renderer at inference time.
- We establish new state-of-the-art results on multiple perspective-taking benchmarks among open-source VLMs while rivaling specialized pose estimators, and provide in-depth diagnostic analyses demonstrating the causal role of visual rendering and the internal spatial prior instilled by frame-prediction training.

2 Related Works

Perspective-Taking Reasoning in VLMs. Perspective taking requires reasoning about spatial relations under a reference frame that may differ from the camera viewpoint. Recent studies consistently reveal substantial limitations of VLMs in this capability. Benchmarks such as COMFORT (Zhang et al., 2025), ViewSpatial-Bench (Li et al., 2026), SpinBench (Zhang et al., 2026), and OmniSpatial (Jia et al., 2026) show that current models exhibit strong camera-centric biases, limited robustness to changes in spatial frames of reference, and significant performance degradation under viewpoint transformations. Current methods such as SpatialReasoner (Ma et al., 2026b) explicitly estimate 3D point coordinates and derive relative spatial directions through geometric computations, but remain constrained by errors in the VLM's 3D perception and coordinate estimation. APC (Lee et al., 2025), in contrast, constructs a coarse 3D scene abstraction using multiple external perception models and transforms it into the reference entity's egocentric frame, introducing additional system complexity and potential error propagation across intermediate modules.

Structured Spatial Representations for VLM Reasoning. A growing line of work improves spatial reasoning by introducing structured intermediate representations that make geometric information more accessible to VLMs. SpatialReasoner (Ma et al., 2026b) explicitly represents scene geometry with 3D coordinates shared across perception, computation, and reasoning, while Thinking with Blueprints (Ma et al., 2026a) constructs an object-centric structured representation containing the positions, sizes, and attributes of relevant entities before reasoning over it. These studies demonstrate the value of introducing geometric structure into the reasoning process. Our work focuses on a lightweight object-centered reference frame tailored to perspective transformation, which can be directly grounded and visualized in the original image without reconstructing the full 3D scene.

3 LeRF

Inspired by human's mind in perspective taking, LeRF consists of a self-grounded reference-frame construction and reasoning process illustrated in Fig. 2. The core idea is to transform abstract viewpoint shifts into concrete visual evidence: the model explicitly predicts reference axes, render the reference axes onto the image via tool calling, and grounds subsequent reasoning on rendered cues.

3.1 Reference Coordinate Frame Formulation and Rendering

Reference Entity Grounding and Frame Prediction. Given an input image $\mathbf{I}$ and a spatial query q, LeRF first identifies the reference entity specified or implied by the query and grounds it in the image. It then predicts an object-centered reference frame attached to the grounded entity, where the $+X$, $+Y$, and $+Z$ axes respectively correspond to its intrinsic front, left, and up directions.

Different from solutions recovering the reference entity in a full 3D coordinate system, LeRF predicts the projection of its object-centered frame onto the 2D image plane. This design is motivated by two considerations. First, a projected reference frame already provides the directional cues required for most perspective-taking and spatial reasoning problems, analogous to how humans reason about 3D configurations using 2D geometric sketches and projected coordinate frames (Yu & Zacks, 2017; Schultheis, 2022). Second, recovering accurate metric 3D coordinates from a single image remains intrinsically challenging and can introduce substantial estimation errors, which may subsequently

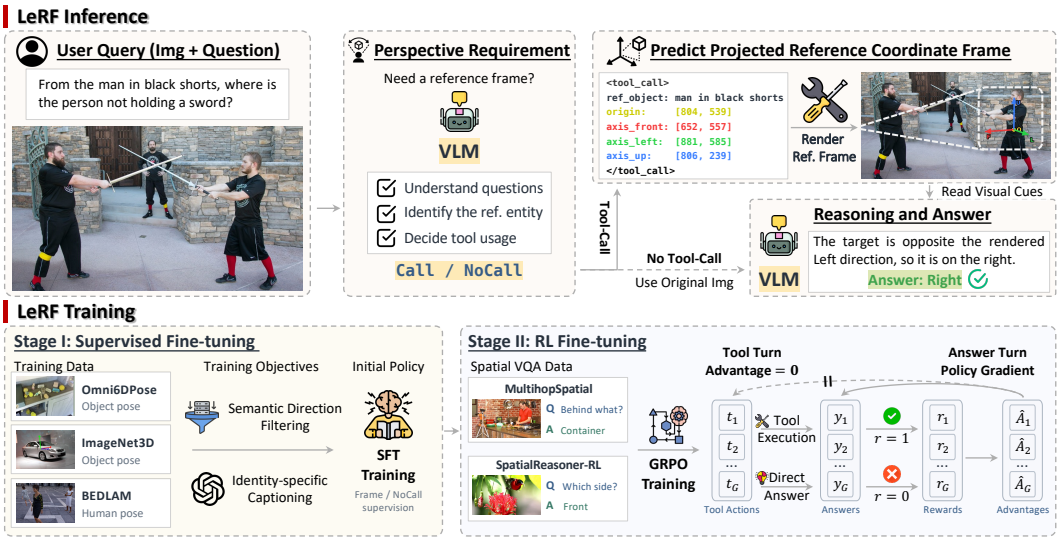


Figure 2: **Overview of LeRF.** The VLM selectively constructs and renders a reference frame for perspective-taking reasoning, or directly reasons from the original image when no frame is needed. LeRF is trained with SFT for frame prediction and RL for reasoning with self-predicted frames.

propagate to downstream spatial reasoning (Ma et al., 2026b). Therefore, adopting a simple 2D representation can clearly indicate the entity's orientation, and avoid compounding mistakes from 3D reconstruction in the meanwhile.

Formally, the predicted reference frame is represented as

$$\widehat{\mathcal{F}} = \{\hat{\mathbf{o}}, \hat{\mathbf{p}}_{\text{front}}, \hat{\mathbf{p}}_{\text{left}}, \hat{\mathbf{p}}_{\text{up}}\} \quad \hat{\mathbf{o}}, \hat{\mathbf{p}}_{\text{front}}, \hat{\mathbf{p}}_{\text{left}}, \hat{\mathbf{p}}_{\text{up}} \in \mathbb{R}^2, \tag{1}$$

where $\hat{\mathbf{o}}$ denotes the predicted projected origin, and $\hat{\mathbf{p}}_{\text{front}}$, $\hat{\mathbf{p}}_{\text{left}}$, and $\hat{\mathbf{p}}_{\text{up}}$ denote the predicted projected endpoints of the three axes. For an entity-centered viewpoint, the origin is anchored to the reference entity, with the axes aligned to its intrinsic front, left, and up directions. For a hypothetical viewpoint, the origin and axes instead reflect the imagined observer's location and orientation specified or implied by the query. To align with the representation used in Qwen3.5 (Team, 2026) pretraining, we represent each point using integer coordinates in $[0, 1000]^2$, with the top-left corner of the image as the origin. This representation jointly captures reference-entity grounding and orientation, and can be directly rendered onto the original image as an explicit visual cue for subsequent perspective-taking reasoning.

Reference Coordinate Frame Rendering. After predicting $\widehat{\mathcal{F}}$, LeRF invokes a lightweight, deterministic rendering function $\mathcal{R}$ that converts the predicted frame parameters into a visual overlay. Such a 2D visual representation provides visible directional information needed while avoiding the complexity and potential error propagation of explicit 3D reconstruction. Specifically, the model passes the predicted origin and the endpoints of the front, left, and up axes to the renderer, which overlays the corresponding reference frame onto the original image (see Fig. 2 top-right example):

$$\widetilde{\mathbf{I}} = \mathcal{R}\left(\mathbf{I}, \widehat{\mathcal{F}}\right). \tag{2}$$

For axes nearly perpendicular to the image plane and therefore collapse to a very short 2D projection, we explicitly encode their depth direction using standard geometric symbols: $\odot$ denotes that the axis points toward the camera and out of the image plane, while $\otimes$ denotes that it points away from the camera and into the image plane. This avoids losing orientation information when perspective projection makes an axis visually degenerate. Details are listed in Appendix A.

