E-ViC: Reasoning Beyond Text via Embodied Visual Chain for Spatial Intelligence

Junbo Qi[1*], Yi Zhang[2*], Hanchu Ni[3], Che Liu[4], Zhimin Yao[5], Ruilin Yang[5], Xiancong Ren[2], Liangjian Wen[6], Wei Ge[5], Yuya Ieiri[1], Osamu Yoshie[1†], Yong Dai[2], Xiaozhu Ju[2†]

[1]Waseda University, [3]Peking University, [4]Imperial College London,
[5]Megvii, [6]Southwestern University of Finance and Economics,
[2]X-Humanoid

junboqi@toki.waseda.jp, enjoyzhangyi@gmail.com
yoshie@waseda.jp, juxiaozhu@outlook.com

Abstract

Precise spatial reasoning is fundamental to embodied intelligence, yet current Vision-Language Models (VLMs) remain bottlenecked by text-based Chain-of-Thought (CoT) that relies solely on textual reasoning trajectories, often bypassing active engagement with fine-grained visual details. To address this, we present **E-ViC** (**E**mbodied **Vi**sual **C**hain), a framework that moves reasoning beyond text and directly into the visual domain. By formulating visual operations (e.g., zooming, marking) as executable primitives, E-ViC transforms perception from static prediction into an active verification process. Distinct from approaches relying on supervised step-wise trajectories, E-ViC is trained via an agentic reinforcement learning paradigm. This enables the model to autonomously discover optimal policies, leading to the emergence of human-like "look-and-confirm" strategies driven solely by task-level rewards. To facilitate this, we curate a comprehensive 24.4K-sample dataset covering diverse embodied tasks. By grounding reasoning in pixel-level interactions, E-ViC reframes spatial intelligence as a verifiable, tool-using capability. Extensive evaluations on external benchmarks demonstrate that our approach consistently outperforms strong VLM baselines with an average gain of 10.1%.

1 Introduction

Vision-Language Models (VLMs) have achieved remarkable progress in general visual question answering (VQA) and 2D scene understanding (Zhang et al., 2024, 2025b). However, deploying these models as an "Embodied Brain" necessitates a fundamental paradigm shift: transitioning from *a passive observer* that describes the world to an active agent that interacts with it. While a passive observer needs only to align visual features with semantic concepts (e.g., identifying a "mug"), an embodied model faces the strictly more demanding challenge of *physical actionability*. This requirement exposes a critical *semantic-spatial gap* in current VLMs (Zhang et al., 2025c). For a robot, vague spatial awareness is insufficient; *valid execution demands geometric precision—the rigorous quantification of physical space*. Consequently, bridging this gap entails two fundamental capabilities: relative spatial reasoning and absolute localization.

To endow VLMs with genuine spatial perception, existing research has bifurcated into two primary paradigms. The first decomposes reasoning into multi-stage pipelines (Zhang et al., 2025d), producing intermediate artifacts like segmentation masks. While interpretable, this modularity is brittle, suffering from severe error propagation across stages. Conversely, recent end-to-end frameworks such as Embodied-R1 (Yuan et al., 2025b) and From Seeing to Doing (FSD) (Yuan et al., 2025a) attempt to internalize spatial reasoning via the textual Chain-of-Thought (CoT). However, we identify a fundamental modality mismatch in this approach: natural language is inherently designed for semantic compression, not geometric precision. *Forcing fine-grained spatial reasoning into a pure textual stream creates a severe information bottleneck*—abstracting away the pixel-level cues essential for manipulation. Consequently, relying solely on linguistic reasoning acts as a "blindfold", detaching the model's decision-making from the physical ground truth it must operate upon.

To dismantle this linguistic blindfold, we present **E-ViC** (**E**mbodied **Vi**sual **C**hain). Our core insight is that spatial reasoning must remain visually native to *preserve geometric fidelity*. Instead of compressing visual observations into lossy textual descriptions, E-ViC transforms perception from a static prediction task into an active verification process (see Figure 1). By formulating visual oper-

*Equal contribution.
†Corresponding authors.

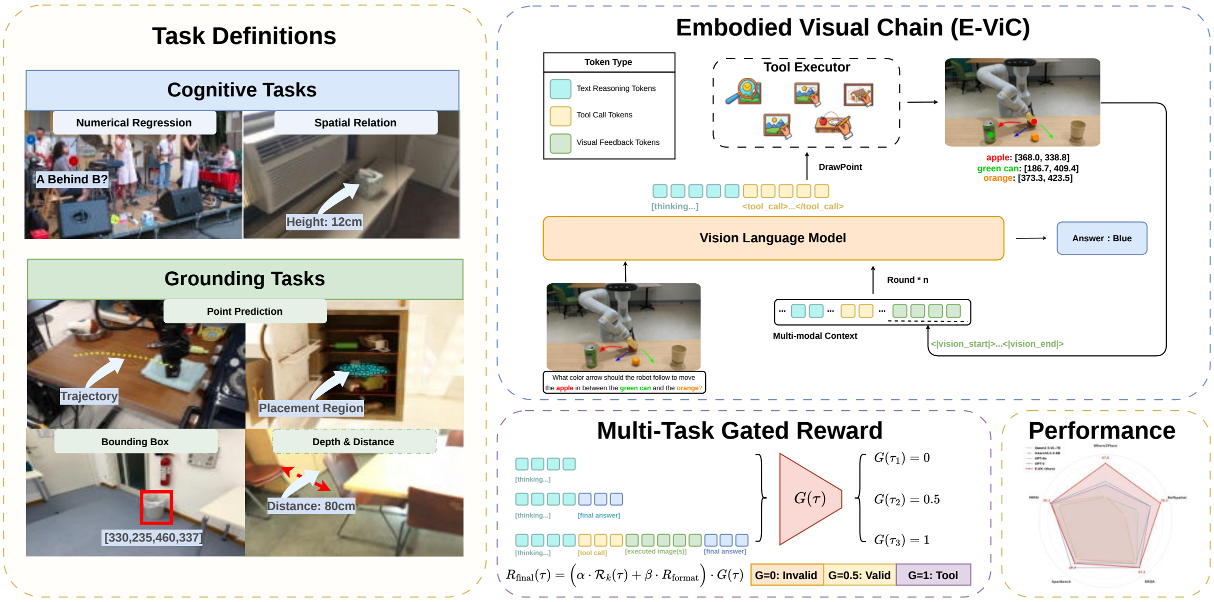

Figure 1: **Overview of the Embodied Visual Chain (E-ViC).** The framework (left) categorizes embodied challenges into cognitive and grounding tasks. The agentic VLM policy (top-right) utilizes multimodal reasoning tokens and a Tool Executor for interactive task execution. To optimize performance, we introduce a **Multi-Task Gated Reward** (bottom-center). Results (bottom-right) show that E-ViC achieves SOTA performance and outperforms baseline model by an average of 10.1% in spatial perception and localization.

ations—such as dynamic zooming and point marking—as executable decision primitives, the model engages in an iterative *"look-and-confirm" loop*. This mechanism mirrors human cognitive strategies: *resolving spatial ambiguity not by guessing, but by actively scrutinizing critical regions to verify hypotheses.* Consequently, E-ViC circumvents the abstraction bottleneck of language, grounding reasoning directly in fine-grained visual feedback to achieve sub-pixel precision.

Concretely, we instantiate E-ViC through an agentic reinforcement learning paradigm akin to the R1-Zero setting. Diverging from conventional approaches that rely on Supervised Fine-Tuning (SFT) to mimic human trajectories, our method is unsupervised in the behavioral space. We do not explicitly teach the model how to observe; rather, we simply incentivize task success. Under this outcome-driven objective, complex spatial reasoning strategies spontaneously emerge: *the model autonomously discovers that proactively invoking visual tools to verify its predictions* is the optimal policy for precise grounding. This *"think-and-refine"* process not only eliminates the dependency on labor-intensive trajectory annotation but also empowers the model to *evolve robust perception strategies that transcend the limits of static imitation*. Our experimental results demonstrate substantial improvements in spatial cognition and grounding tasks. Our contributions are threefold:

- We propose E-ViC, a framework that shifts reasoning from pure textual trajectories to visual chain by formulating tool use as decision primitives. Operating under an agentic reinforcement learning paradigm with a novel multi-task gated reward, the model autonomously discovers effective strategies without supervised reasoning data, preventing degenerate behaviors.

- We curate a training mixture of 24.4K samples, categorized into grounding and cognitive tasks, to facilitate tractable reinforcement learning. Unlike standard aggregations, we apply verifiability-based filtering to ensure high-quality supervision signals, spanning challenges from fine-grained localization to semantic reasoning.

- Validating our approach across a suite of embodied benchmarks, we report a 10.1% average gain over strong baselines, confirming the superiority of pixel-level verification over linguistic abstraction. Through a comprehensive analysis of these results, we offer critical insights to the community that shed light on the future direction of verifiable visual reasoning.

2 Related Work

2.1 Tool-Augmented VLMs

Recent advances in Vision-Language Models (VLMs) have moved beyond single-pass image-to-text prediction toward interactive perception with explicit tool usage. Existing work in this paradigm broadly falls into two categories.

One line of research focuses on programmatic reasoning and sketch-based interfaces, where external tools are used to explicitly structure spatial reasoning. Representative approaches include Visual Program Distillation (VPD) (Hu et al., 2024b), which distills programmatic reasoning and tool usage into VLMs, and Visual Sketchpad (Hu et al., 2024a), where models perform drawing operations on a virtual canvas to support visual reasoning.

Another line emphasizes active perception, encouraging models to resolve perceptual ambiguities through dynamic visual operations such as zooming or cropping. This includes high-level reasoning systems such as OpenAI o1/o3 (Jaech et al., 2024) and GRIT (Fan et al., 2025), as well as RL-based frameworks like DeepEyes (Zheng et al., 2025), Pixel Reasoner (Wang et al., 2025a), and Reinforcing Spatial Reasoning (Wu et al., 2025). While these methods improve spatial understanding, they typically rely on supervised reasoning trajectories, handcrafted programs, or carefully designed cold-start initializations to bootstrap learning.

2.2 Embodied Reasoning and Perception

The integration of complex reasoning into embodied tasks has emerged as a pivotal frontier, requiring models to bridge the gap between high-level semantic understanding and low-level motor control. Early efforts, such as CoT-VLA (Zhao et al., 2025) and ThinkAct (Huang et al., 2025), establish the foundation for this by incorporating visual chain-of-thought (CoT) structures to guide robotic actions. Similarly, Embodied VSR (Zhang et al., 2025d) focuses on enhancing the spatial reasoning capabilities of agents within 3D environments.

