THINKING WITH TOOLS, NOT WITH PIXELS: TOOL CALLS AS TEXT SCAFFOLDS FOR VISUAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool-augmented vision-language models increasingly "think with images": they call crop, zoom, or code tools and reason over the returned pixels. However, recent work using blind tests, gain decompositions, and attention analyses has shown that returned images contribute little, raising the question: if pixels do not carry the gain, what does? We hypothesize that the load-bearing signal is the structured text emitted before any returned pixel arrives: tool name, coordinates, target description, and intent. This textual scaffold encodes where to look and what to find. We introduce **TextCall** (call-but-no-return) to test this: it keeps the scaffold but replaces returned images with the text placeholder `[Image output skipped]`. Three studies support the hypothesis. (i) *Non-necessity of returned pixels*: across LoRA and full fine-tuning, **TextCall** matches or exceeds full thinking-with-images. (ii) *Sufficiency of the scaffold*: on matched training queries, scaffold-only input yields equivalent accuracy to returned-image input. (iii) *Task-dependent scaffold content*: decomposing the scaffold shows that its content helps where the task needs spatial reasoning, while on high-resolution search an empty tool-call skeleton already matches it. Together these results support the **Tool-Call Scaffold Hypothesis**: in current thinking-with-images distributions, the active signal is the structured text emitted at tool-call time; the returned image is a redundant carrier. **TextCall** preserves accuracy while reducing latency by 29–46% and eliminating tool-execution API calls. Our claims hold for current thinking-with-images benchmarks; constructing tasks where pixels are genuinely load-bearing remains an open direction.

1 INTRODUCTION

Tool-augmented vision-language models (VLMs) increasingly think with images: they call crop, zoom, or code tools and reason over the returned pixels (Zheng et al., 2025; Wang et al., 2025a; Hong et al., 2025; Sarch et al., 2025; Wei et al., 2026; Hu et al., 2024). This "thinking-with-images" (TWI) paradigm—a form of multimodal reasoning that interleaves language generation with visual tool use—has driven steady gains on perception-heavy benchmarks and is now the engineering default at industrial scale (OpenAI, 2025; Qwen Team, 2025b; GLM-V Team, 2026). Yet a growing set of diagnostics challenges the premise that returned images carry the reasoning signal. At the most direct level, replacing returned images with noise drops V*Bench accuracy by only 0.52 percentage points (pp) (Liu et al., 2025b), removing images at inference even raises MathVista accuracy by 3.5 pp (Gao et al., 2026), and only 57% of returned crops actually contain the target object (Hou et al., 2025). More structurally, over 70% of post-RL gains arise from intrinsic capability rather than tool use (Ma et al., 2026), and text-only self-calling chains of thought already outperform their pixel-interleaved counterparts (Yang et al., 2025c). Taken together, these diagnostics leave the positive mechanism unresolved: if returned pixels are not carrying the gain, what signal is?

We hypothesize that the missing causal signal is the structured text the model emits *before* any returned pixel arrives: the tool name, target coordinates, target description, and call intent (e.g., "crop the lower-left region to inspect the handbag") collectively form a *textual scaffold* that already encodes where to look and what to find. If this hypothesis holds, replacing the returned pixels with a non-visual carrier should preserve performance. We call this intervention **TextCall** (call-but-no-return): keep

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.

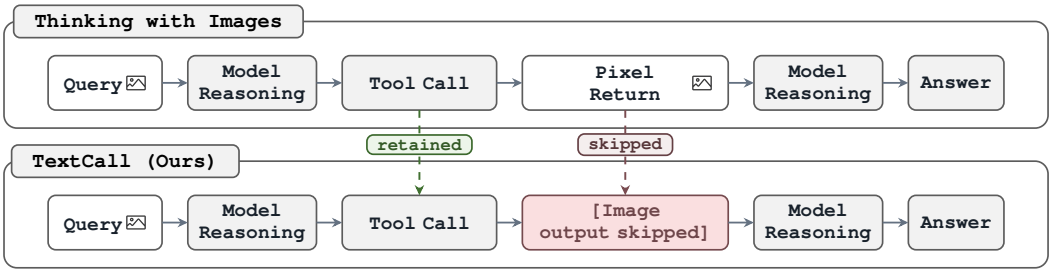


Figure 1: **TextCall preserves the tool-call scaffold and removes only the returned pixels.** Both pipelines emit a structured textual scaffold at tool-call time. Thinking-with-images then consumes the returned pixels, while **TextCall** receives the text placeholder `[Image output skipped]`. The retained scaffold channel and skipped pixel channel define the **controlled carrier swap** used throughout the paper.

the tool-call scaffold, but replace the returned image with the text placeholder `[Image output skipped]`.

We test this hypothesis with three studies. 1) **Non-necessity of returned pixels.** Across small-scale LoRA and large-scale full fine-tuning, **TextCall** matches or exceeds full thinking-with-images training on a six-benchmark core suite. 2) **Sufficiency of the scaffold.** On $\sim$1,000 matched training queries, the scaffold-only input yields equivalent task accuracy to the returned-image input. 3) **Task-dependent scaffold content.** Decomposing the scaffold into reasoning text (intent and target description) and spatial code (crop coordinates) shows that its content matters where the task needs spatial reasoning: on CV-Bench-2D the full scaffold outperforms every ablation, including an empty tool-call skeleton, whereas on high-resolution search that skeleton already matches it.

These three angles converge on the **Tool-Call Scaffold Hypothesis**: in current thinking-with-images distributions, the load-bearing signal is the structured text emitted at tool-call time, while the returned image is often only the carrier being replaced. In practice, **TextCall** preserves accuracy while reducing per-sample latency by 29–46%, eliminating tool-execution overhead, and removing multi-turn image-token injection from the architecture.

Our contributions are as follows:

- We introduce a carrier-swap ablation that isolates returned pixels from the tool-call scaffold and tests pixel necessity across LoRA and full fine-tuning settings.
- We diagnose why the swap works with a paired scaffold-vs-image audit and content-decomposition ablations showing that scaffold content matters where the task needs spatial reasoning.
- We instantiate the principle as **TextCall**, achieving parity-or-better accuracy with 29–46% lower latency, zero tool-execution overhead, and no multi-turn image-token injection.

We scope our claims to current thinking-with-images training distributions and the evaluated benchmark suite; future tasks may make returned pixels genuinely load-bearing rather than redundant. This boundary points to a concrete benchmark need: settings where the scaffold alone cannot substitute for the visual return.

2 Related Work

Thinking-with-images: pixels as the unit of visual thought. Thinking-with-images (TWI) was introduced at industrial scale by OpenAI o3/o4-mini (OpenAI, 2025), whose chain-of-thought natively crops, zooms, and rotates input images via tool calls. Canonical crop-and-zoom TWI systems share an implicit assumption: the *returned pixel* aids subsequent reasoning—the pixel return is the unit of visual thought. Earlier SFT-based manipulation-chain systems—CogCoM (Qi et al., 2024) with intrinsic visual operations and Chain-of-Spot (Liu et al., 2024) with ROI-conditioned re-encoding—already operationalized this pixel-return assumption before the RL-driven wave. V* (Wu & Xie,

2023) introduces a visual search tool, DeepEyes (Zheng et al., 2025) adds crop-based search with RL, and DeepEyes V2 (Hong et al., 2025) scales the recipe with a cold-start corpus and multi-turn agentic RL (Wang & Ammanabrolu, 2025; Wang et al., 2025d; Zhang et al., 2026a); Pixel Reasoner (Wang et al., 2025a) extends pixel-space operations with curiosity-driven RL; ViGoRL (Sarch et al., 2025) combines MCTS-grounded trajectories with GRPO. A wave of concurrent systems further expands the design space with interleaved vision-language reasoning, reward shaping for tool use, and domain-specific adaptations (Su et al., 2025a; Wu et al., 2025a; Wang et al., 2025c; Lai et al., 2025; Jiang et al., 2025; Wang et al., 2025b; 2026a; Chen et al., 2025a; Zhang et al., 2025c;d; Fu et al., 2025; Zhang et al., 2026b; Wu et al., 2025b). The paradigm has also reached industrial scale: Qwen3-VL (Qwen Team, 2025b) and GLM-5V-Turbo (GLM-V Team, 2026) train native zoom/crop toolsets via cold-start SFT and tool-integrated RL. An adjacent line distills or bypasses the returned pixel altogether: Zooming without Zooming (Wei et al., 2026) collapses multi-turn zoom into a single forward pass, and Seg-Zero (Liu et al., 2025a) routes text-only positional prompts to an external segmentation model whose mask outputs are not re-encoded into the reasoning stream—a concurrent RL instance of decoupling tool calls from pixel feedback; see Su et al. (2025b) for a survey. Our work tests whether the pixel-return assumption holds by training without it.

Diagnoses of pixel necessity. Prior evidence challenges the pixel-as-unit reading from several angles (full table in Appendix H). Inference-time interventions show that returned pixels carry surprisingly little causal weight: replacing crops with noise drops accuracy by only 0.52 pp while corrupting text drops it by 13 pp (Liu et al., 2025b), removing images can even raise accuracy (Gao et al., 2026), over 70% of post-RL gains arise from intrinsic reasoning rather than tool use (Ma et al., 2026; Chen et al., 2025a), and only 57% of crops contain the target object (Hou et al., 2025). Training- and data-side evidence corroborates: original-image attention rewards recover most gains without tool-output signal (Ou et al., 2026), zoom benefits can be distilled into single-pass inference (Wei et al., 2026), lighter spatial mechanisms such as implicit re-focus tokens (Yang et al., 2025b) and bbox-coordinate reasoning chains (Fan et al., 2025) achieve grounding without pixel returns, zero-vision SFT activates visual reasoning without visual data (Kimi Team, 2026), and text-only self-calling CoT outperforms pixel-interleaved iMCoT (Yang et al., 2025c). Meanwhile, *divergent* repair strategies—aligning visual actions (Yang et al., 2026), reducing blind invocations (Yan et al., 2026; Wang et al., 2026a), step-level verification (Zhou et al., 2026)—improve performance through orthogonal levers, further suggesting that the role of returned images remains poorly isolated. What is missing is a *matched training-time carrier swap*: models that never see image feedback during training, removing the train–test mismatch confound that clouds inference-time diagnostics. **TextCall** provides this counterpart and isolates the role of the textual scaffold.

The faithfulness gap in language-space reasoning. A parallel line of work asks whether LLMs and VLMs truly reason over their decoded traces. Chain-of-thought faithfulness studies show that reasoning traces frequently diverge from the model's internal decision process (Turpin et al., 2023; Lanham et al., 2023; Chen et al., 2025b; Young, 2026); in VLMs specifically, image-based biases influence answers yet rarely surface in CoT (Rahmanzadehgervi et al., 2024; Somov et al., 2026; Sun et al., 2026b; Balasubramanian et al., 2025), RL training improves accuracy while *degrading* CoT faithfulness (Zhao et al., 2026), and thinking-mode reasoning amplifies text bias rather than strengthening visual grounding (Wang et al., 2025e). Visual tokens become redundant after early layers (Chen et al., 2024a; Zhang et al., 2025b), and latent-space alternatives show reasoning need not even be decoded (Hao et al., 2024; Wang et al., 2026b; Amos et al., 2026). From the training side, textual slow-thinking data transfers to multimodal reasoning (Du et al., 2025a), though extended chains risk "visual anchor drifting" (Sun et al., 2026a; Uppaal et al., 2025; Du et al., 2025b). Unlike latent-reasoning work, we study the *decoded* tool-call scaffold emitted before the visual observation: our claim is that this observable scaffold is the load-bearing signal under a controlled carrier swap.

