THINK-THEN-VERIFY: DECOUPLING TOOL VERIFICATION FROM VISUAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal large language models (MLLMs) can improve visual understanding and reasoning with external tools such as code interpreters. However, when tool use is tightly coupled with visual reasoning, a model must decide whether and how to use tools while pursuing a multi-step solution. On complex problems, these intertwined decisions can lead to unproductive exploration or code checks with answer-irrelevant hypotheses, reducing the effectiveness of verification and ultimately harming answer quality. We propose Think-Then-Verify (TTV), a phase-separated paradigm that decouples executable verification from answer formation. TTV first elicits tool-free reasoning and a draft answer, then extracts explicit, checkable hypotheses and invokes external tools solely to evaluate them. The resulting evidence determines whether the draft answer is retained or revised. By separating reasoning from verification, TTV makes each tool call accountable to a stated hypothesis rather than to an evolving interleaved trajectory. Across open- and closed-source MLLMs, TTV substantially increases targeted verification coverage and improves visual-reasoning accuracy over the interleaved tool-use baseline. We further construct a supervised training corpus for hypothesis extraction and code-based verification, and our fine-tuned checkpoints improve under TTV and transfer verification-oriented tool use to the interleaved tool-use baseline.

1 INTRODUCTION

Code execution has emerged as a powerful mechanism for augmenting multimodal large language models (MLLMs) on visual understanding and reasoning tasks (OpenAI, 2026; Google DeepMind, 2026; Anthropic, 2026; Qwen Team, 2026b;a). It provides access to precise, executable operations—including pixel and coordinate inspection, geometric measurement, visual counting, and numerical or symbolic computation—that complement a model's parametric visual representations. Accordingly, executable tools offer a promising avenue for test-time scaling in visual reasoning.

However, existing tool-augmented MLLMs typically interleave reasoning and execution within a single generation trajectory. Such an arrangement requires the model to determine what evidence to inspect, synthesize and execute a program, interpret the resulting observation, and update its answer concurrently with the formation of its reasoning. This coupling can render tool use opportunistic: execution may be directed toward claims that are convenient to implement rather than those that are consequential for the answer, while early errors can be reinforced through subsequent interactions. We refer to intermediate claims that admit programmatic tests as *hypotheses*. Our paired analysis reveals that interleaved trajectories frequently fail to preserve these claims: only 29.1% of hypotheses extracted from a tool-free trace reappear in the corresponding interleaved trajectory, and only 17.9% receive targeted verification. Thus, the limitation of interleaving is not solely inefficient execution; it is the failure to explicitly represent and validate the propositions on which an answer relies. This issue is particularly consequential for smaller models, for which planning, code generation, and long-horizon context management are less reliable.

We introduce **Think-Then-Verify (TTV)**, a phase-separated framework that organizes inference as *think–hypothesise–verify*. TTV first elicits a tool-free reasoning trace and draft answer, subsequently extracts an explicit queue of verifiable hypotheses, and invokes external tools exclusively to evaluate these hypotheses. The resulting verdicts are then incorporated to either retain or revise

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.

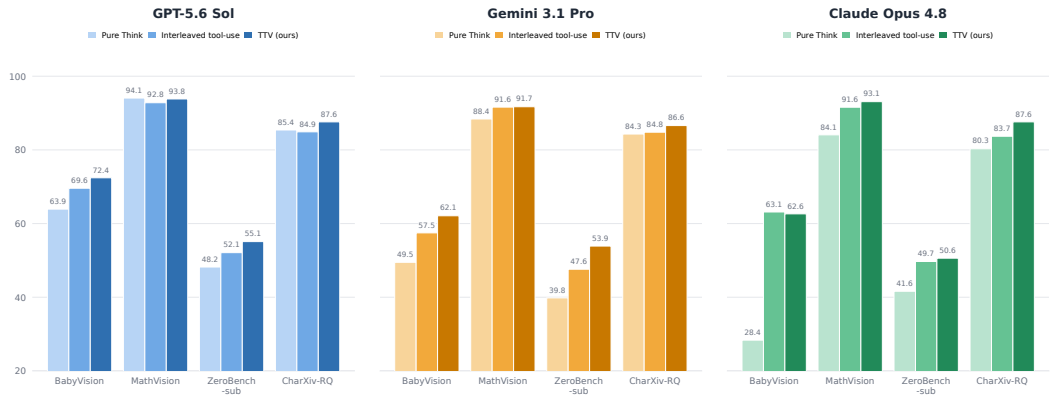

Figure 1: Closed-source model comparison across four visual reasoning benchmarks.

the draft answer. This separation makes each tool call evidence-directed and auditable: under a matched paired protocol, TTV verifies 65.5% of the same hypotheses, compared with 17.9% under the interleaved tool-use baseline, and its completed checks overturn hypotheses more frequently (8.2% versus 4.4%). At matched tool-call budgets, TTV is also generally more accurate and exhibits less degradation as the number of calls increases. These findings indicate that the benefit of TTV arises not simply from invoking tools, but from directing them toward explicitly articulated, answer-relevant hypotheses.

Our contributions are threefold: (i) we introduce **Think-Then-Verify (TTV)**, a phase-separated inference protocol that decouples tool verification from visual reasoning. By formulating explicit, answer-relevant hypotheses before tool invocation, TTV makes verification accountable to the propositions underlying a draft answer; (ii) we construct a supervised training corpus of phase-separated *think–hypothesise–verify* trajectories. The corpus improves performance under TTV and transfers verification-oriented tool-use behavior to the interleaved tool-use baseline; and (iii) across four primary benchmarks, TTV improves the average accuracy of Claude Opus 4.8, Gemini 3.1 Pro, and GPT-5.6 Sol by +14.89, +8.08, and +4.34 points over tool-free *Think*, and by +1.45, +3.21, and +2.39 points over *Interleaved tool-use*, respectively. For Qwen3.5-9B and Qwen3.6-35B-A3B, TTV raises average accuracy by +2.11 and +3.44 points over *Think*, and by +6.86 and +3.33 points over *Interleaved tool-use*, and fine-tuning further improves by 2.70–18.04 points.

