AdaptMMBench: Benchmarking Adaptive Multimodal Reasoning for Mode Selection and Reasoning Process

Xintong Zhang[1,2,*], **Xiaowen Zhang**[3,2,*], **Jingrong Wu**[2,*], **Zhi Gao**[1,2,4,†,♣], **Shilin Yan**[5,†], **Zhenxin Diao**[1,‡], **Kunpeng Gao**[1,‡], **Xuanyan Chen**[1,‡], **Yuwei Wu**[1,4], **Yunde Jia**[4], **Qing Li**[2,♣]

[1]Beijing Key Laboratory of Intelligent Information Technology, School of Computer Science & Technology, Beijing Institute of Technology

[2]State Key Laboratory of General Artificial Intelligence, BIGAI

[3]Xidian University

[4]Guangdong Laboratory of Machine Perception and Intelligent Computing, Shenzhen MSU-BIT University

[5]Alibaba Group

[*]Core contribution, [†]Project supervisor, [‡]Equal contribution, [♣]Corresponding authors

Correspondence: gaozhibit@bit.edu.cn, dylan.liqing@gmail.com

Abstract

Adaptive multimodal reasoning has emerged as a promising frontier in Vision-Language Models (VLMs), aiming to dynamically modulate between tool-augmented visual reasoning and text reasoning to enhance both effectiveness and efficiency. However, existing evaluations often rely on final-answer accuracy. They do not explicitly assess reasoning mode selection, namely, whether a model selects between text-only and tool-augmented visual reasoning according to its own capability boundary, nor do they reveal the quality of the subsequent reasoning and tool-execution process. In this paper, we propose AdaptMMBench, a comprehensive benchmark for adaptive multimodal reasoning across five domains: natural scene, OCR, GUI, knowledge, and math, encompassing both direct perception and complex reasoning tasks. AdaptMMBench utilizes a Matthews Correlation Coefficient (MCC) metric to evaluate the quality of reasoning mode selection, isolating this meta-cognition ability by dynamically identifying task difficulties based on models' capability boundaries. Moreover, AdaptMMBench facilitates multi-dimensional reasoning process evaluation across key step coverage, tool effectiveness, and computational efficiency. Our evaluation shows that strong reasoning mode selection is helpful, but final adaptive gains and efficiency also depend on overall model capability. Process metrics further diagnose bottlenecks in reasoning coherence, tool use, and cost for improving adaptive performance and efficiency. Code and a subset of the data are available at https://github.com/xtong-zhang/AdaptMMBench.

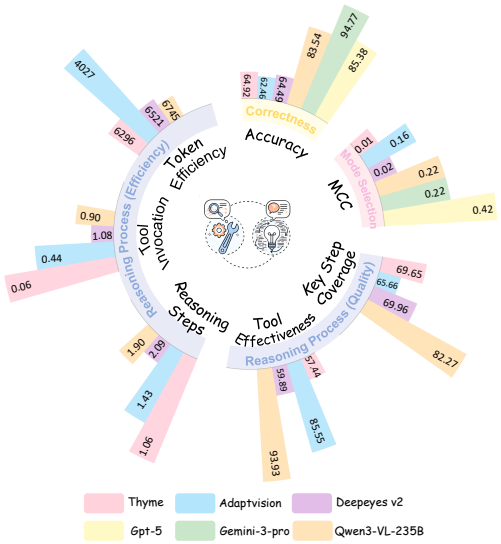


Figure 1: Comparative Analysis of Accuracy, Reasoning Mode Selection, and Reasoning Process.

1 Introduction

Vision Language Models (VLMs) have evolved from passive observers of static visual inputs to proactive models capable of dynamic information seeking. This evolution makes a shift from direct perception and textual chain-of-thought (CoT) to tool-augmented visual reasoning (i.e., think-with-image) (OpenAI, 2025b), where models iteratively manipulate the visual content using visual tools, such as zooming and enhancement (e.g., contrast and rotation) to acquire more visual information and resolve ambiguities (Zheng et al., 2025; Hu et al., 2024). However, this capability introduces computational redundancy. Without the ability to determine whether tool use is necessary for a given task, models may fall into a 'tool-invocation' trap, applying computationally intensive visual tools to

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 49761–49798
October 24-29, 2026 ©2026 Association for Computational Linguistics

problems that could be solved through direct perception or text reasoning, leading to unnecessary inefficiency and high cost. Consequently, **adaptive multimodal reasoning** is a promising direction for VLMs, which balances the necessity of such tool-augmented visual reasoning against text reasoning (Lin et al., 2025b; Wang et al., 2025a).

Despite the emergence of adaptive multimodal reasoning formulations, **evaluating adaptive multimodal reasoning remains an open problem.** Most existing (Zhang et al., 2025b,a; Wang et al., 2025a; Li et al., 2025e) evaluations rely on token-level reduction, coarse tool-call statistics, think rates or final accuracy as proxies for adaptive intelligence. While intuitive, these metrics primarily reflect outcomes rather than evaluating the internal reasoning process itself. They fail to disentangle reasoning mode selection from subsequent reasoning process (show in Fig. 1). The ability to select an appropriate reasoning mode is crucial, as it reflects the difficulty-aware meta-cognition ability. From the data perspective, adaptive reasoning is commonly evaluated on limited domains such as: logic tasks (e.g., math and knowledge reasoning) or high-resolution perception benchmarks (Wu and Xie, 2024; Wang et al., 2025c; Zhang et al., 2025c). Recent efforts such as Omni-AutoThink (Yang et al., 2025a) attempt to quantify adaptiveness through thinking rates under predefined difficulty levels. Since task difficulty is inherently model-dependent, these static predefined levels are not universally applicable across different models, leading to evaluation bias (Fig. 2). Moreover, existing evaluations often overlook reasoning process quality, leaving little process-level insight to guide future multimodal reasoning research.