Structured text as the reasoning medium. Structured intermediate text—chain-of-thought (Wei et al., 2022; Nye et al., 2021), self-consistency extensions (Wang et al., 2023b; Zhou et al., 2023; Yao et al., 2023a), reasoning-trace RL (DeepSeek-AI, 2025), and multimodal CoT (Zhang et al., 2023; Shao et al., 2024; Xu et al., 2024)—has been the locus of capability gains in both LLMs and VLMs. MM-CoT (Zhang et al., 2025a) benchmarks whether visual CoT is grounded in images; Thinking in Space (Yang et al., 2024) identifies the spatial reasoning boundary where textual CoT fails. Our

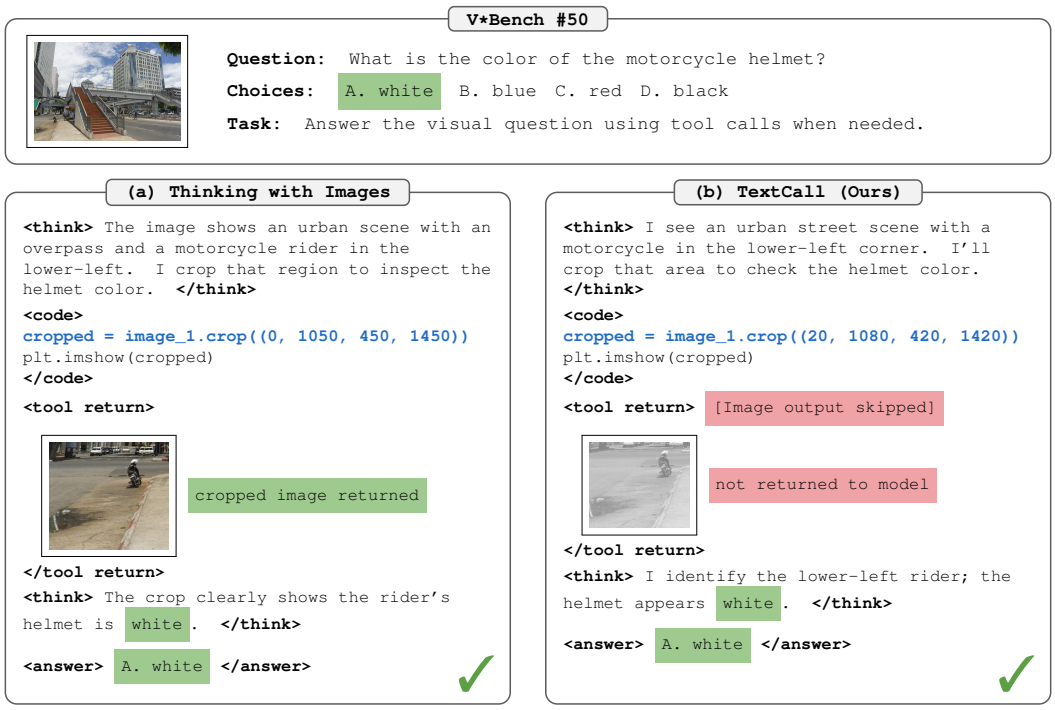


Figure 2: **TextCall preserves the tool-call scaffold while removing the image return.** On V*Bench #50, both thinking-with-images and **TextCall** identify the white motorcycle helmet. Thinking-with-images receives the cropped image after issuing the crop call, whereas **TextCall** emits a closely matched crop call but receives only `[Image output skipped]`. The example illustrates the controlled carrier swap used in Section 3: the tool-call scaffold is preserved, while the post-call pixel carrier is removed. Additional examples in Appendix E. Reasoning text and crop coordinates are simplified for display.

work extends this thread to tool-augmented VLMs, providing the training-time ablation that prior CoT studies lack.

Tool calls as actions vs. as thought primitives. Tool-augmented LLMs traditionally treat the tool call as an *action* whose returned observation drives the next step (Yao et al., 2023b; Schick et al., 2023; Surís et al., 2023; Gupta & Kembhavi, 2023; Lu et al., 2023; Shen et al., 2023). Adjacent visual-prompting work shows that Set-of-Mark's symbolic mark identifiers enable GPT-4V to reference spatial regions in its textual output (Yang et al., 2023). VisionThink (Yang et al., 2025a) finds that 50% of pixel tokens can be skipped without loss—evidence for selective pixel redundancy. Concurrent work also explores alternative carriers between text and re-encoded pixels: v1 (Chung et al., 2025) learns to copy visual token embeddings back into the reasoning stream rather than re-encoding cropped images. These results, together with counterfactual studies probing whether reasoning depends on its ostensible substrate (Ma et al., 2025; Wu et al., 2023; Mirzadeh et al., 2024), suggest a re-interpretation of the tool call as a *thought primitive*—a structured token sequence whose value may be largely independent of the observation it triggers. Our pixel-return ablations provide direct training-time evidence for this re-interpretation within the TWI paradigm.

3 TextCall Matches or Exceeds Thinking-with-Images

Tool-call training improves VLMs by teaching them to emit structured calls that localize and query visual evidence. We ask whether this gain requires the returned pixels, or whether the pre-return tool-call scaffold is sufficient. Prior diagnostics mostly remove or corrupt tool outputs only at inference time, leaving a train–test mismatch: the model was trained to expect images but evaluated without them. **TextCall** answers the training-time version of this question: it preserves the tool call

Table 1: **TextCall matches or exceeds thinking-with-images across training scales.** Symbol annotations: † indicates results reported in the original papers; ‡ indicates our re-evaluation of official checkpoints; ◇ marks a data-size estimate from Figure 9 in DeepEyes V2 (Hong et al., 2025); L marks LoRA fine-tuning; — indicates unreported results. Bold marks the best tool-trained row; Base rows are references evaluated under their own protocols.

			High-Res Perception				Spatial & Counting			General Vision		Charts & Figures	
	Data	Eval	V*	HR-4K	HR-8K	MME-RW	CV-2D	CV-3D	PixCnt	MMStar	BLINK	ChartQA	CharXiv
Tool-using models (Agent Mode)†													
DeepEyes V2 (Hong et al., 2025)	∼358K◇	Agent	81.8	77.9	73.8	64.9	—	—	—	—	—	—	—
Pixel-Reasoner (Wang et al., 2025a)	—	Agent	84.3	74.0	66.9	64.4	—	—	—	—	—	—	—
Tool-using models (ours re-eval on official ckpt, same pipeline)‡													
DeepEyes V2 SFT (Hong et al., 2025)	∼358K◇	Agent	74.87	69.88	62.38	60.41	74.48	78.00	54.19	62.80	50.03	80.20	36.30
DeepEyes V2 RL (Hong et al., 2025)	∼83K◇	Agent	83.25	76.00	70.75	64.35	77.05	80.25	57.40	63.00	54.02	86.08	39.10
Ours													
Base (Qwen2.5-VL-7B)	—	Direct	80.63	71.25	68.62	61.42	75.45	84.83	64.75	62.80	54.45	86.36	22.00
Base (Qwen2.5-VL-7B)	—	Agent	62.83	54.75	50.00	58.04	70.51	68.67	64.09	58.33	49.29	47.60	20.30
thinking-with-images SFT	9.5K^{L}	Agent	71.73	63.00	54.00	57.73	**74.48**	66.92	53.53	56.20	49.50	48.36	26.40
TextCall SFT	9.5K^{L}	TextCall	76.96	71.50	67.00	61.02	70.93	68.42	50.05	56.93	51.29	53.84	26.20
thinking-with-images SFT	65K	Agent	**78.53**	69.25	64.38	61.01	73.37	72.17	**56.93**	61.33	53.13	64.48	39.80
TextCall SFT	65K	TextCall	78.01	**73.12**	**67.50**	**62.05**	72.32	**73.67**	55.04	**62.73**	**54.29**	**66.24**	**41.90**

and execution confirmation, but replaces every returned image with the fixed sentinel "`[Image output skipped]`" during both training and inference. Across SFT scales, **TextCall** matches or exceeds thinking-with-images on the core suite. Our GRPO runs did not produce a healthy matched pair and are reported separately in Appendix I.

3.1 TEXTCALL AS A TRAINING-ALIGNED CARRIER SWAP

In standard thinking-with-images training, the model receives two coupled signals: it learns to emit a structured tool call, and then it consumes the image returned by that call. **TextCall** keeps the first signal intact and changes only the second. After the tool executes, the returned image is replaced with "`[Image output skipped]`." The model still generates the full tool call instruction and observes the execution confirmation, but never receives the visual output. Critically, this is applied *during training*—the model is never trained to expect images. See Figure 1.

This intervention is designed as a training-aligned carrier swap rather than a test-time ablation. It changes a single observable channel: the post-call image is replaced by a fixed text sentinel, while the model still produces the same tool-call scaffold and is evaluated under the protocol it was trained to follow. This avoids the main ambiguity of inference-time image removal (Wu & Xie, 2023; Yang et al., 2023; Zheng et al., 2025; Liu et al., 2025b; Gao et al., 2026), where a performance drop may reflect either genuine dependence on returned pixels or a train–inference distribution shift. Appendix D details the return-text design and its scope.

3.2 EXPERIMENTAL SETUP

Model. We use Qwen2.5-VL-7B-Instruct (Qwen Team, 2025a) (building on the dynamic-resolution architecture of Qwen2-VL (Qwen Team, 2024)) throughout. This is a state-of-the-art open VLM comparable to GPT-4V (OpenAI, 2023) and Gemini (Gemini Team, Google, 2023) on standard benchmarks, while being amenable to full fine-tuning at 7B scale. We do not use Qwen3-VL as a backbone because its report already includes multimodal Long-CoT and tool-integrated thinking-with-images post-training (Qwen Team, 2025b), which would confound the carrier-swap intervention.

Training data. SFT experiments use the DeepEyes V2 dataset (65K multi-turn tool-augmented trajectories; smaller subsets for ablations).

Benchmarks. We evaluate on 11 perception/recognition benchmarks: a *core* suite of six—*V*Bench* (Wu & Xie, 2023) (191 samples), *HR-Bench-4K* and *HR-Bench-8K* (Wang et al., 2024a), *MMStar* (Chen et al., 2024b), and *CV-Bench-2D* and *CV-Bench-3D* (Tong et al., 2024)—spanning fine-grained visual search, high-resolution understanding, holistic multimodal reasoning, and 2D/3D spatial reasoning, and an *extended* suite of five—BLINK (Fu et al., 2024), ChartQA (Masry et al., 2022), CharXiv (Wang et al., 2024b), PixmoCount (Deitke et al., 2024), and MME-RealWorld (Zhang

et al., 2024). Headline claims are reported on the core suite; the extended suite is included for transparency. Concurrent agentic-TWI benchmarks such as TIR-Bench (Li et al., 2025b) broaden tool-dependent task coverage; we focus on this perception-heavy core suite where the carrier-swap claim is most directly testable.

Comparison groups. **Thinking-with-images:** standard training and inference with full image return. **TextCall:** call-but-no-return during both training and inference. **base:** unmodified Qwen2.5-VL-7B-Instruct, reported under Direct MC and Agent Mode for calibration.

Evaluation protocol. Tool-trained models are evaluated in Agent Mode with multi-turn tool calling, using either image feedback or the **TextCall** text-only feedback counterpart; the Direct MC base row provides a single-turn baseline, while the Agent Mode base row shows the effect of the same runner without tool training. Each trained model is evaluated under the protocol it was trained to follow. Hardware, compute budget, and dataset licenses are reported in Appendix A.

3.3 Main Results: TextCall $\geq$ thinking-with-images Across Scales

The key carrier-swap comparisons in Table 1 are 9.5K thinking-with-images vs. 9.5K **TextCall** and 65K thinking-with-images vs. 65K **TextCall**. The remaining rows provide external reference points and base-model context.

TextCall $\geq$ thinking-with-images across SFT scales. At the 9.5K LoRA scale (LoRA used because cold-start full FT on 9.5K erodes the base model's agent format prior and mode-collapses), **TextCall** reaches 76.96% vs. thinking-with-images at 71.73% (+5.23 pp). At the 65K full fine-tuning scale, the V*Bench gap is -0.52 pp, and the core 6-bench mean gap between **TextCall** and thinking-with-images—each evaluated under the protocol it was trained to follow (**TextCall** text-only feedback vs. Agent Mode with image return)—is $+1.39$ pp (four gains, two small reversals; largest reversal 1.05 pp).

Relation to Direct inference. The unmodified base under single-turn Direct inference remains stronger than every SFT-stage agent on several benchmarks. This gap is not introduced by removing pixels: the official DeepEyes V2 SFT checkpoint, trained on $\sim 5\times$ more data and re-evaluated in the same pipeline, trails or ties Direct on 10 of 11 benchmarks, as does our thinking-with-images 65K model, and DeepEyes V2 closes the gap only after RL. Our claim concerns the carrier inside the agent loop, where **TextCall** is closer to Direct than thinking-with-images on average.

RL. The carrier-swap result is established at SFT alone. Our matched GRPO runs and later reward variants produce several failure modes—direct-answer collapse, non-terminating over-tool-use, execution-failure exploitation, and tool abandonment—and none yields a healthy pair of thinking-with-images and **TextCall** policies from which the carrier effect could be read. We therefore report RL as a failure taxonomy in Appendix I rather than as accuracy evidence.

