MUST: Stage-Conditioned Stability Control for Test-Time Scaling in Multimodal Reasoning

Yunfeng Deng
South China Normal University
202438012070@m.scnu.edu.cn

Abstract

Test-time scaling (TTS) improves reasoning by allocating additional inference compute, but in multimodal large language models (MLLMs), more compute is not always more reliable. Longer reasoning may amplify visual misperception, induce answer drift, reinforce majority traps, or turn initially correct predictions into overthought failures. We propose MUST, a training-free framework that recasts multimodal TTS as stage-conditioned stability control. MUST separates inference into reasoning and verification stages and assigns each stage a different reliability criterion. During reasoning, *Contrastive Answer-Manifold Stability* (CAMS) selects answers whose supporting trajectories form compact and separable latent groups. During verification, *Attention-Free Support-Chain Stability* (AF-SCS) evaluates whether explicit visual evidence and subsequent reasoning form a stable latent support chain, without relying on token-to-patch attention maps. Experiments on multimodal reasoning benchmarks show that MUST improves the accuracy–compute trade-off and reduces overthinking-induced errors. These results suggest that reliable MLLM test-time scaling should adaptively control when and how extra compute is trusted, rather than simply increasing inference budget.

1 Introduction

Recent progress in reasoning-oriented inference has made test-time scaling (TTS) a central topic in large language model research. In language-only settings, methods such as self-consistency, iterative self-refinement, and budget-aware search show that additional test-time compute can improve accuracy without updating model parameters [Wang et al., 2022, Madaan et al., 2023, Snell et al., 2024, Muennighoff et al., 2025]. Multimodal large language models (MLLMs), however, make this promise less straightforward. Their answers depend jointly on visual perception and textual reasoning, so longer inference may produce more elaborate explanations without making the answer more reliable. Worse, additional reasoning can amplify an initial visual misperception, induce answer drift, reinforce a frequent but wrong answer, or turn an initially correct prediction into an overthought failure [Liu et al., 2025, Bugaud, 2026, Deng et al., 2025a].

This paper studies why training-free TTS remains unstable in multimodal reasoning. Existing inference-time scaling methods often spend extra compute by sampling more trajectories, extending the reasoning budget, refining previous answers, or aggregating final predictions. These strategies are useful, but they implicitly assume that additional computation provides more reliable evidence. In MLLMs, this assumption can break down: multiple samples may share the same visual misunderstanding [Vo et al., 2025], refinement may revise a correct answer into a wrong one [Wu et al., 2025, Huang et al., 2023], and answer frequency may reflect correlated errors rather than correctness [Bugaud, 2026]. Therefore, before designing a new scaling strategy, we first ask what kinds of failures occur when standard TTS recipes are applied to multimodal reasoning.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Our diagnostic experiments in Section 2 reveal three consistent patterns. First, increasing test-time compute is not monotonically beneficial: larger budgets can change answers and introduce non-trivial correct-to-wrong transitions. Second, common answer-selection signals such as majority vote, entropy, and self-evaluation are not sufficiently reliable on their own. Third, examples differ substantially in how much compute they need: some are already stable under a small budget, while others require additional verification, and some become worse when forced through a larger budget. These findings suggest that multimodal TTS should not be treated as a fixed-budget aggregation problem. Instead, it should be treated as a reliability-control problem: the inference procedure should decide when an answer is stable enough to stop and when further verification is necessary.

Motivated by this view, we propose MUST, a training-free framework for stage-conditioned stability control. The key idea is to separate the roles of inference rather than applying one fixed aggregation rule throughout the whole process. MUST first explores plausible candidate answers, then checks whether the trajectories supporting each candidate form stable and separable latent answer groups, and finally invokes explicit visual-evidence verification only for unstable cases. This design allows the controller to spend additional compute selectively, instead of forcing all examples through the same maximum budget.

To implement this idea, MUST combines two complementary stability criteria. For the reasoning stages, we introduce *Contrastive Answer-Manifold Stability* (CAMS), which measures whether trajectories supporting the same answer are internally coherent and separated from competing answer groups. This helps distinguish reliable agreement from majority traps caused by repeated errors. For the verification stage, we introduce *Attention-Free Support-Chain Stability* (AF-SCS), which estimates whether the model's explicitly generated visual-evidence span and subsequent reasoning span form a stable latent support chain. AF-SCS does not require token-to-patch attention maps, making it compatible with efficient FlashAttention-style inference where such maps may be unavailable or unreliable. [Dao et al., 2022]

Our main contributions are as follows:

- We provide a diagnostic analysis of training-free multimodal test-time scaling and show that standard scaling recipes suffer from budget-induced answer drift, unreliable single-signal selection, and strong instance-level compute heterogeneity.
- We formulate multimodal TTS as a reliability-control problem rather than a fixed-budget aggregation problem, motivating adaptive stopping and selective verification based on answer stability.
- We propose *Contrastive Answer-Manifold Stability* (CAMS) for reasoning-stage selection, which evaluates whether answer-supporting trajectories form compact and separable latent answer groups.
- We propose *Attention-Free Support-Chain Stability* (AF-SCS) for verification-stage selection, which estimates evidence–reasoning support consistency.
- We instantiate these ideas in MUST, a training-free stage-conditioned controller that improves the accuracy–compute trade-off and reduces overthinking-induced errors across multimodal reasoning benchmarks.

2 Diagnostic Experiments on Existing MLLM Test-Time Scaling

The introduction argues that multimodal TTS should be viewed as a reliability control problem. We now provide empirical evidence for this view by diagnosing where standard training-free scaling strategies fail. Rather than evaluating MUST, this section studies representative existing inference recipes on the same Qwen2.5-VL-7B backbone and analyzes how their predictions change as test-time compute increases.

2.1 Diagnostic setup

We instantiate representative training-free baselines on the same Qwen2.5-VL-7B backbone, including direct decoding, long reasoning, self-consistency, and iterative self-refinement. Unless otherwise noted, the diagnostic experiments in this section are conducted on MathVista. For each strategy, we progressively increase test-time compute and track answer trajectories, correctness transitions,

Table 1: Answer drift across test-time scaling dimensions for Qwen2.5-VL-7B on MathVista. Extra compute induces substantial answer changes across token budget, refinement rounds, and sample count; several transitions exhibit non-trivial correct→wrong flips, indicating that more budget is not uniformly beneficial.

Setting type	Transition	Change rate	Correct→Wrong	Wrong→Correct	Avg. entropy Δ
Token budget	$128 \rightarrow 256$	46.7%	7.9%	23.1%	-0.0073
Token budget	$256 \rightarrow 512$	32.3%	6.7%	13.5%	$+0.0105$
Token budget	$512 \rightarrow 1024$	31.1%	10.1%	17.9%	-0.0139
Refine rounds	$0 \rightarrow 1$	37.0%	17.4%	7.8%	-0.0105
Refine rounds	$1 \rightarrow 2$	33.6%	9.3%	6.6%	-0.0042
Refine rounds	$2 \rightarrow 3$	31.8%	16.6%	12.1%	$+0.0064$
Number of samples	$1 \rightarrow 3$	13.8%	7.4%	4.2%	$+0.0026$
Number of samples	$3 \rightarrow 5$	12.1%	3.6%	7.7%	$+0.0007$
Number of samples	$5 \rightarrow 7$	6.5%	2.4%	2.7%	$+0.0016$

Table 2: Existing answer-selection signals for Qwen2.5-VL-7B on MathVista are insufficient for predicting whether the selected answer is correct.

Selection / confidence signal	Acc.	AUROC	ECE
Majority vote	68.9%	0.7916	0.180
Vote margin	67.1%	0.7865	0.176
Entropy-only	65.8%	0.7439	0.221
Self-evaluation confidence	66.4%	0.7598	0.195
Refinement agreement	67.4%	0.7715	0.181

entropy, vote distribution, and compute usage. We focus on three questions: **(Q1)** Is extra compute always helpful? **(Q2)** Are common single-signal answer-selection rules reliable? **(Q3)** Is a fixed maximum-budget policy suitable for all examples?

2.2 Finding I: More compute does not monotonically improve accuracy

Table 1 quantifies answer drift across token budget, refinement depth, and sample count for Qwen2.5-VL-7B on MathVista. Larger budgets clearly change predictions, but the changes are not monotonically beneficial: non-trivial *correct→wrong* transitions persist across all three axes.

Token-budget scaling produces the largest instability, and refinement is the clearest source of overthinking. Increasing sample count changes answers at rates of 13.8%, 12.1%, and 6.5% across the three transitions, respectively. These rates are lower than those for token-budget and refinement scaling at each listed step, but sampling is still not uniformly helpful, as correct-to-wrong transitions persist. The entropy shift is also small and inconsistent across settings. Overall, these results show that multimodal TTS should be controlled by explicit stability criteria rather than a naive "more compute is always better" policy.

2.3 Finding II: Existing single-signal answer selection is unreliable

The second failure mode concerns answer selection. Majority vote, vote margin, entropy, and self-evaluation each capture only one aspect of reliability and can be misled by correlated errors or over-confidence. Table 2 shows that, on Qwen2.5-VL-7B evaluated on MathVista, none of these single-signal criteria is sufficiently reliable on its own, which motivates combining complementary evidence from support, consistency, persistence, and confidence.

Majority vote achieves the highest accuracy (68.9%) and AUROC (0.7916), while vote margin has the lowest ECE (0.176). Entropy-only performs worst in accuracy (65.8%) and AUROC (0.7439) and has the highest ECE (0.221), illustrating the trade-offs among individual selection signals and motivating their combination.

Table 3: Oracle analysis of fixed-budget inference. We force each sample to run through the full budget ladder and assess its trajectory against the listed, potentially overlapping criteria.

Benchmark	Low-budget stable	High-budget beneficial	Overthinking-prone	Oscillating	Unresolved
MMMU	38.4%	21.7%	8.4%	9.6%	25.9%
MathVista	45.4%	20.4%	11.6%	6.8%	21.8%
MMBench	54.9%	15.8%	12.9%	8.7%	19.7%
HallusionBench	45.1%	19.5%	9.1%	19.3%	21.0%

Table 4: Stage-conditioned design of MUST. Each stage has a different inference role and therefore uses a different set of active reliability signals.

