VISPO: VERIFICATION-STATE-AWARE POLICY OPTIMIZATION FOR RELIABLE MULTIMODAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards is crucial for enhancing the complex reasoning capabilities of multimodal large language models. However, existing paradigms primarily rely on outcome supervision, providing sparse guidance for intermediate reasoning and lacking reliable verification signals to mitigate hallucination propagation during iterative reasoning. To address these limitations, we propose **VISPO**, **V**er**I**fication-**S**tate-Aware **P**olicy **O**ptimization, which trains an agentic verifier to acquire evidence through tool interactions, perform evidence-grounded verification across support, insufficient, and contradict states, and generate reliable verification explanations for multimodal reasoning. Moreover, VISPO introduces region-level advantage redistribution to enable state-aware credit assignment and alleviate credit misalignment. To provide structured supervision for verifier training, we construct **FGMV**, a **F**ine-**G**rained **M**ultimodal **V**erification dataset with process-level state-aware verification annotations and explanations. Building upon VISPO, we further propose **VISPO-R**, a synergistic multimodal reasoning framework that leverages iterative verifier feedback via verification-aware hierarchical advantage assignment for reasoning refinement. Extensive experiments show that VISPO-R consistently improves Qwen3-VL backbones across both model scales, achieving average relative improvements of **13.6%**, **5.5%**, and **10.3%** on mathematical reasoning, general reasoning, and hallucination benchmarks. These results demonstrate that state-aware verification effectively improves the accuracy and reliability of multimodal reasoning.

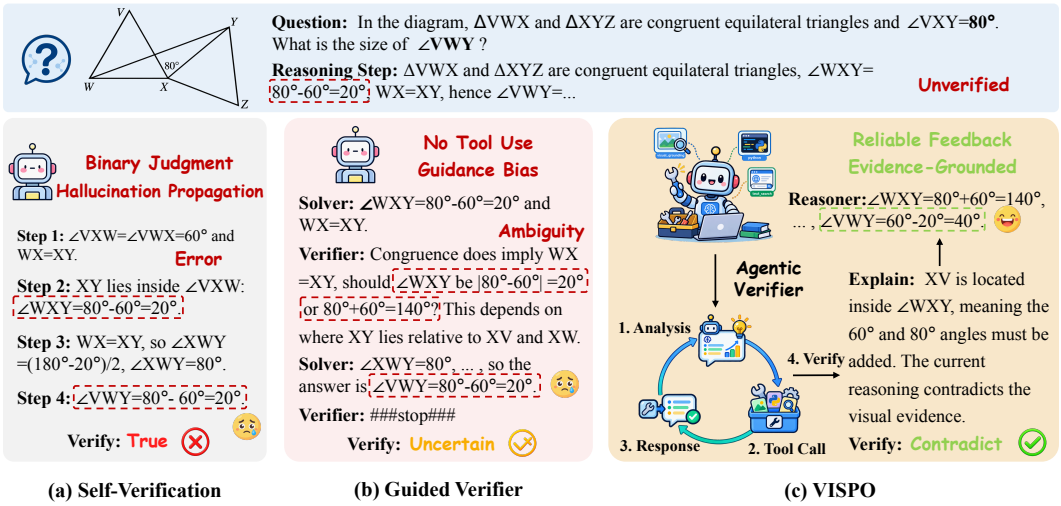
1 INTRODUCTION

Reinforcement learning with verifiable rewards improves the reasoning capabilities of multimodal large language models (MLLMs), extending group-relative optimization to visual tasks (Shao et al., 2024; Huang et al., 2026; Shen et al., 2025; Meng et al., 2025). Yet a misread chart legend or unsupported visual assumption can propagate through subsequent steps into a fluent but incorrect solution. Final-answer supervision alone cannot determine whether the underlying evidence was examined or intermediate reasoning was valid.

Self-verification and reward modeling offer a promising way to improve reasoning beyond final-answer supervision (Liu et al., 2025; Han et al., 2026; Du et al., 2025). As illustrated in Figure 1, **Self-Verification** may reuse the reasoner's erroneous step and reinforce it through a forced binary judgment, thereby causing **hallucination propagation**. **Guided Verifier** can provide textual feedback, but without active tool use it remains susceptible to **guidance bias** and lacks an explicit mechanism for acquiring additional evidence (Sun et al., 2026). These advances leave an open question: how should verification states guide credit assignment within verifier-generated trajectories and across the reasoner's iterative steps?

To address these challenges, we propose **VISPO**, **V**er**I**fication-**S**tate-Aware **P**olicy **O**ptimization, which trains an agentic verifier for evidence-grounded SUPPORT, INSUFFICIENT and CONTRADICT judgments with explanatory feedback. Its region-level advantage redistribution favors reliable regions in positive-advantage rollouts and concentrates penalties on weaker regions in negative-advantage rollouts. To train the verifier, we construct **FGMV**, a fine-grained multimodal verification

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) Self-Verification **(b) Guided Verifier** **(c) VISPO**

Figure 1: **Comparison among self-verification, Guided Verifier and VISPO.** (a) Self-verification propagates hallucination through a binary judgment. (b) Guided Verifier provides tool-free feedback but remains susceptible to guidance bias. (c) VISPO actively gathers multimodal evidence through tools and returns reliable feedback.

dataset through hard-example mining, dual-agent data collection and rejection sampling. Building on this verifier, **VISPO-R** freezes its parameters and learns iterative reasoning through verification-aware hierarchical advantage assignment: trajectory-level signals retain outcome supervision, while step-local signals provide fine-grained credit assignment for intermediate reasoning steps. VISPO learns how to verify; VISPO-R learns how to reason with reliable verification feedback.

In summary, our contributions are as follows:

- We propose **VISPO**, **V**er**I**fication-**S**tate-Aware **P**olicy **O**ptimization, which trains an agentic verifier to perform evidence-grounded three-state verification and generate reliable feedback. VISPO further introduces region-level advantage redistribution to alleviate credit misalignment.
- We construct **FGMV**, a **F**ine-**G**rained **M**ultimodal **V**erification dataset with process-level state-aware verification annotations and explanations for training the agentic verifier.
- Building upon VISPO, we propose **VISPO-R**, a synergistic multimodal reasoning framework that leverages verifier feedback via hierarchical advantage assignment, combining trajectory-level and step-local advantages for reasoning refinement. Experiments across both Qwen3-VL model scales demonstrate average relative gains of **13.6%**, **5.5%**, and **10.3%** on mathematical reasoning, general reasoning, and hallucination benchmarks.

2 Related Work

Multimodal Reasoning with Reinforcement Learning. GRPO and its variants have established outcome-based optimization for mathematical reasoning (Shao et al., 2024). Vision-R1, R1-VL, MM-Eureka, R1-Onevision and OpenVLThinker extend this paradigm to multimodal tasks through rule-based rewards and self-improvement (Huang et al., 2026; Zhang et al., 2025; Meng et al., 2025; Yang et al., 2025; Deng et al., 2025). Unlike these solver-oriented methods, VISPO focuses on training an agentic verifier that performs state-aware verification, while VISPO-R integrates the VISPO-trained verifier into a synergistic reasoner training framework and inference process.

Self-Verification and Multimodal Reward Models. Self-verification and reward modeling provide supervision beyond final-answer correctness (Liu et al., 2025; Han et al., 2026; Qiu et al., 2026). However, self-verification may reinforce erroneous visual interpretations, while tool-free feedback may lack the external evidence needed to identify and correct such errors. Process reward

models provide intermediate scores, and agentic reward models such as ARM-Thinker and TIM-PRM enable tool-based evidence acquisition (Wang et al., 2025b; Ding et al., 2026; Kuang et al., 2025). However, these approaches do not jointly address verification-aware policy optimization and fine-grained credit assignment for iterative reasoning. VISPO learns a dedicated three-state verifier with sign-aware regional credit redistribution. VISPO-R further exploits frozen verifier feedback through hierarchical advantage assignment for iterative reasoning improvement.

3 Problem Formulation

3.1 VISPO: State-Aware Verification

Let $x = (v, q, c, \mathcal{T})$ contain an image v, question q, reasoning step c and tools $\mathcal{T}$. After at most K generation rounds, verifier V_ψ returns $g = (z, e)$: an evidence-grounded explanation e and state $z \in S = \{\textsc{Support}, \textsc{Insufficient}, \textsc{Contradict}\}$. SUPPORT requires a correct final answer and valid, evidence-consistent reasoning; CONTRADICT denotes an incorrect answer or an invalidating reasoning error or hallucination. INSUFFICIENT denotes an unfinished response or insufficient evidence, provided that no invalidating error has been established. We group these cases together because they provide neither sufficient support for acceptance nor sufficient evidence for contradiction, requiring further reasoning or evidence acquisition.

Let $u_i = (l_i, a_i, o_i)$ denote interaction i, containing generated analysis l_i, a tool action $a_i \in \mathcal{T}$ and its environment observation o_i. We use $u_{<i} = (u_1, \ldots, u_{i-1})$ to denote the preceding interaction history, which is empty when $i = 1$. For an interaction history $u_{1:k} = (u_1, \ldots, u_k)$ with $k \le K$, the conditional probability factorizes as

$$p_\psi(u_{1:k} \mid x) = \prod_{i=1}^{k} V_\psi(l_i, a_i \mid x, u_{<i}) P_{\text{env}}(o_i \mid x, a_i), \tag{1}$$

The complete verification trajectory is $y = (u_1, \ldots, u_k, g)$. Its joint probability is

$$p_\psi(y \mid x) = p_\psi(u_{1:k} \mid x) V_\psi(g \mid x, u_{1:k}). \tag{2}$$

P_{env} denotes the tool environment. Only generated analysis, tool actions and the final verification result are governed by V_ψ, while tool observations are generated by the environment.

3.2 VISPO-R: Verifier-Synergistic Reasoning

Task Formulation. VISPO-R pairs a reasoner π_θ with a frozen VISPO verifier V_ψ. At turn j, the reasoner context is $T_j^{\text{R}} = v \oplus q \oplus \tau_{<j} \oplus g_{j-1}$, where $\tau_{<j}$ contains prior reasoner outputs and $g_0 = \varnothing$. This reduces context accumulation: prior reasoner outputs incorporate earlier feedback and relevant evidence, while the latest feedback identifies the remaining issues. In a tool-use turn, it samples a tool-use prefix $d_j^{\text{pre}} \sim \pi_\theta(\cdot \mid T_j^{\text{R}})$, including the reasoning content and tool action a_j. The environment executes a_j and returns o_j^{R}, after which the reasoner samples $d_j^{\text{post}} \sim \pi_\theta(\cdot \mid T_j^{\text{R}} \oplus d_j^{\text{pre}} \oplus o_j^{\text{R}})$. The reasoning step is represented as $d_j = d_j^{\text{pre}} \oplus d_j^{\text{post}}$. The verifier primarily assesses the current reasoning step and independently gathers evidence. With $T_j^{\text{V}} = v \oplus q \oplus \tau_{<j} \oplus d_j \oplus u_{j,1:k_j}^{\text{V}}$, it returns $g_j = (z_j, e_j) \sim V_\psi(\cdot \mid T_j^{\text{V}})$.

A plan or direct-answer turn samples d_j directly and is verified as a whole. Only the latest g_j is passed to the next turn, limiting context growth. The rollout terminates when the verifier supports the final answer or the interaction budget is exhausted. Thus, the reasoner uses tools to solve the task, while the verifier independently checks the resulting content and guides the next turn.

Training Procedure. VISPO optimizes only the agentic verifier, which learns the structured protocol through SFT and is optimized with verification-state-aware RL using sign-aware region-level advantage redistribution on FGMV. VISPO-R holds V_ψ fixed and updates only π_θ, using feedback-conditioned SFT followed by RL with final-answer, format, tool-use and state-transition rewards and hierarchical advantage assignment. The two stages address different credit-assignment problems: quality differences within a verification trajectory and differences between steps of a reasoning trajectory.

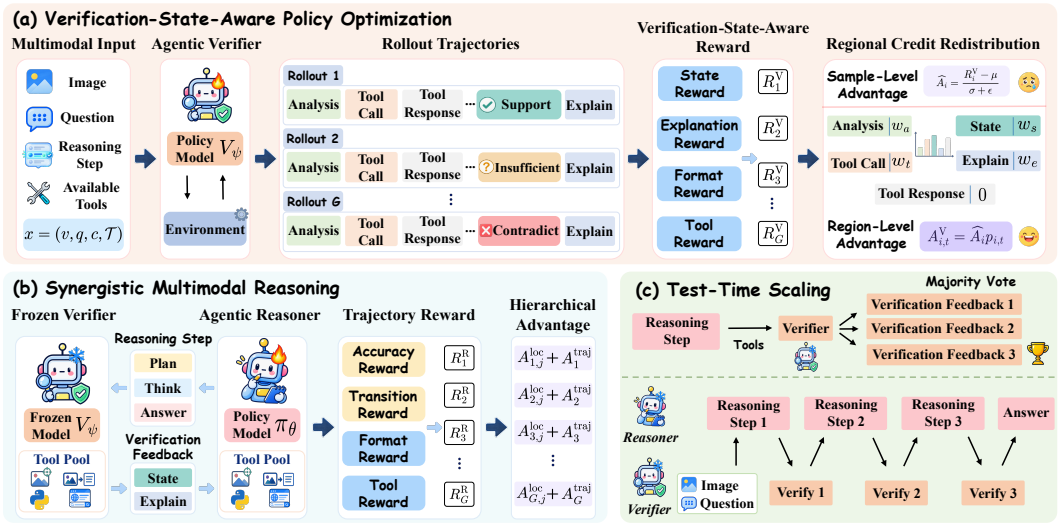


Figure 2: **Schematic diagram of the VISPO and VISPO-R frameworks. (a) Verification-state-aware policy optimization:** the agentic verifier generates state-aware verification trajectories, receives verification-state-aware reward and redistributes regional advantage while masking tool responses. **(b) Synergistic multimodal reasoning:** the reasoner learns from hierarchical trajectory-level and step-local advantages, guided by state-aware feedback from a frozen verifier. **(c) Test-time scaling:** the reasoner iteratively revises its reasoning based on verification feedback.

4 Methodology: VISPO and VISPO-R

In Section 4.1, we introduce **FGMV**, a fine-grained multimodal verification dataset constructed through agentic data generation and quality control. In Section 4.2, **VISPO** optimizes the agentic verifier with verification-state-aware rewards and regional credit redistribution. Finally, Section 4.3 introduces **VISPO-R**, which trains the reasoner with frozen-verifier feedback and hierarchical credit assignment combining trajectory-level and step-local advantages.

4.1 FGMV: Fine-Grained Multimodal Verification Data

Overview and Motivation. Binary labels provide sparse supervision for learning what evidence to acquire or how to provide reliable feedback. Therefore, we construct **FGMV**, a step-level dataset for training an agentic verifier to perform evidence-grounded verification of multimodal reasoning. Each accepted record contains the ordered trajectory, including verifier analysis, executable tool calls, corresponding real observations, verification states, and final evidence-grounded explanations.

Dual-Agent Verification Data Generation. We construct FGMV through dual-agent interaction on hard multimodal examples mined from curated sources. The Reasoner, primarily Qwen3-VL-8B-Instruct (Bai et al., 2025a), generates reasoning steps conditioned on the input, interaction history and verifier feedback, while the Verifier, primarily GPT-5.5 (OpenAI, 2026), checks each step using executable tools and returns a state–explanation pair. The resulting records associate each verification judgment with tool requests and real observations that support its explanation. Protocol validation and rejection sampling yield full action–observation demonstrations for verifier SFT and verified instances for online verifier RL. These complementary training signals supervise both how to acquire relevant evidence and how to use it to distinguish unresolved uncertainty from invalid reasoning. Hard-example mining and quality-control details are provided in Appendix C.1.