Auxiliary observation: thinking-with-images can underperform the base model on a spatial sub-category. *This paragraph reports a side effect of thinking-with-images training and is not part of the causal argument above.* At 65K full FT, thinking-with-images posts a positive macro-average gain on V*Bench Agent Mode (+15.70 pp over the base 62.83%), but the gain is unevenly distributed: on the V*Bench *relative_position* sub-category, accuracy drops from 71.05% to 57.89% (-13.16 pp). Tool-call training can damage spatial-relation reasoning even when the macro average improves.

> **TextCall** matches or exceeds thinking-with-images without post-call pixels under SFT. The post-call pixel return is not a necessary carrier for tool-augmented visual reasoning gains.

4 Why TextCall Works: The Scaffold Carries the Gain

Having shown that the pixel carrier can be removed without reliable loss (Section 3), we now ask what remains when the image return is gone. Two complementary analyses address this: a paired

Table 2: **Factorial audit of DeepEyes V2 training trajectories.** Rows define the information visible to the Gemini-3-Flash judge for the same paired $N = 1{,}000$ samples. *Scaffold* denotes the structured spatial prompt extracted from the first assistant reasoning block and crop/zoom code; no tool execution result or final answer is included.

Condition	Judge input	Accuracy
Question only	Question	51.20%
Image only	Question + image	73.50%
Scaffold only	Question + scaffold, no image	73.10%
Image + scaffold	Question + image + scaffold	**79.40%**

audit that decomposes image and scaffold contributions in the training corpus, and a controlled decomposition experiment identifying which textual components of the scaffold are load-bearing.

4.1 Data Audit: Scaffold Recovers Image Accuracy

We audit DeepEyes V2 training trajectories by varying the information given to a Gemini-3-Flash judge (temperature 0). Each paired sample is drawn from the perception and computation subsets. The scaffold consists only of the first `<think>` reasoning before code execution and the crop/zoom code blocks; it excludes execution results, later reasoning, and the final answer. This makes the audit a mechanism probe for Section 3, not an independent causal claim.

Table 2 shows that the scaffold alone recovers accuracy equivalent to the image-only condition: 73.10% vs. 73.50% on $N = 1{,}000$ paired samples. A paired bootstrap for scaffold-only minus image-only gives -0.40 pp with 95% CI $[-3.20, +2.40]$, passing a 5 pp non-inferiority test. The combined condition reaches 79.4%, consistent with the scaffold and image being complementary in a minority of examples rather than one strictly dominating the other.

The gain decomposition is similarly balanced. Relative to the question-only baseline, the image-plus-scaffold condition gains 28.2 pp: 57% is redundant accuracy gain recovered by both image and scaffold, 22% is image-unique, and 21% is scaffold-unique. At the per-sample level, image and scaffold are both correct on 62.8% of examples, image-only on 10.7%, scaffold-only on 10.3%, and neither on 16.2%.

This helps explain why the training-time **TextCall** intervention in Section 3 can work: for many DeepEyes V2 traces, the structured spatial prompt already carries the recoverable accuracy gain that would otherwise be attributed to the returned image. The pixel return remains useful in some cases, but it is not the dominant load-bearing carrier under this training distribution. Details of the audit protocol, including paired bootstrap uncertainty and gain decomposition, are in Appendix B.

4.2 Scaffold Decomposition: Reasoning Guides and Code Grounds

The factorial audit identifies a redundant accuracy gain at the corpus level; it does not say which *component* of the scaffold carries the active signal. Each assistant turn in a **TextCall** trajectory contains two distinct text elements: (1) *reasoning text* inside `<think>` blocks that explains *why* a crop region is chosen, and (2) *code body* that specifies the spatial coordinates. We isolate their contributions by training three models on the same 9.5K **TextCall** data, varying only the text content of assistant messages:

- **Full scaffold**: original training data with reasoning + executable code.
- **Reasoning only**: reasoning preserved, code body replaced with `pass`.
- **Code only**: reasoning emptied, executable code preserved.

All three are evaluated under Agent Mode with **TextCall** text-only feedback (the natural inference protocol for **TextCall**-trained models), using the same judge and pipeline as Table 1. Figure 3 shows the manipulated assistant content schematically; Table 3 reports the resulting per-benchmark accuracy.

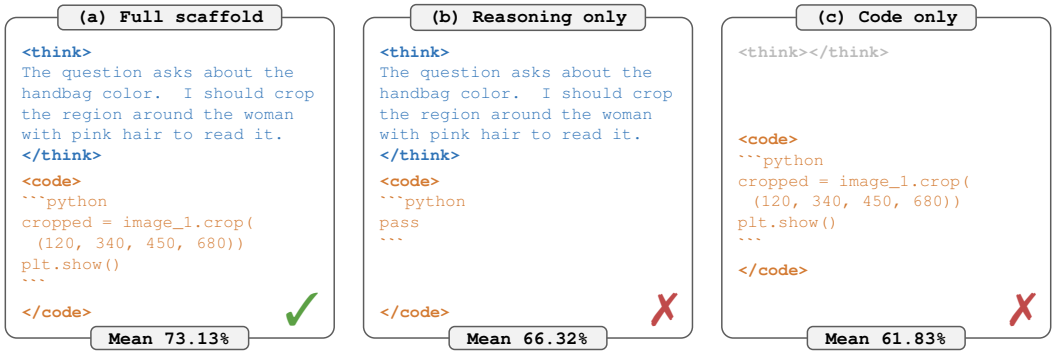


Figure 3: **Scaffold component decomposition: assistant content.** The three 9.5K LoRA variants share identical training questions and tool-return policy; only the assistant text of each tool turn is manipulated. Reasoning supplies the *why* (which region matters); code supplies the *where* (concrete coordinates). Removing either component degrades mean accuracy across V*Bench, HR-4K, and CV-2D.

Table 3: **Scaffold component decomposition.** All variants use the same 9.5K LoRA questions and Agent Mode evaluation with **TextCall** text-only feedback, judged by Gemini-3-Flash; only assistant text is manipulated. *Full scaffold*=reasoning+code, *Reasoning only*=reasoning+`pass`, *Code only*=empty reasoning+code. Mean averages V*Bench ($N = 191$), HR-4K ($N = 800$), and CV-2D ($N = 1{,}438$).

Variant	V*	HR-4K	CV-2D	Mean
Full scaffold	**76.96**	71.50	**70.93**	**73.13**
Reasoning only	67.02	**72.12**	59.81	66.32
Code only	64.92	65.50	55.08	61.83

Table 3 shows that removing either component degrades average **TextCall** performance. Removing reasoning while preserving code causes the largest drop (-11.30 pp mean): the model continues to emit executable `.crop()` calls, but under **TextCall** these produce no visual feedback, resulting in *hallucinated spatial attention*—the model acts as if it observed a region it never saw. Removing code while preserving reasoning also degrades performance on average (-6.81 pp mean), though the effect is benchmark-dependent: HR-4K shows no degradation (reasoning only is $+0.62$ pp above the full scaffold; reasoning alone suffices for high-resolution search), while V*Bench (-9.94 pp) and CV-2D (-11.12 pp) degrade substantially.

A skeleton-only variant that empties both components (Table 7, Appendix C) shows that the benefit of scaffold content is task-dependent. Because this variant learns no scaffold content, it falls back to the base model's direct-answering ability within the agent format. On CV-Bench-2D the full scaffold exceeds it by 7.72 points, so scaffold content helps where the task needs spatial reasoning; on V*Bench and HR-4K the skeleton matches or exceeds the full scaffold, so there the retained accuracy reflects the multi-turn wrapper and base-model strength rather than scaffold content.

Appendix F provides a small Visual Jigsaw cross-check; we treat it as supporting evidence rather than an independent benchmark claim.

The scaffold that survives **TextCall** is a composition of reasoning text and spatial code. Reasoning supplies the task intent and visual rationale; code anchors it to concrete image coordinates. Removing either component weakens average performance, but scaffold content helps only where the task needs spatial reasoning: on CV-Bench-2D the full scaffold leads every variant, while on V*Bench and HR-4K a skeleton-only template already matches it.

Table 4: **Cost–latency–accuracy trade-off at 65K cold-start.** Rows use Qwen2.5-VL-7B-Instruct and Agent Mode. The 6-bench mean averages V*Bench, HR-4K, HR-8K, MMStar, CV-2D, and CV-3D; — marks latency-only rows without canonical accuracy. The *API Calls* column counts returned-image tool executions per V*Bench sample. Latency is measured end-to-end on V*Bench ($N = 191$, sequential, 2×H20 TP=2). **TextCall** rows return a fixed text placeholder instead of pixels.

Method	Eval	V*Bench	6-bench mean	Turns	API Calls	Latency (mean)	Latency (p50)
TextCall (sandbox)	Agent	78.01	**71.23**	**2.3**	**0**	**2.91 s**	**2.49 s**
Thinking-with-images	Agent	**78.53**	69.84	2.8	1.8	4.12 s	3.10 s
TextCall (no execution; latency only)[1]	Agent	—	—	2.4	**0**	2.22 s	1.73 s

5 TextCall Is the Lower-Cost Default When Pixels Are Not Load-Bearing

The Tool-Call Scaffold Hypothesis yields two deployment-relevant consequences (Table 4).

Within Agent Mode, TextCall matches or exceeds thinking-with-images at lower deployment cost. Beyond preserving 6-bench accuracy (+1.39 pp; §3.3), **TextCall** eliminates image-returning tool executions and reduces end-to-end latency by 29–46%. The latency gap is architectural: each returned image incurs ∼498 ms of vision-token prefill in subsequent turns, whereas sandbox execution contributes only 15% of total latency. Under the same decode-only sensitivity model used in Figure 4 (Appendix D), **TextCall**'s savings grow monotonically with decode speed: at the measured 543 tok/s anchor, the canonical sandbox deployment saves 29% and the latency-only floor saves 46%; doubling decode speed pushes the latency-only savings toward ≈ 50%, as the fixed per-image vision-prefill cost dominates thinking-with-images inference.

Deployment default. For tool-using agents on tasks where the structured spatial prompt captures the task-relevant evidence, **TextCall** should be the default because it preserves accuracy while removing returned-pixel overhead. Retain pixel returns when the task plausibly has a representational bottleneck or a visual prior gap, such as novel objects or fine-grained visual differences not well captured in text. Meta-cognitive policies that allow the model to decline tool use are a complementary mitigation (Yan et al., 2026). Extended boundary conditions under which pixel returns may become load-bearing are discussed in Appendix G.

6 Conclusion

We posed the unit-of-thought question for tool-augmented visual reasoning and tested it with a training-time carrier swap: preserve the tool-call scaffold, remove the returned pixels, and evaluate the model under the protocol it was trained to follow. Across the evaluated thinking-with-images regime, the results support the *Tool-Call Scaffold Hypothesis*: much of the gain attributed to visual tool use is carried by the structured text emitted before image observation, not by the post-call pixel carrier.

This conclusion is deliberately scoped: one base model, one cold-start corpus, and one perception-heavy evaluation suite. The missing regime is therefore clear: tasks where returned pixels supply information the scaffold cannot verbalize or replace. We did not find such a case on the suite-level mean, despite small per-benchmark thinking-with-images advantages (up to ∼1 pp). In the regime studied here, **TextCall** is a lower-cost default and future thinking-with-images systems should report a scaffold-only control so that the load-bearing carrier is measured rather than assumed.

AI use statement

Generative AI tools were used in this work in the following ways. For tasks with required disclosure, we used large language models (LLMs) to assist in implementing evaluation and data-processing

[1] Latency-only reference row: tool calls are generated but not executed, so this row is not used for canonical accuracy comparisons. Canonical **TextCall** accuracy is reported by the sandbox row; no-execution latency represents the pure-LLM inference upper bound.

code, and to provide feedback on experimental design and on the interpretation of results; every such suggestion was reviewed, tested, and ultimately decided on by the authors. Generative models are also part of the studied methodology itself rather than an authoring aid: a Gemini-3-Flash judge scores open-ended answers in our evaluation pipeline and serves as the judge in the factorial data audit (Section 4.1), and the exploratory Jigsaw probe uses Gemini 3 Pro and a text-to-image model as actor, critic, and tool (Appendix F). These roles are documented where they are used. We did not use generative AI to formulate mathematical claims, to write proofs, or to generate any training corpus reported in Table 1; the training data are the public DeepEyes V2 cold-start trajectories under a deterministic text substitution. For tasks with recommended disclosure, we used LLMs to edit the manuscript for readability and to help format references; all references were checked against their sources by the authors. All benchmark numbers in this paper are produced by our evaluation pipeline and are traced to raw outputs through a results manifest; no number was generated or estimated by an LLM. We have reviewed all AI-assisted work and take full responsibility for the final content of this paper, including its text, claims, and artifacts.