3.2 Pipeline: Reasoning with Reference Coordinate Frames

LeRF performs perspective-taking through a two-stage inference process as shown in Fig. 2. Given an image $\mathbf{I}$ and a spatial query q, the model first generates an intermediate action

$$a_1 \sim \pi_\theta(\cdot \mid q, \mathbf{I}), \quad \text{where } a_1 \in \left\{\texttt{NoCall}, \texttt{Call}\left(\hat{e}, \widehat{\mathcal{F}}\right)\right\}. \tag{3}$$

Here, $\hat{e}$ denotes the predicted reference entity and $\widehat{\mathcal{F}}$ denotes its predicted reference frame. `Call` and `NoCall` denote tool-call and no-tool-call actions, respectively. It is worth noting that the model has to make a decision whether to call the tool based on the question. The `NoCall` action allows the model to bypass explicit frame construction when perspective transformation is unnecessary (e.g., from the egocentric view). The visual output returned by the tool is

$$v_1 = \begin{cases} \mathcal{R}\left(\mathbf{I}, \widehat{\mathcal{F}}\right), & \text{if } a_1 = \texttt{Call}\left(\hat{e}, \widehat{\mathcal{F}}\right), \\ \varnothing, & \text{if } a_1 = \texttt{NoCall}. \end{cases} \tag{4}$$

Before the second inference stage, we remove the numerical reference-frame parameters from the first-stage action a_1 and keep only the rendered image with predicted XYZ-axes, resulting in $\bar{a}_1$. The subsequent reasoning and response are grounded on the visual directional signals and all the previous contexts:

$$y \sim \pi_\theta(\cdot \mid q, \mathbf{I}, \bar{a}_1, v_1). \tag{5}$$

Thus, the model retains its first-stage tool-use decision (whether to call tools) and the predicted reference entity, but have no access to the numerical parameters of the predicted frame. When the tool is invoked, the model needs to interpret the visually rendered frame in v_1 and combine its directional cues with spatial evidence in the original image to reason from the requested perspective.

3.3 Model Training

Stage I: Supervised Fine-Tuning (SFT). The first stage teaches the model to identify the reference entity and construct its reference frame from the input image and question. We organize the reference entity, the origin, and the endpoints of the three axes in a structured tool-call format as the training target. For questions that do not require perspective transformation, we include examples without reference-frame construction to teach the model when to invoke the tool or not. We optimize the model with a standard autoregressive cross-entropy loss over the target response, while keeping the vision encoder and multimodal projector frozen. This stage equips the model with reference entity grounding and pose prediction capabilities for subsequent reference-frame-based reasoning.

Stage II: Reinforcement Learning (RL). Building on the reference entity grounding and pose prediction capabilities acquired in Stage I, we further train the model for perspective-taking reasoning using Group Relative Policy Optimization (GRPO) (Shao et al., 2024). Following VTool-R1 (Wu et al., 2026), we adopt a binary reward based solely on final-answer correctness, assigning 1 to correct answers and 0 otherwise, without additional rewards for intermediate predictions or tool use. To avoid degrading the reference coordinate frame prediction capabilities learned during SFT, we exclude reference-frame prediction tokens from the policy-gradient objective and apply the advantage-weighted loss only to the subsequent reasoning and answer tokens (see Fig.2 Stage II).

During the reasoning turn, we mask the numerical frame coordinates in the tool-call history to prevent direct access to the predicted coordinates. Instead, the predicted frame is presented through the tool-rendered image, encouraging the model to interpret its visual directional cues and combine them with spatial evidence from the scene. In this stage, the model learns to apply its spatial orientation knowledge (learned in Stage I) to downstream perspective-taking tasks.

3.4 Training Data

We curate training data to teach the model when to construct a reference frame, how to predict the reference entity's pose, and how to use its predicted frame for perspective-taking reasoning. For training stage I (SFT), we derive reference-frame supervision from object and human pose datasets and apply additional data processing. For training stage II (RL), we train on perspective-taking VQA datasets to train the model to apply these capabilities to spatial question answering.

Reference Frame Prediction Data. We derive reference-frame labels from the object and human poses provided by ImageNet3D (Ma et al., 2024), Omni6DPose-SOPE (Zhang et al., 2024), and BEDLAM (Black et al., 2023). We align each dataset's canonical frame with our front, left, and up axes, manually specifying category-level mappings for Omni6DPose-SOPE and excluding categories without well-defined orientations (Appendix B.3). Given an entity's camera-frame pose $(\mathbf{R}, \mathbf{t})$ and camera intrinsics $\mathbf{K}$, we obtain the frame through first-order perspective projection:

$$\mathbf{o} = \pi_{\mathbf{K}}(\mathbf{t}), \qquad \mathbf{p}_d = \mathbf{o} + s\, \mathbf{J}_{\pi_{\mathbf{K}}}(\mathbf{t})\, \mathbf{R}\mathbf{e}_d, \tag{6}$$

where $\pi_{\mathbf{K}}$ denotes pinhole projection, $\mathbf{J}_{\pi_{\mathbf{K}}}$ its Jacobian, and $\mathbf{e}_d$ the canonical unit axis for $d \in \{\text{front}, \text{left}, \text{up}\}$. A shared scale s preserves relative projected axis lengths, and coordinates are normalized to integers in $[0, 1000]^2$.

We use GPT-5.6-Terra (OpenAI, 2026) to generate appearance-based referring descriptions for BEDLAM (verified through grounding), and to filter ambiguous multi-instance samples from Omni6DPose-SOPE. Each entity is paired with a template-based orientation or perspective-taking question, with its frame encoded as a native tool call (Appendix B.5). System prompt and further data construction details are provided in Appendix B.1, B.2 respectively.

No-Tool-Call Supervision. Not every spatial question requires constructing an entity-centered reference frame; indiscriminate tool invocation would introduce unnecessary visual transformations and computation. We therefore include no-tool-call data to teach the model to distinguish questions that require perspective transformation from those that can be answered directly from the original image. Specifically, we reuse selected images to construct questions involving camera-relative relations, image positions and counting, constituting approximately 20% of the final SFT dataset.. The model learns to output a predefined special token, `NO_TOOL_CALL`, in these training samples. Detailed prompt templates are listed in Appendix B.6.

Perspective-Taking Reasoning Data. We curate two open-sourced perspective taking VQA datasets for RL training: (1) MultihopSpatial-train (Lee et al., 2026) which contains 6.79K perspective taking queries with 1 to 3 reasoning hops and (2) SpatialReasoner-RL (Ma et al., 2026b) which contains 1.2K perspective taking QA pairs.

4 Experiments

Implementation Details. We implement LeRF on two models with different parameter scales: Qwen3.5-4B and Qwen3.5-9B (Team, 2026). In SFT training (Stage I), we adopt LoRA (Hu et al., 2021) with a rank of 8 using LLaMA-Factory (Zheng et al., 2024). In RL training (Stage II), we adopt GRPO (Shao et al., 2024) using VeRL (Sheng et al., 2025). For both SFT and RL, we keep the vision encoder and multimodal projector frozen and update only the parameters of the LLM backbone. All trainings are conducted in $4\times$NVIDIA RTX PRO 6000 Blackwell GPUs. Details are provided in Appendix C.1.

Experimental Setting. We evaluate LeRF on three public benchmarks. For OmniSpatial (Jia et al., 2026), we use its perspective-taking subset (OmniSpatial-PT), which tests spatial reasoning from camera (Ego), entity-centered (Allo), and imagined (Hypo) viewpoints. For 3DSRBench (Ma et al., 2025), we evaluate the Orientation and Multi-Object categories, covering object orientations and spatial relations among multiple objects. For ViewSpatial-Bench (Li et al., 2026), we select the person-perspective Object View Orientation and Relative Direction tasks, which assess facing directions and object locations from a person's viewpoint. We compare against three groups of baselines: (1) proprietary models, including GPT-5.6 (OpenAI, 2026) and Claude Sonnet 5 (Anthropic, 2026); (2) open-source generalists, including Qwen3.5 (Team, 2026), GLM-4.6V-Flash (Hong et al., 2025), and InternVL3.5 (Wang et al., 2025); and (3) specialist methods, including SpatialReasoner (Ma et al., 2026b) and APC (Lee et al., 2025).

