# KEEP THE IMAGE IN MIND: A VISUAL BOARD FOR MULTIMODAL AGENTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Visual reasoning agents must retain useful observations across steps without treating earlier interpretations as substitutes for direct visual evidence. Early visual misinterpretations can persist as faulty premises, leading to incorrect answers despite otherwise coherent reasoning. We introduce a visual reasoning harness centered on the Visual Board, a revisable textual workspace that maintains task-relevant observations and intermediate reasoning across steps. With continued access to the original image, the VLM uses the board to guide targeted visual inspection and revise earlier interpretations as new evidence becomes available. Deterministic history compression bounds textual context, while a comparably sized LLM critic reviews the VLM's interaction trace between rounds and provides feedback on reasoning validity. We evaluate the harness on Euclid 30K, SolidGeo, and the Chart QA subset, covering geometric, spatial, and chart reasoning. Our harness improves accuracy over the corresponding model-only baselines across all evaluated configurations, with gains exceeding 18 percentage points on the Chart QA subset. These results support the effectiveness of our Visual Board-centered harness for improving visual reasoning accuracy. Code is available at `https://anonymous.4open.science/r/KeepInMind-1E60/`.

## 1 INTRODUCTION

Answering questions about geometric diagrams (Lian et al., 2026), spatial configurations (Wang et al., 2026), and charts (Masry et al., 2022; 2025) requires both visual observation and reasoning over what is observed (Su et al., 2025). A model may need to read a small label, identify a relation, and combine several observations to derive an answer. Even a logically coherent derivation can yield an incorrect answer when it relies on a mistaken visual premise (Zhang et al., 2026b). Reliable visual reasoning therefore requires not only drawing conclusions from observations, but also keeping those observations open to re-examination as reasoning proceeds.

Across interaction steps, visual observations and derived conclusions are carried forward as generated text (You et al., 2023; Zhou et al., 2025). This text provides context for further reasoning, but does not independently establish that the underlying visual interpretation is correct. Studies of multimodal dialogue (Zhong et al., 2024; Jiang et al., 2026) show that misleading textual histories can cause models to propagate incorrect premises despite access to the image. Image-swap experiments (Shi et al., 2026) further show that self-reflective statements during continuous generation do not reliably trigger visual re-examination, whereas models are capable of re-establishing visual grounding when explicitly instructed in a new dialogue turn. These findings motivate a distinction between revising a reasoning trace and re-examining its visual premises.

A natural response is to make visual inspection an explicit part of the reasoning process. Multi-step frameworks (Xu et al., 2025) and visual tools (Zhang et al., 2026a) allow agents to acquire missing observations and focus on relevant image details (Zhou et al., 2025; Hu et al., 2024; Wu et al., 2026a; Fu et al., 2025). However, renewed visual access alone does not ensure that earlier interpretations are reconsidered. When a visual expert makes an error, downstream agents may continue to rely on its output rather than question it, even when a referee points to a possible mistake (Zhou et al., 2025). Moreover, additional inspection does not necessarily yield a reliable basis for revision, as attempts to gather further information may introduce hallucinations rather than improve the answer (Kumar et al., 2024). These observations motivate a distinction between acquiring additional visual infor-



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.

mation and revising reasoning premises. For a multi-step visual reasoning harness, the challenge is not only to enable further inspection, but also to make claims requiring verification explicit and use new evidence to revise those premises and the reasoning that depends on them.

In this paper, we present a Visual Board-centered harness for multi-step visual question answering. The harness combines a VLM solver equipped with visual inspection tools and a text-only LLM critic. Throughout the interaction, the solver maintains a revisable Visual Board as an explicit reference for the visual premises currently used in reasoning, while the original image remains the authoritative source of visual evidence. As illustrated in Fig. 1, a visual claim in the interaction history may remain unchallenged even when it conflicts with a new observation. By contrast, the Visual Board makes these premises explicit and revisable, helping the agent identify such conflicts and recheck the affected premises against the original image.

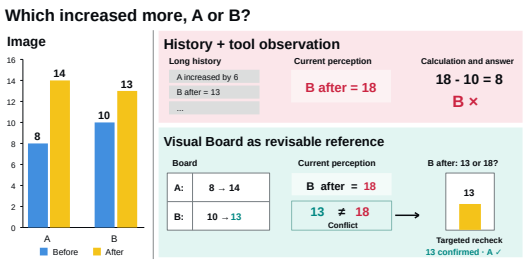



Figure 1: The Visual Board as a revisable reference. **Top**: a long interaction history may fail to correct a mistaken observation. **Bottom**: the board makes conflicting claims explicit, prompting verification against the original image.

Our harness centers on a compact, revisable Visual Board that connects visual inspection, reasoning, and review. The board makes task-relevant visual premises, uncertain details, and intermediate reasoning explicit without requiring an exhaustive image description. Deterministic history compression limits accumulated textual context while preserving the current board and access to the original image. Query-driven visual tools support targeted inspection of answer-critical details that remain unclear, allowing the solver to gather additional evidence and reconsider earlier interpretations. A text-only LLM critic reviews the reasoning against the recorded premises and provides feedback to guide the solver's subsequent reasoning and board revision. Throughout this process, the original image remains the authoritative source of visual evidence, while board entries and tool observations remain provisional rather than being assumed correct.

We evaluate the harness on Euclid 30K, SolidGeo, and Chart QA, covering geometric, spatial, and chart reasoning. Through controlled ablations, we examine the effects of visual board description requirements, history compression, and query-driven tool use. We further investigate the role of supervised fine-tuning (SFT) within the complete harness. Through this harness, the agent uses the Visual Board to guide targeted visual re-examination and incorporate revised premises into subsequent reasoning, supporting more reliable answers grounded in the original image.

Our main contributions are as follows:

- We introduce a Visual Board-centered harness for multi-step visual question answering. The harness integrates a VLM solver, visual inspection tools, and a text-only LLM critic through an explicit, revisable record of the visual premises used in reasoning.
- We design selective board construction, board-preserving history compression, and query-driven visual inspection to support premise revision under compact textual context. These mechanisms preserve access to the original image and allow uncertain details and conflicting claims to guide targeted re-examination.
- We evaluate the harness on Euclid 30K, SolidGeo, and Chart QA, with controlled ablations assessing the effects of visual description requirements, history compression, and query-driven inspection. Additional SFT experiments show that solver training improves both model-only and harness-based inference.

## 2 Related Work

**Multi-Agent Visual Reasoning.** A line of work separates visual perception from language-based reasoning through iterative model collaboration. IdealGPT (You et al., 2023) decomposes questions into visual sub-questions and combines their answers through iterative reasoning. ProReason (Zhou et al., 2025) decouples visual perception from textual reasoning, coordinating multiple agents to

proactively gather task-relevant information and assess its sufficiency before answering. Seeing-Eye (Zhang et al., 2025) uses a tool-equipped VLM to construct question-conditioned structured representations for a text-only LLM, while BeMyEyes (Huang et al., 2025) trains a perceiver to collaborate with a reasoner through multi-turn dialogue. DRP (Jia et al., 2026) addresses visual context degradation by allowing a text-only reasoner to request visual observations on demand. Our harness combines an image-conditioned VLM solver with an LLM critic, using a revisable Visual Board to support continued reasoning and feedback across interaction rounds.

**Context Management for Visual Reasoning.** Several approaches explicitly organize intermediate visual information to support subsequent reasoning. MMCTAgent (Kumar et al., 2024) introduces a vision-based critic to evaluate reasoning and visual evidence. MSSR (Guo et al., 2026) constructs a compact, sufficient set of 3D perception results for spatial question answering. See It, Say It, Sorted (Zhang et al., 2026b) maintains a growing textual visual-evidence pool to guide token-level decoding and requests additional observations when existing evidence is insufficient. Complementing these approaches, CoSee (Zhou, 2026) examines how shared notes can reinforce visual errors in resource-constrained agents and evaluates verification before state commitment on ChartQAPro. Complementary to evidence selection and verification, our Visual Board maintains a revisable reference that explicitly records uncertainties, guiding subsequent queries for missing or uncertain visual details.

**Tool-Augmented Visual Reasoning.** Tool-augmented approaches extend reasoning with explicit visual operations and external feedback. Visual Sketchpad (Hu et al., 2024) supports reasoning through drawing, while VisuoThink (Wang et al., 2025) interleaves visual and textual reasoning with look-ahead tree search. Draft and Refine (Jeong et al., 2026) selects visual experts using a question-conditioned utilization metric, and ForeSight (Wu et al., 2026b) combines low-level visual cues with mask-based feedback to support answer revision. Our harness complements these approaches by using the Visual Board to track visual uncertainties across reasoning steps and guide targeted re-inspection. Both crop-based inspection and whole-image queries require a specific visual question.

**Post-Training for Visual Reasoning.** Recent work trains VLMs to coordinate reasoning and visual actions (Shi et al., 2025). AdaReasoner (Song et al., 2026) introduces Tool-GRPO, extending GRPO (Shao et al., 2024) to optimize multi-turn tool-calling trajectories, while VTool-R1 (Wu et al., 2026a) trains VLMs to interleave textual reasoning with intermediate images produced by Python-based visual editing. VISTA-Gym (Lu et al., 2026) provides a standardized interactive environment for training tool-integrated visual reasoning. Learning also improves perception-reasoning interfaces: RAPID (Gou et al., 2026) optimizes visual descriptions using an external reasoner's answer correctness, and VALOR (Marsili & Gkioxari, 2026) combines LLM-based verification for reasoning reinforcement learning with VLM-based hard-negative mining for visual grounding. Through region-to-image distillation, ZwZ (Wei et al., 2026) improves fine-grained perception without test-time tool use. We further investigate whether supervised fine-tuning enables a 4B VLM solver to use the Visual Board more effectively in multi-step visual reasoning.