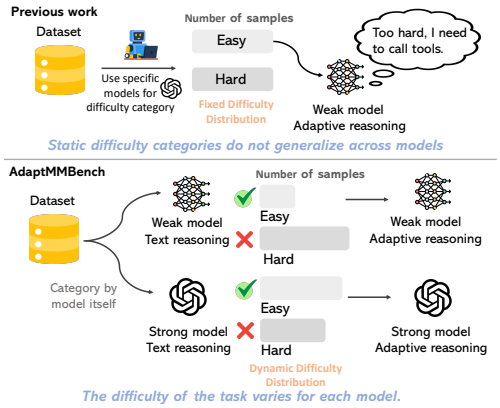


Figure 2: Comparison between our model-specific difficulty evaluation and prior work.

To bridge these gaps, we propose AdaptMMBench to quantify adaptive multimodal reasoning in VLMs. AdaptMMBench includes 1,420 samples across five domains: natural scene, OCR, GUI, knowledge, and math. Each domain contains both tasks solvable via text reasoning and complex scenarios of varying difficulties that require proactive visual tool invocation. AdaptMMBench enables the separate evaluation of reasoning mode selection and the reasoning process. Specifically, it adopts the Matthews Correlation Coefficient (MCC) to evaluate mode selection by dynamically identifying task difficulties based on model capability boundaries. For the reasoning process evaluation, we assess key step coverage, tool invocation effectiveness, and efficiency to measure reasoning coherence, tool correctness, and computational cost.

We evaluate closed-source and open-source VLMs on AdaptMMBench, with results shown in Fig. 1. Experiments show that reasoning mode selection evaluation via MCC captures a different aspect of model capability than final accuracy. Strong reasoning mode selection is helpful, but final adaptive gains and efficiency also depend on overall model capability. And the reasoning process evaluation further diagnose bottlenecks for improving adaptive performance and cost.

Our contributions are summarized as follows.

(1) We propose AdaptMMBench to quantify the adaptive reasoning capabilities of VLMs, which contains 1,420 samples across five domains with detailed annotations for comprehensive evaluations.

(2) We establish a suite of metrics for adaptive multimodal reasoning, which disentangle the adaptive capability from other model capabilities and assess three aspects of the reasoning process, providing detailed and in-depth evaluations.

(3) We provide a practical diagnostic analysis of current VLMs for adaptive reasoning. Our results confirm that adaptive mode selection is a distinct optimization target beyond final accuracy. AdaptMMBench further analyzes the quality and efficiency of the reasoning process, helping to guide the comprehensive enhancement of VLMs.

2 Related Work

2.1 Multimodal Reasoning in VLMs

Early VLMs predominantly rely on text reasoning over fixed visual encodings, imposing a "first-glance" bottleneck that limits access to fine-grained visual details (Lu et al., 2023; Huang et al., 2025; Zhang et al., 2023; Yang et al., 2025b). Recent

Table 1: Comparison with existing related benchmarks.

Benchmark	# Samples	Vision Tool	Mode Select.	Process Eval.	Difficulty Definition
MME-CoT (Jiang et al., 2025)	1,130	✗	✗	✓	✗
VTC-Bench (Zhu et al., 2026)	680	✓	✗	✓	Fixed
TIR-Bench (Li et al., 2025c)	1,000	✓	✗	✗	Fixed
VisualToolBench (Guo et al., 2025b)	1,204	✓	✗	✓	Fixed
AgentVista (Su et al., 2026)	209	✓	✗	✗	✗
HR-MMSearch (Chng et al., 2025)	305	✓	✗	✗	Fixed
Agentic-MME (Wei et al., 2026)	418	✓	✗	✓	Fixed
AdaptiveBenchmark	3,600	✗	✓	✗	Fixed
AdaptMMBench	**1,420**	✓	✓	✓	**Model-specific**

"think-with-image" methods address this limitation by acquiring additional visual information through multi-turn visual search, region zooming, and self-generated visual cues (OpenAI, 2025b; Zheng et al., 2025; Wang et al., 2025b; Lai et al., 2025; Li et al., 2025a; Chern et al., 2025). Adaptive multimodal reasoning further studies when to invoke such tools to balance accuracy and efficiency (Lin et al., 2025b; Zhang et al., 2025a; Wang et al., 2025a; Li et al., 2025d,e). However, these works offer limited evaluation of whether models invoke tool-augmented visual reasoning only when text reasoning is insufficient.

2.2 Benchmarks for VLMs

Existing VLM benchmarks cover chart/document understanding, mathematical reasoning, general-purpose VQA, reasoning-process evaluation, and high-resolution visual perception (Mathew et al., 2021; Masry et al., 2022; Lu et al., 2023; Xiao et al., 2024; Liu et al., 2024; Chen et al., 2024; Jiang et al., 2025; Wu and Xie, 2024; Wang et al., 2025c). Recent visual-grounded benchmarks emphasize fine-grained understanding and active visual reasoning (Lai et al., 2025; Li et al., 2025b,c; Lin et al., 2025a; Guo et al., 2025a). VisualToolBench evaluates tool-enabled image perception, transformation, and reasoning with detailed rubrics, but does not evaluate whether tool use is necessary relative to each model's own capability boundary (Guo et al., 2025b). More broadly, most existing benchmarks focus on task success rather than adaptive mode selection, and efficiency-oriented metrics such as tool-call turns or token reduction (Zhang et al., 2025b,a) do not evaluate adaptive decisions or reasoning quality. Tab. 1 summarizes this gap.

