SELF-VERIFYING VISION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning with verifiable rewards has emerged as a powerful paradigm for improving reasoning in structured domains such as mathematics and code, where deterministic checkers can directly certify generated solutions. Extending this paradigm to visual reasoning is difficult because correctness requires interpreting unstructured visual evidence, and ground-truth is scarce at scale. We introduce a training framework for vision-language models (VLMs) that makes verification a learned capability rather than relying on exact rewards to verify final answers. The model generates a candidate solution, critiques its own reasoning, and then refines its answer, while a simple proxy reward evaluates the resulting generation–verification pair. We find that jointly training for generation and verification creates a positive feedback loop during training that yields more scalable visual reasoning than optimizing answer correctness alone, even when self-verification is disabled at test time. Across diverse benchmarks, our framework improves the base VLM, outperforms answer-supervised training, and surpasses test-time scaling baselines, with the largest gains on 3D-grounded reasoning.

1 INTRODUCTION

Visual reasoning – inferring spatial relationships and object attributes from visual input – benefits from a tight interplay between generation and verification. Consider the query in Figure 1: a strong VLM, Qwen3-VL, answers using the typical size of a piano stool, while ignoring the sofa as the relevant reference for scale. The initial answer is wrong, but the underlying error is recognizable: a verifier that re-examines the reasoning against the image can flag the missing reference and prompt an appropriate correction. Generation proposes, verification catches what generation missed, and the two together reach an answer that neither produces alone.

This interplay is well studied in large language models (LLMs) (Li et al., 2023; Venkatraman et al., 2025; Song et al., 2024) and established reinforcement learning with verifiable rewards (RLVR) (Jaech et al., 2024; Guo et al., 2025). What makes RLVR effective in structured text domains is that verification can be made exact: a calculator, a deterministic checker, or code execution can decide with certainty whether a generated solution is correct, and that decision serves directly as the reward (Shao et al., 2024; Team et al., 2025b; Zeng et al., 2025).

Visual reasoning admits no such exact verifier. Deciding whether a spatial answer is correct requires carefully interpreting unstructured visual evidence, for which no deterministic checker exists. Ground-truth answers could instead serve as verifiers, but are scarce at scale and especially hard to obtain for 3D visual reasoning, where labels are particularly costly. Prior work therefore either invokes external and costly vision specialists at test time (Surís et al., 2023; Gupta & Kembhavi, 2023; Marsili & Gkioxari, 2025) or falls back on supervised training, whose scalability is bounded by the scarcity of diverse annotated data (Sarch et al., 2025; Fan et al., 2025).

We take a different route. Since visual reasoning provides no external signal that can reliably certify a final answer, we make verification a learned model capability. The model generates a candidate solution, critiques its own reasoning, and uses that feedback to refine the answer. *We optimize this self-verification jointly with generation*. Because no exact reward exists, training uses a simple proxy reward over the resulting generation–verification pairs: we evaluate whether the model's feedback is useful and whether acting on it improves the solution. This induces a positive feedback loop – better verification refines generation, while better generations give verification stronger inputs to work 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.

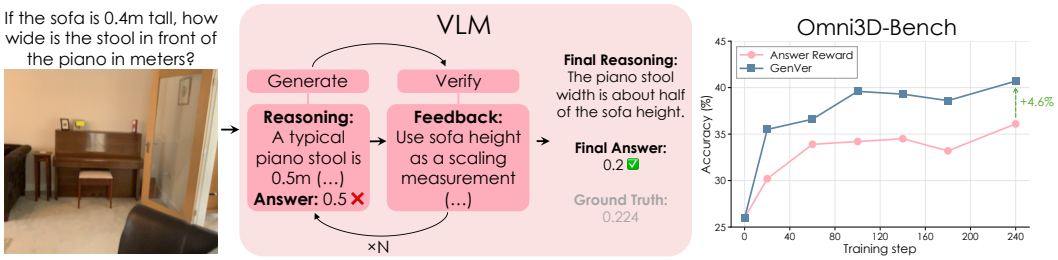

Figure 1: **Left:** We introduce GenVer, a training framework that learns verification jointly with generation, enabling VLMs to critique and refine their own solutions. **Right:** Joint generation and verification training yields more scalable visual reasoning than optimizing answer correctness.

Our framework, GenVer (Figure 1, **left**), requires no ground truth and internalizes verification, avoiding supervised data scarcity and the need for external specialists at inference. Across visual reasoning benchmarks, GenVer improves performance over both the base VLM and answer-supervised training (Figure 1, **right**). On complex queries, GenVer also surpasses traditional test-time scaling baselines, with the largest gains on 3D-grounded reasoning. These performance gains persist even without self-verification at inference, showing that jointly training generation and verification is more scalable than optimizing answer correctness alone when exact rewards are unavailable.

2 Related Work

Visual reasoning in VLMs. Large-scale pretraining on paired text–image data has enabled VLMs to excel at general purpose image understanding. State-of-the-art results are currently dominated by proprietary models (OpenAI, 2025a; Comanici et al., 2025; OpenAI, 2025b), although recent open-source models (Team et al., 2024; Clark et al., 2026; Bai et al., 2025) have reduced the performance gap significantly. Despite recent progress, existing VLMs still struggle with complex spatial reasoning tasks (Rahmanzadehgervi et al., 2024; Tong et al., 2024a; Thrush et al., 2022; Chen et al., 2024), particularly in 3D visual reasoning (Marsili et al., 2025; Jia et al., 2025).

Post-training for visual reasoning. Post-training improves VLMs by equipping them with task-specific capabilities and better aligning them with human intent. In visual reasoning, supervised fine-tuning (SFT) and distillation methods (Xu et al., 2025; Hu et al., 2024; Sarch et al., 2024; Li et al., 2024) typically transfer task-specific skills from annotated datasets or the responses of teacher models, but often lead to forgetting and degraded generalization (Shenfeld et al., 2025; Chu et al., 2025). Reinforcement learning alleviates the issue of forgetting (Shenfeld et al., 2025), with recent approaches interleaving visual grounding and text-based reasoning (Fan et al., 2025; Sarch et al., 2025; Dong et al., 2025; OpenAI, 2025b). These methods rely on ground-truth annotations during model training, whereas GenVer does not require ground-truth and outperforms them.

Training with verifiers. LLM verifiers have been employed to supervise training in text-based domains (Zhang et al., 2025a; Yuan et al., 2024). Self-verification methods such as SCoRe (Zweiger et al., 2025), SQLM (Chen et al., 2025b), and RISE (Qu et al., 2024) generate synthetic training data for reinforcement learning. ReVISE (Lee et al., 2025) adopts a curriculum training approach to teach an LLM to self-verify and subsequently improve text-based reasoning. SWiRL (Goldie et al., 2025) uses external high-capacity verifiers to improve low-capacity models through multi-step reinforcement learning. In contrast, approaches for verifier-based training in visual reasoning remain comparatively limited. VALOR (Marsili & Gkioxari, 2025) uses external LLM and VLM verifiers during post-training to improve visual reasoning, but requires costly tools for inference.

Test-time scaling improves LLM performance by allocating additional inference compute to improve answer quality without changing model weights (Zhang et al., 2025b). Traditional approaches include majority-voting (Wang et al., 2022) and using the same LLM as a judge (Zheng et al., 2023). In text-based domains, self-verification has been used as a framework for test-time scaling via search (Zhao et al., 2025) and iterative refinement (Chen et al., 2025a; Liu et al., 2025; Weng et al., 2023). Our work focuses on training a VLM to self-verify in the unstructured domain of visual reasoning.

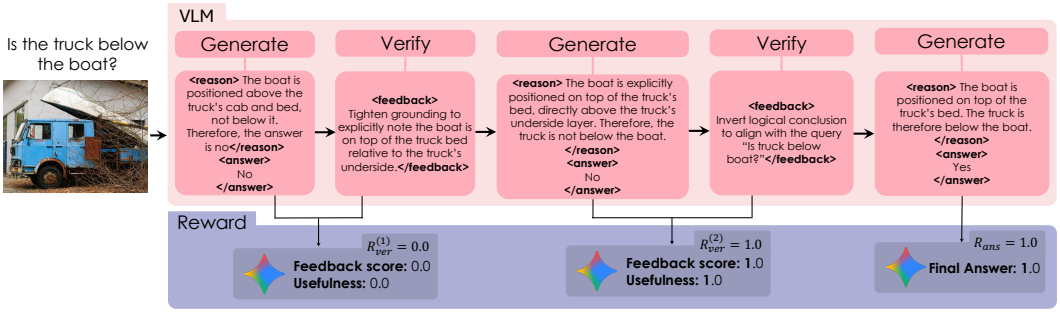

Figure 2: **Method overview.** GenVer jointly improves generation and verification by rewarding both final answer correctness and the quality of self-generated feedback.

