PANOAGENT: EFFICIENTLY UNDERSTANDING IN 360° THROUGH ACTIVE INSPECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained vision-language models (VLMs) exhibit strong perception and reasoning capabilities on perspective images, but panoramic projection distortions limit the use of these capabilities in panoramic understanding. Existing adaptation methods primarily rely on fine-tuning with panoramic question–answer supervision or predefined workflows that convert panoramas into planar views for the model. We introduce PanoAgent, which adapts VLMs to panoramic understanding by learning how to observe. It jointly learns to select question-relevant panoramic transformations and generate answers, allowing the model to actively select observations suited to its existing visual capabilities. Specifically, the model uses panorama-specific tools to select local planar views or update the global reference heading as needed, with returned evidence guiding subsequent observation and reasoning. Trajectory demonstrations connect panoramic concepts to observation actions, while reinforcement learning further refines the policy through outcome and geometric feedback. Experiments across multiple panoramic benchmarks show that PanoAgent achieves leading overall performance and offers a better balance among understanding accuracy, data efficiency, and inference efficiency than existing panoramic adaptation methods. The dataset and trained models will be released publicly.

1 INTRODUCTION

360° panoramic images capture an observer's entire surroundings in a single frame, providing a comprehensive view of the scene and its spatial relationships. It has benefited various applications that require an understanding of what lies in different directions, such as AR/VR and embodied visual reasoning. Yet a panoramic image is useful only if a model can identify the relevant regions and reason about them from the appropriate reference frame. Enabling such reasoning (Yang et al., 2026; Tran et al., 2026) remains challenging for vision-language models (VLMs) developed primarily for perspective images (Zhu et al., 2025; Bai et al., 2025; Tong et al., 2024; Cai et al., 2026).

Existing adaptation methods mainly follow two strategies as shown in Figure 2: post-training a VLM to interpret equirectangular projection (ERP) images directly (Wang et al., 2026; Lin & Zheng, 2026), or converting a panorama into planar views through a predefined workflow before applying a pretrained VLM (Yang et al., 2026; Tran et al., 2026). Nevertheless, both can expose the model to substantial question-irrelevant visual content. As shown in Figure 1, small targets and direction-dependent relations can be obscured by ERP distortion, seams, or misaligned headings.

Inspired by Thinking in 360° (Yu et al., 2026) that explores active search by rotating a view from a given local observation, an intuitive solution is to let the model choose where and how to look based on the question, where our empirical results show that question-relevant planar views improve panoramic perception and spatial reasoning even without updating the VLM's weights. Unlike (Yu et al., 2026), general panoramic understanding requires selecting relevant regions from the full panorama and aligning the reference heading for orientation-dependent questions. This raises a question: can a VLM efficiently learn to select question-relevant observations for robust understanding while drawing on its pretrained knowledge of perspective images?

By this motivation, we propose PANOAGENT, which jointly learns to select panoramic observation actions and generate answers by two core panorama-specific tools. Specifically, PLANARVIEW renders selected regions as local planar views for detailed inspection, while PANOROTATE updates the

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.



(a) Layout distortion (b) Seam separation (c) Polar stretching

Figure 1: **Adapting observations for panoramic understanding.** The top row illustrates three challenges for ERP-only reasoning: **(a)** distorted layouts complicate relative-orientation judgments, **(b)** the horizontal seam separates nearby objects across image boundaries, and **(c)** polar stretching obscures object shape. In the bottom row, PANOAGENT selects question-relevant spherical regions and renders local planar views with PLANARVIEW, revealing cues that support the correct judgments shown. These examples motivate learning to select observations that better support a pretrained VLM's existing visual capabilities. Blue outlines mark the selected fields of view on the ERP panoramas.

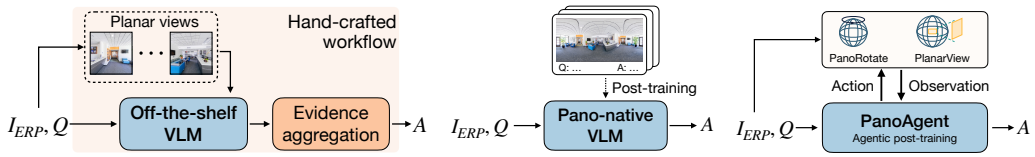
global reference heading for direction-aware reasoning. By learning when and how to use these complementary tools, PANOAGENT connects panoramic concepts to the VLM's existing visual knowledge while focusing its reasoning on the evidence needed for each question.

To learn this policy, we construct panoramic VQA pairs with trajectory supervision that connects questions, observation actions, and reasoning over visual evidence. Demonstrations of directional understanding map directions and heading conditions to local rendering or reference-heading updates. Object-attribute questions illustrate how to inspect a target for relevant details. Multi-object questions demonstrate how to select views that provide evidence for spatial judgments. Supervised fine-tuning uses these representative demonstrations to initialize observation selection and answer generation. Reinforcement learning then refines the policy through task feedback. Final-answer feedback evaluates the resulting judgments, while geometric feedback on local-view coverage and reference-heading alignment guides intermediate observation decisions.

We evaluate PANOAGENT at the 4B and 9B model scales on ODI-Bench (Yang et al., 2026), 360Bench (Tran et al., 2026), and PanoSpace-Bench (Wang et al., 2026), covering panoramic perception and spatial reasoning. Both scales substantially outperform their respective base models, and the 9B model achieves the highest overall accuracy among all compared methods on all three benchmarks (Table 1). For example, it outperforms Gemini-3.8-Flash on ODI-Bench (74.5% vs. 71.1%) and PanoSpace-Bench (71.1% vs. 61.5%) in overall accuracy (Table 1). Compared with existing panoramic adaptation methods, PANOAGENT achieves higher overall accuracy with substantially less training data than PanoWorld. It also outperforms Free360 while running approximately twice as fast with the same backbone (Table 3).

Our contributions are as follows.

- We propose a question-driven approach, PANOAGENT, for panoramic understanding that learns to select relevant observations while drawing on a VLM's pretrained perspective-image capabilities.
- We introduce two complementary panoramic tools: PLANARVIEW renders selected regions as local planar views for detailed inspection, while PANOROTATE adjusts the global reference heading for directional reasoning. We further construct a tool-use trajectory dataset to teach the model when and how to use them.
- We demonstrate strong performance across three panoramic benchmarks, including 360Bench (Tran et al., 2026), ODI-Bench (Yang et al., 2026), and PanoSpace-Bench (Wang et al., 2026), at the 4B and 9B scales, with favorable training-data and inference efficiency compared with existing adaptation methods.



(a) Training-free pipeline (b) Pano-native adaptation (c) Our method

Figure 2: **Different strategies for panoramic adaptation. (a)** Training-free methods (Yang et al., 2026; Tran et al., 2026) use planar views to exploit off-the-shelf VLM capabilities without further training, but rely on manually designed observation strategies that are not jointly optimized with answering. **(b)** Pano-native methods (Wang et al., 2026; Lin & Zheng, 2026) improve perception and spatial reasoning directly from ERP, but rely on extensive panoramic supervision without fully exploiting existing perspective-image capabilities, limiting data efficiency. **(c)** PANOAGENT jointly learns observation selection and answer generation through agentic post-training, with task feedback refining observation decisions. Adapting observations to questions and returned evidence enables better use of existing visual capabilities, yielding stronger understanding with less panoramic supervision and a better balance between data requirements and inference cost.

2 RELATED WORK

Perception and spatial reasoning in perspective images. General-purpose VLMs have made substantial progress in visual perception and reasoning (Tong et al., 2024; Zhu et al., 2025; Bai et al., 2025). For spatial understanding, training data that encode geometry and spatial relations, combined with targeted supervision, further improve visual grounding, 3D understanding, and spatial-relation reasoning (Chen et al., 2024; Cai et al., 2026; Cho et al., 2025). These efforts primarily focus on perspective images, providing visual capabilities that panoramic adaptation can build on.

Tool-augmented visual reasoning and active perception. Visual tools support reasoning through program composition (Gupta & Kembhavi, 2023; Surís et al., 2023), local inspection (Wu & Xie, 2024; Liu et al., 2024), and visual intermediate representations (Hu et al., 2024). Recent work further optimizes multi-step tool interactions through post-training, allowing models to select operations based on the task and returned evidence (Zheng et al., 2026; Song et al., 2026; Lai et al., 2026; Wu et al., 2026; Zhang et al., 2026). Closely related active perception research treats observation selection as part of perception and reasoning, adjusting views, selecting viewpoints, or invoking observation tools according to task goals and current evidence (Wang et al., 2025b; Zheng et al., 2025; Tang et al., 2026). PANOAGENT applies these ideas to panoramic adaptation by selecting spherical-region reprojections or reference-heading updates on a given ERP.

Panoramic understanding and adaptation. PanoWorld (Wang et al., 2026) combines large-scale panoramic instruction data with spherical geometry modeling to improve understanding directly from ERP images. PanoEnv (Lin & Zheng, 2026) constructs panoramic QA data with 3D annotations from synthetic environments and improves spatial reasoning through geometry-aware rewards and curriculum reinforcement learning. Among methods that require no post-training, Omni-CoT (Yang et al., 2026) combines multi-view descriptions with orientation information and question-relevant local visual cues to progressively refine answers. Free360 (Tran et al., 2026) decomposes reasoning into modular steps, applies spherical transformations tailored to each step, and integrates evidence through a scene graph. In contrast to directly learning ERP question answering or organizing transformations through predefined workflows, PANOAGENT jointly learns question-relevant observation selection and answer generation in a shared policy.

3 APPROACH

PANOAGENT learns to select panoramic observations and answer questions in a shared policy, enabling a pretrained VLM to make better use of its existing visual capabilities. Panorama-specific tools provide local view rendering and reference-heading updates. Trajectory demonstrations connect questions, observation actions, and reasoning over evidence to initialize the policy. Reinforcement learning then refines it using answer and geometric feedback from trajectories generated by the policy itself.

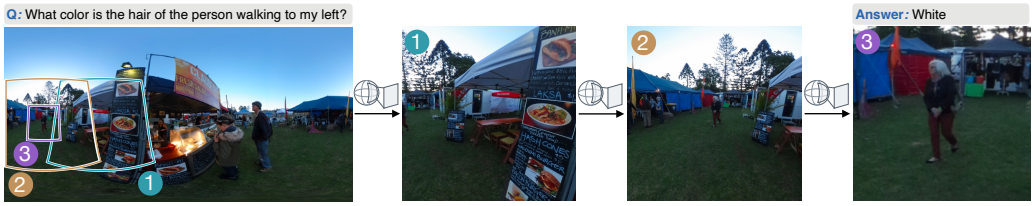


Figure 3: **Panoramic visual reasoning with the PLANARVIEW tool.** In the illustrated interaction, PANOAGENT starts from an ERP panorama and a question and acquires local observations sequentially. After each PLANARVIEW call, the returned view is appended to the visual context. The agent then reasons over the accumulated evidence to select the next viewing direction and field of view or produce the final answer. Projection icons denote tool calls, and numbers indicate their order. Matching colors link each observation to its selected region, with the ERP overlays summarizing the fields of view chosen across the interaction.

3.1 ACTIVE OBSERVATION AND INTERACTION

Given an initial ERP panorama I_0 and a question q, PANOAGENT follows ReAct (Yao et al., 2023) to interleave reasoning with optional panorama-specific tool use. The model reasons over the question and current observations, then selects a tool and its arguments or produces a final answer. The tool executes the selected observation operation and returns an observation for subsequent reasoning. The model can answer directly when the initial ERP provides sufficient evidence.

Let I_t denote the active global ERP before tool call t, starting from $t = 0$. The visual context contains one active global ERP and all previously acquired local views. PANOROTATE replaces the global ERP, whereas PLANARVIEW appends a local view without changing the global ERP. To keep local observations in a consistent reference frame, rotations are permitted only before the first PLANARVIEW call. Multiple rotations may be performed before this point.

PLANARVIEW: local planar observation. PLANARVIEW renders a selected spherical region as a local planar image:

$$\ell_t = \text{PLANARVIEW}(I_t; \mathcal{V}_t, m_t). \tag{1}$$

Here $\mathcal{V}_t$ denotes the spherical region covered by the rendering, and ℓ_t is the returned local image. The policy specifies the view through yaw, pitch, and horizontal and vertical fields of view. These parameters, together with the projection, determine its spherical coverage. We use two projection models, Rectilinear and Panini, with m_t denoting the selected model. Figure 3 illustrates observation selection and reasoning with PLANARVIEW.