3 AdaptMMBench

AdaptMMBench focuses on two perspectives: reasoning mode selection and reasoning process.

3.1 Data Formulation

Formally, AdaptMMBench is constructed as a set of samples $\mathcal{D} = \{d_i\}_{i=1}^{N}$, where each data d_i is defined as

$$d_i = (I, Q, A, E, K). \tag{1}$$

Here, $I \in \mathbb{R}^{H\times W\times 3}$ denotes the input image, Q is the textual query, and A is the ground-truth answer. To support adaptive evaluation, we provide the tool annotation E that specifies how essential visual information can be obtained, including the coordinates of key image regions as well as the required transformations such as rotation and contrast adjustment. $K = \{k_1, \ldots, k_m\}$ is an ordered sequence of human-verified key reasoning steps describing the solution path from (I, Q) to A.

During inference, the model only observes the image I and the query Q. Acquiring the visual information specified by E requires invoking a visual tool $t(I, \tau)$ via code execution or function calls, where $t \in \mathcal{T}$ is a tool from the predefined toolset and τ denotes its execution arguments.

3.2 Data Collection

AdaptMMBench encompasses 1,420 samples spanning five domains: natural scene, OCR, GUI, math, and knowledge, enabling a comprehensive evaluation of adaptive reasoning across diverse scenarios, and some examples are shown in Fig. 3.

The data sources span all five domains. Natural-scene samples are drawn from VisualProbe (Lai et al., 2025) and SA-1B (Kirillov et al., 2023); text-rich and OCR samples come from ChartQA (Masry et al., 2022), DocVQA (Mathew et al., 2021), ChartQA-Pro (Masry et al., 2025), MM-RealWorld (Zhang et al., 2025c), and InSight-o3 (Li et al., 2025b). Mathematical samples build on MathVista (Lu et al., 2023), MathVerse (Zhang et al., 2024), We-Math (Qiao et al., 2025a), LogicVista (Xiao et al., 2024), VisuLogic (Xu et al., 2025), AuxSolidMath (Guo et al., 2025a), and VT-Bench (Lin et al., 2025a). GUI samples use GUI-Knowledge-Bench (Shi et al., 2025), MMBench-GUI (Liu et al., 2024), and WebWalker (Wu et al., 2025a), while knowledge-intensive samples use MMMU (Yue et al., 2024) and SciVerse (Guo et al., 2025c). Further details are provided in App. D.

To ensure difficulty stratification, annotators filter out overly simple candidates when annotating answer-region bounding boxes and key reasoning steps, so that the final benchmark contains both easy and challenging samples. The resulting difficulty composition is analyzed in App. D.3.

Building on prior adaptive reasoning methods (Chern et al., 2025; Zhang et al., 2025b), Adapt-

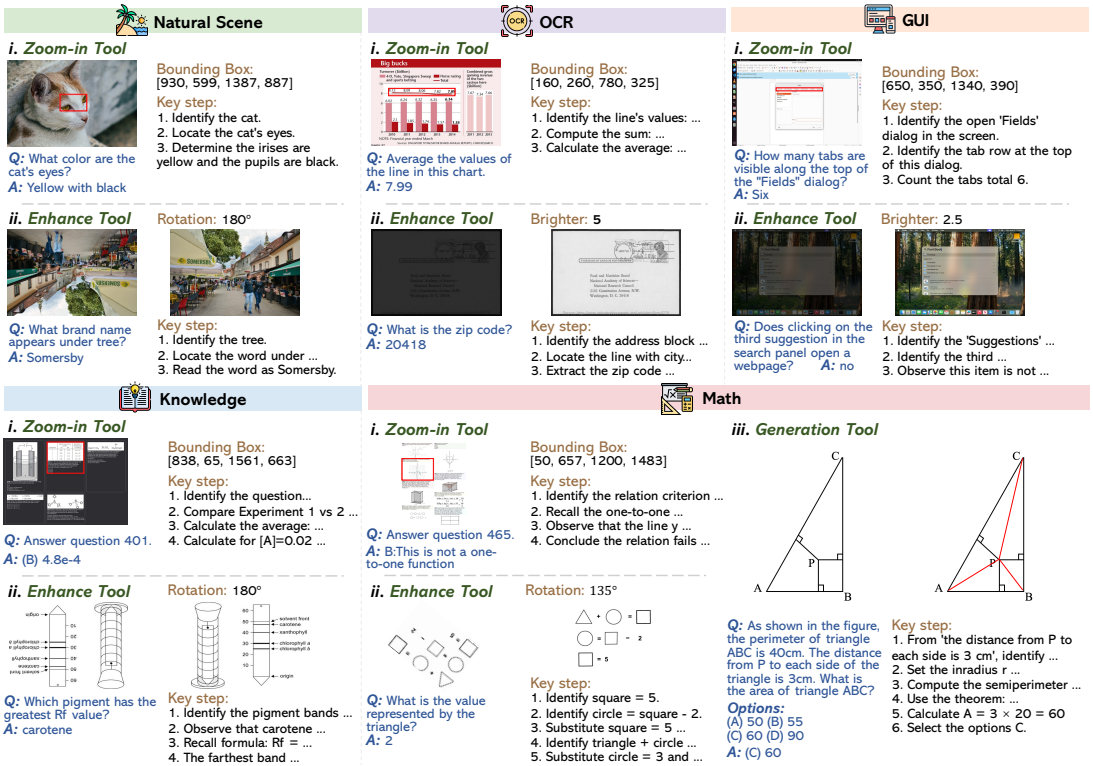


Figure 3: **An Overview of AdaptMMBench.** The benchmark contains data from five domains. Each domain includes samples requiring zooming and enhancement tools.