3 Self-Verifying Vision Language Models

We improve visual reasoning in VLMs by jointly strengthening generation and verification. Our method, shown in Figure 2, trains a model to produce and refine solutions through self-verification. Rather than optimizing only for answer correctness, we treat self-verification as a core capability and optimize it directly as part of training. This joint training objective induces a mutually reinforcing process where improved verification enhances generation quality, and vice versa.

3.1 Preliminaries: Training VLMs with RL

We begin by introducing Group Relative Policy Optimization (GRPO; Guo et al. (2025)), a policy gradient method for tuning VLMs with RL which we adapt for our setup (§3.2). Our goal is to improve a VLM π_θ, parametrized by θ, for the task of visual reasoning. Given an image-query pair (I, q), we sample G responses from π_θ: $\{\tau^{(i)}\}_{i=1}^{G} \sim \pi_\theta(\cdot|I, q)$ and compute centralized advantages using a reward function R:

$$A^{(i)} = R\big(I, q, \tau^{(i)}\big) - \frac{1}{G}\sum_{j=1}^{G} R\big(I, q, \tau^{(j)}\big) \tag{1}$$

The GRPO objective then maximizes expected advantages while limiting policy drift:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\left(\min\left(s^{(i)}A^{(i)}, \text{clip}\Big(s^{(i)}, 1-\epsilon, 1+\epsilon\Big)A^{(i)}\right) - \beta\text{KL}[\pi_\theta||\pi_{\text{ref}}]\right) \tag{2}$$

where $s^{(i)} = \frac{\pi_\theta(\tau^{(i)}|I,q)}{\pi_{\text{old}}(\tau^{(i)}|I,q)}$ is the importance ratio, ϵ is the clipping parameter, β is the KL penalty coefficient, and π_{ref} is the pre-trained base policy.

In structured domains like mathematics and coding, the reward R typically compares the response to a known answer, or executes a deterministic verifier. However, for visual reasoning, we do not have access to either. This motivates learning to *self-verify* intermediate predictions, which is the core idea of our approach, which we introduce next.

3.2 GenVer: Training with Self-Verification

Unlike traditional VLM tuning approaches that optimize only for correct final answers, our goal is to improve visual reasoning by jointly enhancing a model's ability to generate *and* self-verify solutions. Rather than treating verification as an external process, we model it as a capability learned alongside generation. Concretely, the model produces candidate solutions and iteratively refines them through self-verification, learning to critique and improve its outputs. This leads to training objectives that jointly optimize generation quality and verification ability. Our approach requires no ground truth and instead relies on a high-capacity VLM $\mathcal{V}$, used *only during training*, acting as a *proxy* reward model.

Setup. Given an image-query pair (I, q), our VLM π_θ generates a reasoning trace and answer $(r_0, a_0) \sim \pi_\theta(\cdot|I, q)$. The model iteratively refines this solution via self-verification. At each round t, it produces feedback $f_t \sim \pi_\theta(\cdot|I, q, r_t, a_t)$ and uses this to generate a revised solution $(r_{t+1}, a_{t+1}) \sim \pi_\theta(\cdot|I, q, r_t, a_t, f_t)$. After N rounds, this process yields a trajectory $\tau = (r_0, a_0, f_0, r_1, a_1, \ldots, f_{N-1}, r_N, a_N)$, where a_N is taken as the final model prediction.

Training. We optimize π_θ with a multi-objective setup that jointly optimizes answer generation and answer verification. Concretely, we define a GRPO objective for each component:

$$\mathcal{L}(\theta) = \lambda_{\text{ans}}\mathcal{L}_{\text{ans}}(\theta) + \lambda_{\text{ver}}\mathcal{L}_{\text{ver}}(\theta) \tag{3}$$

where each term corresponds to a distinct reward function defined over an output trajectory τ. This formulation directly aligns training with the desired skills.

We observe that early in training, the model's self-verification outputs are unreliable and can degrade answer quality. To stabilize learning, we introduce a simple curriculum: when sampling τ, we replace the model's self-verification feedback with feedback produced by $\mathcal{V}$ with probability p, decayed linearly from 0.8 to 0 during the first epoch. This prevents poor self-verification from hindering solution generation early in training.

Rewards. We design a unique reward model for each objective, spanning generation and self-verification quality. We additionally impose a format constraint: samples that do not adhere to the required output format receive zero reward across both objectives. We detail both rewards below, with additional details in Appendix A.4.3.

Answer Quality (R_{ans}): With no access to an exact reward model, the high-capacity VLM $\mathcal{V}$ provides a proxy reward, grading the model's solution, a_N, on a 0-10 scale, normalized to $[0, 1]$: $R_{\text{ans}}(\tau) = 0.1 \cdot \mathcal{V}(I, q, a_N)$. $\mathcal{V}$ provides a coarse signal that filters egregious errors, but is not reliable alone (Appendix A.2), motivating the complementary verification objective.

Answer Verification Quality (R_{ver}): Lacking an exact answer reward, we introduce a reward to improve the model's self-verification capability. For each round t, let (r_t, a_t) be the current solution, f_t the model's feedback, and (r_{t+1}, a_{t+1}) the refined solution. We prompt $\mathcal{V}$ to score the feedback f_t given the current solution: $s^{(t)}_{\text{edit}} = \mathcal{V}(f_t|I, q, r_t, a_t)$ where $s^{(t)}_{\text{edit}} \in [0, 1]$. Early in training, when policy feedback at round t is replaced with feedback from $\mathcal{V}$, the verification reward is purely score based: $R^{(t)}_{\text{ver}} = s^{(t)}_{\text{edit}}$. When policy feedback is used, we use $\mathcal{V}$ to score both solutions: $s_t = \mathcal{V}(a_t|I, q)$, $s_{t+1} = \mathcal{V}(a_{t+1}|I, q)$ and define usefulness $u^{(t)}$ to assess if feedback improves solutions or does not degrade correct solutions:

$$u^{(t)} = \begin{cases} 1.0 & \text{if } s_{t+1} > s_t \\ 1.0 & \text{if } s_{t+1} = s_t \text{ and } s_t > 0.7 \\ 0.3 & \text{if } s_{t+1} = s_t \text{ and } s_t \leq 0.7 \\ 0.0 & \text{if } s_{t+1} < s_t \end{cases}$$

The reward at t is defined as $R^{(t)}_{\text{ver}} = 0.7s^{(t)}_{\text{edit}} + 0.3u^{(t)}$. We take the mean over all rounds to obtain the final reward over the trajectory τ: $R_{\text{ver}}(\tau) = \frac{1}{N}\sum_{t=0}^{N-1} R^{(t)}_{\text{ver}}$.

Training data. We train on pairs $\{(I_j, q_j)\}_{j=1}^{D}$ of images I and queries q. As GenVer requires no ground-truth responses, we follow Marsili & Gkioxari (2025) and synthesize visual reasoning queries from image corpora. Specifically, we sample images from SA-1B (Kirillov et al., 2023) and the COCO (Lin et al., 2014) training split, and prompt Gemini-3-Flash (Google, 2025) to generate a visual reasoning query. Importantly, Gemini only generates a query, not a corresponding answer or reasoning trace. We generate 4000 queries, evenly split between COCO and SA-1B images to cover diverse indoor and outdoor scenes. Additional details are in Appendix A.4.2.

Implementation. We use Qwen3-VL 4B (Bai et al., 2025) as our base VLM π_θ and Gemini-3-Flash (Google, 2025) as our reward model for all experiments. We train on 8 A100 GPUs for 2 epochs with batch size 32, group size $G = 8$, learning rate 5×10^{-6}, and $N = 1$ self-verification rounds. We build on verl (Sheng et al., 2025) for training. All hyperparameters are in Appendix A.5. We report results for GenVer with a Molmo2 (Clark et al., 2026) base VLM in Appendix A.1.

	Omni3D-Bench	Robo Spatial	Realworld QA	Omni Spatial	CV-Bench	BLINK	VSR
Proprietary VLMs							
Gemini-3-Flash	62.3	74.3	84.3	63.3	89.8	91.6	82.0
GPT-5-mini	40.9	76.3	62.2	54.9	86.0	81.0	84.4
Open-Source VLMs							
Molmo2 4B	37.0	66.3	66.4	44.6	75.2	74.1	63.4
Gemma3 4B	22.0	61.0	45.8	40.8	53.7	65.7	58.0
RL-Tuned VLMs							
GRIT 3B	27.8	58.8	55.6	**47.2**	68.3	70.3	77.6
ViGoRL 7B	27.5	64.9	52.8	39.5	81.1	68.4	**79.1**
VALOR 8B	-	69.5	57.3	-	-	69.2	75.6
Qwen3-VL 4B							
Base Model	26.0	60.9	63.1	42.3	75.2	81.1	68.6
Answer Reward	36.1	81.3	67.4	46.1	81.1	83.1	76.8
GenVer	**40.7**	**81.5**	**69.7**	**47.2**	**81.6**	**83.8**	75.6

Table 1: **Accuracy (%) on visual reasoning benchmarks.** We highlight the **best** and second best open-source model. We report GenVer with $N = 1$ round of test-time self-verification. We omit VALOR on Omni3D-Bench as it was trained on this dataset.