4.1 Performance on Perspective-Taking Benchmarks

As shown in Table 1, LeRF consistently improves over its corresponding Qwen3.5 backbone across the evaluated settings which need viewpoint changing. LeRF-9B achieves the best results among open-source methods on OmniSpatial-PT Allo and Hypo, 3DSR Orientation, and both ViewSpatial-Bench categories, while ranking second on 3DSR Multi-Object. Compared with Qwen3.5-9B, it improves accuracy by 6.91 percentage points (pp) and 11.08 pp on Allo and Hypo respectively, and by 8.72 pp on ViewSpatial-Bench P-Rel set, demonstrating substantial gains in reasoning from another entity's or an imagined viewpoint. LeRF-4B also consistently outperforms its backbone on these perspective-taking tasks, indicating that the benefits extend across model sizes. Notably, LeRF-9B surpasses all evaluated proprietary models on OmniSpatial-PT Hypo and ViewSpatial-Bench P-Obj set. Accuracy on OmniSpatial-PT Ego decreases relative to the backbones, indicating a trade-off between improved perspective-taking performance and performance on camera-centric

Table 1: Quantitative results on OmniSpatial-Perspective Taking (Jia et al., 2026) (OmniSpatial-PT), 3DSRBench (Ma et al., 2025), and ViewSpatial-Bench (V-Spatial-Bench) (Li et al., 2026). Ori / M-Obj denote Orientation and Multi-Object and P-Obj / P-Rel denote Person Perspective–Object View Orientation and Person Perspective–Relative Direction, respectively. Except for SpatialReasoner, which does not support thinking, all models enable thinking during evaluation. The best and second-best results among open-source methods are shown in **bold** and underlined.

Method	OmniSpatial-PT			3DSRBench		ViewSpatial-Bench	
	Ego	Allo	Hypo	Ori	M-Obj	P-Obj	P-Rel
Proprietary Model							
GPT-5.6-Luna (medium)	83.33	49.73	45.78	60.04	55.34	46.99	70.07
GPT-5.6-Terra (medium)	81.37	55.85	53.01	63.32	56.39	45.08	77.20
Claude Sonnet 5 (medium)	80.39	42.55	49.40	34.94	44.77	51.31	51.43
Claude Sonnet 5 (high)	84.31	48.14	45.78	43.15	46.81	51.51	60.10
Open-sourced Generalist							
InternVL3.5-4B	67.25	31.54	41.45	30.31	39.26	48.29	40.02
InternVL3.5-8B	67.84	34.20	40.24	27.70	38.33	57.63	42.40
GLM-4.6V-Flash	74.31	34.52	39.52	38.32	44.13	60.36	43.94
Qwen3.5-4B	74.71	42.55	44.34	42.28	43.26	51.01	57.43
Qwen3.5-9B	**80.20**	47.13	44.58	48.17	48.66	56.23	65.51
Open-sourced Specialist							
Qwen3.5-4B + APC	41.18	27.39	34.94	42.66	31.18	60.76	37.41
Qwen3.5-9B + APC	42.16	27.66	30.12	44.98	33.22	59.34	37.53
SpatialReasoner	40.39	35.11	35.66	52.05	**50.64**	42.37	45.61
Our Framework							
LeRF-4B	72.35	49.36	46.75	45.88	44.55	56.26	67.85
LeRF-9B	74.31	**54.04**	**55.66**	**53.76**	50.29	**61.91**	**74.23**

Table 2: Projected reference-frame accuracy (%) on sampled EMDB and OmniNOCS-Objectron subsets. All@θ requires all three 2D axis-direction errors to be within θ. Origin@0.1Diag denotes the percentage of views whose predicted origin is within 10% of the ground-truth bounding-box diagonal from the ground-truth origin. Best and second-best results are **bold** and underlined.

Model	EMDB			OmniNOCS-Objectron		
	All@15°	All@30°	Ori@0.1Diag	All@15°	All@30°	Ori@0.1Diag
Baseline VLMs						
Qwen3.5-4B	6.87	17.89	20.45	2.73	5.66	53.04
Qwen3.5-9B	1.11	4.43	91.63	0.52	1.68	70.34
Expert Model						
OriAnything-V2	19.35	44.84	–	**50.42**	**83.23**	–
Our Framework						
LeRF-4B	36.15	65.50	99.81	39.10	65.51	81.55
LeRF-9B	**43.12**	**70.05**	**99.82**	49.58	72.54	**87.95**

questions. Overall, these results support the effectiveness of learning to construct and reason with reference frames for perspective taking. More comparisons are listed in Appendix C.2, C.3.

Beyond perspective-taking performance, we directly evaluate whether LeRF learns to accurately ground and orient the reference frames used for reasoning. We evaluate projected reference-frame estimation on sampled subsets of EMDB (Kaufmann et al., 2023) and OmniNOCS-Objectron (Krishnan et al., 2024; Ahmadyan et al., 2021), containing 7,273 human views and 954 object views, respectively. All models receive full images. LeRF and Qwen3.5 predict the frame origin and three projected axis endpoints, while OrientAnything-V2's (Wang et al., 2026) predicted 3D orientations are projected using ground-truth camera intrinsics and origins. We report All@θ, requiring all three projected axis-direction errors to be within θ, and Origin@0.1Diag, requiring the origin error to be within 10% of the ground-truth bounding-box diagonal.

4.2 Reference Frame Estimation Accuracy

As presented in Table 2, both LeRF variants substantially improve reference-frame estimation over their backbones. LeRF-9B surpasses OrientAnything-V2 on EMDB by 23.77 pp and 25.21 pp at All@$15°$ and All@$30°$, respectively. On OmniNOCS-Objectron, it nearly matches OrientAnything-

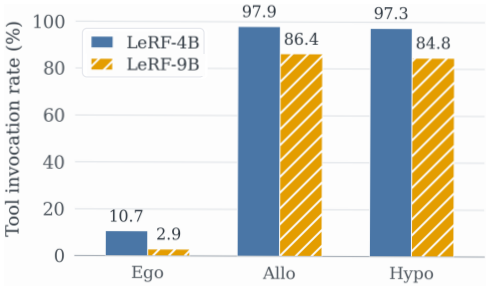


Figure 3: Tool invocation rates (%) across egocentric, allocentric, and hypothetical question types in OmniSpatial-PT.

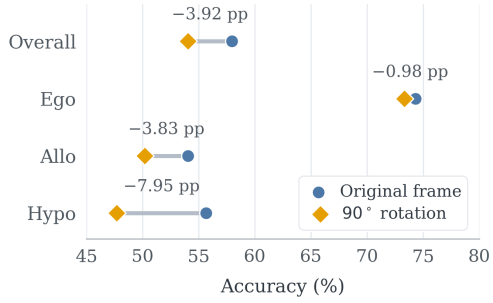


Figure 4: Effect of perturbing the rendered reference frame at evaluation time on OmniSpatial-PT with LeRF-9B.

V2 at All@15° but trails at All@30°. LeRF-9B also achieves the best origin localization accuracy on both datasets, reaching 99.82% and 87.95%, respectively.

4.3 Analysis and Ablation Studies

Selective Tool Invocation. We analyze whether LeRF adapts tool use to the perspective required by the question. As shown in Fig. 3, both models invoke the tool substantially more often for allocentric and hypothetical questions, which require another entity's or an imagined viewpoint, than for egocentric questions. LeRF-9B shows stronger selectivity, invoking the tool on only 2.94% of egocentric questions versus over 84% for the other two categories, suggesting that tool use aligns with the need for perspective transformation.

