Learning Visually-Grounded Active View Selection for Embodied Question Answering

Juil Koo* **Daehyeon Choi*** **Sangwoo Youn*** **Phillip Y. Lee** **Minhyuk Sung**
KAIST
{63days,daehyeonchoi,andy2884,phillip0701,mhsung}@kaist.ac.kr

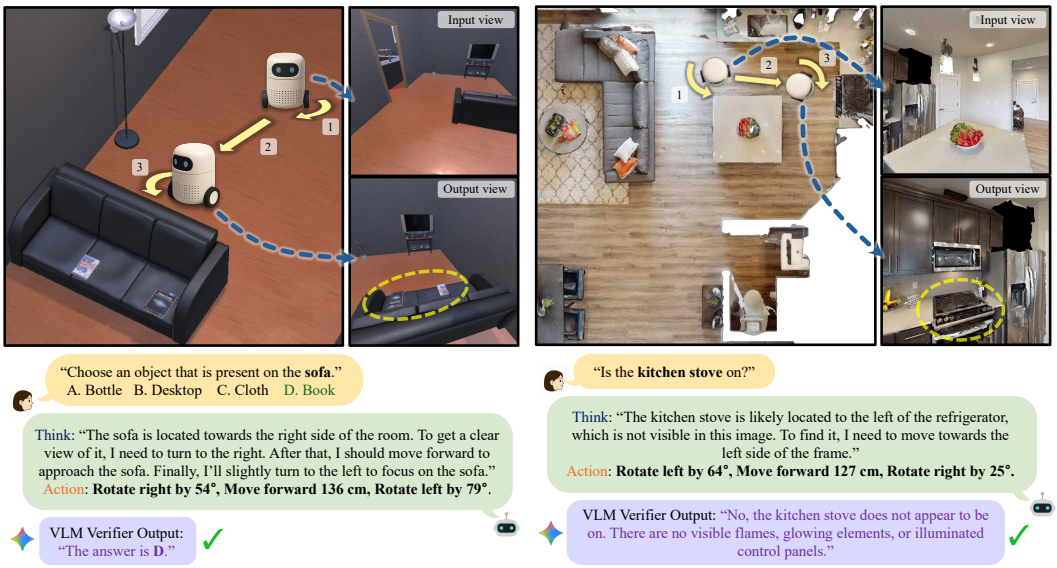


Figure 1: **An overview of our proposed Visually-Grounded Active View Selection (VG-AVS).** Given a 3D environment from synthetic (left) to real scenes (right) and a question, our learning-based active view selection (AVS) framework predicts continuous actions to refine the agent's viewpoint. The refined view is then fed into a VLM verifier, which answers the question based on the improved observation.

Abstract

Embodied Question Answering (EQA) pipelines typically combine scene exploration, scene memory, and a final question-answering step on the resulting view. While exploration and memory have received substantial attention, one component remains underdeveloped: refining the agent's viewpoint near the target so that the final observation actually supports answering. We isolate this as a learnable task, Visually-Grounded Active View Selection (VG-AVS), in which the agent iteratively adjusts its viewpoint based on partial visual clues and terminates once the evidence is sufficient. We construct a synthetic dataset with automatically generated paired query–target views, from single-step adjustments to multi-step refinements through complex spatial layouts, and fine-tune pretrained VLMs through supervised fine-tuning (SFT) followed by RL-based policy optimization. Our approach generalizes to unseen synthetic and real scenes, and plugging the learned

*Equal contribution.

43rd International Conference on Machine Learning (ICML 2026). RLxF: Reinforcement Learning from World Feedback Workshop.

module into existing EQA pipelines improves downstream question-answering accuracy.

1 Introduction

Embodied Question Answering (EQA) [6, 17, 15] requires several capabilities working together: scene exploration [23, 15], scene representation and memory [35, 24], and visual question answering on the resulting observation. Recent progress has largely focused on the first two, yielding more efficient exploration policies and richer 3D scene representations. One step in this pipeline, however, remains underdeveloped: once the agent is near the target, refining its viewpoint so that the final observation actually supports answering the question. This step is where answer accuracy is ultimately decided, yet existing systems either treat the viewing direction as a byproduct of navigation [23, 15] or directly hand off to a VLM verifier on whatever view the navigator stops at. We focus on this specific gap.

Concretely, we consider scenarios where the agent's current view contains only *partial visual clues*: a supporting surface is visible but the target object is occluded, or the object appears too distant for the observation to be sufficient for answering. These clues, however, carry informative signals about where to look next: the back of a sofa suggests objects on its seat or behind it, and a half-visible doorway hints at an adjacent room (Figure 1). A capable model should leverage such cues to actively adjust its viewpoint and gather the missing evidence, rather than passively answering from the current view.

We formalize this as Visually-Grounded Active View Selection (VG-AVS): given an egocentric view and a language query, the model decides at each step whether to adjust its viewpoint to gather more evidence or to stop for answering, until the view is sufficient or a maximum number of steps is reached. Our formulation differs from prior EQA setups in three concrete ways. (1) Beyond deciding where to move, the agent also decides where to look after moving: prior EQA navigators [23, 15] treat the viewing direction as a byproduct of heading, while our action space explicitly controls the post-move viewing direction. (2) The action space is continuous rather than restricted to discrete primitives [34], enabling fine-grained adjustment with fewer steps. (3) Unlike zero-shot EQA pipelines that rely on pretrained VLMs [23, 15, 35, 24], we cast view refinement as a learnable module and train it on a small synthetic dataset curated specifically for this step.

For training, we curate a synthetic dataset AVS, built on ProcTHOR [7], where each sample contains a query view with partial visual evidence and a target view sufficient for answering, along with a question–answer pair. The dataset covers scenarios from single-step viewpoint adjustments to multi-step refinements through complex spatial layouts. We fine-tune pretrained VLMs in two stages: supervised fine-tuning (SFT) on ground-truth actions and stop decisions, followed by reinforcement learning (RL) optimizing whether the final view supports answering. SFT grounds the model through imitation; RL further refines viewpoint adjustments via reward-driven optimization.

Despite using only a small-scale synthetic dataset of binary existence questions and a 7B-parameter backbone, our method generalizes to real scenes and diverse question types unseen during training. Given the current view and a language query, it outperforms baselines on view refinement, including proprietary models such as GPT-5 [19] and Gemini-2.5-Pro [9]. This refinement capability further translates into improved downstream question-answering accuracy when the learned module is plugged into existing EQA pipelines.

2 Related Work

Embodied Question Answering. There has been substantial effort to tackle embodied question answering (EQA), both in constructing benchmarks [6, 17, 23, 15, 37] and in designing sophisticated methods [23, 15, 35, 24, 38]. Since EQA requires multiple abilities, such as large-scale scene exploration, scene memorization, common sense reasoning, and active perception, prior work has mainly focused either on efficient scene exploration [23, 15, 38] or on better representations for memory construction [37, 35, 24]. In contrast, the crucial link between partial visual observations and fine-grained view selection, that is, how an agent should refine its viewpoint based on existing visual clues, remains underexplored. Our work explicitly targets this computer vision challenge and can be adopted as a plug-in module that fills this missing component in EQA, leading to improved overall performance.

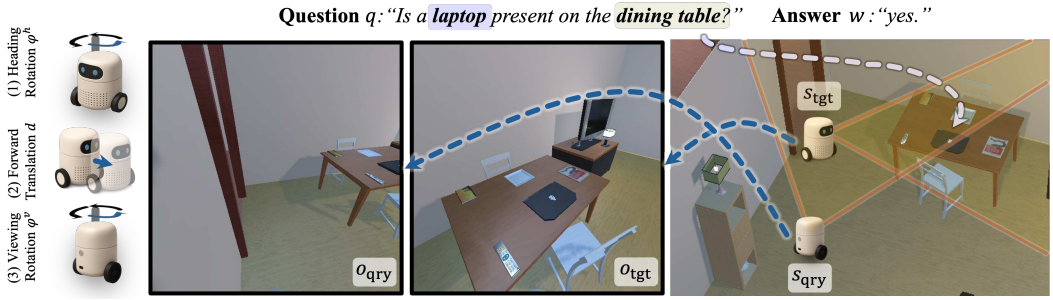


Figure 2: **Our action sequence (left) and AVS dataset sample (right).** The action controls how the agent adjusts its viewpoint. On the right, the target object (*laptop*) is visible only in the target view, while the query view shows only the supporting object (*dining table*), which may serve as a visual clue that motivates active view selection to gather sufficient visual evidence for answering.

Active Perception. Active perception has a long-standing literature that frames perception as an action-driven process, where an agent acquires informative observations through deliberate sensor or viewpoint control [3, 1, 4, 29]. Within this literature, classical next-best-view methods select viewpoints to reduce uncertainty or improve 3D reconstruction coverage [5, 20], and active recognition methods choose views that improve object or scene recognition under partial observability [11, 33]. Beyond these classical methods, recent work has shifted toward leveraging the strong semantic priors of pretrained VLMs for active perception, where the query is specified in natural language, much like an instruction given to a human agent [10, 32, 26, 34]. While these works share our goal of grounding viewpoint selection in language, they largely rely on zero-shot VLM heuristics. In contrast, VG-AVS realizes a learnable policy for language-grounded active perception, explicitly trained for answerability-driven view selection.