4 Experiments

We evaluate GenVer across a suite of visual reasoning benchmarks. Our goals are threefold: (1) assess whether jointly training for generation and self-verification yields more scalable improvements than training for generation alone, (2) compare GenVer to state-of-the-art approaches in visual reasoning, and (3) explore how self-verification behaves at test-time.

Evaluation benchmarks. We experiment on visual reasoning benchmarks covering different skills: Omni3D-Bench (Marsili et al., 2025), RoboSpatial (Song et al., 2025), and CV-Bench (Tong et al., 2024a) for 3D understanding; RealWorldQA (xAI, 2024), BLINK (Fu et al., 2024), and VSR (Liu et al., 2023) for spatial relationships; and OmniSpatial (Jia et al., 2025) for logic and grounded common knowledge. Additional details are in Appendix A.3.

Baselines. We compare against four families of methods:

(1) Direct prediction VLMs: We prompt pre-trained VLMs (Google, 2025; OpenAI, 2025a) to reason and predict answers given an image-query pair. This comparison evaluates GenVer against standard direct inference with both proprietary and open-source models.

(2) Answer reward training: We train Qwen3-VL 4B (Bai et al., 2025) with the same rollout structure as GenVer, including self-verification, but using rewards based only on final answer correctness. This compute-matched comparison isolates the benefit of jointly optimizing self-verification over training with answer rewards alone, while keeping the rollout structure fixed.

(3) RL-tuned VLMs: We compare to recent post-training baselines for improved visual reasoning – Sarch et al. (2025); Fan et al. (2025) jointly ground and reason, while Marsili & Gkioxari (2025) decouple reasoning from grounding via tool use.

(4) Test-time scaling: We sample K solutions from pre-trained VLMs and select the final answer via majority voting (Wang et al., 2022) or using an LLM as a judge (Zheng et al., 2023). This comparison contrasts self-verification with traditional test-time scaling approaches.

4.1 Main Results

Table 1 compares GenVer across visual reasoning benchmarks. We use default sampling and report GenVer with $N = 1$ test-time self-verification rounds. We use LMMs-Eval for evaluations (Zhang et al., 2024). For answer reward training and GenVer, we report means over three seeds.

Our framework improves visual reasoning. Our direct comparisons show consistent gains over the pre-trained Qwen3-VL model from training with self-verification. GenVer improves performance



Figure 3: **Outputs for GenVer with $N=1$ self-verification rounds.** For each example we show the query, original generation, proposed feedback and final response.

across all benchmarks, with the largest gains on complex, 3D-grounded reasoning: $+14.7\%$ on OMNI3D-BENCH (26.0% to 40.7%), $+20.6\%$ on ROBOSPATIAL (60.9% to 81.5%), and $+6.6\%$ on REALWORLDQA (63.1% to 69.7%). Notably, GenVer achieves state-of-the-art performance among open-source models on OMNI3D-BENCH. These results indicate jointly optimizing generation and verification strongly improves complex visual reasoning.

Qualitative examples. We present qualitative results in Figure 3. We show the original input query to GenVer and $N=1$ rounds of feedback. We highlight queries where self-verification steers the model toward correct responses. We find that self-verifier feedback consistently identifies the key reasoning failures and provides targeted corrections. In (a) the generation initially focuses on what lies directly beneath the TV and predicts the glass table, while the feedback redirects reasoning to consider forward fall direction, correctly identifying the lamp as the first collision target. In (b) the feedback guides the model to correctly incorporate the grounding measurement provided in the query. In (c), the verifier encourages the model to move beyond uncertainty and estimate object dimensions directly from the image. In (d), the feedback prompts the model to distinguish perceived color from lighting artifacts, leading to the correct identification of the marking as white. These examples display GenVer's ability to both diagnose reasoning errors and incorporate feedback to improve predictions.

	OMNI3D-BENCH	ROBO SPATIAL	REALWORLD QA	OMNI SPATIAL	CV-BENCH	BLINK	VSR
Qwen3-VL 4B	26.0	60.9	63.1	42.3	75.2	81.1	68.6
Answer Reward (direct)	37.9	78.9	64.6	44.9	78.5	**84.6**	**81.1**
GenVer ($N=0$)	**40.4**	**79.6**	**67.5**	**45.1**	**82.8**	83.9	77.5

Table 2: **Answer-only comparisons.** We compare GenVer with $N=0$ to Answer Reward (direct), a direct prediction variant of Answer Reward that is trained without self-verification. Jointly optimizing generation and verification improves answer generation ability.

	OMNI3D-BENCH	ROBO SPATIAL	REALWORLD QA	OMNI SPATIAL	CV-BENCH	BLINK	VSR
Qwen3-VL 4B	26.0	60.9	63.1	42.3	75.2	81.1	68.6
SFT Distillation	39.3	78.1	60.7	45.9	**82.2**	83.3	**76.9**
GenVer	**40.7**	**81.5**	**69.7**	**47.2**	81.6	**83.8**	75.6

Table 3: **Comparison to Supervised Fine-Tuning.** We compare GenVer to a supervised fine-tuning distillation baseline using responses from the VLM used as our proxy reward.

Self-Verification vs. Answer-Only. We compare GenVer to two Answer Reward controls to isolate the effect of adding a self-verification objective. In Table 1, Answer Reward uses the same rollout structure as GenVer but optimizes only final-answer correctness, providing a compute-matched comparison. This controls for the additional inference steps introduced by self-verification and isolates the contribution of the verification objective itself. GenVer outperforms on several reasoning-heavy benchmarks, including OMNI3D-BENCH (40.7% vs. 36.1%), REALWORLDQA (69.7% vs. 67.4%), and OMNISPATIAL (47.2% vs. 46.1%), indicating the gains are not simply due to longer rollouts. In Table 2, Answer Reward (direct) removes self-verification from both training rollouts and objective, matching GenVer with $N=0$ inference. Under this inference-aligned comparison, GenVer still outperforms direct Answer Reward training across most benchmarks: 82.8% vs. 78.5% on CV-BENCH, 67.5% vs. 64.6% on REALWORLDQA, and 40.4% vs. 37.9% on OMNI3D-BENCH. These results highlight that learning to verify improves answer generation itself, even in the absence of test-time refinement. In contrast, on the simpler benchmarks VSR and BLINK, answer-only training performs slightly better. We attribute this to over-verification, where unnecessary refinement introduces errors for simple queries, a failure mode we discuss further in Section 4.3.

Comparison to Distillation via SFT. We compare to a supervised distillation baseline using Gemini-3-Flash, the VLM used as our proxy reward, in Table 3. We generate Gemini responses on the training queries used for GenVer and fine-tune Qwen3-VL on these outputs with supervised fine-tuning for the same number of steps. GenVer outperforms this baseline on most benchmarks, particularly reasoning-intensive tasks such as REALWORLDQA (69.7% vs. 60.7%), OMNI3D-BENCH (40.7% vs. 39.3%), and ROBOSPATIAL (81.5% vs. 78.1%). These results, together with examples where GenVer succeeds despite rejection by the proxy reward model (Appendix A.2), suggest that the gains are not simply due to imitating Gemini, but arise from jointly optimizing generation and verification.

Comparison to RL-tuned VLMs. We compare GenVer to prior post-training approaches for visual reasoning. GRIT (Fan et al., 2025) and ViGoRL (Sarch et al., 2025) rely on labeled data, while GenVer does not require ground truth answers. VALOR (Marsili & Gkioxari, 2025) also does not require ground truth, but unlike GenVer requires invoking costly tools at inference. Nevertheless, GenVer outperforms all methods on reasoning-heavy benchmarks, with the largest gains on REALWORLDQA (69.7% vs. 57.3% for VALOR) and OMNI3D-BENCH (40.7% vs. 27.8% for GRIT). As noted in (Marsili & Gkioxari, 2025), we observe that GRIT and ViGoRL struggle to reason over grounding measurements. These results suggest teaching VLMs to self-verify is a promising path for scaling visual reasoning in the absence of ground-truth annotations or domain-specific tools.

Comparison to direct-prediction VLMs. Though not our focus, we compare GenVer to open-source and proprierary pre-trained direct-prediction VLMs. GenVer outperforms Molmo2 4B (Clark et al., 2026) and Gemma3 4B (Team et al., 2025a) across all tasks, with the largest gains on 3D benchmarks OMNI3D-BENCH (40.7% vs. 37.0% for Molmo2) and CV-BENCH (81.6% vs. 53.7% for Gemma3). Compared to larger proprietary VLMs, GenVer competes favorably: outperforming Gemini-3-Flash and GPT-5-mini on ROBOSPATIAL (81.5% vs. 76.3% for GPT-5-mini), and approaching GPT-5-mini on OMNI3D-BENCH (40.7% vs. 40.9%), while trailing Gemini-3-Flash.