Stage	Role	Active signals	Main purpose
B_1	Candidate exploration	V, Q, CAMS	Build a low-cost candidate pool
B_2	Reasoning stability assessment	V, Q, CAMS, P	Test cross-budget answer stability
B_3	Evidence-based verification	V, Q, AF-SCS, P	Verify unstable cases with explicit visual evidence

2.4 Finding III: Fixed-budget inference is inefficient and risky

The third failure mode concerns compute allocation. To test whether fixed-budget inference is appropriate, we force each example to run through the full budget ladder and assess its trajectory against the listed criteria. Table 3 shows strong instance-level heterogeneity: many examples are already stable under a low budget, some benefit from additional compute, and a non-trivial subset is overthinking-prone and becomes worse after further inference. These categories are not mutually exclusive: one example may satisfy multiple trajectory criteria, so percentages within a benchmark need not sum to 100%. Fixed maximum-budget policies are therefore both inefficient and risky, motivating adaptive stopping based on answer stability.

Together, these findings define the requirements for our method: it should avoid blindly increasing compute, combine complementary reliability evidence rather than relying on a single signal, and allocate verification only to examples whose answers remain unstable. The next section instantiates these requirements in MUST.

3 Method: MUST as Stage-Conditioned Stability Control

3.1 Overview

Given an image-question pair $x = (I, q)$, MUST performs training-free test-time inference through a monotone stage schedule $\mathcal{B} = \{B_1, B_2, B_3\}$. The method is designed around the diagnostic requirements in Section 2: avoid blindly increasing compute, go beyond single-signal answer selection, and allocate verification only to examples whose answers remain unstable.

Table 4 summarizes the three stages and their active signals. The key design is that MUST does not use the same reliability criterion throughout inference. In the reasoning stages B_1 and B_2, the controller uses CAMS to evaluate whether trajectories supporting the same answer form a coherent and separable latent answer group. In the verification stage B_3, the controller switches to AF-SCS, which evaluates whether the explicitly generated visual evidence and the subsequent reasoning form a stable support chain.

3.2 Answer Candidate Accumulation

Each sampled trajectory ends with an explicit final-answer span. We extract this span from the model response and normalize it into a short answer string. Since later stages are intended to re-evaluate earlier hypotheses rather than restart from scratch, MUST accumulates answer candidates across stages.

For open-ended tasks with long free-form answers, exact string normalization does not reliably identify semantically equivalent responses. A model-based judge can instead compare each answer span with existing answer clusters in the context of the question, assign semantically consistent responses to the same cluster, and use each cluster as a candidate answer in the definitions below. The

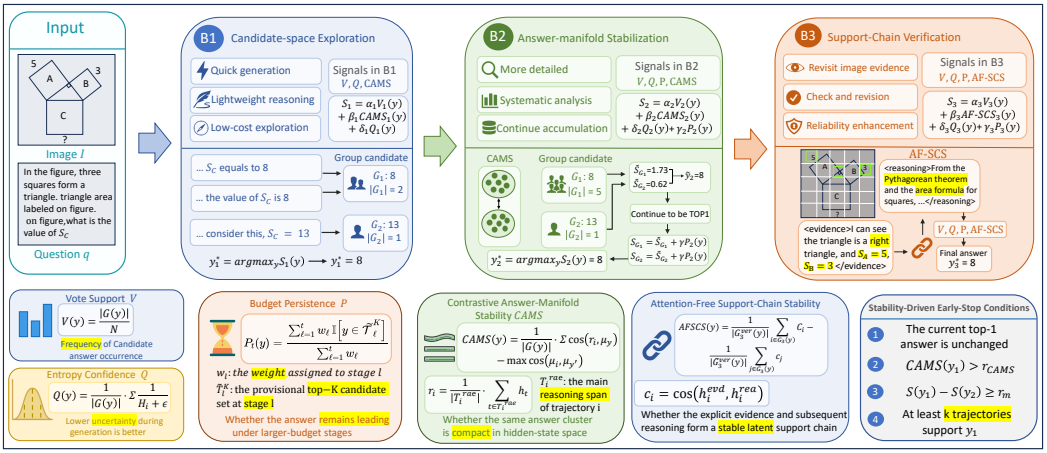


Figure 1: Overview of MUST. B_1 and B_2 use CAMS to test whether answer-supporting reasoning trajectories form coherent and separable latent answer groups. When reasoning-stage stability is insufficient, B_3 switches to AF-SCS, which evaluates whether explicit visual evidence and subsequent reasoning form a stable support chain.

resulting cluster memberships replace exact-string matches when computing support and stability. This is a possible extension; our experiments use normalized short answers and do not evaluate judge-based clustering.

Let y_i denote the canonical answer label for trajectory i: the normalized answer string in our evaluated setting, or the judge-assigned cluster identifier in the extension above. For a candidate label y, we define its accumulated support set at stage B_t as

$$G_t(y) = \{i \mid y_i = y,\ i \text{ belongs to samples accumulated up to } B_t\}, \qquad t \in \{1, 2, 3\}, \tag{1}$$

where N_t denotes the total number of accumulated trajectories up to B_t. These accumulated groups are used to compute vote support, entropy confidence, persistence, and reasoning-stage answer-manifold stability.For verification-stage alignment, we further define $G_3^{\text{ver}}(y)$ as the subset of B_3 trajectories that support answer y and contain explicit `<evidence>` and `<reasoning>` spans.

3.3 Vote Support, Entropy Confidence, and Budget Persistence

Vote support V. At stage B_t, the vote support of candidate answer y is

$$V_t(y) = \frac{|G_t(y)|}{N_t}. \tag{2}$$

Vote support measures how frequently an answer appears among sampled trajectories. It is useful as a first-order support signal, but frequency alone is unreliable in multimodal reasoning because multiple trajectories may share the same visual misperception, arithmetic shortcut, or option prior. Therefore, MUST uses V as one component of the controller rather than as a final majority-vote rule.

Entropy confidence Q. For trajectory i, let the final-answer span contain T_i tokens, and let $p_\tau(v)$ be the token distribution at answer-token position τ. We compute

$$\bar{H}_i = \frac{1}{T_i} \sum_{\tau=1}^{T_i} \left(-\sum_{v} p_\tau(v) \log p_\tau(v) \right), \qquad Q_t(y) = \frac{1}{|G_t(y)|} \sum_{i \in G_t(y)} \frac{1}{\bar{H}_i + \epsilon}. \tag{3}$$

Entropy confidence measures how certain the model is when emitting the final answer [Malinin and Gales, 2018]. In MUST, Q serves as a calibration-oriented auxiliary signal: it helps downweight uncertain answers, but it is not used as a standalone correctness criterion. We analyze the usefulness and limitations of Q in Appendix E.

Budget persistence P. Budget persistence measures whether a candidate answer remains competitive as the inference budget increases. [Liu and Wang, 2025] To avoid circular dependence, we compute P from stage-local provisional rankings rather than from final scores that already include P. At each stage, provisional rankings are computed using the same active signals except P. Let $\widetilde{\mathcal{T}}_\ell^K$ denote the resulting top-K candidate set at stage B_ℓ. Starting from B_2, MUST defines

$$P_t(y) = \frac{\sum_{\ell=1}^{t} w_\ell \, \mathbb{I}\left[y \in \widetilde{\mathcal{T}}_\ell^K\right]}{\sum_{\ell=1}^{t} w_\ell}, \qquad t \in \{2, 3\}, \tag{4}$$

where later stages receive larger weights, i.e., $w_3 > w_2 > w_1$.

Intuitively, $P_t(y)$ is a cross-budget survival signal. An answer that is temporarily frequent but disappears under deeper reasoning or verification is less reliable than an answer that remains among the top candidates as the budget increases. We provide additional analysis of P in Appendix E.

3.4 Contrastive Answer-Manifold Stability for Reasoning Stages

For B_1 and B_2, MUST uses *Contrastive Answer-Manifold Stability* (CAMS). For each reasoning trajectory i, we mean-pool the last-layer hidden states over its main reasoning span T_i^{rea}:

$$r_i = \frac{1}{|T_i^{\mathrm{rea}}|} \sum_{t \in T_i^{\mathrm{rea}}} h_t. \tag{5}$$

For a candidate answer y, let μ_y be the center of the reasoning representations supporting y. We define the answer-manifold margin as

$$\mathrm{CAMS}_t(y) = \underbrace{\frac{1}{|G_t(y)|} \sum_{i \in G_t(y)} \cos(r_i, \mu_y)}_{\text{within-answer coherence}} - \underbrace{\max_{y' \neq y} \cos(\mu_y, \mu_{y'})}_{\text{nearest competing answer similarity}}, \qquad t \in \{1, 2\}. \tag{6}$$

When no competing answer group exists, the competing similarity term is set to zero, reducing CAMS to within-answer coherence.

CAMS evaluates whether the trajectories supporting an answer are both internally coherent and separated from competing answer groups. This contrastive margin is important because majority traps can also be internally compact: if many samples repeat the same wrong visual interpretation or arithmetic shortcut, ordinary compactness may still assign them a high score. CAMS instead favors answer groups whose latent reasoning representations are not only coherent but also distinguishable from plausible alternatives. Detailed ablations and distributional analyses of CAMS are provided in Appendix F.

3.5 Attention-Free Support-Chain Stability for Verification

The verification stage addresses a different failure mode from the reasoning stages. After B_2, an answer may appear stable in the latent reasoning space but still rely on an incorrect or weakly supported visual interpretation. A natural way to assess visual support is to examine how generated reasoning attends to image tokens or patches. However, in efficient MLLM inference, such token-to-patch attention maps are often unavailable, implementation-dependent, or difficult to compare consistently across decoding steps, especially when FlashAttention-style kernels are used. MUST therefore performs verification without requiring explicit attention-map extraction.