2 Related Work

Tool use and visual verification. Tool-augmented agents commonly follow an interleaved *reason–act–observe* loop, alternating between reasoning, action selection, and the incorporation of tool observations (Yao et al., 2023). Code-oriented agents extend this paradigm to executable actions for computation and software engineering (Yang et al., 2024; Wang et al., 2024b;c). In visual domains, Visual ChatGPT, VisProg, ViperGPT, and MM-ReAct invoke programs or external tools for image manipulation, visual inspection, and compositional reasoning (Wu et al., 2023a; Gupta & Kembhavi, 2023; Surís et al., 2023; Yang et al., 2023). Think-with-image approaches likewise use cropping, zooming, visual search, coordinate analysis, or programmatic image processing to acquire and refine evidence during reasoning (Gao et al., 2024; Karthik et al., 2024; Mitra et al., 2024; Wu et al., 2023b). More broadly, reflective agents use feedback to revise an ongoing trajectory (Shinn et al., 2023), while VideoHV-Agent applies clue-guided evidence verification to long-video question answering (Wang et al., 2026). These methods establish the value of executable visual evidence, but typically treat tool outputs, critiques, or visual observations as open-ended refinement signals. In contrast, TTV first externalizes answer-relevant hypotheses that admit programmatic tests, then restricts execution to evaluating those hypotheses before retaining or revising a draft answer.

Control policies. The behavior of an agent depends not only on its base model, but also on its tool schemas, observation interface, context updates, and stopping policy (Yao et al., 2023; Qin et al., 2024; Liu et al., 2024; Yang et al., 2024; Wang et al., 2024b). Existing interleaved policies jointly

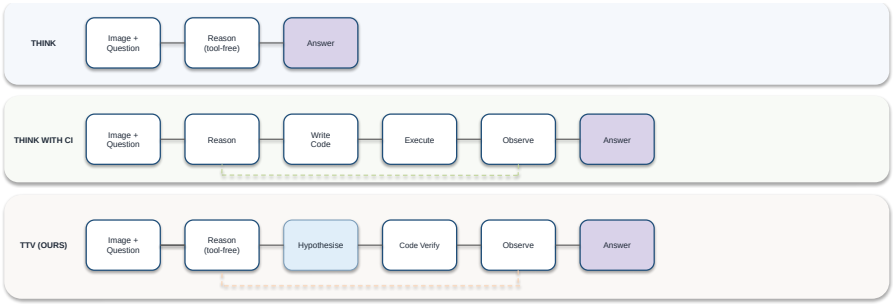

Figure 2: Comparison of inference protocols. *Think* directly produces an answer without tool use. *Interleaved tool-use* repeatedly mixes reasoning, code generation, execution, and response revision in a single trajectory. *TTV* first forms a tool-free draft, externalizes answer-relevant hypotheses, and executes targeted checks whose verdicts determine whether to retain or revise the draft.

decide what to inspect, how to invoke a tool, and how to revise the answer within one trajectory. TTV changes this control policy by separating answer formation and hypothesis selection from program synthesis and execution. Our controlled comparisons consequently hold the model, tool set, execution backend, and observation interface fixed, isolating the effect of phase-separated, verdict-conditioned verification rather than changing the available tools or interfaces.

3 METHOD

3.1 PRELIMINARIES

Think produces an answer in a single tool-free reasoning pass, leaving visual assumptions and intermediate claims implicit. *Interleaved tool-use* instead interleaves reasoning with code generation and execution, allowing the model to inspect tool observations and iteratively revise its response. Although this procedure can ground intermediate claims in executable evidence, it makes tool selection reactive: the model must jointly decide which claims matter, how to verify them, and how to incorporate returned observations within one trajectory. Consequently, answer-critical claims may remain unexamined, while poorly targeted calls can interrupt reasoning or consume computation without improving the final answer. TTV addresses this limitation by separating hypothesis formation from executable verification.

3.2 TTV: PHASE-SEPARATED VERIFICATION

TTV treats verification as an explicit, goal-directed stage rather than a by-product of the *Interleaved tool-use* baseline. Given an input, it first produces a tool-free draft reasoning trace and answer (Figure 2), then extracts concrete, answer-relevant hypotheses: claims that admit programmatic tests. These hypotheses may concern visual evidence, intermediate quantities, spatial or geometric relations, or logical inferences; non-verifiable claims are excluded. An empty hypothesis set is permitted, equivalently the pre-execution state of interleaved tool use, accommodates problems in which code carries out the reasoning itself, rather than testing a separately stated hypothesis. Each retained hypothesis is subsequently translated into a targeted code program for measurement, counting, coordinate or pixel inspection, path tracing, or geometric testing. The resulting observation yields a supporting, contradictory, or inconclusive verdict: supporting evidence preserves the draft, contradictory evidence triggers revision, and inconclusive evidence is not treated as confirmation. By committing to a draft and its hypotheses before execution, TTV decouples the selection of what to verify from the implementation of how to verify it. It therefore supplements tool-free *Think* with an explicit correction mechanism while avoiding the opportunistic calls and feedback entanglement of the interleaved tool-use baseline; verification remains selective and terminates when relevant hypotheses are resolved or the budget is exhausted.

