SELF-EVOLVING VISUAL AGENT VIA MARGIN REWARD

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual agents that interleave reasoning with tool invocation have posed a promising direction for multimodal tasks. However, learning effective tool invocation usually relies on curated question–answer pairs or expert demonstrations at scale, which are expensive to acquire. In this study, we propose Marvo, a simple-yet-effective self-evolution approach that enables learning tool invocation from unlabeled images. Marvo consists of two agents: (1) a challenger that generates question-answer pairs from unlabeled images, and (2) a solver that learns tool invocation by solving questions through reinforcement learning. To encourage effective tool invocation, a good training question should be difficult enough to warrant tool use, yet remain within the solver's capabilities. To this end, we introduce a margin reward to the challenger, defined as solver's performance difference with and without access to tools. With the margin reward, we create an adaptive curriculum that encourages the solver to learn effective tool invocation. Across benchmarks covering agentic reasoning, visual search, mathematical, and general real-world reasoning, Marvo improves average accuracy by up to 7.1% — demonstrating margin reward as a preferred way to scale question complexity without human supervision.

1 INTRODUCTION

Visual agents have extended the capabilities of multimodal models by interleaving reasoning with tool invocation (OpenAI, 2025; Zhao et al., 2025). Instead of producing answers solely from an initial visual observations, these agents have already invoked tools to acquire rich evidence, process visual information, and use the feedback to guide subsequent reasoning. This interaction offers a promising approach to multimodal tasks, but requires agents to learn effective tool invocation–they have to determine which tool is useful and how to incorporate its output into the reasoning process.

Existing approaches learning these behaviors usually rely on curated question–answer pairs through human supervision or expert demonstrations (Zheng et al., 2026; Hong et al., 2026; Zhao et al., 2026b). Such supervision provides task objectives or examples of successful tool use, but is expensive to be collected at scale. Expert demonstrations require constructing suitable interaction trajectories, while question–answer supervision requires identifying meaningful questions and establishing answers. These challenges motivate learning with zero data (Huang et al., 2026) or unlabeled data. Yet converting this potential into useful supervision requires a mechanism for producing valuable tasks and questions, particularly, for visual agents that largely depend on tool invocation.

The value of a training question depends not only on its difficulty, but also on whether tools help the solver to correctly answer it. On the one hand, questions that the solver can answer without tools provide little incentive to acquire tool-use behaviors. On the other hand, questions that remain beyond the solver's capabilities even with tools offer limited opportunities to learn from successful interactions. Useful questions should therefore occupy a region between these extremes: they should pose a challenge beyond the solver's unaided reasoning while remaining solvable with tool invocation. Moreover, this region depends on the solver's current capabilities and may shift as the solver improves. A productive training curriculum should track this changing relationship between question difficulty and tool utility.

In this study, we propose Marvo, a simple yet effective self-evolution approach that enables visual agents learning tool invocation from unlabeled images (Fig. 1a). Marvo consists of two agents with

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.

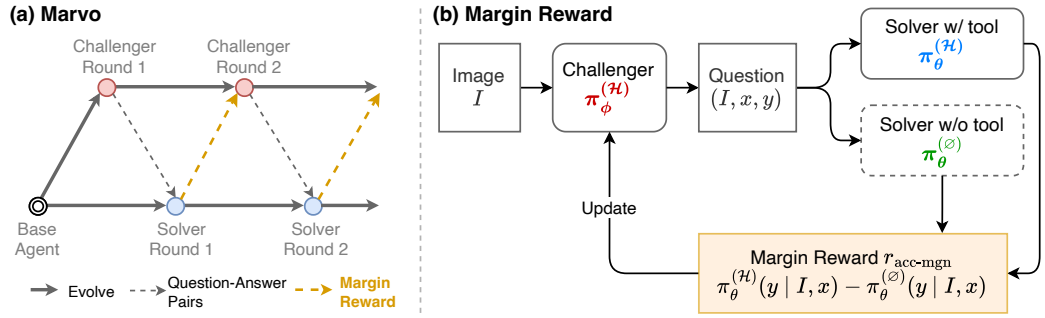


Figure 1: **Flowchart of Marvo.** (a) Marvo employs an evolution loop between a challenger and a solver, driven by the Margin Reward. (b) The Margin Reward quantifies the solver's performance gap with and without tools, encouraging the challenger to propose questions that require tool use.

complementary roles. A challenger generates question–answer pairs from unlabeled images, and a solver learns to answer the generated questions through reinforcement learning with access to tools (Fig. 1b). The challenger supplies training tasks, while the solver's performance provides feedback about their learning values. This interaction allows the training curriculum to adapt to the solver, reducing dependence on a fixed collection of manually curated tasks.

To guide the challenger toward questions that promote tool invocation, we introduce a *margin reward*, which measures solver's performance difference with and without access to tools. A larger margin, $i.e.$, performance difference, indicates that tool access enables the solver to answer a question more successfully than it could through unaided reasoning. Questions that are easily answered in both settings, or remain unsuccessful in both, receive little reward from this objective. Trained with the margin reward, the challenger is encouraged to generate questions that expose opportunities for useful tool invocation.

This reward connects question generation directly to the behavior we seek to improve. Optimizing question difficulty alone can favor tasks that the solver cannot solve, while optimizing tool-assisted success alone can favor tasks that do not require tools. The margin reward instead measures the solver's relative benefit from tool access. As this benefit is evaluated with respect to the current solver, the same objective can guide question generation as the solver's capabilities evolve. The resulting curriculum emphasizes questions for which tools make a practical difference, enabling visual agents to learn effective tool invocation without manually annotated question-answer pairs or expert demonstrations.

Across a diverse suite of benchmarks–11,691 questions spanning agentic reasoning (Li et al., 2025), visual search Wang et al. (2024b); Wu & Xie (2024), mathematical reasoning Zhang et al. (2024); Qiao et al. (2024); Wang et al. (2024a), and general real-world reasoning–Marvo improves the average accuracy of Qwen2.5-VL-7B by 7.1%. Together, these results establish the performance gap between tool-free and tool-augmented answering as a useful signal for generating training questions that improve tool-augmented reasoning.

2 Methodology

2.1 Preliminary

Marvo is built upon the self-evolution framework of R-Zero (Huang et al., 2026), which comprises two roles: a challenger π_ϕ and a solver π_θ. Given an input image I, the challenger proposes question–answer pairs $\{(x, y)\}$, which serve as the training curriculum for the solver. To drive self-evolution, the challenger and the solver are optimized alternately with the following objectives, as

$$\begin{aligned} &\max_{\phi} \mathbb{E}_{I\sim\mathcal{I},(x,y)\sim\pi_\phi(\cdot|I)} \left[r\left(x, y, \pi_\theta\right) \right], \\ &\max_{\theta} \mathbb{E}_{I\sim\mathcal{I},(x,y)\sim\pi_\phi(\cdot|I),\hat{y}\sim\pi_\theta(\cdot|I,x)} \left[\mathbf{1}_{\{y=\hat{y}\}} \right], \end{aligned} \tag{1}$$

where r is the challenger's reward and $\mathcal{I}$ denotes the image pool.

The R-Zero method designs r as the uncertainty score, as

$$r_{\text{uncertainty}}(x, y, \pi_\theta) = 1 - 2\left|\max_{\tilde{y}}\left(\pi_\theta(\tilde{y} \mid I, x)\right) - \frac{1}{2}\right|. \tag{2}$$

Note that $r_{\text{uncertainty}}$ uses a pseudo label $\tilde{y}$ voted by the solver, instead of the challenger proposed label y. $r_{\text{uncertainty}}$ attains its maximum when $\max_{\tilde{y}} \pi_\theta(\tilde{y} \mid I, x) = 1/2$. In other word, $r_{\text{uncertainty}}$ favors problems of intermediate difficulty for π_θ, Fig. 2(a). However, uncertainty alone is an indirect and often unreliable proxy for learning tool invocation. It captures only how confused the solver is, not whether solving a problem actually warrants tool use or would improve the solver's ability to invoke tools.

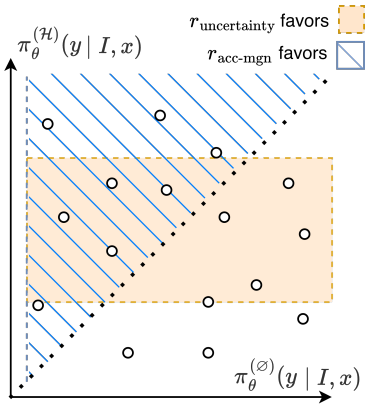


Figure 2: **Question Distribution.**

2.2 Incentivizing with Margin Reward

Building on Eq. (1), we investigate how the challenger's reward should be designed so that it incentivizes effective tool invocation.

In specific, we consider instantiating a solver π_θ deployed with different harnesses, $\pi_\theta^{(\mathcal{H})}$ and $\pi_\theta^{(\varnothing)}$, where $\mathcal{H}$ denotes the harness that grants tool access and $\varnothing$ denotes the empty harness, *i.e.*, answering directly without any tools. The goal is to train the target solver to master the tool-invocation capabilities that $\mathcal{H}$ affords beyond $\varnothing$. We therefore encourage the challenger π_ϕ to generate a problem (x, y) such that it is solvable under $\mathcal{H}$ and not solvable under $\varnothing$. Namely,

$$\textit{find } (x, y) \textit{ such that} \quad y \in \operatorname{supp}\Big(\pi_\theta^{(\mathcal{H})}(\cdot \mid I, x)\Big) \setminus \operatorname{supp}\Big(\pi_\theta^{(\varnothing)}(\cdot \mid I, x)\Big), \tag{3}$$

where $\operatorname{supp}(\cdot)$ denotes the support of a distribution.