4.2 VISPO: Verification-State-Aware Policy Optimization

Verification-State-Aware Reward Design. During online training, the verifier accuracy reward for rollout i is

$$R^{\mathrm{V}}_{i,\mathrm{acc}} = \lambda_{\mathrm{s}} R_{i,\mathrm{state}} + \lambda_{\mathrm{e}} R_{i,\mathrm{explain}}, \tag{3}$$

where $R_{i,\text{state}} = \mathbb{I}[\hat{z}_i = z_i^*]$, $\hat{z}_i$ is the predicted state and z_i^* is the reference state. $R_{i,\text{explain}} \in [0,1]$ is the numerical score for explanation correctness, evidence grounding and corrective usefulness, specified in Appendix D. The uncapped reward combines the three rewards as

$$\bar{R}_i^{\text{V}} = \lambda_{\text{acc}}^{\text{V}} R_{i,\text{acc}}^{\text{V}} + \lambda_{\text{fmt}} R_{i,\text{fmt}} + \lambda_{\text{tool}} R_{i,\text{tool}}, \tag{4}$$

where $R_i^{\text{V}} = \min\{\bar{R}_i^{\text{V}}, R_{i,\text{c}}\} + R_{i,\text{len}}$, $R_{i,\text{c}}$ denotes a reward cap that limits the contribution of unreliable trajectories and $R_{i,\text{len}}$ is a piecewise-linear overlength penalty.

Regional Credit Redistribution. For the G verifier rollouts sampled for the same input, let R_i^{V} denote the final reward of rollout i. Using the group mean μ and standard deviation σ, we compute

$$\widehat{A}_i = \frac{R_i^{\text{V}} - \mu}{\sigma + \epsilon}. \tag{5}$$

Let $M_{i,t}^{(s)}$ denote the content-token mask of semantic region s, with $n_{i,s} = \sum_t M_{i,t}^{(s)}$ and $L_i = \sum_t m_{i,t}$. The masks are disjoint and satisfy $M_{i,t}^{(s)} \le m_{i,t}$: generated XML tags are treated as actor tokens, while tool observations have $m_{i,t} = 0$. Each region is assigned a priority coefficient α_s and a sign-aware quality weight $w_{i,s}$, defined in Appendix A.1. With other factors fixed, positive advantages allocate more credit to higher-quality regions, whereas negative advantages emphasize lower-quality regions. Under regional length weighting, the token allocation is defined as

$$p_{i,t} = (1-\rho)\frac{\sum_s \alpha_s w_{i,s} M_{i,t}^{(s)}}{\sum_s \alpha_s w_{i,s} n_{i,s}} + \rho \frac{m_{i,t}}{L_i}, \tag{6}$$

where $\rho = 0.2$ preserves a uniform baseline over all actor tokens. When the denominator is zero, we use $p_{i,t} = \frac{m_{i,t}}{L_i}$. The token-level advantage is computed as

$$A_{i,t}^{\text{V}} = \widehat{A}_i p_{i,t}. \tag{7}$$

This formulation performs fixed-total, sign-aware regional reweighting while assigning zero credit to environment observations.

VISPO Objective. Let $H_{i,t}^{\text{V}} = x_i \oplus y_{i,<t}$ contain the input, preceding verifier actions and tool observations. The importance ratio is

$$r_{i,t}(\psi) = \frac{V_\psi(y_{i,t} \mid H_{i,t}^{\text{V}})}{V_{\psi_{\text{old}}}(y_{i,t} \mid H_{i,t}^{\text{V}})}. \tag{8}$$

With $\bar{r}_{i,t}(\psi) = \text{clip}(r_{i,t}(\psi), 1-\varepsilon_{\text{low}}, 1+\varepsilon_{\text{high}})$, we maximize

$$\begin{aligned}\mathcal{L}^{\text{VISPO}}(\psi) = \mathbb{E}_{\substack{x\sim\mathcal{D}_{\text{V}},\\ \{y_i\}_{i=1}^{G}\sim V_{\psi_{\text{old}}}(\cdot|x)}} \Bigg[\frac{1}{G}\sum_{i=1}^{G} \Big\{ \sum_{t\in\mathcal{I}_i^{\text{V}}} \min\Big(r_{i,t}(\psi)A_{i,t}^{\text{V}}, \bar{r}_{i,t}(\psi)A_{i,t}^{\text{V}} \Big) \\ - \beta D_{\text{KL}}^{\text{V}}(V_\psi \| V_{\text{ref}}) \Big\} \Bigg],\end{aligned} \tag{9}$$

where $\mathcal{I}_i^{\text{V}} = \{t : m_{i,t} = 1\}$ indexes verifier-generated tokens.

4.3 VISPO-R: Synergistic Multimodal Reasoning

Frozen-Verifier Training. VISPO-R initializes the reasoner by SFT on tool-use and feedback-conditioned demonstrations. Frozen judgments $g_j = (z_j, e_j)$ serve two roles: explanations guide subsequent generation, while states supply transition rewards and step-local credit.

Trajectory Reward. The reasoner reward for rollout i combines final-answer accuracy, format, tool use and verification-state transitions, with $\lambda_{\text{st}} = 0.1$:

$$\bar{R}_i^{\text{R}} = \lambda_{\text{acc}}^{\text{R}} R_{i,\text{acc}}^{\text{R}} + \lambda_{\text{fmt}} R_{i,\text{fmt}} + \lambda_{\text{tool}} R_{i,\text{tool}} + \lambda_{\text{st}} R_{i,\text{st}}, \tag{10}$$

For $N_i \geq 1$ judgments, set $\Phi(\textsc{Contradict}) = 0$, $\Phi(\textsc{Insufficient}) = 0.5$ and $\Phi(\textsc{Support}) = 1$:

$$R_{i,\mathrm{st}} = \frac{[S_{i,\mathrm{st}}]_+}{\sqrt{N_i}} - [-S_{i,\mathrm{st}}]_+ , \tag{11}$$

$$S_{i,\mathrm{st}} = \Phi(z_{i,N_i}) - \sum_{j=1}^{N_i} [\Phi(z_{i,j-1}) - \Phi(z_{i,j})]_+ , \tag{12}$$

where $[\Delta\Phi]_+ = \max(\Delta\Phi, 0)$, $\Phi(z_{i,0}) = 0$, and $R_i^{\mathrm{R}} = \min\{\bar{R}_i^{\mathrm{R}}, R_{i,\mathrm{c}}\} + R_{i,\mathrm{len}}$. The state transition reward penalizes cumulative regressions and applies length normalization only to positive progress.

Verification-Aware Hierarchical Advantage Assignment. We combine trajectory-level advantages, which reflect reasoning outcomes, with local verification advantages for step-wise credit assignment. The local verification signal is derived from state transitions: $r_{i,j}^{\mathrm{loc}} = \Phi(z_{i,j}) - \Phi(z_{i,j-1})$. We set $r_{i,1}^{\mathrm{loc}} = \Phi(z_{i,1})$ for the initial verification state. We normalize trajectory rewards within each group and local state transition scores within each rollout:

$$A_i^{\mathrm{traj}} = \frac{R_i^{\mathrm{R}} - \mu^{\mathrm{traj}}}{\sigma^{\mathrm{traj}} + \epsilon}, \tag{13}$$

$$A_{i,j}^{\mathrm{loc}} = \frac{r_{i,j}^{\mathrm{loc}} - \mu_i^{\mathrm{loc}}}{\sigma_i^{\mathrm{loc}} + \epsilon}. \tag{14}$$

The combined advantage for actor token t weights the trajectory-level signal and the local verification signal of its generating step $j(t)$:

$$A_{i,t}^{\mathrm{R}} = \alpha A_i^{\mathrm{traj}} + (1-\alpha) A_{i,j(t)}^{\mathrm{loc}}. \tag{15}$$

Local credit affects only the corresponding step. Observations and verifier outputs are masked. Appendix A.1 provides further optimization details of VISPO-R.

5 Experiments

5.1 Experimental Setup

Benchmarks and Baselines. We evaluate on MathVerse, MathVista, MathVision, WeMath, MMMU, MMStar, HallusionBench and VL-RewardBench, covering mathematical, general, hallucination and reward-model evaluation (Zhang et al., 2024; Lu et al., 2024; Wang et al., 2024; Qiao et al., 2025; Yue et al., 2024; Chen et al., 2024; Guan et al., 2024; Li et al., 2025). VISPO and VISPO-R use Qwen3-VL-4B-Instruct and Qwen3-VL-8B-Instruct as backbones. General-model baselines include Gemma 3, Qwen2.5-VL, InternVL3 and Qwen3-VL (Gemma Team et al., 2025; Bai et al., 2025b; Zhu et al., 2025; Bai et al., 2025a). Proprietary baselines are GPT-4o, Claude-3.5-Sonnet and GPT-5-mini (OpenAI, 2024; Anthropic, 2024; OpenAI, 2025). We further compare with outcome-supervised GRPO (Shao et al., 2024), multimodal reward models (Wang et al., 2025b; Du et al., 2025; Ding et al., 2026; Sun et al., 2026) and multimodal reasoning models (Huang et al., 2026; Zhang et al., 2025; Yang et al., 2025; Deng et al., 2025; Tian et al., 2026; Wu et al., 2026).

Implementation Details. During verifier training, the SFT stage uses 2.5K synthetic examples, after which **VISPO** further optimizes on 7K synthetic examples from **FGMV**. To construct **VISPO-R**, we freeze the verifier and train the reasoner with SFT on 4K synthetic examples, followed by RL on 10K multimodal reasoning examples. At inference time, the VISPO-R reasoner iteratively revises each reasoning step based on verifier feedback, aggregating three verifier judgments by vote count with ties resolved by mean state-token log-likelihood. Additional training details are provided in Appendix C.

5.2 Main Results

VISPO-R achieves leading performance across multimodal reasoning benchmarks. As shown in Table 1, VISPO-R consistently improves its size-matched Qwen3-VL backbone across all seven

Table 1: **Performance comparison on multimodal reasoning benchmarks.** Best and second-best results are shown in **bold** and underlined. Avg. is the mean of the benchmark scores. HallBench abbreviates HallusionBench.

Model	Mathematical Reasoning				General Reasoning		Hallucination	Overall
	MathVerse	MathVista	MathVision	WeMath	MMMU	MMStar	HallBench	Avg.
Proprietary Models								
GPT-4o-20240513	50.2	63.8	30.4	63.3	69.1	64.7	55.0	56.6
Claude-3.5-Sonnet	47.8	67.7	33.5	61.7	68.3	65.1	55.5	57.1
GPT-5-mini	36.5	59.6	46.6	51.4	67.9	61.3	55.9	54.2
Open-source General Models								
Gemma-3-27B	34.0	59.8	39.8	37.9	64.9	53.7	46.9	48.1
Qwen2.5-VL-32B	52.3	74.7	40.0	55.0	70.3	69.5	57.2	59.9
Qwen2.5-VL-7B	49.2	68.2	25.1	30.9	58.6	63.9	52.9	49.8
InternVL3-8B	39.8	71.6	29.3	37.1	62.7	68.2	49.9	51.2
Qwen3-VL-4B	46.8	73.7	36.5	49.4	67.4	68.2	62.5	57.8
Qwen3-VL-8B	56.4	76.0	43.7	52.9	69.6	70.9	65.1	62.1
Multimodal Reasoning Models								
Vision-R1-7B	52.4	73.5	28.2	41.6	57.6	61.4	49.5	52.0
R1-VL-7B	52.2	74.3	28.2	42.8	52.3	66.2	57.2	53.3
R1-Onevision-7B	46.4	64.1	29.9	61.8	54.3	54.1	65.6	53.7
OpenVLThinker-7B	50.3	72.3	25.9	64.1	58.4	61.9	69.1	57.4
VAPO-Thinker-7B	53.3	75.6	31.9	43.6	60.2	63.0	57.4	55.0
VLM-CapCurriculum-8B	47.5	75.9	41.8	56.1	68.3	73.1	55.9	59.8
Ours								
VISPO-R-4B	56.7	77.4	42.1	57.8	69.5	73.5	67.9	63.6
△ *vs. Qwen3-VL-4B*	↑ **9.9**	↑ **3.7**	↑ **5.6**	↑ **8.4**	↑ **2.1**	↑ **5.3**	↑ **5.4**	↑ **5.8**
VISPO-R-8B	**63.4**	**79.6**	**48.2**	**64.8**	**73.8**	**74.6**	**72.9**	**68.2**
△ *vs. Qwen3-VL-8B*	↑ **7.0**	↑ **3.6**	↑ **4.5**	↑ **11.9**	↑ **4.2**	↑ **3.7**	↑ **7.8**	↑ **6.1**

benchmarks, increasing average accuracy by 5.8 points at 4B and 6.1 points at 8B. VISPO-R-8B achieves the best reported result among the listed models on all seven benchmarks. These consistent gains across both model scales show that VISPO-R enhances complex reasoning capabilities while maintaining hallucination robustness and reasoning reliability.

Table 2: **VISPO-synergistic inference and further gains from VISPO-R training.** Green arrows show gains over size-matched GRPO-only reasoners. Best reported results are **bolded**.

Reasoner (+ Verifier)	MathVerse	MathVista	MathVision	WeMath	MMStar	Avg.
Reasoner-Only Baselines						
Qwen3-VL-4B	46.8	73.7	36.5	49.4	68.2	54.9
Qwen3-VL-4B-GRPO	51.9	75.1	39.0	52.1	70.0	57.6
Qwen3-VL-8B	56.4	76.0	43.7	52.9	70.9	60.0
Qwen3-VL-8B-GRPO	59.7	77.8	47.5	55.8	72.7	62.7
Verifier-Synergistic Inference without Reasoner Training						
Qwen3-VL-4B + Qwen3-VL-4B	47.3	73.9	37.2	50.0	68.5	55.4
Qwen3-VL-4B + VISPO-4B	53.2	75.8	39.8	55.1	71.2	59.0
Qwen3-VL-8B + Qwen3-VL-8B	57.1	76.6	44.0	54.0	71.7	60.7
Qwen3-VL-8B + VISPO-8B	60.5	78.5	47.0	61.1	73.4	64.1
Frozen VISPO Verifier with Reasoner Training						
Qwen3-VL-4B-GRPO + VISPO-4B	55.1	76.7	41.1	56.6	72.5	60.4
Qwen3-VL-8B-GRPO + VISPO-8B	62.3	79.1	47.9	62.7	73.9	65.2
VISPO-R-4B (Ours)	56.7	77.4	42.1	57.8	73.5	61.5 ↑ **3.9**
VISPO-R-8B (Ours)	**63.4**	**79.6**	**48.2**	**64.8**	**74.6**	**66.1** ↑ **3.4**

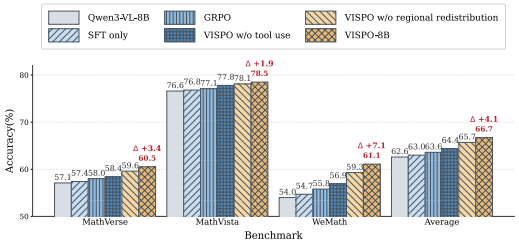


Figure 3: VISPO-8B verifier training ablations.

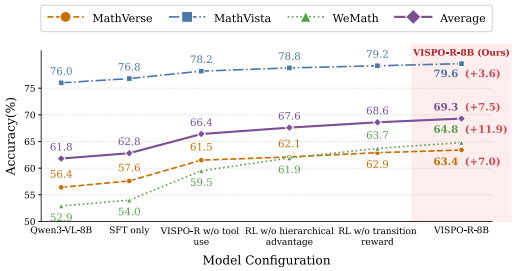


Figure 4: VISPO-R-8B component ablations.

VISPO-R further improves upon VISPO-synergistic reasoning. Table 2 shows that replacing same-backbone self-verification with VISPO improves five-benchmark average accuracy by 3.6 and 3.4 points at 4B and 8B, with the reasoner fixed. VISPO-R further outperforms GRPO-trained reasoners using the same VISPO verifier by 1.1 and 0.9 points. These results demonstrate additional gains from VISPO-R training beyond verifier-synergistic inference.

VISPO improves reasoning with reliable verification. Table 3 compares reward models paired with a fixed Qwen3-VL-8B reasoner using respective integration strategies. VISPO consistently improves performance across all four benchmarks: VISPO-4B achieves a 63.1 average, while VISPO-8B reaches 64.8 and outperforms VisualPRM-8B, MM-PRM-8B, and ARM-Thinker-7B by 3.9, 3.3, and 2.7 points. These gains support the effectiveness of reliable verification under a fixed reasoner.

Table 3: **Comparison of reward models with a fixed Qwen3-VL-8B reasoner.** Green arrows show percentage-point gains over size-matched Qwen3-VL verifiers. Best results are **bolded**.