Active Visual Question Answering. While active perception centers on physical viewpoint selection, active visual question answering studies how models can seek additional visual evidence before answering. PixelReasoner [27] and ToA [13] operate within the given input through pixel-space reasoning or region selection, while Directional Guidance [14] and V* [31] prompt VLMs to direct visual search at test time. MindJourney [34] generates additional observations with a world model under discrete action primitives, SpatialReasoner [39] renders or zooms into task-relevant views in a 3D scene for scene understanding, and Thinking-in-360° [36] explores active perception through rotation at a fixed location. VG-AVS extends this line to physically grounded 3D environments with a learned policy over a continuous action space spanning both translation and post-move gaze control, rather than relying on image-level operations, generated views, or discrete test-time action search.

3 Visually-Grounded Active View Selection

We study the problem of learning a *view-refinement module* for EQA pipelines: given the agent's current egocentric view and a language query, the module decides whether the view already supports answering or how to adjust the viewpoint to gather the missing evidence. We focus on cases where the current observation contains only *partial* visual clues, and the agent iteratively refines its viewpoint through locomotion and rotation in a continuous 3D action space, terminating once the view becomes answerable.

The module consists of three components: a dedicated training dataset (Section 3.1) and a two-stage (SFT+RL) training strategy with a continuous-action policy formulation (Section 3.2).

3.1 AVS Dataset

We curate a synthetic dataset, called AVS, designed to train models that actively adjust their viewpoints to gather sufficient visual evidence for answering a given question. Each sample is represented as a tuple $(q, w, s_{\text{tgt}}, s_{\text{qry}}, o_{\text{tgt}}, o_{\text{qry}})$, where q denotes the language query, w its corresponding answer, s_{tgt} and s_{qry} indicate the target and query camera poses, and o_{tgt} and o_{qry} are their rendered observations.

The target view o_{tgt} contains sufficient visual information to answer the query, while the query view o_{qry} provides only partial visual evidence, encouraging the model to actively adjust its viewpoint

to reveal the missing information. To capture this contextual difference between the two views within each training sample, we build the AVS dataset using ProcTHOR [7], which provides richly annotated indoor 3D scenes with instance-level object labels and explicit surface–object relations, where certain objects are designated as *supporting* objects that serve as surfaces for other assets, such as countertops holding cups or beds supporting pillows, and these supporting objects could offer visual cues about the unseen *target* object referred to by the question.

View Sampling. Let x_{tgt} and x_{sup} denote the target and supporting objects. We first select these two objects and generate a question–answer pair (q, w) from a predefined template referring to the target. We then sample o_{tgt} and o_{qry} using instance segmentation masks that provide pixel-level visibility for every object. The target view o_{tgt} is sampled so that x_{tgt}'s pixel count exceeds an absolute threshold τ_{tgt} and its projected centroid lies within a small radius δ_{center} from the image center. The query view o_{qry} is sampled so that x_{tgt}'s relative visibility (compared to o_{tgt}) falls below ϵ_{tgt}, while x_{sup}'s relative visibility exceeds ϵ_{sup}. Full sampling criteria and threshold values are listed in Section G of the Appendix. The illustration of the dataset example is in the right of Figure 2.

Staged Samples for Complex Layouts. While many samples can be sufficiently addressed with a single viewpoint adjustment, realistic settings often contain obstacles that prevent direct line-of-sight to the target. To reflect this, we include *staged* samples that prepend a *pre-query view* before the query view. The pre-query view provides a partial glimpse from outside a room, where a doorway serves as a navigational cue. The agent must first move into the room before refining its viewpoint toward the target, encouraging multi-step reasoning in complex spatial layouts.

Dataset Overview. The training set covers 1,689 scenes with binary `Existence` questions in three scenario types: `Partial` ($\epsilon_{\text{tgt}} = 0.3$), where the target is partially visible in the query view; `Invisible` ($\epsilon_{\text{tgt}} = 0$), where the target is fully occluded; and `Staged`, which prepend a pre-query view as described above. The set contains 1,756 direct query-target pairs and 300 staged samples. For evaluation, we construct two additional benchmarks. AVS-ProcTHOR extends beyond *Existence* questions to include *Counting* and *State* questions. AVS-HM3D is built from real-world indoor scenes in the Habitat–Matterport 3D dataset [21]. Details are presented in Section 4.

3.2 AVS Framework

As discussed in Section 3, we cast active view selection as learning a policy that iteratively updates the agent's viewpoint toward the *answerable observation space*.

Formally, we model this as a partially observable Markov decision process in a 3D environment. Let $\mathcal{E}$ denote the underlying 3D environment, which remains fixed within an episode, and let $s_t \in \mathcal{S}$ denote the agent pose at step t. Under a limited field of view, the agent receives a partial egocentric observation $o_t = \Omega(\mathcal{E}, s_t)$. The agent conditions its decisions on the available context c_t at step t, which may include the current observation and prior interactions. At each step, the agent must decide not only how to adjust its viewpoint, but also when to terminate once the available visual evidence is deemed sufficient to answer the language query q. Specifically, given (c_t, q), the policy outputs an action

$$a_t \sim \pi_\theta(\cdot \mid c_t, q), \quad a_t \in \{\texttt{MOVE}(\varphi_t^h, d_t, \varphi_t^v),\ \texttt{STOP}\}. \tag{1}$$

If $a_t = \texttt{MOVE}(\varphi_t^h, d_t, \varphi_t^v)$, the agent executes a motion parameterized by three continuous components, which we detail below. The pose is updated via $s_{t+1} = \mathcal{T}(s_t, a_t)$ and yields a new observation $o_{t+1} = \Omega(\mathcal{E}, s_{t+1})$. If $a_t = \texttt{STOP}$, the agent terminates the episode, indicating that the available evidence is sufficient to answer the question.

Action Space Design. We represent the agent pose at step t as $s_t = (x_t, y_t, \varphi_t)$, where (x_t, y_t) denotes the agent's 2D position and φ_t is the azimuth orientation, with height and elevation fixed for simplicity. Our `MOVE` action follows the natural sequence of embodied movement and is parameterized by three continuous components: a **heading rotation** φ_t^h that sets the movement direction relative to φ_t, a **forward translation** d_t along the heading, and a **view rotation** φ_t^v that adjusts the final viewing direction after moving. Positive angles indicate right turns. The transition $\mathcal{T}(s_t, a_t)$ translates the agent by d_t along the heading $\varphi_t + \varphi_t^h$ and updates the orientation to $\varphi_{t+1} = \varphi_t + \varphi_t^h + \varphi_t^v$. Conversely, given consecutive states s_t and s_{t+1}, the ground-truth action $\hat{a}_t$ is analytically computed from view geometry (full equations in Section E of the Appendix). See the left of Figure 2 for an illustration of the action space.

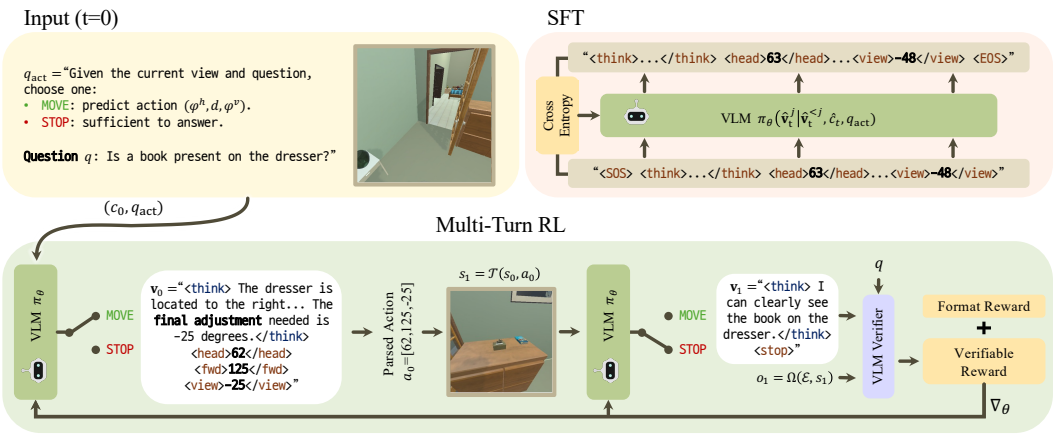


Figure 3: **Overview of training strategies.** Given the context c_t and a question q_{act}, the model produces a reasoning trace followed by an action prediction at each step. In SFT, the model is trained to match ground-truth reasoning and actions via teacher forcing. In RL, the model generates its own reasoning and actions over a multi-step rollout, and is optimized so that trajectories leading to higher rewards become more likely.

Supervised Fine-Tuning. Under the problem formulation discussed in Section 3.2, the most straightforward approach is to adopt a teacher-forcing objective for learning the policy, where the model is supervised to predict the ground-truth action given the current context and the language query.

Note that such supervised fine-tuning is feasible in our setup since our synthetic dataset provides view-level supervision with analytically computable ground-truth actions in a visually-grounded setting. In contrast, EQA setups typically lack such supervision because long-horizon navigation admits many plausible trajectories and step-level actions are often ambiguous.