As the strict support condition in Eq. (3) is rarely satisfied exactly, we relax it using the expected accuracy $\pi_\theta^{(\mathcal{H})}(y \mid I, x)$. For a problem (x, y), we design the challenger's reward as the margin in the solver's accuracy with and without access to tools, as

$$r_{\text{acc-mgn}}(y, \pi_\theta) \;=\; \pi_\theta^{(\mathcal{H})}(y \mid I, x) - \pi_\theta^{(\varnothing)}(y \mid I, x). \tag{4}$$

The margin in Eq. (4) is maximized accordingly when the target solver handles the problem reliably with tools while it fails without them (Fig. 2(a)), thereby steering the challenger toward questions that are difficult enough to warrant tool use, yet within the solver's capabilities when equipped with tools.

Connection with $r_{\text{uncertainty}}$. If we rewrite the policy probability in Eq. (4) using log-odds, *i.e.* $\operatorname{logit}(p) = \log(p/(1-p))$, and perform a first-order Taylor expansion, then Eq. (4) is interpreted as

$$r_{\text{acc-mgn}}(y, \pi_\theta) \;\approx\; \underbrace{\pi_\theta^{(\mathcal{H})}\left(1 - \pi_\theta^{(\mathcal{H})}\right)}_{\text{uncertainty gate}} \left(\operatorname{logit}\left(\pi_\theta^{(\mathcal{H})}\right) - \operatorname{logit}\left(\pi_\theta^{(\varnothing)}\right)\right). \tag{5}$$

Uncertainty is automatically gated by the Bernoulli variance $\pi_\theta^{(\mathcal{H})}(y|I,x)\big(1 - \pi_\theta^{(\mathcal{H})}(y|I,x)\big)$ in Eq. (5), which guarantees the proposed questions are effective incentives to train the solver.

2.3 Equilibrium Analysis

We briefly characterize the equilibrium state induced by Eq. (1) and Eq. (4), deferring the full analysis to Appendix C. At equilibrium, each proposed problem (x, y) falls to one of two categories, as visualized in Fig. 3.

In a *gap-closed* equilibrium,

$$\pi_\theta^{(\mathcal{H})}(y|I,x) - \pi_\theta^{(\varnothing)}(y|I,x) \equiv 0.$$

The two harnesses are indistinguishable. In the solved case, where $\pi_\theta^{(\mathcal{H})}(y|I,x) = \pi_\theta^{(\varnothing)}(y|I,x) = 1$, the tool-use capability that $\mathcal{H}$ affords has been learned by the model itself: under $\varnothing$, it can now answer directly without any external aid.

In a *gap-saturated* equilibrium,

$$\pi_\theta^{(\mathcal{H})}(y|I,x) = 1 > \pi_\theta^{(\varnothing)}(y|I,x).$$

The solver has mastered the proposed problems with tool access while failing them without tool access, which exposes an irreducible gap attributable solely to tool access. Both players are simultaneously at their optima, but the solver's gradient vanishes, so learning stalls. Notably, equilibria never contain partially solved problems, where $0 < \pi_\theta^{(\mathcal{H})}(y|I,x) < 1$. Such states are transient, and the uncertainty gate in Eq. (5) is active precisely along these transients, where the solver still perceives learning signal.

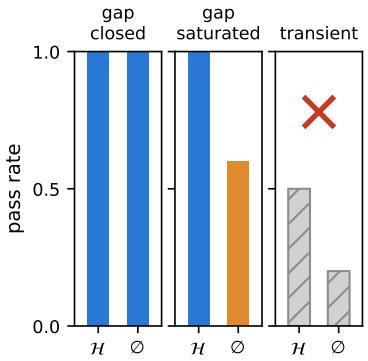


Figure 3: **Equilibrium States.**

2.4 Training Paradigm

Building upon $r_{\text{acc-mgn}}$, we introduce **Marvo** (Self-E**vo**lution via **Mar**gin Reward), a training paradigm where a challenger and a solver co-evolve, as summarized in Alg. 1. Both agents are instantiated from the same base model, yet play opposing roles: the challenger $\pi_\phi^{(\mathcal{H})}$ learns to propose difficult questions, while the solver $\pi_\theta^{(\mathcal{H})}$ learns to answer them. Notably, the challenger is also equipped with the harness $\mathcal{H}$, since posing questions that genuinely warrant tool use requires the challenger to experience the tools firsthand. The two agents are then trained *alternately* through group-mean policy optimization (GRPO)[1].

Challenger Phase. Given an unlabeled image I, the challenger freely investigates the image with $\mathcal{H}$ and proposes question-answer pairs in multiple-choice format. Each proposed question is answered n times each by $\pi_\theta^{(\mathcal{H})}$ and $\pi_\theta^{(\varnothing)}$. The resulting $r_{\text{acc-mgn}}$, combined with a format punishment r_{format}, is computed as the terminal reward for updating the challenger. Specifically, $r_{\text{format}} = 0$ if a valid question is extracted, and -1 otherwise.

Dataset Curation. We apply the challenger to a pool of images to propose candidate questions. Each question is answered n times by the target solver $\pi_\theta^{(\mathcal{H})}$, and questions with a pass rate of 0 or 1 are discarded, as such questions are either too hard or too easy for the current solver and their uniform solver rewards produce zero advantage in GRPO.

Solver Phase. The target solver $\pi_\theta^{(\mathcal{H})}$ is then updated with GRPO on the curated questions, using the rule-based reward $\mathbf{1}_{\{\hat{y}=y\}}$. We only train $\pi_\theta^{(\mathcal{H})}$, as mixing trajectories from $\mathcal{H}$ and $\varnothing$ in the same batch may lead to conflicting gradient updates.

3 Experiment

3.1 Setting

Models & Datasets. PyVision (Zhao et al., 2025) is adopted as the target harness $\mathcal{H}$ to incentivize, which is an interactive, multi-turn framework that enables VLMs to invoke Python-based tools. Qwen2.5-VL-7B-Instruct (Bai et al., 2025b) and Qwen3-VL-4B (Bai et al., 2025a) are adopted as base models. Training are initialized from the base models' Supervised Fine-Tuning (SFT) checkpoints[2]. We sample images from the PyVision-RL training set (Zhao et al., 2026b) and discard those with excessively large resolutions, yielding a pool of 42,124 images. The pool spans high-resolution natural images, mathematical diagrams, and scientific charts.

[1] See details of GRPO implementation in Appendix B.

[2] See details of SFT checkpoints in Appendix D.2.

Algorithm 1: Marvo: Self-Evolution via Margin Reward

Input: image pool $\mathcal{I}$; base model $\mathcal{M}$; rounds R; images per round B; challenger steps m, solver steps n; group size G; target harness $\mathcal{H}$

Output: trained solver $\pi_\theta^{(\mathcal{H})}$, trained challenger $\pi_\phi^{(\mathcal{H})}$

Initialize π_ϕ and π_θ from the same base model $\mathcal{M}$;
for $r \leftarrow 1$ **to** R **do**
  $\mathcal{B}_r \leftarrow$ next B images of the shuffled pool $\mathcal{I}$;
  `// Challenger Phase`
  for m *GRPO steps* **do**
    Sample images $I \sim \mathcal{B}_r$;
    Sample G question–answer pairs $\{(x_i, y_i)\}_{i=1}^{G}$ per image with $\pi_\phi^{(\mathcal{H})}$;
    For each proposed question, compute $\pi_\theta^{(\mathcal{H})}(y_i \mid I, x_i)$ and $\pi_\theta^{(\varnothing)}(y_i \mid I, x_i)$;
    $r_i \leftarrow r_{\text{acc-mgn}}(y_i, \pi_\theta) + r_{\text{format}}$;
    Update ϕ by GRPO;
  end
  `// Dataset Curation`
  $\mathcal{D}_r \leftarrow$ problems (x, y) authored by $\pi_\phi^{(\mathcal{H})}$ on $\mathcal{B}_r$;
  $\mathcal{D}_r \leftarrow$ filter out malformed and trivial questions in $\mathcal{D}_r$;
  `// Solver Phase`
  for n *GRPO steps* **do**
    Sample $(x, y) \sim \mathcal{D}_r$ (one pass; no epochs);
    Sample G episodes per problem $\hat{y}_i \sim \pi_\theta^{(\mathcal{H})}(\cdot \mid I, x)$;
    $r_i \leftarrow \mathbf{1}_{\{y=\hat{y}_i\}}$;
    Update ϕ by GRPO;
  end
end

Evaluation. To demonstrate Marvo's efficacy, the trained models are evaluated on a diverse set of benchmarks. *Agentic reasoning* is measured on TIR-Bench (Li et al., 2025), which requires actively manipulating visual inputs with tools rather than relying on static observation. *Visual search* is assessed on V* (Wu & Xie, 2024) and HRBench-4K (Wang et al., 2024b), which test the grounding of fine-grained details in high-resolution and information-dense images. *Multimodal mathematical reasoning* is evaluated on MathVerse (Zhang et al., 2024), MathVision (Wang et al., 2024a), and WeMath (Qiao et al., 2024), where diagrams, charts, and plots must be interpreted to solve math problems. *real-world understanding* of everyday scenes is tested on RealWorldQA (xAI, 2024).

3.2 Main Results

In Table 1, Marvo is compared against R-Zero across two model families, alongside evaluation results of frontier models (Comanici et al., 2025; OpenAI, 2025) and open-source tool-augmented visual agents (Liu et al., 2025; Hong et al., 2026; Zhao et al., 2026b) for reference.

Marvo prevails at matched iteration. Across both model families, Marvo consistently outperforms R-Zero at matched iterations, with the largest gains appearing exactly where agency matters most: +3.7 on HR4K and +2.1 on V* for the 7B model at Round 3, and +4.7 on V* for the 4B model. The cumulative picture is even clearer. The advantage is most pronounced on agentic reasoning: Marvo delivers gains on both Qwen2.5-VL-7B and Qwen3-VL-4B, whereas R-Zero with Qwen3-VL-4B finishes below its starting point on TIR-Bench. Surprisingly, our evolved 7B model surpasses PyVision-RL on TIR-Bench, despite both being trained from the same SFT checkpoint.