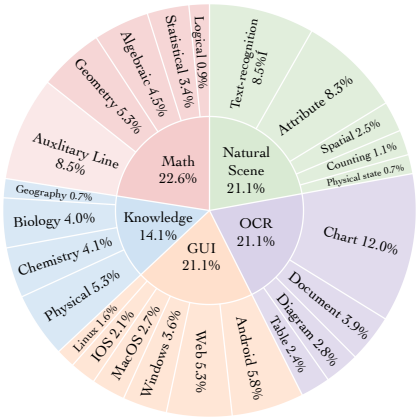


Figure 4: Domains and category of AdaptMMBench.

MMBench evaluates tools beyond zooming, including rotation correction, brightness adjustment, and contrast enhancement. Zooming and transformation samples are collected at a 5:2 ratio. We further collect 120 math samples requiring auxiliary-line generation, suggesting reasoning over self-generated images is an extension of the "think with image" paradigm, as shown in Fig. 4.

3.3 Annotation and Quality Control

Visual Tool & Key Step Annotation. We collect initial data from existing benchmarks, with annotators providing bounding-box annotations for key regions. Visual enhancement annotations, such as brightness, contrast, and rotation parameters, are not manually annotated. They are obtained by applying predefined perturbations to original images and recording the corresponding transformation parameters for recoverability. GPT-5 is used to generate key reasoning steps K, which are manually verified. These components form annotated quintuples (I, Q, A, E, K).

Quality Control. Benchmark quality is ensured through a multi-stage verification pipeline. First, three independent annotators cross-validate each QA pair to remove ambiguity and wrong data. Annotated image transformations and generated key reasoning steps are then reviewed by additional annotators for precision. Inaccurate instances are iteratively refined or re-annotated. This process ensures high-fidelity ground truth and key reasoning steps annotations. Additional statistics information are provided in App. D.

4 Evaluation Strategy

4.1 Evaluation Settings

We define three evaluation settings to systematically assess both mode selection and the reasoning process, controlled by minimal prompt modifications (detailed in App. H).

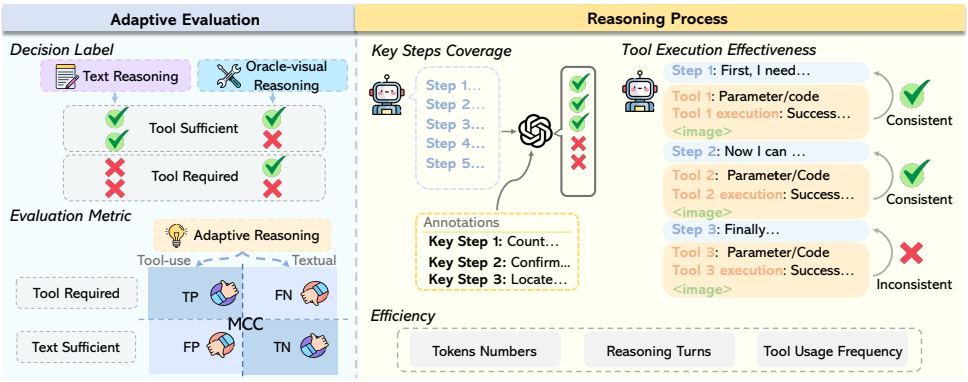


Figure 5: Evaluation pipeline for mode selection and reasoning process.

- **Text-Reasoning Setting:** Given (I, Q), the model relies on text reasoning over the given image, without invoking visual tools, providing a baseline for assessing tool necessity.
- **Adaptive Reasoning Setting:** Given (I, Q), the model adaptively selects between text reasoning and tool-augmented visual reasoning. This mode evaluates whether the model chooses an appropriate reasoning strategy and how effectively it carries out the resulting reasoning process.
- **Oracle-Visual Setting:** Given (I, Q, I_E), where I_E denotes gold-standard visual evidence from annotation E, the model performs text reasoning over the provided visual evidence, providing an upper-bound performance under perfect visual acquisition (App. H).

4.2 Reasoning Mode Selection Evaluation

Adaptive intelligence requires a model not only to answer correctly, but also to decide whether the information currently available is sufficient or whether additional evidence is needed. We therefore evaluate reasoning mode selection as a decision problem distinct from final answer prediction.

Under this principle, tool necessity is induced separately for each evaluated model. For a model m, we first run the Text-Reasoning setting and compare its answer with the dataset ground truth. A correct text-only answer induces a model-specific **Text-Sufficient** label, whereas an incorrect answer induces a **Tool-Required** label. We then compare this reference label with the same model's tool-use decision in the Adaptive setting to obtain TP, TN, FP, and FN. Specifically, TP and FN are Tool-Required cases with and without tool invocation, respectively, while TN and FP are Text-Sufficient cases without and with tool invocation, respectively.

These categories operationalize model-specific difficulty: Text-Sufficient and Tool-Required correspond to relatively easy and relatively difficult instances with respect to a model's own text-reasoning capability boundary, rather than universal easy/hard labels. Consequently, the same instance may receive different reference labels for different models. For Text-Sufficient cases, invoking tools is unnecessary and may introduce noise; for Tool-Required cases, the model should seek additional visual evidence through tools.

We use the **Text-Sufficient** and **Tool-Required** categories as reference labels for evaluating reasoning mode selection through the confusion matrix in Fig. 5. Further discussion is provided in App. A. Importantly, **Tool-Required** indicates that additional visual evidence is necessary for reliable response. Therefore, we construct our benchmark by selecting samples where richer visual evidence supports correct resolution. This is validated by the oracle setting, where introducing gold-standard visual information enables the SOTA model, Gemini-3-Pro, to achieve 100% accuracy (Tab. 4).

