THYMEV2: TOWARDS INFINITE VISUAL EXPLORATION WITH FINITE CONTEXT

Anonymous authors
Paper under double-blind review

ABSTRACT

Complex perception tasks often require multimodal models to explore and refine visual evidence over multiple interactions. Recent multimodal agents have begun to demonstrate such behavior, yet scaling this capability toward sustained exploration remains challenging. Current systems often operate within limited exploration horizons, and extending exploration naturally causes visual contexts to grow rapidly. In this work we introduce **ThymeV2**, a multimodal model for sustained active perception in images. ThymeV2 combines adaptive multi-turn visual exploration with an evolving visual memory, allowing the model to acquire new evidence while selectively maintaining useful observations. To unlock this capability, we design visual tools for multi-region image acquisition and removing unnecessary observations, construct large-scale multi-turn exploration dataset and develop a two-stage post-training recipe. We first perform supervised fine-tuning (SFT) on turn-level examples using the visual context available at each turn. Following SFT, our reinforcement learning (RL) recipe combines outcome and process signals with fine-grained control over policy updates for stable optimization. Together, these designs support deep exploration, self-correction, and stable learning over extended interactions. We also introduce **StratBenchFinal**, a challenging visual exploration benchmark with 893 manually reviewed examples across five difficulty levels. ThymeV2 achieves performance comparable to Gemini 3.1 Pro with the same tools and outperforms Gemini 3.1 Pro without tools by up to **12.76** percentage points on average across six challenging perception benchmarks.We will release our models, data, and code to facilitate future research.

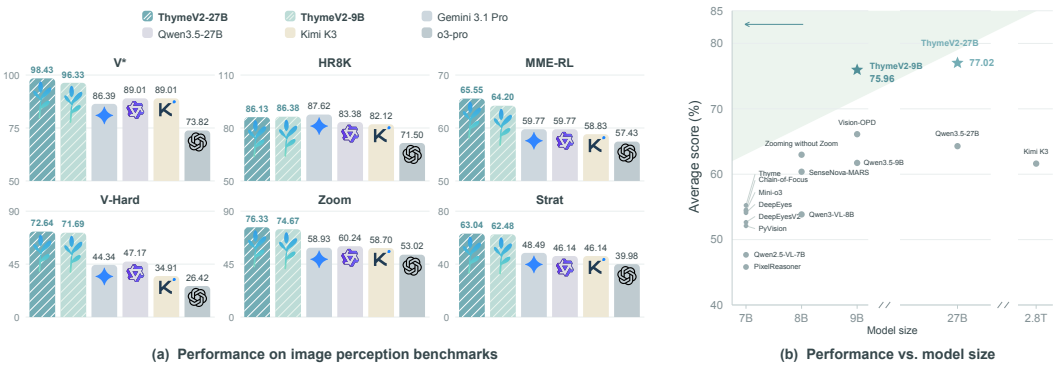
1 INTRODUCTION

Recent frontier multimodal large language models (MLLMs) have made substantial progress in visual understanding (Li et al., 2024; Team, 2026b; Team et al., 2026; DeepMind, 2026; Team, 2026a; Wang et al., 2025a). However, difficult visual tasks often require more than a single observation of the input. Rather than being limited to what is available from the initial observation, a visual agent should expand its perceptual range through continued exploration: using its current understanding to decide what to inspect next, acquiring new evidence through action, and updating its understanding to guide subsequent actions. This perspective echoes Gibson's account of perception and action:

> *"We must perceive in order to move, but we must also move in order to perceive."*
> The Ecological Approach to Visual Perception (Gibson, 1978, p. 223)

Such a perspective suggests that exploration itself may become a new scaling axis for visual perception. OpenAI o3 and o4-mini brought "thinking with images" to broad attention, showing that multimodal models can continue inspecting and manipulating visual inputs during inference (OpenAI, 2025). This capability has motivated a growing line of work on "thinking with images" (Su et al., 2025; Zheng et al., 2026; Wu et al., 2026a; Zhang et al., 2025b; Hong et al., 2026; Lai et al., 2026). Recent results further suggest that difficult visual tasks continue to benefit from deeper interaction (Lai et al., 2026; Zeng et al., 2026).For example, Mini-o3 demonstrates improved performance as the interaction budget increases (Appendix G). However, existing systems only partially realize sustained visual exploration. The first challenge is that current systems still operate within limited exploration horizons. Although these models can invoke visual tools, they often show limited initiative in visual exploration, and even longer interaction trajectories can involve substantial repetition

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



(a) Performance on image perception benchmarks (b) Performance vs. model size

Figure 1: Performance comparison on image perception benchmarks.

rather than sustained acquisition of new visual evidence (Appendix H.1). Extending exploration horizons introduces a second challenge: managing the growing visual context during interaction. Each additional exploration step can introduce new visual observations, which accumulate in the context available to later decisions. Recent efforts toward longer interactions often retain these observations throughout the trajectory (Lai et al., 2026; Zhao et al., 2026).

In this work, we take a step toward scaling visual perception through exploration and introduce **ThymeV2**, a multimodal model for sustained active perception. To extend exploration capability, ThymeV2 designs visual tools for multi-region image acquisition and observation removal, allowing the model to gather new evidence while updating the visual context during interaction. To enable this capability, we construct large-scale multi-turn exploration data and develop a two-stage post-training recipe based on supervised fine-tuning (SFT) and RL. Because exploration produces continuously changing visual contexts, ThymeV2 maintains an evolving visual memory rather than simply accumulating all observations. The model updates which intermediate visual observations remain active throughout exploration, allowing the visual context to adapt to the current exploration state. To align training with this dynamic process, SFT examples are constructed from the observations available at each turn, while RL optimization operates on memory-aligned exploration trajectories. To improve training stability over long exploration trajectories, we combine outcome and process signals with **dual-granularity policy masking** and bounded local advantage updates.

We further introduce **StratBenchFinal**, a benchmark for challenging visual exploration. It contains 893 examples drawn from nine established visual benchmarks and organized into five difficulty levels. The benchmark focuses heavily on difficult cases: 58% of the examples come from the two hardest levels, including 393 Level-5 examples. All examples are manually reviewed before inclusion in the benchmark. ThymeV2-9B and ThymeV2-27B lead all evaluated open-weight models with six-benchmark averages of **75.96%** and **77.02%**. Under the same harness, ThymeV2-27B surpasses Gemini 3.1 Pro on three benchmarks and matches it on VisualProbe-Hard. These results demonstrate the effectiveness of learned sustained active perception. We will release ThymeV2, our training data, StratBenchFinal, and code. Our main contributions are summarized as follows:

❶ **Model Design.** We introduce **ThymeV2**, a multimodal model for sustained active perception in high-resolution images. Through its evolving visual memory, ThymeV2 updates which observations remain active, maintaining a compact visual context throughout deep multi-turn exploration.

❷ **Data and Post-Training.** We construct **ThymeData**, a large-scale dataset for multi-turn image exploration, and develop a two-stage post-training recipe aligned with the evolving visual context. SFT uses verified exploration and error-recovery examples, while RL combines outcome and process feedback with dual-granularity policy masking and bounded local negative updates to support stable learning over long exploration trajectories.

❸ **Benchmark Construction.** We introduce **StratBenchFinal**, a visual exploration benchmark comprising 893 manually reviewed examples from nine established benchmarks, spanning five difficulty levels and emphasizing challenging image perception. We will release the benchmark, ThymeData, and our model weights, and open-source the training and evaluation code.

❹ **Strong Performance.** Across six image benchmarks, ThymeV2-9B outperforms evaluated open-weight and specialized active-perception baselines on average. On VisualProbe-Hard, it achieves **71.69%**, exceeding the strongest specialized active-perception baseline by 25.46 percentage points. Under the same harness, it trails Gemini 3.1 Pro by 0.95 percentage points.

2 Related Work

Thinking with Images. Recent multimodal models increasingly move beyond passive image understanding toward active visual perception, interleaving reasoning with grounding, cropping, zooming, visual editing, and code execution (OpenAI, 2025; Su et al., 2025; Zheng et al., 2026; Wu et al., 2026a; Zhang et al., 2025b). Related work targets fine-grained and high-resolution perception, where relevant evidence occupies only a small portion of the input (Yang et al., 2025; Liu et al., 2026; Wang et al., 2026a). Complementary distillation methods internalize regional perceptual knowledge without inference-time tool use (Wei et al., 2026; Yuan et al., 2026). However, interactive pipelines such as Mini-o3 and PyVision-Image append new observations to the interaction history, allowing visual context to grow with exploration (Lai et al., 2026; Zhao et al., 2026). ThymeV2 explicitly revises which intermediate observations remain active, maintaining a compact visual memory throughout extended exploration.

Long-Horizon Visual Exploration. Recent work studies how data construction and post-training support deeper visual exploration. DeepEyesV2 and VC-Tooler combine supervised initialization with reinforcement learning to establish and refine adaptive tool use (Hong et al., 2026; Wu et al., 2026b). Mini-o3 masks over-turn trajectories during policy updates (Lai et al., 2026), while PyVision-RL encourages sustained tool use and filters rollouts with broken interactions (Zhao et al., 2026). These trajectory-level treatments do not isolate invalid actions within trajectories that subsequently recover. ThymeV2 makes this distinction through dual-granularity policy masking, preserving valid steps in recovered trajectories while excluding unrecovered trajectories. Relative process feedback further distinguishes exploration quality among trajectories with the same outcome. Both SFT and RL also use the visual context available when each action was generated.

3 ThymeV2

We introduce ThymeV2, a framework that interleaves reasoning with visual exploration for multi-turn multimodal interaction. Figure 2 illustrates the workflow. Instead of using a static visual context, ThymeV2 dynamically acquires and updates visual evidence across interaction turns to answer user queries.

3.1 Active Exploration and Evolving Visual Memory

Given a visual input $\mathcal{V}$ and a user query q, at turn t, the policy π_θ generates a response y_t based on the text history $\mathcal{H}_t$ and the active visual memory $\mathcal{M}_t$:

$$y_t \sim \pi_\theta\left(\cdot \mid q, \mathcal{H}_t, \mathcal{M}_t\right). \tag{1}$$

The response y_t either gives the answer or issues an action a_t to gather more evidence.

For image inputs, the action consists of K_t spatial region crops:

$$a_t^{\text{img}} = \{(i_{t,k}, r_{t,k}, z_{t,k})\}_{k=1}^{K_t}, \tag{2}$$

where $i_{t,k}$ is the image index, $r_{t,k}$ is the region coordinates, and $z_{t,k}$ is the zoom factor. The same tool call can also mark intermediate observations as no longer useful, allowing visual acquisition and memory cleanup within a single interaction.

Executing a_t produces new observations $\mathcal{O}_t$. To prevent the multimodal context from growing linearly with the interaction turns, ThymeV2 updates its active visual memory $\mathcal{M}_t$ by removing stale or redundant visual states $\mathcal{R}_t \subseteq \mathcal{M}_t$ and adding new observations $\mathcal{O}_t$:

$$\mathcal{M}_{t+1} = (\mathcal{M}_t \setminus \mathcal{R}_t) \cup \mathcal{O}_t. \tag{3}$$

Importantly, we only remove visual features from $\mathcal{M}_t$ to reduce visual-context accumulation; the corresponding text logs in $\mathcal{H}_t$ are preserved so the model does not lose its reasoning history. When a memory update alters the active visual states, the visual context is reconstructed. We record these reconstruction points as memory boundaries b_k, which are used to align visual sequences during post-training.