Specifically, we parameterize the policy with a VLM. For supervised fine-tuning, we construct a target token sequence $\hat{\mathbf{v}}_t$ at step t that includes a textual reasoning trace followed by an action string $\hat{\mathbf{y}}_t$: $\hat{\mathbf{v}}_t = \texttt{<think>...</think>}\ \hat{\mathbf{y}}_t$. The action string $\hat{\mathbf{y}}_t$ is formatted as

$$\hat{\mathbf{y}}_t = \begin{cases} \texttt{<stop>}, & \text{if } \hat{a}_t = \texttt{STOP}, \\ \texttt{<H>str}(\hat{\varphi}_t^h)\texttt{</H><D>str}(\hat{d}_t)\texttt{</D><V>str}(\hat{\varphi}_t^v)\texttt{</V>}, & \text{otherwise.} \end{cases} \tag{2}$$

Here $\texttt{str}(\cdot)$ maps real-valued numbers to their string representations, and <H>, <D>, and <V> are special tags to delimit each scalar component.

At step t, the teacher-forcing objective is expressed as follows:

$$\mathcal{L}_{\text{SFT}} = -\sum_{j=1}^{|\hat{\mathbf{v}}_t|} \log \pi_\theta\left(\hat{\mathbf{v}}_t^j \mid \hat{\mathbf{v}}_t^{<j},\ \hat{c}_t,\ q_{\text{act}}\right), \tag{3}$$

where q_{act} is the action instruction and $\hat{c}_t = \{o_{1:t}, \hat{\mathbf{v}}_{1:t-1}\}$ is the multimodal context containing the observation history up to o_t and the ground-truth outputs from previous steps. While SFT enables the model to learn which action to predict from a multimodal context, it confines the VLM to the provided expert demonstrations.

Reinforcement Learning. An alternative to supervised imitation is reinforcement learning (RL), which optimizes the policy directly from trajectory-level feedback without requiring ground-truth action labels. Rather than matching expert demonstrations, the agent samples its own trajectory and is rewarded based on whether the terminal observation supports answering the query.

Concretely, at each step t, the model samples an output token sequence $\mathbf{v}_t$, from which the executable action a_t is parsed. Starting from the initial state s_0, the agent sequentially executes parsed actions $a_0, \ldots, a_{T'-1}$ until it either predicts a STOP action or reaches the maximum number of steps T, yielding a terminal observation $o_{T'} = \Omega(\mathcal{E}, s_{T'})$ where $T' \leq T$.

We define two reward functions. A format reward r^{fmt} encourages the model to produce correctly structured action strings throughout the trajectory. A verifier reward r^{ver} evaluates whether the

terminal observation contains sufficient evidence to answer the query, using a frozen pre-trained VLM v_ϕ as an external verifier:

$$r^{\text{ver}}(o, q, w) = \begin{cases} 1, & \text{if } v_\phi(o, q) = w, \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where (q, w) is the question–answer pair and o denotes the terminal observation. The RL objective then maximizes the expected cumulative reward over the sampled trajectory:

$$\max_\theta \; \mathbb{E}_{\tau \sim \pi_\theta} \left[r^{\text{fmt}}(\tau) + r^{\text{ver}}\big(o_{T'}, q, w\big) \right], \tag{5}$$

where $\tau = (o_0, \mathbf{v}_0, o_1, \mathbf{v}_1, \ldots, o_{T'})$ denotes the rollout trajectory, with each $\mathbf{v}_t \sim \pi_\theta(\cdot \mid c_t, q_{\text{act}})$ sampled conditioned on the context c_t at step t. We use Group Relative Policy Optimization (GRPO) [25] for policy optimization.

Bridging SFT and RL. While SFT enables the model to efficiently learn plausible actions from visual–linguistic inputs through explicit supervision from paired query–target viewpoints in our dataset, we empirically observe that SFT alone quickly saturates and provides limited further gains once the basic mapping from observations to actions is learned. Conversely, training with RL from scratch tends to underperform the SFT-trained model and is less stable in the continuous action space, even with carefully designed rewards.

We find that combining the two in a staged manner is effective: SFT warm-up provides a good initialization that captures plausible action magnitudes and directions, while subsequent RL fine-tuning brings additional improvements by allowing the model to explore and refine its viewpoint adjustments over multiple steps, optimizing directly for whether the terminal observation supports answering the query. This two-stage strategy consistently outperforms using either SFT or RL alone.

4 Experiments

See the Appendix for more training details, extended comparisons, and cross-task generalization.

4.1 Experiments on VG-AVS

Setup. We evaluate on two benchmark datasets for the Visually-Grounded Active View Selection (VG-AVS) task: AVS-ProcTHOR and AVS-HM3D. AVS-ProcTHOR is automatically generated following the data curation pipeline introduced in Section 3.1 and covers three scenario types: `Partial`, `Invisible`, and `Staged`. It also includes more diverse question types than the training set, with 381, 321, and 364 samples for `Existence`, `Counting`, and `State`, respectively. AVS-HM3D consists of real indoor scenes constructed from triplets of (question, answer, ground-truth view) in the validation split of Fine-EQA [15]. Due to the absence of object visibility checks in real scenes, we first generate query views by randomly perturbing the ground-truth viewpoints so that the target object becomes invisible, and then manually curate 191 samples where the query view still provides sufficient contextual visual clues. AVS-HM3D spans five question types: `Existence`, `Counting`, `State`, `Attribute`, and `Object`. See Section G of the Appendix for more details on data construction and statistics.

We report VQA accuracy on AVS-ProcTHOR, where questions are either multiple-choice or open-ended (`Counting`) with exact-match evaluation. For AVS-HM3D, which consists of open-ended questions, we use LLM-Match [17], an LLM-based correctness metric where an LLM assigns a discrete score from 1 to 5 by comparing the predicted answer with the annotated ground-truth answer. We use Gemini-2.5-Flash [9] for VQA accuracy and GPT-4 [18] for LLM-Match following prior work [17]. We set the maximum number of refinement steps to T=4. For fair comparison, all models including baselines are evaluated under the same rollout budget.

Comparison with Baselines. We present a quantitative comparison in Table 1. To provide reference bounds that reflect the difficulty of the task, we report scores when the VLM verifier is directly given the query view (lower bound) and the target view (upper bound). For all action models, we feed the final view after action rollout to the verifier.

Open-source VLMs, including Qwen2.5-VL-3B/7B [2] and spatial VLMs such as ViLaSR [30] and SpatialReasoner [16], all fail to achieve meaningful active view selection, and Fine-EQA [15]—an

Table 1: **Quantitative comparison on our AVS benchmark.** We report VQA accuracy on AVS-ProcTHOR and LLM-Match scores on AVS-HM3D, normalized to a percentage scale. The best except for *'No Action'* in each column is in **bold** and second best is <u>underlined</u>.

Action Model		AVS-ProcTHOR [7]				AVS-HM3D [21]					
		Exist	Count	State	Avg.	Exist	Count	State	Attr.	Obj.	Avg.
No Action	Query View	65.62	12.15	55.49	44.42	70.00	56.15	54.24	69.52	48.33	59.65
	Target View	92.91	70.72	90.38	84.67	75.24	89.23	78.25	80.95	79.44	80.62
Open-Source VLMs	Qwen2.5-VL-3B [2]	68.50	17.13	56.87	47.50	66.67	50.00	52.73	57.14	44.44	54.20
	Qwen2.5-VL-7B [2]	64.57	29.91	56.87	50.45	64.76	46.15	67.27	46.67	46.86	54.34
Spatial VLMs	ViLaSR [30]	65.62	24.30	55.22	48.38	68.10	48.46	50.30	51.43	50.00	53.66
	SpatialReasoner [16]	67.19	16.51	51.59	45.10	70.00	50.00	47.88	36.19	40.00	48.81
EQA Pipeline	Fine-EQA [15]	69.48	20.91	55.01	48.47	70.95	52.31	53.03	55.24	44.44	55.19
Proprietary VLMs	GPT-5 [19]	80.84	48.29	**75.82**	68.32	**77.14**	54.62	66.67	60.95	60.00	63.88
	Gemini2.5-Pro [9]	81.36	46.73	<u>75.55</u>	67.88	<u>76.19</u>	59.23	60.91	59.05	60.00	63.08
AVS Framework (Backbone: Qwen2.5-VL-7B [2])	SFT	<u>83.46</u>	<u>51.71</u>	71.98	<u>69.05</u>	66.67	45.38	<u>68.48</u>	<u>70.48</u>	52.78	60.76
	RL	78.74	42.68	61.54	60.99	72.38	**72.31**	64.85	64.76	**66.67**	<u>68.19</u>
	SFT+RL (Ours)	**83.73**	**52.02**	72.53	**69.43**	73.33	<u>66.15</u>	**73.33**	**80.95**	<u>60.56</u>	**70.87**

EQA pipeline aimed at long-horizon navigation—also performs significantly worse on our benchmark, suggesting that its coarse, discrete actions cannot deliver fine-grained viewpoint control. With the same Qwen2.5-VL-7B backbone, our AVS framework consistently outperforms these models, and even surpasses much larger proprietary models including GPT-5 [19] and Gemini2.5-Pro [9] with SFT+RL despite using only a 7B-parameter backbone. This highlights that fine-grained view refinement requires explicit training on this specific component, rather than relying solely on large-scale pretraining.

Training Strategy Analysis. Across different training strategies within our AVS framework, all variants outperform other approaches, while the two-stage training scheme (SFT+RL) yields a further improvement of VQA accuracy compared to either method alone. Moreover, the trained models show strong generalizability to diverse question types that are not seen during training, including `Counting` and `State` questions.

Since our AVS module is orthogonal to the choice of backbone, our framework also scales down: the same SFT+RL recipe applied to a smaller Qwen2.5-VL-3B backbone [2] reaches 62.17% on AVS-ProcTHOR, +14.67%pt over the 3B zero-shot baseline and surpassing the 7B RL-only variant (Section A.1 of the Appendix). This suggests that stronger VLMs could further benefit from our learned view-refinement module.