## 3 METHOD

**Overall pipeline.** As shown in Fig. 2, our harness follows a multi-round solver–critic workflow. At round $I$, the VLM Solver receives the original image $\mathcal{I}$, the question $Q$, and, for $I > 1$, feedback from the preceding critic review. The Solver uses up to 3 steps to inspect the image and generate an answer while maintaining a revisable **Visual Board** that records task-relevant visual facts, uncertain facts, and intermediate reasoning. A text-only LLM Critic then reviews the solver's output in a single step and provides feedback for the next round.

Within each round, the solver reasons over the image and its current working context, using Visual Board to record intermediate progress. When additional visual evidence is needed, it invokes visual tools (Sec. 3.3), with each invocation specifying a concrete visual question. To keep the solver's working context compact, the harness limits the length of individual tool records and applies rule-based compression to accumulated textual history (Sec. 3.2). This process preserves the latest Visual Board and recent tool-call–result pairs while maintaining access to the original image.

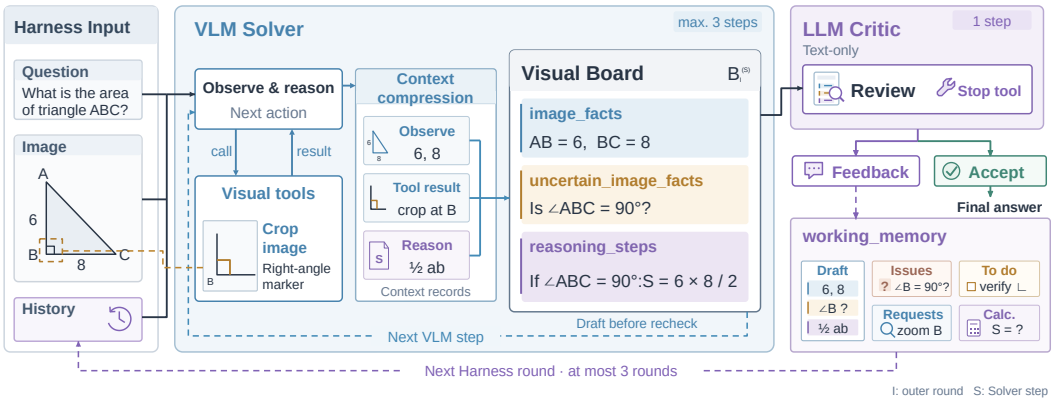



Figure 2: **Overview of the Visual Board-centered harness.** Given an image $\mathcal{I}$ and a question $Q$, the VLM Solver maintains a revisable Visual Board $B_I^S$ and uses query-driven inspection tools to re-examine uncertain visual premises. A text-only LLM Critic reviews the solver's board and reasoning process and provides structured working memory to guide the next round.

To structure communication between the solver and the critic, we equip the solver with a terminal tool for end-of-round submissions. The solver submits a final answer if it is confident in its answer and considers its reasoning sound, or a progress summary if further reasoning is needed. The critic reviews both types of submission, checking whether the solver's reasoning is logically consistent and whether its conclusions are supported by the visual facts recorded on the board. The critic's role is limited to assessment: it neither generates an independent solution nor modifies the board. Algorithm 1 in Appendix D summarizes the inference-time control flow.

### 3.1 The Visual Board

Free-form interaction histories can obscure task-relevant visual observations and allow earlier misinterpretations to persist in subsequent reasoning. We therefore maintain a *Visual Board* as a compact, revisable workspace that records the solver's current solution state rather than a growing sequence of past responses. At Solver step $S$ within outer round $I$, the solver maintains a Visual Board $B_I^S$ consisting of three snapshots:

$$B_I^S = \left(F_I^{S,\text{img}}, U_I^{S,\text{img}}, R_I^S\right). \tag{1}$$

Here, $F_I^{S,\text{img}}$ (`image_facts`) records visual evidence obtained by inspecting the original image; $U_I^{S,\text{img}}$ (`uncertain_image_facts`) records visual information needed to solve the problem but not yet confirmed through visual inspection in the current step; and $R_I^S$ (`reasoning_steps`) records the solver's reasoning process, including inferred relations, formulas, calculations, and intermediate conclusions.

The solver maintains compact working notes during intermediate steps and submits complete snapshots of all three fields only through end-of-round terminal tool calls. These snapshots make task-relevant visual premises explicit for the text-only critic while omitting unrelated image details. However, making these premises explicit does not establish their correctness.

The original image remains the sole source of visual ground truth, and all board entries, including `image_facts`, remain provisional and revisable. Accordingly, unverified visual relations must be marked as *explicit assumptions*, and conclusions that depend on them must remain conditional. When direct inspection leaves an answer-critical detail unreadable or ambiguous, the solver can use visual tools to examine it further.

### 3.2 Rule-Based Compression

The harness controls the growth of textual context through deterministic, tool-aware compression. It limits the length of observation records and selectively reconstructs the interaction history using fixed structural and length-based rules. Neither operation uses additional model calls, semantic summarization, or question-conditioned selection.

**Compact observation records.** The harness applies type-specific length limits to tool outputs before adding them to the solver's context. Outputs from special and terminal tools are exempt from these entry-time limits.

For structured `visual_board_update` records, the harness retains up to 6 candidate facts, 3 corrections, and 3 uncertain observations from the beginning of their respective lists. Individual entries and raw excerpts are truncated to at most 320 and 500 characters, respectively, while source, status, and trace metadata are preserved. Candidate facts remain marked as `NOT COMMITTED`, distinguishing these tool-generated proposals from the solver's submitted Visual Board.

Visual outputs without a valid structured update are truncated to at most $1,200$ characters, and Python outputs to at most $1,800$ characters. All other ordinary tool outputs are subject to `max_observe`, which defaults to $3,000$ characters.

The harness also processes input prompts for completed steps, regardless of the compression mode. A completed-step prompt immediately preceding a tool call is replaced with a short completion marker, while all other completed-step prompts are removed.

**Trigger policy.** Let $H_I^S$ denote the interaction history after non-terminal solver step $S$ in outer round $I$. In the default *threshold* mode, the harness triggers history compaction when

$$N_{\text{msg}}(H_I^S) > 7 \quad \vee \quad N_{\text{char}}(H_I^S) > 4500, \tag{2}$$

where $N_{\text{msg}}$ counts messages and $N_{\text{char}}$ counts characters in message text, excluding image data and tool-call arguments. These thresholds determine when compaction is triggered; they do not impose a token budget on the compacted history.

The *step* mode applies the same threshold test at $S = 1$ and attempts compaction after every subsequent non-terminal step ($S \geq 2$), regardless of whether the thresholds are exceeded. The *always* mode attempts compaction after every non-terminal step. The *off* mode disables routine history compaction while retaining observation truncation and completed-step prompt processing.

When the step limit is reached, the forced-termination procedure makes a final compaction attempt regardless of the selected mode.

**Selective history reconstruction.** When compaction is triggered, the harness retains the original image-bearing user message and the latest assistant tool-call message together with its matching responses. These responses remain in their stored, entry-time-truncated form. An optional completed-step marker and subsequent non-tool messages are also retained.

Older history is replaced with compact records of selected tool outputs, using fixed rules based on tool type and message order. Previous summaries and earlier free-form reasoning are not merged into these records. The reconstructed history thus provides a selective working context rather than a complete interaction record. Further details of this reconstruction procedure, including the selection rules, length limits, and handling of boundary cases, are provided in Appendix A.2.

### 3.3 Query-Driven Visual Tools and Structured Submission

The harness separates visual evidence acquisition from structured state submission. Visual tools support targeted inspection, while terminal tools submit the solver's current solution state for outer-loop review. A step-dependent policy governs tool availability.

**Query-driven visual inspection.** Both VLM-based visual tools receive a focused visual query, along with a verbatim copy of the original question and any answer options.
`grid_caption` supports local inspection using a solver-specified bounding box $b = (x_1, y_1, x_2, y_2) \in [0, 1]^4$ in normalized image coordinates. When the target region is known, the tool answers the query using the crop, for example, to inspect a label, table cell, or diagram marker. `vlm_query`, in contrast, operates on the full original image. It supports queries about relations spanning multiple regions or visual details whose locations are uncertain.

Visual tools are invoked when direct inspection leaves an answer-critical detail unreadable or ambiguous. Queries must remain neutral and observation-oriented, without implying an expected answer or requesting derivations or calculations. The solver may revisit the same visual target only

when the previous observation is insufficient or contradictory, and each revisit must modify the inspected region, the tool, or the requested visual fact.

**Structured terminal submission.** To structure the handoff from the solver to the critic, we define two terminal tools. Both submit complete snapshots of all three Visual Board fields rather than incremental updates. Each call ends the solver's interaction within the current round and passes the submission to the critic for review, without necessarily terminating the overall process.

`terminate_and_answer` additionally submits an answer draft, a confidence level, and zero-based indices of the `image_facts` entries supporting the answer. `need_more_visual_observation` instead identifies missing visual details, specifies their locations and relevance to the answer, and designates a visual tool for the next round. This second tool is reserved for missing visual evidence that can be acquired in a further round, not for arithmetic, general reasoning revision, or redundant confirmation.