3.3 High-Quality Data Construction

We construct the fine-tuning corpus through a five-step pipeline. (i) *Data collection.* Following MMFineReason (Lin et al., 2026), we collect diverse visual reasoning examples as a candidate pool for subsequent processing. (ii) *Data labeling.* An open-weight annotator, Qwen3.5-397B-A17B (Qwen Team, 2026a), first identifies problems that admit code-based verification, including counting, measurement, visual inspection, path tracing, and geometric matching. It then confirms verifiability from an actual *Interleaved tool-use* execution trace; only samples passing both filters are retained. (iii) *Trajectory collection.* We use the same open-weight annotator, Qwen3.5-397B-A17B, to generate phase-separated TTV trajectories and retain correct rollouts as regular samples. For failed rollouts, we collect a replacement trajectory from a closed-source frontier model only when it reaches the correct answer, yielding hard examples that preserve the same phase-separated format. (iv) *Quality validation.* We retain only trajectories with complete and executable phases, verdicts entailed by execution outputs, hypotheses relevant to the draft answer, and faithful revisions after contradiction. We additionally filter answer leakage and degenerate outputs, then use Gemini 3.1 Pro (Google DeepMind, 2026) as a judge model to assess reasoning coherence, hypothesis specificity, and verification usefulness. (v) *Format conversion.* The surviving trajectories are serialized as multi-turn SFT dialogues, with supervision applied to assistant and tool-call turns under a unified system prompt. The resulting corpus spans ten capability families and 204 subtypes; Appendix A reports its detailed composition and distribution.

4 Experiments

4.1 Experimental setup and evaluation protocol

We evaluate Qwen3.5-9B (Qwen Team, 2026a) and Qwen3.6-35B-A3B (Qwen Team, 2026b), representing dense and mixture-of-experts open-weight MLLMs, together with leading closed-source models where available. We compare tool-free *Think*, the *Interleaved tool-use* baseline, and TTV under matched models, images, prompts, tool interfaces, execution backends. Our TT-9B and TT-35B-A3B are initialized from the Qwen3.5-9B and Qwen3.6-35B-A3B weights, respectively, and trained on the phase-separated *think–hypothesise–verify* corpus from Section 3.3.

Our core evaluation comprises four visual reasoning benchmarks. **BabyVision** (Chen et al., 2026) contains 388 examples designed to assess visual reasoning beyond language priors. **CharXiv-Reasoning-val** (Wang et al., 2024d) contains 1,000 chart-based questions requiring quantitative and compositional reasoning. **ZeroBench-sub** (Roberts et al., 2026) contains 334 visually grounded problems designed to be difficult for contemporary MLLMs. **MathVision** (Wang et al., 2024a) contains 3,040 multimodal mathematical reasoning examples.

4.2 Main results

Closed-source models. Table 1 shows that TTV improves the mean accuracy of every closed-source model relative to both comparison protocols. Averaged over the four benchmarks, TTV improves Claude Opus 4.8 by 14.89 points over pure *Think* and 1.45 points over interleaved tool use; the corresponding gains are 8.08 and 3.21 points for Gemini 3.1 Pro, and 4.34 and 2.39 points for GPT-5.6 Sol. Relative to pure *Think*, 11 of 12 model–benchmark comparisons improve; the sole exception is the negligible 0.25-point decrease of GPT-5.6 Sol on MathVision. Relative to interleaved tool use, gains are especially large for Opus 4.8 on CharXiv-RQ ($+3.90$), Gemini 3.1 Pro on ZeroBench-sub ($+6.29$), and GPT-5.6 Sol on ZeroBench-sub ($+3.00$). These results indicate that TTV benefits models with diverse initial capabilities even when they already have access to the same tools and execution environment.

Open-weight base models. TTV likewise improves both open-weight base models against each reference protocol. For Qwen3.5-9B, it increases mean accuracy by 2.11 points over pure *Think* and 6.86 points over interleaved tool use; the latter improvement is positive on all four benchmarks, ranging from $+2.09$ on ZeroBench-sub to $+9.53$ on BabyVision. For Qwen3.6-35B-A3B, TTV yields mean gains of 3.44 points over pure *Think* and 3.33 points over interleaved tool use. Its advantage over interleaved tool use is again positive across all benchmarks ($+2.50$ to $+4.76$), while comparisons with pure *Think* reveal the expected benchmark dependence of tool-assisted inference:

Table 1: Model results across four visual reasoning benchmarks. Models are grouped by inference protocol, with open- and closed-source models reported together.

Model	Type	BabyVision	MathVision	ZeroBench-sub	CharXiv-RQ
Pure Think					
Qwen3.5-9B	Open	28.09	77.27	31.14	72.80
Qwen3.6-35B-A3B	Open	31.96	81.38	32.03	77.30
Opus 4.8	Closed	28.35	84.08	41.62	80.30
Gemini 3.1 Pro Preview	Closed	49.48	88.39	39.82	84.30
GPT-5.6 Sol	Closed	63.92	94.07	48.20	85.40
Interleaved tool-use					
Qwen3.5-9B	Open	24.23	68.51	31.44	66.10
Qwen3.6-35B-A3B	Open	39.69	79.08	30.84	73.50
Opus 4.8	Closed	63.14	91.58	49.70	83.70
Gemini 3.1 Pro Preview	Closed	57.47	91.61	47.60	84.80
GPT-5.6 Sol	Closed	69.59	92.80	52.10	84.90
Think–Then–Verify (ours)					
Qwen3.5-9B	Open	33.76	77.24	33.53	73.20
Qwen3.6-35B-A3B	Open	42.53	81.58	35.60	76.70
TT-9B (final)	Open, SFT	45.10	81.71	39.22	75.90
TT-35B-A3B (final)	Open, SFT	60.57	88.65	41.02	82.10
Opus 4.8	Closed	62.63	93.09	50.60	87.60
Gemini 3.1 Pro Preview	Closed	62.11	91.71	53.89	86.60
GPT-5.6 Sol	Closed	72.42	93.82	55.10	87.60

substantial gains on BabyVision ($+10.57$) and ZeroBench-sub ($+3.57$), near parity on MathVision ($+0.20$), and a modest decrease on CharXiv-RQ (-0.60). Thus, phase separation provides a more dependable tool-use policy than interleaving, including for models with limited agentic tool-use capability.