Recent analyses further examine how multimodal models internally perceive, store, and recall spatial layouts across views and time, revealing persistent limitations in spatial memory and mental map construction (Yang et al., 2025). To further optimize these reasoning chains, recent works have turned to reinforcement learning (RL) and fine-tuning strategies. For instance, Reason-RFT (Tan et al., 2025) introduces reinforcement fine-tuning to bolster the visual reasoning of VLMs, while Embodied-R1 (Yuan et al., 2025b) leverages reinforced embodied reasoning specifically for general robotic manipulation tasks.

E-ViC follows this line of research by treating visual tool use as a native component of the reasoning process. Under an R1-Zero-style agentic RL paradigm, the model acquires multi-step perception and deliberation behaviors directly from task-level rewards, enabling it to iteratively inspect visual evidence, focus on informative regions, and commit to final predictions in a human-like reasoning loop.

3 Method

We present **E-ViC**, a framework designed to enable the use of visual tools for solving embodied tasks. E-ViC projects the reasoning process into the *situated pixel space*, enabling the system to manifest spatial hypotheses through iterative *visual adjustments*, such as adaptive zooming, directly upon the visual input. This closed-loop process allows the system to refine its spatial understanding through continuous physical interaction with the visual context.

We optimize E-ViC by applying an agentic reinforcement learning paradigm with sparse task-level rewards. Section 3.1 introduces the task formulation, modeling embodied spatial reasoning as a multi-step decision process with visual tool use. Section 3.2 presents the agentic reinforcement learning framework used to train the model directly from task-level rewards, without supervised reasoning trajectories. Finally, Section 3.3 outlines the training infrastructure and system design enabling scalable, high-throughput closed-loop reinforcement learning.

3.1 Task Formulation

Given image I and instruction l, our goal is to produce a spatial output y (e.g., coordinates, bounding boxes, or multiple-choice answers). We formulate this problem as a multi-turn reasoning process, in which the model incrementally gathers visual evidence and refines its prediction. Formally, the model generates a reasoning trajectory $\tau = (o_1, o_2, \ldots, o_T)$, where each step o_t belongs to one of the following categories:

- **Text reasoning**: a textual reasoning step, where the model reasons about the current visual input and task instruction.

- **Visual tool call**: a function calling step that applies a predefined visual operation to the image in order to acquire additional visual evidence. Supported operations are listed in Appendix E.
- **Final answer**: the terminal step that outputs the spatial prediction y.

Turn Budget. We define a *turn* as one complete perception-action-feedback cycle. Turn $t = 0$ consists of the initial prompt (I_0, l) and model response o_1. For $t \geq 1$, turn t comprises the tool's visual feedback I_t and the model's subsequent response o_{t+1}. The maximum turn budget $T_{\max}$ thus bounds the total number of perception-reasoning cycles permitted per episode.

3.2 E-ViC Overall Pipeline

Overall Pipeline As shown in Algorithm 1 and Figure 1, our method follows a multi-turn visual reasoning pipeline for embodied tasks. Given an initial image and a language instruction, the model first constructs a unified context representation. At each turn, the policy generates an output conditioned on the current context, which can be one of three types: text reasoning, visual tool call, or final answer. Text reasoning updates the intermediate context state, while visual tool calls execute corresponding operations on the image and return visual feedback; both the action and the feedback are appended to the context to support subsequent reasoning. When the model produces a final answer, the reasoning process terminates and outputs the spatial prediction. By alternating between textual reasoning and visual interaction over multiple turns, the model progressively incorporates visual feedback to revise intermediate judgments, resulting in more stable and accurate spatial reasoning.

A key challenge in realizing such a closed-loop visual reasoning process lies in how to train the model to decide *when* and *how* to perform reasoning or invoke visual tools, without relying on annotated multi-step trajectories. To address this challenge, we formulate the training of E-ViC under an **R1-Zero agentic reinforcement learning** paradigm. This training strategy directly optimizes multi-turn reasoning behaviors from task-level rewards, mirroring the human problem-solving process of decomposing complex scenes, focusing attention on critical regions, and verifying hypotheses through active visual inspection.

Algorithm 1 E-ViC Overall Pipeline

```
Require: image(s) I, instruction l, policy π_θ, max turns T
Ensure: Spatial prediction y
C ← (I, l)
for t = 1 to T do
    o_t ~ π_θ(· | C)
    if o_t is final answer then
        return PARSE(o_t)
    else if o_t is visual tool call then
        I_t ← EXECUTE(o_t) {See Appendix E}
        C ← C ⊕ (o_t, I_t)
    else
        C ← C ⊕ o_t
    end if
end for
return PARSE(o_T)
```

To further stabilize learning across heterogeneous embodied tasks and prevent degenerate reasoning behaviors, we introduce a **Gated Reward Formulation**. This mechanism selectively propagates reward signals only when the reasoning trajectory satisfies structural validity and perceptual grounding constraints, ensuring that visual tools are activated strategically rather than arbitrarily.

Multi-Task Gated Reward To optimize performance across heterogeneous embodied tasks while preventing degenerate reasoning behaviors, we introduce a *Multi-Task Gated Reward* mechanism that selectively propagates reward signals based on trajectory quality. Our formulation decomposes the total reward R_{final} into task-specific accuracy and structural validity components:

$$R_{\text{final}}(\tau) = (\alpha \cdot \mathcal{R}_k(\tau) + \beta \cdot R_{\text{format}}) \cdot G(\tau), \quad (1)$$

where $\mathcal{R}_k(\cdot)$ is a task-specific reward operator tailored to task type k (detailed in Appendix D), R_{format} enforces syntactic validity, and $G(\tau)$ serves as a gating coefficient that filters low-quality reasoning trajectories. Hyperparameters α and β balance task performance against format adherence.

Gating Mechanism. The coefficient $G(\tau)$ implements a hierarchical filtering process that evaluates reasoning trajectories across two critical dimensions:

$$G(\tau) = \mathbb{1}(\mathcal{C}_1) \cdot (0.5 + 0.5 \cdot \mathbb{1}(\mathcal{C}_2)), \quad (2)$$

where $\mathbb{1}(\cdot)$ denotes the indicator function, and $\{\mathcal{C}_i\}_{i=1}^2$ represent hierarchical constraints:

- $\mathcal{C}_1$ **(Behavioral Anti-Degeneracy):** Prevents trivial solutions by penalizing repetitive or stationary behaviors. We enforce state progression by verifying that consecutive actions induce meaningful context updates: $\phi(o_i) \neq \phi(o_{i+1})$, where $\phi(\cdot)$ encodes the semantic state. The tool calling is in JSON format and is compared by exact match. This constraint eliminates reward for trajectories exhibiting action loops or redundant tool invocations—behaviors that exploit reward structure without genuine spatial reasoning.

- $\mathcal{C}_2$ **(Active Perceptual Grounding):** Incentivizes active visual engagement by rewarding trajectories that invoke visual tools (e.g., `zoom_in`, `draw_point`) to anchor reasoning in pixel-level observations. Formally, $\mathcal{C}_2$ denotes $|\tau \cap \mathcal{T}_{\text{visual}}| > 0$, where $\mathcal{T}_{\text{visual}}$ denotes the set of visual operations.

Hierarchical Credit Assignment. Under this formulation, $\mathcal{C}_1$ acts as a strict validity filter: violation results in zero reward ($G = 0$), eliminating gradient signal from degenerate trajectories. In contrast, $\mathcal{C}_2$ introduces *differential scaling* for embodied reasoning. The agent receives partial credit ($G = 0.5$) for text-only reasoning that produces correct answers without visual grounding—acknowledging linguistic capability while penalizing "blind" prediction. Full credit ($G = 1.0$) is reserved for trajectories that actively engage with the visual environment, thereby coupling task success with perceptual verification. This design biases the policy toward human-like spatial problem-solving: decomposing scenes through visual inspection rather than abstracting away spatial structure into language.

Optimization via Trajectory GRPO. We instantiate Agentic RL using Group Relative Policy Optimization (Shao et al., 2024), which computes advantages from group statistics without requiring a learned value function. For each input (I, l), we sample G reasoning trajectories $\{\tau_i\}_{i=1}^{G}$. The advantage for trajectory τ_i is computed as:

$$\hat{A}_i = \frac{R_{\text{final}}(\tau_i) - \text{mean}(\{R_{\text{final}}(\tau_j)\}_{j=1}^{G})}{\text{std}(\{R_{\text{final}}(\tau_j)\}_{j=1}^{G})} \tag{3}$$

Training aims to maximize the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{(I,l)\sim\mathcal{D},\, \{\tau_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}} \Bigg[\frac{1}{G} \sum_{i=1}^{G} \sum_{t=1}^{|\tau_i|} \min\left(\rho_{i,t}(\theta)\hat{A}_i,\ \text{clip}\left(\rho_{i,t}(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_i\right) \Bigg], \tag{4}$$

where the policy ratio $\rho_{i,t}(\theta)$ for step t of trajectory τ_i is defined as:

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t} \mid h_{i,t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid h_{i,t})}. \tag{5}$$

3.3 Training Infrastructure

Training agentic policies introduces challenges from variable episode lengths and blocking tool execution. Episodes exhibit high length variability during training—ranging from direct answers (1 turn) to iterative refinement (up to $T_{\max} = 5$ turns)—creating load imbalance across workers and complicating batch construction.

We employ an asynchronous actor-learner architecture to maintain training efficiency. Distributed workers generate trajectories using local policy copies and deposit completed episodes into a shared buffer. The central learner samples batches with priority weighting based on reward variance, ensuring diverse gradient signals while handling length heterogeneity. Policy parameters are synchronized to workers periodically.

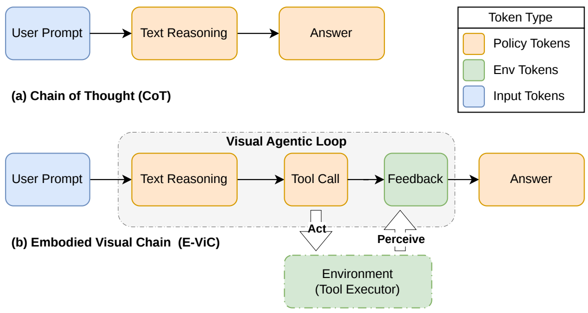

Figure 2: **Token-level factorization for credit assignment.** Policy tokens (orange) include text reasoning, tool calls, and answers. Environment tokens (green) represent deterministic tool responses. Importance sampling applies only to policy tokens.