Marvo shows better stability across rounds. The per-round deltas in Tab. 1 expose a qualitative difference between the two evolution loops. R-Zero's improvements are often non-monotic: its 4B run regresses on four of seven benchmarks in Round 2 alone and its 7B run makes no progress on three benchmarks in Round 2. Aggregating both model families, 9 of R-Zero's 42 per-round

Table 1: **Main results.** For each self-evolution process (one model family with one method), the best value in each column is in **bold**. Parenthesized values are the change versus the previous round. Overall is the mean over the seven benchmarks.

	Agentic	Visual Search		Mathematical Reasoning			General	
	TIR-Bench	V*	HR4K	MathVerse	MathVision	WeMath	RWQA	Overall
Gemini-2.5-Pro	28.9	83.8	87.3	82.9	63.8	78.0	78.0	71.6
o3	26.9	95.7	-	-	73.3	80.6	-	-
Qwen2.5-7B	16.0	78.5	71.6	33.8	25.6	34.6	57.5	34.8
Qwen3-4B	-	74.9	73.5	42.3	44.6	-	70.9	-
DeepEyesV2	-	81.8	77.9	52.7	28.9	38.1	-	-
Agent0-VL	-	-	-	53.1	37.3	71.7	-	-
PyVision-RL	19.8	88.7	78.1	55.8	28.7	47.7	66.3	46.6
Self-Evolution with Qwen2.5-VL-7B								
R-Zero								
SFT	17.1	72.7	67.0	41.5	25.1	31.9	57.5	36.6
Round 1	17.4(+0.3)	73.3(+0.6)	67.5(+0.5)	43.1(+1.6)	**25.6**(+0.5)	47.7(+15.8)	55.8(–1.7)	39.6(+3.0)
Round 2	18.5(+1.1)	74.9(+1.6)	67.5(+0.0)	43.1(+0.0)	23.7(–1.9)	51.1(+3.4)	57.9(+2.1)	39.8(+0.2)
Round 3	**20.7**(+2.2)	**76.4**(+1.5)	**70.1**(+2.6)	**44.2**(+1.1)	**25.6**(+1.9)	**56.5**(+5.4)	**61.4**(+3.5)	**42.2**(+2.4)
Marvo (Ours)								
SFT	17.1	72.7	67.0	41.5	25.1	31.9	57.5	36.6
Round 1	20.2(+3.1)	73.8(+1.1)	67.3(+0.3)	43.2(+1.7)	25.2(+0.1)	48.7(+16.8)	58.0(+0.5)	40.1(+3.5)
Round 2	20.6(+0.4)	75.4(+1.6)	68.6(+1.3)	43.6(+0.4)	25.6(+0.4)	53.4(+4.7)	**65.1**(+7.1)	41.6(+1.5)
Round 3	**21.7**(+1.1)	**78.5**(+3.1)	**73.8**(+5.2)	**44.2**(+0.6)	**27.8**(+2.2)	**58.6**(+5.2)	64.6(–0.5)	**43.7**(+2.1)
Self-Evolution with Qwen3-VL-4B								
R-Zero								
SFT	**22.3**	78.0	71.0	49.7	38.8	57.4	62.2	47.9
Round 1	22.2(–0.1)	**83.3**(+5.3)	75.3(+4.3)	52.9(+3.2)	40.0(+1.2)	59.5(+2.1)	**68.1**(+5.9)	50.4(+2.5)
Round 2	21.6(–0.6)	78.5(–4.8)	**77.0**(+1.7)	**53.6**(+0.7)	39.7(–0.3)	64.3(+4.8)	62.6(–5.5)	50.8(+0.4)
Round 3	22.1(+0.5)	79.1(+0.6)	76.8(–0.2)	52.7(–0.9)	**40.3**(+0.6)	**67.4**(+3.1)	66.3(+3.7)	**51.4**(+0.6)
Marvo (Ours)								
SFT	22.3	78.0	71.0	49.7	38.8	57.4	62.2	47.9
Round 1	22.9(+0.6)	81.7(+3.7)	75.5(+4.5)	52.9(+3.2)	38.1(–0.7)	59.0(+1.6)	**65.1**(+2.9)	49.7(+1.8)
Round 2	22.9(+0.0)	83.3(+1.6)	77.3(+1.8)	53.5(+0.6)	**39.7**(+1.6)	65.5(+6.5)	64.6(–0.5)	51.3(+1.6)
Round 3	**23.8**(+0.9)	**83.8**(+0.5)	**77.9**(+0.6)	**53.7**(+0.2)	39.6(–0.1)	**68.8**(+3.3)	64.2(–0.4)	**52.0**(+0.7)

deltas are negative, compared with 5 of 42 under Marvo, and the largest single-round regression is 5.5 points under R-Zero versus 0.7 under Marvo. Notably, R-Zero's regressions concentrate on the tool-dependent benchmarks (V*, RealworldQA, TIR-Bench), suggesting that its self-generated data distribution shifts substantially between rounds; the near-monotone improvement under Marvo indicates a more stable co-evolution curriculum.

Tool invocation is where the gap opens up. The sharpest contrast appears on benchmarks that cannot be solved without calling tools. On TIR-Bench, three rounds of R-Zero leave the 4B model slightly *worse* than its SFT start, while Marvo improves it by +1.5 and the 7B model by +5.7; TIR-Bench is also where our very first iteration jumps the most. Visual search tells the same story, with cumulative 7B gains of +5.8 on V* and +6.8 on HR4K.

3.3 Training Dynamics

Figure 4 follows the 4B Marvo run through its three evolution rounds, tracking the challenger reward, the solver reward, and a per-round probe of $\pi_\theta^{(\mathcal{H})}$ versus $\pi_\theta^{(\varnothing)}$ accuracy on each round's own questions. The challenger learns on two timescales. Fig. 4a shows that in Round 1 the challenger reward climbs steadily while $r_{\text{acc-mgn}}$ stays flat below zero. In fact, the fraction of responses from which a question–answer pair could not be extracted falls from 0.30 to 0.17 over the run, so nearly all early progress is the model learning to emit well-formed, extractable question–answer pairs. Only

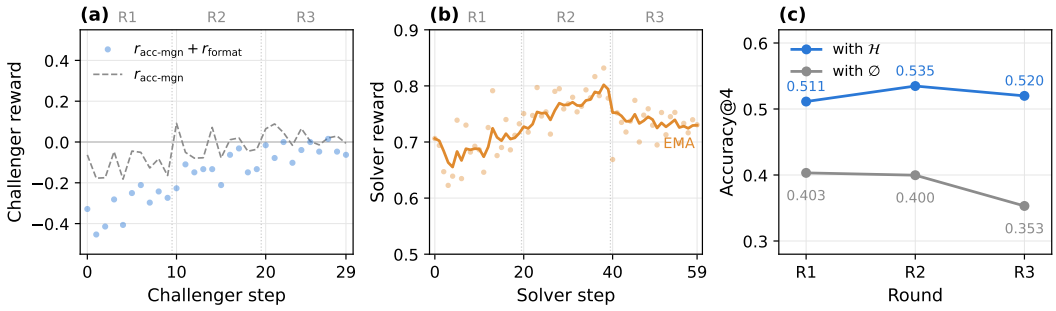


Figure 4: **Training Dynamics.** **(a-b)** Reward progress in challenger and solver training phase. **(c)** Per-round difficulty of challenger-generated datasets, as measured by model at each round.

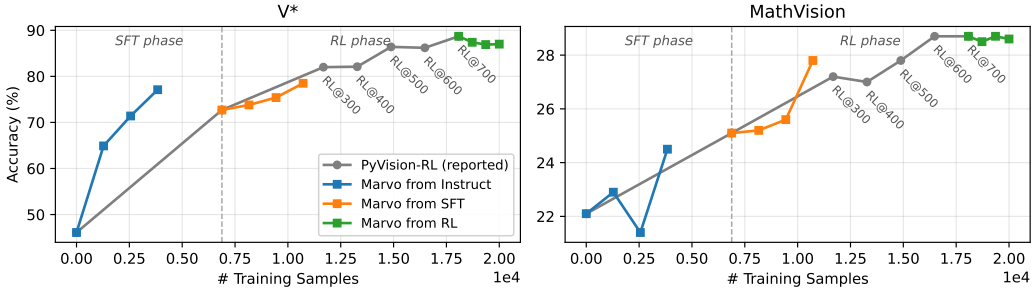


Figure 5: **Marvo across post-training stages.** The gray curve is the reported PyVision-RL training progress. Colored curves are Marvo experiments launched from different checkpoints.

once the format is in place, does the margin itself move. Within each round the solver's reward rises as it masters the current question set (Fig. 4b). At the beginning of each round, the reward levels are reset, as the updated challenger proposes questions the current solver cannot yet answer well. Across all three rounds, the proposed problems stay within the sweet spot of the uncertainty gate in Eq. (5), where the solver keeps receiving a strong training signal (Fig. 4c). When the solver is instead probed without tool access, however, pass rates drop, indicating that the curated problems genuinely require tool invocation.

4 Analysis

Self-Evolution Across Post-Training Stages. To understand *when* Marvo helps in post-training, we plug it at three stages of the post-training pipeline in PyVision-RL (Zhao et al., 2026b) (the Instruct model, the SFT cold-start checkpoint, and the final RL model) and align the evolution curves to the reported training progress (Fig. 5).

At early stages, Marvo-generated curriculum incentivizes the visual agent to learn tool invocation much as human-annotated data does, even though it uses no human-annotated question sets or demonstrations at all. Note that the average number of tool turns per evaluation question rises across different tasks (Fig. 6), so the agent's tool invocation ability does not degrade under self-evolution.

However, improvement is little when Marvo is introduced to the post-trained model. We conjecture that this is an entropy effect. PyVision-RL reports that policy entropy falls from 0.75 to about 0.4 over the course of RL (Zhao et al., 2026b). Entropy is a common proxy for a policy's willingness to explore. Once entropy has collapsed, there is little exploration headroom left for self-generated data to amplify, and Marvo's influence on the agent fades.