Our Fine-tuned models. Phase-separated fine-tuning further improves both open checkpoints under TTV on all four benchmarks. TT-9B exceeds its Qwen3.5-9B TTV counterpart by 11.34 points on BabyVision, 4.47 on MathVision, 5.69 on ZeroBench-sub, and 2.70 on CharXiv-RQ; TT-35B-A3B yields corresponding gains of 18.04, 7.07, 5.42, and 5.40 points. Despite using only 9B and 35B-A3B open-weight backbones, respectively, TT-9B and TT-35B-A3B attain performance competitive with substantially larger closed-source systems: TT-35B-A3B reaches 60.57 on BabyVision and 88.65 on MathVision, surpassing several closed-source pure-*Think* results, while both checkpoints remain competitive on the other benchmarks. Thus, phase-separated supervision substantially narrows the open–closed gap at modest model scales rather than merely improving performance within a fixed open-model setting.

4.3 Ablations

4.3.1 TTV Increases Verification Coverage

We isolate the effect of phase separation on verification coverage. To eliminate hypothesis selection as a confound, we extract each question's programmatically verifiable hypotheses *once* from its tool-free reasoning trace, then evaluate this identical set under TTV and the interleaved tool-use baseline (`with_tool`), while holding the prompts, images, execution backend, and budget fixed. Our primary metric is the fraction of hypotheses receiving targeted verification; we report paired Wilcoxon signed-rank tests and bootstrap confidence intervals for the per-question differences. We randomly sample 900 questions from the four primary benchmarks and evaluate Opus 4.8, GPT-5.6 Sol, and Gemini 3.1 Pro, yielding 3,071 hypotheses. Re-running extraction reproduces the same hypothesis queue in 99.6% of cases.

Figure 3 shows a substantial coverage advantage for phase separation. Pooled across all evaluated cells, TTV targets 65.5% of hypotheses, compared with 17.9% for interleaved tool use. The pattern holds for every model: Opus 4.8 attains 72.0% versus 34.8% ($\Delta = +37.2$; 95% CI $[32.5, 41.8]$),

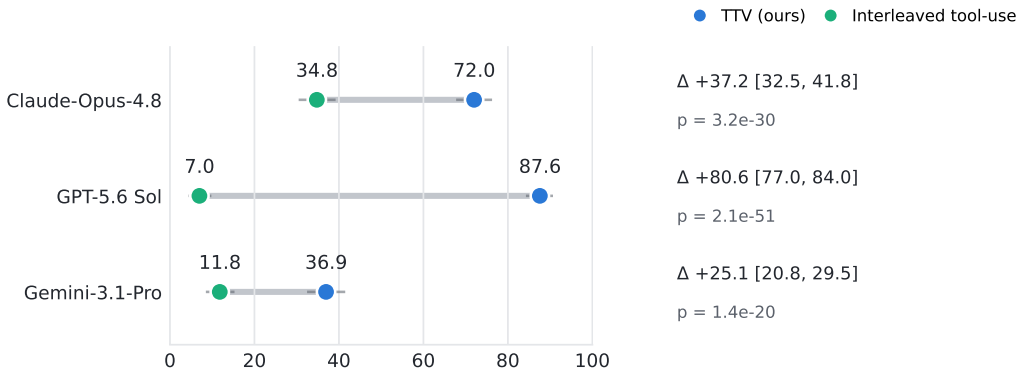

Figure 3: Phase separation substantially improves targeted verification coverage. Results are pooled across the evaluated benchmark subsets for each model. Bars show 95% bootstrap confidence intervals; annotations report paired differences and Wilcoxon p-values. TTV achieves higher coverage for every model.

GPT-5.6 Sol 87.6% versus 7.0% ($\Delta = +80.6$; $[77.0, 84.0]$), and Gemini 3.1 Pro 36.9% versus 11.8% ($\Delta = +25.1$; $[20.8, 29.5]$). All nine model–benchmark cells favor TTV ($p < 10^{-6}$), and pooled per-model tests yield $p \leq 3.2 \times 10^{-30}$. The interleaved trajectories rarely make the hypotheses available for targeted checking: only 29.1% of hypotheses extracted from the tool-free trace appear anywhere in the corresponding trajectory, 8.6% appear in its reasoning segment, and only 15% of traces contain a distinguishable pure-reasoning segment. Thus, interleaving does not simply postpone verification; it often fails to preserve the propositions that would guide it. Moreover, the executed checks are consequential: they overturn 8.2% of TTV hypotheses, compared with 4.4% under the baseline.

4.3.2 TTV Improves Tool Effectiveness at Matched Effort

The aggregate gains in Table 1 may arise either from more tool interaction or from using a comparable amount of interaction more effectively. We distinguish these explanations by conditioning accuracy on the number of tool calls. Figure 4 compares TTV with the interleaved tool-use baseline under matched models, prompts, images, tool interfaces, execution backends, and inference budgets, across all four primary benchmarks. Thus, each panel assesses whether phase separation improves the use of a fixed level of tool interaction.