Deterministic tool execution. A key design principle differentiates E-ViC from standard CoT RL: tool responses constitute deterministic environment transitions rather than stochastic policy samples. This enables a factored objective (Figure 2) where importance sampling applies exclusively to policy tokens:

$$\nabla_\theta \mathcal{J}(\theta) = \mathbb{E}_\tau \left[\sum_t \nabla_\theta \log \pi_\theta(o_t \mid h_t) \cdot A(\tau) \right], \tag{6}$$

where h_t includes all prior policy outputs $\{o_1, \ldots, o_{t-1}\}$ and deterministic tool responses.

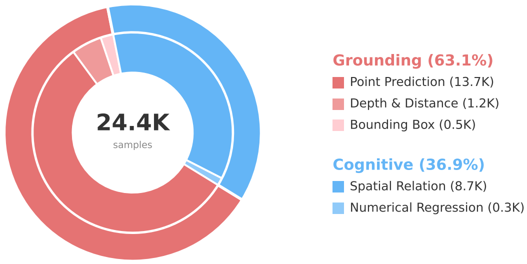

Figure 3: **Training data distribution.**

This factorization provides clearer credit attribution for learning when and how to invoke visual tools.

4 Experiments

We evaluate E-ViC on a comprehensive suite of spatial understanding benchmarks, demonstrating significant improvements over both open-source and proprietary VLMs. Our experiments address three key questions: (1) Does E-ViC improve spatial reasoning over standard VLMs and text-only CoT? (2) Is the gated reward mechanism necessary for multi-turn reasoning to emerge? (3) How does tool diversity affect learning dynamics and final performance?

4.1 Data Collection

To endow our model with robust spatial reasoning and precise point-based grounding capabilities, we curate a diverse training mixture from multiple complementary data sources. We adopt a verifiability-based filtering approach: for each candidate sample, we independently query both Gemini-2.5-Flash and Qwen2.5-VL with 8 rollouts each, retaining only samples where at least one rollout produces a correct answer. After filtering and stratified subsampling to balance task diversity, the final training mixture comprises approximately 24.4K examples spanning two complementary task categories, as illustrated in Figure 3.

Cognitive Tasks. Cognitive tasks encompass choice-based reasoning problems requiring discrete answers, totaling 9.0K examples across two categories. Spatial Relation (8.7K) is aggregated from RefSpatial (Zhou et al., 2025), SPAR-7M (Zhang et al., 2025a), and ViLaSR (Wu et al., 2025), spanning diverse challenges including spatial relation recognition, maze navigation, video-based reasoning, multi-view understanding, and viewpoint transformation. Numerical Regression (0.3K) from ViLaSR (Wu et al., 2025) requires precise numerical outputs for spatial quantities such as counting and measurement.

Grounding Tasks. Grounding tasks require the model to produce precise spatial outputs such as coordinates and bounding boxes, which can be directly verified against ground-truth annotations, totaling 15.4K examples. As shown in Figure 3, these tasks primarily cover point prediction (13.7K), depth and distance estimation (1.2K), and bounding box localization (0.5K), drawn from multiple established embodied and spatial reasoning benchmarks, including RefSpatial, RoboPoint (Yuan et al., 2024), ShareRobot (Ji et al., 2025), and SPAR-7M.

4.2 Implementation Details

Architecture and Infrastructure. We build our codebase on Volcano Engine Reinforcement Learning (veRL) (Sheng et al., 2024). We instantiate E-ViC using Qwen2.5-VL-7B-Instruct (Bai et al., 2025) as the base model, optimized via GRPO (Shao et al., 2024) on 64 NVIDIA A800 GPUs with FSDP (Zhao et al., 2023). Training employs mixed-precision (bfloat16), gradient checkpointing, and flash attention (Dao et al., 2022) for computational efficiency.

Optimization Protocol. We use learning rate 1×10^{-6} , batch size 128, sampling $n = 16$ trajectories per prompt at $\tau = 1.0$. Clipping is $\epsilon = 0.2$ without KL penalty. Maximum $T_{\max} = 5$ turns.

Controlled Comparisons. All trained methods—including ablation variants detailed in Section 4.4—share identical GRPO hyperparameters and training data (24.4K samples). External models (GPT-4o (OpenAI et al., 2024), GPT-5 (OpenAI, 2025), InternVL3.5 (Wang et al., 2025b), Qwen2.5-VL variants) are evaluated zero-shot without tool access. Training stabilizes after 200 optimization steps; we report results from the checkpoint maximizing held-out validation reward.

4.3 Main Results

Table 1 presents comprehensive comparisons on five spatial understanding benchmarks. We evaluate against both open-source VLMs (Qwen2.5-VL, InternVL3.5) and proprietary models (GPT-4o, GPT-5).

Table 1: **Performance comparison on spatial understanding benchmarks.** We compare our E-ViC method against state-of-the-art open-source and proprietary vision-language models. Bold indicates the best performance and underline indicates the second best performance per column. When multiple models tie for the best, all are shown in bold without underline for second place.

Model	**Size**	**Where2Place**	**RefSpatial**	**ERQA**	**SparBench**	**MMSI**	**Avg.**
Open-Source Vision-Language Models							
Qwen2.5-VL-Instruct	7B	19.2	16.7	40.0	34.2	25.9	27.2
Cosmos-Reason1(NVIDIA et al., 2025)	7B	19.0	4.9	38.3	31.1	11.6	21.0
InternVL3.5	8B	<u>32.3</u>	<u>23.2</u>	39.3	37.2	**30.2**	32.4
Qwen2.5-VL-Instruct	32B	21.6	16.4	**43.5**	<u>38.6</u>	28.4	29.7
Proprietary API Models							
GPT-4o	–	20.3	9.9	37.8	34.3	29.9	26.4
GPT-5	–	29.6	23.1	**43.5**	**41.6**	29.4	<u>33.4</u>
Ours							
E-ViC	7B	**47.0**	**28.2**	43.3	37.7	**30.2**	**37.3**
Δ (*vs* Qwen2.5-VL-7B)	–	*+27.8*	*+11.5*	*+3.3*	*+3.5*	*+4.3*	*+10.1*

Comparison with Open-Source Models. E-ViC achieves substantial improvements over its base model Qwen2.5-VL-7B-Instruct across all benchmarks, with an average gain of **+10.1%**. The most pronounced improvements appear on tasks requiring fine-grained localization: **+27.8%** on Where2Place and **+11.5%** on RefSpatial. These results validate our core hypothesis that grounding reasoning directly in pixel-level interactions through visual tool use enables more precise spatial understanding than text-only deliberation. Notably, E-ViC outperforms InternVL3.5-8B (32.4% avg.) despite using comparable model capacity, and even surpasses Qwen2.5-VL-32B-Instruct (29.7% avg.)—a model with 4× more parameters. This demonstrates that E-ViC provides a more effective approach than simply scaling model size for spatial reasoning.

Comparison with Proprietary Models. E-ViC achieves the highest average performance (37.3%) among all evaluated models, surpassing GPT-5 (33.4%) by **+3.9 points**. The performance gap is particularly pronounced on Where2Place (+26.7% vs. GPT-4o, +17.4% vs. GPT-5) , a benchmark requiring precise placement predictions in cluttered scenes. This suggests that proprietary models, despite their general capabilities, lack the iterative visual refinement that E-ViC acquires through agentic RL. On ERQA, E-ViC achieves comparable performance to GPT-5 (43.3% vs. 43.5%), demonstrating competitive embodied reasoning capabilities.

4.4 Ablation Studies

We conduct systematic ablations to isolate the contribution of each design choice in E-ViC. All variants share identical training data, hyperparameters, and computational budget. We evaluate on three complementary benchmarks: Where2Place (fine-grained placement), ERQA Action (embodied action reasoning), and ERQA Point (point localization).

Experimental Setup. We compare E-ViC against three ablated configurations:

- **CoT RL**: Text-only Chain-of-Thought with identical GRPO optimization but *no visual tool access*. This isolates the contribution of visual reasoning from general RL-based improvements.
- **w/o Gate**: Removes gated mechanism (Eq. 2), retaining only the task accuracy term and format term. This tests whether explicit incentivization is necessary for tool-use emergence.
- **w/o Other Tools**: Restricts the tool suite to `draw_point` only. This examines whether tool diversity is necessary or if a single dominant tool suffices.

Finding 1: Visual Tool Use Provides Substantial Gains Over Text-Only Reasoning. Comparing E-ViC against CoT RL in Figure 4(b–d) reveals the fundamental advantage of grounding reasoning in pixel space. At convergence, E-ViC outper-

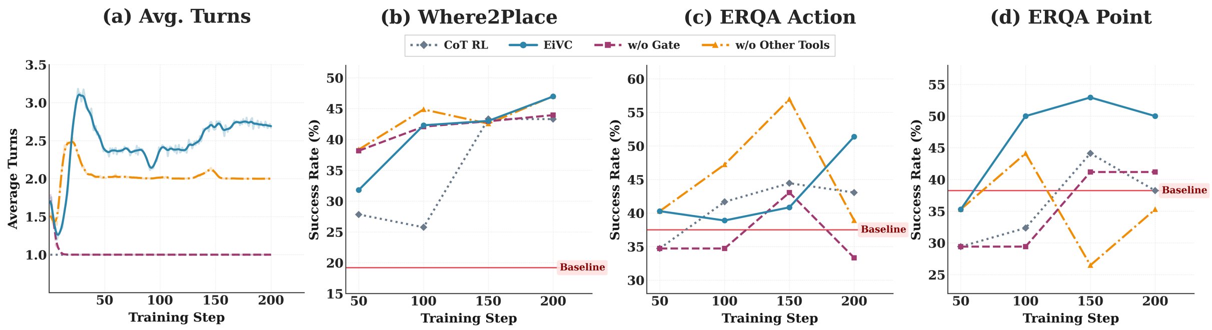

Figure 4: **Ablation study results.** (a) Average turns across training steps for different variants. (b-d) Performance comparison on Where2Place, ERQA Action, and ERQA Point benchmarks. Gated rewards are essential for multi-turn reasoning emergence, visual tools consistently outperform text-only CoT, and tool diversity prevents overfitting and stabilizes training. Dashed line indicates the baseline performance.

forms CoT RL by **+8.3%** on ERQA Action and **+11.8%** on ERQA Point , while achieving notable gains on Where2Place . These improvements are particularly striking given that both methods use identical GRPO optimization—the only difference is access to visual tools. The results demonstrate that *certain spatial information cannot be recovered through text-only reasoning*, regardless of how sophisticated the language-based deliberation becomes. Visual operations like zooming and point annotation provide direct pixel-level evidence that text descriptions can only approximate.

Finding 2: Gated Rewards Are Essential for Multi-Turn Emergence. Figure 4(a) reveals fundamentally different learning dynamics between E-ViC and *w/o Gate*. E-ViC exhibits three distinct phases: *exploration* (steps 1-30, peaking at 3.2 turns), where the model experiments with diverse tool invocation patterns; *consolidation* (steps 30–90), where ineffective strategies are pruned; and *stabilization* ($\sim$2.7 turns), representing learned task-adaptive reasoning depth.