Generalization to Real Scenes. Beyond the synthetic scenes from ProcTHOR [7], the results on AVS-HM3D reported in Table 1 demonstrate that our AVS framework generalizes well to real indoor environments with diverse question types. Despite being trained only on a small-scale synthetic 3D dataset with binary object existence questions, our approach outperforms all baselines by clear margins, achieving 70.87 with SFT+RL compared to 63.88 with GPT-5 [19] in average score. This shows that a well-curated dataset that compactly supervises active view selection is sufficient to enable strong transfer to real-world scenes, even under shifts in both scene distribution and question format. Furthermore, evaluating the trained model on out-of-domain spatial reasoning and visual perception benchmarks (SAT [22], BLINK [8], CV-Bench [28]) shows performance on par with or above the backbone, indicating no significant catastrophic forgetting (Section D of the Appendix).

Rollout Analysis. Our multi-step formulation enables the model to adapt to scene complexity. On average, it takes 2.24 steps for `Partial`, 2.30 for `Invisible`, and 3.39 for `Staged` scenarios, indicating that the model adaptively allocates more refinement steps for more challenging scenarios. Section A.2 of the Appendix further shows that our SFT+RL model consistently outperforms GPT-5 across all $T \in \{1, 2, 3, 4\}$ at inference time, with both improving as T increases.

Stop Decision Analysis. To further verify the stop decision behavior, we feed unseen answerable target views of AVS-ProcTHOR as initial observations. In this setting, the model predicts `STOP` in 85.3% of cases, achieving 90.75% VQA accuracy compared to the oracle upper bound of 92.91%.

Figure 4: **Qualitative results on the Fine-EQA benchmark [15].** Blue highlights denote the object of interest. ✓ corresponds to high LLM-Match scores ($= 5$) and ✗ to low scores (≤ 2).

This confirms its ability to recognize when the current view already contains sufficient evidence without further movement.

4.2 Application to EQA

Setup. Our AVS framework can naturally serve as a plug-and-play component to enhance existing EQA pipelines. Since EQA inherently requires multi-task abilities, prior work [17, 15, 6, 23] has largely underexplored the final step of fine-grained view refinement once the agent stops near the target, which is crucial for gathering the visual evidence needed to answer the question. Our AVS framework directly complements this stage: it takes the terminal view produced by the EQA policy, refines the viewpoint, and feeds the refined observation into the VLM verifier for answering. We evaluate on the Fine-EQA [15] and OpenEQA [17] benchmark and report LLM-Match scores.

Table 2: **Comparison on EQA benchmarks.** We report normalized LLM-Match scores on Fine-EQA [15] and OpenEQA. Plugging our AVS framework into the baseline method [15] improves overall EQA performance. In each column, best in **bold**, second best underlined.

Method	Fine-EQA Avg.	OpenEQA Avg.
Fine-EQA [15]	52.94	44.08
w/ SFT	<u>55.96</u>	41.70
w/ RL	55.01	<u>46.37</u>
w/ SFT + RL (Ours)	**60.48**	**46.57**

Results. As shown in Table 2, our AVS plug-in improves performance on both EQA benchmarks. On Fine-EQA [15], it increases the average normalized LLM-Match score from 52.94 to 55.96 with SFT, 55.01 with RL, and 60.48 when combining SFT and RL. The same plug-in also generalizes to OpenEQA [17] benchmark, improving the LLM-Match score from 44.08 to 46.57. Beyond these benchmark-level results, we further apply our module to another EQA pipeline, 3D-Mem [35], where it improves performance from 54.71 to 57.84. The detailed results are provided in Section C of the Appendix. These results suggest that our AVS module can provide complementary gains across different EQA settings, rather than being specific to a single benchmark or upstream pipeline.

4.3 Qualitative Results

VG-AVS. As shown in Figure 5, our model handles varying spatial complexity—from single-step refinement on partially visible objects (row 1) to multi-step refinement through doorways into adjacent rooms (row 2)—and transfers to real HM3D scenes (rows 5–6). Refer to Section J of the Appendix for representative rollouts under strong occlusions or distant targets, and Section B for additional trajectories from captured real scenes.

EQA. As shown in Figure 4, when the final view produced by the base EQA framework is insufficient for answering a question, our module successfully identifies the region of interest and adjusts the viewpoint to obtain a more informative observation. For example, in the top-right example, only

Figure 5: **Qualitative results in AVS-ProcTHOR (top four rows) and AVS-HM3D (bottom two rows).** Blue denotes the object of interest, and gray denotes surrounding objects that may serve as visual clues for reasoning. ✓ corresponds to correct answers (or LLM-Match scores $= 5$), ✗ corresponds to wrong answers (or LLM-Match scores ≤ 2). The red arrow in the final column marks the region of interest.

the edge of the television is visible in the initial view; our model adjusts the viewing direction to reveal the full television, enabling the verifier to answer correctly.

5 Conclusion

We formalize Visually-Grounded Active View Selection (VG-AVS) as an underdeveloped component of EQA pipelines, refining the agent's viewpoint near the target so that the final observation supports answering. Unlike prior pipelines that handle this step with zero-shot VLMs, we make it learnable through a curated synthetic dataset and an SFT+RL framework. Despite being trained on a modest dataset with a 7B backbone, VG-AVS learns continuous movement and gaze control, generalizes to unseen synthetic and real scenes, improving existing EQA pipelines as a plug-in module.

Limitations. Our learnable formulation targets near-target view refinement, complementing upstream navigation policies responsible for scene-scale exploration. Scaling supervision to longer-horizon planning and outdoor scenes, as well as developing supervision beyond the current VLM verifier, remain promising directions for future work.

References

[1] J. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. *International journal of computer vision*, 1(4):333–356, 1988.

[2] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[3] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[4] D. H. Ballard. Animate vision. *Artificial Intelligence*, 1991.

[5] C. Connolly. The determination of next best views. In *Proceedings. 1985 IEEE international conference on robotics and automation*, volume 2, pages 432–435. IEEE, 1985.

[6] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra. Embodied question answering. In *CVPR*, 2018.

[7] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, J. Salvador, K. Ehsani, W. Han, E. Kolve, A. Farhadi, A. Kembhavi, and R. Mottaghi. ProcTHOR: Large-Scale Embodied AI Using Procedural Generation. In *NeurIPS*, 2022.

[8] X. Fu, Y. Hu, B. Li, Y. Feng, H. Wang, X. Lin, D. Roth, N. A. Smith, W.-C. Ma, and R. Krishna. Blink: Multimodal large language models can see but not perceive. In *ECCV*, 2024.

[9] Google. Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities, 2025.

[10] Y. Huang, Z. Wang, W. Tang, C. Lu, and P. Cai. I-perceive: A foundation model for active perception with language instructions. *arXiv preprint arXiv:2603.00600*, 2026.

[11] D. Jayaraman and K. Grauman. Learning to look around: Intelligently exploring unseen environments for unknown tasks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1238–1247, 2018.

[12] S. Karamcheti, S. Nair, A. Balakrishna, P. Liang, T. Kollar, and D. Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *ICML*, 2024.

[13] M. Liang, Y. Wu, et al. Toa: Task-oriented active vqa. In *NeurIPS*, 2023.

[14] L. Liu, D. Yang, S. Zhong, K. S. S. Tholeti, L. Ding, Y. Zhang, and L. Gilpin. Right this way: Can vlms guide us to see more to answer questions? In *NeurIPS*, 2024.

[15] Y. Liu, Y. Zhang, C. Wang, and H. Li. EXPRESS-Bench: Beyond the destination: A novel benchmark for exploration-aware embodied question answering. In *ICCV*, 2025.

[16] W. Ma, Y.-C. Chou, Q. Liu, X. Wang, C. de Melo, J. Xie, and A. Yuille. Spatialreasoner: Towards explicit and generalizable 3d spatial reasoning. In *NeurIPS*, 2025.

[17] A. Majumdar, A. Ajay, X. Zhang, P. Putta, S. Yenamandra, M. Henaff, S. Silwal, P. Mcvay, O. Maksymets, S. Arnaud, et al. OpenEQA: Embodied question answering in the era of foundation models. In *CVPR*, 2024.

[18] OpenAI. Gpt-4 technical report, 2023.

[19] OpenAI. Introducing gpt-5, 2025.

[20] X. Pan, Z. Lai, S. Song, and G. Huang. Activenerf: Learning where to see with uncertainty estimation. In *European Conference on Computer Vision*, pages 230–246. Springer, 2022.

[21] S. K. Ramakrishnan, A. Gokaslan, E. Wijmans, O. Maksymets, A. Clegg, J. M. Turner, E. Undersander, W. Galuba, A. Westbury, A. X. Chang, M. Savva, Y. Zhao, and D. Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[22] A. Ray, J. Duan, E. Brown, R. Tan, D. Bashkirova, R. Hendrix, K. Ehsani, A. Kembhavi, B. A. Plummer, R. Krishna, et al. Sat: Dynamic spatial aptitude training for multimodal language models. 2025.

[23] A. Z. Ren, J. Clark, A. Dixit, M. Itkina, A. Majumdar, and D. Sadigh. Explore until confident: Efficient exploration for embodied question answering. In *RSS*, 2024.