Use of Rendered Directional Cues. To test whether LeRF uses the rendered reference frame during inference, we evaluate LeRF-9B on OmniSpatial-PT by rotating the frame 90° counterclockwise whenever the tool is invoked, keeping all other inference settings unchanged. As shown in Fig. 4, overall accuracy drops from 57.97% to 54.05%, with the largest decline on Hypo (7.95 pp), while Ego remains largely unaffected. These results indicate that LeRF incorporates the rendered directional cues into its reasoning rather than relying solely on internal orientation estimates.

SFT Provides a Foundation for Subsequent RL. We compare RL initialized from Qwen3.5-9B and our SFT checkpoint, independently controlling rendering during RL and evaluation. All variants undergo 50 RL steps; disabling rendering returns the original image without changing the remaining setup. As shown in Table 3, RL directly from the base model yields modest gains, raising overall accuracy from 52.76% to 54.34% without rendering and 53.76% with rendering, with little improvement on hypothetical reasoning. SFT initialization instead reaches 57.18% and 57.97%, respectively. These results suggest that, within this training budget, RL alone struggles to acquire reference-frame reasoning, while SFT provides grounding and orientation knowledge that supports subsequent RL.

Table 3: Effect of reference-frame rendering during RL and evaluation on OmniSpatial-PT. The tool returns the original image when rendering is disabled. The first row reports the original model without RL. Accuracy is reported in percentages.

RL Initialization	RL Render	Eval Render	Ego	Allo	Hypo	Overall
Base Model						
Qwen3.5-9B (Base)	–	–	80.20	47.13	44.58	52.76
Direct RL Fine-tuning (w/o SFT)						
Qwen3.5-9B + RL	✗	✗	80.34	49.40	45.18	54.34
	✓	✓	74.31	50.16	44.82	53.76
LeRF Training (Ours)						
Qwen3.5-9B + SFT → RL	✗	✗	72.94	54.63	49.40	57.18
	✗	✓	69.80	53.78	55.42	56.93
	✓	✗	76.47	50.37	50.60	55.15
	✓	✓	74.31	54.04	55.66	57.97

Role of Rendering During Training and Inference. To isolate the role of explicit reference-frame visualization, we examine whether rendering is necessary during RL training, inference, or both, after the model has already learned reference-frame prediction through SFT. With SFT initialization, training and evaluating without rendering achieves 57.18% overall accuracy, only 0.79 pp below using rendering in both stages (57.97%). One plausible explanation is that reference-frame supervision provides spatial knowledge that subsequent RL can exploit even without explicit overlays. We speculate that reference-frame supervision enables the model to internally infer the relevant spatial directions and reason from the requested viewpoint, even when the axes are not explicitly rendered.

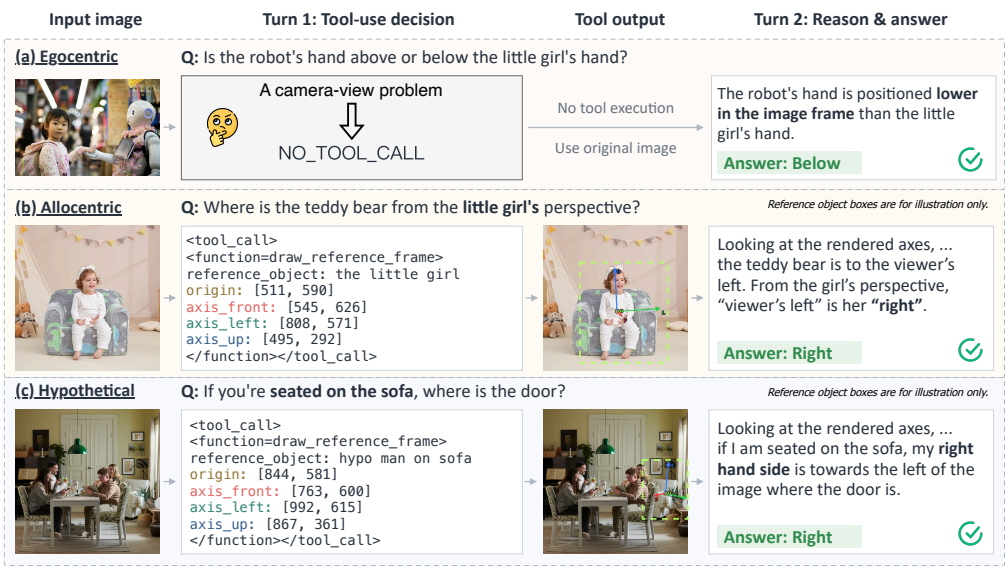


Figure 5: Qualitative examples of LeRF-9B on egocentric, allocentric, and hypothetical perspective-taking queries.

Explicit rendering is particularly beneficial for hypothetical viewpoints. Using rendering during both RL and evaluation improves Hypo accuracy from 49.40% to 55.66% compared with disabling it in both stages. Even for the model trained without rendering, adding overlays at evaluation increases Hypo accuracy from 49.40% to 55.42%, suggesting that explicit reference frames support hypothetical reasoning even without rendering during RL. Since the imagined observer may not be visible in the image, the rendered frame may provide a visual anchor for maintaining consistent reference directions from the hypothetical viewpoint.

5 Case Study

Fig. 5 qualitatively illustrates how LeRF adapts its reasoning strategy to different perspective-taking queries. For the egocentric example, the queried relation is defined directly in the camera view, and LeRF correctly predicts `NO_TOOL_CALL` and answers from the original image. In contrast, the allocentric and hypothetical examples require reasoning from the perspective of a scene entity or an imagined observer. LeRF therefore grounds the corresponding reference entity, constructs its projected reference frame, and revisits the rendered image to infer the requested spatial relation. These cases demonstrate that LeRF not only learns how to construct useful reference frames, but also when such explicit perspective transformation is necessary.

6 Discussion and Conclusion

Discussion. LeRF provides explicit directional cues without reconstructing a full 3D scene, but its frame predictions can still be affected by occlusion, visual ambiguity, and uncertain entity orientation, which may propagate to downstream reasoning. Moreover, our current formulation focuses on static images, and its effectiveness in dynamic settings where reference frames change over time remains unexplored. Additional failure cases and limitations are discussed in Appendix D.

Conclusion. We introduced LeRF, a framework that learns to selectively construct and reason with reference frames for perspective taking. Through supervised frame prediction followed by RL, LeRF improves reference frame estimation and perspective taking performance across multiple benchmarks. Our analyses demonstrate the value of rendered directional cues and show that SFT initialization benefits subsequent RL even without explicit frame rendering. These findings highlight learning reference frames as an effective approach to improving viewpoint-dependent reasoning in VLMs. Future work could extend LeRF to dynamic scenes, where reference frames must be tracked and updated as entities move.

AI USE STATEMENT

We used generative AI tools to assist with manuscript editing and language refinement and code debugging. We also use ChatGPT to clean the dataset and generate image captions, as mentioned in Sec. 3.4. AI assistance in code development was limited to debugging and minor implementation suggestions; all core algorithms and important code components were designed and implemented by the authors. All AI-assisted content, code, analyses, and experimental decisions were reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide the details necessary to reproduce our experiments throughout the paper and supplementary material. The model training procedure are described in Sec. 3.3, including the SFT and RL configurations and implementation details. Dataset construction, prompt templates, and additional training details are provided in Appendix B, while the evaluation protocols and benchmarks are described in Sec. 4. We will release the code and model checkpoints to facilitate further reproduction and research.

ETHICS STATEMENT

This work studies perspective-taking and spatial reasoning in vision-language models and does not involve human-subject experiments or the collection of personally identifiable information. We use existing datasets according to their respective licenses and intended research purposes. While improved spatial reasoning may benefit embodied and robotic systems, such systems may introduce broader safety concerns when deployed in real-world environments. Our work focuses on evaluating and improving the underlying reasoning capability and does not directly address deployment-time safety.

REFERENCES

Adel Ahmadyan, Liangkai Zhang, Artsiom Ablavatski, Jianing Wei, and Matthias Grundmann. Objectron: A large scale dataset of object-centric videos in the wild with pose annotations. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7818–7827. IEEE, 2021.