In stark contrast, *w/o Gate* collapses to single-turn behavior within 20 training steps. Without gate, the model discovers a degenerate but locally optimal strategy: immediate commitment without visual deliberation. This manifests clearly in the benchmark results: while *w/o Gate* achieves reasonable Where2Place performance, it suffers severe degradation on ERQA Action—an **18.1% gap**. The accuracy-only reward creates a shallow local optimum corresponding to direct prediction; the gated mechanism reshapes this landscape by requiring tool engagement for full credit, enabling escape from the single-turn attractor.

Finding 3: Tool Diversity Prevents Overfitting and Stabilizes Learning. The *w/o Other Tools* variant reveals a subtle but critical failure mode best observed in Figure 4(c–d). On Where2Place, this configuration achieves competitive final performance (47.0%). However, the ERQA benchmarks expose severe instability.

On ERQA Action, *w/o Other Tools* exhibits classic overfitting: performance peaks at 56.9% (step 150)—temporarily exceeding all other methods—then crashes to 38.9% by step 200, a **18.0 point degradation**. On ERQA Point, accuracy oscillates wildly between 26.5% and 44.1% across checkpoints, with final performance falling **14.7 points below** E-ViC. In contrast, E-ViC shows monotonic improvement on Where2Place and maintains stable, high performance ($\geq$50%) on ERQA Point from step 100 onward.

We attribute this instability to *gradient homogeneity*: with only `draw_point` available, the model receives narrow feedback that leads to overfitting on specific reasoning patterns. The full tool suite—`zoom_in`, `draw_point`, `draw_box`, `draw_trajectory`, `draw_3d_bbox`—provides diverse supervision signals that regularize learning and prevent memorization of task-specific shortcuts.

5 Conclusion

In this paper, we presented E-ViC, a framework that shifts spatial reasoning from linguistic abstraction to active visual interaction. By formulating visual tools as decision primitives, E-ViC enables VLMs to iteratively refine perception through direct pixel-level feedback. Our agentic reinforcement learning paradigm facilitates the autonomous emergence of deliberation strategies from task-level rewards, bypassing the need for supervised expert trajectories. Training on our curated 24.4K-sample dataset leads to a 10.1% average performance gain over strong baselines. Ultimately, E-ViC reframes spatial intelligence as a verifiable, tool-mediated process for robust embodied systems.

6 Limitations

While E-ViC demonstrates consistent improvements across diverse spatial reasoning benchmarks, we acknowledge that its multi-turn visual reasoning paradigm incurs additional computational overhead compared to single-pass baselines. This reflects an inherent trade-off in deliberative reasoning frameworks that prioritize spatial precision over inference latency, and may limit applicability in real-time or resource-constrained settings.

Additionally, although our evaluation focuses on established benchmarks to ensure reproducibility and fair comparison, effectively deploying such reasoning capabilities in real-world environments requires addressing practical challenges such as sensor noise, partial observability, and environmental dynamics. Ensuring robustness under these conditions remains a non-trivial challenge shared across the embodied AI community.

Accordingly, we view our work as establishing foundational spatial reasoning capabilities that are intended to support and enhance real-world embodied systems, rather than as a complete end-to-end solution for direct deployment in physical environments.

References

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. *Preprint*, arXiv:2502.13923.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Preprint*, arXiv:2205.14135.

David Eigen, Christian Puhrsch, and Rob Fergus. 2014. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. 2010. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338.

Yue Fan, Xuehai He, Diji Yang, Kaizhi Zheng, Ching-Chen Kuo, Yuting Zheng, Sravana Jyothi Narayanaraju, Xinze Guan, and Xin Eric Wang. 2025. Grit: Teaching mllms to think with images. *arXiv preprint arXiv:2505.15879*.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, and Ranjay Krishna. 2024a. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. *Advances in Neural Information Processing Systems*, 37:139348–139379.

Yushi Hu, Otilia Stretcu, Chun-Ta Lu, Krishnamurthy Viswanathan, Kenji Hata, Enming Luo, Ranjay Krishna, and Ariel Fuxman. 2024b. Visual program distillation: Distilling tools and programmatic reasoning into vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9590–9601.

Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. 2025. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. *arXiv preprint arXiv:2507.16815*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Yuheng Ji, Huajie Tan, Jiayu Shi, Xiaoshuai Hao, Yuan Zhang, Hengyuan Zhang, Pengwei Wang, Mengdi Zhao, Yao Mu, Pengju An, and 1 others. 2025. Robobrain: A unified brain model for robotic manipulation from abstract to concrete. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1724–1734.

NVIDIA, :, Alisson Azzolini, Junjie Bai, Hannah Brandon, Jiaxin Cao, Prithvijit Chattopadhyay, Huayu Chen, Jinju Chu, Yin Cui, Jenna Diamond, Yifan Ding, Liang Feng, Francesco Ferroni, Rama Govindaraju, Jinwei Gu, Siddharth Gururani, Imad El Hanafi, Zekun Hao, and 35 others. 2025. Cosmos-reason1: From physical common sense to embodied reasoning. *Preprint*, arXiv:2503.15558.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, and 401 others. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

OpenAI. 2025. Gpt-5 system card.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*.

Huajie Tan, Yuheng Ji, Xiaoshuai Hao, Xiansheng Chen, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. 2025. Reason-rft: Reinforcement fine-tuning for visual reasoning of vision language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Haozhe Wang, Alex Su, Weiming Ren, Fangzhen Lin, and Wenhu Chen. 2025a. Pixel reasoner: Incentivizing pixel-space reasoning with curiosity-driven reinforcement learning. *arXiv preprint arXiv:2505.15966*.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, and 1 others. 2025b. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.

Junfei Wu, Jian Guan, Kaituo Feng, Qiang Liu, Shu Wu, Liang Wang, Wei Wu, and Tieniu Tan. 2025. Reinforcing spatial reasoning in vision-language models with interwoven thinking and visual drawing. *Preprint*, arXiv:2506.09965.

Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. 2025. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10632–10643.

Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. 2024. Robopoint: A vision-language model for spatial affordance prediction for robotics. *Preprint*, arXiv:2406.10721.

Yifu Yuan, Haiqin Cui, Yibin Chen, Zibin Dong, Fei Ni, Longxin Kou, Jinyi Liu, Pengyi Li, Yan Zheng, and Jianye Hao. 2025a. From seeing to doing: Bridging reasoning and decision for robotic manipulation. *Preprint*, arXiv:2505.08548.

Yifu Yuan, Haiqin Cui, Yaoting Huang, Yibin Chen, Fei Ni, Zibin Dong, Pengyi Li, Yan Zheng, and Jianye Hao. 2025b. Embodied-r1: Reinforced embodied reasoning for general robotic manipulation. *Preprint*, arXiv:2508.13998.

Jiahui Zhang, Yurui Chen, Yanpeng Zhou, Yueming Xu, Ze Huang, Jilin Mei, Junhui Chen, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, and Li Zhang. 2025a. From flatland to space: Teaching vision-language models to perceive and reason in 3d. *Preprint*, arXiv:2503.22976.

Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024. Vision-language models for vision tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5625–5644.

Yi Zhang, Che Liu, Xiancong Ren, Hanchu Ni, Shuai Zhang, Zeyuan Ding, Jiayu Hu, Hanzhe Shan, Zhenwei Niu, Zhaoyang Liu, and 1 others. 2025b. Pelican-vl 1.0: A foundation brain model for embodied intelligence. *arXiv preprint arXiv:2511.00108*.

Yi Zhang, Che Liu, Xiancong Ren, Hanchu Ni, Yingji Zhang, Shuai Zhang, Zeyuan Ding, Jiayu Hu, Haozhe Shan, Junbo Qi, and 1 others. 2025c. Bridging vlms and embodied intelligence with deliberate practice policy optimization. *arXiv preprint arXiv:2511.16602*.

Yi Zhang, Qiang Zhang, Xiaozhu Ju, Zhaoyang Liu, Jilei Mao, Jingkai Sun, Jintao Wu, Shixiong Gao, Shihan Cai, Zhiyuan Qin, Linkai Liang, Jiaxu Wang, Yiqun Duan, Jiahang Cao, Renjing Xu, and Jian Tang. 2025d. Embodiedvsr: Dynamic scene graph-guided chain-of-thought reasoning for visual spatial tasks. *Preprint*, arXiv:2503.11089.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, and 1 others. 2025. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1702–1713.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, and 1 others. 2023. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*.

Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. 2025. Deepeyes: Incentivizing" thinking with images" via reinforcement learning. *arXiv preprint arXiv:2505.14362*.

Enshen Zhou, Jingkun An, Cheng Chi, Yi Han, Shanyu Rong, Chi Zhang, Pengwei Wang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, and 1 others. 2025. Roborefer: Towards spatial referring with reasoning in vision-language models for robotics. *arXiv preprint arXiv:2506.04308*.

A Qualitative Case Studies

To illustrate the visual chain reasoning process enabled by E-ViC, we present three representative case studies demonstrating how the model leverages visual tools for spatial reasoning tasks. These examples showcase the iterative, tool-augmented reasoning that distinguishes our approach from purely text-based methods.

A.1 Case 1: Moving Trajectory

Task Description. Given an image containing a red target object, the model must determine the trajectory that an end effector should follow to reach the target. The answer should be formatted as an ordered list of waypoints.

E-ViC Process. As shown in Figure 5, the model employs a three-step visual reasoning strategy:

1. **Target Identification:** The model first invokes `draw_point` to mark the center of the red object at coordinates $(73, 129)$, establishing the goal position.

2. **Start Position Marking:** A second `draw_point` call places a marker at the current end effector position $(25, 90)$.

3. **Trajectory Visualization:** Finally, the model uses `draw_trajectory` to render the path connecting these waypoints, enabling visual verification of the planned motion.

Final Answer. The trajectory is determined as `\boxed{[(25, 90), (73, 129)]}`.

A.2 Case 2: Spatial Relation Recognition

Task Description. Given an image with multiple objects, the model must estimate real-world distances and determine which object is farther from a reference object (a white fluffy pillow at lower right). The options include: (A) black plastic mirror at left, (B) black glossy window at lower right, and (C) metallic silver bolt at lower right.

E-ViC Process. As depicted in Figure 6, the model employs spatial reasoning through visual annotation:

1. **Multi-Point Localization:** The model invokes `draw_point` with coordinates for all objects of interest: the black plastic mirror at $(103, 242)$, the black glossy window at $(423, 344)$, and the metallic silver bolt at $(341, 349)$.