PANOROTATE: global reference-heading update. The horizontal center of the active ERP corresponds to the observer's forward direction. PANOROTATE reorients the panorama horizontally so that a selected direction becomes the new forward heading, while retaining the complete scene. At tool call t, the policy specifies a rotation $\Delta\rho_t$ relative to the current heading. The global ERP update is expressed as

$$I_{t+1} = \text{PANOROTATE}(I_t; \Delta\rho_t). \tag{2}$$

The system accumulates the rotations and renders the updated ERP from the original panorama I_0 at the cumulative offset. This offset is also returned to the model so that it can track the current reference frame (Figure 4).

3.2 LEARNING TO OBSERVE FROM DEMONSTRATIONS

We construct panoramic VQA pairs with trajectory supervision to initialize observation selection and answer generation. The demonstrations comprise direct answers and tool interactions. Tool interactions also include self-correction trajectories that demonstrate how to adjust subsequent observations when the available evidence is insufficient to answer.

Task design and VQA construction. We primarily use entity descriptions, spherical locations, and angular extents from PanoWorld (Wang et al., 2026) to construct questions and associated observation regions. Ambiguous or unmatched descriptions are removed before synthesis to align

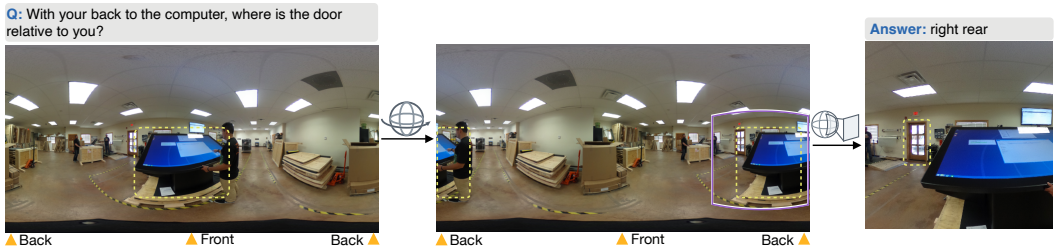


Figure 4: **Panoramic reorientation in PANOAGENT.** PANOROTATE updates the reference heading while preserving the full scene, replacing the global view in the agent's context with the reoriented ERP. The model continues reasoning and selecting observations based on this updated panoramic state. In the illustrated interaction, it selects a direction to inspect in this reference frame, then uses PLANARVIEW to render the corresponding region and verify whether the target lies within it.

question references with geometric annotations. ReplicaPano (Dong et al., 2024) additionally provides object poses for questions about relations in an object's intrinsic frame.

We organize the questions into three groups. *Panoramic direction understanding* covers target bearings under the current heading and after a specified heading change. The corresponding tool demonstrations associate directional descriptions with local views and heading-change conditions with PANOROTATE updates. *Single-object observation* includes attribute recognition and spherical localization, demonstrating how to select local views around a target to mitigate ERP distortion and reveal details. Spherical localization explicitly asks for the target's yaw and pitch. *Multi-object observation* covers egocentric and allocentric spatial relations. Shared views expose the relevant entities and any required object-orientation cues. Separate views are used when a shared view cannot provide sufficient evidence. We retain the relevant object ROIs and the expected headings specified by the questions for geometric reward computation.

Trajectory synthesis and policy initialization. With the ERP as the only visual input and no tool use, we independently sample K answers from the base VLM for each question to assess its direct answering ability. Pairs with at least τ_{direct} correct responses receive direct-answer demonstrations, while those with at most τ_{tool} enter tool-trajectory synthesis.

For tool interactions, we execute observation actions prescribed by predefined blueprints for each question type. A VLM further verifies whether the resulting evidence supports the answer and authors the trajectory reasoning after verification. We perform supervised fine-tuning on this mixture of demonstrations to train the model's reasoning, tool calls, and final answers. The resulting policy initializes reinforcement learning.

3.3 REINFORCEMENT LEARNING WITH TRAJECTORY FEEDBACK

Starting from the demonstration-trained policy, reinforcement learning further optimizes observation selection and answering using trajectories generated by the model. Feedback evaluates answer correctness, interaction validity, and the geometric relevance of observations.

Answer reward. R_{ans} evaluates the final answer. Multiple-choice and yes/no answers receive 1 if correct and 0 otherwise. Angular predictions receive a continuous score based on spherical localization error.

Interaction validity reward. Interaction validity is evaluated through format and argument scores. $R_{\text{fmt}} = 1$ if all responses and tool calls are well formed and the trajectory ends with a final answer, and 0 otherwise. $R_{\text{args}} = 1$ if all tool parameters are valid, and 0 otherwise. Direct answers without tool calls also receive $R_{\text{args}} = 1$.

Geometric reward. Final answers provide limited feedback on intermediate observation decisions. We use the target regions and expected headings retained during VQA construction to evaluate local-view coverage and heading alignment.

For each question, we index its selected target regions and expected headings by $i = 1, \ldots, N$. For a target region, $\Omega_{t,i}$ denotes its spherical ROI. Both $\Omega_{t,i}$ and the local-view coverage region $\mathcal{V}_t$ are defined in the coordinate frame of the active panorama I_t. For an expected heading, $H_{t,i}$ denotes the interval of relative yaw rotations that satisfies the question's heading condition from the current orientation.

For PLANARVIEW, the score measures the fraction of a target region covered by the local view:

$$s_{t,i}^{\text{view}} = \frac{\mu(\mathcal{V}_t \cap \Omega_{t,i})}{\mu(\Omega_{t,i})}. \tag{3}$$

Here $\mathcal{V}_t \cap \Omega_{t,i}$ is the overlap between the view's coverage and the target region on the sphere, and $\mu(\cdot)$ denotes surface area on the unit sphere. A fully covered target region receives a score of 1.

For PANOROTATE, the score measures the distance between the requested rotation and the expected heading interval:

$$s_{t,i}^{\text{rot}} = \left[1 - \frac{d_{\text{yaw}}(\Delta\rho_t, H_{t,i})}{\delta_{\text{rot}}}\right]_+. \tag{4}$$

The function d_{yaw} gives the shortest circular angular distance to the interval, taking zero inside it and accounting for wrap-around at the ERP seam. The score is 1 within the interval and decreases linearly to 0 at distance δ_{rot}. Angles are measured in degrees, and $[x]_+ = \max(0, x)$.

For each target region, we retain the highest coverage score across PLANARVIEW calls. For each expected heading, we retain the highest alignment score across PANOROTATE calls. Denoting the corresponding per-call score by $s_{t,i}$, the geometric reward averages these best scores:

$$R_{\text{geo}} = \frac{1}{N} \sum_{i=1}^{N} \max_t s_{t,i}. \tag{5}$$

Only tool calls that successfully return an observation are included. A target contributes zero if there is no such call of the corresponding tool.

Policy update. Following Group Relative Policy Optimization (GRPO) (Shao et al., 2024), we sample G trajectories for each panorama–question pair. We first combine answer correctness and interaction validity into an outcome reward:

$$R_{\text{out}} = R_{\text{ans}} + \lambda_{\text{fmt}} R_{\text{fmt}} + \lambda_{\text{args}} R_{\text{args}}, \tag{6}$$

where λ_{fmt} and λ_{args} weight the format and argument scores, respectively.

We subtract the group mean from R_{out} and scale by the group standard deviation over all sampled trajectories to obtain the outcome advantage A_j^{out} for trajectory j. Geometric rewards are normalized separately over the subset of trajectories with tool calls that return observations, yielding A_j^{geo}. Trajectories without such calls receive zero geometric advantage.

For generated token k in trajectory j, we combine the advantages as

$$A_{j,k} = A_j^{\text{out}} + \lambda_{\text{geo}} M_{j,k} A_j^{\text{geo}}. \tag{7}$$

The mask $M_{j,k}$ is 1 for reasoning and call tokens in a tool-producing turn that successfully returns an observation, and 0 otherwise. λ_{geo} weights the geometric advantage. Outcome feedback applies to all model-generated tokens, including the final answer. Geometric feedback applies only to observation-producing turns through the mask. The policy loss excludes tool observations.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Training setup. We instantiate PANOAGENT with Qwen3.5-4B and Qwen3.5-9B to evaluate panoramic adaptation at two model scales, using the 4B model for ablations and analyses. For demonstration construction, we sample $K = 8$ direct answers per question and set $\tau_{\text{direct}} = 7$ and $\tau_{\text{tool}} = 5$. We perform SFT on approximately 56K VQA training examples with trajectory supervision, followed by RL on approximately 15K VQA training examples. During RL, we sample $G = 8$ trajectories per question and set $\lambda_{\text{fmt}} = \lambda_{\text{args}} = 0.1$. Geometric feedback uses a rotation decay scale of $\delta_{\text{rot}} = 30^\circ$ and an advantage weight of $\lambda_{\text{geo}} = 0.2$. Both stages use full-parameter training.

Table 1: **Panoramic perception and spatial reasoning.** Best and second-best scores are bold and underlined, respectively.

Method	360Bench					ODI-Bench			PanoSpace-Bench				
	Overall	FP	PP	SR	DG	Overall	General	Spatial	Overall	AbsDir	Sph. Rel.	3D Spatial	Seam
GPT-4o	40.70	40.30	46.50	36.70	38.50	55.79	66.30	45.28	33.66	37.20	29.33	36.40	37.60
Claude Sonnet 4.6	45.20	45.55	42.80	45.22	49.19	65.12	73.86	57.79	53.03	65.60	58.53	38.20	53.60
Gemini-3.8-Flash	58.06	53.32	57.03	**61.80**	62.15	71.13	**80.59**	62.31	61.54	92.80	57.07	47.00	72.80
Qwen3.5-4B	36.28	34.12	39.93	35.34	35.42	61.00	72.22	51.80	31.71	27.20	30.93	36.20	29.60
Cosmos3-Nano	32.15	31.59	34.54	29.55	33.91	57.29	62.32	49.04	30.69	13.60	34.67	34.00	29.20
Qwen3.5-9B	36.54	33.95	38.70	34.89	40.97	60.32	72.07	50.26	31.51	25.20	27.03	36.55	41.20
Qwen3.6-27B	40.47	40.09	39.11	39.94	45.02	61.07	65.32	54.00	38.57	22.80	43.60	38.80	38.80
Free360 (Qwen3.5-4B)	37.60	31.98	47.78	34.66	35.07	54.21	65.92	45.00	30.69	32.80	24.67	33.00	42.00
PanoWorld (9B)	42.80	37.98	45.38	43.76	45.72	52.33	55.49	38.79	57.37	93.70	47.40	50.10	65.50
PANOAGENT (4B)	55.27	53.21	46.14	61.24	65.28	72.36	76.14	64.69	65.54	**98.40**	65.07	49.40	66.40
PANOAGENT (9B)	**59.21**	**55.03**	**57.85**	59.77	**69.33**	**74.47**	78.51	**67.38**	**71.14**	96.40	**68.53**	**61.00**	**74.00**

Table 2: **Panoramic visual search on EAGLE-360 and H*Bench.** Best and second-best scores are bold and underlined, respectively.

Method	EAGLE-360			H*Bench		
	Acc. ↑	GCD (°) ↓	GCD@50 ↑	Overall ↑	HOS ↑	HPS ↑
GPT-4o	7.50	44.26	80.00	30.1	39.1	17.0
Claude Sonnet 4.6	73.61	9.70	93.69	37.3	40.5	32.5
Gemini-3.8-Flash	80.00	5.28	93.89	51.7	49.2	**55.5**
Qwen3.5-4B	8.89	42.41	70.00	19.8	19.7	20.0
Qwen3.5-9B	9.17	40.81	68.06	19.4	26.2	9.3
EAGLE-360	64.44	16.89	96.12	56.1	74.8	28.0
PanoWorld (9B)	96.39	3.96	98.33	56.1	61.8	47.5
PANOAGENT (4B)	95.83	4.37	98.06	62.3	79.5	36.5
PANOAGENT (9B)	**98.61**	**3.35**	**98.61**	**67.6**	**85.3**	41.0

Benchmarks and metrics. We evaluate panoramic perception and spatial reasoning on 360Bench (Tran et al., 2026), ODI-Bench (Yang et al., 2026), and PanoSpace-Bench (Wang et al., 2026). We report accuracy (%) under the benchmark-standard protocols; 360Bench averages over four answer-option permutations per question, and PanoSpace-Bench Overall is the macro-average over seven multiple-choice tasks. We additionally evaluate panoramic visual search on EAGLE-360 (Xu et al., 2026) and H*Bench (Yu et al., 2026), following prior protocols (Wang et al., 2026). We use full ERP inputs for EAGLE-360 evaluation and for all methods on H*Bench. All benchmark evaluations are zero-shot, without benchmark-specific training.