At B_3, the model is prompted to first externalize the visual evidence it uses in a `<evidence>` span and then produce the subsequent reasoning in a `<reasoning>` span. For each verification trajectory i, we mean-pool the hidden states of these two spans into h_i^{evd} and h_i^{rea}, and compute their alignment:

$$c_i = \cos(h_i^{\mathrm{evd}}, h_i^{\mathrm{rea}}). \tag{7}$$

For candidate answer y, AF-SCS compares the support-chain alignment of verification trajectories supporting y against those supporting competing answers:

$$\text{AF-SCS}(y) = \underbrace{\frac{1}{|G_3^{\mathrm{ver}}(y)|} \sum_{i \in G_3^{\mathrm{ver}}(y)} c_i}_{\text{alignment of support chains for } y} - \underbrace{\frac{1}{|\bar{G}_3^{\mathrm{ver}}(y)|} \sum_{j \in \bar{G}_3^{\mathrm{ver}}(y)} c_j}_{\text{alignment of competing support chains}}, \tag{8}$$

where $\bar{G}_3^{\text{ver}}(y)$ denotes B_3 verification trajectories supporting answers other than y. When no competing verification trajectories exist, the competing alignment term is set to zero, reducing AF-SCS to the mean alignment of trajectories supporting y.

AF-SCS measures whether the evidence stated by the model and the reasoning that follows form a consistent latent support chain for the same candidate answer. The comparison against competing verification trajectories makes the signal answer-conditioned: an answer receives a high AF-SCS score when its evidence–reasoning alignment is stronger than that of alternative answers. Thus, B_3 does not simply extend the reasoning budget; it changes the selection criterion from reasoning-manifold separability to evidence–reasoning support consistency. More detailed comparisons between AF-SCS, CAMS, and ordinary verification variants are provided in Appendix G.

3.6 Stage-Conditioned Selection and Early Stopping

After each stage, MUST ranks candidate answers using a weighted combination of the active normalized signals in Table 4. The score at B_1 is used only for provisional ranking and persistence estimation; MUST does not stop after B_1 because this stage is intended for low-cost candidate discovery. The first stopping decision is made after B_2.

After B_2, the controller stops only when the current winner remains consistent with the B_1 winner, has sufficient CAMS separability, is clearly separated from the runner-up candidate, and receives enough trajectory support. These requirements jointly test temporal agreement, answer-manifold stability, decision margin, and minimum support. If any requirement fails, MUST invokes B_3, where the active stability criterion switches from CAMS to AF-SCS. The final answer is then selected using the verification-stage score, which combines support, confidence, persistence, and support-chain stability.

This rule lets stable examples stop early while routing unstable examples to explicit visual-evidence verification. All scoring weights and stopping thresholds are fixed across the main experiments and reported in Appendix B.

4 Experiments

4.1 Experimental Setup

Evaluation benchmarks. We evaluate on five reasoning-intensive multimodal benchmarks chosen to cover complementary mathematical and logical failure modes: **MathVista** [Lu et al., 2023] for mathematically intensive visual reasoning, **MathVerse** [Zhang et al., 2024] for diagram-intensive visual mathematical reasoning, **DynaMath** [Zou et al., 2024] for dynamic multimodal math reasoning with multi-step solution chains, **WE-MATH** [Qiao et al., 2025] for human-like visual mathematical reasoning and knowledge generalization, and **LogicVista** [Xiao et al., 2024] for multimodal logical reasoning in visual contexts.

Baselines. For the main accuracy comparison, we consider four groups of methods: representative open-source MLLMs as strong model-level references, including Qwen2.5-VL-7B [Bai et al., 2025], InternVL3-8B [Zhu et al., 2025], R1-OneVision [Yang et al., 2025a], MM-R1 [Leng et al., 2025], OpenVLThinker [Deng et al., 2025b], and VisionR1 [Huang et al., 2025]; classical TTS methods (BoN [Stiennon et al., 2020], Majority Voting [Wang et al., 2022], and Tree Search [Yao et al., 2023]); recent TTS methods (Shortest Majority Vote [Zeng et al., 2025], Budget Forcing [Muennighoff et al., 2025], and Knockout-style [Chen et al., 2024]); and MUST on the same Qwen2.5-VL-7B base model.

Metrics. We report benchmark accuracy, average generated tokens, average number of trajectories, and early-stop rate. For analysis we also report answer-change rate across budgets, correct→wrong transition rate, cluster purity, and signal-based correctness AUROC.

4.2 Main Results

Table 5 compares open-source MLLMs, classical test-time scaling methods, recent TTS methods, and our MUST. The main pattern is that MUST leads on four of the five benchmarks and achieves the

Table 5: Comparison of open-source reasoning models, classical test-time scaling methods, recent test-time scaling methods, and our MUST across multimodal mathematical reasoning benchmarks.

Model / Method	MathVista	MathVerse	LogicVista	DynaMath	We-Math	Avg.
Open-source MLLMs						
Qwen2.5-VL-7B	66.3	46.9	47.9	58.0	61.0	56.0
InternVL3-8B	70.5	38.5	44.5	54.9	60.3	53.7
R1-OneVision	64.1	46.4	-	53.5	61.8	-
MM-R1	71.0	49.8	45.6	57.5	67.9	58.4
OpenVLThinker	70.2	47.9	46.8	57.4	64.5	57.4
VisionR1	72.3	52.4	-	-	-	-
Classical Test-Time Scaling Methods						
BoN	64.7	45.9	48.5	55.3	61.9	55.3
Majority Voting	68.9	48.0	45.7	59.2	62.5	56.9
Tree Search	70.2	48.4	49.6	59.8	62.7	58.1
Recent Test-Time Scaling Methods						
Shortest Majority Vote	69.5	48.7	51.9	58.3	62.2	58.1
Budget Forcing	68.1	47.3	48.8	57.7	60.8	56.5
Knockout-style	70.8	46.2	**52.3**	58.6	63.3	58.2
MUST(ours)	**73.4**	**54.8**	51.6	**61.4**	**69.3**	**62.1**

highest reported average (62.1). More importantly, the gain supports the stage-conditioned stability hypothesis: using the same criterion across all stages is inferior to switching from answer-manifold stability during reasoning to support-chain stability during verification.

The improvements are broad rather than concentrated on a single benchmark. MUST leads on most tasks and remains competitive on the remaining one, with especially clear benefits on demanding multimodal mathematical reasoning settings. This pattern is consistent with our claim that MLLM TTS benefits from criterion selection, not merely from larger inference budgets.

The comparison against open-source reasoning models points to the same conclusion. A well-controlled inference strategy on top of a fixed backbone can outperform stronger standalone reasoning-oriented models, indicating that stage-conditioned inference-time control is itself a major source of performance gains.

4.3 Ablation Experiments

A central question is whether the improvement of MUST comes from simply using more prompts and trajectories, or from applying the *right stability criterion at the right inference stage*. Table 6 studies this question from three perspectives: removing the stage-conditioned controller, replacing stage-specific stability criteria, and disabling individual auxiliary components. We report average accuracy, average generated tokens, correct-to-wrong transition rate, and early-stop rate to capture not only reliability and compute, but also how often a variant can avoid unnecessary verification.

The controller-level ablations first show that the gain does not come from simply running three stages. Three-stage prompting followed by majority voting substantially reduces accuracy and increases the correct-to-wrong rate, showing that additional prompts alone are insufficient. Fixed scoring across all stages is also worse than the full controller, which directly supports the stage-conditioned criterion-switch hypothesis. Stopping after B_2 is compute-efficient and trivially yields a 100% early-stop rate, but it loses accuracy because uncertain examples do not receive support-chain verification. Conversely, always running B_3 gives slightly higher raw accuracy but eliminates adaptive stopping and requires much more compute. The full controller instead reaches a balanced regime: it stops early for a substantial fraction of examples while preserving most of the benefit of verification when verification is actually needed.

The stage-specific signal ablations further validate the key design of MUST. Removing AF-SCS from B_3 hurts performance, showing that the verification stage cannot simply continue using reasoning-stage answer-manifold stability. Removing the explicit evidence span or using answer-conditioned alignment only also degrades accuracy and increases correct-to-wrong transitions, suggesting that the

Table 6: Ablation study on MathVista. MUST achieves the best accuracy–compute trade-off by using stage-conditioned stability control. Detailed signal-specific ablations are provided in Appendix E, Appendix F, and Appendix G.

Variant	Acc.	Avg. Tokens	C2W Rate	Early-stop Rate
Full MUST	73.4	**4528**	**5.6%**	**51.4%**
Controller-level ablations				
Three-stage prompting + majority voting	70.8	5231	8.6%	0.0%
Fixed scoring across all stages	67.9	4610	7.4%	46.8%
Stop after B_2 without verification	68.5	3129	7.0%	100.0%
No early stop, always run B_3	**73.8**	6012	6.0%	0.0%
Stage-specific signal ablations				
w/o AF-SCS in B_3	71.3	4368	7.3%	50.8%
w/o B_3 evidence span	70.9	4028	7.8%	51.0%
B_3 answer-conditioned alignment only	71.2	4473	7.6%	52.3%
w/o CAMS in B_1/B_2	69.5	4651	8.3%	34.6%
Auxiliary signal ablations				
w/o Budget Persistence P	71.9	4598	8.1%	48.9%
w/o Entropy Confidence Q	70.4	4395	6.2%	55.1%
Vote-only selection	67.7	4255	9.1%	76.7%

benefit of B_3 comes from support-chain verification rather than from an additional generic verification prompt. Removing CAMS from B_1/B_2 also causes a clear drop, confirming that early stages need answer-manifold separability rather than frequency or compactness alone.

The auxiliary-signal ablations show complementary effects. Removing Q causes the larger accuracy drop (to 70.4%, compared with 71.9% without P) and ECE increase (to 0.158, compared with 0.146 without P), whereas removing P yields the higher correct-to-wrong rate (8.1%, compared with 6.2% without Q). These patterns are consistent with Q contributing to confidence calibration and P capturing cross-stage answer stability. Finally, vote-only selection has the lowest accuracy (67.7%) and highest correct-to-wrong rate (9.1%) among the listed controller variants, reinforcing our diagnostic finding that frequency alone cannot resolve majority traps.