Ethics statement

This work is an empirical audit of a training-time intervention for tool-augmented vision-language models. It uses only publicly released base models and benchmarks, involves no human subjects, crowdworkers, or personally identifiable data, and releases no new foundation model or scraped dataset. The intervention removes the returned-image channel from the tool loop and therefore reduces rather than expands the capabilities of the underlying model; we do not identify societal risks beyond those already documented for general-purpose VLMs. All third-party assets are used within their license terms (Appendix A).

Reproducibility statement

The base models (Qwen2.5-VL-7B-Instruct for the main experiments and Qwen3-VL-8B-Instruct for the coordinate-precision ablation), the public DeepEyes V2 cold-start corpus, the deterministic **TextCall** transform, and all evaluation benchmarks are specified in Section 3 and Appendix D. Training recipes, GPU configuration, approximate wall-clock, and asset licenses are listed in Appendix A. The evaluation protocol for every row of Table 1 (Agent Mode with image return, Agent Mode with skipped return, or single-turn Direct inference) is stated in the table and in Section 3. Paired-bootstrap confidence intervals are reported for the data audit, the coordinate-precision ablation, and the Jigsaw probe. Upon acceptance we will release the **TextCall** corpus-construction script and derived corpora, the trained checkpoints for both carrier arms at 9.5K and 65K scale, and the canonical multi-turn evaluation pipeline, together with the exact commands and configuration files used for the reported runs.

References

Ido Amos, Avi Caciularu, Mor Geva, Amir Globerson, Jonathan Herzig, Lior Shani, and Idan Szpektor. Latent reasoning with supervised thinking states. *arXiv preprint arXiv:2602.08332*, 2026.

Sriram Balasubramanian, Samyadeep Basu, and Soheil Feizi. A closer look at bias and chain-of-thought faithfulness of large (vision) language models. *arXiv preprint arXiv:2505.23945*, 2025.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. *arXiv preprint arXiv:2403.06764*, 2024a.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. MMStar: An evaluator-centric benchmark for multi-modal large language models, 2024b.

Yan Chen, Long Li, Teng Xi, Long Zeng, and Jingdong Wang. Perception before reasoning: Two-stage reinforcement learning for visual reasoning. *arXiv preprint arXiv:2509.13031*, 2025a.

Yanda Chen, Joe Benton, Ansh Radhakrishnan, Jonathan Uesato, Carson Denison, John Schulman, Arushi Somani, Peter Hase, Misha Wagner, Fabien Roger, Vlad Mikulik, Samuel R. Bowman, Jan Leike, Jared Kaplan, and Ethan Perez. Reasoning models don't always say what they think. *arXiv preprint arXiv:2505.05410*, 2025b.

Jiwan Chung, Junhyeok Kim, Siyeol Kim, Jaeyoung Lee, Min Soo Kim, et al. v1: Learning to point visual tokens for multimodal grounded reasoning. *arXiv preprint arXiv:2505.18842*, 2025.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, Andrew Head, Rose Hendrix, Favyen Bastani, Eli VanderBilt, Nathan Lambert, Yvonne Chou, Arnavi Chheda, Jenna Sparks, Sam Skjonsberg, Michael Schmitz, Aaron Sarnat, Byron Bischoff, Pete Walsh, Chris Newell, Piper Wolters, Tanmay Gupta, Kuo-Hao Zeng, Jon Borchardt, Dirk Groeneveld, Crystal Nam, Sophie Lebrecht, Caitlin Wittlif, Carissa Schoenick, Oscar Michel, Ranjay Krishna, Luca Weihs, Noah A. Smith, Hannaneh Hajishirzi, Ross Girshick, Ali Farhadi, and Aniruddha Kembhavi. Molmo and PixMo: Open weights and open data for state-of-the-art vision-language models. *arXiv preprint arXiv:2409.17146*, 2024.

Yifan Du, Zikang Liu, Yifan Li, Wayne Xin Zhao, Yuqi Huo, Bingning Wang, Weipeng Chen, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. Virgo: A preliminary exploration on reproducing o1-like MLLM. *arXiv preprint arXiv:2501.01904*, 2025a.

Yifan Du, Kun Zhou, Yingqian Min, Yue Ling, Wayne Xin Zhao, and Youbin Wu. Revisiting the necessity of lengthy chain-of-thought in vision-centric reasoning. *arXiv preprint arXiv:2511.22586*, 2025b.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, Dahua Lin, and Kai Chen. VLMEvalKit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024.

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. *arXiv preprint arXiv:2505.15879*, 2025.

Jiayi Fu, Xuandong Zhao, Chengyuan Yao, Heng Wang, Qi Han, and Yanghua Xiao. Reward shaping to mitigate reward hacking in RLHF. *arXiv preprint arXiv:2502.18770*, 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. *arXiv preprint arXiv:2404.12390*, 2024.

Shujian Gao, Yuan Wang, Jiangtao Yan, Zuxuan Wu, and Yu-Gang Jiang. Thinking with deltas: Incentivizing reinforcement learning via differential visual reasoning policy. *arXiv preprint arXiv:2601.06801*, 2026.

Gemini Team, Google. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

GLM-V Team. GLM-5V-Turbo: Toward a native foundation model for multimodal agents. *arXiv preprint arXiv:2604.26752*, 2026.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. In *arXiv preprint arXiv:2412.06769*, 2024.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, Guohai Xu, and Xing Yu. DeepEyesV2: Toward agentic multimodal model. *arXiv preprint arXiv:2511.05271*, 2025.

Xinhai Hou, Shaoyuan Xu, Manan Biyani, Moyan Li, Jia Liu, Todd C. Hollon, and Bryan Wang. CodeV: Code with images for faithful visual reasoning via tool-aware policy optimization. *arXiv preprint arXiv:2511.19661*, 2025.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, et al. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. *arXiv preprint arXiv:2406.09403*, 2024.

Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, Tianyu Pang, and Wenhu Chen. VerlTool: Towards holistic agentic reinforcement learning with tool use. *arXiv preprint arXiv:2509.01055*, 2025.

Kimi Team. Kimi K2.5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

Xin Lai, Junyi Li, Wei Li, Tao Liu, Tianjian Li, and Hengshuang Zhao. Mini-o3: Scaling reasoning patterns and interaction turns for visual reasoning. *arXiv preprint arXiv:2509.07969*, 2025.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilė Lukošiūtė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, Saurav Kadavath, Shannon Yang, Thomas Henighan, Timothy Maxwell, Timothy Telleen-Lawton, Tristan Hume, Zac Hatfield-Dodds, Jared Kaplan, Jan Brauner, Samuel R. Bowman, and Ethan Perez. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*, 2023.

Ang Li, Charles Wang, Deqing Fu, Kaiyu Yue, Zikui Cai, Wang Bill Zhu, Ollie Liu, Peng Guo, Willie Neiswanger, Furong Huang, Tom Goldstein, and Micah Goldblum. Zebra-cot: A dataset for interleaved vision language reasoning. *arXiv preprint arXiv:2507.16746*, 2025a.

Ming Li, Jike Zhong, Shitian Zhao, Haoquan Zhang, Shaoheng Lin, et al. Tir-bench: A comprehensive benchmark for agentic thinking-with-images reasoning. *arXiv preprint arXiv:2511.01833*, 2025b.

Yuqi Liu, Bohao Peng, Zhisheng Zhong, Zihao Yue, Fanbin Lu, et al. Seg-zero: Reasoning-chain guided segmentation via cognitive reinforcement. *arXiv preprint arXiv:2503.06520*, 2025a.

Zujing Liu, Junwen Pan, Qi She, Yuan Gao, and Guisong Xia. On the faithfulness of visual thinking: Measurement and enhancement. *arXiv preprint arXiv:2510.23482*, 2025b.

Zuyan Liu, Yuhao Dong, Yongming Rao, Jie Zhou, and Jiwen Lu. Chain-of-spot: Interactive reasoning improves large vision-language models. *arXiv preprint arXiv:2403.12966*, 2024.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. Chameleon: Plug-and-play compositional reasoning with large language models. In *Advances in Neural Information Processing Systems*, 2023.

Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*, 2025.

Yan Ma, Weiyu Zhang, Tianle Li, Linge Du, Xuyang Shen, and Pengfei Liu. What does vision tool-use reinforcement learning really learn? disentangling tool-induced and intrinsic effects for crop-and-zoom. *arXiv preprint arXiv:2602.01334*, 2026.

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. *arXiv preprint arXiv:2203.10244*, 2022.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-Symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*, 2024.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Biber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

OpenAI. Thinking with images. https://openai.com/index/thinking-with-images/, 2025. Blog post, April 16, 2025.

Siqu Ou, Tianrui Wan, Zhiyuan Zhao, Junyu Gao, and Xuelong Li. Do MLLMs really see it: Reinforcing visual attention in multimodal LLMs. *arXiv preprint arXiv:2602.08241*, 2026.

Ji Qi, Ming Ding, Weihan Wang, Yushi Bai, Qingsong Lv, et al. Cogcom: A visual language model with chain-of-manipulations reasoning. *arXiv preprint arXiv:2402.04236*, 2024.

Qwen Team. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

Qwen Team. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025a.

Qwen Team. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025b.

Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. Vision language models are blind. *arXiv preprint arXiv:2407.06581*, 2024.

Gabriel Sarch, Snigdha Saha, Naitik Khandelwal, Ayush Jain, Michael J. Tarr, Aviral Kumar, and Katerina Fragkiadaki. ViGoRL: Grounded reinforcement learning for visual reasoning. *arXiv preprint arXiv:2505.23678*, 2025.

Timo Schick, Jane Dwivedi-Yu, Roberto Lipkin, Fabio Petroni, Luke Zettlemoyer, and Mike Lewis. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023.

Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Huang, Letian Gao, Shuai Li, Yu Liu, and Ping Luo. Visual CoT: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. *arXiv preprint arXiv:2403.16999*, 2024.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. In *Advances in Neural Information Processing Systems*, 2023.

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. *arXiv preprint arXiv:2409.19256*, 2024.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Oleg Somov, Mikhail Chaichuk, Mikhail Seleznyov, Alexander Panchenko, and Elena Tutubalina. Breaking the chain: A causal analysis of LLM faithfulness to intermediate structures. *arXiv preprint arXiv:2603.16475*, 2026.

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

Zhaochen Su, Peng Xia, Hangyu Guo, Zhenhua Liu, Yan Ma, Xiaoye Qu, Jiaqi Liu, Yanshu Li, Kaide Zeng, Zhengyuan Yang, Linjie Li, Yu Cheng, Heng Ji, Junxian He, and Yi R. Fung. Thinking with images for multimodal reasoning: Foundations, methods, and future frontiers. *arXiv preprint arXiv:2506.23918*, 2025b.

Xiaokun Sun, Yubo Wang, Haoyu Cao, and Linli Xu. When thinking hurts: Mitigating visual forgetting via frame repetition. *arXiv preprint arXiv:2603.16256*, 2026a.

Yuxi Sun, Aoqi Zuo, Haotian Xie, Wei Gao, Mingming Gong, and Jing Ma. FACT-E: Causality-inspired evaluation for trustworthy chain-of-thought reasoning. *arXiv preprint arXiv:2604.10693*, 2026b.

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual inference via python execution for reasoning. In *International Conference on Computer Vision*, 2023.

Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yifan Ma, Yongming Zhang, Yann LeCun, and Saining Xie. CV-Bench: A computer vision benchmark for evaluating visual perception in multimodal language models. In *NeurIPS 2024 Workshop on Position-Aware AI*, 2024.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, 2023.

Rheeya Uppaal, Phu Mon Htut, Min Bai, Nikolaos Pappas, Zheng Qi, and Sandesh Swamy. Journey before destination: Visual faithfulness in slow thinking. *arXiv preprint arXiv:2512.12218*, 2025.

Fengxiang Wang, Mingshuo Chen, Yueying Li, Yajie Yang, Yifan Zhang, Long Lan, Xue Yang, Hongda Sun, Yulin Wang, Di Wang, Jun Song, Jing Zhang, and Bo Du. GeoEyes: On-demand visual focusing for ultra-high-resolution remote sensing. *arXiv preprint arXiv:2602.14201*, 2026a.