TTV generally outperforms interleaved tool use within matched call buckets. For GPT-5.6, in the 3–4-call bucket, TTV improves accuracy from 79.4% to 88.5% on MathVision, from 61.5% to 77.7% on BabyVision, from 34.8% to 65.8% on ZeroBench-sub, and from 60.0% to 90.5% on CharXiv-RQ. The same qualitative pattern holds across Gemini and Opus, especially beyond the smallest call bucket. Interleaved trajectories are more concentrated in the 1–2-call bucket and often deteriorate as the number of calls increases, whereas TTV maintains stronger accuracy at comparable effort. Some sparse high-call buckets favor the baseline, so these cells should be interpreted cautiously. Overall, the analysis supports a difference in tool-use effectiveness: phase separation improves the selection and interpretation of executable evidence, rather than merely increasing the volume of tool interaction.

4.3.3 TTV Transfers Verification-Oriented Tool Use

We further examine what phase-separated SFT contributes beyond the *think–hypothesise–verify* format used during training. We compare each fine-tuned checkpoint with its corresponding base model under tool-free `Think`, the `Interleaved tool-use` baseline, and `TTV`, while holding prompts, images, execution backend, tool interface, and inference budget fixed. Improvements under tool-free `Think` would indicate broader visual-reasoning gains, whereas improvements confined to TTV could result from matching the training and inference formats. Performance gains under the

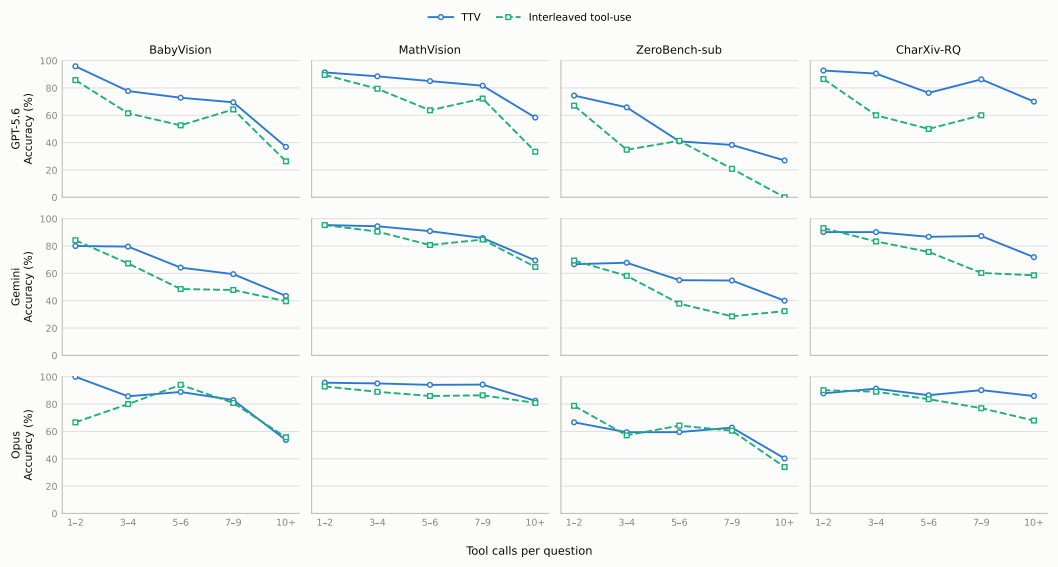

Figure 4: Phase separation improves accuracy at matched tool-use effort. TTV (blue solid) and the *Interleaved tool-use* baseline (green dashed) are evaluated across GPT-5.6, Gemini, and Opus (rows), and BabyVision, MathVision, ZeroBench-sub, and CharXiv-RQ (columns). Points denote mean per-example accuracy within tool-call buckets of 1–2, 3–4, 5–6, 7–9, and 10+ calls. TTV generally attains higher accuracy at the same call budget; sparse high-call buckets should be interpreted cautiously.

Table 2: Protocol-transfer analysis of phase-separated SFT. Each fine-tuned checkpoint is evaluated under tool-free *Think*, the interleaved tool-use baseline, and TTV, with all other conditions held fixed. The corresponding base-model results are reported in Table 1.

Model	Framework	BabyVision	MathVision	ZeroBench-sub	CharXiv-RQ
TT-9B (SFT)	Think	29.64	72.93	32.93	73.60
	Interleaved tool-use	41.75	75.69	32.63	73.00
	TTV	45.10	81.71	39.22	75.90
TT-35B-A3B (SFT)	Think	35.57	79.28	32.63	78.10
	Interleaved tool-use	54.12	81.00	43.41	81.00
	TTV	60.57	88.65	41.02	82.10

format-mismatched interleaved tool-use baseline instead test whether phase-separated SFT transfers verification-oriented tool-use behavior across control policies.

Table 2 indicates that the effect is primarily tool-use specific rather than a broad improvement in tool-free visual reasoning. Under `Think`, fine-tuning changes average accuracy by only -0.05 points for TT-9B and $+0.73$ points for TT-35B-A3B across the four benchmarks. Under TTV, the same checkpoints improve by $+6.05$ and $+8.98$ points, respectively.