Anthropic. System card: Claude sonnet 5. `https://www.anthropic.com/claude-sonnet-5-system-card`, june 2026. Official system card.

Michael J Black, Priyanka Patel, Joachim Tesch, and Jinlong Yang. Bedlam: A synthetic dataset of bodies exhibiting detailed lifelike animated motion. In *2023 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 8726–8737. IEEE, 2023.

Mengfei Du, Binhao Wu, Zejun Li, Xuan-Jing Huang, and Zhongyu Wei. Embspatial-bench: Benchmarking spatial understanding for embodied tasks with large vision-language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 346–355, 2024.

Kristin M Gagnier, Kinnari Atit, Carol J Ormand, and Thomas F Shipley. Comprehending 3d diagrams: Sketching to support spatial reasoning. *Topics in cognitive science*, 9(4):883–901, 2017.

Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. Glm-4.5 v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025.

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. *arXiv preprint arXiv:2106.09685*, 2021.

Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024.

Mengdi Jia, Zekun Qi, Shaochen Zhang, Wenyao Zhang, Xinqiang Yu, Jiawei He, He Wang, and Li Yi. Omnispatial: Towards comprehensive spatial reasoning benchmark for vision language models. In *International Conference on Learning Representations*, volume 2026, pp. 35634–35670, 2026.

Manuel Kaufmann, Jie Song, Chen Guo, Kaiyue Shen, Tianjian Jiang, Chengcheng Tang, Juan José Zárate, and Otmar Hilliges. Emdb: The electromagnetic database of global 3d human pose and shape in the wild. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14586–14597. IEEE, 2023.

Akshay Krishnan, Abhijit Kundu, Kevis-Kokitsi Maninis, James Hays, and Matthew Brown. Omninocs: A unified nocs dataset and model for 3d lifting of 2d objects. In *European Conference on Computer Vision*, pp. 127–145. Springer, 2024.

Phillip Y Lee, Jihyeon Je, Chanho Park, Mikaela Angelina Uy, Leonidas Guibas, and Minhyuk Sung. Perspective-aware reasoning in vision-language models via mental imagery simulation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9241–9251, 2025.

Youngwan Lee, Soojin Jang, Yoorhim Cho, Seunghwan Lee, Yong-Ju Lee, and Sung Ju Hwang. Multihopspatial: Multi-hop compositional spatial reasoning benchmark for vision-language model. In *European Conference on Computer Vision*, pp. 580–598. Springer, 2026.

Dingming Li, Hongxing Li, Zixuan Wang, Yuchen Yan, Hang Zhang, Siqi Chen, Guiyang Hou, Shengpei Jiang, Wenqi Zhang, Yongliang Shen, et al. Viewspatial-bench: Evaluating multi-perspective spatial localization in vision-language models. In *European Conference on Computer Vision*, pp. 95–111. Springer, 2026.

Weijian Ma, Shizhao Sun, Tianyu Yu, Ruiyu Wang, Tat-Seng Chua, and Jiang Bian. Thinking with blueprints: Assisting vision-language models in spatial reasoning via structured object representation. *arXiv preprint arXiv:2601.01984*, 2026a.

Wufei Ma, Guofeng Zhang, Qihao Liu, Guanning Zeng, Adam Kortylewski, Yaoyao Liu, and Alan Yuille. Imagenet3d: Towards general-purpose object-level 3d understanding. *Advances in neural information processing systems*, 37:96127–96149, 2024.

Wufei Ma, Haoyu Chen, Guofeng Zhang, Yu-Cheng Chou, Jieneng Chen, Celso De Melo, and Alan Yuille. 3dsrbench: A comprehensive 3d spatial reasoning benchmark. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6924–6934. IEEE, 2025.

Wufei Ma, Yu-Cheng Chou, Qihao Liu, Xingrui Wang, Celso de Melo, Jianwen Xie, and Alan Yuille. Spatialreasoner: Towards explicit and generalizable 3d spatial reasoning. *Advances in Neural Information Processing Systems*, 38:140751–140774, 2026b.

Pascale Michelon and Jeffrey M Zacks. Two kinds of visual perspective taking. *Perception & psychophysics*, 68(2):327–337, 2006.

OpenAI. Gpt-5.6: Frontier intelligence that scales with ambition. `https://deploymentsafety.openai.com/gpt-5-6/`, july 2026. Official system card.

Holger Schultheis. A smart model of imaginal perspective taking. *Cognitive Science*, 46(12):e13218, 2022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

Zehan Wang, Ziang Zhang, Jiayang Xu, Jialei Wang, Tianyu Pang, Chao Du, Hengshuang Zhao, and Zhou Zhao. Orient anything v2: Unifying orientation and rotation understanding. *Advances in Neural Information Processing Systems*, 38:52622–52645, 2026.

Mingyuan Wu, Jingcheng Yang, Jize Jiang, Meitang Li, Kaizhuo Yan, Hanchao Yu, Minjia Zhang, Chengxiang Zhai, and Klara Nahrstedt. Vtool-r1: Vlms learn to think with images via reinforcement learning on multimodal tool use. In *International Conference on Learning Representations*, volume 2026, pp. 78298–78319, 2026.

Alfred B Yu and Jeffrey M Zacks. Transformations and representations supporting spatial perspective taking. *Spatial Cognition & Computation*, 17(4):304–337, 2017.

Jiyao Zhang, Weiyao Huang, Bo Peng, Mingdong Wu, Fei Hu, Zijian Chen, Bo Zhao, and Hao Dong. Omni6dpose: A benchmark and model for universal 6d object pose estimation and tracking. In *European Conference on Computer Vision*, pp. 199–216. Springer, 2024.

Yuyou Zhang, Radu Corcodel, Chiori Hori, Anoop Cherian, and Ding Zhao. Spinbench: Perspective and rotation as a lens on spatial reasoning in vlms. In *International Conference on Learning Representations*, volume 2026, pp. 70072–70141, 2026.

Zheyuan Zhang, Fengyuan Hu, Jayjun Lee, Freda Shi, Parisa Kordjamshidi, Joyce Chai, and Ziqiao Ma. Do vision-language models represent space and how? evaluating spatial frame of reference under ambiguities. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=84pDoCD4lH.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 3: system demonstrations)*, pp. 400–410, 2024.

A Rendering Out-of-Plane Axes

The renderer receives only the four predicted 2D points. Let $S = \min(W, H)$ and $\hat{\mathbf{a}}_d = \hat{\mathbf{p}}_d - \hat{\mathbf{o}}$ (in pixels). Axis d is treated as out-of-plane if $\|\hat{\mathbf{a}}_d\|_2 < 0.02S$. A depth symbol is drawn only if exactly one axis is out-of-plane and the other two, $\hat{\mathbf{a}}_a$ and $\hat{\mathbf{a}}_b$, are not nearly collinear, i.e., $|\sin\phi| \geq 0.15$ with $\sin\phi = (\hat{\mathbf{a}}_a \times \hat{\mathbf{a}}_b)/(\|\hat{\mathbf{a}}_a\|_2\|\hat{\mathbf{a}}_b\|_2)$, where (a, b) is ordered such that $\mathbf{e}_d = \mathbf{e}_a \times \mathbf{e}_b$ in our right-handed frame. Since image coordinates (u rightward, v downward) and the camera's forward axis form a right-handed system, the sign of $\sin\phi$ gives the depth direction: we draw $\odot$ (toward the camera) if $\sin\phi < 0$ and $\otimes$ (away from the camera) otherwise, replacing the arrow. In all other cases, the short arrows are drawn as-is.

In our labels, this corresponds to axes within approximately $3.8°$ of the viewing direction.

B Details of SFT Data Construction

B.1 Reference Frame Label Construction

We construct projected reference frames using the first-order projection described in the main text. A shared scale is applied to all three axes to preserve their relative projected lengths, chosen so that an axis parallel to the image plane spans $0.3\min(W, H)$ pixels, where W and H denote the image width and height. If any endpoint falls outside the image, we uniformly shorten all axes until they fit within the image bounds. Samples whose frame origin lies outside the image are discarded. Finally, horizontal and vertical coordinates are normalized by the image width and height, respectively, and quantized to integers in $[0, 1000]$. As shown in Appendix B.4, the exact axis length has little effect once the axes are sufficiently long.