[24] S. Saxena, B. Buchanan, C. Paxton, P. Liu, B. Chen, N. Vaskevicius, L. Palmieri, J. Francis, and O. Kroemer. Grapheqa: Using 3d semantic scene graphs for real-time embodied question answering. In *CoRL*, 2025.

[25] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[26] V. Sripada, S. Carter, F. Guerin, and A. Ghalamzan. Scene exploration by vision-language models. *arXiv preprint arXiv:2409.17641*, 2025.

[27] A. Su, H. Wang, W. Ren, F. Lin, and W. Chen. Pixel reasoner: Incentivizing pixel-space reasoning with curiosity-driven reinforcement learning. In *NeurIPS*, 2025.

[28] S. Tong, E. Brown, P. Wu, S. Woo, M. Middepogu, S. C. Akula, J. Yang, S. Yang, A. Iyer, X. Pan, A. Wang, R. Fergus, Y. LeCun, and S. Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. In *NeurIPS*, 2024.

[29] J. Tsotsos and A. Rothenstein. Computational models of visual attention. *Scholarpedia*, 2011.

[30] J. Wu, J. Guan, K. Feng, Q. Liu, S. Wu, L. Wang, W. Wu, and T. Tan. Reinforcing spatial reasoning in vision-language models with interwoven thinking and visual drawing. *arXiv preprint arXiv:2506.09965*, 2025.

[31] P. Wu and S. Xie. V?: Guided visual search as a core mechanism in multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13084–13094, 2024.

[32] J. Yang, Y. Han, Y. Xie, N. Guo, and W. Lian. Look, zoom, understand: The robotic eyeball for embodied perception. *arXiv preprint arXiv:2511.15279*, 2026.

[33] J. Yang, Z. Ren, M. Xu, X. Chen, D. J. Crandall, D. Parikh, and D. Batra. Embodied amodal recognition: Learning to move to perceive objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2040–2050, 2019.

[34] Y. Yang, J. Liu, Z. Zhang, S. Zhou, R. Tan, J. Yang, Y. Du, and C. Gan. Mindjourney: Test-time scaling with world models for spatial reasoning. In *NeurIPS*, 2025.

[35] Y. Yang, H. Yang, J. Zhou, P. Chen, H. Zhang, Y. Du, and C. Gan. 3D-Mem: 3d scene memory for embodied exploration and reasoning. In *CVPR*, 2025.

[36] H. Yu, Y. Han, X. Zhang, B. Yin, B. Chang, X. Han, X. Liu, J. Zhang, M. Pavone, C. Feng, et al. Thinking in 360°: Humanoid visual search in the wild. *arXiv preprint arXiv:2511.20351*, 2025.

[37] M. Zhai, Z. Gao, Y. Wu, and Y. Jia. Memory-centric embodied question answering. *arXiv preprint arXiv:2505.13948*, 2025.

[38] H. Zhang, N. Savaliya, F. Siddiqui, and E. Sachdeva. Fast-eqa: Efficient embodied question answering with global and local region relevancy. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1664–1673, 2026.

[39] H. Zheng, S. Li, Y. Li, and H. Yin. Spatialreasoner: Active perception for large-scale 3d scene understanding. *arXiv preprint arXiv:2512.03284*, 2025.

Table 3: **Quantitative comparison on internal baselines.** We report VQA accuracy on AVS-ProcTHOR and LLM-Match scores on AVS-HM3D, normalized to a percentage scale. The best in each column is in **bold** and second best is underlined.

Action Model		AVS-ProcTHOR [7]				AVS-HM3D [21]					
		Exist	Count	State	Avg.	Exist	Count	State	Attr.	Obj.	Avg.
Open-Source VLMs	Qwen2.5-VL-3B [2]	68.50	17.13	56.87	47.50	66.67	50.00	52.73	57.14	44.44	54.20
	Qwen2.5-VL-7B [2]	64.57	29.91	56.87	50.45	64.76	46.15	67.27	46.67	46.86	54.34
AVS Framework	SFT	83.46	51.71	71.98	69.05	66.67	45.38	68.48	70.48	52.78	60.76
	SFT (Extended)	78.22	41.43	71.98	63.88	74.29	56.15	66.97	68.57	55.56	64.31
	RL	78.74	42.68	61.54	60.99	72.38	**72.31**	64.85	64.76	66.67	68.19
	RL w/ r^{pose}	77.69	37.07	63.46	59.41	**77.14**	56.15	68.79	71.67	**69.01**	68.55
	SFT+RL (3B)	81.63	38.94	65.93	62.17	74.76	70.77	67.88	51.00	53.89	63.66
	SFT+RL	**83.73**	**52.02**	**72.53**	**69.43**	73.33	66.15	**73.33**	**80.95**	60.56	**70.87**

A Extended Experiments

A.1 Comparison with More Training Variants

As discussed in Section 4.1, we further evaluate two internal variants for a comprehensive comparison of training strategies. The quantitative results are summarized in Table 3. Below, we detail each variant and its training setup:

- **SFT (extended)**: to fairly compare against our two-stage training, we extend SFT training until convergence (from 20 to 40 epochs) so that the only difference lies in the training objective. We observe that longer training with SFT alone does not gain further improvement on AVS-ProcTHOR, still underperforming our SFT+RL strategy.
- **RL w/ r^{pose}**: this variant incorporates an additional distance-based reward inversely proportional to the discrepancy between the predicted and target camera poses (position and orientation), thereby injecting weak supervision of the ground-truth action into RL. Nevertheless, it still fails to learn meaningful actions without the SFT warm-up stage, highlighting that a supervised initialization followed by RL is more effective than relying on more sophisticated reward designs for RL alone.
- **AVS-3B**: to assess the scalability of our framework with respect to backbone capacity, we additionally train AVS with a smaller Qwen2.5-VL-3B [2] backbone using our best-performing SFT+RL strategy, denoted as SFT+RL (3B). Despite the reduced model capacity, it achieves 62.17% on AVS-ProcTHOR, outperforming not only the 3B backbone by a substantial margin (+14.67%) but also the RL-only variants of the 7B model, demonstrating that our training framework generalizes well across model scales.

A.2 Results with Varying Maximum Rollout Steps

In Section 4, we evaluate with $T = 4$, matching the maximum rollout step used in training. Here, we vary the maximum rollout steps at inference from 1 to 4 and report performance at each step on AVS-ProcTHOR. As shown in Figure 6, our SFT+RL model consistently outperforms GPT-5 [19] across all values of T with both models improving as T increases.

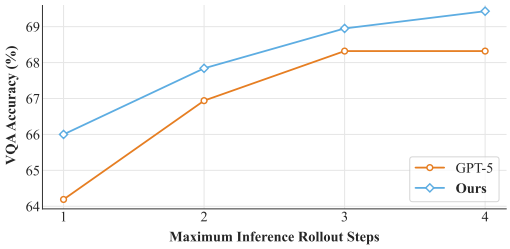


Figure 6: **Maximum Rollout Turn Ablation.** Average accuracy on AVS-ProcTHOR with maximum rollout turns from 1 to 4.

A.3 Evaluation with Target Object Visibility

In Table 4, we additionally report the target object's relative visibility as a verifier-free evaluation metric. Following Equation 9, we measure the ratio of target-object pixels in the resulting view to those in the target view. The results show that our AVS framework outperforms the baselines.

Table 4: **Quantitative comparison on AVS-ProcTHOR.** We report target object visibility (%), indicating pixel ratio of target object relative with ground-truth target view. Best except for *No Action* is in **bold** and second best is underlined.

Model	Exist	Count	State	Avg.
Query View	0.00	0.00	0.00	0.0
Target View	100.00	100.00	100.00	100.00
Qwen2.5-VL-3B	8.05	9.19	9.79	9.01
Qwen2.5-VL-7B	11.34	17.31	14.74	14.46
ViLaSR	15.06	18.42	14.98	16.15
SpatialReasoner	11.50	12.01	11.02	11.51
Fine-EQA	12.73	1.52	10.84	8.36
GPT-5	23.53	28.63	29.40	27.19
Gemini2.5-Pro	29.25	30.15	28.90	29.43
SFT	**53.51**	55.36	52.73	**53.87**
RL	24.67	27.79	22.87	25.11
SFT+RL (Ours)	51.84	**55.44**	**53.85**	53.71

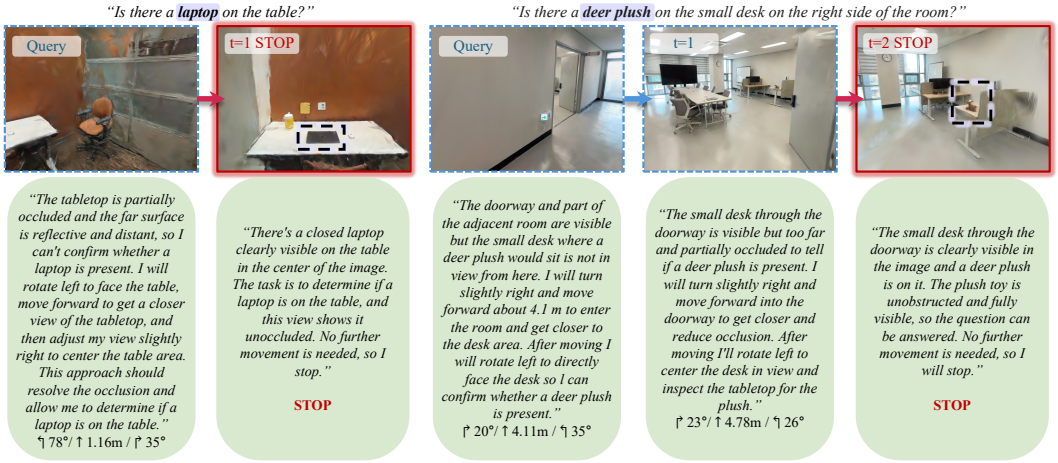


Figure 7: **Examples from captured real scenes.**

B Demo in Captured Real Scenes

Beyond AVS-HM3D and the EQA benchmarks, we further evaluate our AVS framework on captured real-world scenes to assess its generalization. As shown in Figure 7, our method transfers effectively to real-world environments and can handle complex layouts, as illustrated by the second example, where the agent moves from a corridor into a room and then refines its viewpoint within the room.