Comparison methods. We compare with representative proprietary VLMs, open-weight VLMs of comparable size, and two panorama-specific adaptation methods. PanoWorld-9B (Wang et al., 2026) learns to answer directly from ERP through panoramic post-training, while Free360 (Tran et al., 2026) uses a training-free transformation pipeline. We use Qwen3.5-4B for Free360, matching the backbone of our primary model.

4.2 MAIN RESULTS

Panoramic perception and spatial reasoning. The results in Table 1 indicate that PANOAGENT substantially improves over the corresponding base VLMs at both scales. The 4B agent surpasses the larger Qwen3.6-27B in overall accuracy on all three benchmarks. It also outperforms the panorama-specific baselines PanoWorld-9B and Free360, with the latter using the same Qwen3.5-4B backbone. The 9B agent achieves the highest overall accuracy on all three benchmarks among all compared methods, including proprietary VLMs such as Gemini-3.8-Flash.

Table 6: **Training strategies for panoramic adaptation.**

Training	360 Bench	ODI-Bench	PanoSpace-Bench
Base	36.28	61.00	31.71
VQA SFT	45.20	63.94	49.60
RL-only	41.66	65.02	48.00
Traj. SFT	50.05	69.04	54.00
Traj. SFT + RL	**55.27**	**72.36**	**65.54**

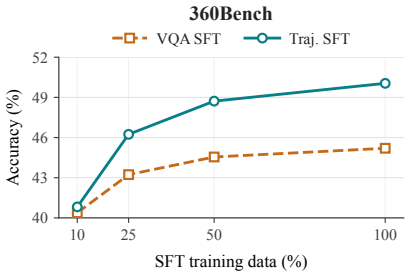


Figure 5: **Effect of training data size on 360Bench.**

Panoramic visual search. Table 2 shows strong visual-search performance at both scales. On EAGLE-360, the 4B agent approaches PanoWorld-9B, while the 9B agent leads on all three localization metrics. Both scales outperform the compared baselines in overall H*Bench success, driven by their advantage in object search.

Training data and inference cost. Table 3 compares panoramic training-data requirements and mean end-to-end inference time on a single H20 GPU. Timing includes all model turns and tool executions. PANOAGENT uses substantially fewer training examples and panoramas than PanoWorld while maintaining comparable inference time. With the same Qwen3.5-4B backbone, it runs faster than Free360 while achieving higher overall accuracy.

Table 3: **Training data and inference cost.** VQA counts include SFT and RL examples.

Method	PanoWorld	PANOAGENT	Free360
VQA examples	3M	56K + 15K	**0**
Panoramas	570K	25K	**0**
Time (s/case)	1.92	**1.86**	4.16

4.3 ABLATIONS AND ANALYSIS

Visual reasoning with prescribed transformations. We first test whether suitable observations improve panoramic understanding without updating the pretrained VLM (Table 4). Using task annotations, we construct question-relevant local views or align the reference heading. Local views are rendered from annotated ROIs supplied by Free360 at the same nominal pixels per degree as the input ERP. Heading alignment reorients the panorama and expresses the question in the aligned frame. Local views improve recognition and object-to-object reasoning, while heading alignment improves rotation and reorientation judgments. These gains show that suitable presentations of panoramic evidence help the model apply its existing perception and reasoning capabilities, motivating adaptation through learned observation.

Table 4: **Oracle panoramic transformations.** Oracle denotes annotation-guided local views or reference-heading alignment.

Method	360Bench			PanoSpace-Bench	
	FP-IR	PP-IR	SR-Os	Rot.	Reori.
ERP	44.09	46.96	32.84	29.60	31.60
Oracle	**79.77**	**65.54**	**54.77**	**61.60**	**56.40**

Contribution of panorama-specific tools. We next examine the learned policy's dependence on these operations by disabling PLANARVIEW and PANOROTATE individually and jointly at inference (Table 5). Removing PLANARVIEW substantially reduces recognition and object-to-object reasoning accuracy, while also affecting heading-dependent judgments. Removing PANOROTATE has smaller effects on recognition and object relations, but substantially reduces Rot. and Reori. accuracy. These differences reflect the complementary roles of local observation and reference-heading updates in panoramic reasoning.

Table 5: **Tool contributions to PANOAGENT.** View and Rotate denote PLANARVIEW and PANOROTATE.

Enabled tools		360Bench			PanoSpace-Bench	
View	Rotate	FP-IR	PP-IR	SR-Os	Rot.	Reori.
✓	✓	**69.09**	**60.51**	**49.89**	**77.20**	**66.80**
–	✓	45.34	42.09	32.72	34.60	40.00
✓	–	66.70	58.17	46.57	13.20	24.80
–	–	39.20	40.79	28.30	12.80	22.30

Learning an observation policy. Table 6 examines how supervision of the observation process affects panoramic adaptation. VQA SFT and trajectory SFT use the same source VQA pairs. The former retains question–answer supervision, while the latter also includes observation actions and reasoning over the returned evidence. Trajectory SFT outperforms VQA SFT on all three benchmarks, and its advantage persists across all evaluated data fractions on 360Bench (Figure 5). With only 25% of the training data, trajectory SFT outperforms VQA SFT trained on the full training data. Supervising the observation process thus reduces the number of training questions needed to reach this level of accuracy.

We then compare RL initialized from the base VLM (RL-only) or from trajectory SFT, using the same RL data, rewards, and training budget. RL-only improves answer accuracy but does not learn to invoke tools for additional observations. Initialized from trajectory SFT, RL further refines the observation and answering policy and improves performance on all three benchmarks. These results support establishing an observation policy through demonstrations and then refining it with task feedback.

Adapting to spherical action representations. Spherical actions are well matched to the geometry of panoramic observation, but differ from the pixel-coordinate representations familiar to pretrained VLMs. We compare normalized ERP boxes (Pixel) and spherical view parameters (Spherical) before and after trajectory SFT to assess their suitability as panoramic action interfaces (Table 7). SFT uses the same source demonstration trajectories, with observation actions adapted to each interface. Before SFT, Pixel performs better on 360Bench and ODI-Bench, with similar accuracy on PanoSpace-Bench. After adaptation, Spherical outperforms Pixel on all three benchmarks. These results suggest that, despite the initial knowledge gap, the model adapts to the spherical action interface and uses its existing visual capabilities to improve panoramic understanding.

Table 7: **Comparison of action interfaces.** Pixel and Spherical interfaces are compared before and after trajectory SFT.

Training	Interface	360 Bench	ODI-Bench	PanoSpace-Bench
None	Pixel	**43.73**	**61.10**	33.26
None	Spherical	38.51	54.40	**33.60**
Traj. SFT	Pixel	48.42	66.42	44.00
Traj. SFT	Spherical	**50.05**	**69.04**	**54.00**

Geometric feedback during reinforcement learning. Starting from the demonstration-initialized policy, we compare two RL feedback settings under matched training conditions (Table 8): answer correctness and interaction validity alone, and the addition of feedback on local-view coverage and reference-heading alignment. Adding geometric feedback improves accuracy on all three benchmarks, with the largest gain on PanoSpace-Bench. This improvement shows that geometric feedback on observation decisions complements final-answer evaluation.

Table 8: **Effect of geometric reward.**

Geometric reward	360 Bench	ODI-Bench	PanoSpace-Bench
w/o	51.04	70.15	60.26
w/	**55.27**	**72.36**	**65.54**

5 Conclusion

We have presented PanoAgent, which efficiently adapts pretrained VLMs to panoramic understanding by jointly learning question-relevant observation selection and answer generation. With panorama-specific tools, the agent selects local planar views or updates the global reference heading as needed, using returned evidence to guide subsequent observation and reasoning. Trajectory demonstrations connect panoramic concepts, observation actions, and reasoning over evidence, while reinforcement learning refines the policy through answer and geometric feedback. Experiments at two model scales show strong overall performance on panoramic understanding benchmarks and visual-search evaluations, together with a better balance of understanding accuracy, training-data efficiency, and inference efficiency than existing panoramic adaptation methods. These results show that learning how to observe supports efficient panoramic adaptation by presenting task-relevant visual evidence in forms that help pretrained models better use their existing perception and reasoning capabilities.

Ethics Statement

The proposed method operates on visual data and inherits potential privacy, bias, and misuse risks from the underlying models and datasets.

Reproducibility Statement

We document the training data construction, tool interfaces, optimization settings, and evaluation protocols. We plan to release code, generated training data, and model checkpoints subject to the licenses of the underlying models and datasets.

AI Use Statement

We used Qwen3.6-27B for caption grounding, VQA authoring, and trajectory reasoning generation in the construction of synthetic training data. We also used generative AI tools to assist with manuscript organization, drafting, and revision, and to check consistency between the manuscript and implementation records. The authors are responsible for verifying all AI-assisted content and take responsibility for the final text, claims, and research artifacts.

References

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025. 1, 3

Zhongang Cai, Ruisi Wang, Chenyang Gu, Fanyi Pu, Junxiang Xu, Yubo Wang, Wanqi Yin, Zhitao Yang, Chen Wei, Tongxi Zhou, Qingping Sun, Hui En Pang, Jiaqi Li, Oscar Qian, Zhiqian Lin, Xuanke Shi, Kewang Deng, Xiaoyang Han, Zukai Chen, Xiangyu Fan, Hanming Deng, Lewei Lu, Liang Pan, Bo Li, Ziwei Liu, Quan Wang, Dahua Lin, and Lei Yang. Scaling spatial intelligence with multimodal foundation models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7879–7890, June 2026. 1, 3

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. SpatialVLM: Endowing vision-language models with spatial reasoning capabilities. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14455–14465, 2024. doi: 10.1109/CVPR52733.2024.01370. 3

Jang Hyun Cho, Boris Ivanovic, Yulong Cao, Edward Schmerling, Yue Wang, Xinshuo Weng, Boyi Li, Yurong You, Philipp Kraehenbuehl, Yan Wang, and Marco Pavone. Language-image models with 3d understanding. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=yaQbTAD2JJ. 3

Yuan Dong, Chuan Fang, Liefeng Bo, Zilong Dong, and Ping Tan. Panocontext-former: Panoramic total scene understanding with a transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28087–28097. IEEE, 2024. 5, 13

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 3

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Ranjay Krishna. Visual Sketchpad: Sketching as a visual chain of thought for multimodal language models. In *Advances in Neural Information Processing Systems*, 2024. 3

Xin Lai, Junyi Li, Wei Li, Tao Liu, Tianjian Li, and Hengshuang Zhao. Mini-o3: Scaling up reasoning patterns and interaction turns for visual search. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Zp2y9O3wEj. 3

Zekai Lin and Xu Zheng. Panoenv: Exploring 3d spatial intelligence in panoramic environments with reinforcement learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9647–9657, June 2026. 1, 3

Zuyan Liu, Yuhao Dong, Yongming Rao, Jie Zhou, and Jiwen Lu. Chain-of-Spot: Interactive reasoning improves large vision-language models. *arXiv preprint arXiv:2403.12966*, 2024. 3

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300. 6

Mingyang Song, Haoyu Sun, Jiawei Gu, Linjie Li, Ranjay Krishna, and Yu Cheng. Adareasoner: Dynamic tool orchestration for iterative visual reasoning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=nUGPEmQ2ut. 3

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual inference via python execution for reasoning. In *IEEE/CVF International Conference on Computer Vision*, 2023. 3

Tianci Tang, Tielong Cai, Hongwei Wang, and Gaoang Wang. See, act, adapt: Active perception for unsupervised cross-domain visual adaptation via personalized VLM-guided agent. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=B6oQ9Cg5Sh. 3

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai C Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37:87310–87356, 2024. 1, 3

Huyen TT Tran, Van-Quang Nguyen, Farros Alferro, Kang-Jun Liu, and Takayuki Okatani. 360° image perception with MLLMs: A comprehensive benchmark and a training-free method. In *European Conference on Computer Vision*, pp. 296–313, 2026. 1, 2, 3, 7, 18, 20, 24

Changpeng Wang, Xin Lin, Junhan Liu, Yuheng Liu, Zhen Wang, Donglian Qi, Yunfeng Yan, and Xi Chen. PanoWorld: Towards spatial supersensing in 360° panorama world. *arXiv preprint arXiv:2605.13169*, 2026. 1, 2, 3, 4, 7, 13, 18, 20

Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Yiping Lu, Zhengyuan Yang, Lijuan Wang, Ranjay Krishna, Jiajun Wu, Fei-Fei Li, Yejin Choi, and Manling Li. VAGEN: Reinforcing world model reasoning for multi-turn VLM agents. In *Advances in Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=xpjWEgf8zi. 21