Additional analyses. To further support the robustness of our findings, we include several additional studies in the appendix. Signal-specific analyses of Q, P, CAMS, and AF-SCS are provided in Appendix E, Appendix F, and Appendix G. We also report statistical significance tests in Appendix H, cross-backbone results in Appendix I, and cross-domain results in Appendix J.

5 Related Work

Sampling and aggregation-based test-time scaling. Self-consistency [Wang et al., 2022] samples multiple reasoning paths and selects the majority answer. Scaling LLM Test-Time Compute Optimally [Snell et al., 2024] studies how to allocate compute between parallel sampling and sequential revision. The Art of Scaling Test-Time Compute [Agarwal et al., 2025] compares majority voting, best-of-N, beam search, and hybrid scaling recipes under different budgets. Short-m@k [Hassid et al., 2025] shows that shorter reasoning chains can sometimes yield a better accuracy–efficiency trade-off than longer ones.

Search, verifier, and refinement-based test-time scaling. Forest-of-Thought [Bi et al., 2024] extends tree-of-thought reasoning to multiple parallel trees and uses forest-level consensus for selection. LE-MCTS [Park et al., 2025] combines language-model ensembles with process-reward-guided Monte Carlo tree search. Math-Shepherd [Wang et al., 2024] uses a process reward model for step-level verification and inference-time selection. ReProbe [Ni et al., 2025] replaces heavy reward models with lightweight probes over internal states for more efficient verifier-guided scaling.

Adaptive and multimodal test-time scaling. *s1* [Muennighoff et al., 2025] uses budget forcing to control reasoning length at inference time. AgentTTS [Wang et al., 2025] treats test-time scaling as a

meta-decision problem where an agent decides how to spend extra compute. VideoChat-R1.5 [Yan et al., 2025] performs visual test-time scaling by iteratively refining visual representations. Test-Time Scaling with World Models [Yang et al., 2025b] uses simulation through world models as an additional compute source for spatial reasoning.

6 Conclusion

We reformulate MLLM test-time scaling as a problem of *stage-conditioned stability control*. By first diagnosing three core pathologies of existing multimodal TTS—budget-induced drift, vote–correctness mismatch, and fixed-budget inefficiency—we identify a deeper stability mismatch: different inference roles require different latent reliability criteria. MUST addresses this mismatch by combining CAMS for reasoning-stage answer-manifold separability, AF-SCS for verification-stage attention-free support-chain stability, and lightweight early stopping. The core message is therefore sharper: in multimodal reasoning, extra compute is valuable only when the controller knows which stability criterion is valid under the current inference role.

Acknowledgments and Disclosure of Funding

Funding. The authors received no third-party funding or support specifically for this research.

Competing Interests. The authors declare no competing financial interests.

References

Aradhye Agarwal, Ayan Sengupta, and Tanmoy Chakraborty. The art of scaling test-time compute for large language models. *arXiv preprint arXiv:2512.02008*, 2025.

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

Zhenni Bi, Kai Han, Chuanjian Liu, Yehui Tang, and Yunhe Wang. Forest-of-thought: Scaling test-time compute for enhancing llm reasoning. *arXiv preprint arXiv:2412.09078*, 2024.

Zacharie Bugaud. Hidden clones: Exposing and fixing family bias in vision-language model ensembles. *arXiv preprint arXiv:2603.17111*, 2026.

Yanxi Chen, Xuchen Pan, Yaliang Li, Bolin Ding, and Jingren Zhou. Provable scaling laws for the test-time compute of large language models. *arXiv preprint arXiv:2411.19477*, 2024.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

Shijian Deng, Kai Wang, Tianyu Yang, Harsh Singh, and Yapeng Tian. Self-improvement in multimodal large language models: A survey. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 1987–2006, 2025a.

Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. Openvlthinker: An early exploration to complex vision-language reasoning via iterative self-improvement. *arXiv e-prints*, pages arXiv–2503, 2025b.

Michael Hassid, Gabriel Synnaeve, Yossi Adi, and Roy Schwartz. Don't overthink it. preferring shorter thinking chains for improved llm reasoning. *arXiv preprint arXiv:2505.17813*, 2025.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*, 2023.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Xu Tang, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

Sicong Leng, Jing Wang, Jiaxi Li, Hao Zhang, Zhiqiang Hu, Boqiang Zhang, Yuming Jiang, Hang Zhang, Xin Li, Lidong Bing, et al. Mmr1: Enhancing multimodal reasoning with variance-aware sampling and open resources. *arXiv preprint arXiv:2509.21268*, 2025.

Chengzhi Liu, Zhongxing Xu, Qingyue Wei, Juncheng Wu, James Zou, Xin Eric Wang, Yuyin Zhou, and Sheng Liu. More thinking, less seeing? assessing amplified hallucination in multimodal reasoning models. *arXiv preprint arXiv:2505.21523*, 2025.

Xin Liu and Lu Wang. Answer convergence as a signal for early stopping in reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 17907–17918, 2025.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2023.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa P. Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023. arXiv:2303.17651.

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *Advances in neural information processing systems*, 31, 2018.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Jingwei Ni, Ekaterina Fadeeva, Tianyi Wu, Mubashara Akhtar, Jiaheng Zhang, Elliott Ash, Markus Leippold, Timothy Baldwin, See-Kiong Ng, Artem Shelmanov, et al. Efficient test-time scaling of multi-step reasoning by probing internal states of large language models. *arXiv preprint arXiv:2511.06209*, 2025.

Sungjin Park, Xiao Liu, Yeyun Gong, and Edward Choi. Ensembling large language models with process reward-guided tree search for better complex reasoning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 10256–10277, 2025.

Runqi Qiao, Qiuna Tan, Guanting Dong, MinhuiWu MinhuiWu, Chong Sun, Xiaoshuai Song, Jiapeng Wang, Zhuoma Gongque, Shanglin Lei, Yifan Zhang, et al. We-math: Does your large multimodal model achieve human-like mathematical reasoning? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20023–20070, 2025.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

An Vo, Khai-Nguyen Nguyen, Mohammad Reza Taesiri, Vy Tuong Dang, Anh Totti Nguyen, and Daeyoung Kim. Vision language models are biased. *arXiv preprint arXiv:2505.23941*, 2025.

Fali Wang, Hui Liu, Zhenwei Dai, Jingying Zeng, Zhiwei Zhang, Zongyu Wu, Chen Luo, Zhen Li, Xianfeng Tang, Qi He, et al. Agenttts: Large language model agent for test-time compute-optimal scaling strategy in complex tasks. *arXiv preprint arXiv:2508.00890*, 2025.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439, 2024.

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. *arXiv preprint arXiv:2203.11171*, 2022.

Xueqing Wu, Yuheng Ding, Bingxuan Li, Pan Lu, Da Yin, Kai-Wei Chang, and Nanyun Peng. Visco: Benchmarking fine-grained critique and correction towards self-improvement in visual reasoning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 9527–9537, 2025.

Yijia Xiao, Edward Sun, Tianyu Liu, and Wei Wang. Logicvista: Multimodal llm logical reasoning benchmark in visual contexts. *arXiv preprint arXiv:2407.04973*, 2024.

Ziang Yan, Xinhao Li, Yinan He, Zhengrong Yue, Xiangyu Zeng, Yali Wang, Yu Qiao, Limin Wang, and Yi Wang. Videochat-r1. 5: Visual test-time scaling to reinforce multimodal reasoning by iterative perception. *arXiv preprint arXiv:2509.21100*, 2025.

Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, et al. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2376–2385, 2025a.

Yuncong Yang, Jiageng Liu, Zheyuan Zhang, Siyuan Zhou, Reuben Tan, Jianwei Yang, Yilun Du, and Chuang Gan. Mindjourney: Test-time scaling with world models for spatial reasoning. *arXiv preprint arXiv:2507.12508*, 2025b.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Zhiyuan Zeng, Qinyuan Cheng, Zhangyue Yin, Yunhua Zhou, and Xipeng Qiu. Revisiting the test-time scaling of o1-like models: Do they truly possess test-time scaling capabilities? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4651–4665, 2025.

Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186. Springer, 2024.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

Chengke Zou, Xingang Guo, Rui Yang, Junyu Zhang, Bin Hu, and Huan Zhang. Dynamath: A dynamic visual benchmark for evaluating mathematical reasoning robustness of vision language models. *arXiv preprint arXiv:2411.00836*, 2024.

Appendix Overview

The appendix is organized into three types of content: **Experimental Setup and Reproducibility**, **Additional Experimental Analyses**, and **Discussion and Limitations**. These categories are intended as an organizational guide rather than mutually exclusive groups, so a section may be relevant to more than one category. Below we briefly summarize the purpose of each appendix section.

Experimental Setup and Reproducibility.

- **Implementation Details and Reproducibility**: summarizes the base model, inference environment, budget schedule, controller hyperparameters, and logging setup needed to reproduce our pipeline.
- **Benchmark and Evaluation Details**: describes the datasets, answer normalization rules, evaluation metrics, and benchmark-specific protocol details used in the experiments.
- **Prompt Design**: shows the stage-specific prompts used by MUST.

Additional Experimental Analyses.

- **Extended Diagnostic Results**: provides additional evidence on compute scaling behavior, answer instability, and selection failures beyond the main text diagnostics.
- **Auxiliary Signal Analysis: Entropy Confidence and Budget Persistence**: examines how entropy confidence and budget persistence contribute as auxiliary signals in the controller.
- **Contrastive Answer-Manifold Stability Analysis**: analyzes why CAMS is effective for reasoning-stage answer selection and early stopping.
- **Attention-Free Support-Chain Stability Analysis**: analyzes why AF-SCS is effective for verification-stage scoring and support-chain reliability estimation.
- **Statistical Significance of Main Performance Gains**: reports confidence intervals and significance tests for the main performance improvements.
- **Cross-Backbone Generalization**: evaluates whether MUST transfers to additional open-source MLLM backbones.
- **Cross-Domain Generalization**: evaluates whether MUST generalizes to broader multimodal and science QA settings beyond the main benchmarks.