C More Experiments on EQA

In the main paper, we report the average EQA results on Fine-EQA, OpenEQA, and 3D-Mem. Here, we provide the full per-category results on Fine-EQA and OpenEQA, together with the detailed 3D-Mem results, to complement the aggregate comparison in the main paper.

C.1 Full Results on Fine-EQA and OpenEQA benchmarks

Table 5 provides the full results on both Fine-EQA [15] and OpenEQA [17]. We adopt the Fine-EQA framework as the base EQA pipeline, following the configuration used in Section 4.2, and incorporate our AVS framework on top. On Fine-EQA, AVS improves the average normalized LLM-Match score from 52.94 to 55.96 with SFT, 55.01 with RL, and 60.48 when combining SFT and RL. On OpenEQA, the same AVS plug-in improves the average LLM-Match score from 44.08 to 46.57, corresponding to a 2.49 percentage-point gain over the Fine-EQA baseline. These results

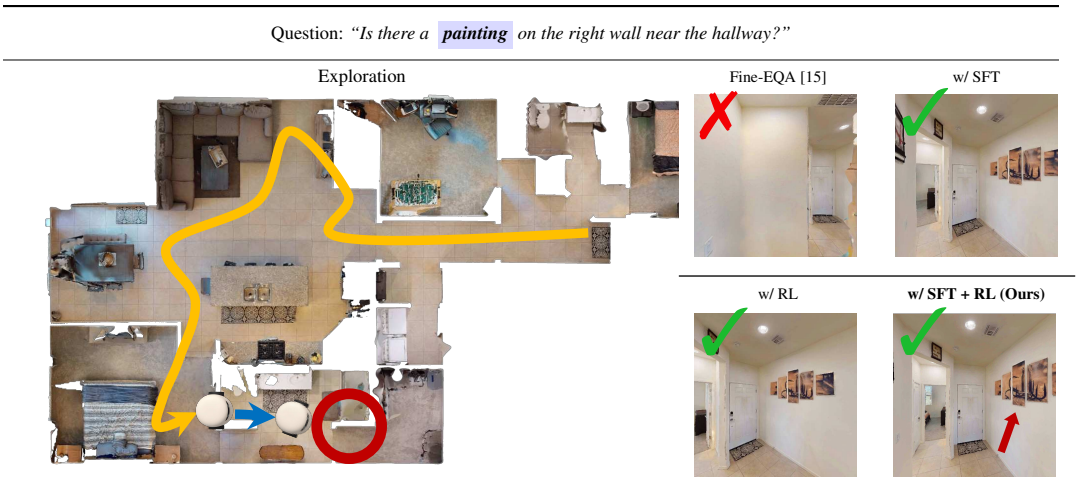


Figure 8: **Illustration of integrating our AVS framework into an EQA pipeline.** On the left, the yellow line represents the exploration path of the EQA pipeline, the blue arrow the final viewpoint refinement by our model, and the red circle the region of interest given the question. On the right, we show the views obtained by each method, where ✓ denotes a correct view with a high LLM-Match score ($= 5$) and ✗ an incorrect view with a low score (≤ 2). Given the question about the *painting*, our AVS framework effectively refines the viewpoint such that the final predicted view reveals the painting on the wall. This demonstrates that the training-free EQA pipeline lacks query-conditioned final viewpoint refinement, whereas our learning-based method can minimally adjust the viewpoint to obtain an answerable view.

Table 5: **Comparison on Fine-EQA and OpenEQA benchmarks.** Each column reports normalized LLM-Match scores for different question types. Plugging our AVS framework into an existing EQA pipeline [15] improves performance. Best in **bold**, second best underlined.

Fine-EQA Benchmark							
Method	**Attribute**	**Counting**	**Existence**	**Object**	**State**	**Location**	**Avg.**
Fine-EQA [15]	49.23	38.75	**67.92**	57.41	**63.20**	41.14	52.94
w/ SFT	55.63	49.38	63.13	50.37	60.81	56.47	55.96
w/ RL	**55.94**	61.88	66.46	46.30	54.20	45.20	55.01
w/ SFT + RL (Ours)	51.38	**66.25**	60.00	**69.26**	55.40	**60.57**	**60.48**
OpenEQA Benchmark							
Method	**Object Recognition**	**Spatial Understanding**	**Object State Recognition**	**Attribute Recognition**	**Object Localization**	**Functional Reasoning**	**Avg.**
Fine-EQA [15]	40.00	45.26	51.11	52.73	36.57	38.82	44.08
w/ SFT	32.00	35.79	49.63	**53.94**	**40.00**	38.82	41.70
w/ RL	**41.60**	50.53	**58.52**	52.73	34.86	40.00	46.37
w/ SFT + RL (Ours)	34.40	**51.58**	54.81	51.52	37.71	**49.41**	**46.57**

Table 6: **Comparison on Fine-EQA benchmark, based on 3D-Mem [35] framework.** Each column reports normalized LLM-Match scores for different question types. Plugging our AVS framework into an exiting EQA pipeline [35] improves performance. Best in **bold**, second best underlined.

Method	**Attribute**	**Counting**	**Existence**	**Object**	**State**	**Location**	**Avg.**
3D-Mem [35]	53.54	51.25	65.42	**55.93**	62.20	40.57	54.71
w/ SFT	**66.56**	43.75	67.50	47.04	57.00	51.18	55.50
w/ RL	57.50	**56.88**	58.96	45.93	62.20	**64.12**	57.60
w/ SFT+RL (Ours)	61.23	54.38	**71.04**	51.85	**67.40**	41.14	**57.84**

show that AVS consistently improves the base EQA pipeline across both benchmarks, while the per-category breakdowns further reveal where the gains come from.

We also illustrate how our AVS framework is integrated into the existing EQA pipeline in Figure 8.

Table 7: **Results on Out-of-Domain benchmark.** We report three benchmark results: SAT (Synthetic, Real), BLINK, and CV-Bench.

Benchmark	Backbone	AVS Framework		
	Qwen2.5-VL-7B [2]	SFT	RL	SFT + RL (Ours)
SAT-Synthetic [22]	61.3	67.9 (+6.6)	59.9 (-1.4)	67.6 (+6.3)
SAT-Real [22]	59.8	79.1 (+19.3)	59.8 (+0.0)	80.5 (+20.7)
BLINK [8]	55.9	56.6 (+0.7)	56.0 (+0.1)	56.6 (+0.7)
CV-Bench [28]	77.2	76.9 (-0.3)	77.4 (+0.2)	76.7 (-0.5)

C.2 Integration with 3D-Mem

To further verify its general applicability, we integrate AVS with another recent EQA pipeline, 3D-Mem [35]. As shown in Table 6, AVS improves the score from 54.71 to 57.84.

D Cross-Dataset Generalization

To assess whether training on our relatively small-scale dataset harms the general spatial understanding of Qwen2.5-VL-7B [2], we evaluate our models on three additional benchmarks: SAT [22], a dedicated spatial reasoning benchmark covering diverse tasks across synthetic and real-image splits, and BLINK [8] and CV-Bench [28], two general visual perception benchmarks.

As shown in Table 7, although our framework is not explicitly trained for these benchmarks, performance remains largely on par with or better than the backbone across most settings, with no significant degradation observed. These results indicate that training on our synthetic dataset does not cause noticeable catastrophic forgetting, and the learned representations transfer reasonably well to broader spatial reasoning tasks.

E Action Space Details

We provide the full equations for the state transition and ground-truth action computation introduced in Section 3.2. With agent pose $s_t = (x_t, y_t, \varphi_t)$ and action $a_t = \texttt{MOVE}(\varphi_t^h, d_t, \varphi_t^v)$, the state transition is:

$$s_{t+1} = \begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \varphi_{t+1} \end{pmatrix} = \mathcal{T}(s_t, a_t) = \begin{pmatrix} x_t + d_t \sin(\varphi_t + \varphi_t^h) \\ y_t + d_t \cos(\varphi_t + \varphi_t^h) \\ \varphi_t + \varphi_t^h + \varphi_t^v \end{pmatrix}. \tag{6}$$

Given consecutive states s_t and s_{t+1}, the ground-truth action $\hat{a}_t$ is analytically computed as:

$$\hat{a}_t = \begin{pmatrix} \hat{\varphi}_t^h \\ \hat{d}_t \\ \hat{\varphi}_t^v \end{pmatrix} = \begin{pmatrix} \operatorname{atan2}(\Delta x_t, \Delta y_t) - \varphi_t \\ \sqrt{(\Delta x_t)^2 + (\Delta y_t)^2} \\ \varphi_{t+1} - (\varphi_t + \hat{\varphi}_t^h) \end{pmatrix}, \tag{7}$$

where $\Delta x_t = x_{t+1} - x_t$ and $\Delta y_t = y_{t+1} - y_t$ denote the positional differences.