Ziyue Wang, Chi Chen, Fuwen Luo, Yurui Dong, Yuanchi Zhang, Yuzhuang Xu, Xiaolong Wang, Peng Li, and Yang Liu. ActiView: Evaluating active perception ability for multimodal large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7605–7633, 2025b. doi: 10.18653/v1/2025.acl-long.376. URL https://aclanthology.org/2025.acl-long.376/. 3

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*, 2026. URL https://openreview.net/forum?id=Idst6X6gmy. 3

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal LLMs. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13084–13094, 2024. doi: 10.1109/CVPR52733.2024.01243. 3

Jingtao Xu, Zizhuo Lin, Jianwen Sun, Yi Yang, and Yawei Luo. EAGLE-360: Embodied active global-to-local exploration in 360°. *arXiv preprint arXiv:2607.02479*, 2026. 7, 20

Liu Yang, Huiyu Duan, Ran Tao, Juntao Cheng, Sijing Wu, Yunhao Li, Jing Liu, Xiongkuo Min, and Guangtao Zhai. ODI-bench: Can MLLMs understand immersive omnidirectional environments? In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=fT8FrhGNo0. 1, 2, 3, 7, 18, 20, 24

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. 4

Heyang Yu, Yinan Han, Xiangyu Zhang, Baiqiao Yin, Bowen Chang, Xiangyu Han, Xinhao Liu, Jing Zhang, Marco Pavone, Chen Feng, Saining Xie, and Yiming Li. Thinking in 360°: Humanoid visual search in the wild. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. 1, 7, 20

YiFan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, Haonan fan, Kaibing Chen, Jiankang Chen, Haojie Ding, Kaiyu Tang, Zhang Zhang, Liang Wang, Fan Yang, Tingting Gao, and Guorui Zhou. Thyme: Think beyond images. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=gCWLkqK45O. 3

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. SWIFT: A scalable lightweight infrastructure for fine-tuning. In *AAAI Conference on Artificial Intelligence*, pp. 29733–29735, 2025. doi: 10.1609/aaai.v39i28.35383. URL https://doi.org/10.1609/aaai.v39i28.35383. 21

Hongpei Zheng, Shijie Li, Yanran Li, and Hujun Yin. SpatialReasoner: Active perception for large-scale 3D scene understanding. *arXiv preprint arXiv:2512.03284*, 2025. URL https://arxiv.org/abs/2512.03284. 3

Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and XingYu. Deepeyes: Incentivizing "thinking with images" via reinforcement learning. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=xUyMXkI958. 3

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, 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. 1, 3

A Method Details

A.1 Panorama-Specific Tools and Interaction

The model-facing tool descriptions are given below, with parameters defined in Table 9.

PlanarView. Render a perspective local view from the current equirectangular panorama. Use this when a spatial reasoning question requires inspecting a smaller less distorted local view.

PanoRotate. Rotate the equirectangular panorama horizontally so that the direction at `yaw_deg` moves to the image center. This simulates turning in place to face `yaw_deg`: the returned panorama follows the usual ERP convention with the new facing at the horizontal center. Useful whenever reasoning is easier in a different heading — e.g. when a question defines the observer's orientation relative to something in the scene (facing it, back to it, sitting at it), or to bring a region of interest to the center.

Table 9: **Model-facing parameters of the panorama-specific tools.** * marks required arguments; defaults accompany optional arguments. Ranges follow the tool schemas.

Argument / type	Range / default	Parameter description
PLANARVIEW		
`center_yaw_deg`* *number*	$[-180, 180]$	Center yaw in degrees. 0 is ERP horizontal center; positive yaw goes right, negative yaw goes left.
`center_pitch_deg`* *number*	$[-90, 90]$	Center pitch in degrees. 0 is the horizon; positive pitch goes up, negative pitch goes down.
`hfov_deg`* *number*	$[5, 240]$	Horizontal field of view in degrees for the perspective camera, not an ERP rectangular crop width. Prefer the smallest view that fully covers the evidence. Values above 160 degrees require panini projection; prefer at most 220 degrees and use the 240-degree limit only when complete joint coverage requires it.
`vfov_deg`* *number*	$[5, 160]$	Vertical field of view in degrees for the perspective camera, not an ERP rectangular crop height.
`projection` *string*	`rectilinear`, `panini` Default: `panini`	Projection model. Use rectilinear for precise local detail. Use panini for wide context views with less edge stretching; panini is required when `hfov_deg` exceeds 160.
PANOROTATE		
`yaw_deg`* *number*	$[-180, 180]$	Direction (degrees) to rotate to the image center. 0 is the current ERP horizontal center; positive goes right, negative goes left.

Local-view resolution. The output resolution of PLANARVIEW scales with its angular field of view to match the input ERP's nominal pixels per degree (PPD). For an input ERP of size $W \times H$ and horizontal and vertical fields of view α and β, the local image has width w and height h given by

$$w = \mathrm{round}\left(W\frac{\alpha}{360^\circ}\right), \qquad h = \mathrm{round}\left(H\frac{\beta}{180^\circ}\right). \tag{8}$$

Both fields of view are measured in degrees, and the output dimensions are determined automatically. The rendered views are passed to the VLM without further resizing.

Visual context and interaction. Each PANOROTATE action is relative to the active heading and reports its cumulative offset from the original panorama. Outputs are rendered from the original input at that cumulative heading, avoiding repeated resampling. The visual context contains one active global ERP and all acquired local observations: rotation replaces the former, while local rendering accumulates the latter. Multiple rotations are permitted before the first PLANARVIEW call; none are permitted afterward, keeping local observations in a consistent reference frame. Direct answering remains available throughout.

A.2 TRAINING DATA CONSTRUCTION

A.2.1 DATA SOURCES AND REFERENCE CLEANING

We construct VQA pairs with trajectory supervision primarily from the released PanoWorld training split (Wang et al., 2026), reserving PanoSpace-Bench for evaluation. Entity captions identify the objects referenced in a question, while their spherical locations and angular extents support directional answers and observation selection. Before question generation, we ground each caption in both the original ERP and a version reoriented by 180°. We retain a reference only when each orientation yields exactly one matched object. This screening reduces ambiguity in entity references across the two ERP layouts. We then apply a random horizontal rotation to each accepted panorama, exposing the retained targets at a broader range of panoramic bearings for subsequent question and trajectory construction. ReplicaPano (Dong et al., 2024) additionally supplies object poses for allo-

centric questions. We use Qwen3.6-27B for caption grounding, question authoring, and trajectory reasoning authorship.

A.2.2 QUESTION FAMILIES

Panoramic direction understanding. These demonstrations connect directional language to panoramic regions and changes in the observer's heading. *Object direction.* Questions such as "Which direction is the cabinet in?" connect a named target to its bearing under the current heading. *Directional navigation.* Questions such as "Which way should I go to reach the cabinet?" express the same bearing as a movement instruction. For both tasks, the blueprint uses the target's annotated bearing to select a local view from predefined directional sectors. *Directional presence.* Questions such as "Does the left side contain the cabinet?" instead specify a sector and require inspecting its contents. The blueprint renders the sector named in the question to check whether the target is present.

A second group changes the reference heading before asking about the target. *Stated rotation.* Questions specify a turn, such as "After turning 90° right, where is the cabinet?" *Object-conditioned heading.* Questions define the heading through another object, as in "With your back to the sofa, where is the cabinet?" The given turn or reference object's bearing determines the heading used by the blueprint. After any prescribed PANOROTATE update, the target's bearing in the active frame determines the local inspection sector. This separates the role of the heading-defining object from that of the object being queried.

Single-object observation. These demonstrations connect target localization to observations that reveal visual details and mitigate ERP distortion. *Entity attributes.* Questions such as "What material is the cabinet made of?" ask about visual properties absent from the target description. The target's position and extent guide a local view that exposes the relevant details. Target-centered planar views are particularly useful for small objects and regions with pronounced ERP distortion, where shape or texture can be difficult to inspect. *Spherical localization.* Questions such as "What are the yaw and pitch of the cabinet?" explicitly require the target's angular coordinates, connecting object identification to panoramic coordinates. These tasks provide supervision for selecting observations that reveal target details and mitigate distortion.

Multi-object observation. These demonstrations require observations that preserve the relevant entities, their relation, and any reference-orientation cues. *Egocentric relations.* Questions such as "From your position, is the cabinet to the left or right of the sofa?" express a relation in the observer's frame. To preserve both objects and their relative layout, the blueprint centers a joint view at the midpoint of the shorter circular arc between their annotated bearings. Its horizontal field of view is set from their angular separation with a margin, accounting for pairs separated by the ERP seam. *Allocentric relations.* Questions such as "In the sofa's own reference frame, where is the cabinet?" instead use the reference object's intrinsic orientation, requiring evidence of that orientation as well as the target's position. The blueprint first fits a joint view to both objects' angular extents. If joint coverage exceeds the prescribed field-of-view bounds or the returned evidence is insufficient, it uses separate views of the reference object and the target. The evidence check includes whether the reference object's intrinsic orientation is readable. These demonstrations connect relation judgments to decisions about view coverage. The synthesized families do not include explicit relative-depth comparison questions.

A.2.3 TRAJECTORY SYNTHESIS

After generating VQA pairs, we assess the base VLM's direct-answer capability through independent responses and assign each pair direct-answer or tool-trajectory supervision using the criteria in Section 3.2. Task-specific blueprints prescribe the tool actions and parameters for tool trajectories. Executing these actions produces global or local observations, which Qwen3.6-27B validates before authoring and reviewing the accompanying reasoning.

Self-correction demonstrations extend accepted inspection trajectories with an intentionally insufficient initial observation. The first view may miss the target, crop identifying features, or show only one member of an object pair. We render both this inspection and its predefined correction, checking that the initial observation has the intended shortcoming and that the corrected observations supply

Table 10: **Prompt template for observation assessment.** Schematic structure shared across task-specific variants.

```
Role: Assess whether the rendered observations provide the evidence
required by the task.

Question: {question}
Target / reference descriptions: {entity_descriptions}
Rendered observations: {rendered_observations}
Assessment checklist: {assessment_checklist}

Evaluate each listed check from the rendered observations and
record the visual evidence supporting the judgment. Provide a
visual prediction when required by the task.

Return one JSON object with the following structure:
{
"entities": [
    {
    "role": <target | reference>,
    "visible": <true | false | null>,
    "identity_matches": <true | false | null>
    }
],
"evidence_sufficient": <true | false | null>,
"task_checks": {
    "{task_specific_check}": <true | false | null>
},
"prediction": <task prediction or null>,
"failure_reasons": [],
"evidence_note": "{visual evidence supporting the assessment}"
}
```

the required evidence. The following stages assess the observations and author the reasoning using dedicated prompts and input contexts.

Validating observations. Before authoring reasoning, we use the VLM to check whether the rendered observations support the final answer (Table 10). Given the question, entity descriptions, and a task-specific checklist, it checks that the relevant entities are visible, match their descriptions, and have intact, readable identifying cues. For attribute questions, the queried property must be visually discernible. For egocentric relations, both objects' relative positions must be clear in the shared view. Allocentric relations additionally require visible cues to the reference object's intrinsic orientation. For heading-change questions, the heading-defining landmark must appear at the expected position in the rotated panorama.

The VLM returns the checklist judgments as structured JSON, together with supporting visual evidence and any failure reasons. For tasks requiring a visual answer prediction, the program also checks that prediction against the annotated answer. We obtain three independent assessments per candidate and aggregate them using the task's majority rules. Ordinary candidates that fail validation are discarded; those that pass supply verified evidence notes and an accepted answer for the final-reasoning authoring stage.

Final-answer reasoning. For each trajectory that passes validation, we synthesize a final explanation of how its observations support the accepted answer. We provide the VLM with the evidence notes from validation, the accepted answer, and the question with its task requirements. The prompt in Table 11 guides the explanation from the relevant visual cues, through their bearing on the question, to the answer. The resulting explanation is returned in the JSON field `answer_thought`. After review, this text is paired with the accepted answer to form the trajectory's final assistant response.

Tool-call reasoning. For each tool call, we synthesize reasoning that connects the question's evidence requirements to the prescribed observation action. The prompt in Table 12 provides the

Table 11: **Prompt template for final-answer reasoning synthesis.** Schematic structure shared across task-specific variants.

```
Role: Write the final reasoning that connects the verified evidence
to the accepted answer.

Question: {question}
Task requirements: {task_requirements}
Verified evidence: {verified_evidence}
Accepted answer: {accepted_answer}

Write a short, coherent explanation:
1. Identify the relevant target or observation and its key cues.
2. Explain how those cues resolve the judgment in the question.
3. Conclude with the accepted answer.

Base the explanation on the supplied evidence.
Return exactly one JSON object:
{
"answer_thought": "The evidence-to-answer explanation."
}
```

question, the information available at the current step, the evidence needed to answer the question, and the tool to be called at this step with its arguments. Depending on the task, this context includes the active ERP, earlier observations, or structured descriptions of their evidence.