Marvo with Another Harness. We further evaluate Marvo and R-Zero under an alternative tool configuration. The VLM is additionally equipped with an image-zoom-in tool, as shown in Ap-

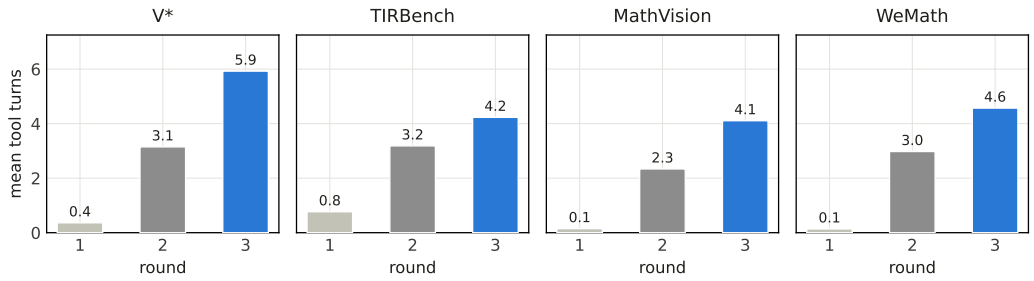


Figure 6: **Mean number of tool turns during evaluation.** Initializing from Instruct model, Marvo incentivizes tool invocation without human-curated data supervision.

Table 2: **Performance evolution** on V* with Zoom-In tool.

Round	Qwen3-VL-4B	1	2	3	4	5
Marvo	84.3	84.8	85.3	89.5	89.0	**91.1**
R-Zero	84.3	86.9	85.9	**89.0**	88.5	86.9

Table 3: **Challenger performance.**

Marvo Round	TIR-Bench	V*
1	19.8	71.7
2	19.5	72.3
3	19.8	73.8

pendix. B.1, which takes a bounding box coordinate as input and returns the cropped image region. Since this tool requires no additional supervision, we train the visual agent directly from the Qwen3-VL-4B checkpoint, bypassing the cold-start stage. As shown in Tab. 2, Marvo improves steadily across self-evolution rounds, rising from 84.29 to 91.10 and largely preserving the consistent growth observed in the PyVision setup. In contrast, R-Zero gains early momentum but oscillates after Round 1. The gap between the two methods widens, suggesting that Marvo transfers robustly across harnesses, maintaining its advantage even when the action space changes from dynamic Python execution to the image-zoom-in tool.

Challenger Performance on Visual Reasoning Tasks. Despite the promising performance of Marvo, a natural question arises: since the challenger $\pi_\phi^{(\mathcal{H})}$ shares the same harness $\mathcal{H}$ as the solver and proposes the question and answer, why do we need to train the solver at all? We evaluate the challenger's per-round answering accuracy on the benchmarks (Table 3) and find that it is consistently worse than the trained solver's and essentially flat across rounds. The challenger's performance oscillates around on TIR-Bench and improving only marginally on V*.

The challenger's training signal comes from its role as a task proposer, so its capacity is allocated to synthesizing diverse, verifiable, and appropriately difficult questions rather than to solving itself. Because a proposed question is sampled from the same trajectory as its answer, its phrasing tends to encode the solution path—an exposure bias that inflates self-reported competence, whereas benchmark questions are exogenous and neutral.

Extension of Margin Reward. The accuracy-margin reward can be generalized to compare the target agent against a *weaker reference agent*, rather than against the model's own baseline:

$$\tilde{r}_{\text{acc-mgn}}(y, \pi_\theta; \mathcal{H}_1, \mathcal{H}_2) \;=\; \pi_\theta^{(\mathcal{H}_1)}(y \mid I, x) - \pi_\theta^{(\mathcal{H}_2)}(y \mid I, x), \tag{6}$$

where $\pi_\theta^{(\mathcal{H}_1)}$ is the target agent we seek to optimize and $\pi_\theta^{(\mathcal{H}_2)}$ is a deliberately weaker visual agent operating under a different harness $\mathcal{H}_2$. Since Eq. (6) has the same margin structure as $r_{\text{acc-mgn}}$, the equilibrium analysis in Sec. 2.2 remains.

A practical challenge, however, is constructing an agent that is *reliably weaker* than $\mathcal{H}_1$: naively simplifying the tool configuration does not necessarily degrade performance, since tool diversity may hurt as often as it helps. We therefore restrict the capability of $\mathcal{H}_1$ by reducing its maximum conversation turns while keeping the tool configuration fixed, which yields a strictly less expressive agent without altering its action space. Specifically, we instantiate $\mathcal{H}_2$ as a truncated variant of PyVision ($\mathcal{H}_1$) whose maximum conversation turns is 3.

Table 4: **Per-round solver performance** under Eq. (6) setup, where $\mathcal{H}_2$ is a truncated variant of $\mathcal{H}_1$. Parenthesized values are the change versus the previous round.

	Agentic	Visual Search	Mathematical Reasoning	
	TIR-Bench	V*	MathVision	WeMath
Qwen3-4B-SFT	22.3	78.0	**38.8**	57.4
Round 1	22.6(+0.3)	78.5(+0.5)	35.1(–3.7)	57.7(+0.3)
Round 2	22.4(–0.2)	**79.1**(+0.5)	35.8(+0.6)	59.0(+1.3)
Round 3	**23.2**(+0.7)	77.5(–1.6)	38.0(+2.1)	**62.5**(+3.5)

Table 4 reports an additional Marvo run under this setting. Although the trained model still improves round by round, its final performance remains below the corresponding results in Tab. 1. We attribute the gap to the sparsity of the extended margin: $\tilde{r}_{\text{acc-mgn}}$ with favors questions requiring more than 3 turns, so $\tilde{r}_{\text{acc-mgn}}$ is near zero on most of the batch, and the effective learning signal is much weaker than that of $r_{\text{acc-mgn}}$, which differs from the direct-answering baseline on a far broader set of prompts.

5 Related Work

Reinforcement Learning for Large Language Models. Reinforcement learning (RL) with verifiable rewards (RLVR) fine-tunes models against rule-based checkers in domains with objective correctness criteria such as mathematics and code (Shao et al., 2024; Yu et al., 2025; Zhao et al., 2026d; Zheng et al., 2025). A common prerequisite of RLVR is a large corpus of human-curated tasks with gold labels, which is costly to produce and expensive to scale (Su et al., 2025). To relax the dependency of human annotation, label-free RL derives reward signals from the model's own outputs (Agarwal et al., 2026; Zuo et al., 2026; Zhang et al., 2026a;b; Zhao et al., 2026a;c; He et al., 2026). These methods remove the need for gold answers but still presuppose a pre-existing corpus of unlabeled questions, falling short of fully autonomous self-improvement.

Self-Evolving Large Language Models. This aims to eliminate both human-written tasks and labels by letting models generate, refine, and learn from their own experience (Tao et al., 2024). Early instantiations adapt self-play to language modeling: a model acts as multiple roles to bootstrap preference or instruction data without annotation (Chen et al., 2024; Yuan et al., 2024; Wu et al., 2025; Huang et al., 2026). Self-challenging agents go further by synthesizing the tasks themselves, but rely on external verifiers to guarantee feasibility and reliability, confining them to domains with an explicit correctness oracle (Zhao et al., 2026a; Wang et al., 2026; Xia et al., 2025; Zhou et al., 2026; Lin et al., 2025; Yue et al., 2026). Despite of the progress, existing self-evolution methods largely rely on pre-defined difficulty or uncertainty metrics to guide the generation of tasks. For visual agents that largely depend on tool invocation, the plausibility of these metrics requires to be further elaborated. In this study, we define the margin reward metric, extending self-evolution beyond text-only reasoning to visual agents, and enabling co-evolution of solvers, challengers and tasks towards higher reasoning capacity in visual domains.

6 Conclusion

We proposed Marvo, a simple yet effective approach for visual agent self-evolution without using human-annotated data. In specific, we introduced a margin reward to the challenger, defined as solver's performance difference with and without access to tools. Based on the defined margin reward, we create an adaptive curriculum that encourages the solver to learn effective tool invocation. Through extensive experiments on challenging benchmarks spanning diverse model architectures and agent harnesses, we validated Marvo's consistently performance gains, with striking contrast to the baseline and the state-of-the-art approaches. Marvo provides a fresh insight to the development of self-evolving agents for a broad range of multimodal tasks.

REFERENCES

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning. *Advances in Neural Information Processing Systems*, 38:107150–107180, 2026.

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

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*, 2024.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Bingxiang He, Yuxin Zuo, Zeyuan Liu, Shangziqi Zhao, Zixuan Fu, Junlin Yang, Cheng Qian, Kaiyan Zhang, Yuchen Fan, Ganqu Cui, et al. How far can unsupervised rlvr scale llm training? In *International Conference on Learning Representations*, volume 2026, pp. 14823–14865, 2026.

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

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-zero: Self-evolving reasoning llm from zero data. In *International Conference on Learning Representations*, volume 2026, pp. 130770–130790, 2026.

Ming Li, Jike Zhong, Shitian Zhao, Haoquan Zhang, Shaoheng Lin, Yuxiang Lai, Chen Wei, Konstantinos Psounis, and Kaipeng Zhang. Tir-bench: A comprehensive benchmark for agentic thinking-with-images reasoning. *arXiv preprint arXiv:2511.01833*, 2025.

Zi Lin, Sheng Shen, Ilia Kulikov, Jingbo Shang, Jason Weston, and Yixin Nie. Learning to solve and verify: A self-play framework for code and test generation. *arXiv preprint arXiv:2502.14948*, 2025.

Jiaqi Liu, Kaiwen Xiong, Peng Xia, Yiyang Zhou, Haonian Ji, Lu Feng, Siwei Han, Mingyu Ding, and Huaxiu Yao. Agent0-vl: Exploring self-evolving agent for tool-integrated vision-language reasoning. *arXiv preprint arXiv:2511.19900*, 2025.