Matthews Correlation Coefficient (MCC). In reasoning mode selection, the proportions of text-sufficient and tool-required cases are model-specific, which can lead to varying degrees of class imbalance in the confusion matrix. To ensure robust evaluation under such imbalance, we report the Matthews correlation coefficient (MCC)

$$\text{MCC} = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)+\epsilon}}, \tag{2}$$

where ϵ is a small constant for numerical stability. MCC ranges from $[-1, 1]$, with 1 indicating perfect agreement with the optimal mode selection, 0 denoting the chance-level performance, and -1 indicating complete misalignment. Detailed analysis

is provided in App. B.3.1.

Prompt Robustness of Text Reasoning. We analyze the effects of minor prompt variations on text and adaptive reasoning. Specifically, we compare answers from the enforced text-reasoning prompt with cases where the model selects text reasoning under the adaptive prompt. Among adaptive-mode samples that choose text reasoning, only 2% produce final answers inconsistent with the enforced text-reasoning run. Details are provided in App. B.2. This indicates that the categories are robust to prompt variation, and the performance difference remains stable.

4.3 Reasoning Process Evaluation

We introduce three process-oriented metrics to evaluate reasoning coherence and tool execution fidelity. A trajectory $\mathcal{R}$ is formalized as an interleaved sequence of thought and tool invocations:

$$\mathcal{R} = \{(s_1, t_1), (s_2, t_2), \ldots, s_n\}, \tag{3}$$

where s_i is the thought at step i and $t_i \in \mathcal{T}$ represents the corresponding tool invocation. The trajectory terminates at the final reasoning step s_n, and produces the answer.

4.3.1 Key Steps Coverage

Following the evaluation paradigm of (Jiang et al., 2025), we assess whether a model's reasoning chain $\{s_i\}_{i=1}^{n}$ covers the essential human-annotated key steps K defined in Sec. 3.1. We employ GPT-5 as an evaluator to identify the presence of these key steps within the generated steps, and define the key step coverage as

$$\text{KCoverage} = \frac{1}{|K|} \max_{1 \le i \le |K|} \left[i \cdot \mathbb{I}(\text{Judge}_{\text{GPT-5}}(k_i, \{s_1, \ldots, s_n\}) = 1) \right]. \tag{4}$$

The judge semantically compares each key step with the complete trajectory, allowing steps to be merged, expressed implicitly, or stated using different wording. The maximization identifies the furthest covered key step and does not require every preceding step to be separately verbalized. Detailed explanation is provided in App. B.4.

4.3.2 Tool Execution Effectiveness

When a model invokes tools, we assess execution quality over the tool-calling segments of its trajectory, checking whether each invocation is semantically appropriate for its corresponding reasoning step, avoids redundant calls, follows tool-use rules, and completes without execution errors. The tool effectiveness is defined as

$$\text{Effect}_{\text{tool}} = \frac{1}{N_{\text{tool}}} \sum_{i=1}^{N_{\text{tool}}} \text{valid}_{\text{GPT-5}}(t_i \mid s_i), \tag{5}$$

where N_{tool} denotes the total number of tool invocations, $t_i \in \mathcal{T}$ is the tool invoked at step i, and $\text{valid}_{\text{GPT-5}}(\cdot) \in \{0, 1\}$ is a semantic validity judgment provided by GPT-5 (detailed in App. B.5).

4.3.3 Reasoning Efficiency

For efficiency, we report token numbers, reasoning turns, and tool usage frequency, collectively capturing the conciseness of reasoning and the computational cost of adaptive execution.

Table 2: Evaluation of mode selection across models. We report TP, TN, FP, FN, and MCC on the 1,300-sample subset using temperature $= 0$ for deterministic evaluation. Randomness robustness is analyzed in App. B.3.3. Best and second-best MCC scores in each category are highlighted in blue and green.

Model	TP ↑	TN ↑	FP ↓	FN ↓	MCC ↑
Open-Source Models					
PixelReasoner	202	500	279	319	0.03
Deepeyes	526	0	774	0	0.00
Thyme	16	795	23	466	0.01
PyVision	405	302	486	107	0.18
Deepeyes v2	522	9	765	4	0.02
AdaptVision	257	273	519	251	0.16
Qwen3-vl-8B-Instruct	225	447	464	164	0.06
Qwen3-vl-32B-Instruct	231	282	753	34	0.14
Qwen3-vl-235B-Instruct	181	548	515	56	0.22
Closed-Source Models					
GPT-5	354	500	409	37	**0.42**
Gemini-3-Pro	167	386	738	9	0.22

Table 3: Comprehensive evaluation of reasoning process, including key step coverage (Key Step Cov.), tool effectiveness (Tool Effect.), and efficiency. LLM judge reliability, randomness and human agreement validation are detailed in App. E.

Model	Key Step Cov. (%) ↑	Tool Effect. (%) ↑	Efficiency		
			Steps ↓	Tools ↓	Tokens ↓
PixelReasoner	67.91	62.99	1.37	0.37	**4004.83**
Deepeyes	65.57	55.17	2.01	1.69	7004.44
Thyme	69.65	57.44	**1.06**	**0.06**	6296.46
PyVision	74.10	62.19	2.56	1.56	6682.46
Deepeyes v2	69.96	59.89	2.09	1.08	6521.65
AdaptVision	65.56	85.55	1.43	0.44	4027.47
Qwen3-vl-8B	76.36	93.79	1.69	1.07	7366.83
Qwen3-vl-32B	81.47	91.42	2.25	1.27	6883.91
Qwen3-vl-235B	**82.27**	**93.93**	1.90	0.90	6744.54
Closed-Source Models (Observable Traces)					
GPT-5	80.13	92.36	2.37	2.73	**2069.05**
Gemini-3-Pro	81.22	93.23	1.02	0.82	4431.86

5 Experiments

We conduct a comprehensive quantitative evaluation of adaptive reasoning on AdaptMMBench,

focusing on three complementary dimensions: (i) reasoning mode selection capability, (ii) quality and efficiency of reasoning process, and (iii) accuracy across evaluation settings.