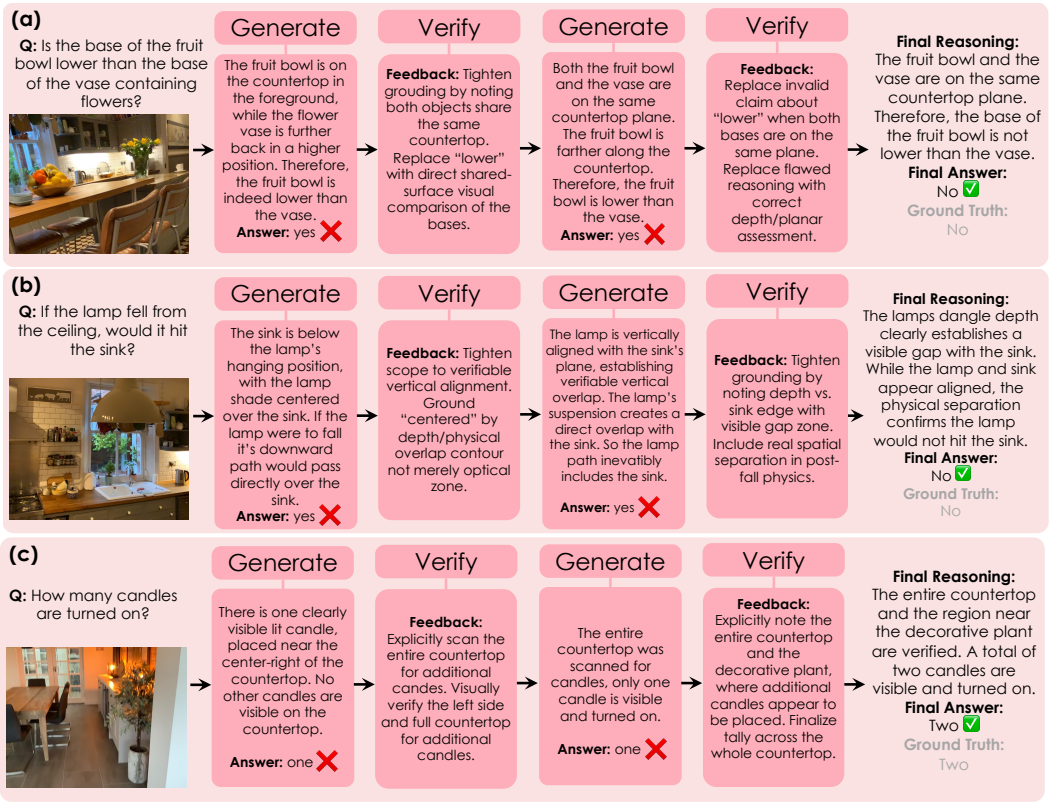

Figure 5: **Outputs for GenVer with $N = 2$ self-verification rounds.** Additional rounds of self-verification helps complex queries by refining occasionally vague first-round feedback.

	OMNI3D-BENCH	ROBO SPATIAL	REALWORLD QA	OMNI SPATIAL	CV-BENCH	BLINK	VSR
Qwen3-VL 4B							
Direct Prompting	26.0	60.9	63.1	42.3	75.2	81.1	68.6
Majority Voting ($K=5$)	35.0	79.9	61.2	44.2	**83.7**	81.8	70.3
LLM Judge ($K=5$)	35.3	78.9	61.3	43.6	83.5	80.4	72.0
GenVer	**40.7**	**81.5**	**69.7**	**47.2**	81.6	**83.8**	**75.6**

Table 4: **Comparisons to traditional test-time scaling baselines.**

Verification and reasoning improve together during training. To examine how self-verification evolves, we evaluate intermediate checkpoints on OMNI3D-BENCH and track verification reward R_{ver} (Figure 4). Both verification reward and downstream accuracy improve over training and are strongly correlated across checkpoints ($r = 0.87$), suggesting that stronger verification develops alongside improved visual reasoning. Appendix Table 5 further isolates verification gains: replacing the learned verifier with the pre-trained Qwen3-VL verifier under identical prompts consistently reduces performance, indicating that training improves verification capability.

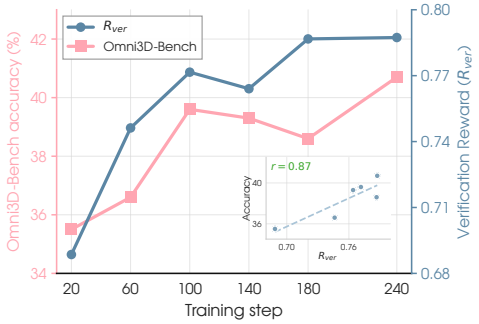

Figure 4: Verification ability and visual reasoning improve together throughout training.

4.2 EXPLORING SELF-VERIFICATION AT TEST-TIME

While our goal is to introduce self-verification as a training paradigm, we next examine its behavior at test time. We compare GenVer to traditional test-time scaling to test whether its gains reduce to increased sampling, and analyze multiple rounds of self-verification to understand iterative refinement. Together, these experiments characterize how self-verification affects visual reasoning at inference.

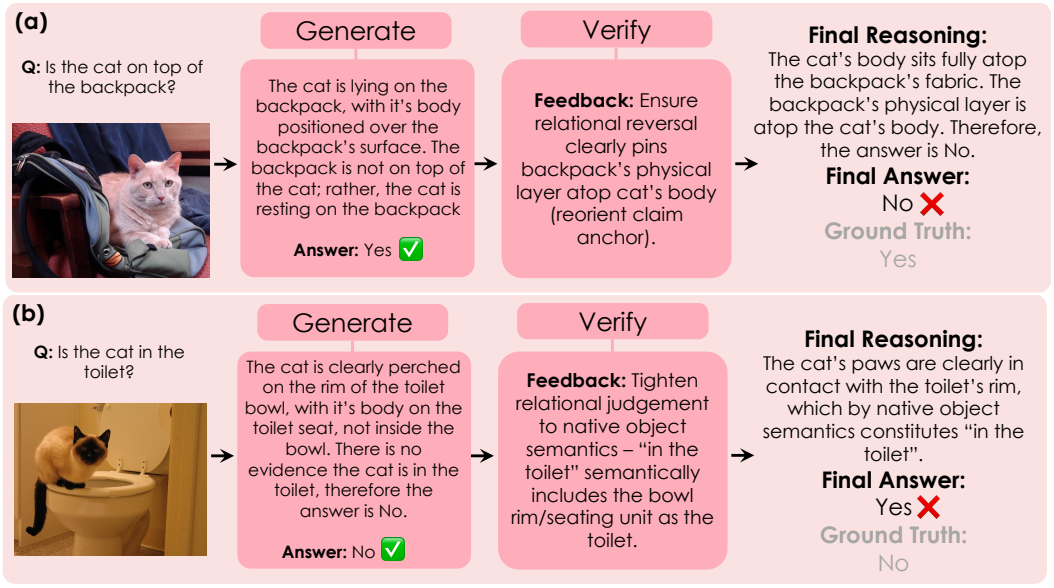

Figure 6: **Examples of over-verification.** We show examples from BLINK and VSR where self-verfication degrades performance on simple queries by adding unnecessary complexity.

Increasing self-verification at test-time. We explore increasing the number of self-verification rounds at test time to understand how iterative refinement improves reasoning beyond the $N = 1$ setting used in training. Figure 5 shows additional rounds improve predictions when first-round feedback is vague. In (a) the first round vaguely suggests the objects share a countertop, while the second makes explicit that this implies equal base-height. In (b), the first round relies on vertical alignment, whereas the second incorporates depth and physical separation to reach the correct conclusion. In (c), the first round fails to consider the region near the plant, while the second guides the model to that area and recovers the correct count. Consistent with these examples, we find increasing the number of verification rounds to $N = 2$ yields further gains on the most reasoning-intensive benchmarks, improving performance to 42.4% on OMNI3D-BENCH and 82.0% on ROBOSPATIAL.

Comparison to traditional test-time scaling. Table 4 compares GenVer to test-time scaling via majority voting (Wang et al., 2022) and LLM-as-a-judge (Zheng et al., 2023). Both use $K = 5$ samples, versus 3 model calls for GenVer. While these baselines improve over direct Qwen3-VL 4B prompting (*e.g.,* 68.6% to 72.0% on VSR), GenVer outperforms them on most benchmarks, especially OMNI3D-BENCH (40.7% vs. 35.0% for majority voting). Since OMNI3D-BENCH contains open-ended queries, naive sampling is unlikely to find correct answers by chance. Thus, GenVer's gains do not arise solely from repeated sampling, but from improved reasoning and verification.