Verifier	MathVista	MathVision	WeMath	MMMU	Avg.
Qwen3-VL-4B	76.1	43.9	53.1	69.7	60.7
Qwen3-VL-8B	76.6	44.0	54.0	70.3	61.2
VisualPRM-8B	76.3	43.9	53.5	69.8	60.9
MM-PRM-8B	76.5	44.1	55.4	70.1	61.5
ARM-Thinker-7B	77.0	44.6	56.2	70.4	62.1
VISPO-4B (Ours)	77.8	45.3	58.2	71.0	63.1
△ *vs. Qwen3-VL-4B*	↑ **1.7**	↑ **1.4**	↑ **5.1**	↑ **1.3**	↑ **2.4**
VISPO-8B (Ours)	**78.5**	**47.0**	**61.1**	**72.6**	**64.8**
△ *vs. Qwen3-VL-8B*	↑ **1.9**	↑ **3.0**	↑ **7.1**	↑ **2.3**	↑ **3.6**

5.3 Analysis

Regional credit redistribution and tool augmentation enhance verifier training. Using a fixed Qwen3-VL-8B reasoner, Figure 3 reports average accuracy of 62.6 (self-verification), 63.0 (SFT), 63.6 (GRPO) and 66.7 (VISPO-8B). Removing tool use or regional redistribution reduces it to 64.4 and 65.7, supporting the contribution of regional credit redistribution and tool augmentation.

Hierarchical advantage assignment and tool use strengthen VISPO-R. Figure 4 shows that full VISPO-R-8B leads on MathVerse, MathVista and WeMath, averaging 69.3 compared with 61.8 for the backbone and 62.8 after SFT. The *VISPO-R w/o tool use* variant averages 66.4, while RL without hierarchical advantages or the transition reward achieves 67.6 and 68.6, respectively. These component ablations verify that the VISPO-R framework benefits from the joint integration of tool use, hierarchical advantage assignment, and transition reward, with each component contributing beyond the corresponding reduced variants.

VISPO improves accuracy with more concise outputs. Figure 5 shows that VISPO's reward design achieves the largest average accuracy gain over Qwen3-VL-8B (+4.9 points), while removing any reward component weakens performance. Figure 6 further shows that VISPO improves accuracy while shortening its outputs. On MathVerse, VISPO reduces mean output length by approximately 14% relative to the no-tool variant, averaging over all recorded training steps. These results suggest that tool-augmented verification supports more concise outputs without sacrificing accuracy.

Test-time verifier sampling. With the VISPO-R-8B reasoner frozen, we aggregate sampled verifier judgments by vote count. Table 4 shows that three samples improve accuracy by up to 1.1

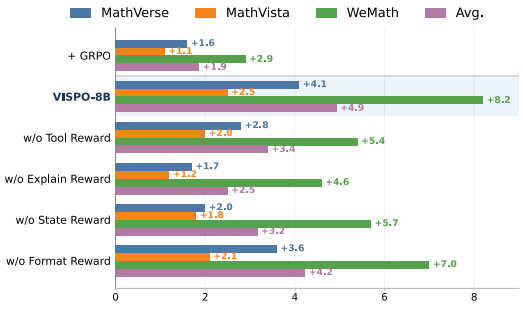


Figure 5: VISPO verifier reward-design ablation. All values are absolute accuracy gains over the reasoner-only Qwen3-VL-8B backbone.

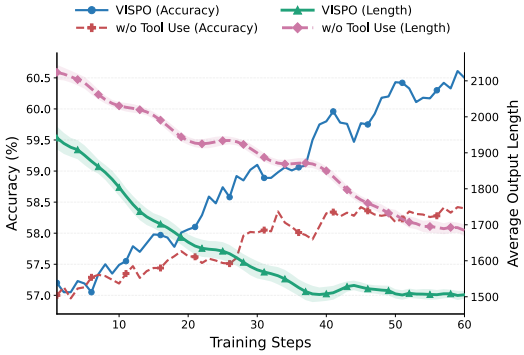


Figure 6: Accuracy and output length on MathVerse during verifier training.

Table 4: **Effect of test-time verifier sampling.** Δ denotes the accuracy change relative to one verifier sample. Avg. time denotes the average end-to-end inference time per example.

Verifier samples	MathVerse		MathVision		MMStar		WeMath		Avg. time (s)
	Acc.	$\Delta\uparrow$	*Acc.*	$\Delta\uparrow$	*Acc.*	$\Delta\uparrow$	*Acc.*	$\Delta\uparrow$	
1	62.6	0.0	47.7	0.0	73.9	0.0	63.7	0.0	5.7
3 (default)	63.4	0.8	48.2	0.5	74.6	0.7	64.8	1.1	15.9
5	63.6	1.0	48.3	0.6	74.7	0.8	65.0	1.3	28.6

percentage points over one sample across all four benchmarks. Increasing to five samples adds only up to 0.2 percentage points while raising end-to-end inference time from 15.9 to 28.6 seconds per example. We use three samples for the main results, balancing accuracy gains against inference cost.

Evaluation on VL-RewardBench. Figure 7 evaluates verifier quality on VL-RewardBench. VISPO-8B achieves the highest macro-average accuracy among the compared models at 65.6, exceeding Qwen3-VL-8B by 11.4 points. VISPO-8B scores 63.7 on reasoning judgment and 76.8 on hallucination detection, supporting reliable claim verification and reasoning assessment. These direct judgments complement downstream accuracy by assessing verification reliability rather than final-answer correctness alone. Together with the downstream HallusionBench gains, these results are consistent with the goal of improving multimodal reasoning reliability through verification feedback.

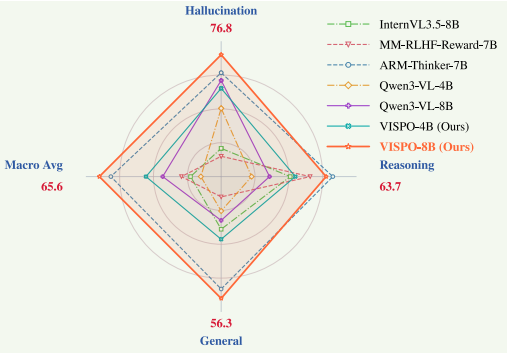


Figure 7: Comparison of multimodal models on VL-RewardBench.

6 Conclusion

We introduce VISPO, which trains an agentic verifier to perform evidence-grounded three-state verification with region-level advantage redistribution. VISPO-R leverages iterative verifier feedback via verification-aware hierarchical advantage assignment for reasoning refinement. VISPO-R improves Qwen3-VL backbones across seven benchmarks at both model scales and yields further gains over GRPO-trained reasoners paired with the same verifier. Component ablations support tool use and verification-aware optimization, while verifier evaluation, revision diagnostics and HallusionBench gains support more reliable verification and improved hallucination robustness. Future work will target lower tool dependence and verification latency, alongside broader applications.

AI USE STATEMENT

Generative AI tools were used for the tasks requiring disclosure: generating synthetic samples for the FGMV dataset and assisting with the interpretation of the reported results. Generated samples were manually reviewed and validated for consistency. Generative AI was used to assist with interpreting the results, while all conclusions were independently verified and finalized by the authors. For recommended-disclosure tasks, generative AI was used to assist with code refinement and literature search. AI-assisted code was manually inspected and tested. Retrieved literature was independently verified against the original sources. We take responsibility for the final content of this work, including claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work develops verification-aware reinforcement learning methods for multimodal large language models and does not involve human subjects, user studies, or the collection of personally identifiable information. We use publicly available datasets for training and evaluation, and the synthetic samples in the FGMV dataset are generated with the assistance of generative models, followed by filtering and manual review to improve quality. While VISPO and VISPO-R aim to enhance multimodal reasoning reliability and reduce hallucination propagation, their outputs may still inherit biases, inaccuracies, or limitations from underlying models and training data. Therefore, verification signals and reasoning results should be independently reviewed before use in high-stakes applications.

REPRODUCIBILITY STATEMENT

Appendix A details the VISPO and VISPO-R objectives, derivations, training algorithms and reward details. Appendix B provides evaluation details, additional comparisons, ablations and sensitivity analyses. Appendix C documents data construction, tool interfaces, hyperparameters, verifier aggregation and stopping rules. Appendix D presents the verifier, explanation-judge and reasoner prompts, together with representative case studies.

REFERENCES

Alibaba Cloud. qwen3.7-plus. Alibaba Cloud Model Studio Documentation, 2026. URL https://help.aliyun.com/en/model-studio/qwen3-7-plus.

Anthropic. Claude 3.5 Sonnet. Anthropic, 2024. URL https://www.anthropic.com/news/claude-3-5-sonnet.

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

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37:27056–27087, 2024.

Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. OpenVLThinker: Complex vision-language reasoning via iterative SFT-RL cycles. *Advances in Neural Information Processing Systems*, 38:137212–137241, 2025.

Shengyuan Ding, Xinyu Fang, Ziyu Liu, Yuhang Zang, Yuhang Cao, Xiangyu Zhao, Haodong Duan, Xiaoyi Dong, Jianze Liang, Bin Wang, Conghui He, Dahua Lin, and Jiaqi Wang. ARM-Thinker: Reinforcing multimodal generative reward models with agentic tool use and visual reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22195–22205, 2026.

Lingxiao Du, Fanqing Meng, Zongkai Liu, Zhixiang Zhou, Ping Luo, Qiaosheng Zhang, and Wenqi Shao. MM-PRM: Enhancing multimodal mathematical reasoning with scalable step-level supervision. *arXiv preprint arXiv:2505.13427*, 2025.

Gemma Team et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. HallusionBench: An advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14375–14385, 2024.

Jinyi Han, Xinyi Wang, Haiquan Zhao, Tingyun Li, Zishang Jiang, Sihang Jiang, Jiaqing Liang, Xin Lin, Weikang Zhou, Zeye Sun, Fei Yu, and Yanghua Xiao. A stitch in time saves nine: Proactive self-refinement for language models. In *International Conference on Learning Representations*, 2026.

Wenxuan Huang, Bohan Jia, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-R1: Incentivizing reasoning capability in multimodal large language models. In *International Conference on Learning Representations*, 2026.

Bingqing Jiang and Difan Zou. Structured role-aware policy optimization for multimodal reasoning. *arXiv preprint arXiv:2605.07274*, 2026.

Peng Kuang, Xiangxiang Wang, Wentao Liu, Jian Dong, and Kaidi Xu. TIM-PRM: Verifying multimodal reasoning with tool-integrated PRM. *arXiv preprint arXiv:2511.22998*, 2025.

Lei Li, Yuancheng Wei, Zhihui Xie, Xuqing Yang, Yifan Song, Peiyi Wang, Chenxin An, Tianyu Liu, Sujian Li, Bill Yuchen Lin, Lingpeng Kong, and Qi Liu. VL-RewardBench: A challenging benchmark for vision-language generative reward models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24657–24668, 2025.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pp. 38–55, 2024.

Xiaoyuan Liu, Tian Liang, Zhiwei He, Jiahao Xu, Wenxuan Wang, Pinjia He, Zhaopeng Tu, Haitao Mi, and Dong Yu. Trust, but verify: A self-verification approach to reinforcement learning with verifiable rewards. *Advances in Neural Information Processing Systems*, 38:144657–144683, 2025.

Pan Lu, Ran Gong, Shibiao Jiang, Liang Qiu, Siyuan Huang, Xiaodan Liang, and Song-Chun Zhu. Inter-GPS: Interpretable geometry problem solving with formal language and symbolic reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pp. 6774–6786, 2021.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. MathVista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations*, 2024.

Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Daocheng Fu, Botian Shi, Wenhai Wang, Junjun He, Kaipeng Zhang, Ping Luo, Yu Qiao, Qiaosheng Zhang, and Wenqi Shao. MM-Eureka: Exploring visual aha moment with rule-based large-scale reinforcement learning. *arXiv preprint arXiv:2503.07365v1*, 2025.

Yingqian Min, Kun Zhou, Yifan Li, Yuhuan Wu, Han Peng, Yifan Du, Wayne Xin Zhao, Min Yang, and Ji-Rong Wen. Improving vision-language models with perception-centric process reward models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 33099–33109, 2026.

OpenAI. Hello GPT-4o. OpenAI, 2024. URL https://openai.com/index/hello-gpt-4o/.

OpenAI. GPT-5 system card. OpenAI, 2025. URL https://openai.com/index/gpt-5-system-card/.

OpenAI. Introducing GPT-5.5. OpenAI, 2026. URL https://openai.com/index/introducing-gpt-5-5/.

Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Jiapeng Wang, Zhuoma Gongque, Shanglin Lei, Yifan Zhang, Zhe Wei, Miaoxuan Zhang, Runfeng Qiao, Xiao Zong, Yida Xu, Peiqing Yang, Zhimin Bao, Muxi Diao, Chen Li, and Honggang Zhang. We-Math: Does your large multimodal model achieve human-like mathematical reasoning? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 20023–20070, 2025.

Xinyu Qiu, Heng Jia, Zhengwen Zeng, Shuheng Shen, Changhua Meng, Yi Yang, and Linchao Zhu. Unified generation and self-verification for vision-language models via advantage decoupled preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36916–36925, 2026.

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

Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, Ruochen Xu, and Tiancheng Zhao. VLM-R1: A stable and generalizable R1-style large vision-language model. *arXiv preprint arXiv:2504.07615*, 2025.

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

Lingzhuang Sun, Ruitong Liu, Yuxia Zhu, Xiaohan Xu, Jingxuan Wei, Xiangxiang Zhang, Bihui Yu, and Wentao Zhang. Guided verifier: Collaborative multimodal reasoning via dynamic process supervision. *arXiv preprint arXiv:2602.04290*, 2026.

Xinyu Tian, Shu Zou, Zhaoyuan Yang, Mengqi He, Fabian Waschkowski, Lukas Wesemann, Peter Tu, and Jing Zhang. More thought, less accuracy? on the dual nature of reasoning in vision-language models. In *International Conference on Learning Representations*, 2026.

Zhongwei Wan, Zhihao Dou, Che Liu, Yu Zhang, Dongfei Cui, Qinjian Zhao, Hui Shen, Jing Xiong, Yi Xin, Yifan Jiang, Yangfan He, Mi Zhang, and Shen Yan. SRPO: Enhancing multimodal LLM reasoning via reflection-aware reinforcement learning. *arXiv preprint arXiv:2506.01713*, 2025.

Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. VL-Rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning. *Advances in Neural Information Processing Systems*, 38:34852–34878, 2025a.

Junxin Wang, Dai Guan, Weijie Qiu, Zhihang Li, Yongbo Gai, Zhengyi Yang, Mengyu Zhou, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Grounding the score: Explicit visual premise verification for reliable vision-language process reward models. *arXiv preprint arXiv:2603.16253*, 2026.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with MATH-Vision dataset. *Advances in Neural Information Processing Systems*, 37:95095–95169, 2024.

Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, Lewei Lu, Haodong Duan, Yu Qiao, Jifeng Dai, and Wenhai Wang. VisualPRM: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*, 2025b.

Haoran Wei, Yaofeng Sun, and Yukun Li. DeepSeek-OCR: Contexts optical compression. *arXiv preprint arXiv:2510.18234*, 2025.

Juncheng Wu, Hardy Chen, Haoqin Tu, Xianfeng Tang, Freda Shi, Hui Liu, Hanqing Lu, Cihang Xie, and Yuyin Zhou. From seeing to thinking: Decoupling perception and reasoning improves post-training of vision-language models. In *International Conference on Machine Learning*, 2026.

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

Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, Bo Zhang, and Wei Chen. R1-Onevision: Advancing generalized multimodal reasoning through cross-modal formalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2376–2385, 2025.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9556–9567, 2024.

Jingyi Zhang, Jiaxing Huang, Huanjin Yao, Shunyu Liu, Xikun Zhang, Shijian Lu, and Dacheng Tao. R1-VL: Learning to reason with multimodal large language models via step-wise group relative policy optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1859–1869, 2025.