2. **Distance Estimation:** With the visual markers rendered, the model reasons about the spatial arrangement and estimates relative distances from the reference pillow.

Final Answer. Based on the visual analysis, the black plastic mirror (option A) is determined to be the farthest: `\boxed{A}`.

A.3 Case 3: Iterative Refinement for Localization

Task Description. The model must identify the 2D pixel location of a silver metallic laptop at the center of the image, with coordinates rounded to three decimal places.

E-ViC Process. As shown in Figure 7, the model employs a two-stage refinement strategy:

1. **Bounding Box Detection:** The model first invokes `draw_box` to render a bounding box around the target laptop with coordinates $[324, 183, 525, 389]$. This establishes the region of interest and provides visual confirmation of object identification.

2. **Center Point Estimation:** The model calculates the midpoint of the bounding box and uses `draw_point` to mark the center at $(427, 289)$. This visual feedback enables verification of the localization accuracy.

Final Answer. The 2D location of the silver metallic laptop center is `\boxed{(427.0, 289.0)}`.

A.4 Discussion

These case studies highlight several key aspects of the E-ViC framework:

- **Adaptive Tool Selection:** The model autonomously selects appropriate visual tools based on task requirements—`draw_trajectory` for motion planning, `draw_point` for spatial reasoning, and `draw_box` followed by `draw_point` for precise localization.

- **Iterative Refinement:** Rather than producing answers in a single forward pass, E-ViC

Project page: https://evic-joy.github.io

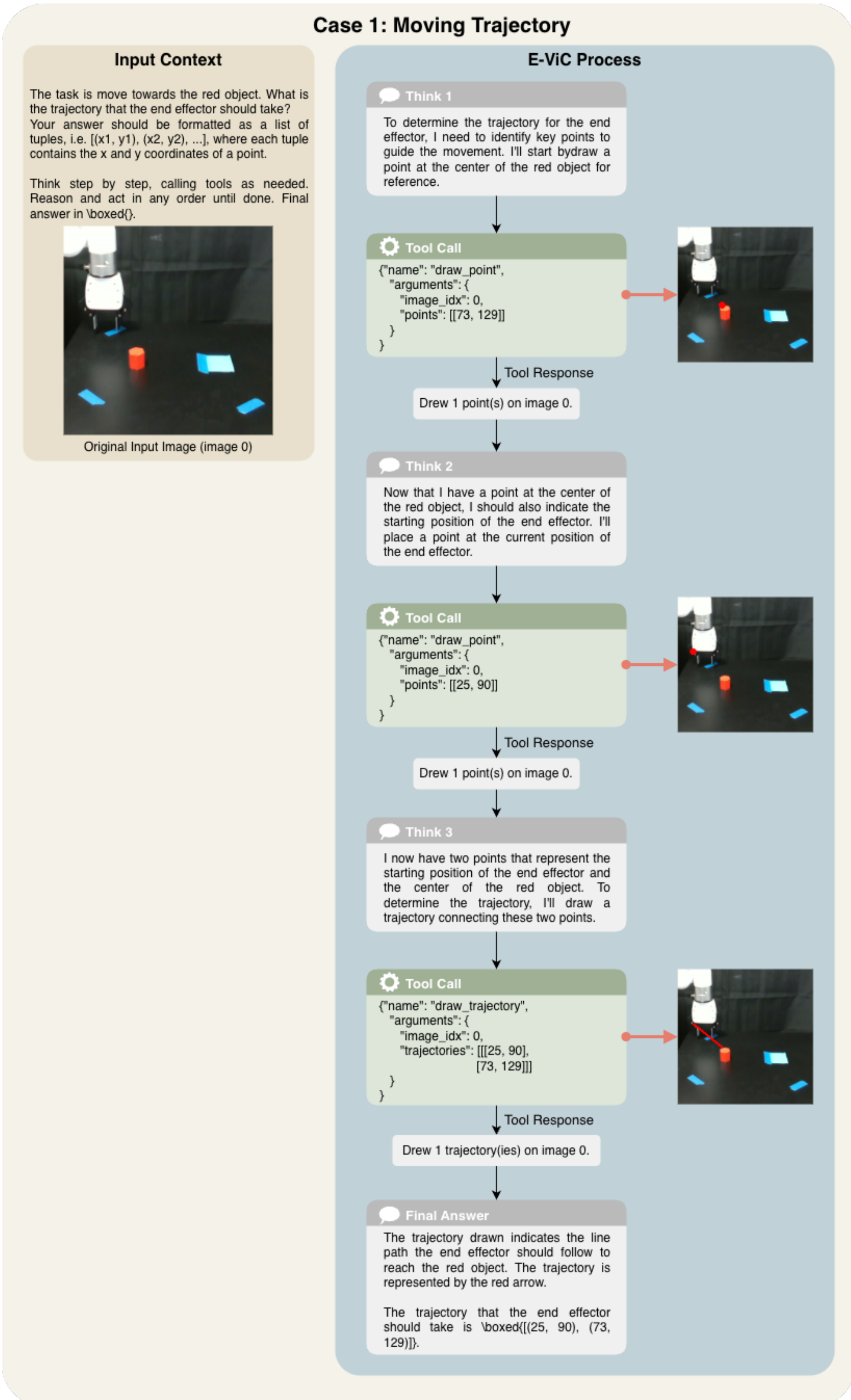

Figure 5: Moving Trajectory (Case 1). The model iteratively marks the start position, target location, and visualizes the trajectory using visual tool invocations.

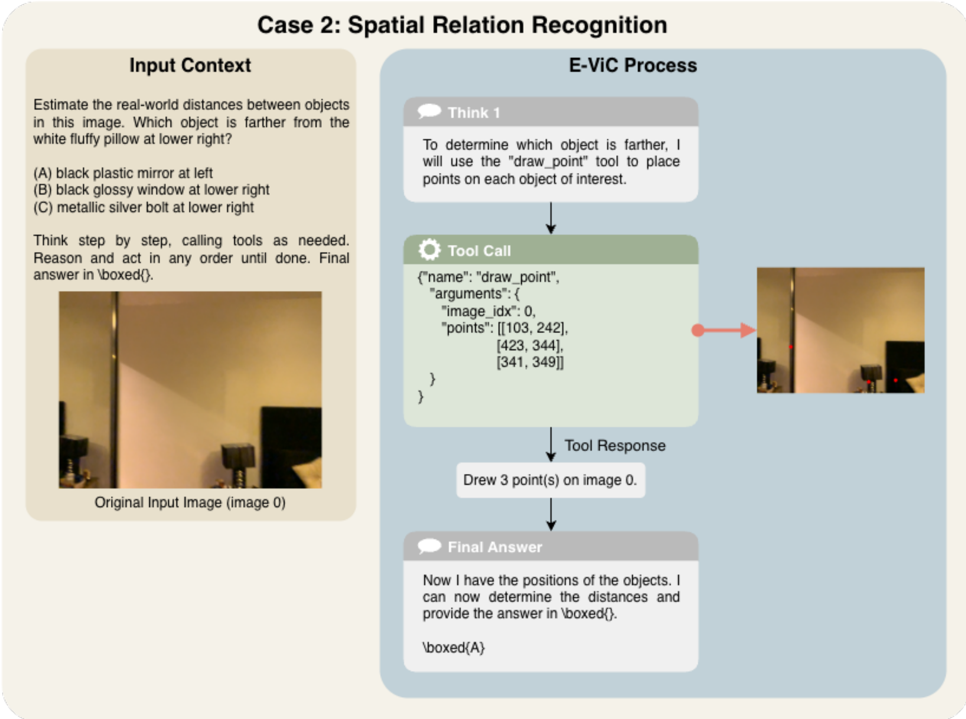

Figure 6: Spatial Relation Recognition (Case 2). The model places point markers on all relevant objects to facilitate distance comparison and spatial reasoning.

engages in multi-step visual reasoning, progressively refining its spatial understanding through tool feedback.

- **Visual Grounding:** By rendering annotations directly on the image, the model grounds its reasoning in fine-grained visual information rather than relying solely on linguistic abstraction.

These emergent behaviors arise naturally from the agentic RL training paradigm, without explicit supervision on tool-use trajectories.

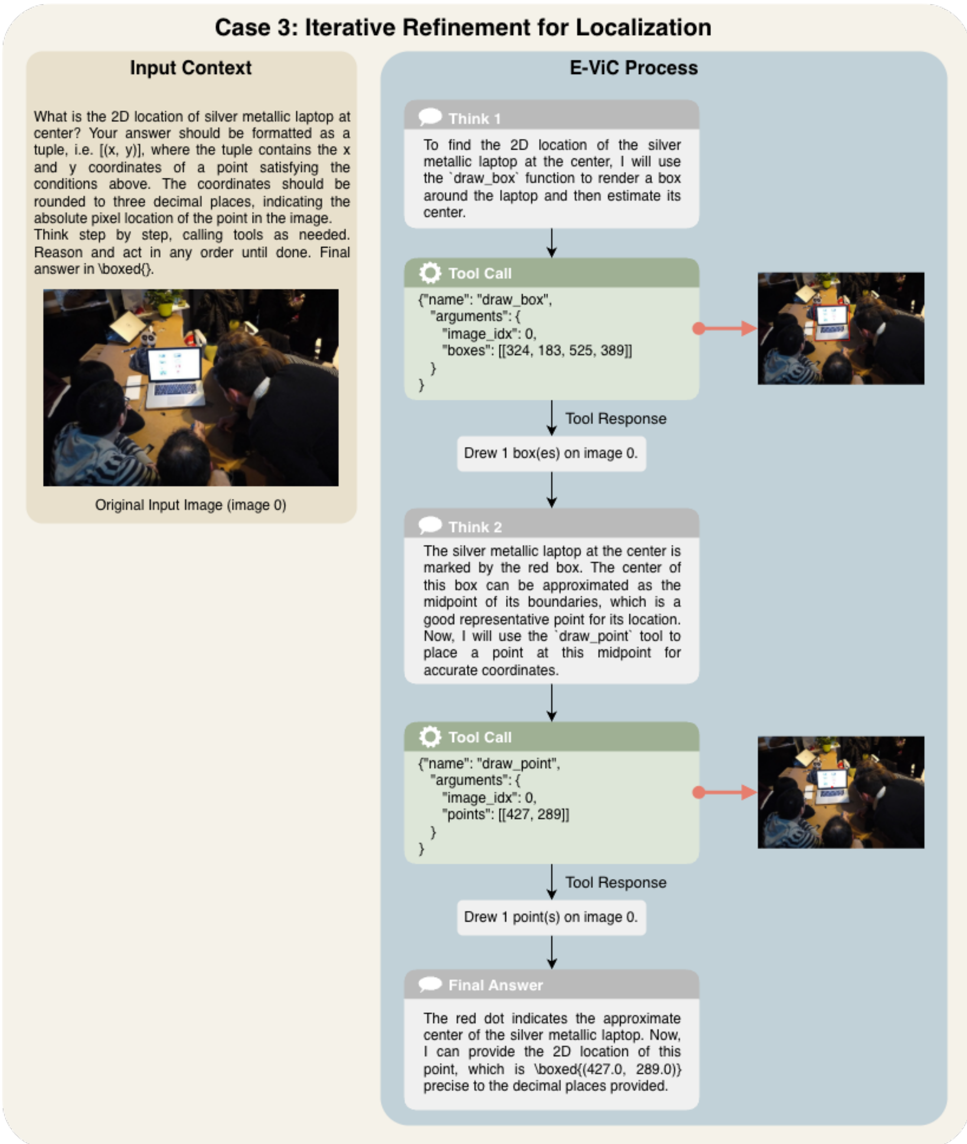

Figure 7: Iterative Refinement for Localization (Case 3). The model first draws a bounding box to localize the object, then refines the estimate by marking the center point.