F Training and Experiment Setup Details

Training. We use Qwen2.5-VL-7B [2] as the backbone VLM, keeping the vision encoder frozen during training. We train the model with a batch size of 32 for SFT and 128 for RL, running 20 epochs of SFT followed by 1 additional epoch of GRPO-based reinforcement learning. The total reward in GRPO is a weighted sum of a format reward and a verifier reward, with weights 0.3 and 1.0, respectively. We set the KL penalty coefficient β to 0.04. During GRPO training, we use a separate, frozen Qwen2.5-VL-7B [2] as the VLM verifier for computing verifier reward r^{ver}, with a group size of 16. We use learning rates of 2×10^{-5} for SFT and 10^{-6} for GRPO.

EQA Baseline. In Table 1, for a comparison with an EQA pipeline, we employ the state-of-the-art Fine-EQA approach [15]. This framework includes both a frontier-based exploration strategy, which simply aims to visit unseen regions, and a goal-oriented exploration (GOE) strategy, which

guides the agent toward semantically informative areas given the current observation and language query. We adopt the GOE strategy so that the framework can actively adjust its viewpoint based on contextual cues, aligning it with our Visually-Grounded Active View Selection (VG-AVS) task. Following Fine-EQA [15], we use `prism-dinosiglip+7b` introduced in Prismatic VLMs [12] as the VLM backbone and replace the older GPT-4 models with the GPT-5 family [19] for region prioritization and exploration termination decisions.

G More Details on Dataset Curation

We provide additional details on the data curation procedure briefly introduced in Sections 3.1 and 4.1. We first describe the AVS dataset and AVS-ProcTHOR, both curated with ProcTHOR [7] 3D scenes, and then present AVS-HM3D, which is constructed from real indoor scenes in Habitat-Matterport3D [21].

G.1 AVS Dataset and AVS-ProcTHOR

Both of our synthetic datasets are constructed using the same fully automatic curation pipeline on ProcTHOR [7] 3D scenes, as introduced in Section 3.1. The AVS dataset is used for training, whereas AVS-ProcTHOR is reserved for evaluation. They share identical scene configurations; the only difference lies in the question types.

The AVS training dataset consists exclusively of binary object–existence questions of the form "*Is there a target object on the supporting object?*", where the correct answer is always "*yes*". In every sample, the target object mentioned in the question is guaranteed to exist on the specified supporting object, so that selecting a viewpoint that clearly reveals the object consistently yields a positive reward from the frozen VLM verifier, while uninformative viewpoints yield zero reward. This design aligns the verifier's feedback directly with the quality of view selection during RL training.

In contrast, AVS-ProcTHOR employs more diverse question types for a more comprehensive evaluation: `Existence`, `Counting`, and `State`. Existence questions ask which object is on the supporting object, counting questions ask how many instances of the target object are on that supporting object, and state questions query the state of the target object. For the `Existence` category, we additionally cast questions into a multiple-choice format to make random guessing less likely to succeed.

Building upon the notations introduced in Section 3.1, the overall curation pipeline consists of three stages:

Stage 1: Scene Modification by Question Type. Given a data sample, we first select a pair of a supporting object and a target object placed on it using ProcTHOR scene metadata, and modify the scene if needed to make it suitable for the specified question type.

The scene modification rules for each question type are as follows:

- `Existence.` No scene modification is performed.
- `Counting.` Place between two and five instances of the target object on or near the supporting object.
- `State.` Choose a target object with a controllable state (e.g., Faucet with on/off, Book with open/closed, Mug with filled/empty) and randomly assign one of its possible states.

Stage 2: Viewpoint Sampling. To obtain a target camera pose s_{tgt}, we leverage the built-in function in ProcTHOR, which returns a set of agent poses (positions and orientations) from which a specified object is both visible and interactable. Specifically, we call the `GetInteractablePoses()` built-in function on the supporting object to retrieve a list of nearby camera poses that can effectively observe it. We then sort these candidate poses by their distance to the supporting object, select the top 10 closest poses. We then randomly sample one pose from this subset and use it as the target camera pose s_{tgt}, provided that it satisfies the rule described in Equation 8.

Once s_{tgt} is determined, we generate candidate query poses s_{qry} by applying actions that move the agent slightly away from the target view. Specifically, we randomly sample 10 actions, parameterized as in Section 3.2, each consisting of a heading rotation in $[-90°, 90°]$, a forward translation in

$[50, 200]$ (centimeter), and a view rotation in $[-90°, 90°]$, where the forward translation is applied in the opposite direction so that the agent moves backward from the target viewpoint. Among the resulting candidates, we randomly select one query pose that satisfies the rule described in Equation 9.

Concretely, the sampling sets are defined as follows. Let $N_p(o, x)$ denote the pixel count of object x in view o, and $c(o, x)$ the normalized distance between the projected centroid of x and the image center:

$$o_{\text{tgt}} \sim \left\{ o \;\middle|\; N_p(o, x_{\text{tgt}}) > \tau_{\text{tgt}},\ c(o, x_{\text{tgt}}) < \delta_{\text{center}} \right\} \tag{8}$$

$$o_{\text{qry}} \sim \left\{ o \;\middle|\; \frac{N_p(o, x_{\text{tgt}})}{N_p(o_{\text{tgt}}, x_{\text{tgt}})} \leq \epsilon_{\text{tgt}},\ \frac{N_p(o, x_{\text{sup}})}{N_p(o_{\text{tgt}}, x_{\text{sup}})} > \epsilon_{\text{sup}} \right\}. \tag{9}$$

We set $\epsilon_{\text{sup}} = 0.05$, $\epsilon_{\text{tgt}} = 0.3$ for `Partial` and $\epsilon_{\text{tgt}} = 0$ for `Invisible`, $\tau_{\text{tgt}} = 30$, and a $90°$ field of view at 512×512 resolution. We use the default camera height of 90 centimeters and the default camera elevation angle of $15°$ downward in ProcTHOR [7].

Stage 3: Question–Answer Generation and Filtering. Given the selected target and supporting objects and their spatial relation, we instantiate rule-based question templates for each question type. For each generated sample, we retain only those for which the VLM verifier correctly answers the question given the target view o_{tgt}.

Stage 4: Chain-of-Thought Reasoning Trace Annotation. (Optional, for SFT training dataset) To enable efficient transfer from SFT to RL, we use CoT reasoning traces as supervision during SFT. To construct these traces, we prompt GPT-5-mini [19] with each step's image observation and its corresponding ground truth action, asking it to generate a brief reasoning trace that justifies the given action based on the current visual context and the navigation goal — specifically, what it observes, what it is trying to find, and why the ground truth action is appropriate. The full prompt is shown in Figure 10. The resulting traces are appended to the training samples as thinking tokens, providing the SFT model with explicit reasoning supervision prior to RL fine-tuning.

G.2 AVS-HM3D

As discussed in Section 4.1, we construct an additional benchmark, AVS-HM3D, on real indoor environments from Habitat-Matterport3D [21] to assess the generalization of our method beyond the synthetic ProcTHOR scenes used for training. We reuse triplets of (question, answer, ground-truth view) from the validation split of the Fine-EQA dataset [15], where the ground-truth view is the human-annotated frame from which the question–answer pair is derived. Among its seven question types, we discard those that rely on external world knowledge or holistic layout priors and retain five locally grounded types: `Attribute`, `Counting`, `Existence`, `Object`, and `State`. Given the ground-truth view, we then sample five candidate query views by applying small actions that move the agent away from the ground-truth view, following the same procedure as in the ProcTHOR setup. Due to the absence of scene metadata (e.g., object visibility and object relations) in real scenes, we manually select for each triplet, the query view that offers partial visual evidence—sufficient to guide the agent toward the ground-truth view, yet insufficient to directly answer the question. Using a GPT-5 model [19], we filter out low-quality items, such as cases where the question is not solvable from the ground-truth view or the ground-truth view has poor visual quality. The resulting curated benchmark comprises 191 samples. (`Attribute`: 21, `Counting`: 26, `Existence`: 42, `Object`: 36, `State`: 66)

H Computing Resources

All training is conducted on a cluster of eight NVIDIA H200 GPUs (141GB VRAM each). Training our models requires approximately 320 GPU hours in total, and inference across all baselines takes approximately 21 GPU hours.

```
Action Prompt:

You are an embodied agent navigating a 3D scene from an egocentric camera.
Given the current view image, a question about the scene, and your navigation
history, decide whether to move to a better viewpoint or stop if you have
enough information.
Action parameters (return integers only):
1) Heading rotation (deg) in [-90, 90]:  Azimuth yaw about your vertical axis
BEFORE moving.
Positive = clockwise/right, negative = counterclockwise/left, 0 = no rotation.
2) Forward distance (cm) >= 0:  Move forward in the NEW facing direction after
the rotation.  0 = no move.
3) View rotation (deg) in [-90, 90]:  Final azimuth adjustment AFTER moving,
relative to your post-move heading.
Same sign convention as rotation.
At each step you must choose one:
- MOVE: Output action parameters to navigate to a better viewpoint.
- STOP: If the current view already provides enough information to answer the
question.
Question:  "{VQA_question}" DO NOT answer the question; ONLY output MOVE/STOP,
and if moving, predict the action parameters.

Format Prompt:

First, output the reasoning process in <think> </think> tags.
Then, output your decision directly using one of the formats below.
If you decide to MOVE, output exactly these three tags in order:
<think> your reasoning </think>
<head> X </head> <fwd> Y </fwd> <view> Z </view>
If you decide to STOP (current view is sufficient), output:
<think> your reasoning </think>
<stop>
The text between <head> and </head> must be the angle in degrees [-90, 90],
<fwd> and </fwd> must be the nonnegative forward distance, and <view> and
</view> must be the final viewing angle in degrees [-90, 90].
For STOP decisions, output only the single tag <stop> without a closing tag.
Each must be exactly one integer number (no units, no extra text).
In the reasoning process, explicitly reason about whether you have enough
information to answer the question from the current view, or if you need to
move to a better viewpoint.
```

Figure 9: **Input system prompts.** The action prompt provides the task context and action parameterization explanation, while the format prompt specifies the required reasoning trace and output tag structure.