Following a visual handoff, the harness protocol requires the designated tool to execute successfully, address the requested visual details, and return a structured result before the next terminal submission can be accepted. For both terminal tools, complete Visual Board snapshots are submitted through the tool-call arguments, while the corresponding responses contain only brief acknowledgements to avoid duplicating the submitted state.

### 3.4 Harness-Aware Training

We initialize the VLM Solver with Qwen3-VL-4B-Instruct (Bai et al., 2025) and adapt it to the harness through supervised fine-tuning. The training data consist of decision-level input–response pairs extracted from teacher trajectories, as described in Appendix C.1.

For each example, let $x = (\mathcal{I}, Q, \widetilde{H}, \mathcal{T})$ denote the input to a Solver decision, where $\mathcal{I}$ and $Q$ denote the image and question, respectively. The context $\widetilde{H}$ is provided by the harness at that decision and includes the current Visual Board and any retained tool observations or Critic feedback. The tool schemas available at that step are specified by $\mathcal{T}$. The target $y = (y_1, \ldots, y_{|y|})$ is the corresponding teacher response, including its text and any serialized tool calls.

For the resulting training set $\mathcal{D}_{\mathrm{SFT}}$, we minimize the token-level negative log-likelihood:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\sum_{(x,y)\in\mathcal{D}_{\mathrm{SFT}}} \sum_{j=1}^{|y|} \log \pi_\theta(y_j \mid x, y_{<j}). \tag{3}$$

Only the current Solver response contributes to the loss; historical Solver replies, tool observations, and Critic feedback serve as conditioning context with masked labels.

## 4 Experiments

### 4.1 Experimental Setup

**Datasets.** We evaluate the harness on three benchmarks covering reasoning over geometric diagrams, three-dimensional structures, and charts. Euclid 30K (Lian et al., 2026) contains plane and solid geometry problems, with 500 examples in its validation split. SolidGeo (Wang et al., 2026) focuses on solid geometry, including projection, unfolding, spatial measurement, and spatial vectors. Our evaluation uses 2,206 single-image examples prepared from this dataset. For chart reasoning, we evaluate on the bar-chart subset of ChartQA (Masry et al., 2022) used in VTool-R1 (Wu et al., 2026a), referred to as Chart QA throughout this paper. This subset contains 826 question–answer pairs, comprising 444 horizontal and 382 vertical bar-chart examples.

**Model-only baselines.** All model-only baseline scores are obtained from our own evaluations using the official implementations for the corresponding benchmarks: Euclid 30K and SolidGeo for their respective datasets, and VTool-R1 for the Chart QA bar-chart subset. We follow the default evaluation configurations provided by these implementations. The exact evaluation settings and prompts used in our experiments are reported in Appendix E.

**Supervised fine-tuning.** We fine-tune Qwen3-VL-4B-Instruct on four A800 GPUs using the decision-level examples described in Appendix C.1. The autoregressive cross-entropy loss is applied only to the current assistant response, including its textual output and serialized tool calls, while all preceding context is masked from the loss. We update the language model and multimodal aligner while keeping the vision encoder frozen. Training runs for one epoch with a learning rate of $10^{-5}$ for all trainable components, using BF16, ZeRO-3, and gradient checkpointing. Examples exceeding either the prompt-length or total sequence-length limit are discarded rather than truncated, preserving the complete recorded context for each supervised decision.

### 4.2 Main Results

Table 1: Comparison with baseline models on three datasets.

| Method | Euclid 30K | SolidGeo | Chart QA |
|---|---|---|---|
| Qwen3-VL-4B | 39.40 | 32.11 | 57.75 |
| Qwen3-VL-4B (ours) | **52.61** 13.21↑ | **37.08** 4.97↑ | **78.69** 20.94↑ |
| Qwen3-VL-8B | 51.40 | 35.99 | 57.99 |
| Qwen3-VL-8B (ours) | **58.75** 7.35↑ | **47.20** 11.21↑ | **81.60** 23.61↑ |
| Qwen3-VL-Flash | 50.80 | 39.83 | 68.52 |
| Qwen3-VL-Flash (ours) | **56.90** 6.10↑ | **58.25** 18.42↑ | **86.56** 18.04↑ |
| Qwen3-VL-32B | 57.40 | 41.02 | 61.86 |
| Qwen3-VL-32B (ours) | **69.08** 11.68↑ | **59.34** 18.32↑ | **86.68** 24.82↑ |

**Comparison with model-only baselines.** Tab. 1 summarizes accuracy across the three benchmarks, including our model-only evaluations of the Qwen3-VL family (Bai et al., 2025). Our harness outperforms the corresponding model-only baselines in every available comparison. Model configurations for all harness components are listed in Appendix A.

With Qwen3-VL-8B, the harness achieves 58.75%, 47.20%, and 81.60% accuracy on Euclid 30K, SolidGeo, and Chart QA, respectively, corresponding to gains of **7.35**, **11.21**, and **23.61** percentage points over the model-only baselines. Among the evaluated configurations, Qwen3-VL-32B attains the highest accuracy on all three benchmarks, reaching 69.08%, 59.34%, and 86.68%, with respective gains of **11.68**, **18.32**, and **24.82** percentage points. The harness also yields substantial improvements with Qwen3-VL-Flash. Its accuracy on Euclid 30K increases from 50.80% to 56.90%, while performance on SolidGeo and Chart QA rises by **18.42** and **18.04** percentage points to 58.25% and 86.56%, respectively. Overall, the gains remain consistent across model configurations and are particularly pronounced on SolidGeo and Chart QA.

As shown in the Chart QA column of Tab. 1, our harness improves accuracy by more than **18** percentage points across all evaluated configurations, demonstrating particularly large gains on structured visual reasoning. Beyond these absolute improvements, Tab. 1 also reveals a notable cross-scale result: Qwen3-VL-8B equipped with our harness outperforms the Qwen3-VL-32B model-only baseline on all three benchmarks. This comparison shows that a smaller model augmented with our harness can outperform a substantially larger model under direct question answering. Further cross-scale comparisons are provided in Appendix C.4 (Fig. 8).

**Comparison with ProReason.** We further compare our harness with ProReason (Zhou et al., 2025), a multi-agent framework that separates visual information gathering from textual reasoning and connects the two through textual memory. In contrast, our harness keeps the VLM responsible for both visual inspection and reasoning, with the Visual Board serving as a revisable reference and the text-only LLM acting as a critic.

Table 2: Comparison on Euclid 30K.

| Method | Accuracy (%) |
|---|---|
| Pro_Reason | 51.60 |
| Ours | **58.75** 7.15↑ |

We evaluate both frameworks on the Euclid 30K validation split using the same Qwen3-VL-8B-Instruct VLM and Qwen3-8B LLM. As shown in Tab. 2, our harness achieves 58.75% accuracy, compared with 51.60% for ProReason, corresponding to an improvement of **7.15** percentage points under the same backbone configuration.

### 4.3 Small-Model Performance and Fine-Tuning

**Training-free performance.** We further examine whether the harness remains effective with a smaller Qwen3-VL-4B solver. As shown in Fig. 3, without additional training, the harness improves over the corresponding model-only baselines on Euclid 30K, SolidGeo, and Chart QA by **13.21**, **4.97**, and **20.94** percentage points, respectively. Notably, on Euclid 30K, the 4B solver equipped with our harness achieves 52.61% accuracy, exceeding the 51.40% accuracy of the Qwen3-VL-8B model-only baseline. These results show that the benefits of the harness extend to smaller model scales without requiring additional training.

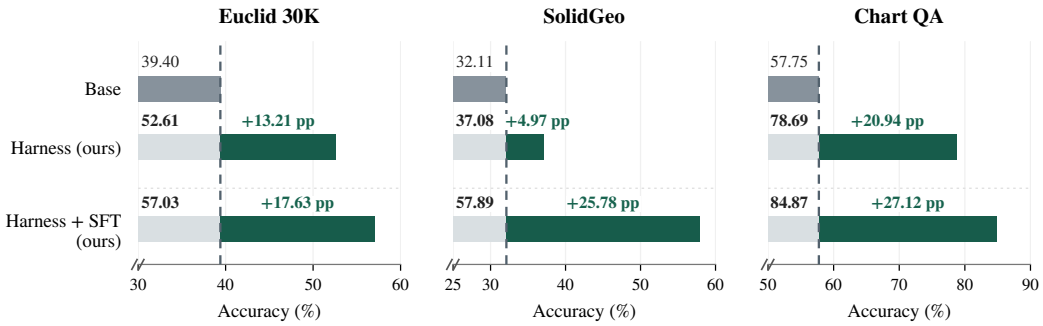



Figure 3: Fine-tuning and cross-benchmark transfer of Qwen3-VL-4B. *Harness* uses the original solver without additional training, whereas *Harness + SFT* uses the same harness with the solver fine-tuned on the Euclid 30K training split.

**Performance after fine-tuning.** We further evaluate the 4B harness after supervised fine-tuning of the solver on the Euclid 30K training split, following the procedure described in Sec. 3.4. The resulting harness achieves 57.03%, 57.89%, and 84.87% accuracy on Euclid 30K, SolidGeo, and Chart QA, respectively, outperforming both the corresponding model-only baselines and the training-free 4B harness across all three benchmarks.

Importantly, the gains are not limited to the training domain. On Chart QA, the fine-tuned 4B harness improves over its training-free counterpart by **6.18** percentage points and exceeds the training-free Qwen3-VL-8B harness by **3.27** percentage points (Tab. 1). These results suggest that fine-tuning the solver to operate within the harness can transfer beyond the geometric reasoning domain used for training. Further cross-domain analysis is provided in Appendix C.3.