Prompt Design for E-ViC

```
system
You are a helpful assistant. You can call functions to assist with the user query. Important: You must call only one function at a time. After each function call, wait for the execution result before making the next function call if needed.

# Tools

You may call one or more functions to assist with the user query.

You are provided with function signatures within <tools></tools> XML tags:
<tools>
{"type": "function", "function": {"name": "image_zoom_in", "description": "Crop and enlarge a specified rectangular region of the image for detailed inspection. The coordinate origin (0,0) is at the top-left corner.", "parameters": {"type": "object", "properties": {"image_idx": {"type": "integer", "description": "Index of the target image (starting from 0)"}, "bbox_2d": {"type": "array", "description": "Region coordinates as [x1, y1, x2, y2], where (x1, y1) is the top-left corner and (x2, y2) is the bottom-right corner. Requires x1 < x2 and y1 < y2."}, "label": {"type": "string", "description": "Optional descriptive annotation for the cropped region"}}, "required": ["bbox_2d"]}}}
{"type": "function", "function": {"name": "draw_point", "description": "Place visible point markers at specified locations on the image to indicate positions, objects, or areas of interest.", "parameters": {"type": "object", "properties": {"image_idx": {"type": "integer", "description": "Index of the target image (starting from 0)"}, "points": {"type": "array", "description": "List of point coordinates, each formatted as [x, y]"}}, "required": ["points"]}}}
{"type": "function", "function": {"name": "draw_box", "description": "Render rectangular bounding boxes on the image to highlight, localize, or annotate specific regions or objects.", "parameters": {"type": "object", "properties": {"image_idx": {"type": "integer", "description": "Index of the target image (starting from 0)"}, "boxes": {"type": "array", "description": "List of bounding boxes, each formatted as [x1, y1, x2, y2] where (x1, y1) is the top-left corner and (x2, y2) is the bottom-right corner"}}, "required": ["boxes"]}}}
{"type": "function", "function": {"name": "draw_trajectory", "description": "Draw connected paths on the image by linking sequential waypoints with lines. Suitable for visualizing motion, flow, or ordered spatial relationships.", "parameters": {"type": "object", "properties": {"image_idx": {"type": "integer", "description": "Index of the target image (starting from 0)"}, "trajectories": {"type": "array", "description": "List of trajectories, where each trajectory is an ordered sequence of at least 2 waypoints formatted as [x, y]"}}, "required": ["trajectories"]}}}
{"type": "function", "function": {"name": "draw_3d_bbox", "description": "Render 3D bounding boxes on the image using 8 projected corner points per box. Front face edges are drawn in red, back face in blue, and connecting edges in green.", "parameters": {"type": "object", "properties": {"image_idx": {"type": "integer", "description": "Index of the target image (starting from 0)"}, "boxes_3d": {"type": "array", "description": "List of 3D boxes, each defined by 8 corner points as [x, y]. Corners 0-3 define the front face, corners 4-7 define the back face. Corner order per face: top-left, top-right, bottom-right, bottom-left."}}, "required": ["boxes_3d"]}}}
</tools>

For each function call, return a json object with function name and arguments within <tool_call></tool_call> XML tags:
<tool_call>
{"name": <function-name>, "arguments": <args-json-object>}
</tool_call>
user

{Query}
Think step by step, calling tools as needed. Reason and act in any order until done. Final answer in \boxed{}.
```

Figure 8: Prompt design for agentic Think-with-Image reasoning. The system prompt defines the visual tool suite via JSON schemas and enforces the single-tool-per-turn constraint. User messages embed images via `<|image_pad|>` tokens. Tool responses inject visual feedback directly into the conversation context, enabling iterative refinement.

B Prompt Design

This section specifies the prompt architecture enabling agentic visual reasoning. Our design philosophy centers on *minimal intervention*: we provide tool affordances and let reinforcement learning discover effective usage patterns. The complete prompt is shown in Figure 8.

B.1 Chat Template Architecture

We extend the Qwen-VL chat template with native tool-calling capabilities. Table 2 summarizes the key tokens.

Token	Function
`<|im_start|>`, `<|im_end|>`	Message boundary
`<|vision_start|>`, `<|vision_end|>`	Visual content scope
`<|image_pad|>`	Image embedding
`<tool_call>`, `</tool_call>`	Tool invocation
`<tool_response>`, `</tool_response>`	Tool output

Table 2: Special tokens in the chat template.

The conversation follows strict role alternation: `system` $\to$ `user` $\to$ `assistant` $\to$ `tool` $\to$ `assistant` $\to \cdots$, where tool responses are formatted as user messages for template compatibility.

B.2 System Prompt

The system prompt establishes the single-tool-per-turn constraint critical for interpretable reasoning:

```
<|im_start|>system
You are a helpful assistant. You can call
functions to assist with the user query.
Important: You must call only one function
at a time. After each function call, wait
for the execution result before making the
next function call if needed.

# Tools
You are provided with function signatures
within <tools></tools> XML tags:
<tools>
{tool_schemas_in_json}
</tools>

For each function call, return a json object
within <tool_call></tool_call> XML tags:
<tool_call>
{"name": <function-name>,
 "arguments": <args-json-object>}
</tool_call><|im_end|>
```

B.3 Visual Context Accumulation

A key feature enabling Think-with-Image is the native embedding of tool outputs as images. Each tool response appends a new image to the visual context:

$$\mathcal{I}_{t+1} = \mathcal{I}_t \cup \{I_{\text{tool}}\} \tag{7}$$

Tool responses are injected with both textual feedback and visual output:

```
<|im_start|>user
<tool_response>
Zoomed in on image 0 to region [120,80,350,240].
<|vision_start|><|image_pad|><|vision_end|>
</tool_response><|im_end|>
```

This enables the model to *see* what the tool produced, supporting visual verification and iterative refinement—the core of agentic visual reasoning.

B.4 Multi-Turn Interaction Example

Algorithm 2 illustrates the Think-Act-Observe loop with actual token structure.

The visual context grows as $|\mathcal{I}| : 1 \to 2 \to 3$, enabling the model to compare annotations against the original image for verification.

B.5 Design Principles

Our design specifies tool availability without prescribing usage, enabling RL-discovered strategies. Tool outputs are embedded as images for closed-loop visual verification, while sequential

Algorithm 2 Agentic Think-with-Image Interaction Protocol

Require: Image set $\mathcal{I}_0$, instruction l, tool suite $\mathcal{T}$, max turns $T_{\max}$
Ensure: Final answer y

/ Context Initialization */*
$\mathcal{H} \leftarrow$ [<|im_start|>system $\circ\, \sigma(\mathcal{T}) \circ$ <|im_end|>]
$\mathcal{H} \leftarrow \mathcal{H} \circ$ [<|im_start|>user $\circ$ EMBED$(\mathcal{I}_0) \circ l \circ$ <|im_end|>]

for $t = 1$ **to** $T_{\max}$ **do**
  / **Think**: Generate reasoning and action */*
  <|im_start|>assistant
  $o_t \sim \pi_\theta(\cdot \mid \mathcal{H})$ ▷ *Reasoning + optional tool call*
  $\mathcal{H} \leftarrow \mathcal{H} \circ [o_t \circ$ <|im_end|>]

  if HASBOXEDANSWER(o_t) **then**
    return EXTRACT(o_t) ▷ *Terminal action*
  end if

  if HASTOOLCALL(o_t) **then**
    / **Act**: Execute visual tool */*
    $(f_k, \texttt{args}) \leftarrow$ PARSETOOLCALL(o_t)
    $I_{\text{out}}, \texttt{msg} \leftarrow f_k(\mathcal{I}_t, \texttt{args})$

    / **Observe**: Inject visual feedback */*
    <|im_start|>user
    <tool_response>
    $\mathcal{H} \leftarrow \mathcal{H} \circ [\texttt{msg} \circ$ EMBED$(I_{\text{out}})]$
    </tool_response><|im_end|>
    $\mathcal{I}_{t+1} \leftarrow \mathcal{I}_t \cup \{I_{\text{out}}\}$ ▷ *Visual context growth*
  end if
end for
return EXTRACT$(o_{T_{\max}})$

single-tool execution ensures interpretable reasoning chain.

C Comparison with Spatial VLMs

Table 3 compares E-ViC against representative spatial VLM baselines: Cosmos-Reason1 (NVIDIA et al., 2025), RoboPoint-13B (Yuan et al., 2024), and FSD-13B (Yuan et al., 2025a), evaluated on five benchmarks: Where2Place, RefSpatial-Bench, ERQA, SparBench, and MMSI. Our model (highlighted in blue) achieves the best average score despite using a smaller backbone than the 13B-scale baselines.

D Task-Specific Accuracy Rewards

In Eq. 1, we set $\alpha = 1.0$, $\beta = 1.0$. This section provides detailed formulations for each task-specific accuracy reward $R_k(\cdot)$ used in our multi-task reward framework.

D.1 Choice Reward

For multiple-choice questions, we employ a binary accuracy metric. Given a predicted answer $\hat{y}$ and ground truth y, the choice reward is defined as:

$$R_{\text{choice}} = \mathbb{1}[\text{norm}(\hat{y}) = \text{norm}(y)] \tag{8}$$

where $\text{norm}(\cdot)$ converts answers to uppercase and removes trailing punctuation (e.g., parentheses, periods).