4.3 OVER-VERIFICATION

We now discuss a limitation of GenVer, termed *over-verification*. While self-verification improves complex reasoning, it can be less effective on the simplest queries in our evaluations. On VSR and BLINK, benchmarks dominated by simple binary or perceptual judgments, refinement can add unnecessary complexity. Figure 6 shows cases where self-verification degrades an initially correct prediction. In (a), the verifier introduces a spurious relational reinterpretation that reverses the spatial relationship and leads to an error. In (b), the verifier overextends the semantics of "in" to include the rim, producing an incorrect correction. While our focus is on learning *how* to verify, these results suggest that learning *when* and *how much* to verify is an important direction for future work.

5 CONCLUSION

We introduce GenVer, a post-training paradigm that enables self-verification without requiring ground-truth answers. Our method treats generation and verification as complementary abilities and trains VLMs to improve both. Our experiments show that jointly training for generation and verification outperforms training generation alone. **We release all data, code, and post-trained models.**

AI USE STATEMENT

In this work, we used generative AI tools to aid with implementation and editing the manuscript for clarity. We did not use AI for ideation or brainstorming, or to discover relevant literature. We used generative AI to generate synthetic data for model training, as described in §3.2. The remainder of the disclosure tasks (assisting with translation, proving mathematical claims, etc.) are not applicable to this work. We have reviewed all AI-assisted writing and implementations. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We include finegrained details for all of the experiments in §4 and in the Appendix (§A.3 and §A.4). We include hyperparameters in §A.5 and all prompts used in §A.6. We will release all code, data, and trained models upon acceptance.

REFERENCES

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*, 2025.

Garrick Brazil, Abhinav Kumar, Julian Straub, Nikhila Ravi, Justin Johnson, and Georgia Gkioxari. Omni3D: A large benchmark and model for 3D object detection in the wild. In *CVPR*, 2023.

Jiefeng Chen, Jie Ren, Xinyun Chen, Chengrun Yang, Ruoxi Sun, Jinsung Yoon, and Sercan Ö Arık. Sets: Leveraging self-verification and self-correction for improved test-time scaling. *arXiv preprint arXiv:2501.19306*, 2025a.

Lili Chen, Mihir Prabhudesai, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. Self-questioning language models. *arXiv preprint arXiv:2508.03682*, 2025b.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? *arXiv preprint arXiv:2403.20330*, 2024.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. In *ICML*, 2025.

Christopher Clark, Jieyu Zhang, Zixian Ma, Jae Sung Park, Mohammadreza Salehi, Rohun Tripathi, Sangho Lee, Zhongzheng Ren, Chris Dongjoo Kim, Yinuo Yang, et al. Molmo2: Open weights and data for vision-language models with video understanding and grounding. *arXiv preprint arXiv:2601.10611*, 2026.

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.

Yuhao Dong, Zuyan Liu, Hai-Long Sun, Jingkang Yang, Winston Hu, Yongming Rao, and Ziwei Liu. Insight-v: Exploring long-chain visual reasoning with multimodal large language models. In *CVPR*, 2025.

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

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A Smith, Wei-Chiu Ma, and Ranjay Krishna. Blink: Multimodal large language models can see but not perceive. In *ECCV*, 2024.

Anna Goldie, Azalia Mirhoseini, Hao Zhou, Irene Cai, and Christopher D Manning. Synthetic data generation & multi-step rl for reasoning & tool use. *arXiv preprint arXiv:2504.04736*, 2025.

Google. Gemini 3, 2025. URL https://blog.google/products-and-platforms/products/gemini/gemini-3/.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *CVPR*, 2023.

Yushi Hu, Otilia Stretcu, Chun-Ta Lu, Krishnamurthy Viswanathan, Kenji Hata, Enming Luo, Ranjay Krishna, and Ariel Fuxman. Visual program distillation: Distilling tools and programmatic reasoning into vision-language models. In *CVPR*, 2024.

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

Mengdi Jia, Zekun Qi, Shaochen Zhang, Wenyao Zhang, Xinqiang Yu, Jiawei He, He Wang, and Li Yi. Omnispatial: Towards comprehensive spatial reasoning benchmark for vision language models. *arXiv preprint arXiv:2506.03135*, 2025.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, 2023.

Hyunseok Lee, Seunghyuk Oh, Jaehyung Kim, Jinwoo Shin, and Jihoon Tack. Revise: Learning to refine at test-time via intrinsic self-verification. In *International Conference on Machine Learning*, pp. 33616–33634. PMLR, 2025.

Xiang Lisa Li, Vaishnavi Shrivastava, Siyan Li, Tatsunori Hashimoto, and Percy Liang. Benchmarking and improving generator-validator consistency of language models. *arXiv preprint arXiv:2310.01846*, 2023.

Zejun Li, Ruipu Luo, Jiwen Zhang, Minghui Qiu, Xuanjing Huang, and Zhongyu Wei. Vocot: Unleashing visually grounded multi-step reasoning in large multi-modal models. *arXiv preprint arXiv:2405.16919*, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Fangyu Liu, Guy Edward Toh Emerson, and Nigel Collier. Visual spatial reasoning. *Transactions of the Association for Computational Linguistics*, 2023.

Runze Liu, Junqi Gao, Jian Zhao, Kaiyan Zhang, Xiu Li, Biqing Qi, Wanli Ouyang, and Bowen Zhou. Can 1b LLM surpass 405b LLM? rethinking compute-optimal test-time scaling. In *ICLRW*, 2025.

Damiano Marsili and Georgia Gkioxari. No labels, no problem: Training visual reasoners with multimodal verifiers. *arXiv preprint arXiv:2512.08889*, 2025.

Damiano Marsili, Rohun Agrawal, Yisong Yue, and Georgia Gkioxari. Visual agentic ai for spatial reasoning with a dynamic api. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19446–19455, 2025.

OpenAI. gpt-5, 2025a. URL https://openai.com/index/introducing-gpt-5/.

OpenAI. o4-mini, 2025b. URL https://openai.com/index/introducing-o3-and-o4-mini/.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. In *NeurIPS*, 2024.

Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. Vision language models are blind: Failing to translate detailed visual features into words. *arXiv preprint arXiv:2407.06581*, 2024.

Gabriel Sarch, Lawrence Jang, Michael Tarr, William W Cohen, Kenneth Marino, and Katerina Fragkiadaki. Vlm agents generate their own memories: Distilling experience into embodied programs of thought. *NeurIPS*, 2024.

Gabriel Sarch, Snigdha Saha, Naitik Khandelwal, Ayush Jain, Michael J Tarr, Aviral Kumar, and Katerina Fragkiadaki. Grounded reinforcement learning for visual reasoning. In *NeurIPS*, 2025.

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

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. Rl's razor: Why online reinforcement learning forgets less. *arXiv preprint arXiv:2509.04259*, 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*, 2025.

Chan Hee Song, Valts Blukis, Jonathan Tremblay, Stephen Tyree, Yu Su, and Stan Birchfield. RoboSpatial: Teaching spatial understanding to 2D and 3D vision-language models for robotics. In *CVPR*, 2025.

Yuda Song, Hanlin Zhang, Carson Eisenach, Sham M Kakade, Dean Foster, and Udaya Ghai. Mind the gap: Examining the self-improvement capabilities of large language models. In *ICLR*, 2024.

Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *ICCV*, 2023.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025a.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025b.

Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. Winoground: Probing vision and language models for visio-linguistic compositionality. In *CVPR*, 2022.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *NeurIPS*, 2024a.

Shengbang Tong, Ellis L Brown II, Penghao Wu, Sanghyun Woo, ADITHYA JAIRAM IYER, Sai Charitha Akula, Shusheng Yang, Jihan Yang, Manoj Middepogu, Ziteng Wang, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. In *NeurIPS*, 2024b.

Siddarth Venkatraman, Vineet Jain, Sarthak Mittal, Vedant Shah, Johan Obando-Ceron, Yoshua Bengio, Brian R Bartoldson, Bhavya Kailkhura, Guillaume Lajoie, Glen Berseth, et al. Recursive self-aggregation unlocks deep thinking in large language models. *arXiv preprint arXiv:2509.26626*, 2025.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *ICLR*, 2022.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2550–2575, 2023.

xAI. Real world qa, 2024. URL https://x.ai/news/grok-1.5v.

Guowei Xu, Peng Jin, Ziang Wu, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. Llava-cot: Let vision language models reason step-by-step. In *ICCV*, 2025.

Jihan Yang, Shusheng Yang, Anjali W. Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in Space: How Multimodal Large Language Models See, Remember and Recall Spaces. *arXiv preprint arXiv:2412.14171*, 2024.

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.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. Lmms-eval: Reality check on the evaluation of large multimodal models, 2024. URL https://arxiv.org/abs/2407.12772.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Seyed Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. In *International Conference on Learning Representations*, volume 2025, pp. 12476–12505, 2025a.

Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Wenyue Hua, Haolun Wu, Zhihan Guo, Yufei Wang, Niklas Muennighoff, et al. A survey on test-time scaling in large language models: What, how, where, and how well? *arXiv preprint arXiv:2503.24235*, 2025b.

Eric Zhao, Pranjal Awasthi, and Sreenivas Gollapudi. Sample, scrutinize and scale: Effective inference-time search by scaling verification. In *ICML*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. In *NeurIPS*, 2023.

Adam Zweiger, Jyothish Pari, Han Guo, Ekin Akyürek, Yoon Kim, and Pulkit Agrawal. Self-adapting language models. *arXiv preprint arXiv:2506.10943*, 2025.

A APPENDIX

CONTENTS

A Appendix **1**

A.1 Additional Experiments 1

A.2 Proxy Reward Model Analysis 2

A.3 Additional Evaluation Data Details 3

A.4 Additional Training Details 4

A.4.1 Experimental Variability 4

A.4.2 Training Data Generation. 4

A.4.3 Rewards 4

A.5 Hyperparameters 5

A.6 Prompts 6

A.1 ADDITIONAL EXPERIMENTS

Improvements to verification. We compare the learned verifier to the pre-trained Qwen3-VL base model in Table 5, isolating gains in verification from training. For GenVer (+ base), we replace self-verifier feedback with base-model feedback under identical prompts. GenVer consistently outperforms this variant across benchmarks, indicating the learned verifier better identifies errors and guides refinement. These results suggest self-verification training improves both verification quality and the ability to incorporate feedback.

	OMNI3D-BENCH	ROBO SPATIAL	REALWORLD QA	OMNI SPATIAL
GenVer (+ base)	36.3	79.1	69.3	45.0
GenVer	**40.7**	**81.5**	**69.7**	**47.2**

Table 5: **Accuracy (%) with base-model verification.** We replace self-verifier feedback with the pre-trained model (+ base) to isolate improvements in verification.

GenVer with Molmo2. The GenVer training framework is largely VLM-agnostic. To validate our approach across VLMs, we additionally train Molmo2-4B (Clark et al., 2026) with our GenVer recipe and report results below.

	OMNI3D-BENCH	ROBOSPATIAL	REALWORLDQA	OMNISPATIAL	CV-BENCH	BLINK	VSR
Base Model	37.0	66.3	66.4	44.6	75.2	74.1	63.4
GenVer	**38.4**	**79.8**	**69.7**	**47.0**	**79.6**	**79.7**	**77.4**

Table 6: Accuracy (%) on visual reasoning benchmarks for GenVer with Molmo2.

We find that the GenVer recipe also yields substantial gains for Molmo2 across visual reasoning benchmarks, including a $+13.5\%$ improvement on ROBOSPATIAL. This suggests that the benefits of jointly optimizing generation and self-verification are not specific to Qwen3-VL. Qualitatively, we observe similar self-verification behavior in Molmo2: the feedback grounds ambiguous claims, prompts re-counting when the initial response hallucinates objects, and encourages the model to respect conditions specified in the query. We also observe a similar over-verification pattern on simpler queries (§ 4.3).

Increasing Self-Verification Rounds. We study the effect of increasing the number of feedback iterations during training. Specifically, we train a GenVer variant with `N=random(1,3)` self-verification rounds. For each prompt group, we sample a number of rounds in $[1, 3]$ and use that value for all rollouts in the group. We report results in the table below.

	OMNI3D-BENCH	ROBOSPATIAL	REALWORLDQA	OMNISPATIAL	CV-BENCH	BLINK	VSR
Base Model	26.0	60.9	63.1	42.3	75.2	81.1	68.6
GenVer ($N = \text{random}(1, 3)$)	39.0	81.4	68.8	**49.0**	**83.5**	**85.3**	74.4
GenVer ($N = 1$)	**40.7**	**81.5**	**69.7**	47.2	81.6	83.8	**75.6**

Table 7: Effect of increasing the number of self-verification rounds during training.

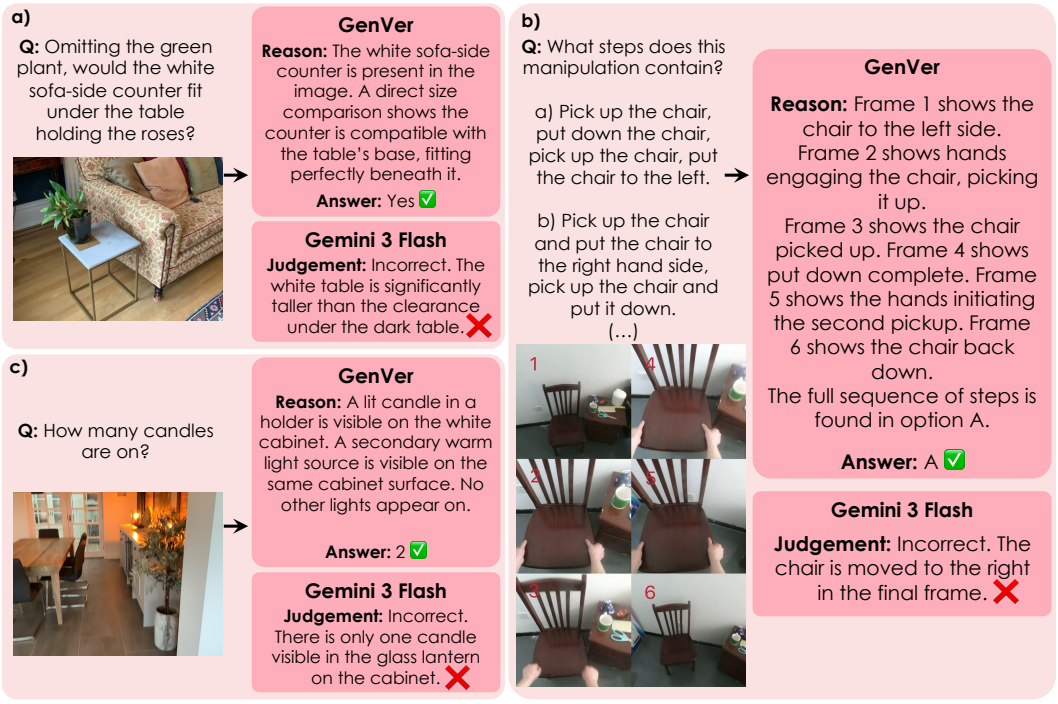

Figure 7: **Proxy reward failures.** Examples where GenVer predicts the correct answer, but the proxy reward model V incorrectly judges the response as incorrect.

Overall, this variant performs similarly to the $N = 1$. It improves on BLINK (85.3% vs. 83.1%) and OMNISPATIAL (49.0% vs. 47.2%), but underperforms on OMNI3D-BENCH (39.0% vs. 40.7%) and REALWORLDQA (68.8% vs. 69.7%), suggesting that simply increasing the number of training-time feedback iterations does not uniformly improve performance.

A.2 PROXY REWARD MODEL ANALYSIS

In this section, we analyze the reliability of the high-capacity VLM $\mathcal{V}$ used as our proxy reward model during training. This analysis is important because, if $\mathcal{V}$ were an exact verifier, visual reasoning could be trained by directly optimizing its answer scores alone. Instead, we find that $\mathcal{V}$ provides a useful but imperfect training signal: it does not consistently recognize correct answers, nor do its scores faithfully recover ground-truth benchmark accuracy. These limitations motivate our use of $\mathcal{V}$ as a proxy reward over generation–verification pairs, rather than as an exact reward for final-answer correctness. We present evidence for these failure modes below.

Quantitative results. We evaluate $\mathcal{V}$ in three ways. First, we test whether $\mathcal{V}$ can reliably recognize correct answers by asking it to score ground-truth responses from our most reasoning-intensive benchmarks: OMNI3D-BENCH and OMNISPATIAL. An exact verifier should assign full credit to these answers, but $\mathcal{V}$ scores them at only 84.2% and 88.0%, respectively, showing that it can reject correct solutions. Such false negatives would penalize valid reasoning traces, making $\mathcal{V}$ unreliable as the sole training objective. Second, we ask $\mathcal{V}$ to score pre-trained Qwen3-VL responses and compare these scores to ground-truth accuracy. $\mathcal{V}$ assigns scores of 40.9% on OMNI3D-BENCH and 70.7% on OMNISPATIAL, well above the true respective accuracies of 26.0% and 42.3%. Finally, we repeat the same analysis on the final answers predicted by GenVer: $\mathcal{V}$ assigns scores of 69.0%

on OMNI3D-BENCH and 77.9% on OMNISPATIAL, again above the true accuracies of 40.7% and 47.2%. This overestimation would assign high reward to many incorrect predictions, further making direct answer-score optimization unreliable.