Crucially, the improvements extend to the format-mismatched interleaved tool-use baseline. TT-9B improves by $+8.20$ points on average relative to Qwen3.5-9B, with gains on every benchmark ranging from $+1.19$ on ZeroBench-sub to $+17.52$ on BabyVision. TT-35B-A3B similarly improves by $+9.11$ points on average relative to Qwen3.6-35B-A3B, ranging from $+1.92$ on MathVision to $+14.43$ on BabyVision. Phase-separated SFT also changes the effect of interleaving itself: while interleaved tool use lowers average accuracy relative to `Think` for both base models (-4.48 and -0.99 points), it improves the fine-tuned checkpoints by $+3.15$ and $+8.99$ points, respectively. These results show that the corpus does not merely teach a TTV-specific serialization; it equips both model scales to use executable evidence more effectively under an interleaved control policy.

Table 3: Thinking capability comparison beyond the four main visual reasoning benchmarks. We compare tool-free thinking, the interleaved tool-use baseline, and TTV for Qwen3.5-9B and Qwen3.6-35B-A3B. Scores are accuracy percentages.

		General reasoning			Math reasoning				Knowledge	
Model	Framework	LogicVista	MMStar	MMBench	MathVerse-mini	WeMath	DynaMath	MathVista-mini	MMMU	MMMU_Pro
Qwen3.5-9B	Think	73.15	79.27	89.40	86.35	82.95	81.80	84.80	77.89	71.50
	Interleaved tool-use	65.32	74.50	84.80	81.80	71.71	81.38	81.60	77.00	66.90
	TTV	73.60	79.80	87.80	86.83	83.14	82.48	84.70	77.10	69.80
Qwen3.6-35B-A3B	Think	77.18	78.93	89.78	88.40	84.48	82.06	86.50	81.77	75.14
	Interleaved tool-use	74.94	78.07	88.70	87.89	82.67	84.29	85.70	79.56	73.64
	TTV	77.63	79.60	90.48	89.75	83.81	84.51	85.80	80.00	74.86

4.3.4 TTV Transfers Beyond the Primary Benchmark Families

To assess transfer beyond the visually grounded and mathematical settings of the primary suite, we evaluate Qwen3.5-9B and Qwen3.6-35B-A3B on nine independently constructed multimodal reasoning benchmarks under tool-free `Think`, the `Interleaved tool-use` baseline, and `TTV`. Table 3 reports accuracy.

TTV matches or exceeds the interleaved tool-use baseline in all 18 model–benchmark pairs. The gains extend beyond the primary suite: on LogicVista and MMStar, TTV improves Qwen3.5-9B by $+8.28$ and $+5.30$ points, while Qwen3.6-35B-A3B improves by $+2.69$ on LogicVista and $+1.78$ on MMBench. Both models also improve on MathVerse-mini ($+5.03$ and $+1.86$) and WeMath ($+11.43$ and $+1.14$). Comparisons with already strong tool-free `Think` remain mixed, particularly on knowledge-heavy MMMU and MMMU_Pro. Thus, phase separation reliably mitigates the failure modes of interleaved tool use when actionable verification is available, rather than universally replacing tool-free reasoning.

5 Discussion and Future Work

TTV is an inference-time control policy, not a mechanism that improves a model's underlying visual reasoning in every setting. Across our broader evaluation, its most consistent benefit is to avoid the degradation induced by interleaved tool use; relative to an already strong tool-free *Think* trace, its effect remains task-dependent. TTV is consequently most appropriate when a draft answer gives rise to answer-relevant hypotheses that admit targeted executable tests. It is less likely to help when relevant evidence is difficult to formalize, available tools are unreliable, or broad knowledge rather than verification is the dominant source of error.

An important direction is to make verification more selective: a model must decide whether a hypothesis merits a tool call, which test is most informative, and when the available evidence is sufficient to stop. Progress also requires more reliable hypothesis extraction, execution, and evidence-grounded revision, since failures at any of these interfaces can convert tool access into noise. Although phase-separated SFT transfers verification-oriented behavior beyond the training serialization, supervised imitation is constrained by demonstration coverage and quality; outcome-supervised or reinforcement-learning objectives could instead optimize hypothesis selection, stopping, and revision directly. Finally, evaluation should characterize tool contribution in addition to final accuracy, including hypothesis coverage, verification precision, evidence quality, tool-induced error, latency, and cost. Such measurements can clarify when tools provide decisive evidence and guide more reliable multimodal and agentic reasoning systems.

6 Conclusion

We presented Think-Then-Verify (TTV), a phase-separated inference protocol for visual reasoning with executable tools. TTV first forms a tool-free draft answer, extracts answer-relevant hypotheses that admit programmatic tests, and then uses the resulting evidence to retain or revise the draft. This separation makes tool calls accountable to explicit hypotheses rather than to an evolving interleaved trajectory. Across open- and closed-source models, TTV improves targeted verification coverage and generally outperforms interleaved tool use, including at matched tool-call effort. Phase-separated SFT further improves the open-weight checkpoints under TTV and trans-

fers verification-oriented behavior to the interleaved tool-use baseline. These findings show that the effectiveness of tool-augmented visual reasoning depends not only on tool access, but also on how reasoning, verification, and revision are organized.

AI USE STATEMENT

We used generative AI tools for language editing, proofreading, and limited brainstorming during manuscript preparation. We did not use generative AI tools to generate experimental results, conduct evaluations, make scientific claims without author verification, or determine authorship; these tasks were performed and reviewed by the authors. All AI-assisted text and suggestions were reviewed and revised by the authors, who take full responsibility for the final content of this work, including its text, claims, and artifacts.