I Prompts

We present our input prompts in Figures 9 and 10. Figure 9 includes the action prompt that enforces the VLM to predict action parameters and the format prompt that specifies the desired output structure. Figure 10 shows the CoT annotation prompt used in Stage 4 of data construction discussed in Section G. In the prompts, `VQA_question` is a placeholder that is replaced by the actual question about the scene.

J Staged Rollout Visualizations

We visualize representative multi-step rollouts in Figure 11 for challenging cases involving strong occlusions or distant targets. Even when the region of interest is occluded by a wall or the target is too far away, our model gradually refines its viewpoint through iterative actions. For instance, when the initial view provides only a partial glimpse from outside a room, the agent first approaches and

```
CoT Annotation Prompt:

You are an annotation assistant for an embodied navigation dataset.
Your objective is to determine the action required to answer the question:
"{VQA_question}".  You will receive an image observation for the current step.
I will give you the observation and the Ground Truth action you must take.
Action space includes:
- <head>:  Azimuth rotation (yaw) in degrees before moving.  Positive is right,
negative is left.
- <fwd>:  Forward movement distance (cm).
- <view>:  Azimuth rotation (yaw) in degrees after moving.  Positive is right,
negative is left.
- <stop>:  Stop and end the episode if the answer is completely visible.
Note:  There is NO tilt or pitch action.  Both head and view are strictly
left/right azimuth (yaw) rotations.
You need to provide the 'thinking' process (about 3 short sentences) that
explains
why you are taking this Ground Truth action based on what you see, what you are
trying to find, and your goal.
Respond ONLY with the thinking content.
```

Figure 10: **CoT annotation prompt.** The prompt provides the task context, action space parameterization, and instructs the model to generate a brief reasoning trace explaining the ground truth action based on the current observation.

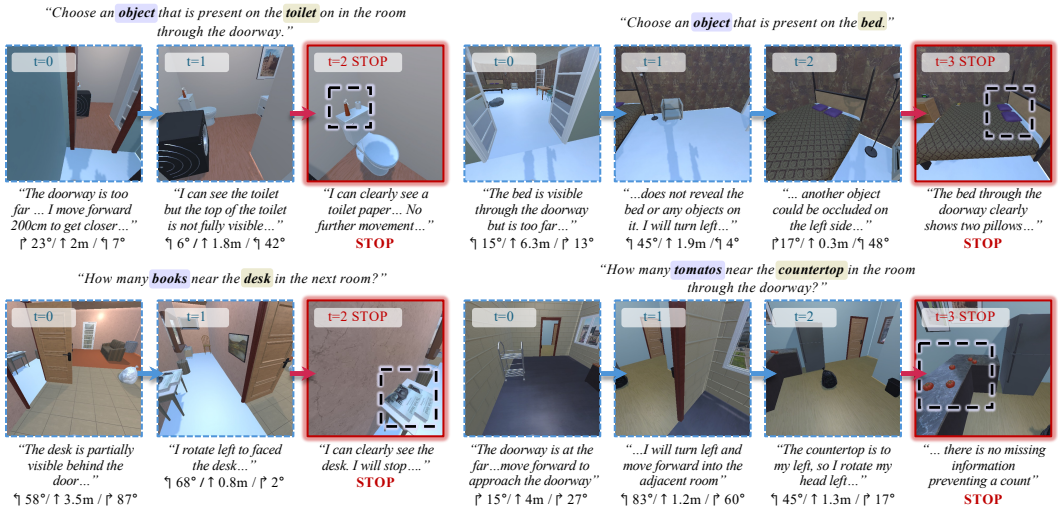


Figure 11: **Examples of multi-turn action rollout.** For samples with complex spatial layouts or distant targets, our model iteratively adjusts its viewpoint to obtain better observation for answering the question.

enters through the doorway, then iteratively adjusts its viewpoint to fully observe the target object (bottom-right example).

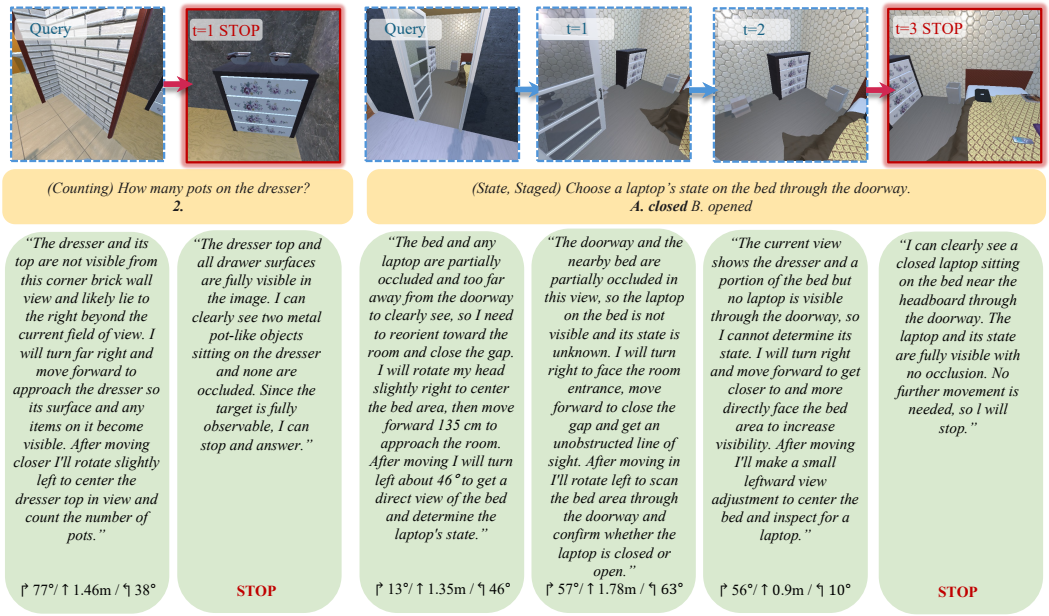
K More Qualitative Comparisons

We provide more qualitative comparison results of Visually-Grounded Active View Selection in Figure 12.

Figure 12: **More qualitative results on AVS-ProcTHOR (top six rows) and AVS-HM3D (bottom three rows).** Blue and gray mark the object of interest and surrounding cue objects, respectively. ✓ denotes correct answers (LLM-Match = 5), ✗ incorrect ones (LLM-Match $\leq$ 2).

L Qualitative Examples of AVS Framework Reasoning

In the following, we present qualitative examples illustrating how our AVS framework reasons from the query view to predict the desired action parameters.



M Broader Impacts

This work aims to improve the reliability of embodied question answering systems by enabling agents to refine their viewpoints and gather better visual evidence before answering. Potential positive impacts include more grounded assistive robots and indoor embodied agents that can answer questions about their surroundings with fewer hallucinations.

However, active visual perception in indoor environments also raises privacy and safety concerns. A system that intentionally changes its viewpoint could be misused for intrusive monitoring in homes, offices, or care facilities. Even when used as intended, incorrect viewpoint selection or overconfident answers may mislead users about the physical environment.

Our work is not a deployable robotic system, and it focuses only on near-target view refinement. Practical deployment would require safeguards such as explicit user consent, clear indication when visual sensing is active, privacy-preserving data handling, and human oversight in high-stakes settings. Since our training data is synthetic and our real-scene evaluation is curated, further work is needed to evaluate robustness, bias, and uncertainty before real-world deployment.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Both abstract and introduction reflect contributions in the main paper.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: Limitations discussed in the conclusion.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

Justification: No theoretical analysis in the paper.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: We provide the implementation details and hyperparameters for reproducibility.

Guidelines:

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

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

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

Answer: [No]

Justification: We will release the code upon acceptance.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide experiment setups and implementation details.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: Due to limited compute, we do not report error bars, but provide a deterministic evaluation metric result "target visibility" in the appendix for reference.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Compute resources described in the appendix.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The work fully conforms to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss potential benefits for assistive embodied agents, as well as privacy, surveillance, safety, and robustness concerns associated with active visual perception in indoor environments.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [N/A]

    Justification: No high-risk data or models released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All used assets are properly cited.

    Guidelines:

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

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [No]

    Justification: We will release the dataset upon acceptance, but we describe the details of the data construction pipeline in the appendix for reproducibility.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects involved.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: No human subjects research conducted.

    Guidelines:

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

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: LLMs/VLMs were used for CoT annotation, sample filtering, VLM verification, and policy backbone.

    Guidelines:

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