Qualitative results. Figure 7 shows examples where GenVer produces the correct answer, but the proxy reward model $\mathcal{V}$ incorrectly judges the response as incorrect. In (a), GenVer correctly reasons that the sofa-side counter can fit beneath the table by comparing the counter to the table base, while $\mathcal{V}$ rejects the answer by incorrectly estimating the counter height. In (b), GenVer correctly identifies the manipulation sequence by tracking the chair across frames, but $\mathcal{V}$ misjudges the response by overemphasizing the chair position in the final frame rather than the full sequence of actions. In (c), GenVer correctly counts two lit candles, while $\mathcal{V}$ overlooks one of the light sources and judges the answer incorrect. These examples show that GenVer can recover correct grounded solutions that are not recognized by $\mathcal{V}$, suggesting that its gains are not simply due to imitating the proxy reward model, but arise from jointly learning to generate and verify visual reasoning responses.

A.3 ADDITIONAL EVALUATION DATA DETAILS

We describe additional details about the VQA benchmarks used for visual reasoning evaluation below. We report exact-match accuracy for all yes/no, multiple choice, and integer response queries, and MRA for floating point response queries following Marsili et al. (2025) and Yang et al. (2024).

Omni3D-Bench is a VQA benchmark designed to evaluate 3D grounding and spatial reasoning capabilities (Marsili et al., 2025). It contains 500 challenging image–question–answer triples drawn from diverse real-world scenes in Omni3D (Brazil et al., 2023). Representative questions include *"If the black table is 1.5m wide, how tall is the TV in meters?"* and *"If the rightmost sofa were placed in front of the fireplace, would the fireplace still be visible?"*.

RealworldQA consists of 765 VQA queries collected from a wide range of real-world scenes (xAI, 2024). The dataset emphasizes general spatial understanding, with a particular focus on outdoor environments. Example questions include *"Is the closest red car to us closer than the closest semi truck?"* and *"Based on the shadow, where is the sun relative to us?"*. We report results on the full `test` split.

CV-Bench is a vision-centric benchmark for VLMs composed of 2,638 manually verified queries (Tong et al., 2024b). Example queries include *"How many cushions are in the image?"* and *"Which object is closer to the camera taking this photo, the table (highlighted by a red box) or the bookcase (highlighted by a blue box)?"*. We evaluate on both the `2D` and `3D` splits.

OmniSpatial is a comprehensive spatial reasoning benchmark grounded in cognitive psychology (Jia et al., 2025). The benchmark covers four major categories: dynamic reasoning, complex spatial logic, spatial interaction, and perspective-taking, with 50 fine-grained subcategories. Examples queries include *"Are the sculptures on the left or right side of mummies?"* and *" "In which direction will the hand move next?"*. We report results on the full `test` split.

RoboSpatial is a large-scale VQA benchmark targeting spatial reasoning in indoor and tabletop robotic settings (Song et al., 2025). Questions are procedurally generated from annotated 3D scans. Example queries include *"Can the speaker fit in front of the monitor?"* and *"Is the box in front of the chair?"*. Our evaluation focuses on the `compatibility` and `configuration` task subsets.

BLINK is a benchmark of 14 visual reasoning challenges that can be solved in a "blink" by humans but remain difficult for VLMs (Fu et al., 2024). We evaluate on single-image queries from the `val` split of the relevant `spatial relationship` task.

VSR is a spatial reasoning dataset with images from COCO (Lin et al., 2014) paired with spatial relationships between two objects in the image (Liu et al., 2023). VLMs are given statements which they must evaluate as `True` or `False`. Examples include *"The potted plant is at the right side of the bench"* and *"The cow is ahead of the person"*. We evaluate on the `test` split.

A.4 Additional Training Details

A.4.1 Experimental Variability

Our results for Answer Reward and GenVer in Table 1 report mean values over three seeds. We provide the standard deviations (in percentage points) of these results in Table 8. We find both methods exhibit similar variability across benchmarks.

	Omni3D-Bench	RoboSpatial	RealWorldQA	OmniSpatial	CV-Bench	BLINK	VSR
Answer Reward	1.1	0.5	1.6	0.2	0.8	1.4	0.5
GenVer ($N = 1$)	0.9	1.6	1.5	0.5	0.3	0.7	0.3

Table 8: Standard deviation of accuracy (percentage points) across three seeds.

A.4.2 Training Data Generation.

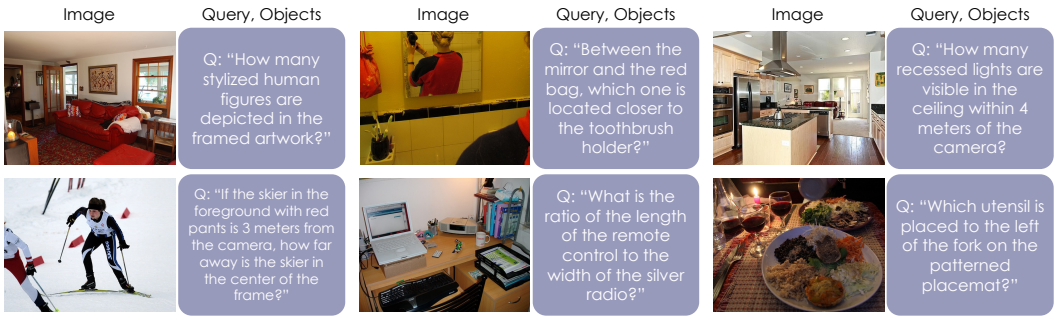

Figure 8: Examples of generated training data.

Since our method does not require ground truth annotations, we can generate synthetic data following Marsili & Gkioxari (2025). Our training relies on pairs $\{(I_j, q_j)\}_{j=1}^{D}$ consisting of an image I and query q. Examples of our generated data are in Figure 8. We provide details on our data generation below.

Images. To ensure our training data captures a diverse set of indoor and outdoor scenes, we source images from SA-1B (Kirillov et al., 2023) and the COCO (Lin et al., 2014) training set. We sample 2000 images randomly from each dataset.

Queries. We leverage Gemini-3-Flash (Google, 2025) to generate visual reasoning queries. We seed the query-generation process with a specific "skill" to increase query diversity and ensure our training dataset is broadly applicable across visual reasoning tasks. The full set of skills is listed in Figure 13. For each image, we randomly select a skill from the list and prompt Gemini to generate three queries for that image that test the given skill. We find that 3D-aware reasoning is particularly challenging for VLMs, so we sample skills 5-10 with double probability.

A.4.3 Rewards

We provide additional details for our rewards used during training. As our method aims to improve generation and verification, we design a unique reward model for each objective. We additionally impose a format constraint: samples that do not adhere to the required output format receive zero reward for both objectives. Specifically, for generation the model is instructed to reason inside `<reason>` tags and answer inside `<answer>` tags, whereas for verification the model is provided an output format to adhere to (Figure 10). Failure to adhere to either of these requirements results in zero reward for both objectives.

Answer Quality (R_{ans}): We use the external VLM verifier $\mathcal{V}$ to grade the solution on a 0-10 scale, normalized to $[0, 1]$: $R_{\text{ans}}(\tau) = 0.1 \cdot \mathcal{V}(I, q, a_N)$. We use Gemini-3-Flash (Google, 2025) as our verifier and include the prompt to used in Figure 11. Gemini is given a rubric to evaluate the solution, to allow partial credit for floating point solutions that are mostly correct. We provide Gemini the

<reason> outputs for context, but instruct it to focus on the final answer and not reward persuasive but incorrect reasoning. We find prompting Gemini to score on a $[0, 10]$ scale and normalizing provides more consistent judgment behavior than prompting in the $[0, 1]$ scale directly.

Answer Verification Quality (R_{ver}): For each round t of self-verification, let (r_t, a_t) be the current solution, f_t the self-verification feedback, and (r_{t+1}, a_{t+1}) be the updated solution. We first prompt $\mathcal{V}$ to score the feedback: $s^{(t)}_{\text{edit}} = \mathcal{V}(f_t|I, q, r_t, a_t)$ where $s^{(t)}_{\text{edit}} \in [0, 1]$. The prompt used to score feedback is shown in Figure 12. We provide Gemini the query, the current solution (r_t, a_t) and the self-edit feedback f_t.