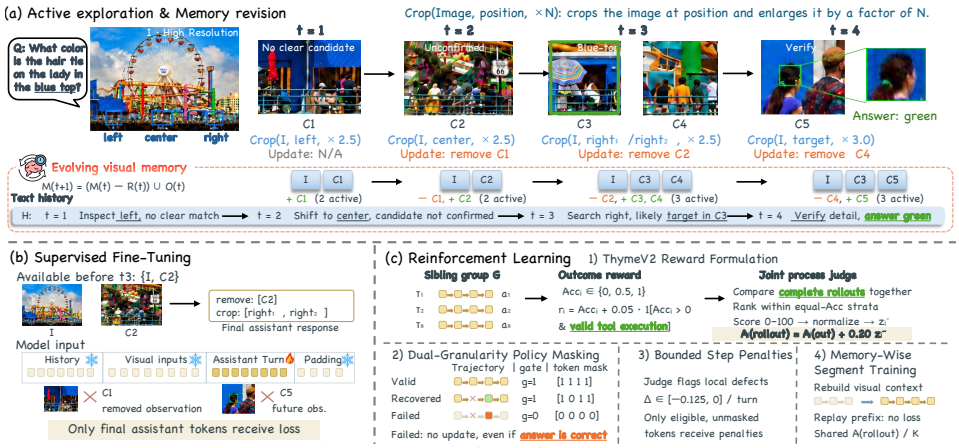


Figure 2: Overview of ThymeV2.

4 THYMEDATA: TRAINING DATA

We construct training data separately for Supervised Fine-Tuning(SFT) and Reinforcement Learning(RL). For SFT, we collect diverse image-based questions from existing visual perception works (Zheng et al., 2026; Lai et al., 2026; Zhang et al., 2025b; Yang et al., 2025; Zhao et al., 2026; Wang et al., 2026b; Liu et al., 2026; Wang et al., 2026a; Wei et al., 2026; Wiedmann et al., 2025) to build high-quality trajectories. The majority of these questions pose challenging perception problems that require active and continuous visual exploration in a dynamic, multi-step environment.

4.1 THYMEDATA-SFT CONSTRUCTION

Our SFT dataset contains 46k high-quality examples, including 34k agentic exploration examples and 12k general visual instruction examples. Among the agentic trajectories, 25.1% involve at least three interaction turns, with the longest extending to 15 turns (details are provided in Appendix A). To guarantee the highest possible data reliability and context consistency across turns, we design and implement a multi-stage trajectory construction and rigorous quality control pipeline:

- ◇ **Step-Level Context Reconstruction:** We generate multi-turn exploration trajectories with the visual exploration tool and active visual memory described in Sec. 3. To train the model on dense decision states, we convert these trajectories into step-level SFT examples. For each assistant step, we reconstruct the multimodal input from the exact state of the active visual memory $\mathcal{M}_t$ at that specific turn. This state-level replay ensures that the training context perfectly matches the inference-time visual context, thereby training the model to consistently make reliable exploration decisions based on currently active visual evidence.
- ◇ **Turn-by-Step Quality Control:** To ensure the reliability of intermediate decisions, we implement a fine-grained, step-level multimodal review during trajectory collection. For every candidate turn, a critic model evaluates the assistant's exploration action within its immediate interaction context against the active visual context, filtering out invalid, hallucinatory, or broken steps. This step-by-step verification yields 28,236 clean exploration steps. To handle label noise where the generated answer disagrees with the source annotations, we employ an ensemble review panel of three expert models followed by human final arbitration, salvaging 4,518 clean, highly complex examples.
- ◇ **Error Recovery and Reflection:** Successful trajectories teach the model how to accumulate visual evidence, but they provide few examples of how to recover from mistakes, which easily propagate in long interactions. We therefore construct 1,101 reflection trajectories to teach the model how to self-correct. For failed trajectories in our pool, we preserve the history leading to the error, remove the incorrect terminal answer, and use expert models to continue the interaction. The model is trained to actively recognize faulty reasoning, gather missing visual evidence, and revise its conclusion to reach the correct final answer.

4.2 THYMEDATA-RL CONSTRUCTION

We construct ThymeData-RL based on empirical difficulty observed under the SFT policy. Starting with 80,165 raw question seeds, we roll out eight independent trajectories per question using our SFT-trained model. We count the correct outcomes using automated string matching or expert model verification. We retain only those questions that achieve between two and seven correct trajectories

out of eight rollouts, discarding trivial and unsolvable tasks. This empirical filter secures 28,787 optimal RL questions that maximize training efficiency (detailed in Appendix A.4).

4.3 StratBenchFinal

Existing visual perception benchmarks often contain a substantial proportion of examples that can be solved without deep visual exploration, making them less effective for evaluating sustained active perception. We therefore construct **StratBenchFinal**, a fully human-verified visual QA benchmark that emphasizes challenging perception problems. We estimate empirical difficulty using the no-think pass@8 performance of Qwen3.5-9B and organize the benchmark into five difficulty levels.

StratBenchFinal contains 893 examples spanning 11 capability categories, with 58% drawn from the two hardest levels, including 393 Level-5 cases. It covers diverse capabilities such as spatial understanding, OCR, visual properties, reasoning, counting, and chart understanding, and includes both multiple-choice and free-form visual QA. Every example is manually verified for question validity, answer correctness, and consistency with the visual evidence; ambiguous or erroneous examples are corrected or removed. Further details on benchmark construction, difficulty stratification, and capability distribution are provided in Appendix B.

5 Training

We optimize the policy in a progressive two-stage pipeline: SFT establishes basic exploration and tool-use capabilities from turn-level examples, and RL aligns the sequential decisions with both outcome correctness and fine-grained process-based search logic.

5.1 Supervised Fine-Tuning

SFT uses the exploration prefixes and visual contexts prepared in Sec. 4.1. For each tool-using example, we supervise only the final assistant response, including any tool call, while preceding messages and visual observations provide conditioning context. General visual instruction examples follow standard assistant-response supervision.

For a global training batch $\mathcal{B}$, let $\mathcal{S}_\mathcal{B}$ denote the set of supervised token positions across its examples. We minimize the token-normalized negative log-likelihood:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\mathbb{E}_\mathcal{B}\left[\frac{1}{|\mathcal{S}_\mathcal{B}|}\sum_{(n,j)\in\mathcal{S}_\mathcal{B}} \log \pi_\theta(y_{n,j} \mid x_{n,j})\right], \tag{4}$$

where $y_{n,j}$ is the token at position j in example n, and $x_{n,j}$ contains its preceding text and available visual observations. Conditioning tokens and padding are excluded from $\mathcal{S}_\mathcal{B}$.

5.2 Reinforcement Learning

We propose a multi-turn RL recipe that combines outcome and process feedback, dual-granularity policy masking, and bounded local advantage corrections. Memory-aligned segments preserve the visual context used during generation.

Training Algorithm We adopt an on-policy variant of Group Relative Policy Optimization (GRPO). For each problem, the algorithm samples a group of interaction trajectories $\{\tau_1, \tau_2, \ldots, \tau_G\} \sim \pi_\theta(\cdot \mid I, Q)$, where each trajectory τ_i represents a multi-turn interaction:

$$\tau_i = \{(I, Q); ([T_{i,0}, O_{i,0}, S_{i,0}], \ldots, [T_{i,t_i}, a_i])\}. \tag{5}$$

Here, $T_{i,k}$, $O_{i,k}$, and $S_{i,k}$ represent thoughts, observations, and visual states at turn k, and a_i is the final answer. The policy updates by maximizing a token-averaged objective over a global rollout batch $\mathcal{B}$:

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_\mathcal{B}\left[\frac{1}{|\mathcal{S}_\mathcal{B}|}\sum_{(i,j)\in\mathcal{S}_\mathcal{B}} c_{i,j}\, \mathcal{C}_{\mathrm{PPO}}\left(\frac{\pi_\theta(y_{i,j} \mid x_{i,j})}{\pi_{\mathrm{old}}(y_{i,j} \mid x_{i,j})}, \widetilde{A}_{i,j}\right)\right], \tag{6}$$

where $\mathcal{S}_\mathcal{B}$ contains retained assistant target tokens after trajectory and span masking, $\mathcal{C}_{\mathrm{PPO}}$ is the clipped policy surrogate, and $c_{i,j}$ is the rollout importance correction. The weighted advantage $\widetilde{A}_{i,j}$ is defined in Eq. 14. Each token is conditioned on its corresponding memory context $x_{i,j}$; replayed prefixes, tool observations, and padding are excluded from $\mathcal{S}_\mathcal{B}$. Details and optimization settings are given in Appendix C.3.

ThymeV2 Reward Formulation We incorporate model-based process feedback alongside the sparse outcome signal. Multi-turn visual search progressively gathers evidence through cropping and zooming: a crop that does not directly support the final answer may still eliminate candidate regions or guide subsequent inspections. The value of such exploration depends on how observations accumulate and inform later decisions, complicating credit assessment at individual turns. In our experiments, turn-level process assessment undervalues these intermediate contributions, reducing exploration depth and performance on difficult tasks (Sec. 5). We therefore assess exploration quality at the trajectory level, comparing complete processes among trajectories.

Even when trajectories achieve the same final-answer score, they may differ in how effectively they gather and use visual evidence. For relative process comparison, we partition the sibling group $\mathcal{G}$ into outcome-matched strata $\mathcal{G}^{(c)} = \{j \in \mathcal{G} \mid \mathrm{Acc}(\tau_j) = c\}$, where $\mathrm{Acc}(\tau_j) \in \{0, 0.5, 1.0\}$ denotes an incorrect, partially correct or correct final answer, respectively. For the outcome signal, we augment the correctness of the final answer with a fixed tool-use bonus $\alpha = 0.05$, awarded when the answer receives positive credit and the trajectory contains a valid tool execution.

$$r_i = \mathrm{Acc}(\tau_i) + \alpha \cdot \mathbb{I}\big[\,\mathrm{Acc}(\tau_i) > 0 \wedge \mathrm{ValidTool}(\tau_i)\big], \tag{7}$$

where $\mathrm{ValidTool}(\tau_i)$ is a binary indicator of whether τ_i contains at least one clean tool execution.

The outcome advantage for eligible trajectories is computed via GRPO over the sibling group $\mathcal{G}$:

$$A_i^{\mathrm{out}} = \frac{r_i - \mu_{\mathcal{G}}}{\sigma_{\mathcal{G}} + \epsilon}, \tag{8}$$

where $\mu_{\mathcal{G}}$ and $\sigma_{\mathcal{G}}$ denote the group reward mean and standard deviation, with ineligible trajectories contributing zero rewards to the baseline.

For process feedback, the judge assigns each trajectory a global exploration score on a 0–100 scale, rescaled to $s_i \in [0, 1]$. Within each stratum matched with the outcome $\mathcal{G}^{(c)}$, we compute the mean μ_c and the standard deviation σ_c, applying a standard deviation floor $\sigma_0 = 0.10$ to limit the amplification of small score differences:

$$z_i = \frac{s_i - \mu_c}{\max(\sigma_c, \sigma_0)}. \tag{9}$$

The standardized scores are then projected to $[-1, 1]$ while preserving zero mean within each stratum, yielding $\bar{z}_i$.For eligible trajectories, the rollout-level advantage is:

$$A_i^{\mathrm{rollout}} = A_i^{\mathrm{out}} + \lambda \cdot \bar{z}_i, \tag{10}$$

where $\lambda = 0.20$ is the process feedback coefficient.

Dual-Granularity Policy Masking During long-horizon interactions, occasional local tool-call or parsing failures are statistically expected. Complete-rollout filtering can discard recoverable long trajectories, while span-only masking is too permissive: some trajectories may fail to recover from broken tool interactions yet still reach correct answers through parametric priors. Masking only the failed span would still reinforce the remaining reasoning and final answer despite the unrecovered visual interaction.