Model	W2P	RSB	ERQA	SpB	MMSI	Avg
E-ViC (7B)	**47.0**	**28.2**	**43.3**	**37.7**	**30.2**	**37.3**
Cosmos-Reason1	19.0	4.9	38.3	31.1	11.6	21.0
RoboPoint-13B	46.0	16.9	29.3	22.9	20.8	27.2
FSD-13B	45.8	8.6	27.3	19.7	22.9	24.9

Table 3: Comparison with spatial VLM baselines. Our method E-ViC (highlighted in blue) achieves the best average score. Abbreviations: **W2P** = Where2Place, **RSB** = RefSpatial-Bench, **SpB** = SparBench; ERQA and MMSI keep their original names. Avg is the mean across the five benchmarks.

D.2 MRA Reward

For regression and numerical prediction tasks, we adopt the Mean Relative Accuracy (MRA) metric (Eigen et al., 2014). Given a predicted value $\hat{v}$ and ground truth value v, the relative error is computed as:

$$\epsilon = \frac{|\hat{v} - v|}{|v|} \tag{9}$$

For the special case where $v = 0$, we define $R_{\text{MRA}} = 1$ if $|\hat{v}| < \epsilon_0$, and $R_{\text{MRA}} = 0$ otherwise. The MRA reward evaluates accuracy across K confidence thresholds $\Theta = \{\theta_1, \theta_2, \ldots, \theta_K\}$:

$$R_{\text{MRA}} = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}[\epsilon < 1 - \theta_k] \tag{10}$$

The continuous MRA value is directly used as the accuracy reward:

$$R_{\text{acc}}^{\text{reg}} = R_{\text{MRA}} \tag{11}$$

In our experiments, we set $\epsilon_0 = 10^{-6}$, $K = 10$, and $\Theta = \{0.50, 0.55, 0.60, 0.65, 0.70, 0.75, 0.80, 0.85, 0.90, 0.95\}$.

D.3 Point Localization Rewards

We employ two distinct reward formulations for point prediction tasks, depending on the ground-truth annotation structure.

D.3.1 Region-Based Gaussian Reward

For affordance prediction tasks such as RoboPoint (Yuan et al., 2024), where the ground truth specifies a target *region* via multiple bounding points $\mathcal{G} = \{g_1, \ldots, g_M\}$ ($M \geq 4$), we employ a Gaussian reward that provides continuous feedback based on proximity to the region center. We first compute the region center and variance:

$$c_x = \frac{\min_j g_j^x + \max_j g_j^x}{2}, \quad c_y = \frac{\min_j g_j^y + \max_j g_j^y}{2} \tag{12}$$

$$\sigma_x = \alpha \cdot (\max_j g_j^x - \min_j g_j^x), \quad \sigma_y = \alpha \cdot (\max_j g_j^y - \min_j g_j^y) \tag{13}$$

where α controls the Gaussian width relative to the region extent.

For each predicted point $p_i = (p_i^x, p_i^y)$, the Gaussian reward is:

$$R_{\text{gauss}}(p_i) = \exp\left(-\frac{1}{2}\left(\frac{(p_i^x - c_x)^2}{\sigma_x^2} + \frac{(p_i^y - c_y)^2}{\sigma_y^2}\right)\right) \tag{14}$$

The final region-based point reward averages over all predictions $\mathcal{P} = \{p_1, \ldots, p_N\}$:

$$R_{\text{region}} = \frac{1}{N}\sum_{i=1}^{N} R_{\text{gauss}}(p_i) \tag{15}$$

We set $\alpha = 0.5$ in our experiments.

D.3.2 Single-Point Gaussian Reward

For tasks requiring prediction of a single precise coordinate (e.g., object center localization), where the ground truth is a single point $g = (g^x, g^y)$, we employ a sharper isotropic Gaussian reward. Given predicted points $\mathcal{P} = \{p_1, \ldots, p_N\}$, we select the closest prediction to the ground truth:

$$p^* = \arg\min_{p_i \in \mathcal{P}} \|p_i - g\|_2 \tag{16}$$

The single-point reward is then computed as:

$$R_{\text{single}} = \exp\left(-\frac{\|p^* - g\|_2^2}{2\sigma_{\text{pt}}^2}\right) \tag{17}$$

where σ_{pt} (in normalized $[0, 1]$ coordinates) controls the sharpness. We set $\sigma_{\text{pt}} = 0.1$, which yields rewards of approximately 0.61 at distance 0.1 and 0.14 at distance 0.2, providing strong gradients toward the exact target.

D.4 Trajectory Reward

For trajectory prediction tasks, we design a reward function that evaluates both endpoint accuracy and path coverage. Given a predicted trajectory $\mathcal{P} = \{p_1, \ldots, p_N\}$ and ground truth trajectory $\mathcal{G} = \{g_1, \ldots, g_M\}$, coordinates are scaled from normalized $[0, 1]$ space to pixel coordinates by multiplying with image width W. The maximum distance is defined as $d_{\max} = W\sqrt{2}$.

Bidirectional Path Distance. To prevent degenerate solutions where predictions collapse to few points (e.g., only start and end), we compute a bidirectional average distance that penalizes incomplete coverage. The GT-to-prediction distance measures *coverage*:

$$d_{\text{GT}\to\mathcal{P}} = \frac{1}{M}\sum_{j=1}^{M} \min_{i \in [N]} \|g_j - p_i\|_2 \tag{18}$$

while the prediction-to-GT distance measures *accuracy*:

$$d_{\mathcal{P}\to\text{GT}} = \frac{1}{N}\sum_{i=1}^{N} \min_{j \in [M]} \|p_i - g_j\|_2 \tag{19}$$

The bidirectional distance combines both terms symmetrically:

$$d_{\text{bi}} = \frac{1}{2}\left(d_{\text{GT}\to\mathcal{P}} + d_{\mathcal{P}\to\text{GT}}\right) \tag{20}$$

The trajectory component reward uses exponential decay with scaling factor β:

$$R_{\text{traj}} = \exp\left(-\frac{d_{\text{bi}}}{\beta \cdot d_{\max}}\right) \tag{21}$$

where β controls the sensitivity to path deviations.

Endpoint Reward. We emphasize accurate start and endpoint prediction, with higher weight on the endpoint reflecting its importance for action completion:

$$R_{\text{goal}} = \lambda_s \cdot \exp\left(-\frac{\|p_1 - g_1\|_2}{d_{\max}}\right) + \lambda_e \cdot \exp\left(-\frac{\|p_N - g_M\|_2}{d_{\max}}\right) \tag{22}$$

where λ_s and λ_e denote the weights for start and endpoint accuracy, respectively, with $\lambda_s + \lambda_e = 1$.

Combined Trajectory Reward. The final trajectory reward combines these components with weighted coefficients:

$$R_{\text{trajectory}} = \omega_{\text{goal}} \cdot R_{\text{goal}} + \omega_{\text{traj}} \cdot R_{\text{traj}} \quad (23)$$

where ω_{goal} and ω_{traj} balance the contribution of endpoint accuracy and path quality. The result is clipped to $[0, 1]$.

In our experiments, we set $W = 256$, $\beta = 0.1$, $\lambda_s = 0.4$, $\lambda_e = 0.6$, $\omega_{\text{goal}} = 0.3$, and $\omega_{\text{traj}} = 0.7$.

D.5 IoU Reward

For 2D bounding box detection tasks, we employ the Intersection over Union (IoU) metric (Everingham et al., 2010). Given a predicted bounding box $\hat{B} = (\hat{x}_1, \hat{y}_1, \hat{x}_2, \hat{y}_2)$ and ground truth box $B = (x_1, y_1, x_2, y_2)$ in normalized coordinates:

$$\text{IoU}(\hat{B}, B) = \frac{|\hat{B} \cap B|}{|\hat{B} \cup B|} \quad (24)$$

where the intersection area is computed as:

$$|\hat{B} \cap B| = \max(0, x_2^{\text{int}} - x_1^{\text{int}}) \cdot \max(0, y_2^{\text{int}} - y_1^{\text{int}}) \quad (25)$$

with $x_1^{\text{int}} = \max(\hat{x}_1, x_1)$, $x_2^{\text{int}} = \min(\hat{x}_2, x_2)$, and similarly for y coordinates. The IoU reward directly uses the continuous IoU value:

$$R_{\text{IoU}} = \text{IoU}(\hat{B}, B) \quad (26)$$

E Visual Tool Suite Specification

This appendix provides formal specifications for the visual tool suite employed in our agentic framework. Each tool operates on pixel-coordinate inputs and returns modified images that augment the agent's evolving visual context. We first present an overview of the tool suite architecture (§E.1), followed by detailed algorithmic specifications for each tool (§E.2–§E.6).

E.1 Overview

Our framework equips the agent with five visual tools that collectively enable hierarchical spatial reasoning across multiple levels of abstraction. Table 4 summarizes the input-output specifications for each tool.

Design Principles. The tool suite is designed around four core principles:

Tool	Input Specification	Output
ZoomIn	$\mathbf{b} \in \mathbb{R}^4$	Cropped region
DrawPoint	$\mathcal{P} \subset \mathbb{R}^2$	Marked image
DrawBox	$\mathcal{B} \subset \mathbb{R}^4$	Boxed image
DrawTraj	$\mathcal{T} \subset (\mathbb{R}^2)^*$	Path overlay
Draw3D	$\mathcal{B}_{3D} \subset (\mathbb{R}^2)^8$	3D wireframe

Table 4: Visual tool specifications. All coordinates are defined in pixel space with origin $(0, 0)$ at the top-left corner. Notation: $(\mathbb{R}^2)^*$ denotes variable-length sequences of 2D points.

1. *Multi-image support*: All tools operate over an indexed image collection $\mathcal{I} = \{I_0, I_1, \ldots, I_{K-1}\}$, enabling reasoning across multiple views, frames, or scene perspectives. Target images are selected via an explicit `image_idx` parameter.
2. *Single-tool-per-turn execution*: Each reasoning turn permits at most one tool invocation, enforcing deliberate and interpretable reasoning steps rather than parallel tool execution.
3. *Compositionality*: Tools can be chained across turns to perform complex multi-step visual reasoning.
4. *Reversibility*: Original images are preserved in the context, enabling the agent to reference unmodified observations alongside annotated versions.