We additionally explore reinforcement learning (RL) for solver post-training, with results and analysis reported in Appendix C.5.

### 4.4 Ablation Studies

**RQ1: Visual Board.** We compare four settings: (i) *Concise (default)*, using the board prompt in Sec. 3.1; (ii) *Exhaustive*, using a detailed prompt to enumerate all visual features and relations; (iii) *No sharing*, retaining the board for solver use while excluding it from the critic's input and the feedback returned to the solver; and (iv) *Board off*, disabling board construction, use, and sharing entirely.

Fig. 4 compares the four board configurations on the same 50 examples from the Euclid 30K validation split, with history compression enabled throughout. The concise shared board achieves both the highest accuracy (64%) and the lowest average execution time (155.04 seconds per example). Retaining the board for the

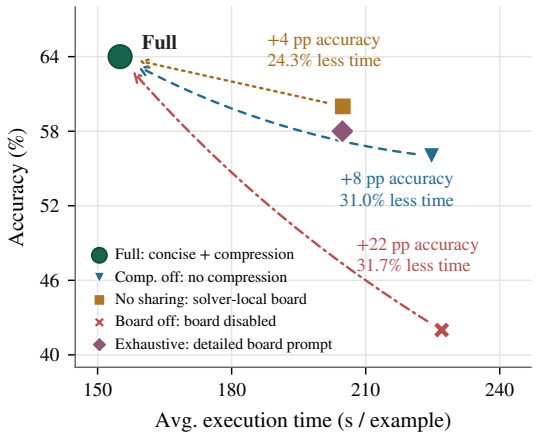



Figure 4: Visual Board ablation study.

solver without sharing it with the critic yields 60% accuracy at 204.87 seconds per example, while removing the board entirely reduces accuracy to 42% at 227.00 seconds per example. Relative to these two settings, the concise shared board improves accuracy by **4** and **22** percentage points and achieves $1.32\times$ and $1.46\times$ speedups, respectively.

The exhaustive board prompt reaches 58% accuracy with an average execution time of 204.79 seconds, indicating that a more detailed board does not provide additional benefit on this 50-example pilot. Overall, these results support the use of a concise Visual Board shared between the solver and critic, rather than either retaining it only within the solver or removing it altogether.

**RQ2: Does compression improve efficiency?** We fix the concise Visual Board configuration and vary only whether history compression is enabled in the VLM Solver. As shown in Fig. 4, enabling compression increases accuracy from 56% to 64% on the same 50-example subset, an improvement of **8** percentage points. It also reduces the average execution time from 224.75 to 155.04 seconds per example, corresponding to a 31.0% reduction and a $1.45\times$ speedup.

On this evaluation subset, history compression improves both accuracy and efficiency, suggesting that reducing accumulated textual context can lower execution cost without necessarily sacrificing reasoning performance.

**RQ3: Effectiveness of visual-tool configuration.** We compare four visual-tool configurations on the same fixed batch: no Query/Crop, Query only, Crop only, and both Query and Crop. As shown in Tab. 3, disabling both tools yields the shortest average execution time but lower accuracy, while the Query-only configuration performs worst in accuracy and incurs the highest execution time.

Table 3: Visual-tool interface ablation study.

| Setting | Acc. (%) | Time (s/ex.) |
|---|---|---|
| Baseline | 60.00 | **146.45** |
| + Query | 58.00 | 224.72 |
| + Crop | **66.00** | 169.41 |
| Both | 64.00 | 155.04 |

Among the tool-enabled settings, Crop-only achieves the highest accuracy at 66%, compared with 64% for the combined configuration. The combined setting, however, reduces the average execution time from 169.41 to 155.04 seconds per example. These results indicate that combining Query and Crop provides a more favorable accuracy–efficiency trade-off on this batch, while Crop alone attains the highest absolute accuracy.

## 5 Conclusion

Multi-step visual reasoning requires retaining useful intermediate evidence without allowing earlier visual misinterpretations to become fixed premises for subsequent reasoning. We address this problem with a Visual Board-centered harness that maintains a selective and revisable reasoning state while preserving direct access to the original image. The harness combines a VLM solver, query-driven visual inspection, deterministic history compression, and a text-only LLM critic.

Across Euclid 30K, SolidGeo, and Chart QA, the training-free harness consistently improves over model-only baselines across model scales, while decision-level supervised fine-tuning provides further gains with a smaller 4B solver. Our ablations further show that concise shared memory and history compression can improve both reasoning accuracy and execution efficiency. These results suggest that effective multi-step visual reasoning depends not only on additional tools or longer interaction trajectories, but also on maintaining intermediate visual evidence in a form that can be selectively retained, re-examined, and revised as reasoning progresses.

**Limitations and future directions.** Our harness relies on prompting to distinguish the original image as the source of visual ground truth from the Visual Board as a revisable reference. While models with 8B parameters or more generally follow this distinction, the 4B model does so less consistently, sometimes treating previously recorded board content as stronger evidence than intended. It also occasionally produces invalid or schema-inconsistent tool calls, which trigger retries and increase execution cost. Supervised fine-tuning mitigates both issues, but improving protocol adherence in smaller models remains an important direction. Targeted reinforcement learning may further encourage reliable use of the board as auxiliary memory, improve tool-call validity, and preserve direct visual verification when earlier observations are uncertain or conflicting.

## 6 AI USE STATEMENT

We used artificial intelligence (AI) tools to assist with language polishing and to support code debugging during the development of this work. All scientific ideas, methodologies, experimental designs, analyses, and conclusions were independently developed and verified by the authors. AI tools were only used as auxiliary tools to improve writing quality and facilitate the debugging process.

## 7 ETHICS STATEMENT

This work does not involve any ethical concerns related to human subjects, personal data, or potentially harmful applications. The authors have carefully considered the ethical implications of the proposed research and believe that the methods and results presented in this paper do not introduce additional ethical risks.

## 8 REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we will publicly release all necessary source code, experimental configurations, and generated figures required to reproduce the results reported in this paper. The proposed methodology is described in detail throughout the paper and its supplementary materials, allowing readers to implement and verify the approach independently.

## REFERENCES

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

Xingyu Fu, Minqian Liu, Zhengyuan Yang, John Richard Corring, Yijuan Lu, Jianwei Yang, Dan Roth, Dinei Florencio, and Cha Zhang. Refocus: Visual editing as a chain of thought for structured image understanding. In *ICLR 2025 Workshop on Foundation Models in the Wild*, 2025. URL https://openreview.net/forum?id=rkvMUjeEcr.

Yunhao Gou, Kai Chen, Zhili Liu, Lanqing Hong, Xin Jin, Zhenguo Li, James Kwok, and Yu Zhang. Reasoning-aligned perception decoupling for scalable multi-modal reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 140359–140415, 2026.

Yejie Guo, Yunzhong Hou, Wufei Ma, Meng Tang, and Ming-Hsuan Yang. Pursuing minimal sufficiency in spatial reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 121192–121222, 2026.

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

James Y. Huang, Sheng Zhang, Qianchu Liu, Guanghui Qin, Tinghui Zhu, Tristan Naumann, Muhao Chen, and Hoifung Poon. Be my eyes: Extending large language models to new modalities through multi-agent collaboration, 2025. URL https://arxiv.org/abs/2511.19417.

Sungheon Jeong, Ryozo Masukawa, Jihong Park, Sanggeon Yun, Wenjun Huang, Hanning Chen, Mahdi Imani, and Mohsen Imani. Draft and refine with visual experts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18816–18826, 2026.

Hongrui Jia, Chaoya Jiang, Shikun Zhang, and Wei Ye. Mitigating visual context degradation in large multimodal models: A training-free decoupled agentic framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9574–9585, 2026.

Yue Jiang, Xue Jiang, Lihua Zhang, Zhiqiang Wang, Yuhang Lu, Peng Wang, Bo Han, Feng Zheng, and Dingkang Yang. MM-snowball: Evaluating and mitigating hallucination snowballing in multimodal multi-turn dialogue. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=vL1xAHOUmd.

Somnath Kumar, Yash Gadhia, Tanuja Ganu, and Akshay Nambi. Mmctagent: Multi-modal critical thinking agent framework for complex visual reasoning. 2024. URL https://www.microsoft.com/en-us/research/publication/mmctagent-multi-modal-critical-thinking-agent-framework-for-complex-visual-reasoning.

Shijie Lian, Changti Wu, Laurence Tianruo Yang, Hang Yuan, Bin Yu, Lei Zhang, and Kai Chen. Euclid's gift: Enhancing spatial perception and reasoning in vision-language models via geometric surrogate tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pp. 9824–9835, June 2026.

Meng Lu, Ran Xu, Yi Fang, Wenxuan Zhang, Yue Yu, Gaurav Srivastava, Yuchen Zhuang, Mohamed Elhoseiny, Charles Fleming, Carl Yang, et al. Scaling agentic reinforcement learning for tool-integrated reasoning in vlms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26518–26529, 2026.

Damiano Marsili and Georgia Gkioxari. No labels, no problem: Training visual reasoners with multimodal verifiers. In *International Conference on Learning Representations*, volume 2026, pp. 72936–72968, 2026.

Ahmed Masry, Jia Qing Tan, Shafiq Joty, Enamul Hoque, et al. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the association for computational linguistics: ACL 2022*, pp. 2263–2279, 2022.