If $\mathcal{V}$ is used for the verifier feedback at round t, the self-verifier reward for this round is purely score based: $R^{(t)}_{\text{ver}} = s^{(t)}_{\text{edit}}$. Otherwise, if self-verifier edits are used, we additionally use $\mathcal{V}$ to score both solutions: $s_t = \mathcal{V}(a_t|I, q),\ s_{t+1} = \mathcal{V}(a_{t+1}|I, q)$ in the manner described above (R_{ans}), and define usefulness $u^{(t)}$ to assess if feedback improves solutions or does not degrade solutions that are already correct:

$$u^{(t)} = \begin{cases} 1 & \text{if } s_{t+1} > s_t \\ 1 & \text{if } s_{t+1} = s_t \text{ and } s_t > 0.7 \\ 0.3 & \text{if } s_{t+1} = s_t \text{ and } s_t \leq 0.7 \\ 0 & \text{if } s_{t+1} < s_t. \end{cases}$$

We reward the self-verifier highly when it does not edit solutions that already score highly ($s_t > 0.7$) in an attempt to counteract the "over-verification" behavior we describe in § 4.3. Moreover, for the same reason, even in cases where solutions do not score highly ($s_t < 0.7$) we penalize feedback that degrades solutions more than feedback that maintains them.

The round reward is defined as $R^{(t)}_{\text{ver}} = 0.7 s^{(t)}_{\text{edit}} + 0.3 u^{(t)}$. We take the mean over all rounds to obtain the final reward: $R_{\text{ver}}(\tau) = \frac{1}{N} \sum_{t=0}^{N-1} R^{(t)}_{\text{ver}}$.

A.5 Hyperparameters

We include all hyperparameters used to train GenVer in the table below. For inference, we use the recommended sampling parameters for Qwen3-VL (Bai et al., 2025).

Hyperparameter	Value
λ_{ans}	2.0
λ_{ver}	1.0
Batch size	32
Group size	8
Learning rate	5e-6
Epochs	2
Self-verification rounds (N)	1
Max image side	480
KL coefficient	0
ϵ_{high}	0.24
ϵ_{low}	0.2
Rollout engine	vLLM
FSDP dtype	bfloat16
Rollout dtype	bfloat16
Temperature	0.7
Top-p	0.8
Top-K	20
Repetition penalty	1.0
Presence penalty	1.5
Frequency penalty	0.0

Table 9: Hyperparameters for training GenVer.

A.6 PROMPTS

```
Please answer the query by first reasoning inside <reason> tags with numbered steps (1., 2., 3., ... one step
per line), then putting ONLY your final answer inside <answer>. Ensure that the answer is either yes/no, one
word, or one number. Unless explicitly specified otherwise, assume all metric quantities are 3D and depth-
aware. Do not round answers, express all ratios as unrounded decimals. Nothing else.

{query}
```

Figure 9: **System prompt.**

```
You are a logic self-verifier for visual reasoning answers.

You will be given:
- the original query,
- a candidate solution containing <reason> and <answer>.

The <reason> should be treated as step-indexed reasoning with numbered lines like:
1. ...
2. ...
3. ...

Task:
Provide up to 2 concrete reasoning refinements that would improve the candidate reasoning.
Focus only on the reasoning and answer.

Output format:
Return ONLY plain text refinements with one action per line (no JSON, no markdown, no preamble).
Each line MUST be exactly:
EDIT_STEP <step_index>: <specific refinement for that numbered reasoning step>

You may output 0, 1, or 2 refinement lines total.
If no refinement is needed, output nothing.

Requirements:
- refinements must be actionable and specific.
- only refine the reasoning; do not directly refine the final answer.
- do not suggest a direct answer replacement; instead suggest how the reasoning should change so the answer
can be recomputed from the revised reasoning.
- every refinement must reference an existing numbered reasoning step.
- if something is missing, attach the refinement to the closest existing step that should be expanded or
corrected.
- prefer small targeted refinements.
- do not write a replacement <reason> block or a final <answer> directly.
- do not state the corrected reasoning or final answer in finished form.
- softly suggest what should change, for example by pointing out what to reconsider, recompute, tighten, or
correct.
- do not say the answer is impossible to determine. Answers like 'cannot determine', 'unknown', 'can't answer
', or 'impossible to answer' are always incorrect. When these appear, instruct the model to produce a concrete
 estimate.
- visual estimation is appropriate when exact measurements are unavailable; relative estimation is valid.
- unless explicitly specified otherwise by the query, assume all metric quantities (height/width/length/depth/
distance/volume) are 3D and depth-aware.
- fractional counts of objects are valid when the query asks how many of object X are needed.
- never output tags or JSON.

Query:
{query}

Candidate answer:
{answer}
```

Figure 10: **Self-verification prompt.**

```
You are a strict visual-answer judge.

Judge ONLY the FINAL <answer> in the candidate response, using the image(s) and question as ground truth.
Use reasoning only for interpretation; do not reward persuasive but wrong reasoning.
If final answer is 'cannot determine', 'unknown', 'can't answer', or 'impossible', score it as incorrect.
Visual estimation is acceptable when exact measurement is unavailable.
Unless explicitly specified otherwise by the question, assume all metric quantities (height/width/length/depth
/distance/volume) are 3D and depth-aware.

Question:
{query}

Candidate response:
{candidate_response}

Scoring (integer 0-10):
- For multiple-choice, yes/no, true/false, and counting queries (including prompts asking for an integer):
  use STRICT binary scoring only:
  - 10 if exactly correct
  - 0 if incorrect
  - no partial credit
- For float/continuous numeric estimation queries (e.g., ratios, approximate measurements):
  partial credit is allowed:
  - 10: clearly correct / equivalent
  - 7-9: mostly correct, small numeric deviation
  - 4-6: partially correct, materially off
  - 1-3: weak but related estimate
  - 0: incorrect

Be concise and strict.
'reason' must be one short sentence (<= 20 words).
Output ONLY JSON (no prose/markdown/code fences).

Return JSON keys in this exact order: 'verdict', 'reason', 'score'.
```

Figure 11: **Answer reward judge prompt.**

```
You are a strict logic-edit judge/teacher for visual reasoning.

Inputs:
- query
- current <reason>/<answer>
- proposed self-edits

Tasks:
1. Score current answer quality: 'current_answer_score' in [0,10].
2. Score usefulness of proposed self-edits: 'self_edit_score' in [0,10].
3. Write 'self_edit_reason' as one short sentence (<= 20 words).
4. Provide 'teacher_edits': up to 2 lines, each `EDIT_STEP k: ...`.

Rules:
- Focus only on reasoning/answer improvement.
- Edit existing numbered steps only.
- Do not rewrite full reason or give final answer directly.
- If no edit is needed, return empty list for teacher_edits.
- 'cannot determine/unknown/can't answer/impossible' is incorrect; prefer grounded visual estimation.
- Unless explicitly specified otherwise by the question, assume all metric quantities (height/width/length/
depth/distance/volume) are 3D and depth-aware.
- Scoring policy by question type:
  - For multiple-choice, yes/no, true/false, and counting/integer-answer queries:
    - 'current_answer_score' must be strict binary: 10 if exactly correct, else 0.
    - 'self_edit_score' must be strict binary: 10 only if the proposed edits are clearly useful for reaching/
    preserving the exact correct discrete answer, else 0.
    - no partial credit.
  - For float/continuous numeric estimation queries:
    - partial credit is allowed for both 'current_answer_score' and 'self_edit_score' on a 0-10 scale.
- Be concise and strict.

Question:
{query}

Current answer:
{current_answer}

Proposed self-verifier edits:
{proposed_self_edits}

Output ONLY JSON with keys in this order:
- 'self_edit_reason'
- 'teacher_edits'
- 'current_answer_score'
- 'self_edit_score'
```

Figure 12: **Self-verifier edit judge prompt.**

```
0: Counting queries. Ask about counts of visible objects (e.g., 'How many X are there?').
1: Simple spatial relationships: Ask about relative position (left/right/above/below/in front of/behind)
between objects.
2: True/False spatial relationships. Form statements like 'The dog is left of the cat. Respond True or False.
3: Depth from camera: Compare which object is closer to the camera.
4: Count with depth. Ask for counts within a camera-distance threshold.
5: Distance from other object: Ask which object is closer to a reference object.
6: Space queries: Ask about how one object shape relates to part of another object
7: Ratios of heights/widths/lengths: Ask for ratios between size attributes of two objects.
8: Hypothetical scaling/grounding: Use a hypothetical size anchor and infer another dimension.
9: Hypothetical depth scaling: Use a hypothetical camera distance for one object and infer camera distance for
 another.
```

Figure 13: **Skills used for data generation.**

```
You are part of a data engine that generates training data for visual reasoning training. You will receive an
image and one skill index (0-9). Generate exactly 3 image-grounded queries that test only that skill. Make the
 queries moderately difficult and non-trivial for a capable model. Use varied wording and synonyms so queries
are not repetitive, while staying faithful to the selected skill definition. Return valid JSON only, matching
the required schema.

Skill: {}
Generate exactly 3 queries for this image that test only this skill.
Make the queries moderately difficult (not trivial), but still answerable from the image.
Randomize phrasing and synonyms across queries while preserving meaning.
Keep output strictly in the required JSON schema.

{
  'queries': [
    {
      'query': 'string'
    },
    {
      'query': 'string'
    },
    {
      'query': 'string'
    }
  ]
}
```

Figure 14: **Data generation prompt.**