Discussion and Limitations.

- **Limitations**: discusses the method's dependence on white-box access, its short-answer evaluation setting and possible judge-based extension to long answers, and failure modes caused by correlated visual misperception.

A Limitations

Although MUSTimproves the accuracy–compute trade-off of training-free MLLM inference, it has several limitations.

Dependence on white-box model access. The current implementation uses hidden states and token-level probability distributions to compute CAMS and entropy confidence, and at B_3 it uses mean-pooled `<evidence>` and `<reasoning>` span representations to compute AF-SCS. This requires access to internal model states and logits. Therefore, the method is directly applicable to open-source models such as Qwen2.5-VL, InternVL, or LLaVA-like models, but is less directly applicable to black-box API models. A black-box approximation could replace CAMS with semantic embedding similarity and AF-SCS with support-chain heuristics over extracted evidence/reasoning text, but such approximations may weaken reliability.

Answer granularity and evaluation setting. The current implementation and experiments use tasks with compact final answers, such as multiple-choice questions, yes/no decisions, numeric answers, chart values, and concise open-ended labels. Exact string normalization alone does not reliably group long answers that express the same meaning in different words. For open-ended generation, a model-based judge could compare answer spans in question context and cluster semantically consistent responses; each cluster would then serve as a candidate for the same support and stability calculations used by MUST. This judge-based extension is not evaluated in this work and would add both inference cost and dependence on judge reliability. Accordingly, our main evaluation is restricted to benchmarks with automatically verifiable or short-form answers.

Correlated visual misperception. If all sampled reasoning trajectories share the same visual misinterpretation, stage-conditioned control can still select a wrong answer with high confidence. For example, if the model consistently misreads a chart axis, CAMS may report a stable wrong answer manifold, and the B_3 evidence spans may still form a coherent but incorrect support chain. AF-SCS should therefore be interpreted as an attention-free proxy for verification-stage reliability, not as proof of pixel-level grounding.

B Implementation Details and Reproducibility

B.1 Base Model and Inference Environment

Table 7 summarizes the default implementation settings. All experiments use the same base model and decoding framework unless otherwise specified. We recommend logging both raw generations and derived statistics because most diagnostic analyses require per-sample trajectories across stages.

Table 7: Default implementation settings.

Item	Default Setting
Base model	Qwen2.5-VL-7B-Instruct
Precision	BF16
Inference framework	HuggingFace Transformers / vLLM-compatible implementation
Hardware	4 RTX 4090 GPUs
Image processing	Official Qwen2.5-VL processor
Min image pixels	256*28*28
Max image pixels	1280*28*28
Batch size	1–8 depending on GPU memory
Temperature	0.7 for sampling, 0.0 for greedy baselines
Top-p	1.0
Repetition penalty	1.0
Random seeds	42
Hidden-state layer	Last decoder layer
Entropy source	Token-level softmax probabilities over answer tokens

B.2 Budget Schedule

The default budget schedule contains three stages, shown in Table 8. The first stage builds a low-cost candidate pool, the second stage checks whether answer clusters remain stable under more systematic reasoning, and the third stage allocates a larger budget to explicit visual evidence verification for only the unstable cases.

Table 8: Default three-stage inference schedule. The precise sample counts can be adjusted according to compute constraints, but the stage-specific signal design remains fixed.

Stage	Role	Max new tokens	samples	Active signals
B_1	Fast hypothesis generation	1024	3	V, Q, CAMS
B_2	Deliberate reasoning	1024	3	V, Q, CAMS, P
B_3	Support-chain verification	2048	3	$V, Q, \text{AF-SCS}, P$

B.3 Controller Hyperparameters

Table 9 summarizes the default controller hyperparameters used by MUST. These values are fixed across the main experiments unless otherwise noted.

Table 9: MUST controller hyperparameters.

Item	Default Setting
Candidate top-K for persistence	$K = 3$
B_1 scoring weights	$\alpha_1 = 0.4,\ \beta_1 = 0.3,\ \delta_1 = 0.3$
B_2 scoring weights	$\alpha_2 = 0.35,\ \beta_2 = 0.25,\ \gamma_2 = 0.2,\ \delta_2 = 0.2$
B_3 scoring weights	$\alpha_3 = 0.35,\ \beta_3 = 0.25,\ \gamma_3 = 0.2,\ \delta_3 = 0.2$
CAMS early-stop threshold	$\tau_{\text{CAMS}} = 0.35$
$S(y_1) - S(y_2)$ threshold	$\tau = 0.3$
CAMS normalization	min-max
AF-SCS normalization	min-max
Early-stop decision point	After B_2 only
Verification trigger	Any B_2 stop condition fail

B.4 Logged Fields

For reproducibility, each inference run should save the fields in Table 10. Without these logs, later analyses such as answer drift, signal complementarity, and early-stop behavior cannot be reconstructed.

Table 10: Recommended per-sample logs.

Field	Description
sample_id	Unique benchmark sample identifier
benchmark	Benchmark name
image_path	Local path or benchmark image identifier
question	Original question text
ground_truth	Ground-truth answer in benchmark evaluation format
stage	Budget stage B_l
raw_response	Full model generation
reasoning_span	Reasoning text if explicitly available
evidence_span	B_3 `<evidence>` text if available
final_answer_raw	Raw final answer string
final_answer	Extracted final answer used for grouping
correct	Whether the extracted final answer matches the ground truth
answer_tokens	Token ids corresponding to the extracted final-answer span
evidence_tokens	Token ids corresponding to the B_3 evidence span
answer_embedding	Mean-pooled hidden-state vector r_i for B_1/B_2
evidence_reasoning_repr	Mean-pooled hidden-state pair $(h_i^{\text{evd}}, h_i^{\text{rea}})$ for B_3
answer_entropy	Average entropy over answer tokens
cluster_id	Candidate answer cluster
V, CAMS, P, Q, AF-SCS, S_t	Candidate-level stage-conditioned scores
top1, top2	Top-1 and top-2 candidate answers at each stage
score_margin	$S_t(y_1) - S_t(y_2)$
early_stop	Whether the current stage triggers early exit
stop_condition_flags	Boolean flags for each early-stop criterion

C Benchmark and Evaluation Details

C.1 Benchmarks

Table 11 summarizes the primary evaluation benchmarks. For all benchmarks, we follow the evaluation protocols officially specified by their respective datasets. Because LogicVista has a relatively small evaluation set (448 examples), we report its final accuracy as the average across three runs.

Table 11: Primary evaluation benchmarks.

Benchmark	Samples	Answer Type	Motivation
MathVista	5141	MCQ / Free-form	Visual mathematical reasoning
MathVerse	2,612	MCQ / Free-form	Diagram-intensive mathematical reasoning
LogicVista	448	Multiple-choice	Multimodal logical reasoning in visual contexts
DynaMath	5010	MCQ / Free-form	Dynamic multi-step mathematical reasoning
WE-MATH	6,500	Multiple-choice	Human-like mathematical reasoning and generalization

Table 12: Budget-induced answer drift on MathVista. Larger token budgets change a substantial fraction of predictions, but the changes are not monotonically beneficial.

Transition	Change rate	Correct→Wrong	Wrong→Correct	Avg. entropy Δ
$128 \rightarrow 256$	46.7%	7.9%	23.1%	-0.0073
$256 \rightarrow 512$	32.3%	6.7%	13.5%	$+0.0105$
$512 \rightarrow 1024$	31.1%	10.1%	17.9%	-0.0139

C.2 Answer Parsing and Evaluation Rules

We use deterministic final-answer parsing whenever possible. For multiple-choice questions, we extract the last valid option letter from the final-answer span, prioritizing explicit patterns such as "Answer: C", "Option C", or "(C)". For yes/no questions, variants such as "yeah", "true", and "correct" are mapped to `yes`, and variants such as "no", "false", and "incorrect" are mapped to `no`. For numeric questions, we extract the first valid number from the final-answer span and compare it with the ground truth under benchmark-specific tolerance. For open short answers, we lowercase the answer, strip whitespace, remove trailing punctuation, and optionally remove articles.

D Extended Diagnostic Results

The main paper provides a compact diagnosis of three failure modes in existing multimodal test-time scaling: budget-induced answer drift, majority-vote traps, and unstable single-signal confidence estimates. This appendix expands these diagnostic results to better characterize when and why these failures occur. The goal of this appendix is not to evaluate the proposed controller, but to provide additional evidence that fixed-budget and fixed-criterion inference pipelines are structurally brittle in multimodal reasoning.

D.1 Budget-Induced Answer Drift on Reasoning and Perception Benchmarks

We first examine whether increasing the token budget reliably improves performance on two representative benchmarks: MathVista, which emphasizes visual mathematical reasoning, and MMStar, which is more perception- and grounding-heavy. Across both settings, larger budgets change answers substantially but do not yield monotonic gains.

Table 12 shows that budget increases strongly affect final predictions on MathVista, with change rates from 31.1% to 46.7%. The key result is that every transition still incurs a sizable correct-to-wrong rate (6.7%–10.1%), so extra reasoning can help some examples but also destabilize previously correct ones.

Table 13 shows a similar pattern on MMStar, though with lower overall drift. The central point is that correct-to-wrong flips remain substantial and even rise to 11.2% at the largest transition, suggesting that when the bottleneck is visual grounding, longer reasoning often reinforces the initial perception rather than fixing it. Taken together, the two tables show that more compute is not a reliable default strategy; extra budget should be allocated selectively, with a stability criterion deciding when further reasoning is worth trusting.

Table 13: Budget-induced answer drift on MMStar. Perception-oriented multimodal examples show lower overall drift than MathVista, but larger budgets still introduce non-trivial correct-to-wrong transitions.