Ahmed Masry, Mohammed Saidul Islam, Mahir Ahmed, Aayush Bajaj, Firoz Kabir, Aaryaman Kartha, Md Tahmid Rahman Laskar, Mizanur Rahman, Shadikur Rahman, Mehrad Shahmohammadi, et al. Chartqapro: A more diverse and challenging benchmark for chart question answering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 19123–19151, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Chufan Shi, Cheng Yang, Yaokang Wu, Linghao Jin, Bo Shui, Taylor Berg-Kirkpatrick, and Xuezhe Ma. Are VLMs seeing or just saying? uncovering the illusion of visual re-examination. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=DdU1o2ZvWi.

Weikang Shi, Aldrich Yu, Rongyao Fang, Houxing Ren, Ke Wang, Aojun Zhou, Changyao Tian, Xinyu Fu, Yuxuan Hu, Zimu Lu, Linjiang Huang, Si Liu, Rui Liu, and Hongsheng Li. Mathcanvas: Intrinsic visual chain-of-thought for multimodal mathematical reasoning, 2025. URL https://arxiv.org/abs/2510.14958.

Mingyang Song, Haoyu Sun, Jiawei Gu, Linjie Li, Ranjay Krishna, and Yu Cheng. Adareasoner: Dynamic tool orchestration for iterative visual reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 142731–142754, 2026.

Zhaochen Su, Peng Xia, Hangyu Guo, Zhenhua Liu, Yan Ma, Xiaoye Qu, Jiaqi Liu, Yanshu Li, Kaide Zeng, Zhengyuan Yang, et al. Thinking with images for multimodal reasoning: Foundations, methods, and future frontiers. *arXiv preprint arXiv:2506.23918*, 2025.

Peijie Wang, Chao Yang, Zhong-Zhi Li, Dekang Ran, Mi Tian, Zhilong Ji, Jinfeng Bai, Cheng-lin Liu, et al. Solidgeo: Measuring multimodal spatial math reasoning in solid geometry. *Advances in Neural Information Processing Systems*, 38, 2026.

Yikun Wang, Siyin Wang, Qinyuan Cheng, Zhaoye Fei, Liang Ding, Qipeng Guo, Dacheng Tao, and Xipeng Qiu. Visuothink: Empowering lvlm reasoning with multimodal tree search. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21707–21719, 2025.

Lai Wei, Liangbo He, jun lan, Lingzhong Dong, Yutong Cai, Siyuan Li, Huijia Zhu, Weiqiang Wang, Linghe Kong, Yue Wang, Zhuosheng Zhang, and Weiran Huang. Zooming without zooming: Region-to-image distillation for fine-grained multimodal perception. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=bVXx0K0j4c.

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

Zhiheng Wu, Tong Wang, Shuning Wang, Naiming Liu, and Yumeng Zhang. See further, think deeper: Advancing vlm's reasoning ability with low-level visual cues and reflection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18870–18880, June 2026b.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2087–2098, October 2025.

Haoxuan You, Rui Sun, Zhecan Wang, Long Chen, Gengyu Wang, Hammad Ayyubi, Kai-Wei Chang, and Shih-Fu Chang. Idealgpt: Iteratively decomposing vision and language reasoning via large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 11289–11303, 2023.

Weijia Zhang, Zijia Liu, Haoru Li, Haoqi Chen, and Jiaxuan You. Seeingeye: Agentic information flow unlocks multimodal reasoning in text-only llms, 2025. URL https://arxiv.org/abs/2510.25092.

YiFan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, Haonan fan, Kaibing Chen, Jiankang Chen, Haojie Ding, Kaiyu Tang, Zhang Zhang, Liang Wang, Fan Yang, Tingting Gao, and Guorui Zhou. Thyme: Think beyond images. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=gCWLkqK45O.

Yongchang Zhang, Oliver Ma, Tianyi Liu, Guangquan Zhou, and Yang Chen. See it, say it, sorted: An iterative training-free framework for visually-grounded multimodal reasoning in lvlms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11933–11942, June 2026b.

Weihong Zhong, Xiaocheng Feng, Liang Zhao, Qiming Li, Lei Huang, Yuxuan Gu, Weitao Ma, Yuan Xu, and Bing Qin. Investigating and mitigating the multimodal hallucination snowballing in large vision-language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11991–12011, 2024.

Jingqi Zhou, Sheng Wang, Jingwei Dong, Kai Liu, Lei Li, Jiahui Gao, Jiyue Jiang, Lingpeng Kong, and Chuan Wu. Proreason: Multi-modal proactive reasoning with decoupled eyesight and wisdom. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 31650–31679, 2025.

Yunpeng Zhou. Diagnosing failure modes of shared-state collaboration in resource-constrained visual agents. In *Workshop on Failure Modes of Agentic AI at ICML 2026*, 2026. URL https://openreview.net/forum?id=LvvVx0RieJ.

## APPENDIX CONTENTS

A Harness Settings . . . . . 13
B Qualitative Analysis . . . . . 14
C Additional Experimental Details and Analysis . . . . . 17
D Harness Algorithm . . . . . 19
E Experimental Settings and System Prompts . . . . . 20

# A HARNESS SETTINGS

## A.1 BASIC CONFIGURATION

**Model configuration.** The harness has three model-backed components: the VLM Solver, the backend model serving the visual-tool API calls made by the Solver (Crop Grid and Query VLM), and the text-only LLM Critic. Tab. 4 lists the configurations used for the harness results in Tab. 1. For the 8B and 32B configurations, the visual tools use the same model as the Solver, while the Critic uses the correspondingly sized Qwen3 LLM. The Flash configuration likewise uses Qwen3-VL-Flash for both the Solver and visual tools, paired with Qwen3-Max-Preview as the Critic.

Table 4: Model configurations for the harness results in Tab. 1. The visual-tool API backend serves both Crop Grid and Query VLM.

| Solver | Visual-tool API backend | Critic |
|---|---|---|
| Qwen3-VL-8B-Instruct | Qwen3-VL-8B-Instruct | Qwen3-8B |
| Qwen3-VL-Flash | Qwen3-VL-Flash | Qwen3-Max-Preview |
| Qwen3-VL-32B-Instruct | Qwen3-VL-32B-Instruct | Qwen3-32B |
| Qwen3-VL-4B-Instruct | Qwen3-VL-Flash | Qwen3-Max-Preview |

**The 4B configuration.** The 4B Solver uses a different configuration: Qwen3-VL-Flash serves the visual-tool API calls, and Qwen3-Max-Preview acts as the Critic. Accordingly, "4B" identifies the Solver size, not the size of every model in the harness.

**Ablation study configuration.** All ablation experiments use Qwen3-VL-32B-Instruct for both the Solver and the visual-tool API backend, and Qwen3-32B as the Critic. For a controlled comparison, all configurations are evaluated on the same 50 examples sampled from the Euclid 30K validation split with random seed 1. The harness processes for all ablation runs are launched on the same host machine.

## A.2 SELECTIVE HISTORY RECONSTRUCTION

This subsection specifies the reconstruction rules applied when history compaction is triggered (Sec. 3.2).

**Directly retained messages.** The harness preserves the original image-bearing user message and the latest assistant tool-call message together with its matching tool responses, thereby maintaining call–response pairing. These responses are retained in their stored, entry-time-truncated form, without applying the additional limits for older records described below. An optional completed-step marker and subsequent non-tool messages are also retained.

**Selection from older tool outputs.** The remaining history is replaced with compact records drawn only from older tool messages. Selection follows fixed rules based on tool type and message order, retaining the following results when available:

- **Visual tools:** the first three visual results. Structured updates are retained when present; otherwise, each result's text is truncated to at most 700 characters.
- **Python:** the most recent Python result, truncated to at most 1,200 characters.
- **Other tools:** the first three other-tool results.

Previous summaries and earlier free-form reasoning are not merged into these records and may be discarded. Thus, reconstruction provides a selective working context rather than a complete interaction record.

**Boundary cases.** Reconstruction is skipped if no assistant tool-call message is present. If no image-bearing user message is available, the first user message is retained instead.

# B Qualitative Analysis

We examine recorded execution traces to illustrate how the Solver uses visual tools and revises the Visual Board.

**Regular steps and recovery calls.** The protocol allows up to three regular Solver steps per round, with two additional calls permitted for protocol recovery. First, if a generated tool call does not conform to the required schema, the harness rejects the call and allows one corrective retry. Second, if the Solver reaches the three-step limit without a valid terminal submission, for example after three visual-tool calls, the harness applies the forced-termination policy and requests a final answer through `terminate_and_answer`.

These recovery calls do not count toward the three-step regular budget. Thus, a round may contain a fourth Solver call even though it contains at most three regular reasoning steps: a retry replaces an invalid call, whereas forced termination closes an otherwise unfinished round. For the qualitative traces reported below, the execution logger was configured with `max_steps=4`. Consequently, an S4 label indicates only the fourth recorded Solver call and does not by itself imply that the call was a retry or a forced-termination step.

## B.1 Matched-Run Analysis

We compare the *Concise (default)* and *No sharing* settings from Sec. 4.4 on Euclid validation example 14. Both use the Board prompt in Sec. 3.1 and share the input, models, decoding settings, and interaction budget. In *No sharing*, the Board remains available for the Solver's own use but is excluded from both the Critic's input and the feedback returned to the Solver. The default setting instead shares the Board through this feedback loop.

**No sharing (Fig. 5b).** The Solver still queries the image, but focuses on whether the two angles have explicit equality marks. It records the absence of such marks and returns the incorrect answer, `no`, which the text-only Critic accepts.