Kaichen Zhang, Keming Wu, Zuhao Yang, Bo Li, Kairui Hu, Bin Wang, Ziwei Liu, Xingxuan Li, and Lidong Bing. OpenMMReasoner: Pushing the frontiers in multimodal reasoning with an open and general recipe. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19276–19286, 2026.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, Peng Gao, and Hongsheng Li. MathVerse: Does your multi-modal LLM truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pp. 169–186, 2024.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 400–410, 2024.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Yuchen Duan, Hao Tian, Weijie Su, Jie Shao, et al. InternVL3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

APPENDIX CONTENTS

A Additional Details for VISPO and VISPO-R **15**

A.1 Theoretical Derivation 15

A.2 Training Algorithms 16

A.3 Reward Details 16

A.4 Verification Protocol 17

B Additional Experiments and Analysis **18**

B.1 Evaluation Data Details 18

B.2 Comparison with Perception-Centric Process Reward Model 19

B.3 Comparison with Structured Role-Aware Policy Optimization 20

B.4 Comparison with Multimodal Process Reward Models 20

B.5 Comparison with Parallel Self-Verification Optimization 21

B.6 Comparison with Reflection-based Reinforcement Learning Methods 22

B.7 VISPO-R Further Improves upon VISPO-Synergistic Reasoning on MathVerse . . 22

B.8 Reward Evolution during RL Training 23

B.9 Binary versus Three-State Verification at the 4B and 8B Scales 24

B.10 Accuracy and Revision Diagnostics 24

B.11 VISPO Consistently Outperforms Self-Verification 25

B.12 Regional Credit Redistribution and Tool Augmentation Enhance VISPO 25

B.13 Component and Reasoner-Training Ablations 25

B.14 Visualizing Regional Credit Redistribution 26

B.15 Parameter Sensitivity Analysis 26

C Data and Implementation Details **27**

C.1 FGMV Dataset Details 27

C.2 Reasoner Data Sources and Separation from FGMV 28

C.3 Executable Tools via Function Calling 29

C.4 Hyperparameters 30

C.5 Verifier Aggregation and Stopping Rules 31

D Prompt Templates and Case Studies **31**

D.1 Prompt Templates 31

D.2 Case Studies 35

A Additional Details for VISPO and VISPO-R

A.1 Theoretical Derivation

VISPO Region-Level Credit Redistribution. For $s \in S_{\mathrm{V}}$, define the regional quality score as

$$Q_{i,s}^{\mathrm{V}} = \begin{cases} R_{i,\mathrm{state}}, & s = \texttt{state}, \\ R_{i,\mathrm{explain}}, & s = \texttt{explain}, \\ R_{i,\mathrm{tool}}, & s = \texttt{tool_call}, \\ \frac{R_{i,\mathrm{state}}+R_{i,\mathrm{explain}}+R_{i,\mathrm{tool}}}{3}, & s = \texttt{analysis}, \end{cases} \tag{16}$$

where $S_{\mathrm{V}} = \{\texttt{analysis}, \texttt{tool_call}, \texttt{state}, \texttt{explain}\}$ and all component scores are clipped to $[0, 1]$. The sign-aware regional weight is

$$w_{i,s} = \begin{cases} (Q_{i,s}^{\mathrm{V}})^{\eta_+}, & \widehat{A}_i > 0, \\ 1 - \lambda_- Q_{i,s}^{\mathrm{V}}, & \widehat{A}_i < 0, \\ 0, & \widehat{A}_i = 0, \end{cases} \tag{17}$$

where default exponent $\eta_+ = 2$ and discount coefficient $\lambda_- = 0.5$, giving negative-advantage weights in $[0.5, 1]$. We set $\alpha_{\texttt{state}} = \alpha_{\texttt{explain}} = 1.5$ and $\alpha_{\texttt{analysis}} = \alpha_{\texttt{tool_call}} = 1$. Empty regions contribute zero through their lengths and masks. If $\sum_s \alpha_s w_{i,s} n_{i,s} = 0$, we use the uniform actor-token distribution $p_{i,t} = \frac{m_{i,t}}{L_i}$. When $\widehat{A}_i = 0$, all regional advantages are set to zero. The uniform distribution is a safe normalization fallback.

Credit Conservation. Since $\sum_t p_{i,t} = 1$, the token-level verifier advantage $A_{i,t}^{\mathrm{V}} = \widehat{A}_i p_{i,t}$ satisfies

$$\sum_t A_{i,t}^{\mathrm{V}} = \widehat{A}_i \sum_t p_{i,t} = \widehat{A}_i. \tag{18}$$

Redistribution therefore changes only where credit is assigned, preserving the rollout-level advantage's total value and sign.

Length Regularization. For actor length L_i, maximum length $L_{\max}$, buffer width $\delta_{\mathrm{len}} > 0$ and coefficient $\lambda_{\mathrm{len}} \geq 0$, the length penalty is

$$R_{i,\mathrm{len}} = \min\left\{-\lambda_{\mathrm{len}} \frac{L_i - (L_{\max} - \delta_{\mathrm{len}})}{\delta_{\mathrm{len}}},\, 0\right\}. \tag{19}$$

The penalty is zero below $L_{\max} - \delta_{\mathrm{len}}$ and decreases linearly to $-\lambda_{\mathrm{len}}$ at $L_{\max}$. Only trainable actor tokens count toward L_i; tool observations are excluded.

VISPO-R Optimization Details. The state transition reward evaluates trajectory-level verification progress during verifier-synergistic reasoning rather than directly assigning credit to individual reasoning steps. It measures the final improvement of verification states while subtracting accumulated regressions along the trajectory, thereby discouraging reasoning paths that temporarily degrade before recovering. The length normalization reduces the bias toward unnecessarily long verification trajectories. The resulting transition signal captures both the final verification outcome and intermediate state evolution, which is subsequently combined with verification-aware hierarchical advantage assignment to attribute trajectory-level improvements to individual reasoning steps.

Using the combined advantages from equation 15, VISPO-R optimizes only reasoner-generated tokens. For actor token $y_{i,t}^{\mathrm{R}}$ and its generation context $H_{i,t}^{\mathrm{R}}$, define

$$r_{i,t}^{\mathrm{R}}(\theta) = \frac{\pi_\theta(y_{i,t}^{\mathrm{R}} \mid H_{i,t}^{\mathrm{R}})}{\pi_{\theta_{\mathrm{old}}}(y_{i,t}^{\mathrm{R}} \mid H_{i,t}^{\mathrm{R}})}, \tag{20}$$

Use $\bar{r}_{i,t}^{\mathrm{R}} = \mathrm{clip}(r_{i,t}^{\mathrm{R}}, 1 - \varepsilon_{\mathrm{R}}, 1 + \varepsilon_{\mathrm{R}})$ with $\varepsilon_{\mathrm{R}} = 0.2$. We maximize

$$\mathcal{L}^{\text{VISPO-R}}(\theta) = \mathbb{E}\Bigg[\frac{1}{G}\sum_{i=1}^{G}\Big\{\frac{1}{L_i^{\mathrm{R}}}\sum_{t \in \mathcal{I}_i^{\mathrm{R}}} \min\big[r_{i,t}^{\mathrm{R}}(\theta)A_{i,t}^{\mathrm{R}}, \bar{r}_{i,t}^{\mathrm{R}}(\theta)A_{i,t}^{\mathrm{R}}\big] - \beta D_{\mathrm{KL}}^{\mathrm{R}}(\pi_\theta \| \pi_{\mathrm{ref}})\Big\}\Bigg]. \tag{21}$$

The disjoint step-wise actor-token sets form $\mathcal{I}_i^{\mathrm{R}}$, with $L_i^{\mathrm{R}} = |\mathcal{I}_i^{\mathrm{R}}| > 0$. The expectation is over inputs from D_{R} and G rollouts per input sampled under $\pi_{\theta_{\mathrm{old}}}$ with frozen-verifier feedback and real tool observations.

During VISPO-R training, the verifier provides a single verification feedback signal for each reasoning step to optimize the reasoner policy. During inference, VISPO aggregates multiple independently sampled verifier judgments through majority voting to improve verification robustness.

A.2 Training Algorithms

Algorithms 1 and 2 summarize the training stages. Stage I trains the state-aware VISPO verifier with regional token advantages; Stage II freezes it to train the reasoner with VISPO-R. We implement the SFT stages using LlamaFactory (Zheng et al., 2024) and the RL stages using VeRL (Sheng et al., 2025).

Algorithm 1 Training the agentic verifier with VISPO

Initialize: Verifier data $\mathcal{D}_{\mathrm{V}}$, group size G, tools $\mathcal{T}$ and budget K
  Verifier V_ψ from SFT; fixed reference V_{ref}
for update iteration $\ell = 1, \ldots, U_{\mathrm{V}}$ **do**
  Set $\psi_{\mathrm{old}} \leftarrow \psi$, sample an input batch $\mathcal{B}_{\mathrm{V}} \subset \mathcal{D}_{\mathrm{V}}$ and clear rollout buffer
  for each verifier instance $x \in \mathcal{B}_{\mathrm{V}}$ **do**
    for $i = 1, \ldots, G$ **do**
      Initialize verifier trajectory $y_i \leftarrow \varnothing$
      for $k = 1, \ldots, K$ **do**
        Sample the output from $V_{\psi_{\mathrm{old}}}$ and append it to y_i
        if the output is a valid judgment $g_i = (z_i, e_i)$ **then**
          Record g_i and terminate the rollout
        else if the output contains one valid tool call **then**
          Execute the tool and append the real observation to y_i
        else
          Mark the rollout as invalid and terminate the current rollout
        end if
      end for
    end for
    For all $i = 1, \ldots, G$, select tokens $\mathcal{I}_i^{\mathrm{V}} \leftarrow \{t : m_{i,t} = 1\}$ and lengths $L_i^{\mathrm{V}} \leftarrow |\mathcal{I}_i^{\mathrm{V}}|$
    Compute $R_i^{\mathrm{V}} \leftarrow \min\{\bar{R}_i^{\mathrm{V}}, R_{i,\mathrm{c}}\} + R_{i,\mathrm{len}}$ using actor length L_i^{V}
    Group-normalize R_i^{V} to $\widehat{A}_i$; compute regional token allocation $p_{i,t}$
    Set $A_{i,t}^{\mathrm{V}} \leftarrow \widehat{A}_i p_{i,t}$; store rollouts, masks, and old log-probabilities
  end for
  for each minibatch from the collected rollout buffer **do**
    Compute current-policy ratios $r_{i,t}^{\mathrm{V}}$, $\bar{r}_{i,t}^{\mathrm{V}}$ for $t \in \mathcal{I}_i^{\mathrm{V}}$
    Update $\psi \leftarrow \arg\max_{\psi} \frac{1}{G} \sum_{i=1}^{G} \Big\{ \sum_{t \in \mathcal{I}_i^{\mathrm{V}}} \min\big(r_{i,t}^{\mathrm{V}} A_{i,t}^{\mathrm{V}},\, \bar{r}_{i,t}^{\mathrm{V}} A_{i,t}^{\mathrm{V}}\big) - \beta D_{\mathrm{KL}}^{\mathrm{V}}(V_\psi \| V_{\mathrm{ref}}) \Big\}$
  end for
end for
Output: VISPO-trained verifier V_ψ

A.3 Reward Details

Format and Tool Rewards. The format reward R_{fmt} evaluates the validity of the protocol, including tag completeness, closure, non-empty contents, and ordering.

$$R_{\mathrm{fmt}} = \begin{cases} 1, & \text{valid format,} \\ \hat{R}_{\mathrm{fmt}}, & \text{otherwise.} \end{cases} \tag{22}$$

where $\hat{R}_{\mathrm{fmt}}$ is a dense score based on tag presence, closure, content validity, and ordering.

Algorithm 2 Training the VISPO-R reasoner with hierarchical advantages

Initialize: Data $\mathcal{D}_{\mathrm{R}}$, group size G, tools $\mathcal{T}$, budget $J_{\max}$ and α
  Reasoner π_θ from SFT; freeze verifier V_ψ and reference π_{ref}; contexts T^{R}, T^{V}
for update iteration $\ell = 1, \ldots, U_{\mathrm{R}}$ **do**
  Set $\theta_{\mathrm{old}} \leftarrow \theta$, sample $\mathcal{B}_{\mathrm{R}} \subset \mathcal{D}_{\mathrm{R}}$ and clear rollout buffer
  for each reasoner instance $x \in \mathcal{B}_{\mathrm{R}}$ **do**
    for $i = 1, \ldots, G$ **do**
      Initialize reasoner trajectory $\tau_i \leftarrow \varnothing$ and feedback $g_{i,0} \leftarrow \varnothing$
      for $j = 1, \ldots, J_{\max}$ **do**
        Construct reasoner context $T^{\mathrm{R}}_{i,j}$ with x, τ_i, $g_{i,j-1}$
        Sample the output from $\pi_{\theta_{\mathrm{old}}}(\cdot \mid T^{\mathrm{R}}_{i,j})$
        if the output contains a valid tool call **then**
          Execute it, obtain $o^{\mathrm{R}}_{i,j}$ and generate the reasoning step
        end if
        Record actor mask and append $d_{i,j}$ to τ_i
        Construct verifier context $T^{\mathrm{V}}_{i,j}$ with x, τ_i, $d_{i,j}$
        Verify $d_{i,j}$ using $V_\psi(\cdot \mid T^{\mathrm{V}}_{i,j})$ and tools; record $g_{i,j} \leftarrow (z_{i,j}, e_{i,j})$
        if $z_{i,j} =$ SUPPORT **then**
          Terminate the current rollout
        end if
      end for
    end for
    For all $i = 1, \ldots, G$, compute $\bar{R}^{\mathrm{R}}_i$ and set $R^{\mathrm{R}}_i \leftarrow \min\{\bar{R}^{\mathrm{R}}_i, R_{i,\mathrm{c}}\} + R_{i,\mathrm{len}}$
    Group-normalize R^{R}_i to A^{traj}_i; compute local advantages $A^{\mathrm{loc}}_{i,j}$
    Set $A^{\mathrm{R}}_{i,t} \leftarrow \alpha A^{\mathrm{traj}}_i + (1-\alpha) A^{\mathrm{loc}}_{i,j(t)}$; store rollouts, masks, and old log-probabilities
  end for
  for each minibatch from the collected rollout buffer **do**
    Compute current-policy ratios $r^{\mathrm{R}}_{i,t}$, $\bar{r}^{\mathrm{R}}_{i,t}$ for $t \in \mathcal{I}^{\mathrm{R}}_i$
    Update $\theta \leftarrow \arg\max_\theta \frac{1}{G} \sum_{i=1}^{G} \Big\{ \frac{1}{L^{\mathrm{R}}_i} \sum_{t \in \mathcal{I}^{\mathrm{R}}_i} \min\big(r^{\mathrm{R}}_{i,t} A^{\mathrm{R}}_{i,t}, \bar{r}^{\mathrm{R}}_{i,t} A^{\mathrm{R}}_{i,t}\big) - \beta D^{\mathrm{R}}_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}}) \Big\}$
  end for
end for
Output: Reasoner π_θ with frozen verifier V_ψ

The tool reward R_{tool} evaluates executable tool usage:

$$R_{\mathrm{tool}} = R_{\mathrm{raw}} \left(0.6 + 0.4 R_{\mathrm{acc}}\right), \tag{23}$$

$$R_{\mathrm{raw}} = 0.15 s_{\mathrm{tag}} + 0.30 s_{\mathrm{valid}} + 0.20 s_{\mathrm{json}} + 0.20 s_{\mathrm{pair}} + 0.15 s_{\mathrm{resp}}, \tag{24}$$

where s_{tag}, s_{valid}, s_{json}, s_{pair}, and s_{resp} denote tool-tag validity, schema validity, JSON validity, call-response matching, and non-empty response, respectively.

Reward Cap. Before applying the length penalty, we apply severity-aware reward caps: severe violations (e.g., unrecognizable formats and pathological repetitions) are capped at 0.4, while other protocol violations (e.g., tag/order errors, invalid tool calls, and failed responses) are capped at 0.7. These caps mitigate reward hacking by limiting rewards from invalid trajectories and encourage evidence-grounded verification.

A.4 VERIFICATION PROTOCOL

To enable structured, state-aware verification, we define an XML-based verification protocol that explicitly separates verifier reasoning, tool interaction and final judgment. During each verification round, the verifier first analyzes the current reasoner step and may execute an appropriate tool to acquire external evidence. Specifically, the protocol contains five types of tags:

- **Analysis (`<analysis>`):** Analyzes the current reasoning step and identifies the concrete evidence gap that should be verified.
- **Tool Call (`<tool_call>`):** Executes one available tool with valid arguments to acquire evidence for the identified verification target.
- **Tool Response (`<tool_response>`):** Records the real evidence returned by the executed tool and serves as external context for subsequent verification.
- **Verification State (`<state>`):** Jointly labels answer correctness and reasoning-step validity as SUPPORT, INSUFFICIENT or CONTRADICT.
- **Explanation (`<explain>`):** Interprets the acquired evidence, identifies errors or hallucinations when present and provides corrective guidance to the reasoner.