The VLM first identifies which part of the question cannot yet be resolved from the available observations. It then explains how the given viewing direction, extent, or heading change will help obtain the missing evidence, and specifies what the returned observation must show to support an answer. This progression is expressed through three ordered units: `evidence_state` describes the current evidence and what still needs confirmation, `action_rationale` connects that need to the tool call, and `success_and_continuation` states the required observation. After review, the units are joined into the reasoning preceding the call.

For self-correction steps, the authoring context additionally includes feedback on what the preceding observation failed to reveal. This feedback grounds the explanation of how the prescribed change in viewing direction or extent addresses the missing evidence.

Reasoning review and trajectory assembly. The tool-call and final-answer reasoning generated above is reviewed against the information used to author it. The review prompt in Table 13 provides the VLM with the question, reasoning text, and its corresponding context. For tool-call reasoning, the VLM checks whether the explanation connects the evidence needed to answer the question to the tool call. In self-correction steps, this includes explaining how the next action addresses the shortcomings of the previous observation. For final-answer reasoning, it checks whether the explanation derives the accepted answer from the verified evidence. Both forms of reasoning are also checked for unsupported visual details and unclear or inconsistent statements.

The VLM returns an acceptance decision together with specific issues and revision guidance. Reasoning that requires revision is sent back to the authoring call with this feedback and reviewed again after modification. Once accepted, the tool-call reasoning is placed before each corresponding call, and the final-answer reasoning is paired with the answer. These texts, the tool calls, and their actual observations form the complete training trajectory.

Figures 6 and 7 show training demonstrations for the question families described above. The examples connect the question, inspection actions, returned observations, and final answer.

Table 12: **Prompt template for tool-call reasoning and self-correction.** Schematic structure shared across task-specific variants.

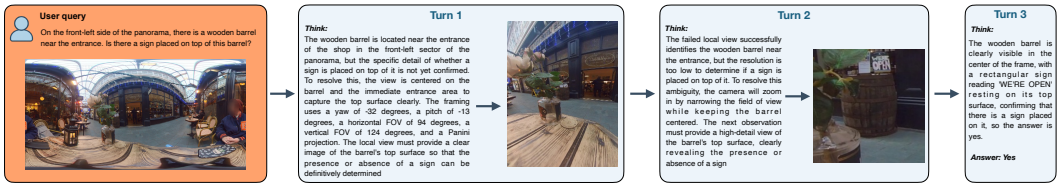
```
Role: Write the reasoning that motivates the given tool call.

Question: {question}
Available context before this call: {available_context}
Tool and arguments: {fixed_action}
Task-specific evidence requirements: {required_evidence}
Previous observation feedback, if applicable:
↪  {previous_observation_feedback}

Explain:
1. What is already known and what evidence is still missing.
If feedback is provided, explain what the previous observation
established and what it failed to reveal.
2. How the given tool and arguments address the missing evidence.
For a corrective action, explain why this change is needed.
3. What the next observation must establish to support the judgment.

Use only information available before the call and preserve the
specified action. Return three ordered reasoning units:
{
"reasoning_units": [
    {"kind": "evidence_state", "text": "..."},
    {"kind": "action_rationale", "text": "..."},
    {"kind": "success_and_continuation", "text": "..."}
]
}
```

Table 13: **Prompt template for reasoning review and revision.** Schematic structure shared across task-specific variants.

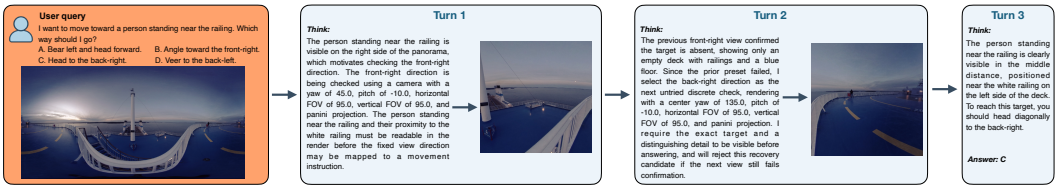
```
Role: Review the candidate reasoning for the specified stage.

Question: {question}
Stage: {tool_call / final_answer}
Context for this stage: {review_context}
Candidate reasoning: {candidate_reasoning}

Check the explanation against the context provided for this stage:
- Tool-call: Does it identify the evidence needed to answer the
question and explain how the tool call will obtain it?
For self-correction steps, does it explain how the next action
addresses the previous observation's shortcoming?
- Final-answer: Does it explain how the verified evidence supports
the accepted answer?

Also check for unsupported visual claims, inconsistent statements,
and unclear or repetitive language. Review the reasoning while
keeping the given action or accepted answer unchanged.

Return exactly one JSON object:
{
"accepted": <true | false>,
"issues": ["specific problem, if any"],
"feedback": "concrete revision guidance, or an empty string"
}
```



(a) Object-conditioned heading



(b) Entity attributes



(c) Directional navigation

Figure 6: **Training trajectory examples.** Heading alignment, object inspection, and directional questions.

B Experimental Details

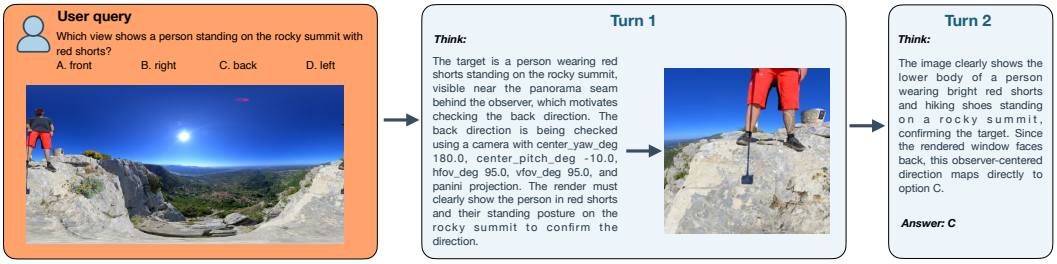
B.1 Benchmarks and Metrics

The three primary benchmarks evaluate panoramic perception and spatial reasoning. The visual-search evaluations additionally test whether the agent can locate a target through angular predictions or search decisions.

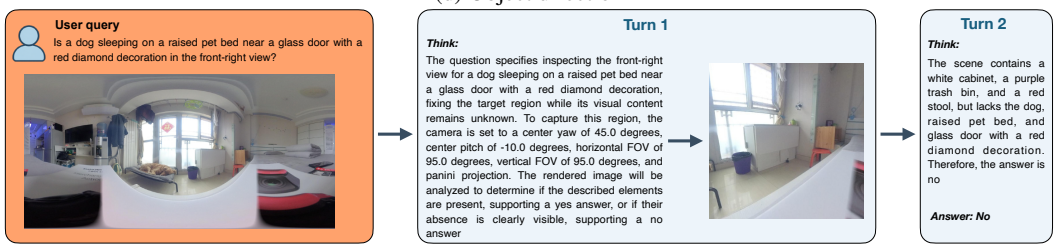
ODI-Bench. ODI-Bench (Yang et al., 2026) divides its tasks into general and spatial understanding. General understanding comprises object attributes (OA), human attributes (HA), existence, counting, and OCR. Spatial understanding comprises egocentric view orientation (EVO), allocentric view orientation (AVO), scene simulation (SS), relative direction (RD), and ODI reasoning (OR). The benchmark therefore tests both the identification of visual evidence and judgments under different spatial reference frames. We retain the supplied questions and answer formats and report accuracy. Overall is the accuracy across all evaluation examples, while General and Spatial each average the accuracies of their five constituent tasks.

360Bench. 360Bench (Tran et al., 2026) evaluates fine-grained perception (FP) and projection-distorted perception (PP), each through instance recognition (IR) and instance counting (IC). It also evaluates spatial reasoning (SR) through object-to-object (SR-Os) and object-to-viewer (SR-OV) tasks, and direction giving (DG). The evaluation contains 1,532 unique questions, each with four answer-option permutations. Accuracy is averaged over the resulting 6,128 rows.

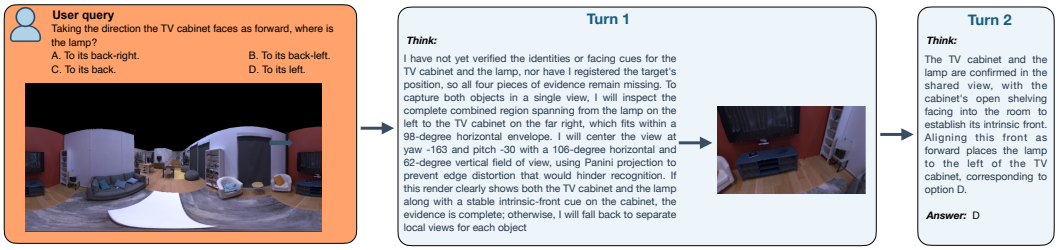
PanoSpace-Bench. PanoSpace-Bench (Wang et al., 2026) evaluates absolute and relative direction, camera rotation, object-conditioned reorientation, observer distance, relative 3D position, and ERP seam continuity. We group relative direction, rotation, and reorientation under Sph. Rel., and distance and relative 3D position under 3D Spatial. Our Overall is the arithmetic mean of these seven multiple-choice task accuracies.



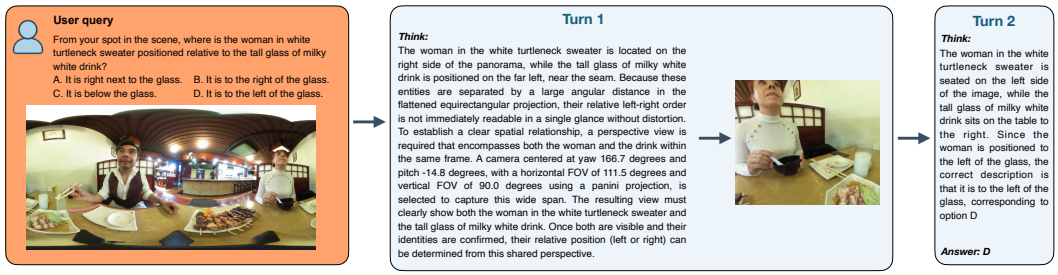
(d) Object direction



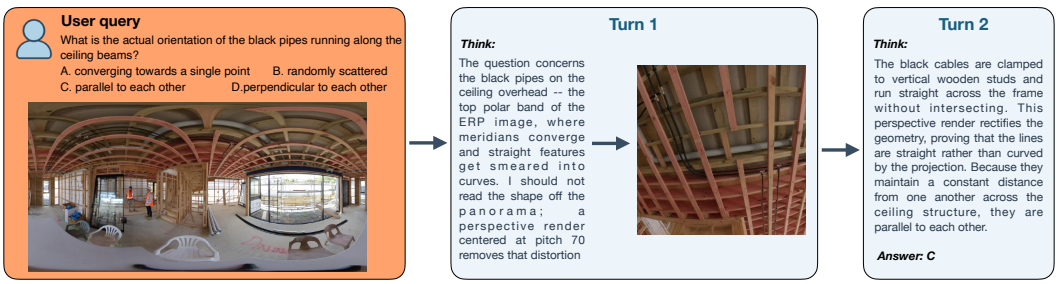
(e) Directional presence



(f) Allocentric relations



(g) Egocentric relations



(h) Polar-region attribute inspection

Figure 7: **More training trajectory examples.** Direction understanding, spatial relations, and polar-region inspection.

EAGLE-360. EAGLE-360 (Xu et al., 2026) evaluates target yaw and pitch using great-circle distance (GCD). Acc. denotes adaptive-threshold accuracy, where a hit falls within half the annotated angular bounding-box diagonal. GCD@50 measures the percentage of predictions with error below 50°. Acc. and GCD@50 are reported as percentages, and mean GCD in degrees.

H*Bench. H*Bench (Yu et al., 2026) evaluates Humanoid Object Search (HOS) and Humanoid Path Search (HPS). We report success rates for both tasks and their overall result as percentages. Our evaluation provides the full ERP panorama as the visual input for both tasks.

B.2 Evaluation Protocol and Prompts

The agent receives the ERP and question, with global panoramas capped at 2048 pixels in width while preserving aspect ratio. We use temperature 0 and generate one trajectory per evaluation instance, with a maximum of 2048 tokens per model response.