Haozhe Wang, Alex Su, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel-reasoner: Incentivizing pixel-scale reasoning with curiosity-driven reinforcement learning. *arXiv preprint arXiv:2505.15966*, 2025a.

Jiecong Wang, Hao Peng, and Chunyang Liu. PLaT: Latent chain-of-thought as planning: Decoupling reasoning from verbalization. *arXiv preprint arXiv:2601.21358*, 2026b.

Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Yiping Lu, Zhengyuan Yang, Lijuan Wang, Ranjay Krishna, Jiajun Wu, Li Fei-Fei, Yejin Choi, and Manling Li. VAGEN: Reinforcing world model reasoning for multi-turn VLM agents. *arXiv preprint arXiv:2510.16907*, 2025b.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023a.

Ruiyi Wang and Prithviraj Ammanabrolu. A practitioner's guide to multi-turn agentic reinforcement learning. *arXiv preprint arXiv:2510.01132*, 2025.

Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, and Dacheng Tao. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. *arXiv preprint arXiv:2408.15556*, 2024a.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023b.

Ye Wang, Qianglong Chen, Zejun Li, Siyuan Wang, Shijie Guo, Zhirui Zhang, and Zhongyu Wei. Simple o3: Towards interleaved vision-language reasoning. *arXiv preprint arXiv:2508.12109*, 2025c.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025d.

Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, Alexis Chevalier, Sanjeev Arora, and Danqi Chen. CharXiv: Charting gaps in realistic chart understanding in multimodal LLMs. *arXiv preprint arXiv:2406.18521*, 2024b.

Ziteng Wang, Yujie He, Guanliang Li, Siqi Yang, Jiaqi Xiong, and Songxiang Liu. V-FAT: Benchmarking visual fidelity against text-bias. *arXiv preprint arXiv:2601.04897*, 2025e.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, 2022.

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. *arXiv preprint arXiv:2602.11858*, 2026.

Jialong Wu, Xiaoying Zhang, Hongyi Yuan, Xiangcheng Zhang, Tianhao Huang, Changjing He, Chaoyi Deng, Renrui Zhang, Youbin Wu, and Mingsheng Long. Visual generation unlocks human-like reasoning through multimodal world models. *arXiv preprint arXiv:2601.19834*, 2026.

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. *arXiv preprint arXiv:2505.19255*, 2025a.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal LLMs. *arXiv preprint arXiv:2312.14135*, 2023.

Wenxun Wu, Yuanyang Li, Guhan Chen, Linyue Wang, and Hongyang Chen. Tool-augmented policy optimization. *arXiv preprint arXiv:2510.07038*, 2025b.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. *arXiv preprint arXiv:2307.02477*, 2023.

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. *arXiv preprint arXiv:2411.10440*, 2024.

Shilin Yan, Jintao Tong, Hongwei Xue, Xiaojun Tang, Yangyang Wang, Kunyu Shi, Guannan Zhang, Ruixuan Li, and Yixiong Zou. Act wisely: Cultivating meta-cognitive tool use in agentic multimodal models. *arXiv preprint arXiv:2604.08545*, 2026.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in GPT-4V. *arXiv preprint arXiv:2310.11441*, 2023.

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.

Senqiao Yang, Junyi Li, Xin Lai, Bei Yu, Hengshuang Zhao, and Jiaya Jia. VisionThink: Smart and efficient visual language model via reinforcement learning. *arXiv preprint arXiv:2507.13348*, 2025a.

Shuo Yang, Yuwei Niu, Yuyang Liu, Yang Ye, Bin Lin, and Li Yuan. Look-back: Implicit visual re-focusing in MLLM reasoning. *arXiv preprint arXiv:2507.03019*, 2025b.

Wenhao Yang, Yu Xia, Jinlong Huang, Shiyin Lu, Qing-Guo Chen, Zhao Xu, Weihua Luo, Kaifu Zhang, Yuchen Zhou, Xiaobo Xia, Yuanyu Wan, Lijun Zhang, and Tat-Seng Chua. Walk the talk: Bridging the reasoning-action gap for thinking with images via multimodal agentic policy optimization. *arXiv preprint arXiv:2604.06777*, 2026.

Wenxi Yang, Yuzhong Zhao, Fang Wan, and Qixiang Ye. Thinking with images via self-calling agent. *arXiv preprint arXiv:2512.08511*, 2025c.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, 2023a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023b.

Richard J. Young. Lie to me: How faithful is chain-of-thought reasoning in reasoning models? *arXiv preprint arXiv:2603.22582*, 2026.

Hao Zhang, Mingjie Liu, Shaokun Zhang, Songyang Han, Jian Hu, Zhenghui Jin, Yuchi Zhang, Shizhe Diao, Ximing Lu, Binfeng Xu, Zhiding Yu, Jan Kautz, and Yi Dong. ProRL agent: Rollout-as-a-service for reinforcement learning training of multi-turn LLM agents. *arXiv preprint arXiv:2603.18815*, 2026a.

Jusheng Zhang, Kaitong Cai, Xiaoyang Guo, Sidi Liu, Qinhan Lv, Ruiqi Chen, Jing Yang, Yijia Fan, Xiaofei Sun, Jian Wang, Ziliang Chen, Liang Lin, and Keze Wang. MM-CoT: A benchmark for probing visual chain-of-thought reasoning. *arXiv preprint arXiv:2512.08228*, 2025a.

Shaolei Zhang, Qingkai Fang, Zhe Yang, and Yang Feng. LLaVA-Mini: Efficient image and video large multimodal models with one vision token. *arXiv preprint arXiv:2501.03895*, 2025b.

Yi-Fan Zhang, Huanyu Zhang, Haochen Tian, Chaoyou Fu, Shuangqing Zhang, Junfei Wu, Feng Li, Kun Wang, Qingsong Wen, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. MME-RealWorld: Could your multimodal LLM challenge high-resolution real-world scenarios that are difficult for humans? *arXiv preprint arXiv:2408.13257*, 2024.

Yi-Fan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, et al. Thyme: Think beyond images. *arXiv preprint arXiv:2508.11630*, 2025c.

Yifan Zhang, Liang Hu, Haofeng Sun, Peiyu Wang, Yichen Wei, et al. Skywork-r1v4: Toward agentic multimodal intelligence through interleaved thinking with images and deepresearch. *arXiv preprint arXiv:2512.02395*, 2025d.

Zhen Zhang, Kaiqiang Song, Xun Wang, Yebowen Hu, Weixiang Yan, Chenyang Zhao, Henry Peng Zou, Haoyun Deng, Sathish Reddy Indurthi, Shujian Liu, Simin Ma, Xiaoyang Wang, Xin Eric Wang, and Song Wang. CM2: Reinforcement learning with checklist rewards for multi-turn and multi-step agentic tool use. *arXiv preprint arXiv:2602.12268*, 2026b.

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*, 2023.

Rosie Zhao, Anshul Shah, Xiaoyu Zhu, Xinke Deng, Zhongyu Jiang, Yang Yang, Joerg Liebelt, and Arnab Mondal. On robustness and chain-of-thought consistency of RL-finetuned VLMs. *arXiv preprint arXiv:2602.12506*, 2026.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. *arXiv preprint arXiv:2312.07104*, 2023.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, 2024.

Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. Deepeyes: Incentivizing "thinking with images" via reinforcement learning. *arXiv preprint arXiv:2505.14362*, 2025.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations*, 2023.

Yiyang Zhou, Haoqin Tu, Zijun Wang, Zeyu Wang, Niklas Muennighoff, Fan Nie, Yejin Choi, James Zou, Chaorui Deng, Shen Yan, Haoqi Fan, Cihang Xie, Huaxiu Yao, and Qinghao Ye. When visualizing is the first step to reasoning: MIRA, a benchmark for visual chain-of-thought. *arXiv preprint arXiv:2511.02779*, 2025.

Yujin Zhou, Pengcheng Wen, Jiale Chen, Boqin Yin, Han Zhu, Jiaming Ji, Juntao Dai, Chi-Min Chan, and Sirui Han. Process reward for thinking with images. *arXiv preprint arXiv:2602.08346*, 2026.

A Additional Experimental Setup

Hardware. All training and evaluation jobs use NVIDIA H20 GPUs. Per-run configurations are listed in Table 5.

Compute usage. Table 5 lists GPU configuration and approximate wall-clock per run for the experiments reported in the main paper. Wall-clock figures are rounded to reflect shared-cluster contention and exclude scheduler bring-up; the Agent Mode eval range spans V*Bench (191 samples, $\sim$5 min on DP=4) at the low end and MME-RealWorld (23K samples, $\sim$10–12 h including judge tail) at the high end. Totals are dominated by 65K full FT, RL post-training, and the canonical multi-benchmark Agent Mode evaluation sweep over MME-RealWorld.

Table 5: **Compute usage per reported run.** H20-class GPUs throughout; wall-clock values are approximate and exclude scheduler bring-up. † and ‡ are defined below the table.

Run	Configuration	Wall-clock
9.5K LoRA SFT (rank 8, per group)	$1 \times 8\times$H20	$\sim$1 h
65K full FT SFT (cold-start, per group)	$8 \times 8\times$H20	$\sim$13 h
RL post-training (GRPO, step 800)†	$8 \times 8\times$H20	$\sim$25–30 h
Visual Jigsaw probe RL (GRPO, $n = 50$)	$1 \times 8\times$H20	$\sim$8 h
Agent Mode eval (per ckpt $\times$ bench, DP=4)‡	$1 \times 8\times$H20 + judge	$\sim$0.1–12 h

† $\sim$1 epoch, batch 128, lr 1×10^{-6}. ‡ SGLang (Zheng et al., 2023) runner.

Main SFT and evaluation setup. The main 65K thinking-with-images/**TextCall** SFT runs follow the public DeepEyesV2 cold-start configuration unchanged so that the only manipulated variable is the tool-return path. The 9.5K runs use LoRA fine-tuning (rank 8, learning rate 10^{-4}, 3 epochs), matching the small-scale protocol used for the corresponding ablations. Tool-trained models are evaluated in Agent Mode with either image feedback or **TextCall** text-only feedback. The base model is reported under Direct MC as a single-turn baseline and under Agent Mode for runner calibration. All Agent/**TextCall** evaluations use the same VLMEvalKit-canonical runner and judge configuration across compared checkpoints.

Existing assets and licenses. Table 6 lists the third-party assets used in this work and their licenses or terms of use. All assets are used within their stated terms.

Table 6: **Existing assets used in this work.**

Asset	Type	License / Terms
Qwen2.5-VL-7B-Instruct (Qwen Team, 2025a)	Base model	Apache-2.0
DeepEyesV2 cold-start corpus (Hong et al., 2025)	Training data (SFT)	Research-only, per official release
DeepEyesV2 RL corpus (Hong et al., 2025)	Training data (RL)	Research-only, per official release
V*Bench (Wu & Xie, 2023)	Benchmark	MIT, per official release
HR-Bench-4K / 8K (Wang et al., 2024a)	Benchmark	Apache-2.0, per official release
MMStar (Chen et al., 2024b)	Benchmark	Apache-2.0, per official release
CV-Bench (2D / 3D) (Tong et al., 2024)	Benchmark	MIT, per official release
MME-RealWorld (Zhang et al., 2024)	Benchmark	Research-only, per official release
BLINK (Fu et al., 2024)	Benchmark	CC-BY-4.0, per official release
CharXiv (Wang et al., 2024b)	Benchmark	MIT, per official release
ChartQA (Masry et al., 2022)	Benchmark	GPL-3.0, per official release
PixmoCount (Deitke et al., 2024)	Benchmark	Apache-2.0, per official release
Zebra-CoT (Visual Jigsaw subset) (Li et al., 2025a)	Benchmark / data	Research-only, per official release
verl (Sheng et al., 2024)	Training framework	Apache-2.0
LlamaFactory (Zheng et al., 2024)	Training framework	Apache-2.0
VLMEvalKit (Duan et al., 2024)	Eval framework	Apache-2.0
Gemini 3 Pro / Nano Banana Pro (Gemini Team, Google, 2023)	API (data synthesis)	Per Google ToS

B Data Audit Details

The headline statistics in Section 4 come from a four-condition factorial audit over all $N = 1{,}000$ sampled DeepEyesV2 SFT trajectories (500 perception + 500 computation, seed 42). A Gemini-3-Flash judge (temperature 0) answers the same question under four input conditions: question only, question plus original image, question plus structured spatial prompt without image, and question plus both image and prompt.