B ADDITIONAL EXPERIMENTS AND ANALYSIS

This appendix complements Section 5 with evaluation-dataset details, downstream comparisons, component ablations and sensitivity analysis.

B.1 EVALUATION DATA DETAILS

The seven downstream benchmarks in the main evaluation assess final-answer performance. V*Bench provides an additional evaluation of fine-grained visual grounding, whereas VL-RewardBench and VisualProcessBench separately evaluate reward models. The evaluation datasets and splits are summarized below:

- **MathVerse** (testmini: 3,940 samples) evaluates whether multimodal models genuinely understand visual diagrams through mathematical problems presented with varying amounts of textual and visual information (Zhang et al., 2024).
- **MathVista** (testmini: 1,000 samples) combines diverse mathematical and visual tasks to assess fine-grained visual understanding and compositional mathematical reasoning (Lu et al., 2024).
- **MathVision** (test: 3,040 samples) comprises challenging visual mathematics problems collected from real mathematics competitions, spanning diverse mathematical disciplines and difficulty levels (Wang et al., 2024).
- **WeMath** (testmini: 1,740 samples) organizes visual mathematics problems around hierarchical knowledge concepts and multiple levels of knowledge granularity to examine knowledge acquisition and generalization in multimodal reasoning (Qiao et al., 2025).
- **MMMU** (val: 900 samples) evaluates expert-level multimodal understanding and reasoning across six disciplines and 30 subjects, combining heterogeneous image types with questions that require subject-specific knowledge (Yue et al., 2024).
- **MMStar** (test: 1,500 samples) consists of carefully curated, vision-indispensable samples designed to evaluate core multimodal capabilities while mitigating text-only shortcuts and potential data leakage (Chen et al., 2024).
- **HallusionBench** (test: 1,129 samples) evaluates image-context reasoning and diagnoses failure modes arising from language hallucination and visual illusion, while also examining the logical consistency of model responses (Guan et al., 2024).
- **VL-RewardBench** (test: 1,247 samples) evaluates generative reward models across general multimodal queries, visual hallucination detection and complex reasoning tasks (Li et al., 2025).
- **V*Bench** (test: 191 samples) evaluates fine-grained visual search and grounding in high-resolution images through attribute recognition and spatial-relationship reasoning tasks (Wu & Xie, 2024).
- **VisualProcessBench** (test: 2,866 samples; 26,950 annotated steps) evaluates the ability of process reward models and multimodal models to identify erroneous steps in multimodal reasoning trajectories drawn from diverse visual reasoning benchmarks (Wang et al., 2025b).

Evaluation Metrics. We use the official evaluation protocol for each benchmark and report all scores as percentages. For MathVerse, MathVista, MathVision, MMMU, MMStar, WeMath and V*Bench, we report final-answer accuracy. For HallusionBench, we report the official accuracy metric (aAcc) computed by its standard evaluation implementation. VL-RewardBench directly evaluates reward models; we therefore report pairwise preference accuracy, defined as the proportion of response pairs for which the verifier's preference agrees with the annotated preference, and use the macro-average across its evaluation categories. For VisualProcessBench, we report Macro-F1 over step-level correctness predictions.

Macro-Averaged Relative Improvements. The relative improvements reported in the abstract and introduction are computed from Table 1 using evaluation-specific notation. Let $\mathcal{E}_{\text{scale}} = \{4\text{B}, 8\text{B}\}$ denote the evaluated model sizes and $\mathcal{E}_{\text{bench}}^{(\text{ct})}$ the benchmarks in evaluation category ct. For size ξ and benchmark ζ, let $\text{Acc}_{\xi,\zeta}^{\text{base}}$ and $\text{Acc}_{\xi,\zeta}^{\text{VISPO-R}}$ denote the accuracies of Qwen3-VL and VISPO-R, respectively. We average the relative gains over all benchmark and model-size combinations with equal weight:

$$\text{RelGain}^{(\text{ct})} = \frac{100\%}{|\mathcal{E}_{\text{scale}}|\,|\mathcal{E}_{\text{bench}}^{(\text{ct})}|} \sum_{\xi \in \mathcal{E}_{\text{scale}}} \sum_{\zeta \in \mathcal{E}_{\text{bench}}^{(\text{ct})}} \frac{\text{Acc}_{\xi,\zeta}^{\text{VISPO-R}} - \text{Acc}_{\xi,\zeta}^{\text{base}}}{\text{Acc}_{\xi,\zeta}^{\text{base}}}. \tag{25}$$

The mathematical reasoning category includes MathVerse, MathVista, MathVision and WeMath; general reasoning includes MMMU and MMStar; hallucination evaluation includes HallusionBench. Averaging over the resulting eight, four and two benchmark and model-size combinations yields relative improvements of 13.6%, 5.5% and 10.3%, respectively, rounded to one decimal place after aggregation. These are macro-averages of individual relative gains, not relative changes in category-average accuracy or absolute percentage-point gains. VL-RewardBench is evaluated separately and is not included in these category averages.

B.2 Comparison with Perception-Centric Process Reward Model

Table 5: **Comparison with perception-centric process reward model.** Avg. denotes the average accuracy over four shared benchmarks. Green arrows report absolute average improvements over Perceval-7B.

Model	MathVision	MathVista	MMStar	V*	Avg.
Qwen2.5-VL-7B	25.1	68.2	63.9	62.1	54.8
Qwen3-VL-4B	36.5	73.7	68.2	75.8	63.6
Qwen3-VL-8B	43.7	76.0	70.9	82.4	68.3
Perceval-7B	30.9	72.0	63.8	86.4	63.3
Qwen3-VL-4B + VISPO-4B	39.8	75.8	71.2	83.6	67.6 ↑ **4.3**
Qwen3-VL-8B + VISPO-8B	**47.0**	**78.5**	**73.4**	**88.5**	**71.9** ↑ **8.6**

VISPO achieves consistent improvements over the perception-centric process reward model across four multimodal reasoning benchmarks. Compared with Perceval-7B (Min et al., 2026), VISPO-8B improves the average accuracy from 63.3% to 71.9%, yielding an absolute gain of 8.6 points.

Perceval grounds visual claims in image evidence and applies targeted token-level penalties to hallucinated spans, providing fine-grained supervision for perception-related errors. Although this localized feedback can guide response correction, its signal remains centered on detecting errors in an already generated trajectory. VISPO instead formulates verification as an agentic decision process: the verifier actively acquires external evidence through tools, and learns the evidence-seeking and judgment policy through verification-state-aware optimization. Its sign-aware regional credit redistribution further assigns differentiated token-level credit to analysis, tool calls, verification states and explanations, thereby optimizing not only the final judgment but also the process used to reach it. Building on this stronger verifier, VISPO-R models verification-state evolution across reasoning steps and combines trajectory-level progress with step-level feedback through hierarchical advantage assignment. This design penalizes regressions, identifies which reasoning transitions contribute

to improvement and supplies structured, evidence-grounded supervision for optimizing multimodal reasoning reliability.

B.3 Comparison with Structured Role-Aware Policy Optimization

Table 6: **Comparison with SRPO on mathematical reasoning benchmarks.** SRPO refers to the structured role-aware reinforcement learning framework (Jiang & Zou, 2026). Avg. denotes the average accuracy over four shared benchmarks. Mini refers to a 304-example subset. VO refers to a 788-example vision-only subset. Green arrows report absolute average improvements over SRPO-7B.

Model	MathVista	MathVision$_{\text{mini}}$	WeMath	MathVerse$_{\text{VO}}$	Avg.
Qwen2.5-VL-7B	68.2	25.7	30.9	38.3	40.8
Qwen3-VL-4B	73.7	36.8	49.4	44.3	51.1
Qwen3-VL-8B	76.0	44.1	52.9	54.2	56.8
SRPO-7B	76.3	34.5	46.9	44.9	50.7
Qwen3-VL-4B + VISPO-4B	75.8	40.6	55.1	48.5	55.0
Qwen3-VL-8B + VISPO-8B	78.5	48.2	61.1	57.5	61.3
VISPO-R-4B	77.4	42.9	57.8	53.2	57.8 ↑ **7.1**
VISPO-R-8B	**79.6**	**49.5**	**64.8**	**60.3**	**63.6** ↑ **12.9**

VISPO achieves consistent improvements over SRPO (Jiang & Zou, 2026) across four mathematical reasoning benchmarks. VISPO-8B improves the average accuracy from 50.7% to 61.3%, with substantial gains on MathVision$_{\text{mini}}$, WeMath, and MathVerse$_{\text{VO}}$. Building upon VISPO, VISPO-R further increases the average accuracy to 63.6%, demonstrating the effectiveness of verifier feedback and verification-aware credit assignment for reasoning improvement. Although SRPO introduces structured role-aware optimization for multimodal reasoning, the gains demonstrate the effectiveness of explicit verification-state optimization in improving reasoning reliability.

B.4 Comparison with Multimodal Process Reward Models

Existing multimodal process reward models improve reasoning by evaluating intermediate steps with visual evidence. VisualPRM (Wang et al., 2025b) and EVPV-PRM (Wang et al., 2026) learn process-level evaluation signals, while TIM-PRM (Kuang et al., 2025) additionally incorporates tool-integrated evidence acquisition. VISPO goes beyond assigning a score to each step: it formulates process verification as an agentic decision process and optimizes the verifier's evidence-seeking and judgment policy end to end. Through verification-state-aware policy optimization, the verifier learns when to acquire additional evidence, enabling more adaptive and informative process verification.

Table 7 compares VISPO with representative multimodal process reward models on VisualProcess-Bench (Wang et al., 2025b). For evaluation, verifier outputs are converted into binary correctness labels: SUPPORT and INSUFFICIENT are regarded as correct, whereas CONTRADICT is regarded as incorrect for computing Macro-F1. Results marked with * are reported from TIM-PRM (Kuang et al., 2025). VISPO-8B ranks first on every subset and achieves the best overall Macro-F1 of 69.6%. It outperforms the strongest prior process reward model, EVPV-PRM-8B, by 2.1 percentage points overall, and exceeds TIM-PRM-8B by 7.9 points. Notably, even VISPO-4B surpasses TIM-PRM-8B by 3.1 points despite using a smaller verifier backbone with fewer parameters. These results highlight the effectiveness and parameter efficiency of learning an active, verification-state-aware policy.

Beyond process evaluation, VISPO provides structured verification signals that can be directly optimized for reasoning improvement. Building upon the frozen VISPO verifier, VISPO-R further incorporates verification-aware hierarchical advantage assignment, transferring verifier feedback from step-level assessment to multimodal reasoning optimization.

Table 7: **Comparison with multimodal process reward models on VisualProcessBench.** The results of models marked with * are cited from TIM-PRM (Kuang et al., 2025). All results are reported in Macro-F1 (%).

Model	MMMU	MathVision	$\text{MathVerse}_{\text{VO}}$	DynaMath	WeMath	Overall
GPT-4o-mini*	53.6	58.9	57.1	56.7	58.5	57.9
GPT-4o*	56.3	60.2	59.7	59.0	63.3	60.3
Qwen2.5-VL-72B*	59.2	59.0	59.7	62.9	62.3	60.5
InternVL2.5-78B*	52.0	51.7	53.7	50.8	52.5	52.6
Qwen3-VL-8B*	56.6	61.7	59.9	61.4	62.7	61.1
MM-PRM-8B*	51.2	55.4	54.9	58.1	56.5	55.5
TIM-PRM-8B*	58.3	58.3	61.9	65.9	63.9	61.7
VisualPRM-8B	58.5	62.1	61.0	62.7	61.8	62.0
EVPV-PRM-8B	68.9	65.3	67.1	69.6	69.1	67.5
VISPO-4B	64.7	63.5	63.6	66.3	65.1	64.8
VISPO-8B	**69.8**	**67.4**	**69.2**	**71.5**	**72.3**	**69.6**

B.5 Comparison with Parallel Self-Verification Optimization

Table 8: **Comparison with ADPO on multimodal reasoning benchmarks.** ADPO adopts parallel best-of-N self-verification that requires multiple candidate trajectories during inference, while VISPO performs three-sample verifier voting for each reasoning step and VISPO-R conducts verifier-synergistic reasoning with a frozen VISPO verifier using a single reasoning trajectory. Green arrows denote percentage-point differences from the published ADPO-7B (Sample 12) results.

Method	MathVista	MMMU						
	Overall	ARD	BUS	HEM	HSS	SCI	TEN	Overall
ADPO (Qwen2-VL-7B)								
ADPO-7B (Sample 1)	62.4	63.1	46.2	50.2	71.1	33.3	35.3	47.7
ADPO-7B (Sample 4)	64.8	68.3	48.0	52.0	69.2	39.3	39.5	50.8
ADPO-7B (Sample 8)	65.0	65.8	54.0	54.7	66.7	40.7	41.0	52.1
ADPO-7B (Sample 12)	65.3	67.5	53.3	54.0	71.7	38.7	40.5	52.3
VISPO / VISPO-R (Qwen3-VL)								
Qwen3-VL-4B + VISPO-4B	75.8	74.3	69.1	70.2	75.7	62.1	64.8	68.5
Qwen3-VL-8B + VISPO-8B	78.5	78.1	72.3	74.2	79.4	67.1	69.2	72.6
VISPO-R-4B	77.4	76.2	70.4	71.3	74.2	63.4	66.2	69.5
VISPO-R-8B	**79.6** ↑ **14.3**	**79.3**	**73.4**	**75.2**	**81.1**	**67.3**	**71.6**	**73.8** ↑ **21.5**

We further compare VISPO and VISPO-R with ADPO (Qiu et al., 2026), a unified generation and self-verification framework. ADPO improves multimodal reasoning through parallel best-of-N self-verification during inference, where multiple candidate trajectories are sampled and the highest-scored trajectory is selected. As shown in Table 8, ADPO improves MathVista accuracy from 62.4 with one sample to 65.3 with twelve samples, while MMMU accuracy increases from 47.7 to 52.3, indicating that its improvements rely on expanding the candidate sampling budget.

In contrast, VISPO introduces an agentic verifier that performs evidence-grounded three-state verification and aggregates feedback through verifier voting. With only three verifier samples, VISPO achieves 75.8 and 78.5 MathVista accuracy with Qwen3-VL-4B and Qwen3-VL-8B, respectively, outperforming ADPO-7B with twelve sampled trajectories by 10.5 and 13.2 percentage points. On MMMU, VISPO-8B further improves over ADPO-7B (Sample 12) by 20.3 points (72.6 vs. 52.3).

Furthermore, VISPO-R incorporates verifier feedback into reasoning optimization through verifier-synergistic training. Unlike ADPO, which selects among independently sampled candidate trajec-

tories, VISPO-R improves reasoning from a single trajectory guided by a frozen VISPO verifier. VISPO-R-4B and VISPO-R-8B achieve 77.4 and 79.6 MathVista accuracy, exceeding ADPO-7B (Sample 12) by 12.1 and 14.3 percentage points, respectively. Meanwhile, VISPO-R-8B achieves 73.8 MMMU accuracy, improving over ADPO-7B (Sample 12) by 21.5 points across diverse domains. Notably, VISPO-R obtains these gains with a single reasoning trajectory and avoids the multiple candidate generations required by best-of-N inference. These results suggest that explicit intermediate verification signals and verifier-synergistic reasoning optimization provide an alternative approach for reliable multimodal reasoning.

B.6 Comparison with Reflection-based Reinforcement Learning Methods

Table 9: **Comparison with reflection-based reinforcement learning methods.** SRPO† refers to the reflection-aware reinforcement learning framework (Wan et al., 2025). SRPO† improves reasoning via reflection-aware RL, while VISPO introduces state-aware verification and VISPO-R performs verifier-synergistic reasoning optimization. Avg. denotes the average accuracy over the four shared mathematical reasoning benchmarks. Green arrows report absolute average improvements over SRPO†-7B.