Evaluation permits up to five tool interactions, including rejected calls. When this budget is exhausted, the model receives one final request without tool schemas to produce an answer. It may answer earlier, including directly from the initial ERP. Invalid tool calls and execution failures return error feedback to the model, allowing it to revise its next action.

We use baseline results reported in PanoWorld (Wang et al., 2026), ODI-Bench (Yang et al., 2026), and Free360 (Tran et al., 2026) where available and conduct our own evaluations otherwise.

Tables 14–16 present the shared system prompt, general question templates, and visual-search localization prompt used for our agent evaluations. Dataset fields replace the placeholders.

Table 14: **System prompt for panoramic reasoning.**

You are a panoramic spatial reasoning agent.

Interpret the ERP/equirectangular panorama as a continuous 360-degree observation from a single fixed viewpoint, not as a standard perspective image. Answer using visual evidence from the provided panorama and any local perspective views that appear in the conversation.

For egocentric direction questions, interpret front/back/left/right from the viewpoint of the camera that captured the panorama. When an observer-centered frame is used, the ERP horizontal center (yaw=0) is the current front direction. If the question specifies a different reference frame, follow that frame.

Camera rotation means an in-place change of observer orientation without changing observer position. If a question asks about facing an object, treat that object's center direction as the new front direction.

You may explain your reasoning in the final response, but the final answer itself must be wrapped in: `<Answer>your final answer</Answer>`

Table 15: **User prompt templates for multiple-choice, yes/no, and short-answer questions.**

Multiple-choice template

Question: {question}
Options:
{option lines}
Please select the correct option letter.

Yes/no template

Question: {question}
The final answer should contain only yes or no.

Short-text template

Question: {question}
The final answer should be a short word or phrase.

Table 16: **User prompt for the visual search task.**

Question: {question}
Report the answer as yaw and pitch in degrees, using the same convention as camera yaw/pitch: yaw 0 is the panorama's front/horizontal center, positive is right and negative is left (-180 to 180); pitch 0 is the horizon, positive is up and negative is down (-90 to 90). Answer in exactly this format: `yaw=<value>, pitch=<value>`

B.3 Training Configurations

Supervised fine-tuning. We use SWIFT (Zhao et al., 2025) to perform SFT on panoramic VQA pairs with direct-answer or tool-interaction demonstrations, including self-correction trajectories. For self-correction, the reasoning and tool call for the initial insufficient inspection are retained as context but excluded from the SFT loss. Subsequent corrective steps and the final answer remain supervised. Tables 17 and 18 report the shared SFT configuration and the distribution of VQA training examples across question types and supervision forms.

Table 17: **Shared SFT settings for the 4B and 9B backbones.**

Category	Hyperparameter	Value / Setting
Model	Trainable modules	Language model
	Vision encoder frozen	True
	Aligner frozen	True
Data	Maximum sequence length	16,384 tokens
	Maximum image tokens	2,048
Training	Batch size per GPU	1
	Gradient accumulation steps	4
	Effective batch size	32
	Learning rate	10^{-5}
	Epochs	1
	LR scheduler	Cosine
	Warmup ratio	0.03
	Mixed precision	BF16
	Maximum gradient norm	1.0
	DeepSpeed stage	ZeRO-2
	Gradient checkpointing	Enabled
	GPUs	8

Reinforcement learning. We initialize the RL policy from the SFT checkpoint and train it using Vagen (Wang et al., 2025a). SFT and RL use disjoint sets of synthesized VQA pairs, with 15,500 pairs in the RL set. We select RL questions by running the SFT-trained model four times on each candidate and retaining those answered correctly in one to three runs.

Tables 19 and 20 report the optimization, rollout, and reward settings and the distribution of VQA training examples. During training, the policy samples a group of trajectories for each VQA pair. We use the reference answer and geometric annotations to compute the answer and geometric rewards for these trajectories. Batch and mini-batch sizes count questions before rollout expansion. The response-length limit covers the full interaction after the initial prompt, including model outputs and tool observations.

We optimize the combined token advantage in Equation 7 with the clipped GRPO objective, averaging the loss over model-generated tokens. A low-variance KL term regularizes the policy toward the fixed SFT reference policy and is added to the actor loss.

Angular answer scoring. For spherical-localization answers, let d be the great-circle angular distance between the predicted and annotated target directions. The acceptance radius δ_{ans} is supplied with each training example. The answer evaluator uses the reward-specific tolerance when provided

Table 18: **SFT data by question type and supervision form.** Counts are training examples.

Question type	Direct answer	Tool interaction		Total
		Without self-correction	With self-correction	
Panoramic direction understanding				
Object direction	1,894	3,042	1,424	6,360
Directional navigation	1,799	3,258	1,548	6,605
Directional presence	2,755	3,684	0	6,439
Object-conditioned heading	2,504	3,000	1,952	7,456
Stated rotation	1,042	1,407	987	3,436
Subtotal	9,994	14,391	5,911	30,296
Single-object observation				
Entity attribute	3,800	5,186	1,149	10135
Spherical localization	1,432	2,941	1,256	5,629
Subtotal	5,232	8,127	2,405	15,764
Multi-object observation				
Egocentric relation	2,090	2,254	1,394	5,738
Allocentric relation	786	2,300	1,240	4,326
Subtotal	2,876	4,554	2,634	10,064
Total	18,102	27,072	10,950	56,124

Table 19: **RL optimization, rollout, and reward settings.**

Category	Hyperparameter	Value / Setting
Policy	Learning rate	10^{-6}
	Batch size (questions)	64
	PPO mini-batch size (questions)	16
	PPO epochs	1
	Trainer steps	240
	PPO clip ratio	0.2
	Dual-clip coefficient	3.0
	KL loss coefficient	0.001
	KL estimator	Low variance (k_3)
	Loss aggregation	Token mean
	Trainable modules	Language model
	Frozen modules	ViT and aligner
Rollout	Rollouts per question (G)	8
	Maximum model turns	5
	Maximum tool calls	3
	Maximum prompt tokens	4,096
	Maximum response tokens	10,240
	Sampling temperature	0.7
Reward	Format weight (λ_{fmt})	0.1
	Argument weight (λ_{args})	0.1
	Rotation scale (δ_{rot}, degrees)	30.0
	Geometric advantage weight (λ_{geo})	0.2

Table 20: **RL data by question type.** Counts are VQA training examples before rollout sampling.

Question type	Training examples	Share
Panoramic direction understanding		
Stated rotation	1,500	9.7%
Object direction	1,500	9.7%
Directional presence	1,500	9.7%
Object-conditioned heading	1,500	9.7%
Directional navigation	1,500	9.7%
Subtotal	7,500	48.4%
Single-object observation		
Entity attribute	2,000	12.9%
Spherical localization	2,000	12.9%
Subtotal	4,000	25.8%
Multi-object observation		
Egocentric relation	2,000	12.9%
Allocentric relation	2,000	12.9%
Subtotal	4,000	25.8%
Total	15,500	100.0%

and otherwise uses the example's success tolerance. The continuous answer score is

$$R_{\text{ans}} = \begin{cases} 1, & d \le \delta_{\text{ans}}, \\ \max\left(0, \dfrac{c-d}{c-\delta_{\text{ans}}}\right), & d > \delta_{\text{ans}}, \end{cases} \tag{9}$$

where all angles are measured in degrees and $c = \max(90, \delta_{\text{ans}} + 10^{-9})$ keeps the zero-score endpoint above the acceptance radius. The score is constant within the acceptance radius and decreases linearly to zero outside it. Answers without valid angular coordinates receive zero.

Geometric feedback computation. We approximate the local-view coverage in Equation 3 using a grid of directions sampled uniformly in longitude and in the sine of latitude within each target ROI. These samples represent equal areas on the unit sphere. We map each direction into the selected view using its projection model and estimate coverage as the fraction lying within the image bounds.

For PANOROTATE, the VQA synthesis pipeline provides the expected heading in the original ERP frame. We accumulate the agent's relative rotations to express the resulting heading in this frame, then measure its circular angular distance to the target.

Within each question's rollout group, geometric rewards are normalized over trajectories with successful tool observations. If this subset has fewer than two trajectories, their geometric advantage is zero. Otherwise, we subtract the subset mean and divide by its sample standard deviation plus 10^{-6}. Identical scores therefore also yield zero geometric advantage.

C SUPPLEMENTARY RESULTS

C.1 DETAILED BENCHMARK RESULTS

360Bench. Table 21 resolves the aggregate gains into recognition, counting, and spatial judgments. Relative to Qwen3.5-4B, PANOAGENT-4B improves instance recognition in both fine-grained and projection-distorted settings, by 25.00 and 13.55 percentage points, respectively. Its object-to-viewer reasoning and direction-giving scores also improve substantially, reaching 72.33% and 65.28%. Compared with Free360 using the same Qwen3.5-4B backbone, the agent gains 17.67 points overall and performs better on six of the seven tasks. The 9B agent achieves the highest overall accuracy and leads on both recognition tasks, object-to-viewer reasoning, and direction giving. Counting remains less consistent: the 4B agent scores slightly below its initialization on PP-IC, and Gemini retains the highest accuracy on both counting tasks. These results show larger gains in recognition and directional judgments than in counting.

Table 21: **Task-level results on 360Bench.** FP/PP denote fine-grained/projection-distorted perception; IR/IC denote instance recognition/counting. SR-Os/SR-OV denote object-to-object/ object-to-viewer reasoning; DG denotes direction giving. Accuracy is averaged over four option permutations per question. GPT-4o and the Qwen2.5-VL-7B-based results are from Tran et al. (2026). Best and second-best scores are bold and underlined.

Method	Format	FP-IR	FP-IC	PP-IR	PP-IC	SR-Os	SR-OV	DG	Overall
Closed-source frontier models									
GPT-4o	ERP	48.60	32.00	54.20	38.80	37.70	35.70	38.50	40.70
Claude Sonnet 4.6	ERP	55.91	35.38	52.45	33.10	39.66	50.67	49.19	45.20
Gemini-3.8-Flash	ERP	65.80	**41.07**	<u>63.55</u>	**50.47**	**53.07**	70.33	62.15	<u>58.06</u>
Open-source models of comparable scale									
Qwen2.5-VL-7B	ERP	59.50	27.90	42.20	36.00	30.70	31.20	33.70	37.30
Qwen3.6-27B	ERP	50.91	29.46	46.38	31.81	34.66	45.11	45.02	40.47
Cosmos3-Nano	ERP	43.64	19.75	40.42	28.64	26.25	32.78	33.91	32.15
Panorama-specific methods									
Omni-CoT(Qwen2.5-VL-7B)	Hybrid	58.90	24.30	49.50	40.30	28.30	41.00	35.00	39.50
Free360(Qwen2.5-VL-7B)	Hybrid	60.70	30.00	54.70	43.90	33.60	53.30	41.40	45.30
Free360(Qwen3.5-4B)	Hybrid	41.93	22.21	58.18	37.32	38.18	31.22	35.07	37.60
PanoWorld-9B	ERP	47.39	28.57	45.91	44.84	38.52	49.00	45.72	42.80
PANOAGENT – 4B									
Qwen3.5-4B	ERP	44.09	24.33	46.96	32.86	32.84	37.78	35.42	36.28
SFT	Agent	54.20	24.22	63.08	39.44	42.73	65.11	61.92	50.05
SFT + RL	Agent	<u>69.09</u>	37.61	60.51	31.69	<u>49.89</u>	<u>72.33</u>	<u>65.28</u>	55.27
PANOAGENT – 9B									
Qwen3.5-9B	ERP	46.93	21.21	44.86	32.51	32.84	36.89	40.97	36.54
SFT	Agent	52.50	31.47	62.03	44.01	41.82	67.22	57.87	50.96
SFT+RL	Agent	**70.22**	<u>40.11</u>	**65.51**	<u>50.16</u>	45.45	**73.78**	**69.33**	**59.21**

ODI-Bench. Table 22 shows particularly strong gains in egocentric view orientation. PANOAGENT raises EVO accuracy from 50.43% to 90.55% at 4B and from 46.63% to 91.78% at 9B, consistent with its emphasis on interpreting evidence under an active panoramic heading. At 4B, the agent exceeds Omni-CoT (o3) by 2.33 points overall, with higher EVO and OCR accuracy but slightly lower scene-simulation accuracy. The 9B agent achieves the highest overall score and also improves allocentric orientation to 50.73%, compared with 32.29% for its initialization. The remaining gap between egocentric and allocentric orientation highlights the difficulty of reasoning from an object's intrinsic frame, even as both improve with training.