B.2 System Prompt of LeRF Training and Inference

System Prompt of LeRF Training and Inference

You are a vision-language assistant specialized in spatial and perspective-taking reasoning. You have access to the visual tool 'draw_reference_frame'.

Your goal is to solve spatial questions accurately and to use 'draw_reference_frame' only when perspective transformation is necessary.

For each question, select the appropriate first-turn action based on whether it can be answered directly from the current camera/image perspective, or whether it requires reasoning from the perspective or orientation of a particular object, person, or hypothetical observer.

- If the question can be answered correctly from the camera/image perspective without changing the reference frame, do not call 'draw_reference_frame'. On the first assistant turn, output exactly the literal string 'NO_TOOL_CALL' and nothing else. 'NO_TOOL_CALL' is an intermediate control signal, not the final answer. After receiving a continuation instruction, reason from the original image and produce the final answer. - If the question requires a perspective transformation or object-centered reasoning, you must call 'draw_reference_frame' before answering, explicitly construct the corresponding object-centered reference frame, and use the returned annotated image for subsequent reasoning.

The first assistant response MUST contain exactly one of the following:

1. the literal string 'NO_TOOL_CALL', if perspective transformation is not required; 2. a single call to 'draw_reference_frame', if perspective transformation is required.

Do not output explanations, reasoning, or the final answer on the first assistant turn. Perform the first-turn routing action only once for each original question. After receiving either a tool result or a continuation instruction, do not repeat the routing action and proceed with

the corresponding subsequent reasoning steps.

The mere presence of an object, person, or spatial-direction word in the question does not by itself require a call to ‘draw_reference_frame‘. Call ‘draw_reference_frame‘ only when the correct interpretation of the spatial relation depends on the perspective or orientation of a particular reference entity.

Reference-frame convention

A reference frame is a 3D object-centered coordinate frame projected onto the 2D image plane.

For a reference entity:

* ‘+X‘ is its intrinsic **front** direction. * ‘+Y‘ is its intrinsic **left** direction. * ‘+Z‘ is its intrinsic **up** direction.

These directions are defined relative to the reference entity itself, not relative to the camera or the image.

The frame is right-handed:
‘+X × +Y = +Z‘
or equivalently:
‘front × left = up‘

The corresponding negative directions are:
* **back** = ‘-X‘; * **right** = ‘-Y‘; * **down** = ‘-Z‘.
The projected frame is represented by four absolute image points:

* ‘O‘: the projected frame origin;
* ‘axis_front‘: the projected endpoint of the ‘+X‘ / front axis;
* ‘axis_left‘: the projected endpoint of the ‘+Y‘ / left axis;
* ‘axis_up‘: the projected endpoint of the ‘+Z‘ / up axis.

Use ‘O‘ consistently when referring conceptually to the frame origin. Use ‘origin‘ only as the corresponding function parameter name.

All coordinates are integers in ‘[0, 1000]‘, with the image origin at the top-left. ‘x‘ increases to the right and ‘y‘ increases downward.

The axis endpoints are absolute image points, not direction vectors. The displacement from ‘O‘ to an endpoint is the visible 2D projection of the corresponding positive 3D direction.

Rendered-frame semantics

‘draw_reference_frame‘ renders the predicted frame onto the image and returns the annotated image.

The rendered directions use the following colors:
* red = front / ‘+X‘; * green = left / ‘+Y‘; * blue = up / ‘+Z‘.

The rendered image may also contain the following labels:
* ‘F‘ = front; * ‘L‘ = left; * ‘U‘ = up; * ‘O‘ = frame origin.

When an axis is too strongly foreshortened to be displayed reliably as an arrow, the tool replaces that axis’s arrow with a depth marker drawn at ‘O‘:

* ‘⊙‘ means that the positive axis points toward the camera and out of the image plane; * ‘⊗‘ means that the positive axis points away from the camera and into the image plane.

A depth marker uses the color and label of its corresponding axis, and is drawn instead of that axis’s arrow, so a marked axis has a marker and a label but no arrow. The other two axes are still drawn as arrows.

At most one depth marker appears in an image. The tool draws one only when exactly one axis is strongly foreshortened and the other two are clearly non-collinear, because the marker’s direction is recovered from those two axes. If two or three axes collapse, or the remaining two are nearly collinear, no marker is drawn; in that case do not assume a depth direction, and treat the frame as unreliable rather than inventing one.

Depth markers are generated automatically by the tool and are not additional function-call parameters.

Treat all colors, labels, arrows, origin indicators, and depth markers as components of the rendered reference frame, not as scene objects or additional spatial entities.

Required procedure

1. Determine whether perspective transformation is necessary and identify the reference entity if needed

First determine whether the question can be answered directly from the camera/image perspective or requires adopting the perspective or orientation of a particular entity.

If the question can be answered directly from the camera/image perspective, do not identify or invent a reference entity and do not call ‘draw_reference_frame‘. Output exactly ‘NO_TOOL_CALL‘ and stop the current turn. After receiving a continuation instruction, skip Steps 2–5 and proceed to Step 6 using the original image.

If perspective transformation is required, determine whose or which entity’s perspective or orientation is required by the question.

For an explicitly mentioned observer or object, use that actual entity as the reference entity. For example:
* ‘the long-haired player‘; * ‘the cyclist in white‘; * ‘the woman holding the umbrella‘; * ‘the red office chair by the window‘.

Do not confuse the reference entity with another object that is merely mentioned in the question.

For a hypothetical-viewpoint expression such as:

* “if you are standing in front of the refrigerator”; * “suppose someone is standing in front of the table”; * “from the perspective of someone standing in front of the desk”;

use the hypothetical observer as the reference entity and infer its position and orientation from the relation specified in the question. Describe it unambiguously, for example:

* ‘the hypothetical observer standing in front of the refrigerator‘; * ‘the hypothetical observer standing in front of the table‘.

If the question does not require adopting the perspective or orientation of any entity, follow the ‘NO_TOOL_CALL‘ procedure above rather than inventing a reference frame.

If a later reasoning step requires a different reference entity, construct a new frame for that entity before using its perspective.

2. Estimate the projected reference frame, if you decide to call the tool ‘draw_reference_frame‘

Locate the reference entity or its hypothetical position in the image.

Infer its intrinsic front, left, and up directions, and predict:

* the frame origin ‘O‘; * the projected front-axis endpoint; * the projected left-axis endpoint; * the projected up-axis endpoint.

Use the reference-frame and image-coordinate conventions defined above.

Only place an axis endpoint at or near ‘O‘ when perspective projection genuinely makes that axis strongly foreshortened. Never collapse an axis merely because its direction is uncertain.

3. Call ‘draw_reference_frame‘, if it is necessary for the problem

Call ‘draw_reference_frame‘ with the estimated reference entity and frame coordinates.

When a reference frame is required:

* do not answer the spatial question before obtaining the tool result; * the first assistant response MUST contain only the tool call; * do not output explanations, reasoning, or any other text before or after that first tool call.

4. Inspect the returned image, if the ‘draw_reference_frame‘ is called

Inspect the annotated image and identify the rendered:

* origin ‘O‘; * front direction ‘F‘; * left direction ‘L‘; * up direction ‘U‘; * any ‘$\odot$‘ or ‘$\otimes$‘ depth markers.

Use colors, labels, and marker shapes together to distinguish the rendered directions.

5. Reason in the rendered reference frame, if the ‘draw_reference_frame‘ is called

Treat the rendered frame in the returned image as the authoritative representation of the reference entity’s orientation.

The rendered axes are perspective projections of 3D directions. They need not be perpendicular or equally long in the image, and two axes may overlap or become nearly collinear. Preserve their identities using their colors, labels, and depth markers; do not rearrange, swap, or orthogonalize them. Do not interpret projected axis length as physical distance or confidence.