Trajectory parsing. Each trajectory is parsed into the user question, the original image, the first assistant reasoning block before code execution, the code blocks that specify crop/zoom operations, and the final answer. The structured spatial prompt used in the audit is restricted to the first `<think>` block and code blocks. It excludes tool execution results, later reasoning, and the final answer. We verified that the final-answer literal appears in fewer than 1% of scaffold code blocks (5 out of 1,000 samples), indicating that direct answer copying is not driving the scaffold-only result.

Judge conditions. The *question-only* condition gives the judge only the question. The *image-only* condition adds the original image. The *scaffold-only* condition adds the structured spatial prompt but withholds the image. The *combined* condition gives both the image and the prompt and serves as the reference. The resulting accuracies are 51.20%, 73.50%, 73.10%, and 79.40%, respectively.

Paired uncertainty. We compare *scaffold-only* against *image-only* with paired bootstrap resampling over the same $N = 1{,}000$ samples. The difference is -0.40 pp with 95% CI $[-3.20, +2.40]$, which passes a 5 pp non-inferiority margin. Thus the structured spatial prompt recovers equivalent accuracy to the original image under this audit protocol.

Gain decomposition. Relative to the *question-only* baseline, the *combined* condition gains 28.2 pp. Of this total gain, 57% is redundant accuracy gain recovered by both image and scaffold, 22% is image-unique, and 21% is scaffold-unique. The paired overlap is: both correct 62.8%, image-only 10.7%, scaffold-only 10.3%, and neither 16.2%.

Caveats. The audit is descriptive mechanism evidence, not the primary causal claim. The causal claim is established by the training-time intervention in Section 3.3; this audit explains why that intervention can preserve accuracy under the evaluated training distribution.

C TextCall Scaffold-Decomposition Details

Section 4.2 decomposes the **TextCall** scaffold while keeping the multi-turn return policy fixed. All variants use the same 9.5K **TextCall** LoRA setting (rank 8, learning rate 10^{-4}, 3 epochs) and the same Agent Mode evaluation protocol with **TextCall** text-only feedback and Gemini-3-Flash judging. The only manipulated field is the assistant text inside each trajectory.

Table 7: **Scaffold-decomposition variants.** All rows use the same 9.5K LoRA questions and Agent Mode evaluation with **TextCall** text-only feedback. The first three rows match Section 4.2; *Skeleton only* keeps the multi-turn wrapper but removes both reasoning text and executable code.

Variant	Assistant content	V*	HR-4K	CV-2D	Mean
Full scaffold	reasoning + executable code	76.96	71.50	**70.93**	**73.13**
Reasoning only	reasoning + `pass`	67.02	72.12	59.81	66.32
Code only	empty reasoning + executable code	64.92	65.50	55.08	61.83
Skeleton only	empty reasoning + `pass`	**79.06**	**72.62**	63.21	71.63

The decomposition supports a benchmark-dependent reading. Reasoning alone is sufficient on HR-4K but loses spatial grounding on V*Bench and CV-Bench-2D, while code without reasoning produces hallucinated spatial attention under the no-pixel-return protocol. The skeleton-only variant shows that the multi-turn wrapper itself is benign: because both reasoning and code are empty, the model learns no scaffold content and effectively falls back to the base model's direct-answering capability within the agent format. Its high accuracy therefore reflects base-model strength, not scaffold contribution,

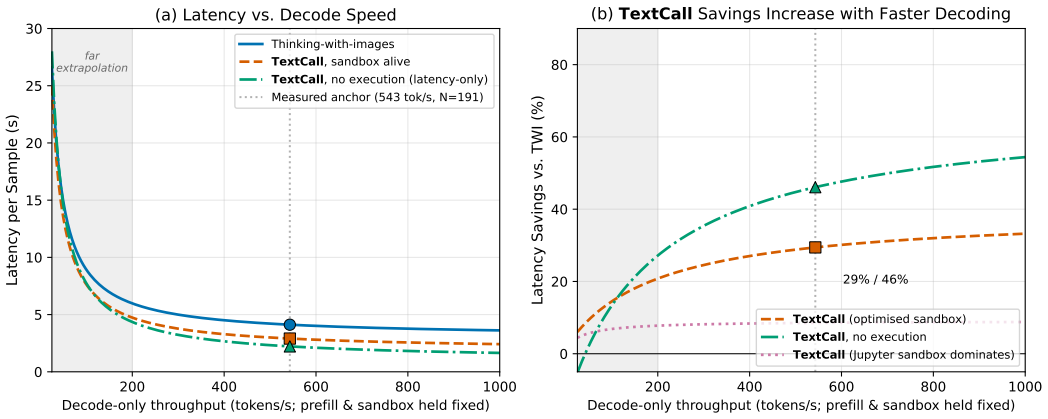


Figure 4: **Token-speed sweep: TextCall's latency advantage increases with faster decoding.** Curves vary decode speed while holding measured non-decode costs fixed. Markers show the measured V*Bench means at 543 tok/s; the shaded region below 200 tok/s marks far extrapolation where batching and scheduler effects are not modeled. (a) Absolute latency per sample. **TextCall** avoids returned-image vision prefill, giving it a lower latency floor under our sequential per-sample implementation. (b) Latency savings relative to thinking-with-images. The dotted line shows the original kernel-based sandbox setting, where a large shared sandbox cost reduces the savings at the measured anchor.

and should not be read as evidence that structure alone carries the gain. It also bounds the scaffold's contribution: on V*Bench and HR-4K the full scaffold does not exceed this fallback, whereas on CV-Bench-2D it leads it by 7.72 points.

D TEXTCALL RETURN-TEXT ABLATION

Our **TextCall** intervention replaces every returned image with the literal text `[Image output skipped]` and changes nothing else in the trajectory. This fixed-sentinel return policy keeps the model's tool-call scaffold intact (the model still produces `crop`/`zoom` arguments, code, search queries, etc.) while removing the post-call pixel carrier.

No caption or oracle information enters training. All **TextCall** runs reported in Table 1 (**TextCall** at 9.5K LoRA and **TextCall** at 65K full FT) use the same fixed placeholder: 100% of samples replace the returned image with `[Image output skipped]`. We never provide captions of returned images or ground-truth-derived answer hints, so the reported carrier-swap results isolate the fixed text sentinel from caption-return or oracle-return alternatives.

What this ablation does *not* address. The fixed-placeholder intervention cannot tell us whether a higher-quality external caption would have helped the model, nor whether oracle hints would have closed the residual gaps on benchmarks where neither thinking-with-images nor **TextCall** reaches state-of-the-art. We treat such positive-evidence experiments as future work, consistent with the limitations discussed in Section 6.

Execution policies and latency measurement. The canonical **TextCall** accuracy rows execute the same agent loop as thinking-with-images, but the observation returned to the model is always the fixed text placeholder rather than a new image. The latency-only no-execution row in Table 4 is a separate reference: the model still emits tool-call text, but the sandbox call is skipped after generation. This row estimates the pure inference lower bound and is not used for accuracy comparisons.

E Qualitative Scaffold Examples

This appendix complements the main-paper scaffold overview (Figure 2) with additional V*Bench cases where the two Agent Mode paradigms, thinking-with-images and **TextCall**, answer correctly. Each case uses the same two-column layout as the main figure: the left panel shows the thinking-with-images model's multi-turn interaction with a real cropped image returned, while the right panel shows **TextCall**'s closely matched tool-call scaffold with the image return replaced by a text placeholder. These qualitative cases illustrate the carrier-swap behavior behind the controlled experiments; as in the main-paper figure, reasoning text and crop coordinates are simplified for display. Figures 5 and 6 present two additional cases spanning color and spatial-relation queries.

F Visual Jigsaw Probe: Cross-Tool-Family Consistency Check

The main experiments (Section 3) use crop/zoom/code tools. To check whether the no-degradation pattern extends to a qualitatively different tool family, we construct a small *Visual Jigsaw* probe ($n = 50$) using a text-to-image (T2I) generator. The probe is a consistency check, not a benchmark: it uses 50 manually verified samples filtered only for annotation validity, never on model outcomes.

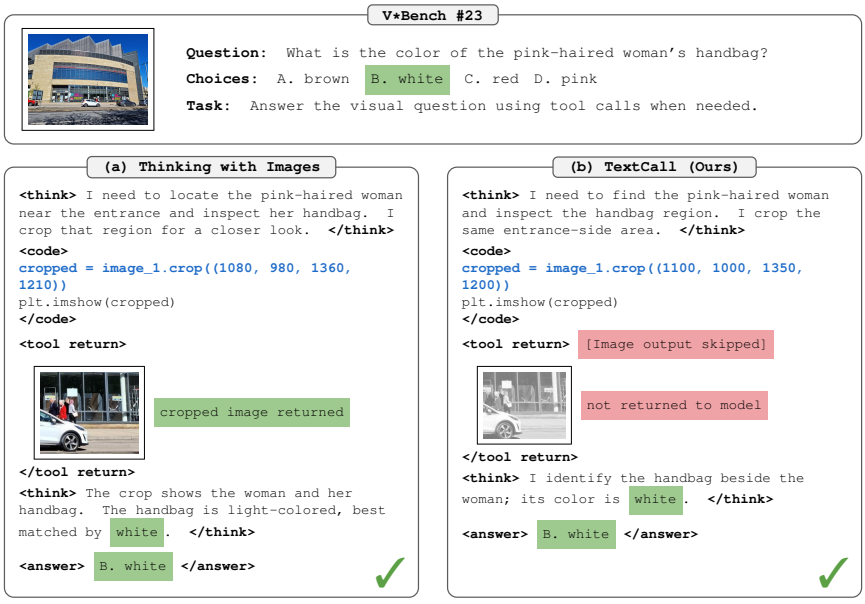


Figure 5: **Qualitative example A (V*Bench #23, direct_attributes).** "What is the color of the pink-haired woman's handbag?" The thinking-with-images model crops the region around the woman and reads the handbag color from the returned image; **TextCall** issues a closely matched crop but receives only a text placeholder, yet still correctly identifies the white handbag from the spatial scaffold context.

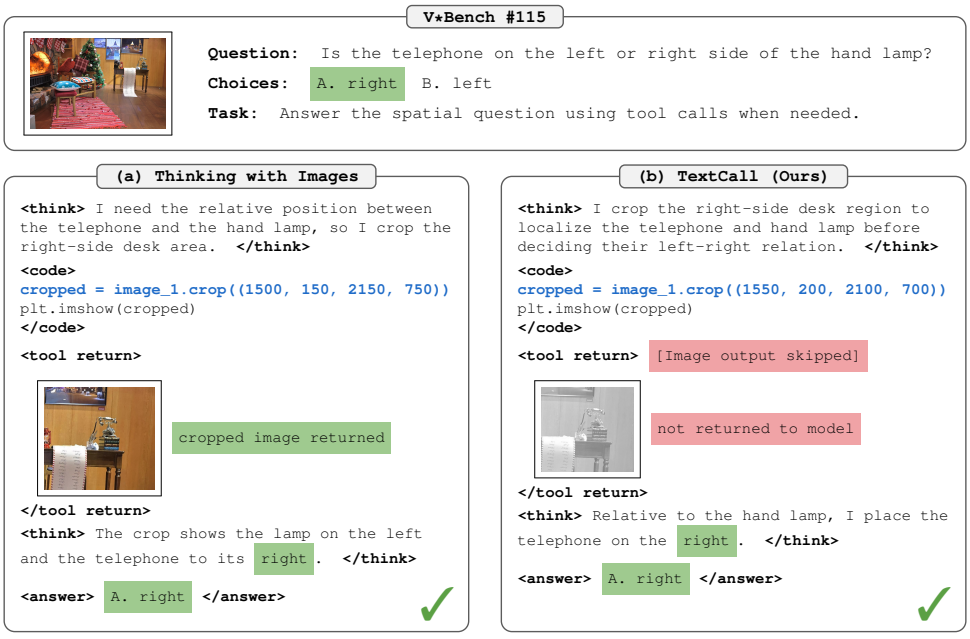
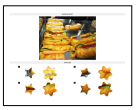


Figure 6: **Qualitative example B (V*Bench #115, relative_position).** "Is the telephone on the left or right side of the hand lamp?" A spatial-relation question requiring precise localization. Both paradigms crop the desk area; **TextCall** infers the relative position from the coordinate scaffold without seeing the cropped pixels.

Representative Visual Jigsaw trajectory



Question: Identify which option fills the gray jigsaw holes.