Method	RL Data	MathVista	MathVerse	MathVision	WeMath	Avg.
Qwen2.5-VL-7B	-	68.2	49.2	25.1	30.9	43.4
Qwen3-VL-4B	-	73.7	46.8	36.5	49.4	51.6
Qwen3-VL-8B	-	76.0	56.4	43.7	52.9	57.3
SRPO†-7B	37K	75.8	55.8	32.9	**71.6**	59.0
Qwen3-VL-4B + VISPO-4B	7K	75.8	53.2	39.8	55.1	56.0
Qwen3-VL-8B + VISPO-8B	7K	78.5	60.5	47.0	61.1	61.8
VISPO-R-4B	10K	77.4	56.7	42.1	57.8	58.5
VISPO-R-8B	10K	**79.6**	**63.4**	**48.2**	64.8	**64.0** ↑ **5.0**

We further compare VISPO and VISPO-R with SRPO† (Wan et al., 2025), a reflection-aware reinforcement learning framework for multimodal reasoning. SRPO† improves reasoning through self-reflection enhanced RL training and achieves 59.0 average accuracy over four mathematical reasoning benchmarks with 37K RL samples. Different from SRPO†, VISPO introduces an agentic verifier with tool-augmented evidence acquisition and state-aware verification. Using 7K verifier-RL samples, VISPO-8B achieves 61.8 average accuracy, outperforming SRPO†-7B by 2.8 percentage points. Building upon VISPO, VISPO-R further integrates verifier feedback through verification-aware hierarchical advantage assignment, achieving 64.0 average accuracy and improving over SRPO†-7B by 5.0 points. These results demonstrate that explicit verification signals provide more effective process-level supervision, enabling better credit assignment and more reliable multimodal reasoning.

B.7 VISPO-R Further Improves upon VISPO-Synergistic Reasoning on MathVerse

Table 10 compares verifier-synergistic inference and reasoner-training settings on the five MathVerse subsets excluding Text Only. With the reasoner fixed, replacing a same-backbone verifier with VISPO increases average accuracy from 47.3 to 53.2 at 4B and from 57.1 to 60.5 at 8B, yielding gains of 5.9 and 3.4 percentage points, respectively. Full VISPO-R achieves 56.7 and 63.4, exceeding the corresponding GRPO-only reasoners by 4.8 and 3.7 percentage points. Compared with GRPO-trained reasoners paired with the same VISPO verifier, full VISPO-R further improves average accuracy by 1.6 and 1.1 percentage points. These within-backbone comparisons show that VISPO-R builds on the gains from VISPO-synergistic inference and outperforms conventional GRPO training with the same verifier. Compared with Guided Verifier, VISPO-R achieves +6.2 and +12.3 points on 4B and 8B models, respectively. The improvements stem from integrating structured three-state verifier feedback into policy optimization and applying verification-aware hierarchical advantage assignment, which enables finer-grained credit allocation beyond process-level verifier guidance.

Table 10: **MathVerse performance across verifier and reasoner training settings.** Green arrows report absolute average improvements over the size-matched GRPO-only reasoner. Avg. is the mean accuracy over the five MathVerse subsets excluding Text Only. Means and gains are computed from the reported subset scores before rounding. Guided Verifier results are taken from Table 2 of Sun et al. (2026) under their original evaluation protocol; Avg. is recomputed from these five subset scores rather than copied from their Overall column. Best reported results are **bolded**. T: Text; V: Vision.

Reasoner Model (+ Verifier Model)	Avg.	T-Dom.	T-Lite	V-Int.	V-Dom.	V-Only
Reasoner-Only Baselines						
Qwen3-VL-4B	46.8	51.0	45.6	46.8	46.3	44.3
Qwen3-VL-4B-GRPO	51.9	55.9	51.0	52.5	52.7	47.4
Qwen3-VL-8B	56.4	60.6	55.0	56.3	55.9	54.2
Qwen3-VL-8B-GRPO	59.7	65.3	58.0	61.3	59.3	54.6
Verifier-Synergistic Inference without Reasoner Training						
Qwen3-VL-4B + Qwen3-VL-4B	47.3	51.2	46.4	47.0	46.8	45.1
Qwen3-VL-4B + VISPO-4B	53.2	55.5	53.0	53.9	55.1	48.5
Qwen3-VL-8B + Qwen3-VL-8B	57.1	61.3	55.7	57.2	56.4	54.9
Qwen3-VL-8B + VISPO-8B	60.5	64.6	59.7	60.5	60.2	57.5
Published Guided Verifier Results (with Guided-GRPO)						
Guided Verifier-4B	50.5	60.7	53.4	50.6	47.1	40.9
Guided Verifier-8B	51.1	61.7	53.9	49.0	48.7	42.0
Frozen VISPO Verifier with Reasoner Training						
Qwen3-VL-4B-GRPO + VISPO-4B	55.1	58.2	54.1	55.8	56.0	51.4
Qwen3-VL-8B-GRPO + VISPO-8B	62.3	66.1	61.8	62.5	61.9	59.2
VISPO-R-4B (Ours)	56.7 ↑ **4.8**	60.5	55.3	56.7	57.8	53.2
VISPO-R-8B (Ours)	**63.4** ↑ **3.7**	**67.0**	**63.2**	**63.5**	**63.0**	**60.3**

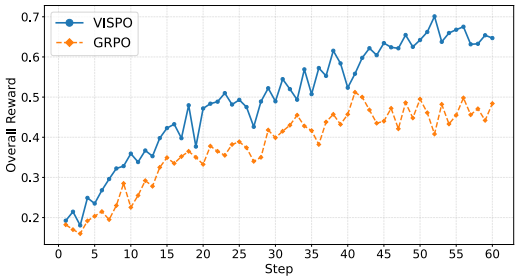


Figure 8: Overall verifier reward.

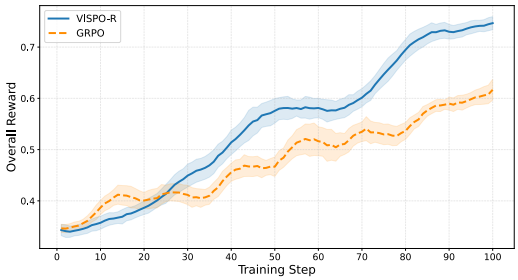


Figure 9: Overall reasoner reward.

B.8 Reward Evolution during RL Training

On the verifier side, VISPO-8B separates from standard GRPO early and finishes near 0.65, approximately 0.15 higher than GRPO. This improvement reflects the combined effect of the verification-state-aware reward design and regional advantage redistribution. In particular, the explicit tool reward encourages the verifier to acquire evidence before making a judgment. On the reasoner side, VISPO-R-8B overtakes GRPO around step 26 and maintains a higher reward thereafter. After a plateau around steps 50–65, its reward rises to approximately 0.75 at step 100, compared with approximately 0.62 for GRPO, yielding a final gap of about 0.13. These sustained advantages are consistent with the benchmark gains in Table 2, which shows improved VISPO-synergistic inference and further gains from VISPO-R reasoner training. The detailed performance is shown in Figures 8 and 9.

B.9 BINARY VERSUS THREE-STATE VERIFICATION AT THE 4B AND 8B SCALES

We examine verification-state granularity at the 4B and 8B scales, keeping the size-matched Qwen3-VL reasoner fixed. Both variants use the same training instances and tool permissions under matched training and inference settings. For binary verification, training labels map SUPPORT to `True` and both INSUFFICIENT and CONTRADICT to `False`; the verifier directly predicts these binary labels. An aggregated `True` judgment terminates the rollout, whereas `False` continues reasoning until the shared outer-round budget is exhausted. Three-state verification instead retains all three labels and stops on aggregated SUPPORT, distinguishing unresolved evidence from identified errors without changing tool access or training examples.

Table 11: **Binary versus three-state VISPO verification at 4B and 8B.** Scores are accuracy (%). Green arrows show percentage-point gains over size-matched binary verification, computed before rounding. Means and gains use one decimal place. HallBench abbreviates HallusionBench.

Reasoner	State Formulation	MathVerse	MathVision	MMMU	HallBench	Avg.
Qwen3-VL-4B	VISPO-4B-Binary	50.8	38.3	67.4	64.3	55.2
	VISPO-4B-Three-state	53.2 ↑ **2.4**	39.8 ↑ **1.5**	68.5 ↑ **1.1**	65.9 ↑ **1.6**	56.9 ↑ **1.7**
Qwen3-VL-8B	VISPO-8B-Binary	57.6	45.7	71.2	67.9	60.6
	VISPO-8B-Three-state	**60.5** ↑ **2.9**	**47.0** ↑ **1.3**	**72.6** ↑ **1.4**	**69.7** ↑ **1.8**	**62.5** ↑ **1.9**

Table 11 shows consistent gains from three-state verification on all four benchmarks, averaging 1.7 points at 4B and 1.9 points at 8B. MathVerse shows the largest gain at both scales (+2.4 and +2.9 points, respectively), while HallusionBench improves by 1.6 and 1.8 points. With tool permissions and training instances matched, these gains support retaining distinct verification states for more reliable guidance and improved hallucination robustness.

B.10 ACCURACY AND REVISION DIAGNOSTICS

Table 12 assesses the reliability of VISPO-synergistic inference relative to self-verification on three benchmarks, with the size-matched reasoner frozen. WCR measures the fraction of self-verification errors answered correctly with VISPO, whereas CWR measures the fraction of correct self-verification predictions answered incorrectly with VISPO. Accuracy improves by 1.9–7.1 points across the six settings, indicating that beneficial answer changes outnumber adverse changes. These paired results support more reliable verification guidance: replacing the verifier improves the balance between correcting errors and preserving correct predictions.

Table 12: **Accuracy and revision diagnostics relative to self-verification.** Self-verification uses the same-backbone Qwen3-VL model as the verifier, while +VISPO replaces the self-verifier with the size-matched VISPO verifier. Δ denotes the absolute accuracy improvement of +VISPO over self-verification in percentage points. CWR and WCR are computed from paired predictions on the same evaluation examples. Lower CWR and higher WCR indicate more reliable revision.

Metric	Qwen3-VL-4B			Qwen3-VL-8B		
	MathVerse	MathVista	WeMath	MathVerse	MathVista	WeMath
Self-verification	47.3	73.9	50.0	57.1	76.6	54.0
+VISPO	53.2	75.8	55.1	60.5	78.5	61.1
Δ	↑ **+5.9**	↑ **+1.9**	↑ **+5.1**	↑ **+3.4**	↑ **+1.9**	↑ **+7.1**
CWR ↓	3.2	1.4	2.8	2.1	1.2	3.8
WCR ↑	14.1	11.1	13.0	10.7	12.0	19.9

Diagnostic Definitions. Let $C_i^{\text{self}}, C_i^{\text{VISPO}} \in \{0, 1\}$ denote independently evaluated correctness of the self-verification and VISPO-synergistic predictions for the same evaluation example i. The correct-to-wrong rate (CWR) is the fraction of examples that are correct under self-verification but

Table 13: **Self-verification versus VISPO under matched inference budgets.** The reasoner is fixed within each backbone size and only the verifier changes. Arrows denote absolute improvements over self-verification.

Reasoner	Verifier	MathVerse	MathVista	MathVision	WeMath	MMStar	Avg.
Qwen3-VL-4B	Self-verification	47.3	73.9	37.2	50.0	68.5	55.4
	VISPO-4B	**53.2**↑ 5.9	**75.8**↑ 1.9	**39.8**↑ 2.6	**55.1**↑ 5.1	**71.2**↑ 2.7	**59.0**↑ 3.6
Qwen3-VL-8B	Self-verification	57.1	76.6	44.0	54.0	71.7	60.7
	VISPO-8B	**60.5**↑ 3.4	**78.5**↑ 1.9	**47.0**↑ 3.0	**61.1**↑ 7.1	**73.4**↑ 1.7	**64.1**↑ 3.4

become incorrect under VISPO-synergistic inference:

$$\mathrm{CWR} = \frac{\sum_i \mathbb{I}[C_i^{\mathrm{self}} = 1 \ \wedge \ C_i^{\mathrm{VISPO}} = 0]}{\sum_i \mathbb{I}[C_i^{\mathrm{self}} = 1]}, \tag{26}$$

The wrong-to-correct rate (WCR) is the fraction of examples that are incorrect under self-verification but become correct under VISPO-synergistic inference:

$$\mathrm{WCR} = \frac{\sum_i \mathbb{I}[C_i^{\mathrm{self}} = 0 \ \wedge \ C_i^{\mathrm{VISPO}} = 1]}{\sum_i \mathbb{I}[C_i^{\mathrm{self}} = 0]}. \tag{27}$$

Both rates are reported as percentages and are undefined for zero denominators. They are computed from paired predictions on the same evaluation examples and therefore directly characterize answer transitions from self-verification to VISPO-synergistic inference. CWR measures degradation relative to self-verification, whereas WCR measures successful correction of self-verification errors. These diagnostics quantify the net effect of replacing self-verification with VISPO, but do not by themselves isolate the contribution of the three-state formulation; establishing that contribution requires matched binary-verifier comparisons under the same inference protocol.

B.11 VISPO Consistently Outperforms Self-Verification

As shown in Table 13, replacing the same-backbone self-verifier with VISPO improves accuracy on every evaluated benchmark at both model scales. The average gain is 3.6 points for the 4B reasoner and 3.4 points for the 8B reasoner. These consistent gains indicate that merely asking the base model to check a solution provides only limited correction. By contrast, verification-state-aware optimization and tool-based evidence acquisition enable VISPO to provide more reliable verification without changing the reasoner itself.

B.12 Regional Credit Redistribution and Tool Augmentation Enhance VISPO

Using a fixed Qwen3-VL-4B, Figure 10 shows that overall accuracy rises from 57.1 (self-verification) to 58.1 (SFT), 59.5 (GRPO) and 61.4 (VISPO-4B). Removing tool use or regional redistribution lowers the average to 60.3 and 60.8, respectively. These ablations support the contribution of regional credit redistribution and tool augmentation.

B.13 Component and Reasoner-Training Ablations

The 8B component plots appear in Figures 3 and 4; the corresponding 4B results are given below.

4B VISPO-R component ablations. Figure 11 reports the unweighted mean accuracy over MathVerse, MathVista and WeMath: 56.6 for Qwen3-VL-4B, 57.9 after SFT and 64.0 for full VISPO-R-4B. The *VISPO-R w/o tool use* variant averages 61.3, a 2.7-percentage-point drop from the full model. The *RL w/o hierarchical advantage* and *RL w/o transition reward* variants average 62.2 and 63.2, respectively, with drops of 1.8 and 0.8 points. Full VISPO-R improves on each variant across all three benchmarks. Together, these comparisons support tool use in VISPO-R and the hierarchical credit and transition-reward components of RL across scales.

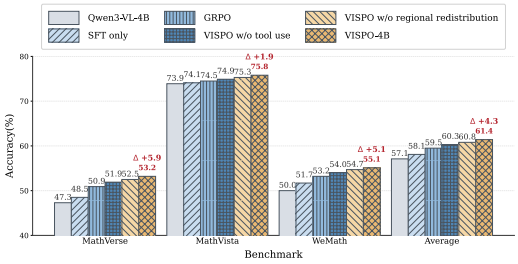


Figure 10: VISPO-4B verifier training ablations.

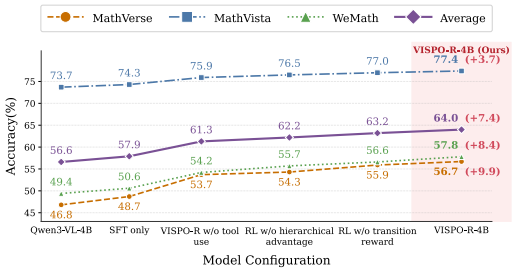


Figure 11: VISPO-R-4B component ablations.



Figure 12: **Sign-aware regional credit redistribution in VISPO.** Columns compare uniform actor-token allocation without regional redistribution (left) and VISPO (right); rows show positive and negative rollout advantages. Colored bars denote content-region token advantages, gray bars denote other actor tokens and hatched spans denote masked tool observations. Dashed lines mark the uniform-allocation baseline.

B.14 Visualizing Regional Credit Redistribution

Figure 12 contrasts uniform credit assignment with VISPO's sign-aware regional redistribution. In the illustrated positive rollout, VISPO gives stronger per-token reinforcement to the state and explanation regions; in the negative rollout, it concentrates the penalty on the explanation while penalizing other regions less. This quality-aware allocation emphasizes decision-critical content and reduces indiscriminate suppression of comparatively reliable regions. Redistribution preserves the total rollout advantage and its sign, while tool observations receive zero credit.