Table 22: **Task-level results on ODI-Bench (Yang et al., 2026).** OA/HA denote object/human attributes; EVO/AVO denote egocentric/allocentric view orientation; SS denotes scene simulation; RD denotes relative direction; OR denotes ODI reasoning.

Method	Overall	OA	HA	Exist.	Count.	OCR	EVO	AVO	SS	RD	OR
Closed-source frontier models											
GPT-4o	55.79	74.43	67.76	65.50	49.42	74.38	43.24	32.49	39.60	57.55	53.50
o3	62.62	75.82	74.01	75.00	57.56	77.69	56.20	39.62	46.60	70.20	59.50
Claude Sonnet 4.6	65.12	77.38	76.32	78.00	55.81	81.82	60.12	41.09	49.60	72.65	**65.50**
Gemini-3.8-Flash	71.13	78.03	**83.55**	<u>83.00</u>	**67.44**	**90.91**	83.07	29.14	**58.80**	**82.04**	58.50
Open-source models of comparable scale											
Cosmos3-Nano	57.29	74.34	71.05	39.00	51.16	76.03	53.13	41.30	32.00	63.27	55.50
Qwen3.6-27B	61.07	77.70	<u>80.92</u>	38.50	47.67	81.82	51.90	36.06	50.60	<u>77.96</u>	53.50
Panorama-specific methods											
Omni-CoT (GPT-4o)	62.08	73.77	68.42	75.00	54.07	76.03	71.53	37.94	37.60	52.65	58.50
Omni-CoT (o3)	70.03	76.15	75.66	81.00	<u>64.53</u>	78.51	82.60	42.98	55.20	71.02	62.00
Free360 (Qwen3.5-4B)	54.21	76.15	74.67	52.50	49.42	76.86	31.53	24.95	41.60	64.90	62.00
PanoWorld	52.33	71.64	66.78	11.50	52.33	75.21	65.77	20.34	24.40	51.43	32.00
PANOAGENT – 4B											
Qwen3.5-4B	61.00	77.79	75.99	75.00	54.65	77.69	50.43	32.29	42.00	74.29	60.00
SFT	69.04	74.10	75.66	79.50	50.00	80.17	85.52	44.86	53.60	66.12	60.00
SFT + RL	<u>72.36</u>	<u>78.36</u>	75.33	**84.50**	54.07	88.43	<u>90.55</u>	45.49	54.00	71.43	62.00
PANOAGENT – 9B											
Qwen3.5-9B	60.32	77.62	76.97	71.00	58.72	76.03	46.63	32.29	46.80	68.57	57.00
SFT	69.06	74.18	75.00	81.00	55.81	77.69	88.71	<u>49.27</u>	50.40	60.82	57.50
SFT+RL	**74.47**	**79.84**	80.59	**84.50**	57.55	<u>90.08</u>	**91.78**	**50.73**	<u>56.60</u>	74.29	<u>63.50</u>

PanoSpace-Bench. Table 23 further examines direction and reference-frame reasoning. Both model scales improve over their initializations on all seven tasks. PANOAGENT-4B exceeds PanoWorld by 8.17 points overall, with gains of 24.80 points on camera rotation and 19.60 on object-conditioned reorientation. Its observer-distance accuracy remains below PanoWorld's. The

training-stage comparison shows a pronounced contribution from RL: for the 4B model, Rot. increases from 62.40% to 77.20% and Reori. from 22.40% to 66.80% relative to trajectory SFT. At 9B, RL improves all seven tasks over trajectory SFT, producing the highest overall score of 71.14% and the strongest results on rotation, reorientation, distance, relative 3D position, and seam continuity.

Table 23: **Task-level results on PanoSpace-Bench.** Overall averages the seven task accuracies. AbsDir/RelDir denote absolute/relative direction; Rot./Reori. denote camera rotation/ object-conditioned reorientation; Dist. denotes observer distance; Rel3D denotes relative 3D position; Seam denotes ERP seam continuity. Best and second-best scores are bold and underlined.

Method	Overall	AbsDir	RelDir	Rot.	Reori.	Dist.	Rel3D	Seam
Closed-source frontier models								
GPT-4o	33.66	37.20	34.80	28.40	24.80	45.60	27.20	37.60
Claude Sonnet 4.6	53.03	65.60	**52.80**	67.20	55.60	49.60	26.80	53.60
Gemini-3.8-Flash	61.54	92.80	46.00	67.60	57.60	54.80	39.20	<u>72.80</u>
Open-source models of comparable scale								
Cosmos3-Nano	30.69	13.60	34.80	30.40	38.80	47.60	20.40	29.20
Qwen3.5-9B	31.51	25.20	32.20	22.60	26.30	48.60	24.50	41.20
Qwen3.6-27B	38.57	22.80	38.00	51.20	41.60	57.20	20.40	38.80
+ visual prompt	39.01	55.20	34.00	36.40	28.80	46.00	26.20	46.50
Panorama-specific methods								
Free360(Qwen3.5-4B)	30.69	32.80	32.40	10.80	30.80	48.00	18.00	42.00
PanoWorld	57.37	93.70	42.60	52.40	47.20	<u>59.60</u>	40.60	65.50
PANOAGENT – 4B								
Qwen3.5-4B	31.71	27.20	31.60	29.60	31.60	48.00	24.40	29.60
SFT	54.00	95.20	48.00	62.40	22.40	48.40	37.60	64.00
SFT + RL	<u>65.54</u>	**98.40**	<u>51.20</u>	<u>77.20</u>	<u>66.80</u>	53.60	<u>45.20</u>	66.40
PANOAGENT – 9B								
Qwen3.5-9B	31.51	25.20	32.20	22.60	26.30	48.60	24.50	41.20
SFT	62.57	94.80	49.20	67.60	60.40	56.80	40.80	68.40
SFT+RL	**71.14**	<u>96.40</u>	<u>51.20</u>	**83.20**	**71.20**	**63.20**	**58.80**	**74.00**

C.2 ABLATION DETAILS

Oracle local views. For FP-IR and PP-IR, the target ROI determines the rendered region; for SR-Os, the view covers the joint angular extent of both referenced targets. We use Rectilinear projection for narrow views and Panini for wider views, rendering each local view at the same pixels per degree (PPD) as the original ERP, without resizing during model preprocessing. Both the ERP and oracle-view conditions use direct answering with tools and thinking disabled.

Oracle heading alignment. On PanoSpace-Bench, Rot. uses the turn specified in the question to reorient the ERP, while Reori. uses the reference object's annotated yaw. The question is adjusted to ask for the target direction in the updated frame, with the original answer options retained, so that the turn is not applied twice. This diagnostic evaluates a pre-aligned panorama and its corresponding question.

Pixel tool interfaces. The Pixel and Spherical variants share the same source demonstration trajectories. For Pixel SFT, we retain the original observation images while converting the tool actions to the Pixel representation. Tables 24 and 25 specify the Pixel tool schemas.

Training-time conversion (Spherical to Pixel). For each spherical local-view action, we project points sampled from its coverage onto the active ERP and compute a bounding box that accounts for horizontal periodicity. This box approximates the original observation region. It is represented as `bbox=`$[x_1, y_1, x_2, y_2]$, with coordinates normalized to $[0, 1000]$ along the ERP width and height. A box with $x_1 > x_2$ crosses the left–right seam. Rotation actions are converted to normalized horizontal heading coordinates, `center_x`.

Test-time execution (Pixel to Spherical). We sample points within the requested ERP box and back-project them onto the sphere. From these samples, we estimate the local-view center (yaw and pitch) and horizontal and vertical FOVs. Projection options, FOV bounds, and the output-resolution rule follow the Spherical interface in Section A.1. The rotation tool moves the direction specified by `center_x` to the panorama center: 500 denotes the current front, and 0 and 1000 denote the current back.

These region conversions are approximate, so executing a converted action need not exactly reproduce the original demonstration image. The ablation evaluates the practical suitability of the two action interfaces for panoramic observation.

Table 24: **Pixel local-view function schema.**

```
{
  "type": "function",
  "function": {
    "name": "look_at_region",
    "description": "Request a close-up render of a rectangular region of
    ↪  the current equirectangular panorama. Use this when a spatial
    ↪  reasoning question requires inspecting a smaller less distorted
    ↪  local view.",
    "parameters": {"type": "object",
      "properties": {"bbox": {
          "type": "array",
          "items": {"type": "number", "minimum": 0, "maximum": 1000},
          "minItems": 4,
          "maxItems": 4,
          "description": "[x1, y1, x2, y2] region to inspect, normalized
          ↪  0-1000 over the panorama's width and height respectively.
          ↪  Horizontal intervals run from x1 to x2 with wraparound:
          ↪  x1>x2 denotes a region crossing the ERP seam. The
          ↪  horizontal extent must be positive, and y1<y2. Box the
          ↪  region as large or small as the context you need: a tight
          ↪  box gives a close, less distorted view of a small area; a
          ↪  wide box gives more surrounding context."
        },
        "projection": {
          "type": "string", "enum": ["rectilinear", "panini"],
          "default": "panini",
          "description": "Projection model. Use panini when the
          ↪  converted horizontal FOV exceeds 160 degrees."
        }
      }, "required": ["bbox"]
    }
  }
}
```

Table 25: **Pixel rotation function schema.**

```
{
  "type": "function",
  "function": {
    "name": "rotate_panorama",
    "description": "Rotate the current equirectangular panorama
    ↪  horizontally so that the direction at center_x moves to the
    ↪  image center. Coordinates are normalized from 0 to 1000 over the
    ↪  current panorama width: 500 is the current front, values above
    ↪  500 turn right, values below 500 turn left, and 0/1000 indicate
    ↪  the current back direction.",
    "parameters": {"type": "object",
      "properties": {"center_x": {
          "type": "number",
          "minimum": 0,
          "maximum": 1000,
          "description": "Normalized horizontal coordinate to move to
          ↪  the image center. 500 is the current center/front."
        }
      }, "required": ["center_x"]
    }
  }
}
```

Training configurations. Table 6 compares two choices in policy training: the supervision used for SFT and the initialization used for RL. The SFT variants start from Qwen3.5-4B and share the same source VQA pairs. VQA SFT retains ERP–question–answer supervision and has no tool access during training or evaluation. Trajectory SFT retains the accompanying reasoning and any tool interactions, including the returned observations.

For the data-size comparison in Figure 5, we fix the number of optimization steps across data fractions to the number required for one epoch on the full training set.

For the RL comparison, RL-only starts from Qwen3.5-4B, while Traj. SFT + RL starts from the checkpoint reported in the Traj. SFT row. Both runs use the same RL data and rewards and the training budget in Table 19. Both PLANARVIEW and PANOROTATE are available during RL training and evaluation.

Inference-time tool masking. We evaluate the same trained PANOAGENT model with different tools enabled (Table 5). We remove disabled tools from the model-facing schemas. Calls to disabled tools are rejected with error feedback, while the remaining tools follow the same interaction protocol.

Efficiency measurement. We measure inference time on the full evaluation sets of 360Bench, ODI-Bench, and PanoSpace-Bench using a single H20 GPU, a shared vLLM serving configuration, and an ERP-equivalent input width of 2048 pixels. Cases are processed sequentially. Timing covers the complete inference procedure through the final answer, including all model calls and tool executions. We report the mean per-case time over all cases pooled across the three benchmarks.

Free360 retains the official multi-stage inference pipeline, using Qwen3.5-4B with thinking disabled and CMP inputs generated at the shared input scale. PANOAGENT reuses the KV cache across interaction turns. PanoWorld's released model includes a custom positional-encoding module that prevents direct loading in vLLM. We therefore use its Qwen3.5-9B backbone to approximate the single-response runtime under the same serving backend. This backbone proxy excludes any additional overhead from the custom module.

C.3 TASK-LEVEL TOOL USE

Figure 8 summarizes the observation patterns of PANOAGENT-4B. Local inspection dominates on 360Bench, while both tools are used in most scene-simulation questions on ODI-Bench and rotation and reorientation questions on PanoSpace-Bench. Direct answering is more common for ODI-Bench counting, accounting for 57.6% of examples.