**Concise (default) (Fig. 5c).** The Solver initially records $\angle DFG = 105^\circ$ and answers `no`. The Solver flags this visual premise for re-examination, and the next round carries the disputed Board entry and a recheck request in working memory. The Solver then queries the relevant point ordering and ray alignment. The tool outputs report that $FD$ and $FC$ lie on the same ray, as do $FG$ and $FE$. Using these outputs, the Solver revises the Board and changes its answer to the reference answer, `yes`, while retaining uncertainty about the angle relation. The run reaches its budget with `needs_revision`, so answer correctness does not imply Critic acceptance. The two traces illustrate different inspection targets: explicit angle marks under *No sharing* versus the disputed ray relations under *Concise (default)*.

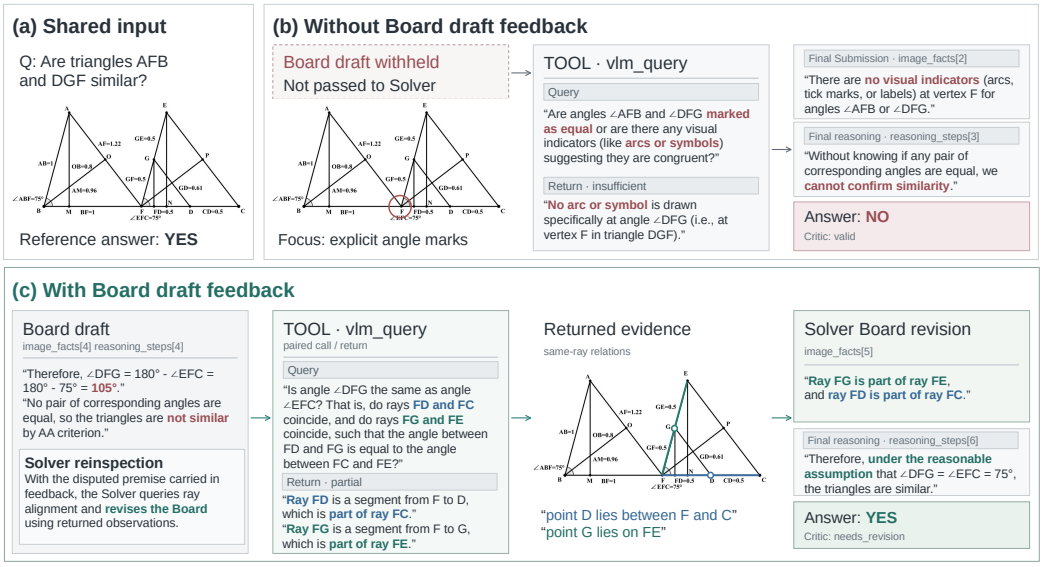



Figure 5: Solver reinspection and Board revision in matched runs. (a) Shared input and reference answer. (b) Without Board feedback, the Solver focuses on explicit angle marks and answers `no`. (c) With Board feedback, the Solver queries ray alignment, revises the Board using returned evidence, and answers `yes`. Bracketed indices are zero-based Board-entry positions.

### B.2 Additional Visual Board Visualizations

The following two recorded items illustrate complementary outcomes: a corrected visual description with an unchanged answer, and a persistent visual error despite repeated inspection. Both use Qwen3-VL-32B-Instruct for the Solver and visual tools, with Qwen3-32B as the Critic. Their recorded configurations allow up to three outer iterations and four nominal Solver steps per iteration, including terminal submission. The figures show selected tool interactions and submitted Board snapshots, rather than every call in the traces. I$n$/S$m$ denotes the recorded iteration and step, and bracketed indices identify zero-based Board entries.

**Observation repair with a stable correct answer.** Fig. 6 shows SolidGeo item 2037 (recorded split: `train`; episode `0eeaa8cd56a080eb487b7edd`). The task asks for the base radius of a cone formed from sector $ADE$ in rectangle $ABCD$, given $AB = BE = 3$. The Solver's first submission already gives the correct answer, B, $3\sqrt{2}/8$, but its Board incorrectly describes the shaded region as triangle $EDC$.

In the second round, Crop Grid inspects the region around $E$, $D$, and $C$. Its output identifies a curved boundary but leaves the region's geometric identity uncertain. A subsequent whole-image Query VLM call specifies the boundary as segments $EC$, $CD$, and arc $DE$. After checking the radius calculation with Python, the Solver replaces `image_facts[3]` with this boundary description and resubmits the same answer, which the Critic accepts. Thus, the observed improvement is a repair of the recorded visual premise, not a change from an incorrect to a correct answer. Critic acceptance also does not imply a fully clean trace: the saved audit records one image-fact contamination and marks the episode as not strictly eligible.

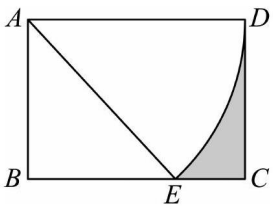
| (a) Input | (b) Draft | (c) Reinspection | (d) Board state | (e) Output |
|---|---|---|---|---|
| <br>AB = BE = 3. Sector ADE is folded into a cone.<br>**Question.** Base radius? | I1/S2 · [3]<br>"The shaded region is **triangle EDC** ..."<br>Draft: B | I2/S1 · Crop<br>Curved boundary; identity uncertain.<br>I2/S2 · Query<br>EC, CD, arc DE.<br><br>Tool-returned crop | I2/S4 · [3]<br>"The shaded region is bounded by segment EC, segment CD, and **arc DE**." | **B: 3√2/8**<br>Unchanged<br>Critic: accepted |

Figure 6: Visual-description repair on SolidGeo item 2037. Crop and whole-image queries lead the Solver to replace the triangle description with an arc boundary. Panel (c) shows the tool-returned crop. The correct answer, B ($3\sqrt{2}/8$), remains unchanged.

**Failure under repeated inspection.** Fig. 7 shows Euclid30K validation item 0 (episode `2f51b6b7aee8f8c41c774e60`) under *Concise (default)*. The task asks for the shaded area in a semicircle construction on a right triangle with $AB = 5$, $AC = 4$, and $\angle C = 90°$; the reference answer is 6. An early visual query incorrectly reports that the entire semicircle on hypotenuse $AB$ is shaded. Later queries give conflicting accounts of the shaded regions, including the portions of the smaller semicircles outside the larger one, but the Solver's final Board still claims that the semicircle on $AB$ is fully shaded. Its final answer remains $6 + 25\pi/8$, incorrectly including the area of that semicircle.

The trace also contains rejected answer submissions with invalid tool arguments and ends with an unsatisfied required action (`required_action_not_satisfied`). On the same item, *No sharing* returns 6, but also records a tool-argument error and reaches the round limit without acceptance (`max_iterations_needs_revision`). Neither run receives final Critic acceptance. This case shows that repeated visual queries need not resolve an incorrect premise: conflicting observations can leave the Board, and hence the subsequent reasoning, unchanged in the crucial respect.

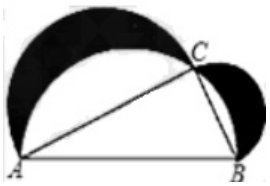
| (a) Input | (b) Draft | (c) Reinspection | (d) Board state | (e) Output |
|---|---|---|---|---|
| <br>∠C = 90°, AB = 5, AC = 4.<br>**Question.** Shaded area?<br>Reference: 6 | I1/S2 · Query<br>Semicircle on AB is **shaded black**. | I2/S1-S3;<br>I3/S1-S2 · Query<br>Conflicting descriptions of the shaded regions. | I3/S4 · [2]<br>Semicircle on AB "is drawn outward from the triangle and is **fully shaded**." | **6 + 25π/8**<br>Required action not satisfied |

Figure 7: Persistent visual error on Euclid30K validation item 0. Repeated queries yield conflicting shading descriptions, but the final Board retains the fully shaded-semicircle claim. The Solver returns $6 + 25\pi/8$ instead of 6, and the required action remains unsatisfied.

# C Additional Experimental Details and Analysis

## C.1 SFT Data Construction

We collect 1,557 rollouts from the Euclid 30K training split using the complete harness, with Qwen3-VL-Flash serving as both the VLM Solver and the visual tool backend, and Qwen3-Max as the LLM Critic. We retain rollouts with correct final answers, valid interaction protocols, and well-formed Visual Boards.

From each retained rollout, every accepted and successfully executed Solver decision is converted into a separate training example. Each example pairs the corresponding teacher response with the multimodal context available at decision time, including the image and question, interaction history, tool observations, critic feedback, and available tool schemas. A single trajectory can therefore yield multiple decision-level examples.

We partition source questions into training and validation sets before expanding trajectories into decision-level examples, ensuring that all examples derived from the same source question remain within a single split.

## C.2 Source composition and SolidGeo.

The 1,557 collected rollouts are sampled from the 30,000-example Euclid 30K training split rather than collected over the full split. Euclid 30K includes SolidGeo among its data sources (Lian et al., 2026), but the number of SolidGeo-derived examples in the training split or in our sampled rollouts is not recorded. Our SFT data are therefore constructed only from the retained rollouts within this sampled collection, rather than from all 30K training examples.

Because SolidGeo does not provide a dedicated validation split, the available provenance information is insufficient to determine how many of the 1,557 rollouts originate from SolidGeo or whether any overlap with the SolidGeo evaluation instances. For this reason, we do not emphasize the SolidGeo performance of the 4B Solver fine-tuned on Euclid 30K rollouts, nor do we interpret it as evidence of out-of-domain generalization.