5.1 Experiment Setting

We evaluate a set of VLMs to establish baselines for AdaptMMBench. For closed-source models, we select GPT-5 (OpenAI, 2025a) and Gemini3 (Google DeepMind, 2025). For open-source models, we include the Qwen3-VL family (Bai et al., 2025) at multiple scales (8B, 32B, and 235B). In addition, we evaluate several specialized adaptive reasoning models, including DeepEyes (Zheng et al., 2025; Hong et al., 2025), PixelReasoner (Wang et al., 2025b), Thyme (Zhang et al., 2025b), PyVision (Zhao et al., 2025), and AdaptVision (Lin et al., 2025b). For all evaluated models, we follow the implementation details provided in their official codebases.

5.2 Reasoning Mode Selection Capability

Reasoning mode selection capabilities differ across models. In Tabs. 2 and 4, *mode selection capability does not exhibit a strong correlation with final task accuracy*. Although MCC uses model-relative labels, we discuss its cross-model comparability and robustness in App. B.3.2 and App. B.3.3, respectively. Tab. 4 reports results on 1,300 samples, while the 120 mathematical problems requiring auxiliary-line generation are analyzed separately in Tab. 5, as this task family depends on self-generated visual cues beyond the standard transformation tools in our main evaluation.

Reasoning mode selection captures a different aspect of model capability than final accuracy. GPT-5 attains the highest MCC of 0.42, indicating stronger mode-selection capability than Gemini-3-Pro, while Gemini-3-Pro achieves the best adaptive reasoning performance.

Strong reasoning mode selection is helpful, but final adaptive gains and efficiency also depend on overall model capability. Specifically, GPT-5 makes more correct tool-required decisions (TP: 354 vs. 167) and more correct tool-redundant decisions (TN: 500 vs. 386), while producing substantially fewer unnecessary tool invocations (FP: 409 vs. 738). Accordingly, GPT-5 obtains a larger adaptive gain in Tab. 4 (15.46 vs. 8.31). However, Gemini-3-Pro still attains higher final adaptive accuracy because its text reasoning is much stronger, while GPT-5 starts from a lower baseline, likely partly due to its compressed visual-token setting. Similarly, although AdaptVision exhibits relatively strong reasoning mode selection capability among open-source models, its performance in the adaptive setting does not substantially outperform the text-only setting, primarily because its key-step coverage is quite weak (as shown in Tab. 3). Additional evidence and detailed correlation analyses between diagnostic metrics and final accuracy are provided in App. C.

Imbalanced mode selection behavior is observed in some models. Several specialized adaptive models exhibit imbalanced mode selection behavior, either invoking tools excessively or rarely. For example, Deepeyes invokes tools for all 1,300 samples, while Deepeyes v2 invokes tools for 1,287 samples (99.0%). In contrast, Thyme triggers tool usage in only 39 samples (3.0%). Such imbalanced patterns are associated with lower mode selection performance, despite competitive accuracy.

5.3 Quality and Efficiency of the Reasoning Process

Tab. 3 evaluates key-step coverage, tool effectiveness, and efficiency. For closed-source models, this evaluation is restricted to externally observable traces and does not assess unexposed internal reasoning. Qwen3-VL-235B attains the strongest key-step coverage among open-source models, while GPT-5 and Gemini-3-Pro achieve comparable process quality on their observable traces. Larger open-source models also demonstrate stronger tool effectiveness, better aligning tool usage with reasoning intent.

Tool effectiveness varies with models. Qwen3 family shows strong performance, while some smaller models are less effective. This may stem from repeated or unnecessary tool calls, as well as code-based tool invocation in Deepeyes v2, Thyme, and PyVision, which introduces more complexity than the function-call interface used by Qwen models.

Token usage is not positively correlated with steps or tool calls. Considering efficiency, token usage varies across models and does not correspond to the number of reasoning steps or tool calls. For example, Thyme uses the fewest steps and tool invocations, yet consumes more tokens than PyVision, which has the most steps. This suggests that token cost is affected by image-input configurations, such as the maximum pixel budget, rather than solely by reasoning length or tool frequency.

Table 4: Accuracy across different domains under three evaluation settings. Results are reported on 1,300 Adapt-MMBench samples. The remaining 120 samples require visual generation ability and are analyzed separately in Tab. 5. * indicates that the model supports enhancement operations. "w/o enh." corresponds to the non-transformed subset, and "All" denotes overall accuracy.