Choices: A B C D

Task: Test candidate fragment sets with T2I edits; answer the matching option.

Converted SFT chat trajectory (1/3)

assistant:

Image Analysis: The puzzle shows a tray of hot dog buns. There are two gray star-shaped gaps.
1. **Top Gap**: Located on the sandwich in the middle row. It interrupts a line of red ketchup and is immediately to the left of a green cucumber slice.
2. **Bottom Gap**: Located on the sandwich in the bottom row, towards the right end. It covers the bun and the end of the sausage.

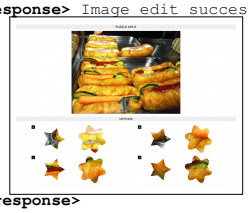
Strategy: I will test each option (A, B, C, D) by checking if its two pieces contain the specific visual features required to seamlessly fill the two gaps.
Now I need to: Examine Option A's pieces against the gaps.

<tool_call>

{"name": "image_edit", "arguments": {"prompt":
Edit the image: Test Option A by placing its two star-shaped pieces into the gray gaps in the main puzzle image.
1. **SELECT the pieces from Option A**: Locate the two star-shaped image fragments under the label 'A' at the bottom left.
2. **IDENTIFY the target gaps**: Locate the two gray star-shaped holes in the 'PUZZLE INPUT' image (one on the top-right sandwich, one on the bottom-center sandwich).
3. **PLACE the pieces**:
- Take the left piece from Option A (showing a metal tray edge) and place it over one of the gaps.
- Take the right piece from Option A (showing ham and bread) and place it over the other gap.
4. **BLEND**: Attempt to align the edges. The goal is to see if the ingredients (sausage, cucumber, bread, tray) match the surrounding context perfectly.
5. **OUTPUT**: Show the modified main image with the Option A pieces filled in.
, "img_idx": 0}}

</tool_call>

<tool_response> Image edit successful. **<image>**



</tool_response>

---

assistant:

Let me check the result: The user successfully placed the pieces from Option A into the designated gaps. However, the content of the pieces does not match the surrounding visual context.
The visualization is correct, but this option doesn't match. Let me test another option.
Option A has been eliminated. The plan must continue to test the remaining options sequentially to find the correct match.
Now I need to: Examine Option B's pieces against the gaps. Place the left piece in the top gap and the right piece in the bottom gap.

<tool_call>

{"name": "image_edit", "arguments": {"prompt":
Edit the image: Test Option B by placing its pieces into the puzzle gaps.
1. **SELECT the pieces from Option B** at the bottom of the image:
- Left piece: Shows golden fried dough texture with a dark metallic tray edge.
- Right piece: Shows a segment of an orange sausage on a bun.
2. **PLACE the Left Piece** into the **Top Gap**:
- The top gap is the gray star shape on the sandwich in the middle-right.
- Align the piece so the texture fills the hole.
3. **PLACE the Right Piece** into the **Bottom Gap**:
- The bottom gap is the gray star shape on the sandwich in the bottom-left (cutting through the sausage).
- Align the sausage segment in the piece with the sausage in the sandwich.
4. **PRESERVE** the rest of the image exactly as is.
, "img_idx": 0}}

</tool_call>

<tool_response> Image edit successful. **<image>**



</tool_response>

Figure 7: **Visual Jigsaw probe: converted SFT chat trajectory.** The figure follows the post-conversion chat format: Agent messages interleave raw `image_edit` tool calls with returned images while testing candidate options before committing to Option C.

Converted SFT chat trajectory (2/3)

assistant:

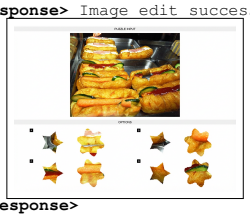
Let me check the result: The edit failed to execute the instructions correctly. The Top Gap remains completely empty and gray. The Bottom Gap has been filled, but with the wrong piece.
It seems I made a mistake in the visualization. Let me try again.
The previous attempt failed to fill the top gap and swapped the pieces for the bottom gap. To correctly test Option B, I must explicitly instruct the model to take the specific source patches from Option B and place them into the specific destination gaps.

<tool_call>

{"name": "image_edit", "arguments": {"prompt":
Edit the image to visually test Option B by filling the puzzle gaps.
1. **Source - Option B Left Piece**: Identify the left piece under 'B'. It is a star shape containing a metallic tray edge on the left and fried dough on the right.
2. **Destination - Top Gap**: Place this Left Piece into the gray star-shaped gap located on the sandwich on the right side of the tray.
3. **Source - Option B Right Piece**: Identify the right piece under 'B'. It is a star shape containing an orange sausage segment at the top and fried dough at the bottom.
4. **Destination - Bottom Gap**: Place this Right Piece into the gray star-shaped gap located on the bottom-left sandwich. Align the orange sausage segment in the piece with the sausage in the sandwich.
5. **Refinement**: Resize and rotate the pieces slightly if necessary to fit the gray star masks exactly. Remove the gray color completely. Do not alter the surrounding sandwiches.
, "img_idx": 0}}

</tool_call>

<tool_response> Image edit successful. **<image>**



</tool_response>

---

assistant:

Let me check the result: The edit successfully executed the instructions to place the pieces from Option B into the gaps. However, the content does not match the scene.
Placement successful, but the content mismatch confirms this is not the right choice.
The correct option must consist entirely of food textures. Specifically, the top gap requires a piece with bun, cucumber, and ketchup, while the bottom gap requires a piece with bun and a sausage segment.
Based on the failure of A and B, and the visual analysis of the gaps, the plan should proceed to test Option C.

<tool_call>

{"name": "image_edit", "arguments": {"prompt":
Edit the image to test Option C by placing its pieces into the puzzle gaps.
1. **Identify the Gaps**:
- Top Gap: Star-shaped hole on the right sandwich (middle row).
- Bottom Gap: Star-shaped hole on the left sandwich (bottom row).
2. **Place Option C Pieces**:
- Take the **left piece of Option C** (containing the red ketchup line and green cucumber slice) and place it into the **Top Gap**. Align the red ketchup line and green cucumber with the existing ingredients on the sandwich.
- Take the **right piece of Option C** (containing the orange sausage segment and golden bun texture) and place it into the **Bottom Gap**. Align the sausage segment so it connects the severed sausage ends in the sandwich.
3. **Refine**:
- Ensure the textures (bun, sausage, sauce) blend seamlessly with the surrounding image.
- Remove the gray placeholder color completely in the filled areas.
, "img_idx": 0}}

</tool_call>

<tool_response> Image edit successful. **<image>**



</tool_response>

Figure 7: **Visual Jigsaw probe: converted SFT chat trajectory, continued.** The second page shows the retry of Option B after the previous tool response failed to fill the top gap.

Converted SFT chat trajectory (3/3)

assistant:

```
Let me check the result:  The edit successfully placed the pieces from Option C into the designated gaps.
   The top gap was filled with the piece containing the ketchup line and cucumber, and the bottom gap was
   filled with the piece containing the sausage end.
The visualization looks correct.
Evaluation:
1.  **Option A**:  Eliminated.
2.  **Option B**:  Eliminated.
3.  **Option C**:  Success.
4.  **Option D**:  Not tested, but unnecessary.
Based on the visual analysis above, I can conclude that:  Option C
```


Figure 7: **Visual Jigsaw probe: converted SFT chat trajectory, continued.** All four tool-call blocks preserve the raw `image_edit` prompts from the training JSON. Table 8 reports the exploratory placeholder-return comparison.

Design. The probe is built on the *Visual Jigsaw* subset of Zebra-CoT (Li et al., 2025a). We synthesize paired SFT corpora with a Planner–Actor–Critic–Replanner multi-agent pipeline implemented in LangGraph, following the plan-and-solve and reflective-critic patterns established in prior agent work (Wang et al., 2023a; Yao et al., 2023b; Shinn et al., 2023): a Gemini 3 Pro *planner* decomposes each Zebra-CoT ground-truth solution into a structured step list; a Gemini 3 Pro *actor* rewrites the current step into a text-to-image instruction; Nano Banana Pro executes the image edit; a Gemini 3 Pro *critic* verifies whether the returned image matches the expected sub-goal; and a Gemini 3 Pro *replanner* decides whether to accept, retry the current step, or revise downstream steps based on the critic's verdict. The two corpora differ only in the tool-return slot: one keeps the T2I image, the other replaces it with "`Image edit successful. [no image returned]`". All other tokens are byte-identical. Figure 7 illustrates the converted SFT chat trajectory after this pipeline: Agent outputs, tool calls, and tool responses appear in message order, with the raw scaffold content preserved inside each tool-call prompt.

Both runs use the same base (Qwen2.5-VL-7B-Instruct), SFT config (LoRA rank-8, 10 epochs), and RL config (GRPO via verl, 8×H20). The only manipulated variable is the tool-return path. The real-tool run is evaluated under two policies: *with-tools* (T2I image returned) and *no-tool* (image suppressed at eval).

Results. We retain RL checkpoints with tool-execution failure $< 40\%$. Trajectory-mean accuracy across retained checkpoints is indistinguishable: 41.1% (placeholder return) vs. 41.2% (real-tool with-tools) and 44.6% (real-tool no-tool). Table 8 reports per-run peaks with paired-bootstrap CIs.

Table 8: **Visual Jigsaw probe results ($n = 50$, paired).** Traj. mean averages retained RL checkpoints; Peak acc. reports best checkpoint accuracy and step. Δ is computed against the placeholder-return peak with paired bootstrap $B = 10{,}000$; — marks the reference row.

Run (eval return policy)	Traj. mean	Peak acc. (step)	Δ vs. placeholder (95% CI, p)
Placeholder return	41.1%	**33/50=66%** (s120)	—
Real-tool (with-tools)	41.2%	$26/50 = 52\%$ (s50/90)	$+14$pp $[-4, +32]$, $p = 0.13$
Real-tool (no-tool)	**44.6%**	$27/50 = 54\%$ (s100)	$+12$pp $[-6, +30]$, $p = 0.23$

Interpretation. The trajectory-mean parity is consistent with the no-degradation pattern observed in the main crop/zoom experiments. The wide bootstrap CIs reflect the small sample size ($n = 50$) and preclude strong per-checkpoint claims. In particular, the apparent 66% placeholder-return peak is a selected checkpoint; the mean over retained checkpoints is 41.1%, essentially identical to the 41.2% real-tool mean. We therefore treat the probe as an exploratory cross-tool-family null result, not as an independent benchmark or mechanism claim.

An earlier three-level content ablation is omitted because its 50-sample SFT replication was null (38%, 30%, and 36%) and tool-call count was not controlled across arms. It cannot support a claim that richer scaffold content causes the retained accuracy.

G Extended Boundary Discussion

As a hypothesis-level extension to the limitations in Section 6, two candidate conditions may make returned pixels load-bearing: (i) a *representational bottleneck* where the visual evidence cannot be losslessly verbalized, and (ii) a *prior-knowledge gap* concentrated in visual pretraining and poorly represented in text corpora. The benchmarks in our suite do not appear to jointly stress both conditions strongly. We expand here on candidate task families where these conditions might become active; verifying whether thinking-with-images then outperforms **TextCall** on such tasks remains future work.

Tasks requiring visual simulation. Wu et al. (2026) demonstrate that visual generation can unlock spatial reasoning that text-only training cannot, particularly for tasks requiring the model to mentally simulate physical transformations (e.g., rotating, folding, or intersecting objects). These tasks may satisfy both conditions: the spatial relationships are hard to describe verbally (bottleneck), and the dynamics are learned from visual experience (knowledge gap).

Tasks requiring fine-grained visual comparison. Design and construction tasks require the model to compare subtle visual differences across intermediate states. If the differences exceed the granularity of natural language—e.g., slight curve variations in architectural drawings, or minor alignment offsets in mechanical assemblies—both conditions could be met: the differences are hard to verbalize (bottleneck) and the comparison skill is rooted in visual experience (knowledge gap).

We emphasize that these are *hypotheses*, not conclusions. Verifying whether generated images provide genuine causal benefits on such tasks—using the same training-time intervention methodology—is a necessary next step. Our **TextCall** framework provides the experimental tool: if **TextCall** training matches thinking-with-images on these candidate tasks, returned pixels are still not empirically load-bearing under the tested protocol; if thinking-with-images significantly outperforms **TextCall**, the result would indicate that at least one candidate condition is active and should be analyzed directly. See Section 5 for the corresponding deployment recommendations.