OpenAI. Introducing o3 and o4-mini, 2025.

Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Zhuoma GongQue, Shanglin Lei, Zhe Wei, Miaoxuan Zhang, et al. We-math: Does your large multimodal model achieve human-like mathematical reasoning? *arXiv preprint arXiv:2407.01284*, 2024.

Qwen Team. Qwen3.8-Max: A new bar for coding and cowork, August 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yi Su, Dian Yu, Linfeng Song, Juntao Li, Haitao Mi, Zhaopeng Tu, Min Zhang, and Dong Yu. Crossing the reward bridge: Expanding rl with verifiable rewards across diverse domains. *arXiv preprint arXiv:2503.23829*, 2025.

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. A survey on self-evolution of large language models. *arXiv preprint arXiv:2404.14387*, 2024.

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. In *NeurIPS*, 2024a.

Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, and Dacheng Tao. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. *arXiv preprint*, 2024b.

Yinjie Wang, Ling Yang, Ye Tian, Ke Shen, and Mengdi Wang. Co-evolving llm coder and unit tester via reinforcement learning. *Advances in Neural Information Processing Systems*, 38:143630–143664, 2026.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal llms. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pp. 13084–13094. IEEE, 2024.

Yue Wu, Zhiqing Sun, Rina Hughes, Kaixuan Ji, Yiming Yang, and Quanquan Gu. Self-play preference optimization for language model alignment. In *International Conference on Learning Representations*, volume 2025, pp. 91558–91582, 2025.

xAI. Realworldqa: Real-world spatial understanding benchmark, 2024.

Peng Xia, Kaide Zeng, Jiaqi Liu, Can Qin, Fang Wu, Yiyang Zhou, Caiming Xiong, and Huaxiu Yao. Agent0: Unleashing self-evolving agents from zero data via tool-integrated reasoning. *arXiv preprint arXiv:2511.16043*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: an open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, 2025.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Zhenrui Yue, Kartikeya Upasani, Xianjun Yang, Suyu Ge, Shaoliang Nie, Yuning Mao, Zhe Liu, and Dong Wang. Dr. zero: Self-evolving search agents without training data. *arXiv preprint arXiv:2601.07055*, 2026.

Kongcheng Zhang, Qi Yao, Shunyu Liu, Yingjie Wang, Baisheng Lai, Jieping Ye, Mingli Song, and Dacheng Tao. Consistent paths lead to truth: Self-rewarding reinforcement learning for llm reasoning. *Advances in Neural Information Processing Systems*, 38:59849–59887, 2026a.

Qingyang Zhang, Haitao Wu, Changqing Zhang, Peilin Zhao, and Yatao Bian. Right question is already half the answer: Fully unsupervised llm reasoning incentivization. *Advances in neural information processing systems*, 38:67345–67372, 2026b.

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 *ECCV*, volume 15066, pp. 169–186. Springer, 2024.

Andrew Zhao, Yiran Wu, Tong Wu, Quentin Xu, Yang Yue, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. *Advances in Neural Information Processing Systems*, 38:105816–105879, 2026a.

Shitian Zhao, Haoquan Zhang, Shaoheng Lin, Ming Li, Qilong Wu, Kaipeng Zhang, and Chen Wei. Pyvision: Agentic vision with dynamic tooling. *arXiv preprint arXiv:2507.07998*, 2025.

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

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. In *International Conference on Learning Representations*, volume 2026, pp. 2548–2581, 2026c.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, Fang Wan, et al. Geometric-mean policy optimization. In *International Conference on Learning Representations*, volume 2026, pp. 41354–41374, 2026d.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arxiv preprint arxiv:2507.18071*, 2025.

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

Yifei Zhou, Sergey Levine, Jason Weston, Xian Li, and Sainbayar Sukhbaatar. Self-challenging language model agents. *Advances in Neural Information Processing Systems*, 38:113959–113991, 2026.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Xinwei Long, Ermo Hua, Biqing Qi, et al. TTRL: Test-time reinforcement learning. *Advances in Neural Information Processing Systems*, 38:131459–131483, 2026.

A LLM AND AI USAGE STATEMENT

In this work, we used generative AI tools for improving language clarity and grammar, writing utility scripts for data preprocessing and postprocessing, debugging, and restructuring the code base; core algorithmic implementation was done and reviewed by the authors. We have not used generative AI tools for experimental design, or drafting scientific arguments, and figure generation is not applicable to this work. Outputs from the AI-assisted tools are meticulously reviewed, revised, and verified by the authors, who retain full responsibility for all content presented in this paper.

B PRELIMINARIES

B.1 VISUAL AGENTS

Visual agents extend LLMs with a closed observe–reason–act loop over images: rather than answering in a single pass, they iteratively acquire visual evidence before responding. In this study, we mainly consider the following two types of visual agents, as shown in Fig. 7.

Zoom-in–only agents (e.g., DeepEyes (Zheng et al., 2026)). These agents treat high-resolution understanding as sequential search. Given a downsampled full image, the agent issues zoom-in actions that crop and enlarge regions from the original resolution, progressively resolving fine details (small text, dense charts, tiny UI elements). The minimal action space (zoom-in + answer) makes the policy simple and robust, but offers no computational manipulation — all evidence must be located via spatial attention alone.

Code-execution agents (e.g., PyVision Zhao et al., 2025). Here the action space is a Python interpreter: the agent writes code to crop regions, run OCR, detect contours, sample pixels, or extract chart values, computing visual evidence rather than merely viewing it. This enables precise quantitative answers but risks buggy code, fragile heuristics, and long unproductive traces, complicating control and termination.

B.2 REINFORCEMENT LEARNING WITH GRPO

Both agents are trained with Group Relative Policy Optimization (GRPO) (Shao et al., 2024). For each prompt q, the behavior policy $\pi_{\theta_{\text{old}}}$ samples a group of G responses $\{o_1, \ldots, o_G\}$ with scalar rewards $\{r_1, \ldots, r_G\}$. In place of a learned value function, GRPO uses the group mean as a baseline, normalizing each reward within its group:

$$\hat{A}_i \;=\; \frac{r_i - \text{mean}(\{r_j\}_{j=1}^{G})}{\text{std}(\{r_j\}_{j=1}^{G})}, \tag{7}$$

and maximizes a clipped surrogate over all response tokens:

$$\mathcal{J}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\Big(\rho_{i,t}\,\hat{A}_i,\ \text{clip}\left(\rho_{i,t},\,1-\varepsilon_{\text{lo}},\,1+\varepsilon_{\text{hi}}\right)\hat{A}_i\Big)\right], \tag{8}$$

where $\rho_{i,t} = \pi_\theta(o_{i,t} \mid q, o_{i,<t}) \,/\, \pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})$ is the per-token importance ratio against the behavior policy. We use asymmetric clipping bounds $\varepsilon_{\text{lo}} = 0.2$, $\varepsilon_{\text{hi}} = 0.28$ and no KL penalty against a reference model.

B.3 ASYNCHRONOUS TRAINING

In synchronous on-policy training, each batch is generated by the current policy, $\pi_{\theta_{\text{old}}} = \pi_\theta$, and the importance ratios in Eq. (8) start at 1. In the *asynchronous* variant used for Solver training, generation of the next batch overlaps the gradient step on the current one, so each batch is produced by a policy that lags the learner by one step: for the batch consumed at learner step θ_t,

$$\pi_{\theta_{\text{old}}} \;=\; \pi_{\theta_{t-1}}, \qquad \rho_{i,t} \;=\; \frac{\pi_{\theta_t}(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{t-1}}(o_{i,t} \mid q, o_{i,<t})}. \tag{9}$$

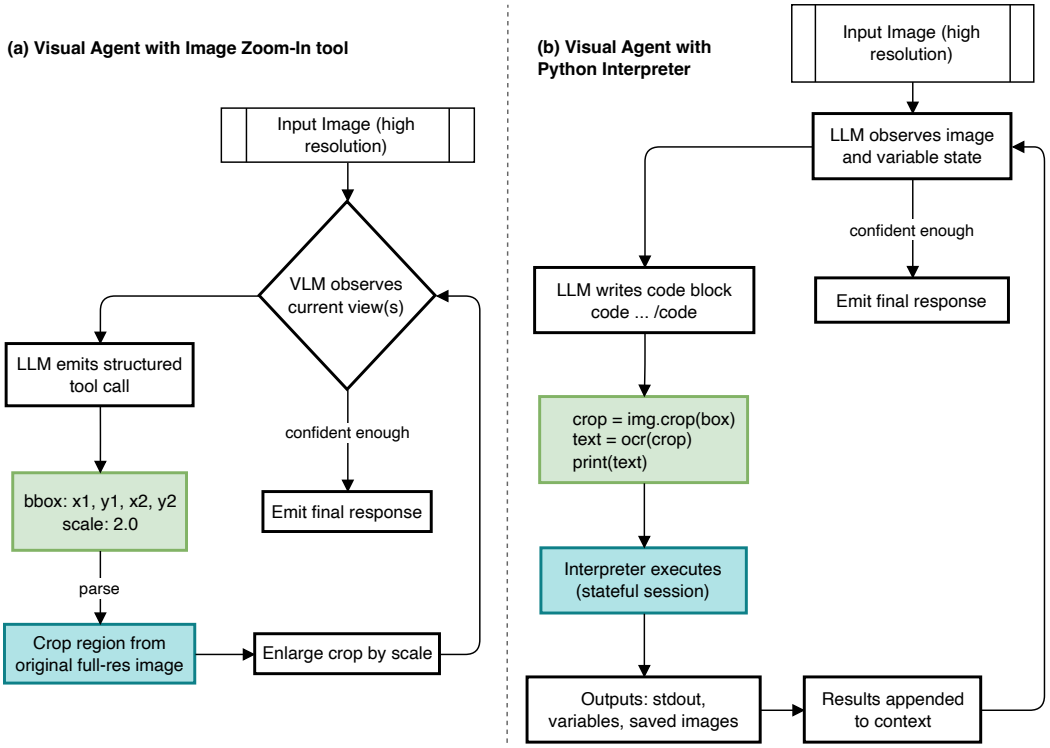


Figure 7: Visual Agent Flowcharts

The objective is otherwise unchanged: the importance ratio in Eq. (8) corrects for the one-step staleness exactly, and the clip bounds the deviation it can introduce. The group-relative advantage of Eq. (7) is unaffected, since it depends only on rewards within a batch, all sampled from the same stale policy.