Transition	Change rate	Correct→Wrong	Wrong→Correct	Avg. entropy Δ
$128 \rightarrow 256$	34.8%	9.4%	14.8%	-0.0048
$256 \rightarrow 512$	27.3%	8.9%	10.7%	$+0.0061$
$512 \rightarrow 1024$	25.1%	11.2%	9.4%	$+0.0034$

Table 14: Taxonomy of majority-vote traps on MathVista. Ratios are computed among examples where majority voting selects an incorrect answer.

Failure type	Ratio
Correlated arithmetic shortcut	26.4%
Visual attribute misread	22.7%
Option-prior bias	15.8%
Fragmented correct forms	14.6%
Overconfident wrong cluster	12.1%
Parsing or normalization ambiguity	8.4%

D.2 Majority-Trap Taxonomy on MathVista

We next analyze majority-vote failures on MathVista. Majority voting is often used as a simple reliability proxy under test-time scaling, but its correctness depends on the assumption that independently sampled trajectories make sufficiently diverse errors. In multimodal reasoning, this assumption is frequently violated: different samples can share the same visual misunderstanding, arithmetic shortcut, or option prior. We therefore categorize majority-vote errors into several failure types.

The dominant failure types are not random sampling noise. Correlated arithmetic shortcuts and visual attribute misreads together account for nearly half of the majority-vote failures, indicating that multiple trajectories often repeat the same underlying mistake. This explains why majority support alone cannot be treated as a sufficient reliability signal. A wrong answer can become frequent precisely because the model has a stable but incorrect interpretation of the visual or mathematical structure.

This result motivates the need to go beyond frequency-based aggregation. If an answer cluster is frequent but built on a shared wrong perception or shortcut, then vote support alone will reinforce the error. A stronger diagnostic signal should distinguish between superficial agreement and reliable agreement. In other words, test-time scaling should evaluate not only how many trajectories support an answer, but also whether that support remains stable under more deliberate reasoning and whether later verification provides consistent evidence for the answer.

D.3 Entropy-Confidence Failure Cases

We further examine whether entropy-based confidence can serve as a standalone selection criterion. Lower answer entropy generally indicates that the model is more certain when emitting the final answer. However, multimodal reasoning can produce overconfident errors: once the model commits to a wrong visual interpretation or a wrong intermediate computation, it may output the final answer with high certainty. Conversely, some correct answers may have high entropy because the model remains uncertain about formatting, units, or equivalent expressions.

Table 15 groups MathVista examples into five groups according to the average token-level entropy of the generated final-answer span, where lower entropy indicates higher model confidence. Accuracy denotes the correctness of the selected answer within each bin. C2W rate measures the fraction of examples that change from correct to wrong when the inference budget is increased, reflecting overthinking-induced instability. Avg. vote margin is the average gap between the vote ratio of the most frequent answer and that of the second most frequent answer, indicating how concentrated the sampled answers are.

Table 15: Entropy-confidence failure cases on MathVista. Accuracy is generally lower in high-entropy bins, but low entropy does not eliminate overconfident errors.

Entropy bin	Accuracy	C2W rate	Avg. vote margin
Lowest 20%	67.5%	11.7%	0.72
20–40%	69.0%	9.8%	0.58
40–60%	66.5%	8.3%	0.41
60–80%	63.5%	6.9%	0.35
Highest 20%	56.5%	6.4%	0.18

Table 16: Risk–gain comparison across test-time scaling axes. Refinement has the highest risk-to-gain ratio; increasing the number of samples changes fewer answers overall, although correlated errors remain possible.

Scaling axis	change rate	W2C	C2W	Risk-to-gain	Diagnosis
Token budget	36.7%	18.2%	8.2%	0.45	Useful but unstable
Refinement rounds	34.1%	8.8%	14.4%	1.64	Highest overthinking risk
Sample count	10.8%	4.9%	4.5%	0.92	Fewest changes; non-monotonic

The table shows that entropy is informative but insufficient. Accuracy is highest in the 20–40% entropy bin (69.0%) and then declines to 56.5% in the highest-entropy bin, indicating lower accuracy at high entropy overall, although the trend is not strictly monotonic. Crucially, the lowest-entropy bin achieves 67.5% accuracy but has the highest correct-to-wrong rate (11.7%), compared with 6.4% in the highest-entropy bin. Thus, low entropy can indicate confidence without guaranteeing stability; a low-entropy final answer may reflect premature commitment rather than genuine reliability.

This result supports the use of entropy confidence only as one component of a broader stability estimate. Entropy is useful for detecting uncertain generations, but it cannot identify whether an answer is visually grounded, whether the supporting trajectories are separated from competing answer clusters, or whether the answer survives increased reasoning budget. Therefore, entropy should be combined with complementary stability signals rather than used as a standalone answer-selection rule.

D.4 Scaling-Axis Risk Comparison

Finally, we compare the risk profiles of different test-time scaling axes: longer token budgets, additional refinement rounds, and more sampled trajectories. For each axis, we report the average answer-change rate, the average wrong-to-correct rate, the average correct-to-wrong rate, and a risk-to-gain ratio:

$$\text{Risk-to-Gain} = \frac{\text{Correct} \to \text{Wrong}}{\text{Wrong} \to \text{Correct} + \epsilon}. \tag{9}$$

A lower value indicates that the scaling axis produces more beneficial corrections relative to harmful flips, while a higher value indicates a larger overthinking or destabilization risk.

The three scaling axes exhibit different failure patterns. Increasing the token budget yields the largest beneficial correction rate, but it also changes a large fraction of answers, meaning that the method needs to decide when the additional reasoning should be trusted. Refinement rounds are the riskiest axis: the correct-to-wrong rate exceeds the wrong-to-correct rate, suggesting that repeated self-revision often encourages the model to abandon correct intermediate answers. Increasing the number of samples is less disruptive in absolute terms, with the lowest answer-change rate (10.8%) and correct-to-wrong rate (4.5%), although its risk-to-gain ratio (0.92) is higher than that of token-budget scaling (0.45). It also does not remove majority traps because correlated errors can still dominate the vote distribution.

These results suggest that multimodal test-time scaling should not be treated as a single-dimensional compute allocation problem. Different forms of extra compute introduce different reliability risks. A robust inference-time method should therefore adapt its decision criterion to the current stage of inference: cheap sampling can help form candidate hypotheses, deliberate reasoning can test whether

Table 17: Effect of entropy confidence Q on correctness prediction and calibration.

Signal set	Selection Acc.	AUROC	AUPRC	ECE	Brier
Q only	62.8%	0.7212	0.9116	0.203	0.229
V only	67.7%	0.7823	0.9353	0.184	0.216
$V + Q$	67.9%	0.7891	0.9403	0.171	0.204
$V + \text{CAMS}$	68.8%	0.7937	0.9461	0.169	0.201
$V + \text{CAMS} + Q$	69.6%	0.7985	0.9498	0.158	0.193
$V + \text{AF-SCS} + P$	70.4%	0.8088	0.9574	0.158	0.186
$V + \text{AF-SCS} + P + Q$	73.4%	0.8146	0.9608	0.139	0.178

candidate answers remain stable, and support-chain verification should be used selectively when earlier stages do not provide sufficient stability.

E Auxiliary Signal Analysis: Entropy Confidence and Budget Persistence

This section analyzes the two auxiliary signals used in MUST: entropy confidence Q and budget persistence P. Unlike vote support V, which is a common aggregation signal in test-time scaling, Q and P provide complementary information about generation confidence and cross-budget stability. Our goal here is not to argue that either signal is sufficient on its own, but to show that they improve reliability estimation when combined with stage-specific stability signals.

E.1 Entropy Confidence Improves Calibration but Is Not Sufficient

Entropy confidence Q measures how certain the model is when emitting the final-answer span. Lower answer entropy usually indicates that the model assigns more probability mass to the generated answer tokens. However, as shown in the diagnostic analysis, low entropy does not guarantee correctness because multimodal models can confidently commit to wrong visual interpretations or incorrect intermediate computations. We therefore evaluate Q as an auxiliary signal rather than a standalone selection rule.

Table 17 reports the effect of adding entropy confidence to different signal combinations. The Q-only selector performs worse than vote-based selection, confirming that entropy alone cannot identify reliable answers. Nevertheless, adding Q consistently improves calibration and correctness prediction. In particular, $V + Q$ improves over V-only, and adding Q to $V + \text{CAMS}$ or $V + \text{AF-SCS} + P$ further reduces ECE and Brier score.

These results show that entropy confidence is useful mainly as a calibration signal. It helps distinguish uncertain generations from confident ones and improves the reliability of combined scoring rules. However, its standalone performance is limited because low-entropy answers may still be visually unsupported or reasoning-incorrect. This motivates our use of Q as an auxiliary confidence term rather than as the primary criterion for answer selection.

E.2 Budget Persistence Captures Cross-Stage Answer Survival

Budget persistence P measures whether a candidate answer remains competitive as inference moves from cheaper stages to more deliberate or verification-oriented stages. This signal is different from vote support. Vote support measures how many trajectories currently support an answer, whereas budget persistence measures whether the answer survives increased reasoning budget. A candidate that is temporarily frequent at one stage but disappears after more careful reasoning should be treated as less reliable than a candidate that remains stable across stages. For analysis, we divide examples into three persistence levels according to the agreement pattern of the stage-wise top-1 answers across B_1, B_2, and B_3: *low persistence* means the three stages have different top-1 answers, *medium persistence* means two stages share the same top-1 answer, and *high persistence* means all three stages share the same top-1 answer.

Table 18 shows that higher persistence is associated with higher accuracy and lower entropy. High persistence accounts for the largest share of examples (52.2%) and achieves the highest accuracy

Table 18: Correctness and stability under different budget-persistence levels.

Persistence group	Ratio	Accuracy	C2W rate	Avg. vote margin	Avg. entropy
Low persistence	18.2%	56.6%	13.6%	0.34	1.49
Medium persistence	29.6%	68.9%	9.7%	0.32	1.36
High persistence	52.2%	81.8%	-	0.47	1.20

Table 19: Comparison of answer-manifold-stability variants.