## C.3 Cross-domain transfer from geometry to charts.

The solver is fine-tuned only on harness trajectories collected from the Euclid 30K training split, with no Chart QA examples used for SFT. Chart QA therefore evaluates transfer from geometry-domain fine-tuning to chart reasoning rather than in-domain performance on the training source. On Chart QA, the fine-tuned 4B solver within the same harness achieves 84.87% accuracy, improving over its training-free counterpart by 6.18 percentage points (Tab. 1). This result provides evidence that the learned solver behavior transfers beyond the geometry domain used for fine-tuning. In contrast, because of the shared source provenance discussed above, the improvement on SolidGeo should not be interpreted as unambiguous evidence of transfer to an unseen data source.

## C.4 Cross-Scale Comparisons

**Smaller solvers outperform larger Base models.** Fig. 8 examines whether the harness enables a smaller solver to outperform a larger Base model without additional SFT. Triangles compare 4B Harness with 8B Base, while ellipses compare 8B Harness with 32B Base. The horizontal axis shows the accuracy margin over the larger Base model, and the vertical axis shows the improvement over the solver's own Base, both in percentage points (pp). All six points are positive along both axes, indicating that the harness improves each solver relative to its own Base while also outperforming the corresponding larger Base model across all three benchmarks.

**Gains across solver sizes.** With the 4B solver, the harness outperforms 8B Base by 1.21 pp on Euclid 30K, 1.09 pp on SolidGeo, and 20.70 pp on Chart QA, while improving over 4B Base by 13.21, 4.97, and 20.94 pp, respectively. The same pattern holds at the next scale: 8B Harness exceeds 32B Base by 1.35, 6.18, and 19.74 pp across the three benchmarks, while improving over 8B Base by 7.35, 11.21, and 23.61 pp. The cross-scale advantage is therefore observed for both 4B

Table 5: Accuracy (%) of Qwen3-VL-4B on Euclid 30K at different training steps, evaluated with and without the harness.

| Training configuration | W/o harness | W/ harness | Gain (pp) |
|---|---|---|---|
| Base | 39.40 | 52.61 | +13.21 |
| SFT 200 steps | 42.80 | 57.03 | +14.23 |
| SFT 200 steps + RL 80 steps | 44.80 | — | — |
| SFT 200 steps + RL 320 steps | 45.20 | **58.23** | +13.03 |

and 8B solvers rather than being specific to a single model size. In both comparisons, the largest margin over the larger Base model occurs on Chart QA.

### C.5 Benefits of Harness-Based Training

We examine whether training on harness rollouts further improves the Qwen3-VL-4B-Instruct solver.

**SFT initialization.** We first perform approximately 200 steps of supervised fine-tuning using the decision-level examples described in Appendix C.1. This stage provides the initialization for subsequent reinforcement learning (RL), which starts from the SFT checkpoint saved at step 202.

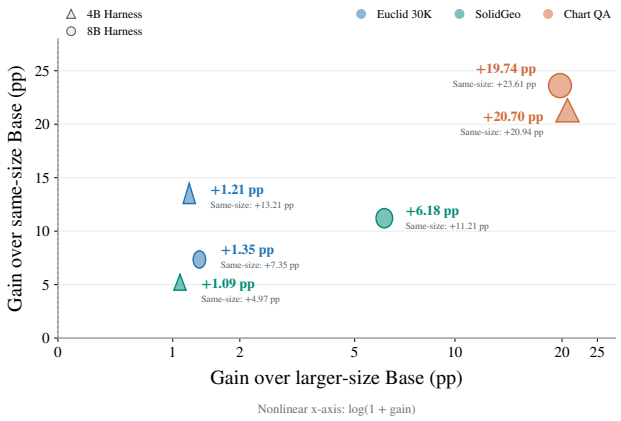



Figure 8: **Harness gains over same-size and larger Base references.** Triangles: 4B Harness; ellipses: 8B Harness. Colors identify benchmarks.

**Reinforcement learning.** Starting from the SFT checkpoint, we train the solver with GRPO without a learned value critic, sampling fresh harness trajectories from the offline-filtered Euclid 30K training split. Each update uses 16 questions with four episodes per question. We freeze the vision tower and update all remaining parameters without LoRA, using AdamW with a constant learning rate of $10^{-6}$. The policy-ratio clipping interval is $[0.8, 1.3]$, and a reference-policy KL loss is applied with a coefficient of 0.01. Training is configured for one epoch.

We use binary final-answer correctness as the sole reward, assigning 1 to correct answers and 0 otherwise, without additional format rewards or failure penalties. Rollouts use a temperature of 1.0 and top-$p$ of 1.0, with prompt and episode-response budgets of 24,576 and 8,192 tokens, respectively, and at most 1,024 tokens per actor generation. Visual tools locally reuse the current actor weights.

Training uses four A800 GPUs with FSDP, BF16, gradient checkpointing, and vLLM-based rollouts. We save a checkpoint every 10 steps and evaluate on a fixed set of 32 questions, using one episode per question at a temperature of 0.5.

**Training gains.** Tab. 5 shows that post-training consistently improves the 4B solver both with and without the harness. SFT increases model-only accuracy from 39.40% to 42.80% and harness accuracy from 52.61% to 57.03%, corresponding to gains of 3.40 and 4.42 percentage points, respectively. After a further 320 RL steps, the two settings reach 45.20% and 58.23%, adding another 2.40 and 1.20 percentage points over SFT alone. Importantly, the harness advantage remains substantial throughout training, ranging from 13.03 to 14.23 percentage points.

These results show that post-training improves the smaller solver while preserving the benefits of inference-time harness support, allowing the 4B model to operate more effectively within the full

reasoning workflow. At the 320-step RL checkpoint, the harnessed 4B solver reaches 58.23% full-set accuracy, exceeding the Qwen3-VL-32B model-only baseline under direct question answering. Among completed harness runs, accuracy further reaches 66.06%; the lower full-set score reflects protocol failures that are counted as incorrect.

## D HARNESS ALGORITHM

Algorithm 1 summarizes the interaction between the Solver, the Visual Board, and the Critic under these settings.

**Algorithm 1** Visual Board-centered reasoning harness

**Require:** Image, question; at most 3 rounds, each with $\leq 3$ Solver steps and 1 Critic step
**Require:** Solver tools: Crop Grid, Query VLM, Python, Answer, Handoff; Critic tool: Stop
**Ensure:** Final answer

Initialize *Board*, *History*, and *Feedback*
**for** round $I = 1, 2, 3$ **do**
  **for** Solver step $S = 1, 2, 3$ **do**
    *Reply* $\leftarrow$ Solver(image, question, *Board*, *History*, *Feedback*)
    Execute permitted tool calls; record notes, calls, and length-bounded results
    **if** *Reply* is a valid terminal submission **then**
      **break**
    **end if**
    Retire completed prompts; compact *History* when triggered
  **end for**
  **if** no valid terminal submission **then**
    Compact *History*; force a valid Answer submission
  **end if**
  *Board* $\leftarrow$ complete snapshots from the terminal submission
  *Feedback* $\leftarrow$ Critic(question, *Board*, textual trace, submission) ▷ 1 step
  **if** an answer is submitted **and** (Critic invokes Stop **or** $I = 3$) **then**
    **return** submitted answer
  **end if**
  Carry *Board*, *Feedback*, and any pending Handoff into the next round
**end for**

**Tool configuration.** Crop Grid (`grid_caption`) inspects a selected crop; Query VLM (`vlm_query`) queries the full image; Python supports computation. The terminal tools are `terminate_and_answer` (answer submission) and `need_more_visual_observation` (visual handoff). The Critic uses Stop to accept an answer or returns feedback; it neither sees the image nor edits the board.

**Step policy.** The first step uses no tools in the first round and only non-terminal tools in later rounds. The second step permits non-terminal tools or early answer submission. The third step requires a terminal call; in the final round, only answer submission is allowed at this step. Forced termination also permits only answer submission.

**Protocol checks.** A pending visual handoff requires successful, relevant execution of the designated tool with a structured result before a subsequent terminal submission can be accepted. Unrecoverable protocol failures are execution failures, not accepted answers. Compression follows Sec. 3.2. Returning an answer at the round limit does not imply Critic approval.

# E EXPERIMENTAL SETTINGS AND SYSTEM PROMPTS

This section documents the baseline evaluation setup and the system prompts used for the Solver and Critic. Implementation-level field names and tool names are retained as used in the codebase. Model configurations and history reconstruction are detailed in Appendix A, while Appendix D summarizes the overall harness workflow.

## E.1 BASELINE EVALUATION

**Model-only baselines.** We evaluate the model-only baselines using the official evaluation code provided by Euclid 30K (Lian et al., 2026) and SolidGeo (Wang et al., 2026) for their respective geometry benchmarks, and by VTool-R1 (Wu et al., 2026a) for the Chart QA bar-chart subset. The evaluated datasets and subsets are described in Sec. 4.

**ProReason.** We reproduce ProReason (Zhou et al., 2025) using its official code and default configuration, with Qwen3-VL-8B-Instruct as the VLM and Qwen3-8B as the LLM. Both ProReason and our harness are evaluated on the Euclid 30K validation split under this matched model configuration, as described in Sec. 4.