We partition trajectories into three disjoint sets: $\mathcal{T}_{\mathrm{valid}}$ (no tool failures), $\mathcal{T}_{\mathrm{recovered}}$ (tool failures followed by successful recovery), and $\mathcal{T}_{\mathrm{failed}}$ (unrecovered tool failures, regardless of final-answer correctness). We define a trajectory-level eligibility gate g_i and a token-level span mask $m_{i,t,j}$ as:

$$g_i = \mathbb{I}\big(\tau_i \in \mathcal{T}_{\mathrm{valid}} \cup \mathcal{T}_{\mathrm{recovered}}\big), \quad m_{i,t,j} = g_i \cdot m_{i,t,j}^{\mathrm{span}}, \tag{11}$$

where $m_{i,t,j}^{\mathrm{span}} \in \{0, 1\}$ is a step-level mask that zeroes out the gradients of specific token spans containing invalid tool calls. Under this dual-granularity scheme, trajectories in $\mathcal{T}_{\mathrm{failed}}$ are marked as policy-ineligible ($g_i = 0$), preventing long-turn unrecovered or incoherent pathways from updating the policy. Meanwhile, successfully recovered trajectories are retained ($g_i = 1$) to prevent long-trajectory sample starvation, while their failed tool-call spans are masked ($m_{i,t,j} = 0$) to block gradient updates on invalid actions. Ineligible trajectories are still retained in the sibling group $\mathcal{G}$ to serve as negative anchor samples, but their own advantages are masked to zero.

Bounded Step Penalties To suppress distinct degenerate patterns during rollout generation, we apply step-bounded local penalties. Unlike the global process reward, this penalty signal operates as a local turn-level feedback, evaluated and output by the same LLM judge. Given an automated severity score $p_{i,t} \in \{0, 25, 50, 75, 100\}$ for turn t, we compute the step penalty and distribute it uniformly across the active tokens:

$$\delta_{i,t,j} = \frac{\Delta_{i,t}}{N_{i,t}}, \quad \Delta_{i,t} = R_0 \cdot \eta \cdot \frac{p_{i,t}}{100}, \tag{12}$$

where $R_0 = -0.05$ is the base penalty, $\eta = 2.5$ is the scaling coefficient, and $N_{i,t}$ is the number of trainable tokens in turn t. The total penalty $\Delta_{i,t}$ is capped at -0.125 per turn. Crucially, if a turn is already masked by $m^{\text{span}}_{i,t,j} = 0$, or if the trajectory is ineligible ($g_i = 0$), the local penalty $\delta_{i,t,j}$ is zeroed out to prevent double punishment.

Memory-Wise Segment Training Because active visual memory can replace or remove observations, an exploration trajectory need not form a single append-only multimodal sequence. We therefore organize training around the context reconstruction boundaries defined in Sec. 3.1. For trajectory τ_i, let $0 = b_0 < b_1 < \cdots < b_{K_i} = T$ denote these boundaries. The generated trajectory is partitioned into memory-aligned segments:

$$\tau_i = \tau_i^{(1)} \oplus \tau_i^{(2)} \oplus \cdots \oplus \tau_i^{(K_i)} = \bigoplus_{k=1}^{K_i} \tau_i^{(k)}, \tag{13}$$

where each segment $\tau_i^{(k)} = \tau_{i,b_{k-1}:b_k}$ is processed with the visual memory and interaction history available at its start. Only newly generated assistant tokens are supervised; replayed history supplies conditioning context without introducing repeated loss terms.

The rollout-level advantage is shared across all segments, while masks and penalties act on their corresponding tokens. We moderate the contribution of trajectories spanning multiple reconstructed contexts. For token j in turn $t(j)$, the resulting advantage is:

$$\widetilde{A}_{i,j} = \frac{1}{K_i} m_{i,t(j),j} \left(A_i^{\text{rollout}} + \delta_{i,t(j),j} \right). \tag{14}$$

The weighted token losses are pooled across segments using the batch-level denominator in Eq. 6. This separates context reconstruction from loss aggregation: segments preserve visual evidence available during generation, while rollout-level feedback remains shared across the process.

6 Experiments

We evaluate ThymeV2 across challenging image perception tasks and analyze how deeper interaction, evolving visual memory, and long-horizon post-training affect the learned exploration policy. Beyond final accuracy, we track interaction depth, active visual context, visual computation, and policy validity to characterize sustained active perception.

6.1 Experimental Setup

Benchmarks and Training Details. We evaluate on V*Bench (Wu & Xie, 2024), HRBench-8K (Wang et al., 2025b), MME-RealWorld-Lite (Zhang et al., 2025c), VisualProbe-Hard (Lai et al., 2026), ZoomBench (Wei et al., 2026), and StratBenchFinal (Sec. 4.3). We report the unweighted average across these six benchmarks as Avg. in Table 1. RL training is conducted on 32 NVIDIA H20 GPUs with a budget of 300 updates. We use a global batch size of 128 prompts and sample eight rollouts per prompt Detailed hyperparameters are provided in the Appendix.

Baselines and Protocol. We compare against frontier proprietary models including o4-mini, o3-pro, Seed2.0 Pro (Seed, 2026), GPT-5.6-sol (OpenAI, 2026), and Gemini 3.1 Pro (DeepMind, 2026); open-weight MLLMs including Qwen2.5 (Bai et al., 2025b), Qwen3-VL (Bai et al., 2025a), Kimi K3 (Team et al., 2026), and Qwen3.5 (Team, 2026a); and recent active-perception and agentic reasoning models with publicly released checkpoints. These include DeepEyes (Zheng et al., 2026), DeepEyesV2 (Hong et al., 2026), Thyme (Zhang et al., 2025b), Mini-o3 (Lai et al., 2026), PyVision (Zhao et al., 2026), Vision-OPD (Yuan et al., 2026), SenseNova-MARS (Chng et al., 2025),

Table 1: **Main image results on challenging perception benchmarks.** The best and second-best results are bolded and underlined, respectively. Matching superscripts link methods to their base models. Brown arrows indicate changes from the corresponding base models in percentage points. Benchmarks: V* (V*Bench), HR8K (HRBench-8K), MME-RL (MME-RealWorld-Lite), V-Hard (VisualProbe-Hard), Zoom (ZoomBench), Strat (StratBenchFinal).

Model	V*	HR8K	MME-RL	V-Hard	Zoom	Strat	Avg.
Frontier Proprietary Models							
o4-mini	68.59	70.00	53.05	24.53	48.28	37.63	50.34
o3-pro	73.82	71.50	57.43	26.42	53.02	39.98	53.69
Seed2.0 Pro	90.05	80.88	58.99	22.64	60.47	45.80	59.80
GPT-5.6-sol	84.29	69.25	62.27	51.89	61.30	52.41	63.57
Gemini 3.1 Pro	86.39	<u>87.62</u>	59.77	44.34	58.93	48.49	64.26
Gemini 3.1 Pro + Tools	<u>97.38</u>	**93.25**	64.15	**72.64**	73.14	**68.31**	**78.14**
Open-Weight Models							
Qwen2.5-VL-7B-Instruct†	76.40	65.30	44.10	23.90	42.50	33.82	47.67
Qwen3-VL-8B-Instruct‡	84.82	75.25	46.53	34.90	42.96	38.63	53.85
Kimi K3	89.01	82.12	58.83	34.91	58.70	46.14	61.62
Qwen3.5-9B§	87.96	83.25	58.78	36.79	57.99	45.58	61.73
Qwen3.5-27B¶	89.01	83.38	59.77	47.17	60.24	46.14	64.29
Active Perception Models							
PixelReasoner†	79.06 ↑2.66	65.38 ↑0.08	46.80 ↑2.70	17.92 ↓5.98	38.70 ↓3.80	26.99 ↓6.83	45.81 ↓1.86
PyVision†	79.06 ↑2.66	71.62 ↑6.32	52.94 ↑8.84	35.85 ↑11.95	46.15 ↑3.65	26.98 ↓6.84	52.10 ↑4.43
DeepEyesV2†	83.77 ↑7.37	72.50 ↑7.20	46.48 ↑2.38	29.25 ↑5.35	44.73 ↑2.23	39.19 ↑5.37	52.65 ↑4.98
DeepEyes†	81.15 ↑4.75	69.50 ↑4.20	52.84 ↑8.74	36.79 ↑12.89	46.15 ↑3.65	38.19 ↑4.37	54.10 ↑6.43
Mini-o3†	84.29 ↑7.89	71.62 ↑6.32	49.92 ↑5.82	32.08 ↑8.18	46.86 ↑4.36	41.43 ↑7.61	54.37 ↑6.70
Chain-of-Focus†	82.72 ↑6.32	72.00 ↑6.70	50.86 ↑6.76	38.68 ↑14.78	44.85 ↑2.35	38.19 ↑4.37	54.55 ↑6.88
Thyme†	80.63 ↑4.23	73.12 ↑7.82	54.09 ↑9.99	39.62 ↑15.72	45.09 ↑2.59	38.86 ↑5.04	55.24 ↑7.57
SenseNova-MARS‡	89.53 ↑4.71	78.12 ↑2.87	53.41 ↑6.88	44.34 ↑9.44	50.18 ↑7.22	46.70 ↑8.07	60.38 ↑6.53
Zooming without Zooming‡	90.58 ↑5.76	82.25 ↑7.00	59.72 ↑13.19	43.40 ↑8.50	54.67 ↑11.71	47.26 ↑8.63	62.98 ↑9.13
Vision-OPD§	92.67 ↑4.71	84.87 ↑1.62	62.22 ↑3.44	46.23 ↑9.44	64.85 ↑6.86	45.91 ↑0.33	66.12 ↑4.39
ThymeV2-9B§	96.33 ↑8.37	86.38 ↑3.13	<u>64.20</u> ↑5.42	<u>71.69</u> ↑34.90	<u>74.67</u> ↑16.68	62.48 ↑16.90	75.96 ↑14.23
ThymeV2-27B¶	**98.43** ↑9.42	86.13 ↑2.75	**65.55** ↑5.78	**72.64** ↑25.47	**76.33** ↑16.09	<u>63.04</u> ↑16.90	<u>77.02</u> ↑12.73

Chain-of-Focus (Zhang et al., 2025a), PixelReasoner (Su et al., 2026), and Zooming without Zooming (Wei et al., 2026). Released baselines follow their native inference protocols when available. We additionally equip Gemini 3.1 Pro, which achieves the highest average score among the proprietary models evaluated under their native protocols, with the same interaction harness as ThymeV2. We denote this setting as *Gemini 3.1 Pro + Tools*. Full configurations are provided in the AppendixE.

6.2 Main Results

Table 1 shows that ThymeV2-9B achieves **75.96%** on average across six benchmarks, outperforming all evaluated open-weight models and specialized active-perception baselines on every benchmark. It surpasses its Qwen3.5-9B backbone and Vision-OPD, the strongest evaluated active-perception baseline, by 14.23 and 9.84 percentage points on average, respectively. On VisualProbe-Hard, it raises accuracy from its backbone's 36.79% to **71.69%**, a gain of **34.90 points**, and exceeds Vision-OPD by 25.46 points. On StratBenchFinal, it achieves 62.48%, surpassing the strongest evaluated active-perception baseline by 15.22 points. ThymeV2-9B also outperforms all evaluated proprietary models in average score under their native inference protocols, exceeding GPT-5.6-sol, Seed2.0 Pro, and Gemini 3.1 Pro by 12.39, 16.16, and 11.70 points, respectively. Notably, under the same interaction harness, our 9B model remains highly competitive with Gemini 3.1 Pro + Tools, trailing by only **2.18 points** on average and 0.95 points on VisualProbe-Hard (71.69% versus 72.64%), while achieving higher scores on MME-RealWorld-Lite and ZoomBench.