ETHICS STATEMENT

This work does not involve human participants, personally identifiable information, private user data, medical data, or vulnerable populations. Our experiments use publicly available or internally permitted benchmark data and model APIs with sandboxed code execution. The principal risk is that erroneous or selectively interpreted tool outputs may create unwarranted confidence in an answer. We therefore treat tool outputs as evidence rather than ground truth and manually verify the reported results and conclusions. Any future release will exclude sensitive prompts, trajectories, credentials, and artifacts, and will respect the licenses and access conditions of the evaluated models and benchmarks.

REPRODUCIBILITY STATEMENT

Sections 3 and 4 specify the TTV protocol, phase structure, hypothesis extraction and verification procedure, evaluation settings, and matched comparison conditions. Section 3.3 describes the construction of the phase-separated supervision data, and Appendix A documents its capability-family and subtype composition. The main text reports the evaluated model identifiers, inference protocols, and benchmark conditions; for proprietary models and restricted benchmarks, we will document access dates, inference settings, and execution environments to the extent permitted by their terms. We intend to release the TTV implementation, prompts, evaluation configurations, and non-sensitive derived artifacts, subject to the licenses and access conditions of the underlying models and benchmarks.

REFERENCES

Anthropic. Introducing claude opus 4.8, 2026. URL https://www.anthropic.com/news/claude-opus-4-8.

Liang Chen, Weichu Xie, Yiyan Liang, Hongfeng He, Hans Zhao, Zhibo Yang, et al. Babyvision: Visual reasoning beyond language. *arXiv preprint arXiv:2601.06521*, 2026.

Difei Gao et al. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *Advances in Neural Information Processing Systems*, 2024.

Google DeepMind. Gemini 3.1 pro, 2026. URL https://deepmind.google/models/gemini/pro/.

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

Shyamgopal Karthik, Karsten Roth, Massimiliano Mancini, and Zeynep Akata. Image-of-thought: Towards visual reasoning with image operations. In *International Conference on Learning Representations (ICLR)*, 2024.

Honglin Lin, Zheng Liu, Yun Zhu, Chonghan Qin, Juekai Lin, Xiaoran Shang, Conghui He, Wentao Zhang, and Lijun Wu. Mmfinereason: Closing the multimodal reasoning gap via open data-centric methods. *arXiv preprint arXiv:2601.21821*, 2026.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Yuxian Gu, Zhengxiao Du, Peng Zhang, et al. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Chanju Mitra et al. Visual chain-of-thought. In *Advances in Neural Information Processing Systems*, 2024.

OpenAI. Introducing gpt-5.6 sol, 2026. URL https://openai.com/index/introducing-gpt-5-6-sol/.

Yujia Qin, Shengding Liang, Yining Ye, Kunlun Zhu, Linyang Yan, Yaxi Lin, Xin Cong, Xiangru Tang, Bill Yuchen Qian, Sihan Zhao, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations (ICLR)*, 2024.

Qwen Team. Qwen3.5: Toward native multimodal agents, 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.6-35b-a3b: Agentic coding power, now open to all, 2026b. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

Jonathan Roberts, Mohammad Reza Taesiri, Ansh Sharma, Akash Gupta, Samuel Roberts, Ioana Croitoru, et al. Zerobench: An impossible visual benchmark for contemporary large multimodal models. In *International Conference on Machine Learning (ICML)*, 2026.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *arXiv preprint arXiv:2303.11366*, 2023.

Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Mingjie Zhan, and Hongsheng Li. Measuring multimodal mathematical reasoning with math-vision dataset. *arXiv preprint arXiv:2402.14804*, 2024a.

Xingyao Wang et al. Openhands: An open platform for ai software developers as generalist agents. *arXiv preprint arXiv:2407.16741*, 2024b.

Yining Wang et al. Executable code actions elicit better llm agents. *arXiv preprint arXiv:2402.01030*, 2024c.

Zheng Wang, Haoran Chen, Haoxuan Qin, Zhipeng Wei, Tianwen Qian, and Cong Bai. Think, then verify: A hypothesis–verification multi-agent framework for long video understanding. *arXiv preprint arXiv:2603.04977*, 2026.

Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, et al. Charxiv: Charting gaps in realistic chart understanding in multimodal llms. *arXiv preprint arXiv:2406.18521*, 2024d.

Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Ze Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023a.

Hao Wu et al. V*: Guided visual search as a core mechanism in multimodal llms. *arXiv preprint arXiv:2312.14135*, 2023b.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *arXiv preprint arXiv:2405.15793*, 2024.

Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Ce Liu, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023.

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*, 2023.

A Fine-tuning Corpus Composition

This appendix documents the composition of the phase-separated SFT corpus and the closed-source evaluation configuration. All main experimental results and ablations use the evaluation protocol described in Section 4.

Figures 5 and 6 characterize the corpus produced by the pipeline in Section 3.3. The family distribution summarizes the verification strategies represented in the training data, while the subtype breakdown resolves the visual operations within each family. Together, these figures make the coverage and long-tailed composition of the phase-separated supervision data explicit.