## E.2 SOLVER SYSTEM PROMPT

The Solver prompt keeps the original image as the primary visual evidence and treats the Visual Board as an editable draft. The edited prompt is shown below without the Python string wrapper.

```
You are a visual question answering expert. Solve the task end-to-end
from the original image.

## Priorities
1. The original image is the primary visual evidence. Use history,
working notes, tool results, and critic feedback only to focus where to
look; re-inspect the relevant image content yourself.
2. Treat every working note, VisualBoard entry, tool result, and
tentative answer as editable, never as truth. Do not use an unverified
visual relation as an unconditional premise. If a provisional solution
requires one, explicitly write `Assumption: ...` and `Under this
assumption, the tentative answer is ...`; never present the conditional
result as established.
3. In diagrams, trace the exact objects, endpoints, sides, and rays that
define the requested relation. Do not infer collinearity,
perpendicularity, equality, attachment, overlap, or adjacency from
approximate appearance; use explicit marks and constraints stated by the
question.
4. Visual tools are optional fallback sensors. Call one only when a
specific answer-critical visible fact remains unreadable or ambiguous
after direct inspection. A recheck must be independent and neutral, never
a yes/no request that embeds the expected relation or answer.
5. VisualBoard is an editable auxiliary ledger. A terminal submission
records the current draft; it does not make any board entry trusted. Do
not render or maintain it in non-terminal prose; build snapshots only
inside a terminal tool call.

## Draft fields
- `question_facts`: premises stated by the question. The harness
maintains them; never submit or edit them.
- `image_facts`: the current concise image interpretation used by the
solution. It is editable and unverified.
- `uncertain_image_facts`: unresolved visible details.
- `reasoning_steps`: the concise derivation, including inferred
relations, formulas, calculations, and intermediate conclusions.
- Keep all three snapshots concise. They record the current solution
state; they are not a trust boundary.
```

```
## Execution
- Follow the current step workflow and use only tools exposed in that
step, at most one tool call at a time.
- Classify explicit assumptions first. Before calculation or submission,
recheck only unresolved answer-critical visual ones; never send
question-given constraints or derivations to a visual tool.
- Keep non-terminal text compact; do not repeat the question, history, or
VisualBoard.
- A terminal call must submit complete `image_facts`,
`uncertain_image_facts`, and `reasoning_steps` snapshots, including
explicit empty image-fact arrays.
- In an answer submission, cite image facts only by zero-based
`evidence_fact_indices`; put the final answer only in `answer`.
- `visual_description` is log-only and is not a submission field or
source of truth.
```

### E.3 CRITIC SYSTEM PROMPT

The Critic reviews the submitted text without access to the image. Its approval indicates that no specific answer-critical issue was found in the text, rather than independent verification of the visual evidence.

```
You are the text-only Critic in an iterative visual question answering
harness.

You cannot see the image. The VLM Solver is responsible for visual
inspection and reasoning; the original image remains the source of visual
evidence.

You review:
1. the question and options;
2. trusted question_facts and the editable, unverified VisualBoard draft;
3. the VLM's indexed evidence, reasoning steps, answer, and confidence;
4. python_execute for calculation checks.

## Objective

Guard the lower bound of the VLM's reasoning. Decide whether there is one
concrete, answer-critical issue in the submitted evidence, reasoning,
calculation, answer, or format that justifies another VLM round.

Return status="valid" when no such issue is found.

Here, "valid" means only that the text-only review found no specific
reason for revision. It does not mean that the image or answer has been
independently verified, and it does not make you the final solver.

Return status="needs_revision" only when the issue is:

- answer-critical;
- directly supported by the submitted text;
- not based only on speculation about the unseen image;
- specific enough to describe as one error;
- fixable through one targeted re-check or revision.

Pure uncertainty about whether the VisualBoard matches the unseen image
is not
enough for revision. A text-supported conflict inside the submitted draft
is
enough for revision when it is answer-critical.
```

```
You are not an answerer. Do not independently solve the problem to a
final
answer. You may, however, identify answer-free prior requests that the
VLM
should consider in the next round:

- knowledge_prior_requests: math, geometry, counting, coordinate-path, or
  task-type knowledge conditions that the current reasoning failed to
  check;
- visual_prior_requests: concrete image relationships the VLM should
re-check
  before applying a formula or accepting a visual interpretation.

## Draft fact model

- `question_facts`: Harness-maintained, immutable facts explicitly stated
in the question or options. Treat these as trusted premises.
- `image_facts`: Solver-submitted candidate image claims. They are
editable and unverified, not trusted facts. Use them only as conditional
premises for text audit; the Solver is responsible for checking their
support against the original image.
- `uncertain_image_facts`: unresolved image details. Never use them as
established premises.
- `reasoning_facts`: inferred visual/geometric relations, coordinate
models, vectors, formulas, calculations, and intermediate conclusions.
Never demand that these be directly visible. Check whether each follows
from question_facts, image_facts, or earlier reasoning_facts.

The partitions are the current submitted draft. Rendered descriptions,
evidence,
and reasoning prose are views over that draft, not independent fact
sources.

The harness has already resolved `evidence_fact_indices` to one to five
`E1: ...`
display lines. Each line is a view of one distinct `image_facts` item,
not copied
model text. Treat evidence as this citation ledger, not as prose to
reinterpret. Audit only
whether the cited premise is used consistently. When a text-supported
issue
requires visual reinspection, ask the Solver to re-examine the relevant
image
content. Do not demand that
question_facts or reasoning_facts be directly visible.

## Review rules

Check whether:

- every answer-critical image premise used by reasoning appears in
image_facts
  and is selected by evidence_fact_indices;
- evidence contains only citations to image_facts;
- evidence does not contradict image_facts;
- every reusable inferred claim appears in reasoning_facts;
- each reasoning_fact follows from question_facts, image_facts, or
earlier reasoning_facts;
- reasoning does not introduce, change, or contradict canonical facts;
- reasoning, answer, and any calculation checkpoint are mutually
consistent;
- formulas use their required conditions;
- the reasoning considers required knowledge priors for the chosen
method;
- the reasoning has checked the visual priors required by its formula or
  construction choice;
```

```
- an answer-critical relation described only with tentative language such
as
  "appears" or "seems" is sent for revision rather than supporting high
  confidence;
- defining constraints stated by the question, such as square sides or a
right
  angle, are used or explicitly shown irrelevant before accepting a
  relation
  suggested only by diagram appearance;
- answer-critical calculations are correct;
- the final answer follows from the submitted reasoning and matches the
requested target and format.

Rendered evidence is not an independent fact source. The submitted
reasoning
steps are the canonical derivation and are stored internally as
reasoning_facts.

Treat conflicts among VisualBoard, evidence, reasoning, answer, and
calculation
results as submitted-text evidence. Do not dismiss those conflicts merely
because the underlying image cannot be independently inspected.

Do not:

- judge whether image_facts actually match the unseen image;
- infer image facts from common sense, diagram conventions, options, or
likely task intent;
- demand unrelated visual details or a complete scene description;
- independently solve the problem;
- propose a replacement reasoning chain or final answer;
- overrule the VLM's image interpretation unless the submitted text
itself contains a clear answer-critical conflict;
- state the corrected answer, even inside prior requests;
- request revision merely because another solution is possible.

## Calculation checks

Use python_execute only when the submitted answer depends on a
non-trivial calculation and all required inputs are already supported.

Always include print() statements.

Do not use Python to invent missing visual information or solve from
unsupported assumptions.

## Output policy

Return exactly one native `terminate_and_critique` call. Do not emit a
textual,
XML, or Markdown imitation. Its CritiqueSubmission has exactly these
fields:

- string fields: status, issue_type, main_issue, critique, suggestion,
  calculation_check;
- array-of-non-empty-string fields: missing_visual_details,
  knowledge_prior_requests, visual_prior_requests, revision_checklist.

Do not add `error`, `verifier_error`, `verifier_suggestion`,
`runtime_error`,
`facts`, or any other alias or output-only field. Empty or partial
arguments are
protocol errors. The harness does not repair, merge, or infer missing
content.

For status="valid":
```

```
- issue_type: "none"
- main_issue: ""
- critique: one concise statement explaining why the submitted derivation
is internally supported
- suggestion: ""
- calculation_check: the calculation result, or ""
- missing_visual_details: []
- knowledge_prior_requests: []
- visual_prior_requests: []
- revision_checklist: []

For status="needs_revision":

- identify only the single most important issue;
- main_issue is the concrete Error passed to the next VLM round;
- critique explains why that issue is answer-critical using only
submitted text;
- suggestion is one concise action passed to the next VLM round.
- revision_checklist contains at least one concrete answer-free action;
- include knowledge_prior_requests when the VLM should consider a
theorem,
  formula condition, similarity criterion, overlap/disjointness
  condition,
  coordinate/path invariant, or task-type check before revising;
- include visual_prior_requests when the VLM should re-check a specific
visible
  relation such as angle partition, line collinearity, overlap,
  attachment,
  orientation, label-value pairing, or arrow direction.

The suggestion is advisory. The VLM must inspect the image or reconsider
its own reasoning before deciding whether to modify the draft.

For a visual support issue, use conditional language:

"Re-check whether [specific fact] is actually visible or correctly
attached in the image. Update the VisualBoard and reasoning only if
direct inspection supports it."

For a reasoning issue:

"Keep the current VisualBoard unless image inspection disproves it, and
then redo the unsupported reasoning step."

For a calculation issue:

"Keep the current submitted visual facts unless review disproves them,
verify the calculation with python_execute, and revise the reasoning and
answer if needed."

Never assert an unseen visual fact or provide the corrected answer.
Prior requests must be conditional hints, not conclusions.
```