C Nash Equilibrium Analysis

We formalize the equilibrium characterization stated in Section 2.3. For generality, we use the extended form (Eq. (6)) of the accuracy margin to develop the analysis. Throughout, we regard the alternating optimization in Eq. (1) as a two-player game between the challenger and the solver.

Setup. For notational simplicity, let $q = (I, x, y)$ denote a proposed problem, and define

$$p_i(q;\theta) := \pi_\theta^{(\mathcal{H}_i)}(y \mid I, x), \qquad i \in \{1, 2\}, \tag{10}$$

together with the harness margin

$$\Delta_\theta(q) := p_1(q;\theta) - p_2(q;\theta). \tag{11}$$

Let ρ_ϕ denote the joint distribution over q induced by $I \sim \mathcal{I}$ and $(x, y) \sim \pi_\phi(\cdot \mid I)$.

Two-Player Formulation. Conditioned on a proposed pair (I, x, y), the expected solver accuracy is

$$\mathbb{E}_{\hat{y}\sim\pi_\theta^{(\mathcal{H}_1)}(\cdot|I,x)}\left[\mathbf{1}_{\{\hat{y}=y\}}\right] = p_1(q;\theta). \tag{12}$$

Hence the challenger and solver payoffs can be written as

$$U_{\mathrm{C}}(\phi,\theta) = \mathbb{E}_{q\sim\rho_\phi}\left[\Delta_\theta(q)\right], \quad \text{and} \quad U_{\mathrm{S}}(\phi,\theta) = \mathbb{E}_{q\sim\rho_\phi}\left[p_1(q;\theta)\right], \tag{13}$$

respectively. A pair (ϕ^*, θ^*) is a Nash equilibrium if

$$U_{\mathrm{C}}(\phi^*,\theta^*) \geq U_{\mathrm{C}}(\phi,\theta^*) \quad \text{and} \quad U_{\mathrm{S}}(\phi^*,\theta^*) \geq U_{\mathrm{S}}(\phi^*,\theta) \quad \text{for all admissible } \phi, \theta. \tag{14}$$

Proposition 1 (Support optimality of the challenger)**.** *Fix the solver parameters θ, and let $\mathcal{Q}$ denote the feasible set of proposed problems $q = (I, x, y)$. Assume that the harness margin $\Delta_\theta : \mathcal{Q} \to \mathbb{R}$ is bounded and that*

$$m^*_\theta := \max_{q' \in \mathcal{Q}} \Delta_\theta(q') \tag{15}$$

is attained. Then a challenger distribution ρ_ϕ is a best response to θ if and only if

$$\rho_\phi(\{q \in \mathcal{Q} : \Delta_\theta(q) = m^*_\theta\}) = 1. \tag{16}$$

Proof. For fixed θ, the challenger payoff is $U_{\mathrm{C}}(\phi, \theta) = \mathbb{E}_{q\sim\rho_\phi}[\Delta_\theta(q)]$. Since $\Delta_\theta(q) \le m^*_\theta$ for every q, we immediately have $U_{\mathrm{C}}(\phi, \theta) \le m^*_\theta$. If ρ_ϕ is supported on $\{q : \Delta_\theta(q) = m^*_\theta\}$, then $U_{\mathrm{C}}(\phi, \theta) = m^*_\theta$, so ρ_ϕ is a best response.

Conversely, suppose that ρ_ϕ assigns positive probability to the suboptimal set

$$A := \{q \in \mathcal{Q} : \Delta_\theta(q) < m^*_\theta\}. \tag{17}$$

Then

$$\begin{aligned} U_{\mathrm{C}}(\phi, \theta) &= \int_A \Delta_\theta(q)\, d\rho_\phi(q) + \int_{A^c} \Delta_\theta(q)\, d\rho_\phi(q) \\ &< m^*_\theta \rho_\phi(A) + m^*_\theta \rho_\phi(A^c) \\ &= m^*_\theta. \end{aligned} \tag{18}$$

Thus ρ_ϕ is not a best response. Therefore, a best response must satisfy Eq. (16). □

Solver regularity. Let $\mathcal{A}_\theta := \{q : p_1(q; \theta) < 1\}$ be the set of unsaturated problems of solver $\pi_\theta^{(\mathcal{H}_1)}$. To exclude plateau equilibria at intermediate accuracy, we use the following local regularity condition: if $\rho_\theta(\mathcal{A}_\theta) > 0$, then there exists some update vector v and some $\varepsilon > 0$ such that

$$\mathbb{E}_{q\sim\rho_\phi}\left[p_1(q; \theta + \varepsilon v) - p_1(q; \theta)\right] > 0. \tag{19}$$

If p_1 is differentiable in θ, then it is equivalent to

$$\mathbb{E}_{q\sim\rho_\phi}\left[D_v p_1(q; \theta)\right] > 0 \quad \text{for some } v,$$

where D_v denotes directional derivative. We propose this condition to rule out parameter plateaus within Θ in which a problem remains partially solved only because no feasible solver update can increase its accuracy.

Proposition 2. *Let (ϕ^*, θ^*) be a Nash equilibrium satisfying Prop. 1 and the local regularity condition Eq. (19). Then, for almost every $q \sim \rho_{\phi^*}$,*

$$p_1(q; \theta^*) = 1. \tag{20}$$

Moreover, exactly one of the following two cases holds on the equilibrium support:

1. ***Gap-closed equilibrium.*** *If $m^* = 0$, then*

$$p_1(q; \theta^*) = p_2(q; \theta^*) = 1. \tag{21}$$

   The two harnesses are indistinguishable on the equilibrium problem distribution. In particular, the orchestration behavior exercised by $\mathcal{H}_1$ on these problems has been transferred to $\mathcal{H}_2$ through the shared solver weights.

2. ***Gap-saturated equilibrium.*** *If $m^* > 0$, then*

$$p_1(q; \theta^*) = 1, \qquad p_2(q; \theta^*) = 1 - m^* < 1. \tag{22}$$

   The solver has saturated its accuracy under $\mathcal{H}_1$, while the same weights remain inaccurate under $\mathcal{H}_2$. The residual margin therefore identifies a harness-specific capability gap.

In particular, no positive-mass equilibrium problem satisfies

$$0 < p_1(q; \theta^*) < 1. \tag{23}$$

Proof. Suppose, for contradiction, that there is a positive-measure set of equilibrium-support problems with $p_1(q;\theta^*) < 1$. By Eq. (19), there exists an admissible solver update that strictly increases $\mathbb{E}_q[p_1]$. This update increases $U_{\mathrm{S}}(\phi^*, \theta)$, contradicting the assumption that θ^* is a best response to $U_{\mathrm{S}}(\phi^*, \theta)$. Hence Eq. (20) holds.

Proposition 1 gives $\Delta_{\theta^*}(q) = 1 - p_2(q;\theta^*) = m^*$ throughout the equilibrium support. If $m^* = 0$, then $p_2(q;\theta^*) = 1$, yielding the gap-closed case. If $m^* > 0$, then $p_2(q;\theta^*) = 1 - m^* < 1$, yielding the gap-saturated case. Since probabilities are bounded above by one, no negative-margin case can occur once p_1 is saturated. □

D Implementation Details

D.1 Base Models

To demonstrate that **Marvo** generalizes beyond a specific model family or scale, we instantiate our framework with two base models that differ substantially in architecture, training pipeline, and capacity: Qwen2.5-VL-7B (Bai et al., 2025b) and Qwen3-VL-4B (Bai et al., 2025a).

Although both models follow the three-module VLM paradigm (vision encoder, vision-language merger, and LLM), their designs diverge in nearly every component: Qwen2.5-VL employs a ViT trained from scratch with window attention, compresses every 2×2 group of adjacent patch features into a single token via an MLP merger injected only at the LLM input, and uses MRoPE with contiguous temporal/height/width sections aligned to absolute time; in contrast, Qwen3-VL adopts a pretrained SigLIP-2 encoder continually trained at dynamic resolutions, introduces DeepStack to inject multi-level ViT features into the first several LLM layers, and uses Interleaved-MRoPE together with explicit text-based timestamps for temporal grounding. The LLM backbones also belong to different generations (Qwen2.5 vs. Qwen3) with distinct pretraining corpora and post-training recipes, and the two models differ considerably in capacity (7B vs. 4B).

Importantly, **Marvo** imposes no requirement on the base model beyond the ability to invoke tools and produce verifiable answers—the challenger–solver co-evolution, the accuracy-margin reward, and the GRPO-based updates are all model-agnostic—and as shown in Sec. 3, both models achieve consistent improvements after training, indicating that the gains stem from the paradigm itself rather than from idiosyncrasies of a particular backbone.

D.2 SFT Details

For the PyVision harness, SFT checkpoints serve as the cold-start for self-evolution, as the base model struggles to operate the code interpreter directly. For Qwen2.5-VL-7B, an SFT checkpoint is readily provided by Zhao et al. (2026b); for Qwen3-VL-4B, the base model is instead finetuned on the same SFT data. The cold-start stage is lightweight, aiming only to familiarize the model with the harness.