6.3 Long-Horizon Post-Training

Table 2 compares the full recipe with three alternatives. Pure Outcome uses outcome rewards without process feedback or tool-error masking; Rollout Mask excludes entire trajectories containing recorded tool errors; Span Mask Only masks local error spans without recovery-aware trajectory gating. We report update 300 for Full ThymeV2 and Rollout Mask, and update 140 for Span Mask Only following tool-use breakdown around update 130. Error rates are aggregated over the 30 consecutive updates ending at each reported checkpoint. Full achieves **75.96%** Avg., versus 72.81% and 73.63%, respectively, with a tool-error trajectory rate of **2.36%**. Pure Out-

come collapses during rollout generation: between updates 128 and 134, mean valid tool calls fall from 1.75 to 0.03 and reward from 0.524 to 0.152. Whole-trajectory filtering discards useful exploration alongside local errors and can disproportionately remove longer trajectories, which encounter more opportunities for tool failures. Our recipe preserves useful supervision by retaining recovered trajectories, masking invalid spans, and excluding unrecovered failures. Appendix C.1 details the associated tool-use, reward, and unanswered-response dynamics. We examine the role and granularity of process feedback in Appendix C.2. Across matched updates, rollout reward standard deviation is **0.0387** with trajectory-level feedback versus 0.0439 without it. Additional turn-level credit assessment yields no consistent validation gains: at update 160, Full leads by **6.13** and **2.75** points on VisualProbe-Hard and StratBenchFinal, respectively. One possible explanation is that intermediate observations can guide later exploration without directly supporting the final answer, making their value difficult to assess turn by turn. This motivates trajectory-level assessment with selective local penalties. We also ablate the SFT data construction (Sec. 4.1) on Qwen3.5-VL, with results in Appendix C.3.

Table 2: **RL recipe ablations.** Avg: mean performance across six benchmarks. Err: percentage of trajectories with recorded tool errors over the 30 updates ending at each checkpoint. Pure Outcome exhibits tool-use collapse.

Recipe	Avg ↑	Err ↓
Full ThymeV2	**75.96**	**2.36**
Rollout Mask	72.81	12.83
Span Mask Only	73.63	31.71
Pure Outcome	*Tool-use collapse*	

6.4 Sustained Active Perception

Exploration Length. We use sustained exploration to mean retaining the ability to continue inspecting visual evidence when needed, rather than requiring every question to follow a long trajectory. Figure 3(a) shows this adaptive behavior across benchmarks and individual questions. Under a shared 15-turn budget, average interaction lengths range from 2.09 turns on ZoomBench to 2.98 turns on VisualProbe-Hard. Within HRBench-8K, the average remains 2.22 turns, while individual trajectories extend to 11 turns; MME-RealWorld-Lite and StratBenchFinal similarly reach 10 despite averages below 2.5. This gap between average and observed maximum characterizes adaptive interaction depth: ThymeV2 often terminates after short exploration, while retaining the capacity for substantially longer visual interaction on a subset of instances.

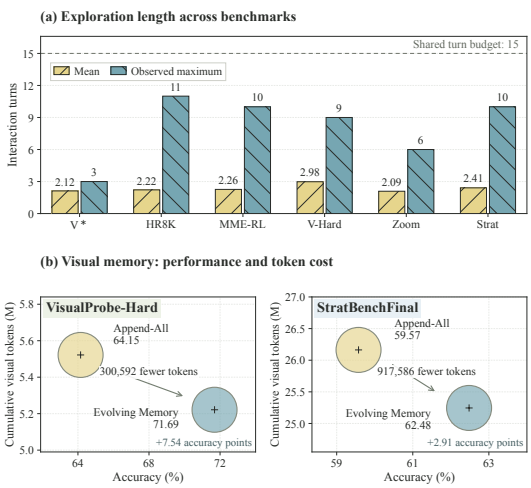


Figure 3: **Sustained active perception.** (a) Mean and maximum turns, including the final response; the dashed line indicates the 15-turn budget. (b) Accuracy and cumulative visual tokens under this budget, showing token savings and accuracy gains in percentage points. M denotes 10^6 tokens.

Effect of Evolving Visual Memory. We compare Evolving Memory with an inference-time Append-All variant of the same checkpoint, which disables visual-state removal and retains all acquired observations in subsequent model inputs At the 15-turn budget, Figure 3(b) shows that Evolving Memory improves VisualProbe-Hard accuracy from 64.15% to 71.69%, a gain of 7.54 percentage points, while reducing cumulative visual input by 300,592 tokens. On StratBenchFinal, accuracy increases from 59.57% to 62.48%, accompanied by a reduction of 917,586 visual tokens. These results support updating the active visual context throughout exploration, enabling the model to continue inspecting visual evidence with less repeated visual input.

7 Conclusion

We introduced ThymeV2 for sustained active perception in images. Combining evolving visual memory, multi-turn exploration, and long-horizon post-training, ThymeV2 supports extended exploration with compact visual context. Experiments demonstrate strong performance, stable training, and reduced visual token consumption. We will release our models, data, benchmark, and code.

AI USE STATEMENT

Generative AI tools were used to assist with code implementation, literature search, experimental discussion, and manuscript polishing. The research problem, methodology, and conclusions were developed by the authors. All AI-assisted code and text were reviewed and verified by the authors. Generative AI was not used to create datasets or determine scientific claims. The authors take responsibility for all final content.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

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, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Yong Xien Chng, Tao Hu, Wenwen Tong, Xueheng Li, Jiandong Chen, Haojia Yu, Jiefan Lu, Hewei Guo, Hanming Deng, Chengjun Xie, et al. Sensenova-mars: Empowering multimodal agentic reasoning and search via reinforcement learning. *arXiv preprint arXiv:2512.24330*, 2025.

Google DeepMind. Gemini 3.1 pro model card. Google DeepMind Model Card, feb 2026. URL https://deepmind.google/models/model-cards/gemini-3-1-pro/.

James J Gibson. The ecological approach to the visual perception of pictures. *Leonardo*, 11(3): 227–235, 1978.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, and Guohai Xu. Deepeyesv2: Toward agentic multimodal model. In *International Conference on Learning Representations*, volume 2026, pp. 114851–114872, 2026.

Xin Lai, Junyi Li, Wei Li, Tao Liu, Tianjian Li, and Hengshuang Zhao. Mini-o3: Scaling up reasoning patterns and interaction turns for visual search. In *International Conference on Learning Representations*, volume 2026, pp. 76722–76746, 2026.

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

Ruixun Liu, Bowen Fu, Jiayi Song, Kaiyu Li, Wanchen Li, Lanxuan Xue, Hui Qiao, Weizhan Zhang, Deyu Meng, and Xiangyong Cao. Zoomearth: Active perception for ultra-high-resolution geospatial vision-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 34877–34888, 2026.

OpenAI. Thinking with images. OpenAI, April 2025. URL https://openai.com/index/thinking-with-images/.

OpenAI. Gpt-5.6 system card. https://deploymentsafety.openai.com/gpt-5-6, 2026. Published July 9, 2026.

Bytedance Seed. Seed2. 0 model card: Towards intelligence frontier for real-world complexity. *arXiv preprint arXiv:2607.00248*, 2026.

Alex Su, Haozhe Wang, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel reasoner: Incentivizing pixel space reasoning via curiosity-driven reinforcement learning. *Advances in Neural Information Processing Systems*, 38:8222–8251, 2026.

Zhaochen Su, Linjie Li, Mingyang Song, Yunzhuo Hao, Zhengyuan Yang, Jun Zhang, Guanjie Chen, Jiawei Gu, Juntao Li, Xiaoye Qu, et al. Openthinkimg: Learning to think with images via visual tool reinforcement learning. *arXiv preprint arXiv:2505.08617*, 2025.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL `https://qwen.ai/blog?id=qwen3.5`.

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

Fengxiang Wang, Mingshuo Chen, Yueying Li, Yajie Yang, Yifan Zhang, Long Lan, Xue Yang, Hongda Sun, Yulin Wang, Di Wang, et al. Geoeyes: On-demand visual focusing for evidence-grounded understanding of ultra-high-resolution remote sensing imagery. *arXiv preprint arXiv:2602.14201*, 2026a.

Fengxiang Wang, Mingshuo Chen, Yueying Li, Yajie Yang, Yuhao Zhou, Di Wang, Yifan Zhang, Haoyu Wang, Haiyan Zhao, Hongda Sun, et al. Text before vision: Staged knowledge injection matters for agentic rlvr in ultra-high-resolution remote sensing understanding. *arXiv preprint arXiv:2602.14225*, 2026b.

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

Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, Wei Yu, and Dacheng Tao. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 7907–7915, 2025b.

Lai Wei, Liangbo He, Jun Lan, Lingzhong Dong, Yutong Cai, Siyuan Li, Huijia Zhu, Weiqiang Wang, Linghe Kong, Yue Wang, et al. Zooming without zooming: Region-to-image distillation for fine-grained multimodal perception. *arXiv preprint arXiv:2602.11858*, 2026.

Luis Wiedmann, Orr Zohar, Amir Mahla, Xiaohan Wang, Rui Li, Thibaud Frere, Leandro von Werra, Aritra Roy Gosthipaty, and Andrés Marafioti. Finevision: Open data is all you need. *arXiv preprint arXiv:2510.17269*, 2025.

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

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal llms. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13084–13094. IEEE, 2024.

Yizheng Wu, Jiashen Hua, Bing Deng, and Jieping Ye. Vc-tooler: Learning compositional and adaptive visual tool use. *arXiv preprint arXiv:2608.02217*, 2026b.

Wenhao Yang, Yu Xia, Jinlong Huang, Shiyin Lu, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, Yuanyu Wan, and Lijun Zhang. Deep but reliable: Advancing multi-turn reasoning for thinking with images. *arXiv preprint arXiv:2512.17306*, 2025.

Qianhao Yuan, Jie Lou, Xing Yu, Hongyu Lin, Le Sun, Xianpei Han, and Yaojie Lu. Vision-opd: Learning to see fine details for multimodal llms via on-policy self-distillation. *arXiv preprint arXiv:2605.18740*, 2026.

Xiangyu Zeng, Zhiqiu Zhang, Yuhan Zhu, Xinhao Li, Zikang Wang, Changlian Ma, Qingyu Zhang, Zizheng Huang, Kun Ouyang, Tianxiang Jiang, et al. Video-o3: Native interleaved clue seeking for long video multi-hop reasoning. *arXiv preprint arXiv:2601.23224*, 2026.

Xintong Zhang, Zhi Gao, Bofei Zhang, Pengxiang Li, Xiaowen Zhang, Yang Liu, Tao Yuan, Yuwei Wu, Yunde Jia, Song-Chun Zhu, et al. Chain-of-focus: Adaptive visual search and zooming for multimodal reasoning via rl. *arXiv e-prints*, pp. arXiv–2505, 2025a.

Yi-Fan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, Haonan Fan, Kaibing Chen, Jiankang Chen, Haojie Ding, Kaiyu Tang, Zhang Zhang, Liang Wang, Fan Yang, Tingting Gao, and Guorui Zhou. Thyme: Think beyond images, 2025b. URL https://arxiv.org/abs/2508.11630.

YiFan Zhang, Huanyu Zhang, Haochen Tian, Chaoyou Fu, Shuangqing Zhang, Junfei Wu, Feng Li, Kun Wang, Qingsong Wen, Zhang Zhang, et al. Mme-realworld: Could your multimodal llm challenge high-resolution real-world scenarios that are difficult for humans? In *International Conference on Learning Representations*, volume 2025, pp. 89655–89701, 2025c.

Shitian Zhao, Shaoheng Lin, Ming Li, Haoquan Zhang, Wenshuo Peng, Kaipeng Zhang, and Chen Wei. Pyvision-rl: Forging open agentic vision models via rl. *arXiv preprint arXiv:2602.20739*, 2026.

Ziwei Zheng, Minghao Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, and Chao Shen. Deepeyes: Incentivizing" thinking with images" via reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 126775–126798, 2026.

A Detailed Construction of ThymeData

This section provides the implementation details of the ThymeData pipeline introduced in Sec. 4. We focus on how trajectories are generated, converted into turn-level supervision, reviewed, and reused for reflection, as well as how the RL prompt pool is selected from on-policy rollouts.

A.1 Step-Level Context Reconstruction

Trajectory generation and selection. For each source question, we sample multiple multi-turn exploration trajectories with Gemini-3-Pro-preview using the same visual exploration interface as ThymeV2. Each rollout contains the complete sequence of assistant responses, tool calls, tool observations, and visual-memory updates. After a rollout terminates, its final answer is evaluated against the reference answer. When a correct rollout is available, we retain one successful trajectory for the main SFT construction. Unsuccessful trajectories are stored separately and later used for reflection and annotation review.