Stability signal	Dynamic range	AUROC	Selection Acc.	ECE
Canonicalized answer compactness	0.06	0.7515	68.9%	0.211
Main reasoning compactness only	0.24	0.7748	72.1%	0.184
Inter-cluster distance only	0.31	0.7812	69.4%	0.178
Proposed CAMS	0.53	0.8146	73.4%	0.139

(81.8%), the largest average vote margin (0.47), and the lowest average entropy (1.20). Medium persistence has 68.9% accuracy and 1.36 average entropy. Low persistence is the least reliable regime: it has 56.6% accuracy, the highest reported correct-to-wrong rate (13.6%), and the highest entropy (1.49). The correct-to- wrong rate is not reported for the high-persistence group because its stage-wise top-1 answer does not change across stages.

The results indicate that budget persistence provides a temporal notion of stability that cannot be captured by per-stage voting alone. Accuracy increases and entropy decreases across the persistence groups, while vote margin is highest for the high-persistence group but is not strictly ordered between the low- and medium-persistence groups. The low-persistence group also has the highest reported correct-to-wrong rate, consistent with greater answer instability. This supports using P from B_2 onward, where the controller begins to test whether an answer survives deliberate reasoning and verification.

F Contrastive Answer-Manifold Stability Analysis

This section analyzes Contrastive Answer-Manifold Stability (CAMS), the main stability criterion used in B1 and B2. The purpose of CAMS is to determine whether reasoning trajectories supporting the same answer are not only internally coherent but also separated from trajectories supporting competing answers. This differs from a simple compactness score, which may incorrectly reward a group of trajectories that make the same mistake in a consistent way.

F.1 Why CAMS Is Better than Canonicalized Answer Compactness

A natural way to estimate answer-manifold stability is to compute the similarity among hidden states associated with the final answer. However, final-answer spans are often short and template-like. For example, multiple trajectories may all end with a single option token such as "B" or a short numeric answer. As a result, canonicalized answer representations tend to collapse and provide limited discriminative power. MUST instead computes CAMS from the main reasoning trajectories and uses a contrastive margin between within-answer coherence and nearest competing-cluster similarity.

To show how different B1/B2 stability signals affect end-to-end MUST performance, Table 19 compares several answer-manifold-stability variants in terms of dynamic range and downstream answer-selection quality (AUROC, selection accuracy, and ECE). Canonicalized answer compactness has the narrowest dynamic range and weakest correctness prediction. Main reasoning compactness improves the signal because it captures actual reasoning-path variation, but it still lacks an explicit comparison against competing answers. The proposed CAMS achieves the best AUROC, selection accuracy, and calibration.

These results suggest that CAMS should not be implemented as ordinary hidden-state compactness over short answer spans. The main reasoning span contains richer information about how an answer is reached, while the contrastive margin prevents the controller from rewarding answer clusters that

Table 20: Interaction between vote support and Contrastive Answer-Manifold Stability (CAMS).

Group	Ratio	Accuracy	Majority error rate	C2W rate	Interpretation
High V, High CAMS	31.3%	82.7%	9.8%	5.1%	reliable consensus
High V, Low CAMS	18.9%	68.8%	27.4%	11.6%	majority trap
Low V, High CAMS	17.4%	76.4%	18.7%	7.2%	fragmented support
Low V, Low CAMS	32.4%	65.1%	34.2%	14.6%	unstable candidates

Table 21: Effect of CAMS on B2 answer selection and early-stage stability.

B2 scoring rule	B2 Acc.	Stop@B2 Acc.	False-stop rate	C2W avoided
$V + Q$	67.9%	70.4%	15.8%	6.2%
$V +$ CAMS	69.2%	73.0%	14.4%	5.8%
$V +$ CAMS $+Q$	68.3%	73.5%	12.9%	8.5%
$V +$ CAMS $+P + Q$	69.8%	75.8%	13.6%	7.6%

are internally similar but not well separated from competing explanations. This supports the design of CAMS as a contrastive answer-manifold stability signal for early-stage reasoning.

F.2 CAMS Detects Majority Traps in Early Stages

We next study whether CAMS provides information beyond vote support. If CAMS were merely a proxy for voting, high-vote answers would always have high answer-manifold stability. In practice, this is not the case. Some answers obtain high vote support because multiple trajectories repeat the same visual or mathematical shortcut, but their reasoning representations are not well separated from competing answer clusters. Conversely, some lower-frequency answers may have more coherent and better-separated reasoning trajectories.

Table 20 groups answer candidates according to vote support and Contrastive Answer-Manifold Stability (CAMS). The high-V, high-CAMS group has the highest accuracy (82.7%) and lowest correct-to-wrong rate (5.1%), consistent with reliable consensus. Although the low-V, high-CAMS group is less frequent (17.4% of candidates), it achieves 76.4% accuracy, outperforming both low-CAMS groups. By contrast, the high-V, low-CAMS group has 68.8% accuracy and a 27.4% majority error rate, while the low-V, low-CAMS group has the lowest accuracy (65.1%), highest majority error rate (34.2%), and highest correct-to-wrong rate (14.6%). These results show that vote frequency alone can be misleading and that low support combined with low manifold stability marks particularly unstable candidates.

The contrast between the high-V, high-CAMS and high-V, low-CAMS groups shows that CAMS is not redundant with vote support: within the high-vote groups, accuracy is 82.7% versus 68.8%, while majority error is 9.8% versus 27.4%. Vote support captures how many trajectories produce the same answer, while CAMS captures whether the reasoning trajectories supporting that answer form a coherent and separable cluster. This helps explain why MUST does not rely on majority voting alone: frequency is useful, but it must be checked against answer-manifold stability during B1 and B2.

F.3 CAMS Improves B1/B2 Selection and Early-Stage Stopping Reliability

Since CAMS is used in the early reasoning stages, we further evaluate its effect on B2 answer selection and early stopping. A reliable B2 decision is important because the controller may stop after B2 when the answer is already stable enough. If CAMS is removed, the controller has less information about whether the current top answer is supported by coherent reasoning trajectories.

Table 21 compares B2 scoring rules with and without CAMS. Adding CAMS improves B2 accuracy over $V + Q$, and combining CAMS with P and Q further improves stop-stage reliability. The false-stop rate decreases when CAMS is included, showing that answer-manifold stability helps prevent the controller from stopping early on unstable or weakly separated answers.

These results show that CAMS directly improves early-stage decision quality. It not only increases B2 selection accuracy but also makes early stopping safer by reducing false stops. This supports the role of CAMS as the main stability signal for B1/B2: before the controller enters explicit support-

Table 22: Comparison of verification-stage stability signals.

B3 signal variant	B3 Acc.	B3 rescue rate	B3 harm rate	AUROC	ECE
B3 with CAMS instead of AF-SCS	70.9%	14.7%	8.6%	0.7924	0.146
B3 answer-span alignment only	67.9%	13.9%	8.9%	0.7816	0.174
B3 reasoning-span compactness only	71.1%	12.2%	8.1%	0.7987	0.161
B3 evidence-span compactness only	70.3%	15.8%	7.8%	0.8019	0.157
B3 self-evaluation confidence	68.7%	11.4%	9.5%	0.7863	0.171
Proposed AF-SCS	73.4%	15.6%	8.5%	0.8146	0.139

chain verification, the most useful stability evidence comes from the structure of the main reasoning trajectories.

G Attention-Free Support-Chain Stability Analysis

This section analyzes Attention-Free Support-Chain Stability (AF-SCS), the verification-stage criterion used in B3. Once the controller enters B3, the role of inference changes: the model is no longer merely generating or deliberating over candidate answers, but explicitly verifying them against visual evidence. Therefore, the stability criterion should also change. Instead of continuing to use CAMS over reasoning trajectories, MUST measures whether the generated `<evidence>` span and the subsequent `<reasoning>` span form a stable support chain for the same answer.

G.1 AF-SCS Is Better than Using CAMS at the Verification Stage

We first compare AF-SCS with alternative B3 reliability signals. Continuing to use CAMS at B3 treats verification as another reasoning stage and ignores the fact that B3 explicitly asks the model to re-check visual evidence. Similarly, answer-span alignment is too short and template-like, while reasoning-only or evidence-only compactness cannot determine whether the evidence actually supports the reasoning.

Table 22 shows that the proposed AF-SCS achieves the best B3 accuracy, AUROC, and calibration (lowest ECE), while its rescue rate (15.6%) is second only to evidence-span compactness (15.8%). Replacing AF-SCS with CAMS reduces the rescue rate and increases the harm rate, indicating that answer-manifold stability is not the most suitable criterion once the model enters visual verification. Although evidence-span compactness has a slightly higher rescue rate, both evidence-only and reasoning-only variants have lower harm rates but also lower B3 accuracy and AUROC, as well as higher ECE, than the proposed AF-SCS. This pattern suggests that the useful signal comes from the alignment between evidence and reasoning rather than from either span alone.

These results support the stage-conditioned design choice of replacing CAMS with AF-SCS in B3. Early reasoning stability and verification-stage evidence stability measure different properties. CAMS evaluates whether reasoning trajectories are coherent and separable, while AF-SCS evaluates whether explicit visual evidence and subsequent reasoning are aligned. Therefore, B3 should not simply reuse the same criterion as B1/B2.

G.2 Support-Chain Margin Separates Correct and Incorrect B3 Answers

We further examine whether the support-chain margin is predictive of B3 correctness. Table 23 groups B3 examples by AF-SCS score. Accuracy varies across bins, and the highest-AF-SCS group achieves the highest accuracy (75.2%), largest rescue rate (17.8%), and lowest harm rate (7.2%). In contrast, the lowest-AF-SCS group has the second-lowest accuracy (72.4%) and second-highest harm rate (9.1%), suggesting that weaker support-chain alignment can be a warning sign for verification reliability.

This result shows that AF-SCS is not merely an additional confidence score. It captures a verification-specific property: whether the generated visual evidence and the reasoning that follows form a consistent support chain. A high AF-SCS score indicates that B3 verification is likely to be useful, while a low AF-SCS score signals that the verification trajectory may be visually unsupported or internally misaligned.