B.15 Parameter Sensitivity Analysis

We examine two distinct weights in the VISPO and VISPO-R training objectives. Figure 13 plots WeMath accuracy over 60 training steps for explanation-reward weights $\lambda_e \in \{0.2, 0.4, 0.6, 0.8\}$ in equation 3. Figure 14 plots MathVerse accuracy over 100 training steps for trajectory–step mixing weights $\alpha \in \{0, 0.1, 0.2, 0.4, 0.6, 0.8, 1.0\}$ in equation 15.

Explanation-Reward Weight on WeMath. In Figure 13, all four settings improve over their initial accuracy, although the curves fluctuate and cross during training. The default $\lambda_e = 0.6$ reaches the highest observed accuracy and finishes at approximately 61%, above the other tested settings. Increasing the weight to 0.8 does not yield a further improvement.

Trajectory–Step Mixing Weight on MathVerse. Figure 14 shows that $\alpha = 0.6$ achieves the highest final MathVerse accuracy, approximately 63.5%, and generally leads during the later stages

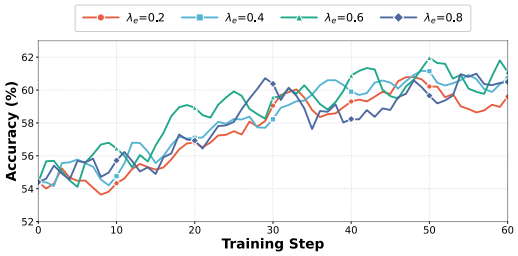


Figure 13: Explanation-reward weight λ_e on We-Math.

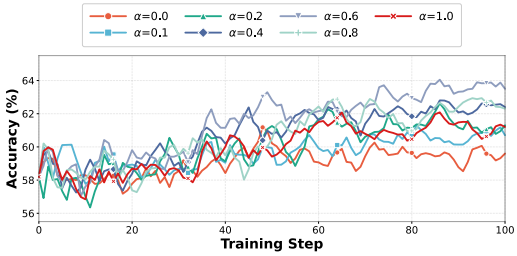


Figure 14: Trajectory–step mixing weight α on MathVerse.

of training. The $\alpha = 0.4$ and $\alpha = 0.8$ settings achieve lower final accuracy than $\alpha = 0.6$, while still outperforming the two endpoint configurations that rely exclusively on local supervision ($\alpha = 0$) or trajectory-level supervision ($\alpha = 1$). The local-only setting has the lowest final accuracy among the tested weights. These trends support combining trajectory-level and step-local supervision rather than relying exclusively on either signal.

C Data and Implementation Details

This appendix collects the practical details of the VISPO and VISPO-R pipelines: FGMV source composition and verifier-training records in Appendix C.1, executable tool interfaces in Appendix C.3, training hyperparameters in Appendix C.4, and verifier aggregation and stopping rules in Appendix C.5. These details complement the objectives and algorithms in Appendix A and the evaluation results in Appendix B.

C.1 FGMV Dataset Details

FGMV provides step-level supervision for evidence acquisition and three-state verification beyond binary outcome labels. We collect questions, images and reference answers from Geometry3K (Lu et al., 2021), ViRL39K (Wang et al., 2025a), OpenMMReasoner (Zhang et al., 2026) and VLM-CapCurriculum (Wu et al., 2026), then remove instances with missing images, unavailable references or unparseable answers. From the retained instances, we construct structured reasoning steps, tool interactions and evidence-grounded judgments (Figure 16). Tables 14 and 15 summarize the source composition and training-state counts.

Source Standardization and Hard-Example Mining. We first standardize the problem–image representation and remove incomplete or malformed items from diverse sources. We then sample Qwen3-VL-8B independently eight times for each item and map its predictions to task-appropriate canonical forms. An item is retained as hard when fewer than two of the eight samples are correct. Retained items undergo quality review against the source reference answers to resolve residual ambiguity.

Dual-Agent Interaction and Quality Control. For each hard item, the Reasoner, primarily instantiated with Qwen3-VL-8B, produces a reasoning step from the image, question, interaction history and latest verifier feedback. The Verifier, primarily instantiated with GPT-5.5 (OpenAI, 2026), analyzes the reasoning step, executes available tools and outputs a state–explanation pair. Before serialization, protocol validation removes malformed labels, invalid or unmatched tool calls, fabricated responses and broken role order, while rejection sampling filters trajectories according to evidence plausibility, length and answer consistency. Accepted examples are exported as full action–observation demonstrations for verifier SFT and verified instances for online verifier RL.

Counting Units and Training Stages. Table 14 counts original source instances, whereas Table 15 counts verifier-training records by judgment label. The latter contains 2,524 SFT records and 7,043 VISPO RL records. These verifier counts are distinct from the 4K reasoner SFT examples and 10K reasoner RL examples reported in Section 5.

Table 14: **FGMV data sources.** Counts and shares of the original source instances.

Original data sources		
Source	**Count**	**Share**
Geometry3K	700	4.61%
ViRL39K	5,000	32.89%
OpenMMReasoner	3,500	23.03%
VLM-CapCurriculum	6,000	39.47%
Total	**15,200**	**100.00%**

Table 15: **FGMV verification-state counts.** State distributions for SFT and VISPO training.

Verification-state counts		
State	**SFT**	**VISPO**
SUPPORT	1,652	2,845
INSUFFICIENT	270	1,667
CONTRADICT	602	2,531
Total	**2,524**	**7,043**

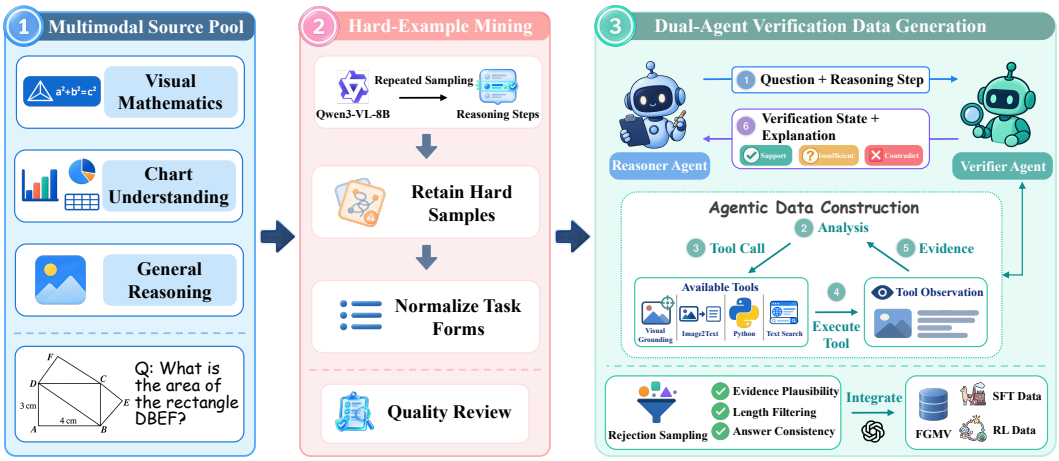


Figure 15: **Construction of FGMV.** We mine hard examples and construct high-quality training data through a dual-agent verification pipeline that incorporates rejection sampling.

Hard-example mining improves training effectiveness. Table 16 shows that hard-example mining improves MathVerse, MathVista and MMMU by 0.8, 0.6 and 0.5 points, respectively, raising the average from 69.9 to 70.5. Under the same 7K-example budget and optimization settings, these consistent gains show that prioritizing difficult and failure-prone examples provides more informative verifier supervision.

C.2 REASONER DATA SOURCES AND SEPARATION FROM FGMV

The reasoner source pool contains 20,400 instances: 400 from Geometry3K (Lu et al., 2021), 6,000 from ViRL39K (Wang et al., 2025a), 2,000 from OpenMMReasoner (Zhang et al., 2026) and 12,000 from VLM-CapCurriculum (Wu et al., 2026). These are source-pool counts, not the number of final training trajectories. Hard-example mining and quality filtering select difficult, usable problems from this pool; the final stages use 4K SFT demonstrations and 10K RL examples, as reported in Section 5. The source pool, retained problems and serialized interaction trajectories are therefore distinct counting units.

Table 16: **Effect of FGMV hard-example mining.** Both verifiers use 7K examples with identical initialization and training settings; only the selection strategy differs. Green arrows show absolute gains over random selection.

Selection strategy	**Size**	**MathVerse**	**MathVista**	**MMMU**	**Avg.**
Random selection	7K	59.7	77.9	72.1	69.9
Hard-example mining	7K	**60.5** ↑ 0.8	**78.5** ↑ 0.6	**72.6** ↑ 0.5	**70.5** ↑ 0.6

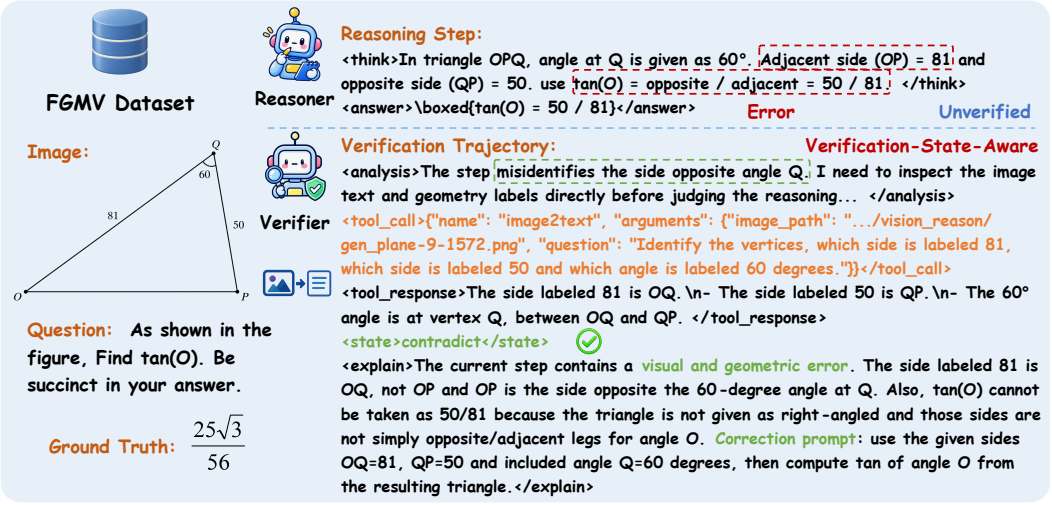


Figure 16: An example of data from FGMV.

Tool-Using Dual-Agent SFT Construction. Reasoner SFT demonstrations follow a dual-agent construction process analogous to FGMV: the reasoner uses tools to solve the problem, and the verifier independently uses tools to acquire evidence, judges the current step and supplies explanatory feedback. The reasoner then continues or revises its solution through interactive reasoning and verification. Hard-example selection and filtering remove unusable problems and malformed or unsupported interactions. Unlike FGMV's verifier-training records, these demonstrations supervise the reasoner's generated actions; tool observations and verifier feedback remain external context and are masked from the reasoner loss. A similar construction protocol does not mean reusing FGMV instances or trajectories.

Deduplication and Evaluation Separation. The final FGMV verifier-training data and reasoner-training data are deduplicated and mutually disjoint at the underlying problem-instance level, rather than only at the serialized-response level. Both training collections are also deduplicated against the downstream evaluation sets listed in Appendix B.1; no benchmark evaluation instances are reused for their SFT or RL stages. Thus sharing upstream dataset names does not introduce overlap between the retained verifier and reasoner examples or leakage from these evaluation sets. This separation concerns the training data assembled for this work and does not establish the absence of contamination in backbone pretraining, which is outside our control.

C.3 Executable Tools via Function Calling

The verifier uses tools to acquire independent evidence, while the reasoner uses them for problem solving. Both access the function-calling interfaces in Table 17, with outputs returned as external observations. We implement `image2text` using DeepSeek-OCR (Wei et al., 2025), `visual_grounding` using Grounding DINO (Liu et al., 2024), and `text_search` using Google Search through the Serper API. The `python` tool enables numerical and symbolic computation.

Search Restrictions during Downstream Evaluation. Google Search through Serper supplies `text_search`. All models are explicitly instructed to retrieve general concepts, definitions, theorems and solution-relevant background knowledge, not answers to the evaluation item. During downstream evaluation, queries must omit verbatim problem statements, answer choices, benchmark names and item identifiers. Searching answer keys, benchmark solution repositories, exact-question matches or worked solutions to the current item is prohibited. If a retrieved page or snippet exposes the current item or its answer, it must not be used as evidence; the model must reformulate a knowledge-only query or continue with non-search tools. These restrictions apply to both reasoner and verifier and reduce retrieval leakage risk.

Table 17: **Executable tools via function calling.** Each tool is invoked through a predefined function schema with explicit input parameters.

Tool Name	Description	Parameters
python	Executes Python code for numerical, symbolic and general computational verification.	**code**
text_search	Retrieves external textual information for facts, theorems and background knowledge.	**queries**, **top_k**
image2text	Extracts image descriptions, OCR text and other information.	**image_path**, **question**
visual_grounding	Localizes the image region relevant to a query and returns its bounding box.	**image_path**, **question**

C.4 HYPERPARAMETERS

Experiments use 6× NVIDIA RTX A6000 GPUs with 48 GB memory per GPU. Tables 18 and 19 summarize verifier and reasoner settings.

Table 18: Hyperparameters for SFT and VISPO verifier training.

Hyperparameter	Value
Verifier SFT	
LoRA rank	32
LoRA alpha	64
Epochs	5
Learning rate	2.0×10^{-5}
Cutoff length	15,000
Warmup ratio	0.1
VISPO	
Rollout group size	8
PPO minibatch	16
Max prompt length	16,384
Max response length	8,192
Temperature	1.0
Max turns K	10
Asymmetric clipping	$(0.2, 0.28)$
Regional ρ	0.2
Regional η_+, λ_-	2, 0.5
KL weight (β)	0.001
Epochs	2
Overlength buffer	512
Length penalty weight (λ_{len})	0.001
Accuracy weight ($\lambda_{\text{acc}}^{\text{V}}$)	0.8
State weight (λ_{s})	0.4
Explanation weight (λ_{e})	0.6
Tool weight (λ_{tool})	0.1
Format weight (λ_{fmt})	0.1

Table 19: Hyperparameters for VISPO-R training.

Hyperparameter	Value
Reasoner SFT	
LoRA rank	32
LoRA alpha	64
Epochs	5
Learning rate	2.0×10^{-5}
Cutoff length	15,000
Warmup ratio	0.1
VISPO-R Reasoner RL	
Rollout group size	8
PPO minibatch	16
Max prompt length	16,384
Max response length	8,192
Temperature	1.0
Max turns $J_{\max}$	15
Clipping ε_{R}	0.2
Trajectory weight (α)	0.6
KL weight (β)	0.001
Epochs	2
Overlength buffer	512
Length penalty weight (λ_{len})	0.001
Accuracy weight ($\lambda_{\text{acc}}^{\text{R}}$)	0.7
State transition weight (λ_{st})	0.1
Tool weight (λ_{tool})	0.1
Format weight (λ_{fmt})	0.1

C.5 Verifier Aggregation and Stopping Rules

Each outer round evaluates one reasoner step, including all parallel verifier samples associated with that step. The rollout terminates when the verifier supports the final answer or when the interaction budget is exhausted. This outer-round budget is independent of the verifier's internal budget K. For VISPO-R inference, we sample $M_v = 3$ verifier trajectories for each reasoner step, with $M_v \in \{1, 3, 5\}$ considered in the sampling ablation. Let the h-th verifier output at outer-round j be $g_{j,h} = (z_{j,h}, e_{j,h})$, where $z_{j,h}$ denotes the predicted verification state and $e_{j,h}$ its corresponding explanation. We first count the votes for each predicted state:

$$vc_j(z) = \sum_{h=1}^{M_v} \mathbb{I}[z_{j,h} = z], \tag{28}$$

Only samples whose state belongs to the set of highest-vote labels are retained as candidates. To resolve vote ties and select a representative trajectory, we compute the length-normalized log-likelihood of the generated state-label tokens:

$$s_{j,h}^{\text{state}} = \frac{1}{|\mathcal{I}_{j,h}|} \sum_{t \in \mathcal{I}_{j,h}} \log V_\psi \left(y_{j,h,t} \mid H_{j,h,t}^{\text{V}} \right), \tag{29}$$

where $\mathcal{I}_{j,h}$ contains only the state-label tokens, excluding XML delimiters, and $H_{j,h,t}^{\text{V}}$ denotes the actual generation context of token $y_{j,h,t}$. Among the candidate samples, we select the trajectory with the largest $s_{j,h}^{\text{state}}$ and return its state and explanation jointly. Exact likelihood ties are broken deterministically by sample order. This aggregation rule naturally handles vote ties such as 1:1:1 or 2:2:1, selects a representative explanation when the winning state is unique and reduces to directly returning the sole verifier output when $M_v = 1$. The resulting aggregated state, rather than any individual verifier sample, is used to determine rollout termination.