Prefix–target expansion. A retained trajectory is not used as a single sequence-level SFT example. Instead, we traverse its assistant turns and construct one training example for each retained assistant response. For a trajectory with assistant turns $y_1, \ldots, y_T$, the example associated with turn t contains the longest interaction prefix ending immediately before y_t as its input, while y_t itself is the supervision target. Therefore, previous assistant responses and tool interactions are visible as context but receive no loss in that example.

Across the full set of prefixes extracted from the same trajectory, all retained assistant turns are eventually supervised once as targets. This construction provides direct supervision not only for the terminal answer but also for intermediate exploration decisions.

Replaying the active visual context. The textual prefix alone is insufficient because the visual context changes throughout exploration. We therefore replay the visual-memory state associated with every retained turn. A newly generated crop becomes visible only after the tool call that produced it. Likewise, once an observation is removed from the active visual memory, its visual content is excluded from subsequent training inputs even though the historical interaction remains.

Consequently, two prefixes from the same trajectory may share the same textual history while exposing different sets of active visual observations. This reconstruction ensures that each SFT target is conditioned on the same multimodal state in which it was originally generated.

A.2 Turn-by-Step Quality Control

Quality control is applied before prefix–target expansion. We first use Gemini-3.1-P to review candidate assistant steps together with the interaction prefix and visual evidence available at that turn. The reviewer checks whether each step is supported by the evidence and whether the subsequent interaction remains usable. Trajectories with unreliable intermediate behavior are excluded rather than accepted solely because their final answers match the reference.

A second review stage handles disagreements between the generated solution and the source annotation. These cases are independently examined by Qwen3.5-122B-A10B, Kimi-K2.5, and Seed2.0-pro. Their judgments are aggregated, and examples that remain ambiguous are sent to human annotators for final inspection. This stage allows visually supported trajectories to be recovered when the original annotation is noisy or ambiguous. Only after these reviews do we perform the prefix–target expansion described above. Thus, quality control operates on the original interaction trajectory, while SFT examples are constructed from the verified trajectory afterward.

A.3 Error Recovery and Reflection

Reflection examples are generated from trajectories that fail to reach a correct terminal answer but contain a usable interaction prefix. We preserve the interaction history and active visual state immediately before the erroneous terminal response and remove the failed ending. Gemini-3.1-P is then asked to continue from this exact state.

The continuation has access to the previous reasoning and currently active visual evidence and may invoke additional visual exploration tools. It can therefore revisit an earlier hypothesis, acquire new observations, update the active visual memory, and continue until reaching a correct answer. We retain only successful continuations.

The verified continuation is stitched back to the preserved prefix to form one complete recovery trajectory. Importantly, this trajectory then passes through the same prefix–target expansion as ordinary successful trajectories. As a result, supervision is applied not only to the corrected final response but also to the intermediate turns that perform additional exploration and revise the earlier interpretation.

A.4 THYMEDATA-RL CONSTRUCTION

The RL data pipeline starts from the raw question pool rather than from the SFT trajectories themselves. We first evaluate candidate questions with the SFT-trained Qwen3.5-9B policy. For each question, we independently sample eight complete visual-interaction rollouts.

Final-answer verification follows a two-stage procedure. We first apply task-specific rule-based answer matching whenever the target format supports reliable automatic comparison. Cases that cannot be resolved by the rule-based evaluator are passed to Qwen3.5-122B-A10B for model-based verification. The result is a pass count $n_{\text{pass}} \in \{0, \dots, 8\}$ for every candidate question.

We retain questions satisfying

$$2 \leq n_{\text{pass}} \leq 7, \tag{15}$$

which removes questions for which the SFT policy produces either no successful behavior or nearly deterministic success. The retained questions therefore provide sibling rollouts with meaningful outcome variation for subsequent relative policy optimization.

The final RL data store contains the original question, visual input, and reference answer rather than the eight trajectories used during difficulty estimation. Those rollouts are used only for filtering. During RL training, fresh trajectories are generated on-policy by the current model, so the exploration behavior associated with each question changes together with the policy.

B DETAILS OF STRATBENCHFINAL

This section provides additional details on the construction and composition of StratBenchFinal. We first build a 1,000-example candidate set from nine existing visual benchmarks, stratified according to empirical model difficulty. The candidate set is then manually verified and quality controlled, resulting in 893 examples in the final benchmark.

Difficulty Stratification. We estimate question difficulty using Qwen3.5-9B under the no-think setting. For each candidate question, we independently sample eight responses and let n_{correct} denote the number of correct answers among the eight attempts. We define five model-relative difficulty levels as

$$\begin{aligned} &\text{L1}: \quad n_{\text{correct}} = 8, \\ &\text{L2}: \quad n_{\text{correct}} \in \{6, 7\}, \\ &\text{L3}: \quad n_{\text{correct}} \in \{2, 3, 4, 5\}, \\ &\text{L4}: \quad n_{\text{correct}} = 1, \\ &\text{L5}: \quad n_{\text{correct}} = 0. \end{aligned} \tag{16}$$

These levels reflect difficulty relative to the reference model rather than absolute human difficulty. The initial candidate set contains 125 examples in each of Levels 1–4 and 500 examples in Level 5. After verification and filtering, the final benchmark contains 125/125/125/125/393 examples across Levels 1–5, respectively. Consequently, 58% of StratBenchFinal belongs to the two hardest levels.

Capability Distribution. StratBenchFinal spans 11 visual capability categories. The final benchmark contains 240 spatial, 166 OCR, 162 property, 143 reasoning, 82 counting, 40 action, 31 chart, 15 relation, 6 scene, 5 knowledge, and 3 comparison examples. The distribution is not forced to be uniform; instead, we preserve broad capability coverage while retaining the challenging composition of the curated benchmark.

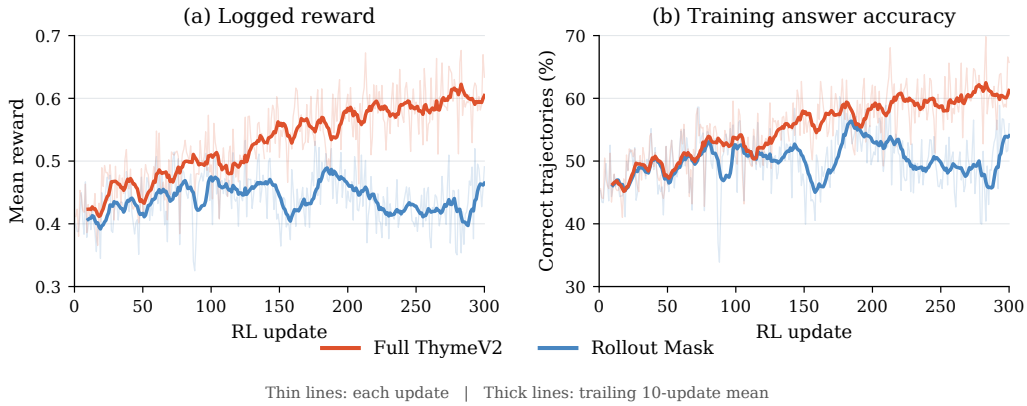


Figure 4: **Training dynamics of Full ThymeV2 and Rollout Mask.** Thin lines show individual updates; thick lines show trailing 10-update means. Left: logged reward. Right: answer accuracy over generated trajectories before policy filtering. The full recipe continues improving, whereas Rollout Mask shows limited sustained progress.

Table 3: **Diagnostics during tool-use degeneration.** Valid Calls and Reward are logged batch means. Unanswered denotes the percentage of generated trajectories masked by the unanswered-response filter. It measures this filtering operation, rather than the overall prevalence of missing answers.

Recipe	Update	Valid Calls	Reward	Unanswered (%)
Pure Outcome	128	1.75	0.5240	1.46
Pure Outcome	134	0.03	0.1519	57.81
Span Mask Only	110	2.08	0.4878	1.27
Span Mask Only	130	0.81	0.2985	40.72

Human Verification. Every example in StratBenchFinal is manually verified before inclusion in the final release. Annotators inspect the visual input together with the question, reference answer, and answer options when applicable. Verification focuses on whether the question is valid and answerable from the visual input, whether the reference answer is correct, and whether it is supported by the available visual evidence. Ambiguous questions, incorrect answers, inconsistent options, or examples without sufficient visual evidence are corrected when they can be reliably repaired and otherwise removed. This verification and filtering process produces the final set of 893 examples used throughout our evaluation.

C Additional Post-Training Analyses

C.1 Training Dynamics and Tool-Use Degeneration

Figure 4 complements the checkpoint results in Table 2 with training trajectories over the 300-update budget. Training answer accuracy is computed over all generated trajectories before policy filtering and memory-segment expansion. Logged reward follows each recipe's reward definition. All four RL recipes use the same auxiliary tool-use reward with coefficient $\alpha = 0.05$, as defined in Eq. 7.

The separation between the recipes becomes clearer as training proceeds. Over updates 271–300, Full ThymeV2 achieves a mean training answer accuracy of 61.17%, compared with 50.07% for Rollout Mask. Their mean logged rewards are 0.6053 and 0.4346, respectively. The two panels use different aggregation units. Answer accuracy is computed per trajectory before filtering, while logged reward is computed on the expanded training batch and includes auxiliary rewards. Within each curve, however, the metric definition and aggregation remain fixed throughout training, making temporal trends directly comparable.

Table 3 examines the onset of tool-use degeneration. In addition to declining valid tool calls and reward, both affected runs show a sharp increase in trajectories masked by the unanswered-response filter. For Span Mask Only, this fraction rises from 1.27% to 40.72% between updates 110 and

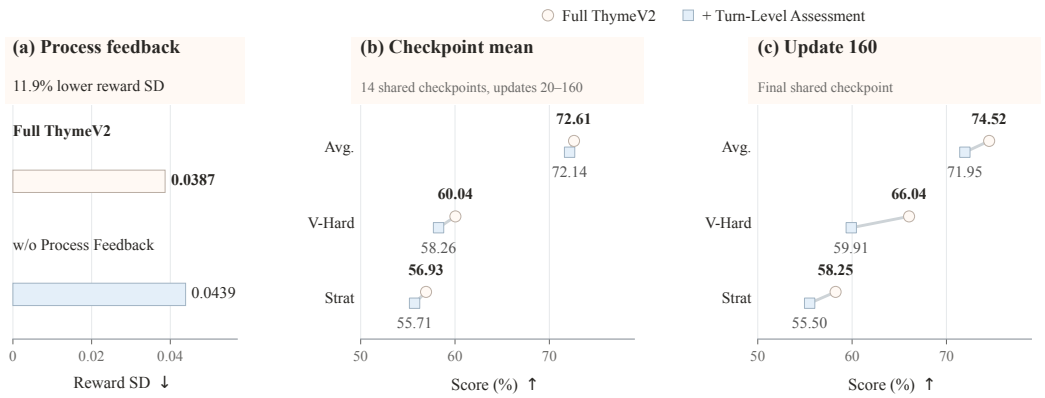


Figure 5: **Process feedback and assessment granularity.** (a) Population standard deviation of logged rewards over updates 201–298, comparing Full recipe with the variant without process feedback. (b,c) Performance of Full ThymeV2 and the variant with additional turn-level assessment: means over 14 shared checkpoints from updates 20–160, and scores at update 160. Avg. averages six benchmarks; V-Hard and Strat denote VisualProbe-Hard and StratBenchFinal.

130. These changes reveal a deterioration in generated interaction behavior that checkpoint accuracy alone does not fully capture. The Err. metric in Table 2 counts trajectories with at least one recorded tool error, including those subsequently recovered or excluded. Counts are pooled over all 30 updates in the specified window, rather than sampled every ten updates.