SFT configuration	
Base model	Qwen3-VL-4B-Instruct
Epochs	1 (shuffled)
Global batch size	16
Micro batch size	1
Steps	430 (6,880 episodes)
Learning rate	1×10^{-5}
LR schedule	cosine, min 1×10^{-6}
Warmup fraction	0.1
Optimizer	Adam (β_1=0.9, β_2=0.95)
Weight decay	0.1
Sequence length	16,384
Vision tower	frozen

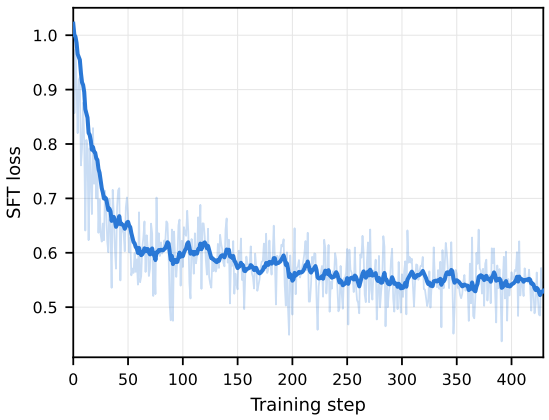


Figure 8: **Left:** SFT hyperparameters. **Right:** SFT training loss.

D.3 RL HYPERPARAMETERS

Table 5 lists the hyperparameters of the accuracy-margin run summarized by Algorithm 1. A skeleton code base is enclosed in the supplementary.

Table 5: Training Hyperparameters.

	Value
Run / data	
Rounds R	3
Images per round B	1,280 (cyclic disjoint slices; pool 44,643, seed 42)
Image long-edge cap	1,536 pixels
Challenger (GRPO)	
Steps per round m	10
Rollout batch size	32
Group size G	4
Learning rate	1×10^{-6}
Rollout temperature	1.0
Max response length	4,096 tokens
Code-turn budget	8 turns
Sandbox timeout	30 s
Reward samples k_r	1 (temperature 1.0, both harnesses)
Parse-failure reward	-1.0
Data filter	
Filter solves k_f	4 (temperature 1.0)
Candidates per image	4
Kept per round	1,280 (cap)
Solver (GRPO)	
Steps per round n	20 (one pass; no epochs)
Rollout batch size	64
Group size G	8
Learning rate	1×10^{-6}
Rollout temperature	1.0
Max response length	8,192 tokens
Max images per episode	20
Tool-use bonus	$+0.2$
Format penalty	-0.5
Evaluation	
Sampling	t=0.6, top-p=1.0, top-k=20, min-p=0
Max tokens	16,000
Max images per episode	20

D.4 EVALUATION BENCHMARKS

We evaluate the solver on seven benchmarks spanning three families: agentic thinking-with-images, high-resolution visual search, and multimodal mathematical reasoning. Unless stated otherwise, we report accuracy with greedy decoding, and we run the solver in its full agentic configuration (i.e., with access to the image zoom-in tool).

TIRBench. TIRBench (Li et al., 2025) is a benchmark for agentic thinking-with-images reasoning. It comprises 13 tasks and 1,215 samples, each designed to be unsolvable by static observation alone and instead requiring multi-step, tool-based image manipulation—e.g., zooming, rotating, contrast enhancement, drawing auxiliary lines, or segmenting and reassembling pieces. Because TIRBench directly measures tool-orchestration capability, it is the primary in-domain benchmark for our method.

HRBench. HRBench (Wang et al., 2024b) targets fine-grained visual perception and spatial reasoning over ultra-high-resolution images (the HRBench-4K subset), where the decisive visual evidence occupies only a small region of a very large image. It stresses precisely the regime in which

a zoom-in tool should help most, and in which naive resizing destroys the information needed to answer.

V* Bench. V*Bench (Wu & Xie, 2024) evaluates visual search over high-resolution, visually crowded images (191 samples, average resolution 2246×1582). It contains two subtasks: attribute recognition (115 samples), which asks for a specific attribute (e.g., color, material) of an object in the scene, and spatial relationship reasoning (76 samples), which asks for the relative position of two objects. Both require the model to localize small, task-relevant regions before answering.

RealWorldQA. RealWorldQA (xAI, 2024) consists of over 700 real-world photographs—largely captured from vehicles and other everyday settings—each paired with a question and a short, easily verifiable answer. It measures practical spatial understanding and grounding in naturalistic scenes.

MathVision. MathVision (Wang et al., 2024a) contains 3,040 competition-level mathematics problems embedded in visual contexts (figures, diagrams, charts), spanning 16 mathematical disciplines across 5 difficulty levels. Problems are drawn from real high-school and university competitions, emphasizing deep reasoning over diagrams rather than perception alone.

MathVerse. MathVerse (Zhang et al., 2024) is an all-around visual math benchmark of 2,612 high-quality multi-subject problems with diagrams. Each problem is human-transformed into six distinct versions that vary the degree of information provided in text versus vision, yielding 15K test instances in total. This design tests whether a model truly interprets the diagram rather than shortcutting from the textual question; we report results under the CoT evaluation protocol.

WeMath. WeMath (Qiao et al., 2024) is a large-scale benchmark of visual math problems organized into 67 hierarchical knowledge concepts. Rather than scoring end-to-end correctness only, it decomposes problems into individual knowledge steps, enabling a fine-grained, process-oriented diagnosis of mathematical reasoning in visual contexts.

D.5 Training Data

We extract images from the RL training dataset from PyVision-RL (Zhao et al., 2026b) and use these unlabeled images as the only inputs throughout Marvo experiments. Due to computation limitation, we filter out over-sized images, resulting a pool of 42,124 images spanning various sources and resolution scales (Fig. 9).

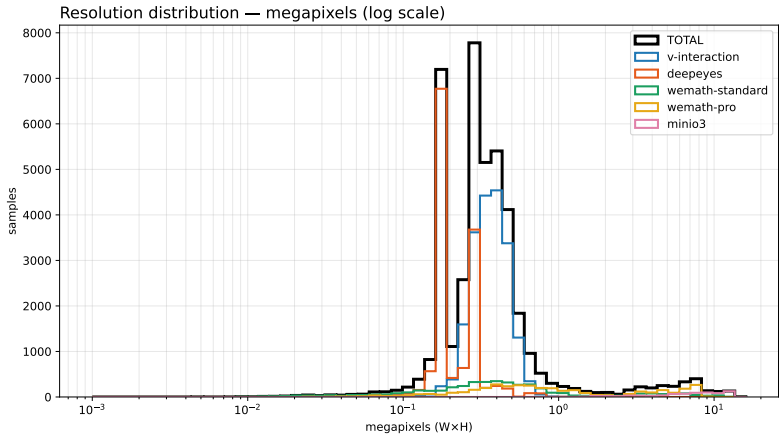


Figure 9: Resolution Distribution of Training Image Pool.

E CHALLENGER PROMPT

Listing 1 shows the complete prompt of Challenger with the PyVision harness Zhao et al. (2025). Following the PyVision-RL wire format, the episode is a *single user message with no system prompt*: the image is embedded in `<image_clue_0>` tags, and the {width}/{height} placeholders are filled with the image's pixel dimensions. The template re-points the reference solver scaffold at question *authoring*: the finalize contract is a single fenced `json` block (question, four options, gold letter) in place of the solver's boxed answer, and the prompt requires the model to inspect at least one cropped region with code before authoring.

Listing 1: The Challenger Prompt.

```
<image_clue_0><image></image_clue_0>

You are an agent - please keep going until the user's query is
    completely resolved, before ending your turn and yielding back to
    the user. Only terminate your turn when you are sure that the
    problem is solved.

Your task: author **exactly one** challenging multiple-choice question
    (four options A-D) about the image. The question must require fine-
    grained visual examination of the image to answer correctly, must
    be answerable purely from the image (no outside knowledge), and the
     gold answer must be unambiguous.

You now have the ability to selectively write executable Python code to
     enhance your inspection of the image. The Python code will be
    executed by an external sandbox.

You MUST plan extensively before each function call, and reflect
    extensively on the outcomes of the previous function calls. DO NOT
    do this entire process by making function calls only, as this can
    impair your ability to explore the image and think insightfully.

For all the provided images, in order, the i-th image has already been
    read into the global variable `image_clue_i` using the PIL.Image.
    open() function. When writing Python code, you can directly use
    these variables without needing to read them again.

You MUST use the python tool (e.g., matplotlib library) to inspect the
    image before authoring the question | zoom into at least one region
     that looks interesting (small text, faces, objects in clutter,
    fine textures, partially occluded items, etc.) and base the
    question on something you observed in the close-up views. This
    includes but is not limited to zooming in, rotating, adjusting
    contrast, computing statistics, or isolating features.

Note that when you use matplotlib to visualize data or further process
    images, you need to use plt.show() to display these images; there
    is no need to save them. Do not use image processing libraries like
     cv2 or PIL. If you want to check the value of a variable, you MUST
     use print() to check it.

The output (wrapped in `<interpreter>output_str</interpreter>`) can be
    returned to aid your inspection. The Python code should be complete
     scripts, including necessary imports.
Each code snippet is wrapped with `<code>
```python
code snippet
```
</code>`.
```

```
*image resolution:*

Image Width: {width}; Image Height: {height}

*your task:*
Explore the image with code first; do NOT propose the final MCQ until
    you have inspected at least one cropped region with the tool. When
    | and only when | you are ready to finalize, stop writing code and
    output a single fenced ```json ... ``` block with keys "question" (
    question text only, no options), "options" (A-D), "answer" (one
    letter):

```json
{
  "question": "<question text only, no options>",
  "options": {"A": "...", "B": "...", "C": "...", "D": "..."},
  "answer": "<one of A, B, C, D>"
}
```

The JSON block is the signal that you are done; do not emit it earlier.
```

F Analysis

F.1 Per-Round Data Distribution

We sample 100 images from the question pool used in each training round and 100 unique images from each evaluation benchmark, deduplicating benchmarks that attach multiple questions to one image. Every image is encoded independently by the frozen Qwen3-VL-4B vision tower, and we take the mean of its merged patch embeddings followed by L2 normalization as the image representation. All 1000 embeddings are then projected jointly: PCA first reduces the 2560-dimensional vectors to 50 dimensions, and t-SNE (perplexity 30, PCA initialization) maps them to two dimensions. Because a single shared projection is fit on all points, coordinates are directly comparable across the three per-round panels, which differ only in which round's generated images are highlighted.