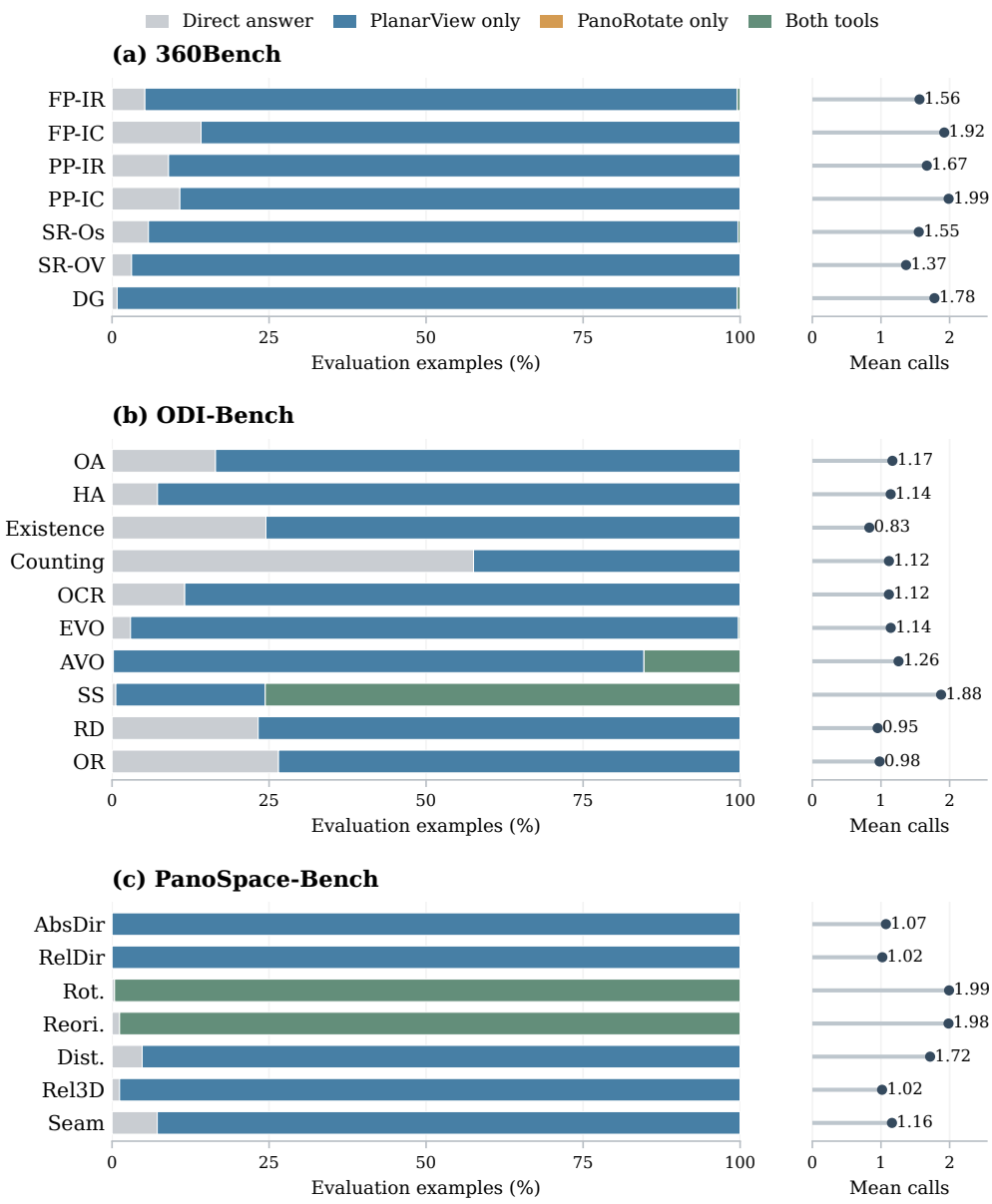


Figure 8: **Task-level tool use by PANOAGENT.** Bars partition evaluation examples into direct answers, PLANARVIEW only, PANOROTATE only, and use of both tools. Dots show mean invocations per example, including zero-call answers. Counts include unsuccessful calls. 360Bench includes all four option permutations; PanoSpace-Bench covers the seven multiple-choice tasks.

C.4 INFERENCE EXAMPLES

Figures 9–12 illustrate how PANOAGENT selects observations and uses the returned evidence to answer panoramic questions.

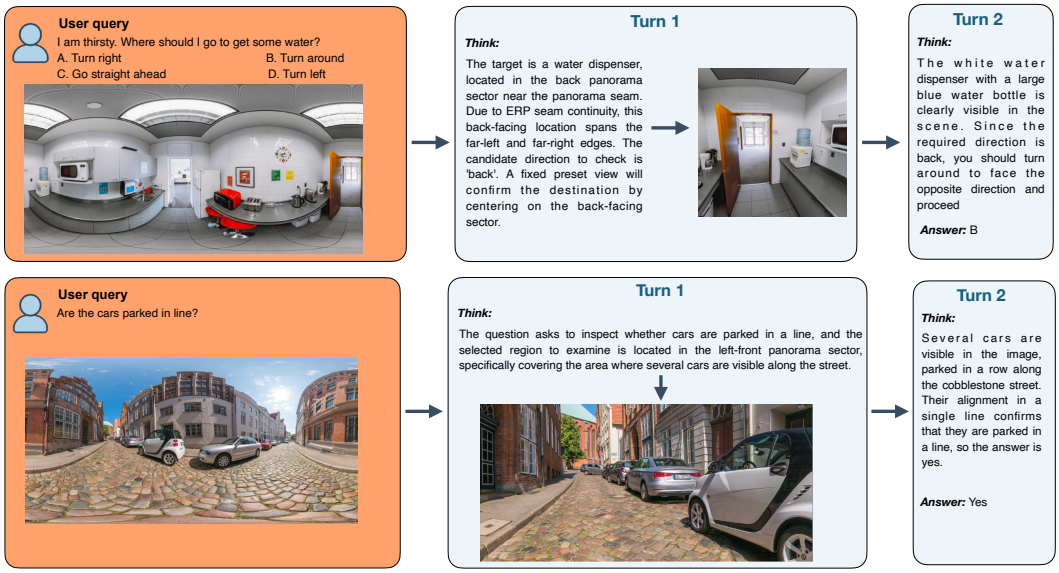


Figure 9: **Local inspection for navigation and scene layout.** The selected views reveal the destination (top) and the arrangement of parked cars (bottom), providing evidence for the subsequent responses.

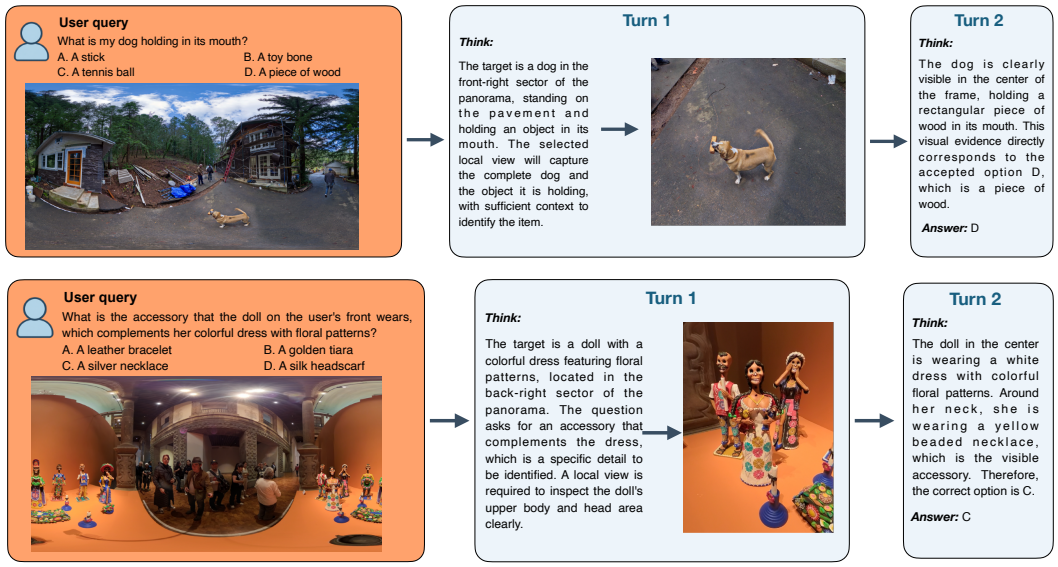


Figure 10: **Target-centered attribute inspection.** Local views isolate the queried object and its relevant details in the dog (top) and doll (bottom) examples.

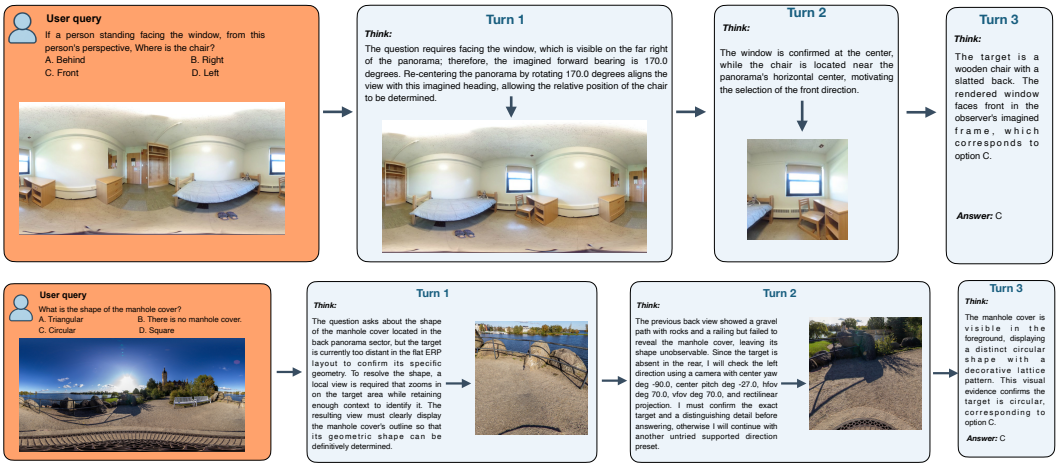


Figure 11: **Sequential panoramic inspection.** The upper example updates the global heading before acquiring a local view. The lower example changes the viewing direction after the first observation misses the target.

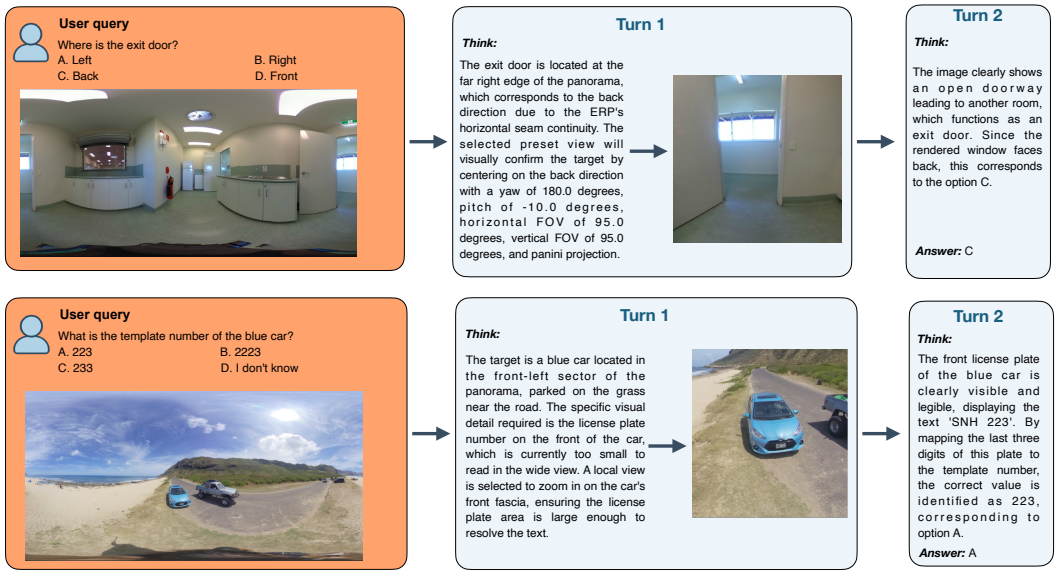


Figure 12: **Local evidence for direction and text recognition.** The agent inspects a doorway (top) and a vehicle's license plate (bottom) before answering.

D Discussion and Limitations

Scope of training tasks. Our training data focus on basic panoramic concepts and representative observation strategies to teach question-relevant observation selection. The current tasks mainly involve short observation sequences. Tasks centered on sustained exploration, multi-step planning, and evidence integration over long observation sequences are not yet included in the training data. Extending the data to these settings would support further study of longer-horizon panoramic observation and reasoning.

Blueprint-based trajectory synthesis. Predefined trajectory blueprints specify the observation strategies represented in the initial demonstrations, providing structured supervision for policy initialization. Subsequent RL further refines the observation and answering policy through outcome and geometric feedback. Expanding the diversity and complexity of the demonstrated action sequences while maintaining demonstration quality remains an open direction.

Future directions. We plan to investigate data synthesis in which models actively explore panoramic scenes and use observation feedback to generate more complex tasks and corresponding interaction trajectories. This could broaden both task coverage and the range of observation strategies in the training data.