Model	Mode	Natural scene		OCR		GUI		Knowledge		Math (w/o aux)		Overall Accuracy	
		w/o enh.	All	w/o enh.	All	w/o enh.	All	w/o enh.	All	w/o enh.	All	w/o enh.	All
Open-Source Models													
PixelReasoner	Text	60.56	59.00	77.87	70.67	64.26	62.67	59.88	57.50	46.37	43.50	62.81	59.92
	Adaptive	64.79	63.67	79.92	73.00	63.54	62.00	57.49	57.50	43.58	41.50	63.34	61.08
	Oracle	79.58	79.00	87.30	85.00	70.40	69.33	62.87	65.00	60.89	60.00	73.68	73.08
Deepeyes	Text	66.90	65.33	71.31	63.00	66.79	65.33	52.10	50.50	49.16	46.00	62.90	59.54
	Adaptive	66.55	65.33	72.54	65.00	64.62	63.00	58.08	55.50	52.51	49.00	63.94	60.69
	Oracle	80.99	80.33	80.33	78.33	75.09	75.33	68.86	70.50	64.25	63.50	75.07	74.62
Thyme*	Text	69.72	68.00	79.92	70.33	66.43	64.67	62.28	57.50	50.84	47.00	67.07	62.92
	Adaptive	74.30	72.67	83.20	74.67	70.04	67.67	55.09	50.00	51.96	49.50	68.90	64.92
	Oracle	86.62	86.00	86.89	85.33	75.09	75.00	67.07	68.50	65.36	65.00	77.76	77.38
PyVision*	Text	57.75	57.67	78.28	71.67	84.48	81.33	47.31	45.00	35.75	33.00	63.60	60.62
	Adaptive	60.92	59.67	82.38	76.00	87.73	84.67	68.26	62.50	36.87	35.00	69.24	65.85
	Oracle	96.13	94.00	81.15	80.33	92.42	90.67	55.09	58.00	51.40	50.00	79.15	77.77
Deepeyes v2*	Text	72.18	70.33	77.46	69.67	66.79	65.00	53.29	50.50	45.81	43.00	65.16	61.69
	Adaptive	72.89	71.00	78.69	70.00	71.48	69.67	59.88	55.00	50.28	49.50	68.38	64.69
	Oracle	83.80	83.00	86.48	85.00	77.62	77.67	60.48	61.50	63.69	63.00	76.37	75.85
AdaptVision	Text	64.08	62.33	79.10	71.00	62.82	61.00	56.29	56.50	51.40	48.00	63.86	60.92
	Adaptive	64.08	62.33	82.38	74.00	66.06	64.33	60.48	55.50	53.63	49.50	66.29	62.46
	Oracle	83.80	83.00	89.34	88.00	71.48	72.00	73.05	74.50	72.63	72.00	78.71	78.62
Qwen3-vl -8B-Instruct	Text	72.54	69.33	83.61	73.67	74.01	72.00	77.84	71.50	65.36	61.50	74.89	70.08
	Adaptive	68.31	67.00	86.07	77.00	75.09	72.67	86.83	80.00	69.27	65.00	76.54	72.31
	Oracle	89.79	89.00	93.44	90.33	83.03	81.67	86.23	84.50	83.24	83.50	87.40	86.08
Qwen3-vl -32B-Instruct	Text	71.48	69.00	93.03	84.00	87.73	84.00	91.02	84.50	82.68	77.50	84.54	79.62
	Adaptive	77.46	75.67	95.49	87.33	88.09	83.67	90.42	85.50	83.24	78.50	86.62	82.15
	Oracle	95.77	95.00	97.95	97.33	90.97	87.67	98.80	98.50	94.97	95.00	95.40	94.38
Qwen3-vl -235B-Instruct	Text	73.94	70.67	93.44	84.67	90.61	87.33	91.02	85.50	88.83	82.00	86.88	81.77
	Adaptive	77.46	74.67	94.26	86.00	90.97	87.67	97.60	91.50	84.92	79.00	88.36	83.54
	Oracle	94.72	93.00	97.54	96.33	99.64	97.67	98.20	98.00	97.77	97.50	97.48	96.31
Closed-Source Models													
GPT-5*	Text	63.03	62.67	87.70	80.00	87.73	85.33	73.05	67.50	48.04	45.00	73.33	69.92
	Adaptive	79.23	77.67	95.90	88.67	93.14	91.33	95.21	89.00	81.01	79.50	88.71	85.38
	Oracle	98.94	97.33	99.18	96.33	98.19	97.00	100.00	98.00	98.88	96.50	98.96	97.00
Gemini-3-Pro*	Text	73.59	72.00	95.49	92.33	95.67	94.67	93.41	90.50	86.59	83.00	88.44	86.46
	Adaptive	**90.14**	**88.67**	**97.54**	**95.33**	**97.47**	**97.67**	**98.80**	**98.50**	**97.77**	**95.00**	**95.92**	**94.77**
	Oracle	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00

5.4 Accuracy across Evaluation Settings

We analyze model performance across evaluation settings, including text, adaptive, and oracle tool reasoning, as shown in Tab. 4 and Tab. 5.

Significant performance gap between adaptive and oracle reasoning. As shown in Tab. 4, adaptive reasoning often achieves performance close to or slightly better than text reasoning, suggesting that additional visual evidence can reduce uncertainty and improve accuracy. However, adaptive reasoning still remains below the oracle setting, with an oracle-adaptive gap ranging from 5.23 to 16.16 percentage points across models. We further analyze the causes of this adaptive-oracle gap of GPT-5 in Sec. 5.5. This gap reflects multiple coupled factors, including incorrect mode selection, multi-step visual noise, and residual model limitations, but most of the recoverable gap stems from imperfect tool invocation.

Generation-Based Tools Are Beneficial for Certain Tasks. We conduct an exploratory analysis on auxiliary-line generation tasks, as shown in Tab. 5. For most models in our evaluation, adaptive reasoning yields limited or negative gains over text reasoning, whereas oracle access to ground-truth auxiliary lines brings substantial improvements. Additional experiments with visual-generation models (App. B.6) show that visual auxiliary information provides the main spatial benefit, while textual descriptions support construction planning. Misleading auxiliary evidence consistently degrades performance, and the remaining gaps reveal bottlenecks in both tool-based construction and downstream visual reasoning.

5.5 Error Analysis

We analyze GPT-5 errors under adaptive mode to understand the gap between adaptive and oracle-

Table 5: Experimental results on geometric auxiliary-line problems across different Evaluation settings.