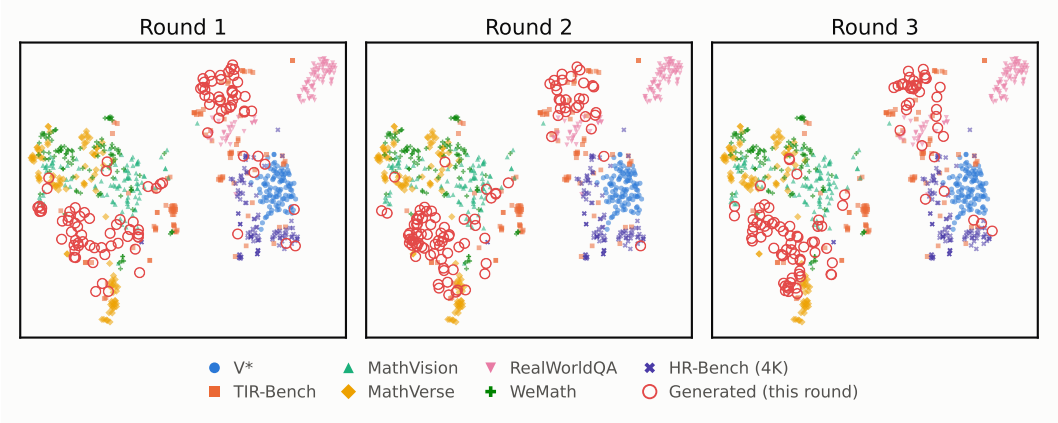


Figure 10: **Image-embedding space of generated questions vs. evaluation benchmarks across training rounds.** Each panel shows a shared 2D projection (PCA + t-SNE over Qwen3-VL-4B vision-tower embeddings) of 100 images sampled from each of seven evaluation benchmarks (colored points) and 100 images underlying the questions generated in the indicated training round (red rings).

In each round, sampled images concentrate in the math-diagram region and near TIR-Bench, while the natural-image benchmarks form distinct clusters that the generated distribution largely does not cover, indicating that evaluation gains on these benchmarks arise from transfer rather than distributional overlap. The generated distribution remains broadly stable across rounds, with a mild shift toward diagram-style content in rounds 2–3. As these images constitute the sole input information during Marvo, their distribution naturally correlates with the per-round improvement. For example, the RealworldQA cluster is isolated from the others and is not covered by the training images, resulting in the smallest performance gains among all benchmarks.

F.2 Iteration Scaling

We continue to scale the evolution rounds of Marvo to 8 rounds. As reported in Tab.6 instead of consistent improvement, Marvo evolution suffers with a degradation trend across evaluated tasks, similar to the scaling bound observed in R-Zero (Huang et al., 2026).

Concretely, most benchmarks reach their peak at round 5 or 6 and deteriorate thereafter. MathVision-Mini drops steadily from 27.3 at round 5 to 23.4 at round 7, recovering only marginally to 23.7 at round 8. V* peaks at 78.5 in round 6 before falling sharply to 72.3 in round 7, and V* and MathVision-Mini both end round 8 well below their round-5 scores. Text-centric benchmarks degrade more mildly: TIR-Bench declines from 21.7 to 21.0, while WeMath oscillates in the 62–64 range and never exceeds its round-5 score by a meaningful margin.

	Round 5	Round 6	Round 7	Round 8
WeMath	63.2	62.1	64.2	64.3
MathVision-Mini	27.3	24.0	23.4	23.7
V*	77.5	78.5	72.3	74.4
TIR-Bench	21.7	20.7	21.1	21.0

Table 6: **Performance of Marvo across evolution rounds 5–8 on four benchmarks.** Scores are reported in percentages; the best result for each benchmark is achieved no later than round 6, after which performance degrades.

We attribute the instability and degradation under iteration scaling to the quality degradation of the generated solver training data. To verify this hypothesis, we use Qwen3.8-27B (Qwen Team, 2026) to probe the accuracy of the proposed questions $\{(I_i, x_i, y_i)\}_{i=1}^{N}$ at each round. Note that even though the proposed labels are guaranteed by the consensus between the challenger and the solver, Table 7 shows that the proposed questions are in fact getting worse: judge accuracy starts at 68.5% in round 1, remains stable around 62–67% through round 4, and then drops sharply to 58.7% at round 5 and below 47% from round 6 onward, ending at 43.6% in round 8. This decline closely mirrors the benchmark trend in Table 6, where performance peaks around rounds 5–6 and deteriorates thereafter, suggesting that the scaling bound is driven by the decay of training signal quality rather than by optimization limits. R-Zero relies on an additional round of majority voting from solver answers to generate the answer y, but as both models drift away from the true data distribution over iterations, the consensus between two degraded models is no longer a faithful filter, and it suffers from the same degradation.

Round	1	2	3	4	5	6	7	8
n	1201	1280	1281	1280	1281	1280	1280	1280
Judge Acc. (%)	68.5	62.2	66.1	67.1	58.7	46.1	46.2	43.6

Table 7: Accuracy of the proposed solver training questions $\{(I_i, x_i, y_i)\}_{i=1}^{N}$ at each evolution round, probed by Qwen3.8-27B. Despite consensus-based filtering, question quality degrades from 68.5% to 43.6%, mirroring the performance degradation in Table 6.

G CASE STUDY

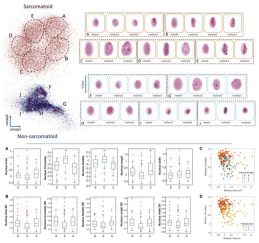
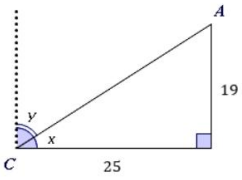
G.1 CHALLENGER EXAMPLE TRACE

We present a full real trajectory in Fig. 11 about how a challenger equipped with PyVision harness proposes a visual question pair from an unlabeled image.

G.2 EXAMPLE PROPOSED QUESTIONS

We present some of proposed questions on the same set of images in Tab. 8. As Marvo proceeds, the complexity of proposed question increases.

Table 8: Proposed Questions.

Input Image	Round 1	Round 2	Round 3
	Which cluster in the Sarcomatoid section shows nuclei with the most consistent, round shape based on the mean nucleus images? (A) Cluster A (B) Cluster B (C) Cluster C (D) Cluster D	Which of the following statements is true based on the visual data in the image? (A) In the sarcomatoid region, the nucleus circularity for group A is consistently higher than for group B. (B) The non-sarcomatoid region (F-J) shows a wider range of nucleus solidity values compared to the sarcomatoid region (A-E). (C) The scatter plot in panel D shows that predicted labels for 'E' are clustered with ground truth labels for 'S'.	In region E, which nucleus image shows the most irregular shape compared to the others? (A) meanE (B) nucleus1 (C) nucleus2 (D) nucleus3
	What number is visible on the blue plaque affixed to the building wall? **(A)** 7 (B) 8 (C) 9 (D) 6	What is the distinct emblem located in the center of the car's front grille? **(A)** A circular BMW logo with a blue and white checkered pattern (B) A rectangular emblem with vertical slats (C) A silver shield with a stylized eagle (D) A black and white checkered flag design	Which of the following is NOT a visible feature on the street sign attached to the pole in the image? (A) A vertical metal pole with three horizontal brackets (B) A house number plaque reading '7' (C) A decorative arched wooden door with carvings **(D)** The text 'BMW' on the front grille of the green car
	Which arc is above the hypotenuse, and which is below? (A) x is above, y is below **(B)** y is above, x is below (C) x is above, x is below (D) y is above, y is below	Based on the image, which angle is adjacent to the side of length 19? (A) x (B) y (C) Both are adjacent to the side of length 19 **(D)** Neither is adjacent to the side of length 19	What is the value of angle x (at C) in degrees? **(A)** 38.9 (B) 51.1 (C) 45 (D) 60



Proposed Question

What language is visible on the text of the books on the second shelf from the top?

A. English. B. French C. Japanese D. Chinese

Let's proceed step by step:

Step 1: Plan
- The user wants a challenging multiple-choice question about the image, requiring fine-grained visual examination.
- The question must be answerable purely from the image, and the answer must be unambiguous.
- I will inspect the image for details that require close-up or zoom-in, such as small text, faces, objects, or fine textures.
- The image shows a modern interior with a fireplace, bookshelves, and sculptures. To make the question challenging, I will look for subtle or small details that are not immediately obvious.

Step 2: Zoom In on a Region
I will zoom in on the top left corner of the bookshelf, where there is a sculpture and some books, to check for any small text or distinctive features.

```
<code>
import matplotlib.pyplot as plt

# Crop the top left corner of the bookshelf (where the
sculpture is)
# Approximate coordinates based on image size and layout
left = 350; upper = 110; right = 650; lower = 320

cropped = image_clue_0.crop((left, upper, right, lower))
plt.imshow(cropped)
plt.axis('off')
plt.show()
</code>
```



```
<code>
# Crop the second shelf of books (from the top)
# Approximate coordinates for the second shelf
left = 350; upper = 310; right = 650; lower = 420

cropped_books2 = image_clue_0.crop((left, upper, right,
lower))

plt.imshow(cropped_books2)
plt.axis('off')
plt.show()
</code>
```



Reflection: Upon zooming in on the second shelf of books, it is clear that the text on the spines of the books is in Chinese characters. The characters are relatively small but visible, and there are distinct names and titles. This is a fine-grained visual detail that requires close inspection.

Question Planning: A challenging question can be: "Which of the following is the visible language of the text on the books on the second shelf from the top?" The options should include Chinese, English, French, and Japanese.

Final Step: Let's finalize the question and options in json, based on the evidence from the image.

```
{
  "question": "What language is visible on the text of the books on the second shelf
from the top?",
  "options": {"A": "English", "B": "French", "C": "Japanese", "D": "Chinese"},
  "answer": "D"
}
```

Figure 11: Example Trajectory of Challenger Agent.