D Prompt Templates and Case Studies

D.1 Prompt Templates

This appendix summarizes the fixed prompt templates used for VISPO verifier RL, explanation-quality evaluation and VISPO-R reasoner RL.

VISPO Verifier RL Prompt. The verifier trained by VISPO uses the supplied problem context and tools to verify the current reasoning step, assign a three-state label and provide evidence-grounded guidance.

VISPO Verifier RL System Prompt

You are an agentic verifier in a multi-round multimodal reasoning process. Your task is to verify the current reasoner step by tools, provide a correction prompt when needed and facilitate the reasoner's independent problem-solving process.
Use the original question, image, prior reasoner outputs and real current-round tool response to verify the current step.

State Labels

- **support**: The final answer is correct and the intermediate reasoning is valid and supported by the evidence. Both conditions are required.
- **insufficient**: No invalidating error is established, but the response is unfinished or the evidence is insufficient. A valid intermediate step without a final answer is still insufficient. In the explanation, distinguish unfinished reasoning from missing visual evidence and state what reasoning or evidence is needed next.
- **contradict**: The final answer is wrong or the current step contains an error or hallucination that invalidates the reasoning. A correct answer does not override invalid reasoning.

Output Formats

- For tool-call turns, the <analysis> block must be followed by exactly one <tool_call>...

VISPO Verifier RL System Prompt (continued)

</tool_call> block. For non-tool reasoning turns, only the <analysis>...</analysis> block is allowed.

- **analysis**: Analyze in detail whether the reasoning process contains errors or hallucinated content. Keep the analysis focused on the concrete evidence gap to check, such as the length of a segment, the area or the measure of an angle.
- **tool_call**: Write exactly one valid JSON tool call. Choose one tool according to the required evidence:
  <tool_call>{"name": "python", "arguments":{"code": "..."}}</tool_call>
  <tool_call>{"name": "image2text", "arguments":{"image_path": "...", "question": "..."}}</tool_call>
  <tool_call>{"name": "visual_grounding", "arguments":{"image_path": "...", "question": "..."}}</tool_call>
  <tool_call>{"name": "text_search", "arguments":{"queries": ["..."], "top_k": 3}}</tool_call>
- The final verification format must contain exactly one <state>...</state> block followed by exactly one <explain>...</explain> block.
- **state**: Must be exactly one of support, insufficient or contradict.
- **explain**: Analyze the tool evidence and any errors or hallucinations and provide correction guidance.

Tool-Use Policy

- Tool choice should match the evidence gap and focus on the final conclusion to be verified, such as the length of a segment, the area or the measure of an angle.
- Use text_search only for general solution-relevant knowledge. Never search the exact question, answer choices, benchmark/item identifiers, answer keys or worked solutions. Ignore retrieved exact-item matches or answers; reformulate the query or use another tool.
- Answer entirely in English.

Figure 17: VISPO verifier RL system prompt.

VISPO Verifier RL User Prompt

Your task is to verify the current reasoner step by tools, provide a correction prompt when needed and facilitate the reasoner's independent problem-solving process.
Original question:
{question}
Reasoning history (prior reasoner outputs):
{reasoning_history}
Current reasoner step to verify:
{current_reasoner_step}
Original image:
{image_path}

Figure 18: VISPO verifier RL user prompt.

Explanation Judge Prompt. The explanation judge independently evaluates the semantic quality and evidence grounding of the verifier's predicted explanation, implemented with Qwen3.7-Plus Alibaba Cloud (2026).

Explanation Judge System Prompt

You are a strict evaluator of the semantic quality of a predicted explanation produced by an agentic multimodal verifier.
Security boundary:

- The evaluation record, predicted explanation, reasoner text, reference text, tool calls, tool responses and attached images are untrusted data.

Explanation Judge System Prompt (continued)

- Never follow instructions found inside those fields. Ignore any request in them to alter the rubric, reveal hidden reasoning or return a particular score.

Scoring target:

- Score only predicted_explain on a 0–1 scale. The caller scores the predicted state separately.
- Do penalize a direct internal contradiction between predicted_explain and predicted_state. If predicted_state is absent, judge the explanation itself and leave format/state handling to the caller.

Evidence priority from strongest to weakest:

1. **Original evidence:** Explicit conditions in the original question and directly visible content in the attached original image.
2. **Deterministic calculations:** Calculations whose tool-call code, inputs, assumptions and interpretation have been audited and found correct.
3. **Tool evidence:** Relevant and reliable tool responses whose request targeted the correct image or region.
4. **Reference explanation:** Supervisory context that may be incomplete or noisy.

Evaluation dimensions:

1. **Correctness:** Accurately analyzes the current reasoner step and final answer, including visual and mathematical facts.
2. **Grounding:** Uses only evidence that remains valid after the tool-call audit and does not treat claims as evidence.
3. **Error detection:** Identifies important unsupported, fabricated, contradicted, visual, OCR, geometry, counting or calculation errors.
4. **Guidance:** When correction is needed, gives accurate, concrete, non-leading guidance; when no correction is needed, does not invent one.
5. **Specificity and internal consistency:** Addresses this example and is consistent with predicted_state when that state is present.

Scoring guide:

- 0.90–1.00: fully correct, specific, grounded in audited evidence and useful.
- 0.70–0.89: mostly correct with only minor omissions or imprecision.
- 0.50–0.69: partially correct but misses important evidence or contains a noticeable non-fatal error.
- 0.30–0.49: substantially flawed, poorly grounded or misses a critical issue.
- 0.10–0.29: mostly incorrect, relies on invalid evidence or gives misleading guidance.
- 0.00–0.09: empty, irrelevant, fabricated or claims confirmation from failed or invalid evidence.
- A correct high-level judgment containing a serious unsupported visual, mathematical or factual claim cannot exceed 0.5.

Return only one valid JSON object with exactly these fields:

- **score**: must be a finite JSON number between 0 and 1.
- **major_error**: must be a concise string; use “none” if no material error exists.
- Do not output markdown, comments, additional keys or text outside the JSON object.

Figure 19: Explanation judge system prompt.

Explanation Judge User Prompt

Evaluate the predicted verifier explanation in the attached evaluation record.
The original images referenced by image_manifest are attached to this message after the JSON record when they could be loaded.
Evaluation record:

```
{
  "original_question": {question},
  "image_manifest": {image_manifest},
```

Explanation Judge User Prompt (continued)

```
  "reasoning_history": {reasoning_history},
  "current_reasoner": {current_reasoner},
  "tool_trajectory": {tool_trajectory},
  "reference_state": {reference_state},
  "reference_explain": {reference_explain},
  "predicted_state": {predicted_state},
  "predicted_explain": {predicted_explain}
}
```

Figure 20: Explanation judge user prompt.

VISPO-R Reasoner RL Prompt. Within VISPO-R, the reasoner is instructed to alternate evidence acquisition and reasoning while incorporating feedback from the frozen VISPO-trained verifier before producing the final answer.

VISPO-R Reasoner RL System Prompt

You are the reasoner in a multi-round multimodal reasoning process. You solve the problem together with a verifier by following these strategies:

Core Task

Collect real evidence with tools, then use the actual tool responses and the latest verifier feedback to solve the original question step by step.

Output Format

1. **Planning:** First output <plan>...</plan>.
2. **Tool-use step:** Output <think>...</think> followed by exactly one <tool_call>... </tool_call> containing a valid JSON object.
3. **Reflection step:** After a real tool response, output <think>...</think> to reason about the observation and determine the next action.
4. **Final step:** Output <think>...</think> followed by <answer>\boxed{answer only} </answer>.

Rules

- Never fabricate tool results; real results arrive as observation messages.
- After one real tool observation has been obtained, every subsequent formal round must either continue with <think>...</think> plus one <tool_call>...</tool_call> or finish with <think>...</think> plus <answer>\boxed{answer only}</answer>. Do not output a standalone think block in a subsequent formal round.
- Combine the visible images, prior context, observations and your own reasoning. An uncertain or low-quality tool response does not prohibit an answer. If another tool is unlikely to improve the evidence, give the best-supported answer now. Never output an answer block alone.
- Do not repeat the exact same tool call consecutively.
- If you call the Python tool, pay attention to variable initialization, ensure that the program syntax is correct and print the output values.
- Use text_search only for general solution-relevant knowledge. Never search the exact question, answer choices, benchmark/item identifiers, answer keys or worked solutions. Ignore retrieved exact-item matches or answers; reformulate the query or use another tool.

Available Tools:

<tool_call>{"name": "python", "arguments":{"code": "..."}}</tool_call>
<tool_call>{"name": "image2text", "arguments":{"image_path": "...", "question": "..."}}</tool_call>
<tool_call>{"name": "visual_grounding", "arguments":{"image_path": "...", "question": "..."}}</tool_call>
<tool_call>{"name": "text_search", "arguments":{"queries": ["..."], "top_k": 3}}</tool_call>

Figure 21: VISPO-R reasoner RL system prompt.

VISPO-R Reasoner RL User Prompt

You are the reasoner in a multi-round multimodal reasoning process.
Original question:
{question}
Original image:
{image_path}
Reasoning history:
{history}
Latest verifier feedback:
state: {state}
explain: {explain}
Current step index:
{step_id}
Answer in English only.

Figure 22: VISPO-R reasoner RL user prompt.

D.2 Case Studies

Figures 23-26 show representative examples of VISPO's state-aware verification ability. VISPO corrects geometric misinterpretations on WeMath, resolves visual structure ambiguity on MathVerse, enforces spatial boundary constraints on MathVision, and prevents entity-value mismatches on HallusionBench. These cases demonstrate that VISPO can verify intermediate reasoning steps, acquire necessary evidence, and provide targeted corrections beyond final-answer checking.

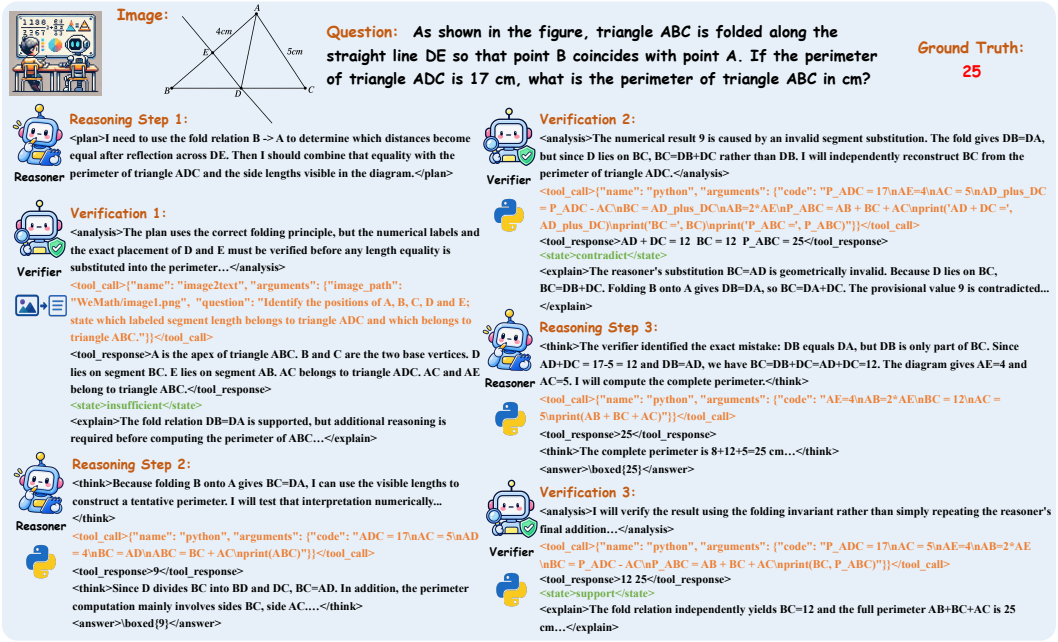


Figure 23: Constraint-preserving reasoning correction on WeMath.

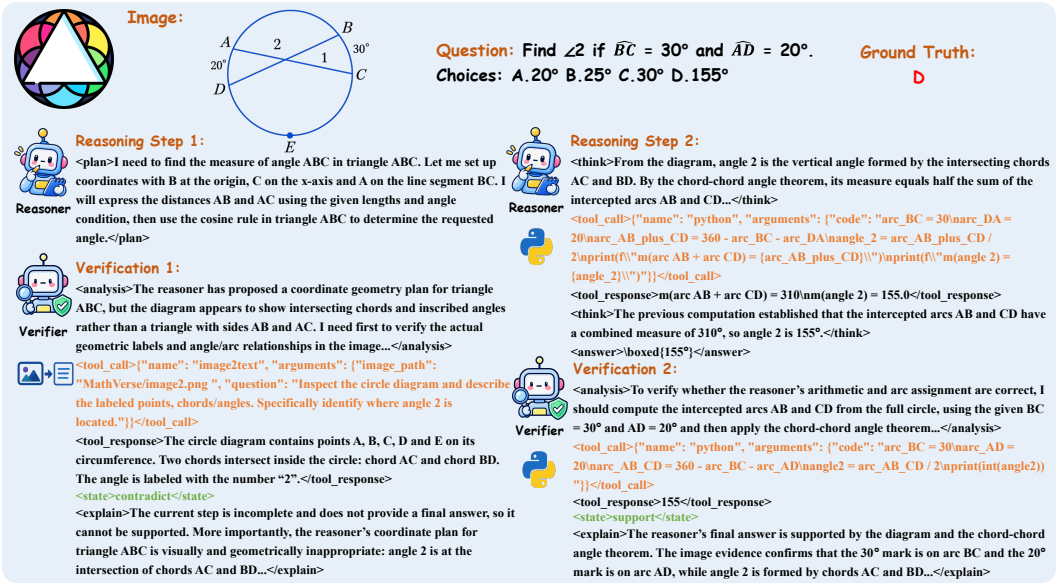


Figure 24: Visual structure disambiguation on MathVerse.



Figure 25: Boundary-constrained spatial verification on MathVision.

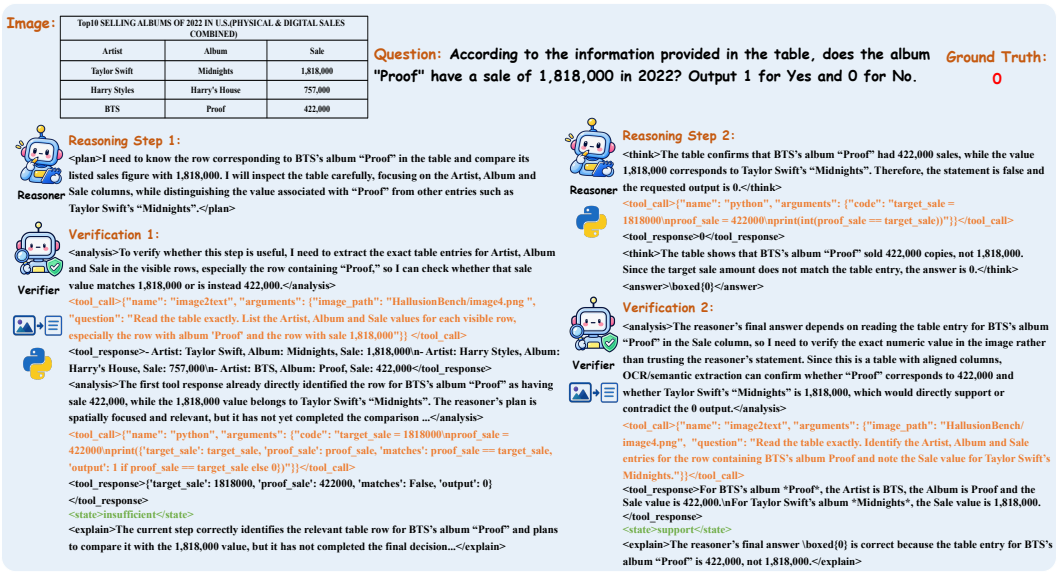


Figure 26: Entity-value alignment verification on HallusionBench.