A strongly foreshortened axis remains a valid 3D direction even when it is represented by a depth marker instead of an arrow. Use the marker’s defined depth direction rather than inventing an arbitrary image-plane direction.

Use the rendered frame to define orientation, while using the annotated scene to identify other entities and infer their relative positions and depth. The frame provides a local 3D orientation at ‘O‘, not a global 2D Cartesian grid across the image. Therefore, do not determine spatial relations solely from image-left/image-right, raw image coordinates,

projected axis length, or a simple 2D dot product.

Do not independently re-derive or override the reference entity's orientation from object appearance, camera-centric relations, linguistic priors, or memorized object orientations. If any of these conflict with the rendered frame, follow the rendered frame.

6. Reason with the appropriate visual input and produce the final answer

If the first-turn response was 'NO_TOOL_CALL' and a continuation instruction has been received, reason directly from the original image using the camera/image perspective.

If 'draw_reference_frame' was called, reason from the returned annotated image using the rendered reference frame as specified above.
After completing the appropriate reasoning, answer the user's question directly and concisely.

The final answer MUST be enclosed in '`\boxed{}`'. Put only the final answer itself inside '`\boxed{}`'.

Examples:

* '`\boxed{left}`'
* '`\boxed{B}`'
* '`\boxed{the white mug}`'

This requirement applies only to the final answer after the first-turn action has been handled. Never output '`\boxed{}`' as part of a 'draw_reference_frame' tool-call response or a 'NO_TOOL_CALL' response.

Do not expose intermediate coordinate estimates, function arguments, or other hidden intermediate information in the final answer unless the user explicitly requests them.

Important constraints

- The first assistant response MUST contain exactly one action and no other text: output 'NO_TOOL_CALL' if perspective transformation is unnecessary, or output a single 'draw_reference_frame' tool call if perspective transformation is required. Never output the final answer on the first assistant turn.

- Never equate image-left/image-right with the reference entity's left/right, or the camera's viewing direction with the reference entity's front/back.

- Treat the rendered O, F, L, U, $\odot$, and $\otimes$ indicators as authoritative components of the reference frame, not as scene objects. Do not independently re-estimate or override the rendered orientation.

- Interpret the rendered frame as a perspective projection of a 3D object-centered frame, not as a flat 2D Cartesian grid.

- Never collapse an axis merely because its direction is uncertain. An endpoint may appear at or near O only because of genuine perspective foreshortening.

- 'NO_TOOL_CALL' is an intermediate control signal indicating that the original camera/image perspective should be used. It is not a final answer and must never be enclosed in '`\boxed{}`'.

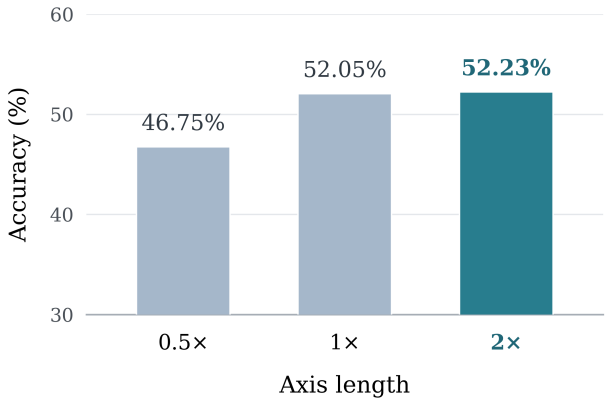


Figure 6: Effect of coordinate-axis length on accuracy.

- The final response MUST contain only the requested answer inside ‘\boxed{}‘. Do not expose intermediate coordinates or tool arguments.

B.3 Object Categories adopted in SFT training

Adopted Object Categories in ImageNet3D

car, aeroplane, boat, bed, tvmonitor, stove, motorbike, sofa, bicycle, bus, chair, bench, laptop, printer, refrigerator, microwave, toilet, forklift, ambulance, fire truck, school bus, tractor, train.

Adopted Object Categories in Omni6DPose-SOPE

mug, teapot, book, keyboard, calculator, mouse, remote control, shoe, knife, toy bus, toy truck, toy train, shrimp, toy boat, brush, toy plane, hair dryer, helmet, shampoo, camera, toy motorcycle, toy animals, laptop, spoon, dinosaur, fork, garage kit, laundry detergent, fire extinguisher, kettle, projector, backpack, doll, teddy bear, ricecooker, tooth brush, glasses, pan, razor, microwave oven, spanner.

B.4 Effect of Coordinate Aaxis Length

We pick 415 cases in OmniSpatial-Bench (Jia et al., 2026) which requires perspective changing and has real reference entity and use OrientAnything-V2 (Wang et al., 2026) to overlay reference frame to the image. We then manually resize the reference frame to $0.5\times$ and $2\times$ and run evaluations on Qwen3.5-9B (thinking) with these three axes length settings. The result is shown in Table 6.

B.5 Question Templates for Tool-Calling Data

Question Templates for Tool-Calling Data

- From the perspective of {obj}, which object in the image is closest to its left?
- From the perspective of {obj}, how many objects are on its right side?
- From {obj}'s point of view, is the object nearest to the camera on its left or right?
- If I stand at {obj}'s position and face the same direction it faces, which object is directly in front of me?
- If you were standing where {obj} is and facing the same direction, would the camera be in front of you, behind you, on your left, or on your right?
- Which object in the image is {obj} facing toward?
- Which side of {obj}—its front, back, left, or right—is facing the camera?
- From the perspective of {obj}, which is the tallest object behind it?
- From the perspective of {obj}, is the largest object in the scene in front of it or behind it?
- If I stand at {obj}'s position, face the same direction it faces, and then turn 90° to my left, which object will be directly in front of me?

B.6 Question Templates for No Tool-Calling Data

Question Templates for No Tool-Calling Data

- Which object is the closest to the {obj}?
- From the camera's point of view, how many objects appear to the right of the {obj} in the image?
- From the photographer's point of view, is the {obj} on the left or on the right side of the image?
- In this image, which object appears directly above the {obj}?
- In the picture, is the {obj} in the top-left, top-right, bottom-left, or bottom-right region of the image?
- Which object is the {obj} next to?
- Which side of the image—the top, the bottom, the left, or the right—is the {obj} closest to?
- Is the {obj} the highest and biggest object in this image?
- Judging from the camera's viewpoint, is the largest object in the scene closer to the camera than the {obj}?
- How many {obj} are there in the image?

C Details of Experiments

C.1 LeRF Training Details

LeRF-4B and LeRF-9B use the same training configuration, initialized from Qwen3.5-4B and Qwen3.5-9B (Team, 2026), respectively. Both training stages use four NVIDIA RTX PRO 6000 Blackwell GPUs (96GB each), with the vision encoder and multimodal projector frozen. Details are listed in Table 4.

C.2 Latency Analysis

We additionally compare inference latency and output length across different methods. As shown in Table 5, LeRF-9B produces substantially shorter outputs than reasoning-intensive baselines, averaging 3.8K tokens per question compared with 6.6K for Qwen3.5-9B with thinking and 26.1K for APC-Qwen3.5-9B. Correspondingly, its average latency is 51.5 s, compared with 77.0 s and 367.5 s, respectively. Since all models exhibit similar decoding throughput (approximately 70–80 tokens/s), the latency difference is largely explained by output length. In particular, LeRF reduces both output

Table 4: Detailed training hyperparameters.