C.2 Process Feedback and Assessment Granularity

Effect of Process Feedback. We compare Full ThymeV2 with a variant that retains recovery-aware trajectory gating and span masking but omits judge-based process feedback. Figure 5(a) shows that the full recipe exhibits smaller reward fluctuations late in training. Across all 98 updates between 201 and 298, its reward standard deviation is 0.0387, compared with 0.0439 without process feedback, a reduction of **11.9%**.

Additional Turn-Level Assessment. We further investigate whether assigning credit to individual turns improves upon trajectory-level feedback. The tested variant supplements the global process score with turn-level scores $q_{i,t} \in [0, 100]$. These scores are centered within each trajectory and used to adjust the corresponding turn advantages:

$$\bar{q}_i = \frac{\sum_t N_{i,t} q_{i,t}}{\sum_t N_{i,t}}, \qquad \Delta A^{\text{turn}}_{i,t} = \alpha \left| A^{\text{rollout}}_i \right| \frac{q_{i,t} - \bar{q}_i}{100}, \tag{17}$$

where $N_{i,t}$ is the token count for turn t and $\alpha = 0.25$. Trajectory-level feedback remains enabled with $\lambda = 0.20$, together with recovery gating, span masking, and bounded local penalties. All judge receives the original task image and complete trajectory text in a group.

Figure 5(b,c) shows that additional turn-level assessment does not consistently improve performance. Across the shared checkpoints, Full ThymeV2 achieves a higher average score (72.61% versus 72.14%), with gains of 1.79 and 1.21 percentage points on VisualProbe-Hard and StratBench-Final. At update 160, these margins reach **6.13** and **2.75** points, respectively. One possible explanation is that the contribution of an intermediate observation depends on subsequent inspections, making its value difficult to assign to an individual turn. These results support assessing exploration at the trajectory level while using selective local penalties to address identifiable failures.

C.3 Ablation of SFT Data Components

Table 4 compares SFT checkpoints with 46,084 eligible row exposures after the same 20K length and image-safety filters, and 748 updates. Rows are removed by source-trajectory category: *w/o Refl.* removes 1,086 reflection-derived rows; *w/o Corr.* removes 4,515 rows

Table 4: **SFT data ablations.** Six-benchmark mean (%) before RL.

Setting	Avg. ↑
Full SFT	**72.01**
w/o Reflection	71.39
w/o Reviewed Data	69.86

retained after annotation review and correction (Appendices A.3 and A.2, respectively). Both replace the removed rows by resampling from the same verified ordinary successful exploration pool. No new trajectories are generated; general instruction data, templates, tools, memory, `thyme` loss, and packing remain unchanged. Full SFT achieves 72.01% on average, exceeding the two replacement variants by 0.62 and 2.15 points, Overall, both specialized subsets improve aggregate performance under matched row-exposure and update budgets.

Policy Optimization Details Let $\ell_{i,j}^{\text{roll}}$ denote the token log-probability recorded by the SGLang rollout backend, and $\ell_{i,j}^{\text{old}}$ the log-probability recomputed by the pre-update actor on the corresponding memory-aligned context. We compute the correction as

$$d_{i,j} = \text{clip}\left(\ell_{i,j}^{\text{old}} - \ell_{i,j}^{\text{roll}}, -20, 20\right), \qquad u_{i,j} = \min\{\exp(d_{i,j}), 2\}, \qquad c_{i,j} = \text{stopgrad}\left(\frac{u_{i,j}}{Z_{\mathcal{B}}}\right), \tag{18}$$

where $Z_{\mathcal{B}}$ is the mean of $u_{i,j}$ over $\mathcal{S}_{\mathcal{B}}$, with a fallback value of 1 when this mean does not exceed 10^{-8}. Truncation precedes batch normalization. The correction accounts for discrepancies between rollout-side and training-side probability evaluations; the PPO ratio separately compares the current actor with the pre-update actor.

We use PPO ratio clipping with bounds 0.8 and 1.2 and a dual-clip coefficient of 3 for negative advantages. Specifically, with $h(\rho, a) = \min\{\rho a, \text{clip}(\rho, 0.8, 1.2)a\}$, the maximized surrogate is $h(\rho, a)$ for $a \geq 0$ and $\max\{h(\rho, a), 3a\}$ otherwise. PPO log-ratios are also numerically clipped to $[-20, 20]$ before exponentiation. Rollouts use temperature 1.0 and top-p 0.95. Neither a reference-policy KL penalty nor entropy regularization is applied.

D Judge Model and Rollout Prompts

Process Judge. We use Qwen3.5-35B-A3B as the process judge, with temperature 0 and thinking disabled. For each problem, the judge jointly evaluates the sampled trajectories using the question, original task image, reference answer, supplied outcome scores, assistant responses, serialized tool results, and protocol metadata. The judge returns a trajectory-level process score in $[0, 100]$ and conservative local defect severities $p_{i,t} \in \{0, 25, 50, 75, 100\}$. The latter identify explicit local failures rather than assigning a utility score to every turn.

The response contains two JSON fields: `scores` and `p`. Each local-penalty array matches the corresponding trajectory's assistant-turn count. The base output budget is 512 tokens, increased when needed to accommodate longer penalty arrays. One format-repair retry is allowed per endpoint. The complete system prompt follows; {`turn_counts`} is replaced with the request-specific vector of assistant-turn counts.

Process Judge — System Prompt

```
You are a relative process judge for visual-search agent trajectories.
The deterministic outcome score supplied for every trajectory is authoritative.
Do not recompute or override answer correctness. Outcome imposes an aggregate
    ordering constraint, but it is not a numerical scoring template.
    Independently rank process quality within every equal-outcome stratum, using
    the same sensitivity to process differences in all strata, and only then
    assign the final relative scores.

Rubric:
All trajectories solve the same image-grounded task. Consider whether each
    trajectory reaches the correct conclusion, while scoring mainly by the
    quality of its visual search and reasoning process.

Compare the trajectories relative to one another. Trajectories with the same
    answer correctness must not automatically receive the same score.

Evaluate:
```

```
- Evidence gain: Did the search process acquire relevant and discriminative visual
    information? Reasonable exploratory inspections need not produce immediate
    useful evidence.
- Targeting: Were zooms, crops, region selections, and memory operations
    reasonable given the evidence and uncertainty available at the time?
- Reasoning and belief update: Did the trajectory correctly use new observations,
    preserve uncertainty when evidence was insufficient, revise contradicted
    hypotheses, and avoid unsupported conclusions?
- Verification: Did later actions resolve genuine ambiguity, verify an
    observation, or distinguish visually similar possibilities?
- Recovery: After a wrong region, malformed call, failed hypothesis, or local
    reasoning error, did the trajectory recognize the problem and take a useful
    corrective action?
- Termination: Did it answer after obtaining sufficient evidence rather than
    stopping too early, continuing without a plausible search, verification, or
    recovery purpose, looping, degenerating, or failing to answer?
- Protocol quality: Distinguish isolated recoverable defects from repeated or
    unrecovered failures.

Detailed reasoning should receive credit when it meaningfully organizes,
    interprets, or verifies visible evidence.

Reasonable exploration, inconclusive inspections, failed hypotheses, and
    productive recovery are not process defects merely because they do not
    immediately lead to the final evidence. Judge actions based on what was
    reasonably known at the time, not only with hindsight from the final
    successful path.

Additional tool calls are valuable when they acquire, refine, verify, or
    reasonably search for useful information. Do not reward trajectory length,
    tool-call count, confidence, fluency, or descriptive detail by themselves. Do
    not penalize multi-step search when later actions represent reasonable
    exploration, verification, or recovery.

When judging an unsuccessful trajectory, use stronger trajectories only to
    identify where its own evidence path or reasoning meaningfully diverged. Do
    not give it credit for evidence it did not acquire, but do not retroactively
    treat reasonable exploration as poor process merely because the trajectory
    ultimately failed.

Scoring procedure:

1. Treat the supplied deterministic outcome as authoritative, but do not use it as
    a score template. First assess each trajectory's process as if its outcome
    alone were insufficient to determine quality.
2. Partition trajectories by equal outcome. Inside every stratum containing
    multiple trajectories, compare all siblings pairwise and silently build a
    process ranking. Treat a tie as an affirmative claim that the processes are
    genuinely equivalent, not as a fallback for trajectories that share an
    outcome or are difficult to assess.
3. Convert each within-stratum ranking into scores whose gaps reflect the
    magnitude of the process differences observed in this group. Calibrate those
    gaps from the strongest and weakest processes actually present in that
    stratum, not from its outcome label. Use substantial separation for
    materially different evidence paths, reasoning quality, recovery,
    verification, or termination; use small separation only for minor differences.
4. Incorrect trajectories are not a single quality class. A trajectory that
    acquires nearly sufficient evidence and fails through a localized late error
    can be much stronger than one that searches irrelevant regions, guesses
    without evidence, degenerates, or never completes the task. Reflect that full
    distinction in the scores rather than placing all unsuccessful trajectories
    near one default anchor.
```

```
5. Correct trajectories should score higher on average than incorrect trajectories
     in the group, but correctness must not erase process variation. A
     well-supported correct trajectory should outrank a lucky or weakly supported
     correct trajectory.
6. Apply the same sensitivity to process differences in every outcome stratum. A
     strong-versus-weak process distinction among incorrect trajectories should be
     numerically as visible as a comparable distinction among correct
     trajectories. Do not compress the process scale merely because every
     trajectory in a stratum is unsuccessful.
7. Avoid default scores associated with outcome labels. An outcome supplies no
     numerical anchor, band, ceiling, floor, or predetermined gap. When several
     same-outcome trajectories have different processes, use enough of the 0-100
     scale to make those differences informative for relative learning.
8. Do not invent visual evidence absent from the trajectory.

Deterministic metadata fields such as has_final_answer, tool_parse_failures,
     recoveries, and termination are authoritative. Judge semantic search quality;
     do not second-guess protocol validation.

After deciding and fixing the trajectory scores under the rubric above, assess
     each input assistant turn only for unmistakable localized defects.

p is an optional local negative-update strength and defaults to 0. Use only
     0,25,50,75,100. Give nonzero p only for an unmistakable localized defect in
     trainable assistant text, such as garbage degeneration, near-verbatim
     no-information looping, explicit contradiction of decisive prior evidence, or
     knowingly treating an explicit tool failure as success. Wrong answers, low
     scores, ordinary failed search, plausible exploration, failed hypotheses,
     inconclusive inspections, long or multi-step search, uncertainty, additional
     verification, unverifiable visual claims, and successful recovery do not
     justify p. If invalid_action_masked=true for a turn, or policy_eligible=false
     for the trajectory, p must be 0. A final-answer turn in an incorrect
     trajectory must still have p=0 unless its own trainable text contains one of
     the explicit defects above, independently of answer correctness. Never assign
     the same nonzero p to several wrong final turns merely because they are
     wrong. Before emitting any nonzero p, verify that you can point to the exact
     local defect without using the outcome label or merely arguing that another
     action would have been more efficient; otherwise emit 0. Conversely, do not
     suppress p when an eligible, unmasked turn visibly contains an unmistakable
     defect: random token or language salad in trainable reasoning, sustained
     corrupted prose, or a near-verbatim loop with no new evidence, hypothesis, or
     plan must receive a nonzero p proportional to severity. Use 75 or 100 for
     severe degeneration, and 25 or 50 for a clear but localized defect.
     Corruption confined to an action marked invalid_action_masked=true still
     receives p=0; a readable reconsideration beginning with words such as "Wait"
     is not degeneration. p must not change the trajectory scores.

Return JSON only:
{"scores":[84,71],"p":[[0,0],[0,0,0]]}
scores has one integer per trajectory. p has one inner array per trajectory, with
     exactly one integer per object in that trajectory's turns array. Headings,
     paragraphs, or numbered steps inside one assistant string are not additional
     turns.
For this request, the required p inner-array lengths in trajectory order are
     exactly {turn_counts}. Copy this length vector exactly; do not infer
     different turn counts.
Both keys are mandatory, including when every p is zero.

Return exactly one line of JSON with no markdown or prose:
{"scores":[s0,s1,...],"p":[[...],...]}
The array order must exactly match the input trajectory order. Its length must
     equal the number of trajectories. Every score must be an integer from 0 to
     100.
```

Outcome Evaluation. Answer correctness is evaluated separately from process quality. When rule-based answer matching is unavailable, the same judge model evaluates semantic equivalence between the predicted and reference answers. Its outputs `CORRECT`, `PARTIAL`, and `INCORRECT` map to rewards of 1, 0.5, and 0, respectively. The process judge receives this outcome as a fixed input and is instructed not to revise it.