Table 23: B3 correctness under different Attention-Free Support-Chain Stability (AF-SCS) levels.

AF-SCS bin	B3 Acc.	Rescue rate	Harm rate
Lowest 20%	72.4%	13.3%	9.1%
20–40%	72.6%	14.7%	10.3%
40–60%	72.1%	15.8%	8.2%
60–80%	74.7%	16.4%	7.9%
Highest 20%	75.2%	17.8%	7.2%

H Statistical Significance of Main Performance Gains

To examine whether the performance improvements of MUST over the Qwen2.5-VL-7B baseline are statistically reliable, we employ paired significance tests that account for the fact that both methods are evaluated on the same benchmark instances. Unlike independent proportion tests, paired tests explicitly consider instance-level agreement and disagreement between the two methods.

For MathVista, MathVerse, DynaMath, and WE-MATH, we adopt the two-sided exact McNemar's test to assess the significance of accuracy differences. The test is based on discordant prediction pairs, namely instances for which exactly one of the two methods produces the correct answer. We also estimate 95% confidence intervals (CIs) for accuracy improvements using 20,000 paired bootstrap resamples. In each resample, paired predictions from the two methods are sampled jointly, preserving their instance-level dependence.

For LogicVista, where the reported accuracy is averaged over three runs, we account for the repeated evaluation structure at the question level. Specifically, we compute the mean correctness difference between the two methods across the three runs for each question. Confidence intervals are estimated by resampling questions while preserving their corresponding paired three-run predictions. Statistical significance is assessed using a two-sided paired sign-flip permutation test with 200,000 Monte Carlo permutations over question-level accuracy differences. This procedure avoids treating repeated predictions on the same question as independent observations.

Table 24: Paired statistical significance analysis of MUST over the Qwen2.5-VL-7B baseline across five multimodal reasoning benchmarks. Δ Acc. denotes the absolute accuracy improvement in percentage points.

Benchmark	Qwen2.5-VL-7B	MUST	Δ Acc.	95% CI	p-value
MathVista	66.3	73.4	+7.1	[4.3, 10.0]	< 0.001
MathVerse	46.9	54.8	+7.9	[5.2, 10.6]	< 0.001
LogicVista	47.9	51.6	+3.7	[-0.3, 7.7]	0.108
DynaMath	58.0	61.4	+3.4	[1.0, 5.8]	0.008
WE-MATH	61.0	69.3	+8.3	[6.0, 10.6]	< 0.001
Macro Avg.	56.0	62.1	+6.1	–	–

The paired analysis in Table 24 shows positive accuracy improvements on all five benchmarks. The gains are statistically significant on MathVista, MathVerse, DynaMath, and WE-MATH, with 95% confidence intervals that exclude zero. MathVista, MathVerse, and WE-MATH show gains of 7.1, 7.9, and 8.3 percentage points, respectively, while DynaMath improves by 3.4 points. For LogicVista, the 95% confidence interval includes zero and the permutation-test p-value is 0.108, so the improvement is not statistically significant.

Across all five primary benchmarks, the macro-average accuracy increases from 56.0% to 62.1%, corresponding to an improvement of 6.1 percentage points. This macro-average is computed by assigning equal weight to each benchmark, independently of its sample size; no separate significance test is reported for the aggregate

Table 25: Cross-backbone generalization of MUST. We report the performance of each backbone before and after applying MUST across five multimodal reasoning benchmarks.

Backbone	Method	MathVista	MathVerse	LogicVista	DynaMath	WE-MATH	Avg.
Qwen2.5-VL-3B	Base	52.2	31.8	36.0	12.5	20.7	30.6
	+ MUST	58.5	38.6	39.9	14.6	27.4	35.8
	Δ	+6.3	+6.8	+3.9	+2.1	+6.7	+5.2
InternVL3-8B	Base	70.5	38.5	44.5	54.9	60.3	53.7
	+ MUST	74.1	42.3	49.2	57.2	64.7	57.5
	Δ	+3.6	+3.8	+4.7	+2.3	+4.4	+3.8

Table 26: Cross-domain evaluation on general multimodal benchmarks. We compare Qwen2.5-VL-7B with the same backbone enhanced by MUST.

Method	MMMU	HallusionBench	MMVP	MMStar	Avg.
Qwen2.5-VL-7B	58.6	52.9	47.3	63.9	55.7
Qwen2.5-VL-7B + MUST	62.9	58.8	50.1	65.7	59.4
Δ	+4.3	+5.9	+2.8	+1.8	+3.7

I Cross-Backbone Generalization

This section evaluates whether the proposed MUST generalizes beyond the Qwen2.5-VL-7B backbone used in the main experiments. Since MUST is a training-free inference-time controller, it should be applicable to different open-source MLLMs as long as hidden states and token-level probabilities are accessible. To test this, we apply MUST to two additional backbones: Qwen2.5-VL-3B and InternVL3-8B. We use the same stage-conditioned inference procedure and the same benchmark evaluation protocol as in the main experiments.

The results in Table 25 show that MUST consistently improves both backbones, with average gains of $+5.2$ on Qwen2.5-VL-3B and $+3.8$ on InternVL3-8B. For InternVL3-8B, the largest improvements appear on LogicVista and WE-MATH. This suggests that the proposed controller remains effective across backbones and is not specific to Qwen2.5-VL-7B.

J Cross-Domain Generalization

This appendix evaluates whether MUST generalizes beyond the main math and logic benchmarks. We test Qwen2.5-VL-7B on general multimodal benchmarks and on MM-K12 science question answering.

J.1 General Multimodal Benchmarks

Table 26 reports results on MMMU, HallusionBench, MMVP, and MMStar.

MUST improves all four general benchmarks, with an average gain of $+3.7$ points. In particular, the gains on HallusionBench and MMVP suggest that the method is helpful for visually sensitive and hallucination-prone examples, not only for mathematical reasoning tasks.

J.2 Science Question Answering on MM-K12

We further evaluate MUST on the Math, Physics, Chemistry, and Biology subsets of MM-K12.

Table 27 shows consistent gains on all four MM-K12 subsets, with an average improvement of $+6.0$ points and the largest gain on Math. This pattern indicates that the method transfers beyond the main benchmarks and remains especially useful on multi-step multimodal reasoning.

Table 27: Cross-domain evaluation on MM-K12 science question answering subsets. We compare Qwen2.5-VL-7B with the same backbone enhanced by MUST.

Method	Math	Physics	Chemistry	Biology	Avg.
Qwen2.5-VL-7B	58.1	45.7	57.1	54.5	53.9
Qwen2.5-VL-7B + MUST	66.2	50.3	62.9	60.0	59.9
Δ	+8.1	+4.6	+5.8	+5.5	+6.0

K Prompt Design

B1 prompt

Look at the image and solve the following problem quickly. Give a brief solution.

Problem:{query}
{choices_block}

Use this exact format:
<reasoning>Your brief reasoning.</reasoning>

<final_answer>Your final answer.</final_answer>

B2 prompt

Analyze each option systematically if choices are provided. Otherwise, solve the problem step by step.

Use this exact format:
<reasoning>Step 1: ...Step 2: .....</reasoning>

<final_answer>Your final answer.</final_answer>

B3 prompt

Look at the image and solve the following problem. You MUST first state what you see in the image, then solve it.

Problem:{query}{choices_block}Write your answer in three parts using XML tags.

First, state the visual evidence relevant to the problem:<evidence>Describe the relevant visual information.</evidence>

Then, solve the problem step by step using that evidence:
<reasoning>Your step-by-step reasoning.</reasoning>

Finally, give only your final answer:
<final_answer>Your final answer.</final_answer>

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the central claim that reliable multimodal test-time scaling requires stage-conditioned stability criteria, and the method and experiments are scoped to training-free inference with MLLMs. The claims are supported by the diagnostic experiments, main results, and ablations on CAMS, AF-SCS, persistence, and early stopping.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a dedicated Limitations section discussing white-box access, short-answer evaluation, correlated visual misperception. These limitations clarify the scope of the method and the settings where the results may not directly generalize.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper is primarily an empirical and algorithmic study of training-free multimodal test-time scaling and does not present formal theoretical theorems or proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper describes the base model, inference stages, prompts/output structure, scoring signals, early-stopping rules, benchmark choices, answer parsing, logging fields, and evaluation metrics. Additional implementation and reproducibility details are provided in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper's data and code are open-sourced, and all benchmarks are described in detail.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the evaluation benchmarks, base model, inference framework, decoding settings, budget schedule, batch size range, precision, hidden-state layer, answer parsing rules, and signal computation details. Since the method is training-free, optimizer and training hyperparameters are not applicable.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report paired 95% confidence intervals and statistical tests in Appendix H: exact McNemar's tests for MathVista, MathVerse, DynaMath, and WE-MATH, and a paired sign-flip permutation test for LogicVista.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports the inference environment, including the Qwen2.5-VL-7B-Instruct backbone, BF16 precision, HuggingFace/vLLM-compatible inference, 4 RTX 4090 GPUs, image-processing settings, batch size range, and decoding configuration.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We promise that this paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The paper explores a test-time scaling method whose findings may have implications for the development of multimodal large language models.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release a new high-risk model, scraped dataset, or generative system. It proposes a training-free inference-time controller evaluated on existing models and public benchmarks, so additional release safeguards are not applicable beyond respecting the licenses and usage terms of existing assets.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper properly cites and credits the original sources of benchmark (MathVista,LogicVista) and models (Qwen2.5-VL) used in the experiments.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not introduce or release a new dataset, pretrained model, or benchmark asset. The proposed contribution is an inference-time algorithm and evaluation protocol built on existing models and datasets.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing experiments, new annotation tasks, user studies, or research with human subjects. Therefore participant instructions, screenshots, and compensation details are not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human-subject research, so IRB approval or equivalent review is not applicable. All experiments are conducted on existing public benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.