H Diagnostic Evidence in the Literature

Table 9 consolidates prior evidence on the role of returned images in thinking-with-images pipelines, complementing the prose summary in Section 2. The picture is mixed: some studies find that returned pixels are under-used or redundant, while others show genuine benefits on perception-heavy tasks. Our contribution is the matched training-time carrier-swap counterpart that isolates the textual scaffold's role.

I RL Training Dynamics: A Failure Taxonomy

We apply GRPO on top of both the matched thinking-with-images and **TextCall** 65K full-FT SFT checkpoints, using identical hyperparameters (batch 128, lr 1×10^{-6}, 64 GPUs, $\sim$1 epoch). The only difference is the tool-return policy during rollout: thinking-with-images returns full images; **TextCall** returns `[Image output skipped]`. We track tool-call rate (fraction of evaluation samples where the model emits at least one `<code>` block) across training steps.

Thinking-with-images collapses to direct answering. Table 10 shows that the thinking-with-images model abandons tool use by step 300 and never recovers. At step 800 (and step 900, as a confirmation), tool-call rate is 0% across all six core benchmarks. The model reverts to single-turn direct answering despite being evaluated in the full agent loop with available tools. Accuracy remains reasonable (e.g., V*Bench 75–80%) because the base model's text reasoning is strong enough for direct answering on many samples—but the numbers no longer reflect tool-augmented exploration.

TextCall also degrades. The **TextCall** model shows a transient 100% no-code collapse at step 200, later recovers, and reaches an 83.4% mean tool-call rate at step 700. By step 800, however, that rate has fallen to 75.2%, with HR-4K dropping from 79.6% to 59.0%. This is a selected usable checkpoint in an unstable trajectory, not evidence that the text-only return policy makes RL intrinsically stable.

Table 9: **Diagnostic evidence about returned images in thinking-with-images systems.** Rows group prior inference-time, trace-audit, and training/data-side diagnostics; the final row denotes the matched training-time carrier-swap setting evaluated in this work.

Paradigm	Paper	Key Finding
Visual-mark prompting	Set-of-Mark (Yang et al., 2023)	Symbolic mark IDs enable textual spatial reference
Visual-CoT oracle probing	MIRA (Zhou et al., 2025)	Oracle visual cues yield +33.7% gain, but models cannot self-generate them
End-to-end agentic RL	DeepEyes (Zheng et al., 2025)	iMCoT (crop+text) outperforms text-only CoT; largest gap on high-res (+11.8pp HR-8K)
Causal intervention	Faithfulness (Liu et al., 2025b)	Image→noise: -0.52%; text corrupted: -13.09%
Blind testing	Deltas (Gao et al., 2026)	Removing images: $+3.5\%$
Statistical decomposition	MED (Ma et al., 2026)	>70% gains from intrinsic reasoning
Faithfulness evaluation	CodeV (Hou et al., 2025)	57% of crops contain the target
Attention reward	SAYO (Ou et al., 2026)	Original-image attention reward recovers most gains
Distillation	ZwZ (Wei et al., 2026)	Zoom benefits distilled into single-pass inference
Zero-vision SFT	Kimi K2.5 (Kimi Team, 2026)	Visual reasoning can activate without visual data
Non-interleaved agentic CoT	sCoT (Yang et al., 2025c)	Textual reasoning trajectory (subagents see crops, return text only) outperforms pixel-interleaved iMCoT
Training-time ablation	**Ours**	**TextCall $\geq$ thinking-with-images under aligned training/eval**

Reward variants fail differently. The same agent loop remains brittle under later reward designs. A status-matrix reward (exp19) raises training reward while driving the policy toward the turn limit and collapsing V*Bench, a classic reward–capability divergence. A commit-gated reward (exp22) prevents endless interaction but produces two new shortcuts: the thinking-with-images policy reaches 100% execution failure with no successful tool calls, while the **TextCall** policy reaches 100% no-code behavior. These variants show that emitting code, executing a useful tool, and committing an answer are separate behaviors that a scalar answer reward does not reliably align.

Comparison to Hong et al. (2025) and scope. Hong et al. (2025) report successful thinking-with-images RL on top of a larger cold-start corpus that includes an unreleased Long-CoT component. Their ablation also reveals an agent-format tax: placing the base model in the agent setting sharply reduces V*Bench accuracy before cold-start SFT recovers it. Our setup differs in data, RL algorithm (GRPO vs. DAPO), reward, and prompt-level filtering, so our collapse is not a replication of their training result. More importantly, DeepEyes V2 does not report a tools-off evaluation of its final checkpoint, so its gains cannot by themselves identify the causal contribution of returned pixels.

Table 10: **Tool-call rate (%) across RL training steps.** Entries report the percentage of evaluation samples that emit at least one `<code>` block under Agent Mode. Step labels are checkpoint numbers; Mean averages the six listed benchmarks.

	V*Bench	HR-4K	HR-8K	MMStar	CV-2D	CV-3D	Mean
Matched thinking-with-images RL							
Step 800	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Step 900	0.0	0.0	0.0	0.5	0.0	0.0	0.1
*Matched **TextCall** RL*							
Step 700	**86.9**	**79.6**	**75.8**	**72.5**	85.7	**100.0**	**83.4**
Step 800	84.8	59.0	63.5	57.8	**87.3**	98.8	75.2

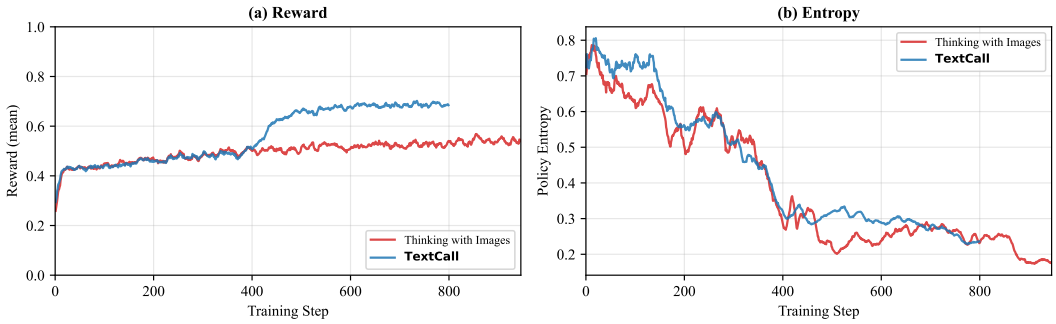


Figure 8: **RL training dynamics from matched GRPO runs.** Curves show 10-step-smoothed training reward and policy entropy from the matched thinking-with-images and **TextCall** runs. These logged metrics are not benchmark accuracy; Table 10 shows that both policies exhibit unstable tool use, with different failure modes.

Why RL is excluded from the main accuracy claim. At the evaluated checkpoints, the two policies no longer realize the intended matched intervention: one often answers directly, while later variants exploit execution failure or omit code. Benchmark accuracy from these policies cannot be interpreted as a clean pixel-return contrast. We therefore use SFT for the carrier-swap result and report RL only as evidence that tool-integrated optimization is brittle under our recipes.

Caveats. These failures do not show that thinking-with-images RL is inherently impossible. They show that our reward and agent loop did not produce a healthy matched pair from which pixel causality could be inferred. Establishing such a pair would require joint reporting of answer accuracy, successful tool execution, commitment rate, and tool-use behavior throughout training.

J Coordinate Precision Within the Code Body

Section 4.2 shows that removing the code body degrades V*Bench by 9.94 pp and CV-2D by 11.12 pp, establishing that the syntactic structure of `.crop()` calls is load-bearing. This leaves open whether the code component works by providing *accurate numeric spatial evidence* or by imposing an *explicit structured region-selection cue* regardless of coordinate accuracy. We test this with a coordinate-precision ablation: matched models trained on correct versus shuffled coordinates.

Design. We render 3,317 verified training samples into matched arms differing only in the spatial evidence within the assistant reasoning: *correct coordinates* (real bounding-box values from verified object proposals), *shuffled coordinates* (values randomly permuted across samples—syntactically valid `.crop()` calls with spatially meaningless arguments), and *free CoT* (no coordinates or spatial markers; descriptive reference only, not matched to the Section 4.2 code-removal ablation). All arms train Qwen3-VL-8B-Instruct (Qwen Team, 2025b) with LoRA rank 8 for 3 epochs and are evaluated under Direct MC think-mode with heuristic scoring on four benchmarks (HR-4K, MMStar, CV-2D, CV-3D). Three seeds (42, 43, 44) are run for the primary correct-vs.-shuffled comparison.

Results. Table 11 reports the cross-seed means. Correct and shuffled coordinates are statistically equivalent: the item-level mean difference across all 14,806 paired predictions (3 seeds $\times$ 4 benchmarks) is -0.11 pp (90% CI $[-0.74, +0.51]$), passing a paired-bootstrap TOST equivalence test at δ=2 pp (p=0.006). The equivalence margin is less than one-third of the 9.94 pp code-removal effect in Section 4.2. At the per-sample level, the two models produce identical predictions on 87.6% of instances; on V*Bench (seed 42, 191 samples), concordance reaches 90.6% with a symmetric discordant split (8 vs. 7, McNemar p=1.0). A single-seed replication on the weaker Qwen2.5-VL-7B (Qwen Team, 2025a) base is consistent with the same pattern (correct minus shuffled: $+0.13$ pp).

Scale validation (9.5K, matched to Section 4.2). The 3K experiment above leaves open whether coordinate correctness might matter at larger data scales where the code-removal effect is established. We therefore repeat the ablation on the full 9,500-sample **TextCall** training set used in Section 4.2,

Table 11: **Coordinate precision ablation** (Qwen3-VL-8B, 3 seeds $\times$ 4 benchmarks, Direct MC think-mode). Mean $\pm$ std across seeds. TOST equivalence at δ=2 pp: p=0.006.

Arm	HR-4K	MMStar	CV-2D	CV-3D	Mean
Correct coordinates	$78.9_{\pm 0.5}$	$65.1_{\pm 1.0}$	$81.3_{\pm 0.6}$	$88.0_{\pm 0.8}$	78.3
Shuffled coordinates	$78.3_{\pm 1.1}$	$66.2_{\pm 0.3}$	$81.2_{\pm 0.4}$	$88.8_{\pm 0.2}$	78.6
Free CoT (no coords)	$78.4_{\pm 0.9}$	$64.6_{\pm 0.5}$	$81.3_{\pm 0.8}$	$87.5_{\pm 0.4}$	77.9
Correct $-$ Shuffled	$+0.6$	-1.1	$+0.1$	-0.7	-0.1

training Qwen2.5-VL-7B with the same LoRA configuration. Coordinates in 2,193 samples containing `.crop()` calls are cross-sample permuted within coordinate type (pixel values shuffled with pixel values, ratios with ratios); the remaining 7,307 samples without `.crop()` are identical across arms. Table 12 shows the same qualitative pattern: the mean difference is -0.65 pp, far smaller than the $\sim$10 pp code-removal effect from Section 4.2.

Table 12: **Coordinate correctness at 9.5K scale** (Qwen2.5-VL-7B, seed 42, Direct MC think-mode, heuristic scoring). Same training recipe as the Section 4.2 models.

Arm	HR-4K	MMStar	CV-2D	CV-3D	Mean
Correct coordinates	58.5	54.5	68.3	75.2	64.1
Shuffled coordinates	61.3	54.5	69.3	73.9	64.8
Correct $-$ Shuffled	-2.8	0.0	-1.0	$+1.3$	-0.7

Scope. The 3K multi-seed test establishes equivalence; the 9.5K matched-scale run shows the same qualitative pattern, arguing against a scale-specific artifact. An important caveat: both experiments operate at the *supervised fine-tuning* stage, where the base model has already acquired spatial understanding from vision–language pre-training. Coordinate precision may well matter during *pre-training*, where grounding data teaches the model what spatial coordinates mean in the first place (e.g., the bounding-box pre-training stages in Qwen2.5-VL (Qwen Team, 2025a) and similar models). Our finding is specific to the post-training regime: once a model already possesses spatial grounding from pre-training, the coordinate values in fine-tuning data serve as a structural cue rather than a source of new spatial knowledge.

Interpretation. Combined with Section 4.2, these results support a two-level decomposition: the *format* of spatial code (having `.crop()` calls at all) matters substantially, while the *numeric content* (whether coordinate values are instance-correct) does not detectably matter. This pattern is consistent with the code body acting as a structured procedural cue that requires the model to specify a region before continuing its reasoning, rather than serving as a numerically accurate spatial reference.