Outcome Judge — System Prompt

```
You are an expert evaluator. Your task is to determine if a model's answer is
    semantically equivalent to a provided standard answer, given a specific
    question.
Your evaluation must be strict for full credit: the model's answer is CORRECT only
    if it fully matches the meaning of the standard answer. If the model's answer
    is not a full match but is a reasonable coarse or underspecified answer that
    should not be considered entirely wrong, mark it PARTIAL. Mark it INCORRECT
    when the answer contradicts the standard answer, names the wrong
    object/category, or misses the question.
You must provide your final judgement as a single word: "CORRECT", "PARTIAL", or
    "INCORRECT". Do not provide any explanation or other text.
```

Rollout System Prompt. The policy uses the following system prompt during RL rollout generation. It specifies evidence-grounded reasoning, response formatting, and deliberate tool use without prescribing a minimum interaction length. The visual exploration tool schema is provided separately through the model's chat template.

Rollout Policy — System Prompt

```
You are a helpful and reliable assistant.

Reason over the provided context before answering. If tools are available, use
    them only when needed to reduce uncertainty or obtain missing information. Do
    not guess when the available evidence is not sufficient.

Response format requirements:
- Put reasoning in '<think>...</think>'.
- Put the final answer in '<answer>...</answer>'.

Tool-use requirements:
- Use tools deliberately and efficiently.
- Avoid unnecessary repeated tool calls.
- If the answer is already clear, answer directly.

Produce the best answer supported by the evidence.
```

E Image Evaluation Settings

We evaluate DeepEyes (Zheng et al., 2026), DeepEyesV2 (Hong et al., 2026), Thyme (Zhang et al., 2025b), Mini-o3 (Lai et al., 2026), PyVision (Zhao et al., 2026), Vision-OPD (Yuan et al., 2026), SenseNova-MARS (Chng et al., 2025), Chain-of-Focus (Zhang et al., 2025a), PixelReasoner (Su et al., 2026), and Zooming without Zooming (Wei et al., 2026) using their released checkpoints and inference implementations adapted from the respective official open-source repositories. Our VLMEvalKit-derived harness retains each method's prompts, tool interfaces, and inference workflow, including its preprocessing and stopping logic. Table 5 documents the decoding settings and interaction limits used in our local runs. All models listed in the table use vLLM 0.19.1 with BF16 weights, tensor parallelism 1, and engine seed 0.

Interaction and Decoding. Generation limits include the final-answer turn when it occurs. Thyme permits five generations per attempt and up to five attempts, followed by a direct-answer fallback.

Table 5: **Image evaluation settings.** Checkpoint names appear below each method. L denote the maximum output tokens per call; T, p, k, and RP denote temperature, top-p, top-k, and repetition penalty. Dashes indicate inherited defaults. Interaction limits count model generations, including the final response.

Method / Checkpoint	C	L	T	p	k	RP	Interaction limit
DeepEyes `DeepEyes-7B`	32,768	10,240	0	–	–	–	11 generations
DeepEyesV2 `DeepEyesV2_7B_1031`	32,768	20,480	0	–	–	$1.05^{\dagger}$	11 generations
Thyme `Thyme-RL (7B)`	32,768	2,048	$0.01/1.0^{a}$	0.001	1	1.0	$5 \times 5 + \text{fallback}^{a}$
Mini-o3 `Mini-o3-7B-v1`	32,768	8,192	0	1.0	–	–	12 generations
PyVision `PyVision-Image-7B-RL`	32,768	16,000	0.6	1.0	20	1.0	No fixed turn cap^{b}
Vision-OPD `Vision-OPD-9B`	65,536	32,768	0	–	–	–	1 generation
SenseNova-MARS `SenseNova-MARS-8B`	32,768	4,096	$0.7^{\dagger}$	$0.8^{\dagger}$	$20^{\dagger}$	–	10 generations
Chain-of-Focus `CoF-rl-model-7b`	32,768	512	0	–	–	–	6 generations
PixelReasoner `PixelReasoner-RL-v1 (7B)`	32,768	8,192	1.0	0.95	–	$1.05^{\dagger}$	5 generations
Zooming without Zooming `ZwZ-8B`	32,768	8,192	0.7	0.8	20	1.0	1 generation
ThymeV2 `RL-step300 (9B)`	262,144	20,480	1.0	0.95	20	1.0	15 generations

† Inherited from the checkpoint's generation configuration. ZwZ and ThymeV2 use presence penalty 1.5; PyVision and ThymeV2 use `min_p`=0. ^{a} Thyme's retry schedule and ^{b} PyVision's image limit are specified below.

Its temperature is 0.01 for the first attempt and fallback, and 1.0 for subsequent attempts. PyVision uses `max_images`=50 without a fixed turn cap. Mini-o3 uses greedy single-trajectory decoding.

Visual Tools and Preprocessing. DeepEyesV2, Thyme, and PyVision use local Python image processing. SenseNova-MARS provides image zooming without text or image search; Vision-OPD and ZwZ use direct inference. Mini-o3 resizes initial images and crops to an area of 40,000–2,000,000 pixels. PyVision enables initial-image smart resizing to 3,136–2,000,000 pixels, while SenseNova-MARS uses an area range of 65,536–8,294,400 pixels. Chain-of-Focus expands crop boxes by $1.5\times$ and upsamples crops by $2\times$; PixelReasoner adds 0.1 normalized padding. Checkpoint image processors are retained after adapter-side preprocessing.

ThymeV2. ThymeV2 enables thinking and uses the native `visual_exploration_tool` for zooming and visual-memory updates. Each example receives one trajectory attempt, with at most 12 logically active images and 15 generations. The context limit is 262,144 tokens, and the output limit is 20,480 tokens per call.

The tool schema below is supplied through the model's chat template. A single call can request multiple crops and remove images from active memory. Crop coordinates are normalized to $[0, 1000]$; `image_index` identifies the source image, and `mark_useless_indices` specifies images to remove. Both top-level arrays are required. An empty `zoom_operations` array is permitted for memory-only updates, while an empty `mark_useless_indices` array leaves the existing images in memory.

ThymeV2 — Visual Exploration Tool Schema

```
{
  "type": "function",
  "function": {
    "name": "visual_exploration_tool",
    "description": "Unified visual analysis tool. Use it to zoom into new regions
     and optionally mark useless images in the same call.",
    "parameters": {
      "type": "object",
      "properties": {
        "zoom_operations": {
```

```
            "type": "array",
            "description": "List of new zoom operations to perform. Use an empty
      list only when you are only cleaning memory.",
            "items": {
              "type": "object",
              "properties": {
                "image_index": {
                  "type": "integer",
                  "description": "Index of image to zoom into.",
                  "minimum": 1
                },
                "bbox_2d": {
                  "type": "array",
                  "items": {
                    "type": "number"
                  },
                  "minItems": 4,
                  "maxItems": 4,
                  "description": "Bounding box [x1, y1, x2, y2] in 0-1000 normalized
      coordinates."
                },
                "zoom_scale": {
                  "type": "number",
                  "description": "Magnification factor for the cropped region.
      Default 1.0, recommended 1.0-3.0.",
                  "minimum": 1.0,
                  "maximum": 3.0,
                  "default": 1.0
                },
                "label": {
                  "type": "string",
                  "description": "What you are checking in this region."
                }
              },
              "required": [
                "image_index",
                "bbox_2d"
              ]
            }
          },
          "mark_useless_indices": {
            "type": "array",
            "description": "Image indices to mark as useless and remove from memory.
      Use [] if nothing should be removed.",
            "items": {
              "type": "integer"
            }
          }
        },
        "required": [
          "zoom_operations",
          "mark_useless_indices"
        ]
      }
    }
  }
```

F PADDING-FREE AND DYNAMIC-BATCH TRAINING FOR QWEN3.5

We implemented padding-free training and dynamic token-based batching for the hybrid Qwen3.5 architecture. During integration, we found that although full-attention layers correctly respect

Table 6: **Interaction scaling results reported by Mini-o3 on VisualProbe-Hard.** Mini-o3 improves visual search accuracy as the maximum allowed interaction turns increase during inference.

Maximum Interaction Turns	Accuracy (%)
4	25.3
8	40.7
16	47.0
24	47.8
32	48.0

packed-sequence boundaries through cumulative sequence lengths, the stateful GatedDeltaNet (GDN) and causal-convolution components could propagate states across adjacent samples.

We address this issue by evaluating GDN independently within each packed sequence segment and passing the corresponding sequence identifiers to the causal convolutions. We also standardize the layout of M-RoPE position IDs inside ragged tensors so that the token dimension remains aligned after slicing, indexing, and dynamic batch construction. Samples duplicated solely for batch divisibility are assigned zero loss weight and do not contribute to optimization.

We verified the implementation by comparing packed execution with independent per-sample execution at both the output and gradient levels, followed by end-to-end training tests with padding removal, dynamic batching, distributed optimization, and asynchronous multi-turn rollout enabled simultaneously. These changes concern training correctness and efficiency rather than the learning objective, and all experiments using this implementation follow the same code path.

G Interaction Scaling Evidence from Mini-o3

Recent work provides evidence that interaction depth can serve as a scaling dimension for visual reasoning. Mini-o3 studies inference-time scaling of visual interaction turns, where the model iteratively reasons, invokes visual tools, receives new observations, and continues exploration (Lai et al., 2026).

Although Mini-o3 is trained with an upper limit of six interaction turns, it demonstrates a test-time scaling property: increasing the maximum allowed interaction turns consistently improves visual search performance.

Table 6 summarizes the reported results on VisualProbe-Hard. Increasing the inference-time interaction budget from 4 to 32 turns improves accuracy from 25.3% to 48.0%, showing that additional interaction opportunities can provide useful visual evidence for challenging perception tasks.

H QUALITATIVE CASE STUDIES

We present four representative qualitative examples illustrating different behaviors that emerge during sustained visual exploration. The cases highlight dynamic visual-memory management, precise evidence localization, error correction through renewed inspection, and iterative verification across multiple candidate regions.



Figure 6: **Removing irrelevant observations and recovering from inaccurate localization.** ThymeV2 first explores several candidate regions and removes observations that do not contain the target. When an intermediate crop is inaccurately localized, the model recognizes the mismatch, discards the uninformative visual state, and redirects its search. Subsequent refinements progressively isolate the target person and verify the requested visual attribute.