Multi-Image Indexing. For tasks involving multiple input images (e.g., multi-view reasoning, video understanding, or comparative analysis), the agent maintains an ordered image collection $\mathcal{I}$. Each tool accepts an optional `image_idx` $\in \{0, 1, \ldots, |\mathcal{I}| - 1\}$ parameter specifying the target image. When omitted, the default index is 0. This design enables:

- Cross-view spatial reasoning by annotating corresponding regions across different viewpoints
- Temporal analysis by marking motion trajectories across video frames
- Comparative inspection by zooming into analogous regions in paired images

Turn-Based Execution Model. To promote structured reasoning and ensure interpretability, our framework enforces a *single-tool-per-turn* constraint. At each reasoning step t, the policy π_θ generates exactly one of:

Algorithm 3 IMAGEZOOMIN($I_k, \mathbf{b}, \ell$)

Require: Image $I_k \in \mathbb{R}^{H \times W \times 3}$ selected by index k, bounding box $\mathbf{b} = (x_1, y_1, x_2, y_2)$, optional label ℓ
Ensure: Cropped image $I' \in \mathbb{R}^{H' \times W' \times 3}$ or error message

// Coordinate validation and clamping
$(x_1, y_1) \leftarrow (\max(0, x_1), \max(0, y_1))$
$(x_2, y_2) \leftarrow (\min(W, x_2), \min(H, y_2))$
if $x_1 \geq x_2 \vee y_1 \geq y_2$ **then**
    return ERROR("Invalid bounding box: requires $x_1 < x_2 \wedge y_1 < y_2$")
end if
// Minimum dimension enforcement
if $\min(x_2 - x_1, y_2 - y_1) < d_{\min}$ **then**
    $\mathbf{b} \leftarrow$ EXPANDTOMINIMUM($\mathbf{b}, d_{\min}, W, H$)
end if
// Aspect ratio validation
if $\max(x_2 - x_1, y_2 - y_1) / \min(x_2 - x_1, y_2 - y_1) > \rho_{\max}$ **then**
    return ERROR("Aspect ratio exceeds maximum threshold $\rho_{\max}$")
end if
return $I_k[y_1 : y_2, x_1 : x_2]$

- A single tool invocation $(f_k, \texttt{args})$ where $f_k \in \mathcal{T}$
- A terminal action producing the final answer $y \in \mathcal{A}$

This constraint prevents degenerate behaviors such as redundant parallel annotations and encourages the agent to develop purposeful, sequential reasoning strategies. The maximum number of turns is bounded by $T_{\max}$ to ensure computational tractability.

E.2 Image Zoom-In

The ZOOMIN tool enables multi-scale visual analysis by extracting and enlarging specified image regions. This operation is fundamental for fine-grained spatial reasoning where global context provides insufficient resolution.

Parameters.

- `image_idx`: Index $k \in \{0, \ldots, |\mathcal{I}| - 1\}$ selecting the target image from the collection (default: 0).
- `bbox_2d`: Region coordinates (x_1, y_1, x_2, y_2) specifying top-left and bottom-right corners.
- `label`: Optional semantic annotation for the cropped region.

Geometric Constraints. The tool enforces several validity constraints: (i) proper ordering $x_1 < x_2 \wedge y_1 < y_2$; (ii) bounds containment within $[0, W] \times [0, H]$; (iii) minimum dimension $d_{\min} =$ 28 pixels to ensure visual interpretability; and (iv) maximum aspect ratio $\rho_{\max} = 100$ to prevent degenerate crops.

Algorithm 4 DRAWPOINT($I_k, \mathcal{P}$)

Require: Image $I_k \in \mathbb{R}^{H \times W \times 3}$ selected by index k, point set $\mathcal{P} = \{(x_i, y_i)\}_{i=1}^N$
Ensure: Annotated image I'_k or error message

$I'_k \leftarrow$ COPY(I_k)
for $(x, y) \in \mathcal{P}$ **do**
    if $x \notin [0, W] \vee y \notin [0, H]$ **then**
        return ERROR("Point (x, y) exceeds image bounds")
    end if
    RENDERCIRCLE($I'_k, (x, y), r, \texttt{color} = \text{red}, \texttt{fill} = \text{True}$)
end for
return I'_k

Algorithm 5 DRAWBOX($I_k, \mathcal{B}$)

Require: Image $I_k \in \mathbb{R}^{H \times W \times 3}$ selected by index k, box set $\mathcal{B} = \{\mathbf{b}_j\}_{j=1}^M$
Ensure: Annotated image I'_k or error message

$I'_k \leftarrow$ COPY(I_k)
for $\mathbf{b} = (x_1, y_1, x_2, y_2) \in \mathcal{B}$ **do**
    if $\neg$ISVALIDBBOX($\mathbf{b}, W, H$) **then**
        return ERROR("Invalid bounding box coordinates")
    end if
    RENDERRECTANGLE($I'_k, \mathbf{b}, \texttt{color} = \text{red}, \texttt{width} = 3$)
end for
return I'_k

E.3 Draw Point

The DRAWPOINT tool renders visible markers at specified locations, enabling keypoint annotation and spatial hypothesis externalization.

Parameters.

- `image_idx`: Index k selecting the target image (default: 0).
- `points`: List of 2D coordinates $[(x_1, y_1), \ldots, (x_N, y_N)]$.

Rendering. Points are rendered as filled circles with radius $r = 5$ pixels in red (`#FF0000`), providing sufficient visibility against diverse backgrounds.

E.4 Draw Bounding Box

The DRAWBOX tool renders rectangular annotations for region-level spatial reasoning, supporting object localization and area demarcation.

Algorithm 6 DRAWTRAJECTORY$(I_k, \mathcal{T})$

Require: Image $I_k \in \mathbb{R}^{H\times W\times 3}$ selected by index k, trajectory set $\mathcal{T} = \{\tau_j\}_{j=1}^{J}$
Ensure: Annotated image I'_k or error message

$I'_k \leftarrow$ COPY(I_k)
$\mathcal{C} \leftarrow$ [red, blue, green, yellow, orange, purple, cyan, magenta]
for $j = 1, \dots, J$ **do**
  if $|\tau_j| < 2$ **then**
    return ERROR("Trajectory requires $\geq$ 2 waypoints")
  end if
  $c \leftarrow \mathcal{C}[j \mod |\mathcal{C}|]$
  for $k = 1, \dots, |\tau_j| - 1$ **do**
    RENDERLINE$(I'_k, \tau_j[k], \tau_j[k+1], \texttt{color} = c, \texttt{width} = 3)$
  end for
end for
return I'_k

Parameters.

- `image_idx`: Index k selecting the target image (default: 0).
- `boxes`: List of bounding boxes, each as (x_1, y_1, x_2, y_2).

Coordinate Convention. Bounding boxes follow the $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$ format. Boxes are rendered as red outlines with stroke width of 3 pixels.

E.5 Draw Trajectory

The DRAWTRAJECTORY tool visualizes motion paths by connecting sequential waypoints, supporting action planning and temporal-spatial relationship reasoning.

Parameters.

- `image_idx`: Index k selecting the target image (default: 0).
- `trajectories`: List of trajectories, each an ordered sequence of ≥ 2 waypoints.

Multi-Trajectory Support. Multiple trajectories are rendered with distinct colors via cyclic indexing over an 8-color palette, enabling visualization of alternative motion hypotheses.

E.6 Draw 3D Bounding Box

The DRAW3DBBOX tool renders volumetric annotations using 8-corner wireframe projections, enabling spatial reasoning about object extent and orientation in three-dimensional space.

Algorithm 7 DRAW3DBBOX$(I_k, \mathcal{B}_{3D})$

Require: Image $I_k \in \mathbb{R}^{H\times W\times 3}$ selected by index k, 3D box set $\mathcal{B}_{3D} = \{\mathbf{B}_j\}_{j=1}^{M}$
Ensure: Annotated image I'_k or error message

$I'_k \leftarrow$ COPY(I_k)
for $\mathbf{B} = [(x_0, y_0), \dots, (x_7, y_7)] \in \mathcal{B}_{3D}$ **do**
  if $|\mathbf{B}| \neq 8$ **then**
    return ERROR("3D box requires exactly 8 corners")
  end if
  // Front face: corners 0–3 (red)
  for $i = 0, 1, 2, 3$ **do**
    RENDERLINE$(I'_k, \mathbf{B}[i], \mathbf{B}[(i+1) \mod 4], \text{red})$
  end for
  // Back face: corners 4–7 (blue)
  for $i = 0, 1, 2, 3$ **do**
    RENDERLINE$(I'_k, \mathbf{B}[4+i], \mathbf{B}[4+(i+1) \mod 4], \text{blue})$
  end for
  // Depth edges: front-to-back connections (green)
  for $i = 0, 1, 2, 3$ **do**
    RENDERLINE$(I'_k, \mathbf{B}[i], \mathbf{B}[i+4], \text{green})$
  end for
end for
return I'_k

Parameters.

- `image_idx`: Index k selecting the target image (default: 0).
- `boxes_3d`: List of 3D boxes, each containing exactly 8 corner points.

Corner Ordering Convention. Each 3D bounding box is defined by 8 projected corner points:

- **Front face** (indices 0–3): Closer to camera, rendered in red. Order: top-left $\to$ top-right $\to$ bottom-right $\to$ bottom-left.
- **Back face** (indices 4–7): Farther from camera, rendered in blue. Same clockwise ordering.
- **Depth edges**: Connections $0 \leftrightarrow 4$, $1 \leftrightarrow 5$, $2 \leftrightarrow 6$, $3 \leftrightarrow 7$, rendered in green.

E.7 Execution Semantics

Image Index Validation. All tools validate the `image_idx` parameter against the current image collection size. An out-of-bounds index $k \geq |\mathcal{I}|$ returns an error message specifying the valid range $[0, |\mathcal{I}| - 1]$, enabling the agent to recover and retry with a corrected index.

Single-Tool Constraint Enforcement. The execution environment parses model outputs for tool invocations and enforces the single-tool-per-turn constraint at the infrastructure level. If multiple <tool_call> blocks are detected within a single turn, only the first is executed; subsequent invocations are ignored with a warning logged for analysis.

Error Handling. All tools implement comprehensive input validation following a hierarchical scheme:

1. **Index validation**: `image_idx` $\in \{0, \ldots, |\mathcal{I}| - 1\}$
2. **Type checking**: Input arrays validated for correct dimensionality
3. **Cardinality constraints**: Minimum element counts enforced
4. **Bounds checking**: Coordinates validated against $[0, W] \times [0, H]$
5. **Geometric validity**: Tool-specific constraints verified

Failed executions return descriptive error messages and incur a small negative reward ($r = -0.05$) to discourage malformed invocations while preserving gradient signal.

E.8 Tool Invocation Protocol

Tools are invoked via structured JSON within XML delimiters. Each turn permits *at most one* tool call:

```
<tool_call>
{"name": "draw_point",
 "arguments": {"image_idx": 2,
              "points": [[156, 234], [312, 189]]}}
</tool_call>
```

In this example, the agent annotates the *third* image ($k = 2$) in a multi-image collection. The resulting annotated image is appended to the visual context, enabling the agent to condition subsequent reasoning on the feedback. This closed-loop, turn-based interaction continues until the model produces a terminal action or reaches $T_{\max}$.