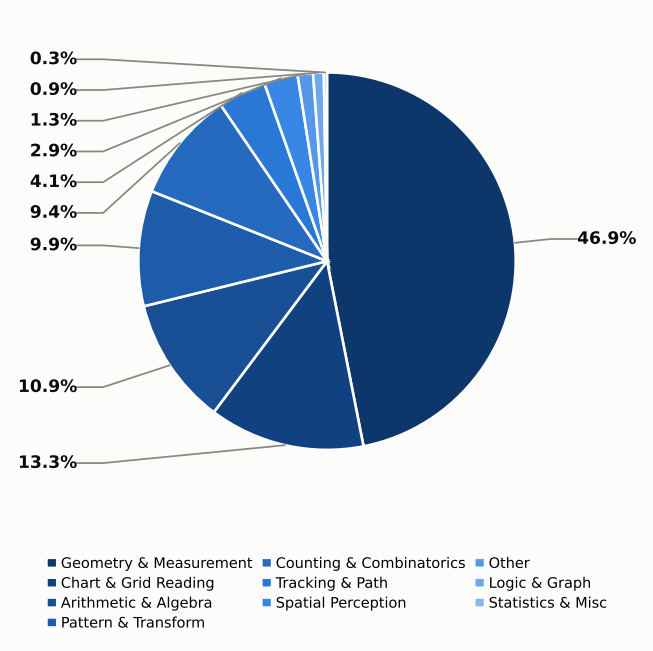

Figure 5: Capability-family composition of the fine-tuning corpus (87,550 samples). All ten families are labeled, with proportions placed outside the pie and category names listed below.

B Closed-Source Evaluation Configuration

We evaluate GPT-5.6 Sol, Gemini 3.1 Pro Preview, and Claude Opus 4.8 through their respective provider APIs, using the same images, prompts, tool interface, execution backend, and inference budget for *Think*, *Interleaved tool-use*, and TTV. We request the provider's highest available reasoning setting ("xhigh" where supported), use a maximum of 65,536 output tokens per LLM call, and clamp the effective context to 81,920 tokens, including prompt and generation. Each problem permits at most 20 tool-call iterations, with a 2,400-second wall-clock limit and a 300-second code-cell timeout.

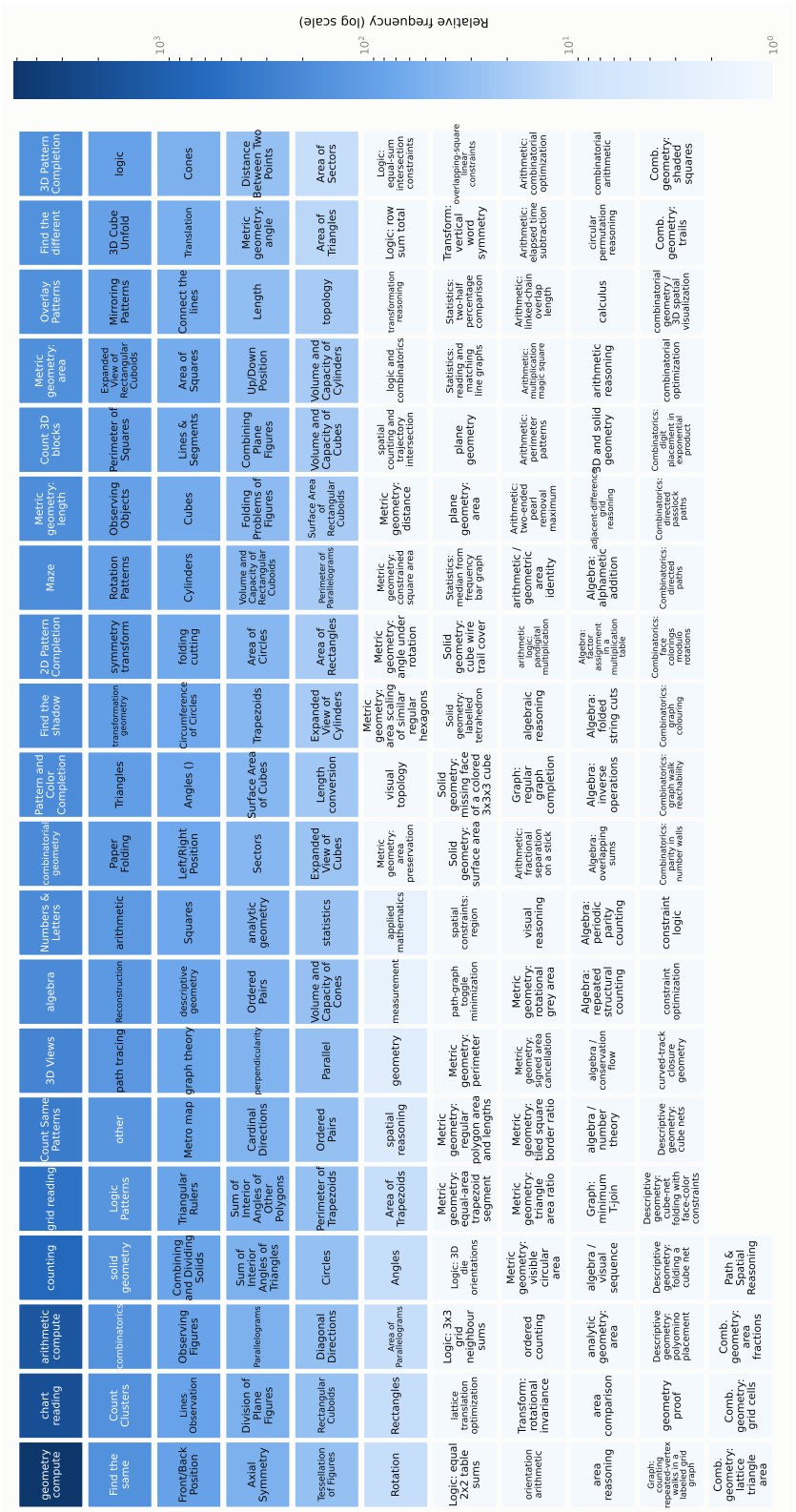

Figure 6: Subtype frequency heatmap of the fine-tuning corpus, shown on a log color scale. All 204 subtypes are displayed, including the 99 rare subtypes that contribute 106 samples in total. The figure is set in landscape to preserve label legibility.