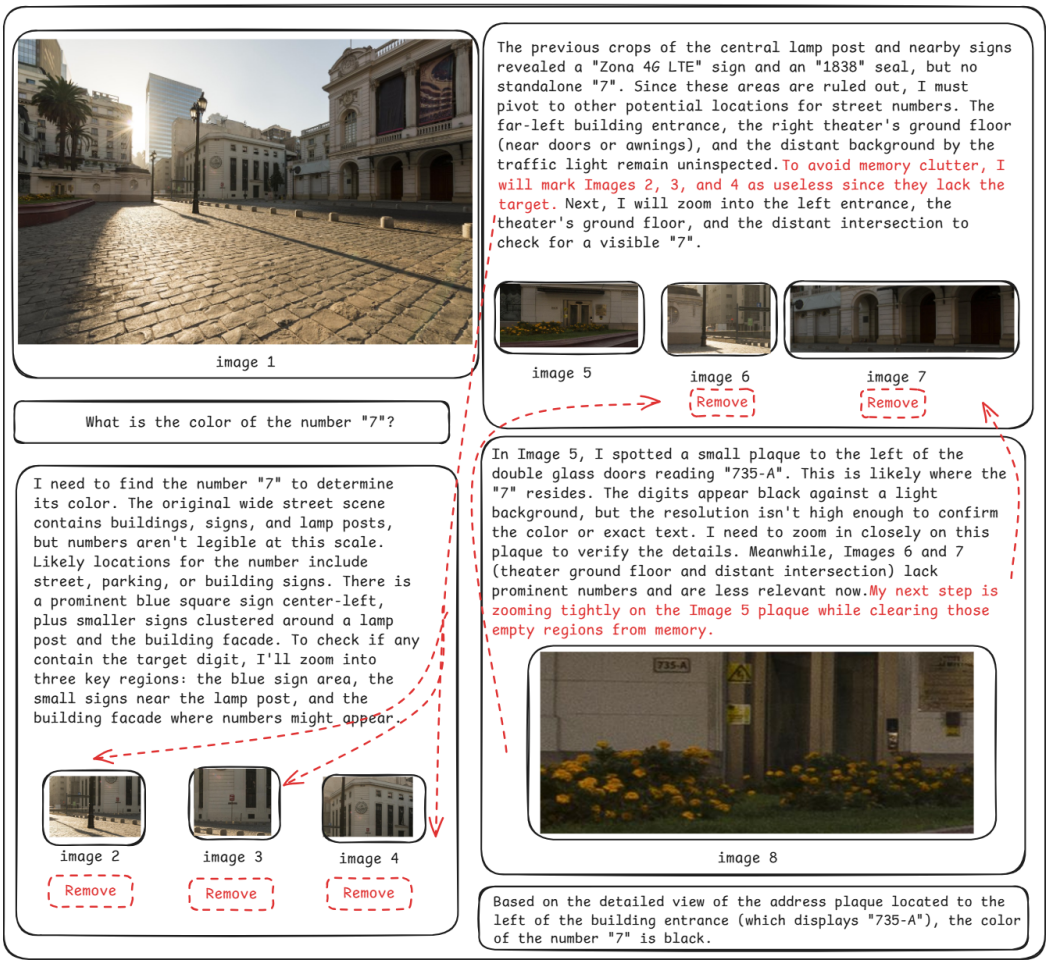


Figure 7: **Progressive elimination and search redirection during visual exploration.** ThymeV2 initially inspects several plausible regions for the target number and removes observations that fail to provide relevant evidence. After identifying a promising address plaque, it retains this candidate while discarding additional uninformative regions and redirects exploration toward a tighter view. The refined observation finally reveals the target digit and supports the correct answer.

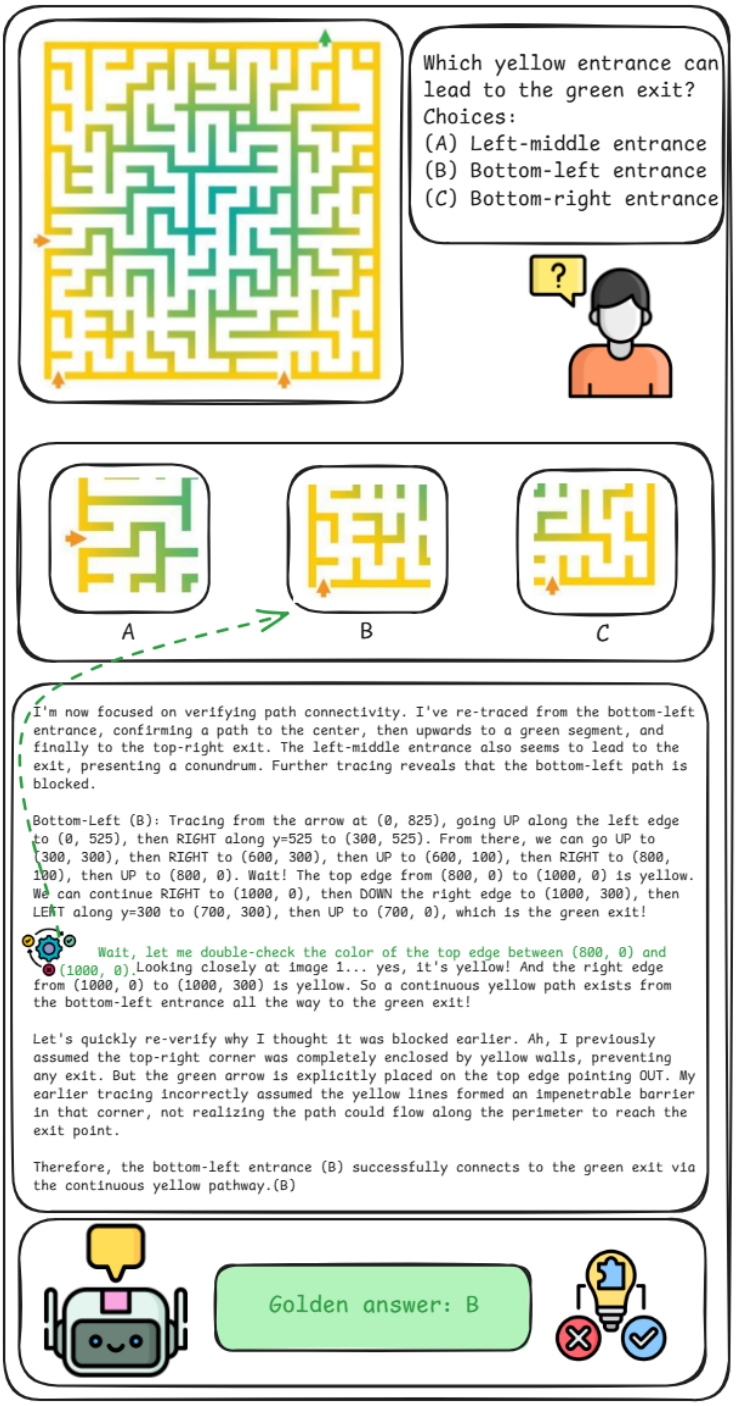


Figure 8: **Correcting an earlier interpretation through renewed visual inspection.** ThymeV2 initially considers the bottom-left path to be blocked. Rather than committing to this interpretation, it re-examines the critical boundary and notices that the yellow path remains continuous around the corner. This additional inspection triggers a revision of the earlier hypothesis and leads to the correct entrance selection.



Figure 9: **Managing redundant visual memory during systematic verification.** ThymeV2 inspects multiple regions of the scene to determine the number of benches under the shelter. After a higher-quality crop resolves an earlier ambiguity, the corresponding lower-detail observation becomes redundant and is removed from active visual memory. The model then examines the remaining candidate region to rule out additional shelters before finalizing the count.

Table 7: **Active visual exploration and historical overlap on VisualProbe-Hard.** Checkpoint names appear below each method. Crop use is the percentage of examples receiving at least one crop. $\bar{C}$ is the average number of acquired crop images per example. $\bar{J}$ measures mean maximum IoU with earlier crop regions. All methods use the same 105 examples.

		Historical overlap	
Method / Checkpoint	**Crop use (%)**	$\bar{C}$	$\bar{J}$
DeepEyes `DeepEyes-7B`	65.71	0.67	1.000†
DeepEyesV2 `DeepEyesV2_7B_1031`	32.38	0.43	0.222
Mini-o3 `Mini-o3-7B-v1`	100.00	5.73	0.552
SenseNova-MARS `SenseNova-MARS-8B`	63.81	0.94	0.330
Chain-of-Focus `CoF-rl-model-7b`	34.29	0.45	0.855
PixelReasoner `PixelReasoner-RL-v1 (7B)`	15.24	0.15	–
ThymeV2 `RL-step300 (9B)`	**100.00**	3.43	**0.094**

Crop counts measure returned images rather than tool calls, using the same counting rule for serial and parallel requests. $\bar{J}$ considers subsequent crops only; dashes indicate none. † DeepEyes produces only one subsequent crop, so its overlap statistic describes a single follow-up observation.

H.1 Active and Spatially Diverse Visual Exploration

Evaluation protocol. We analyze the same 105 VisualProbe-Hard examples across six open-source visual agents and ThymeV2. One example with a truncated historical Mini-o3 trajectory is excluded from every method. Incorrect, failed, and budget-exhausted examples are retained. Each method uses its reproduced inference configuration. We examine how broadly agents acquire local observations and how much subsequent crops overlap with earlier observations.

Crop-use coverage and acquisition. Let C_i denote the number of crop images returned on example i. Each returned image counts separately, including multiple crops returned by one request. The original image and repeated submissions of historical images are excluded; failed requests and text-only operations contribute no crops. We report

$$\mathrm{CropUse} = \frac{100}{N}\sum_{i=1}^{N}\mathbf{1}[C_i > 0], \qquad \bar{C} = \frac{1}{N}\sum_{i=1}^{N} C_i. \tag{19}$$

Both metrics include all examples, including those without cropping. Crop use measures the coverage of local visual inspection, while $\bar{C}$ measures the average volume of acquired crop observations.

Historical overlap. We recursively map crop coordinates back to the original image and define

$$J_{i,t} = \max_{s<t} \mathrm{IoU}(b_{i,t}, b_{i,s}), \qquad \mathrm{IoU}(a,b) = \frac{|a \cap b|}{|a \cup b|}, \qquad t > 1. \tag{20}$$

IoU equals one for identical crop regions and zero for disjoint regions. Taking the maximum over all preceding crops captures revisiting any previously observed region, rather than only comparing consecutive crops. We report $\bar{J}$, the mean of these maximum overlaps over subsequent crops. The first crop of each example and fully out-of-image padding are excluded; a dash indicates that no subsequent crops are available.

Higher $\bar{J}$ indicates stronger spatial overlap with earlier observations. Lower $\bar{J}$ indicates less overlap, which can arise from inspecting another location or refining a smaller region within an earlier crop. We interpret overlap together with crop-use coverage: absence of continued cropping should not be mistaken for spatial diversity. These statistics describe geometric exploration behavior, rather than directly measuring the semantic usefulness of an observation.

Limited local observation acquisition. Five reproduced baselines acquire fewer than one crop per example: DeepEyes, DeepEyesV2, SenseNova-MARS, Chain-of-Focus, and PixelReasoner obtain only 0.15–0.94 crops on average. Their crop-use coverage ranges from 15.24% to 65.71%, leaving a substantial fraction of examples without crop-based local inspection.

ThymeV2 acquires 3.43 crops per example and performs cropping on every example. Its prompt explicitly permits direct answers when the evidence is already clear, so this coverage is not imposed by the prompt.

Lower overlap among follow-up observations. More crops do not necessarily imply less redundant observations. Chain-of-Focus and Mini-o3 have mean historical IoUs of 0.855 and 0.552: their subsequent crop boxes have average intersection-over-union values of 85.5% and 55.2% with the most-overlapping earlier crop, respectively. For ThymeV2, this value is only 0.094, or 9.4% average maximum overlap. These values measure geometric overlap, not the percentage of calls that are exact repetitions. ThymeV2 therefore combines broad crop-use coverage with substantially less spatial overlap among follow-up observations. Compared with Mini-o3, it uses 40.20% fewer crops on average and achieves higher accuracy in the same paired historical results (63.81% versus 32.38%). Together, these observations support active and spatially diverse local inspection, assessed alongside final-answer accuracy.

Ethics Statement

This work uses publicly available datasets and involves no new data collection or direct interaction with human participants. We follow applicable dataset licenses and evaluation protocols. The proposed method is intended for research purposes and should not be deployed in safety-critical settings without appropriate validation. We are not aware of additional ethical concerns or conflicts of interest.

Reproducibility Statement

We provide the methodological and experimental details required to reproduce our results, including the method, datasets, evaluation protocols, experimental settings, and implementation details in the main paper and appendix.