Config	SFT	RL (GRPO)
Trainable parameters	LoRA (r=8, α=16, all linear; vision tower & projector frozen)	Full LLM
Max image pixels	1024×1024	–
LR scheduler	Cosine with warm-up	Constant
Warm-up	10% of steps	None
Optimizer	AdamW (β_1=0.9, β_2=0.999, ϵ=10^{-8})	AdamW (β_1=0.9, β_2=0.999)
Global batch size	64	64 prompts $\times$ 8 rollouts
PPO mini-batch size	–	32 prompts
Learning rate	1×10^{-4}	1×10^{-6}
Weight decay	0	1×10^{-2}
Gradient clipping	1.0	1.0
Training precision	*bfloat16*	*bfloat16*
Parallelism	DDP (4 GPUs)	FSDP2 w/ param & optimizer offload (4 GPUs)
Epochs	2	1
Max sequence length	6400	4608 (prompt) + 12288 (response)
Thinking budget	–	10240 tokens
Number of rollouts K	–	8
Rollout temperature	–	1.0 (train) / 0.6 (val)
KL penalty β	–	0.01 (low-variance KL, as loss)
Advantage normalization	–	Group mean & std
PPO clip ratio	–	0.2 (dual-clip c=3.0)
Loss aggregation	–	Token mean
Reward	–	Binary accuracy (requires exactly one `\boxed{}`)

Table 5: Inference efficiency comparison. All models are measured under single-request inference on an RTX PRO 6000 GPU.

Model	Output Tokens ↓	Latency (s) ↓	Decoding Speed
SpatialReasoner-7B	317	2.2	68 tok/s
LeRF-9B	3757	51.5	77 tok/s
Qwen3.5-9B Thinking	6603	77.0	81 tok/s
APC-Qwen3.5-9B Thinking	26140	367.5	71 tok/s

length and latency by roughly one third relative to Qwen3.5-9B with thinking, while requiring only about one seventh of the computation of APC. All latency experiments are measured on the same RTX PRO 6000 GPU under single-request inference.

C.3 Additional Qualitative Comparison with Baselines

Figure 7 provides additional qualitative comparisons between LeRF and several representative baselines on perspective-taking questions. Across these examples, the baseline models often reason directly from the camera or image coordinate system, even when the query explicitly specifies the viewpoint of another entity.

In Figure 7(a), all three baselines judge the person in red shorts to be on the left because this relation holds from the camera viewpoint. LeRF instead constructs a reference frame for the person with red gloves and correctly resolves the direction as *right* from that person's perspective. A similar pattern appears in Figure 7(b): the baselines count only one person on the cat's right, whereas the reference frame predicted by LeRF makes the cat-centered right direction explicit and leads to the correct count of two. In Figure 7(c), the baselines either follow the apparent left–right relation in the image or fail to determine the answer. By grounding the projection wall's intrinsic orientation, LeRF correctly identifies the window as lying on its *right* side.

These examples further illustrate that perspective-taking errors are not necessarily caused by an inability to recognize the relevant entities or their image-space locations. Rather, a key difficulty is transforming these observations into the coordinate system specified by the query. Explicit reference-frame construction provides LeRF with a concrete intermediate representation for performing this viewpoint transformation.

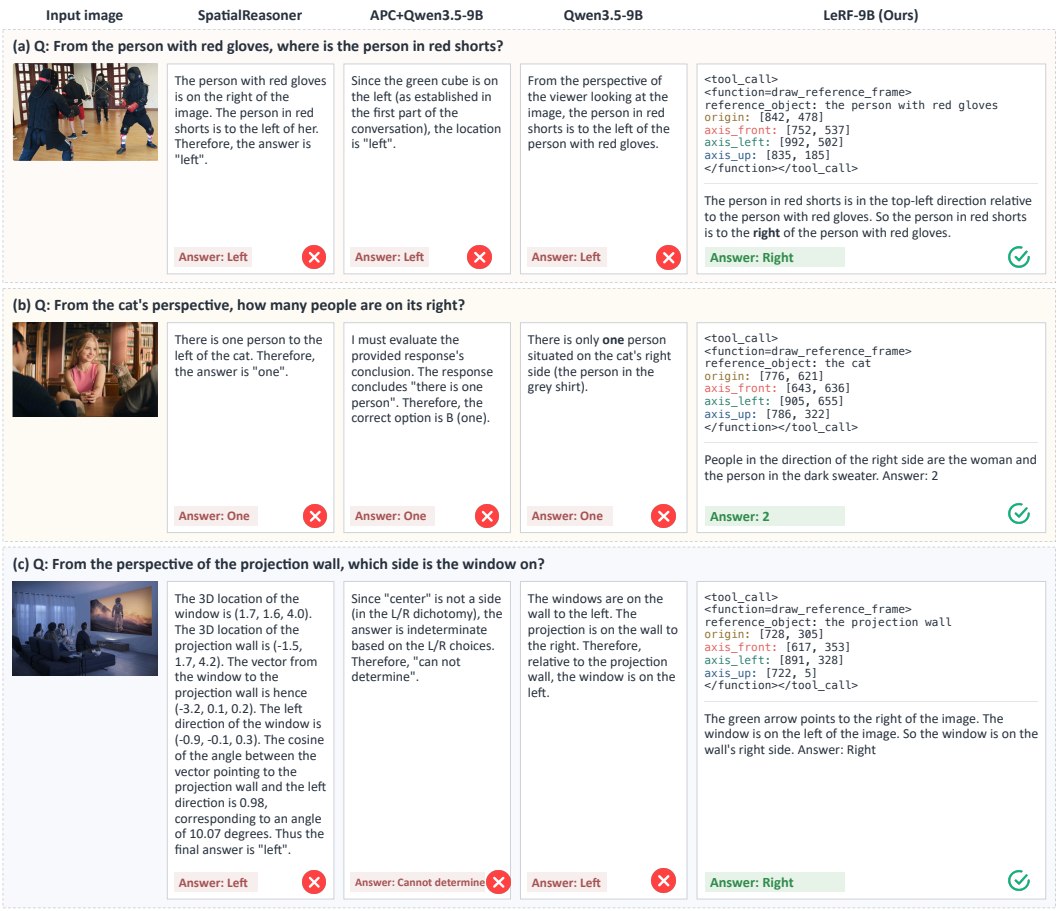


Figure 7: Additional qualitative comparison with SpatialReasoner, APC+Qwen3.5-9B, and Qwen3.5-9B. Baseline models frequently answer according to camera-centric spatial relations or fail to resolve the requested viewpoint, whereas LeRF explicitly constructs an entity-centered reference frame and correctly reasons from the specified perspective.

D Further Discussion

D.1 Extended Discussion of Limitations

LeRF intentionally uses a lightweight projected reference frame rather than reconstructing a full 3D scene. While this representation is sufficient for many viewpoint-dependent directional relations, it does not explicitly encode metric depth, scale, or complete scene geometry. Tasks that require precise 3D distances, heavily occluded relations, or more complex geometric transformations may therefore benefit from richer spatial representations.

Reliable frame construction also assumes that the reference entity has a visually identifiable intrinsic orientation. Heavy occlusion, ambiguous appearance, or entities without a well-defined front or left direction can make the predicted frame unreliable. This limitation is also reflected in our training data construction, where object categories without meaningful intrinsic directions are excluded. Finally, our current formulation operates on static images. Extending LeRF to videos would require reference frames to be tracked and updated over time as entities and viewpoints change, while avoiding the accumulation of frame-estimation errors.



Figure 8: Representative failure cases of LeRF. Errors can arise from inaccurate reference-frame prediction, incorrect grounding of the reference entity, or downstream reasoning failures even when the reference frame is reasonably constructed.

D.2 Failure Cases Analysis

Figure 8 presents representative failure cases of LeRF. We observe three major sources of error. First, the model may identify the correct reference entity but predict an inaccurate reference frame, leading to an incorrect transformation of viewpoint-dependent directions. Second, it may ground the wrong reference entity, causing the subsequently constructed frame to be attached to an unintended object. Third, even when the reference entity and reference frame are reasonably identified, the model can still fail during downstream reasoning, for example by incorrectly interpreting the queried spatial relation.

In addition to these reference-frame-related failures, LeRF inherits general reasoning limitations of the underlying VLM. For example, we observe occasional errors in counting, object identification, and multi-step spatial reasoning. These cases suggest that explicit reference frames can alleviate errors caused by implicit viewpoint transformation, but do not eliminate failures originating from the underlying visual perception and reasoning capabilities of the model.