Model	Text Acc	Adaptive Acc	Oracle Acc
Open-Source VLMs			
Thyme	21.67	21.67	24.17
PyVision	15.83	29.17	32.50
Deepeyes v2	19.17	19.17	25.83
Qwen3-vl-8B	50.00	46.67	62.50
Qwen3-vl-32B	63.33	58.33	79.17
Qwen3-vl-235B	62.50	68.33	84.17
Closed-Source Models			
Gemini-3-Pro	**85.00**	78.33	**94.17**
GPT-5	75.00	**86.67**	89.17

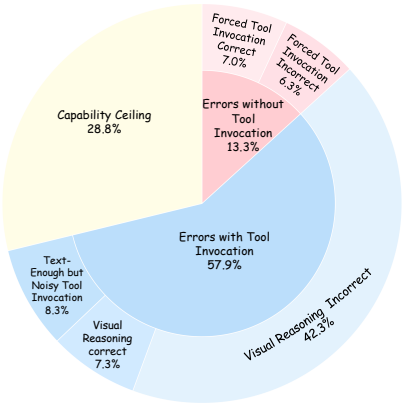


Figure 6: Error Analysis on GPT-5.

visual reasoning. As shown in Fig. 6, 42.3% of errors come from visual reasoning failures, such as incorrect zooming or image transformations, and 8.3% from incorrect mode selection, where the model invokes tools despite text reasoning being sufficient. Another 7.3% occur after correct visual reasoning, suggesting that intermediate images in multi-step reasoning may introduce noise. For no-tool cases, forced tool invocation corrects 7.0% of errors, while 6.3% remain incorrect; the remaining 28.8% exceed GPT-5's capability. Overall, among the 71.2% of errors recovered by oracle-visual mode, directly tool-related factors account for 71.1% of the recoverable gap, indicating that imperfect tool invocation is the main bottleneck.

6 Conclusion

We present AdaptMMBench, a benchmark for adaptive multimodal reasoning in VLMs. AdaptMMBench defines text-sufficient and tool-required cases in a model-specific manner and evaluates mode selection, reasoning quality, and efficiency. Results show that accuracy and MCC capture complementary capabilities, while analyses of reasoning quality and efficiency guide improvements in VLM performance and efficiency. We hope AdaptMMBench can support future work on VLMs that decide when to use tools, execute them effectively, and reason efficiently over multimodal evidence.

Limitations

AdaptMMBench is currently limited to static think-with-image tasks. Future work can extend the benchmark to video-based settings, where adaptive reasoning must decide when to inspect temporal evidence, track objects, or invoke frame-level tools. AdaptMMBench also focuses on a predefined set of visual tools, including zooming, enhancement, and auxiliary-line generation; broader tool ecosystems and more open-ended agentic actions remain important directions. Each instance currently contains a single reference key-step sequence. Although our semantic, max-based evaluator accommodates merged, implicit, or skipped intermediate steps, it may still underestimate a substantially different but valid reasoning path. For closed-source models, process evaluation is restricted to externally observable traces and cannot assess unexposed internal reasoning. Finally, although we cover several major domains, the benchmark can be further expanded to specialized scenarios such as 3D understanding, medical imaging, and other high-stakes visual reasoning domains.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant No. 62406009, the Opening Project of the State Key Laboratory of General Artificial Intelligence under Grant No. SKLAGI2025OP07, the Fundamental Research Funds for the Central Universities under Grant No. 2025CX11016, and the Key Research Platforms and Projects of the Guangdong Provincial Department of Education under Grant No. 2023ZDZX1034.

References

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. 2024. Are we on the right way for evaluating large vision-language

models? *Advances in Neural Information Processing Systems*, 37:27056–27087.

Ethan Chern, Zhulin Hu, Steffi Chern, Siqi Kou, Jiadi Su, Yan Ma, Zhijie Deng, and Pengfei Liu. 2025. Thinking with generated images. *arXiv preprint arXiv:2505.22525*.

Yong Xien Chng, Tao Hu, Wenwen Tong, Xueheng Li, Jiandong Chen, Haojia Yu, Jiefan Lu, Hewei Guo, Hanming Deng, Chengjun Xie, Gao Huang, Dahua Lin, and Lewei Lu. 2025. Sensenova-mars: Empowering multimodal agentic reasoning and search via reinforcement learning. *arXiv preprint arXiv:2512.24330*.

Chengqi Duan, Kaiyue Sun, Rongyao Fang, Manyuan Zhang, Yan Feng, Ying Luo, Yufang Liu, Ke Wang, Peng Pei, Xunliang Cai, Hongsheng Li, Yi Ma, and Xihui Liu. 2025. CodePlot-CoT: Mathematical visual reasoning by thinking with code-driven images. *arXiv preprint arXiv:2510.11718*.

Google DeepMind. 2025. A New Era of Intelligence with Gemini 3.

Shasha Guo, Liang Pang, Xi Wang, Yanling Wang, Huawei Shen, and Jing Zhang. 2025a. Geovlmath: Enhancing geometry reasoning in vision-language models via cross-modal reward for auxiliary line creation. *arXiv preprint arXiv:2510.11020*.

Xingang Guo, Utkarsh Tyagi, Advait Gosai, Paula Vergara, Jayeon Park, Ernesto Gabriel Hernández Montoya, Chen Bo Calvin Zhang, Bin Hu, Yunzhong He, Bing Liu, and Rakshith Sharma Srinivasa. 2025b. Beyond seeing: Evaluating multimodal LLMs on tool-enabled image perception, transformation, and reasoning. *arXiv preprint arXiv:2510.12712*.

Ziyu Guo, Renrui Zhang, Hao Chen, Jialin Gao, Dongzhi Jiang, Jiaze Wang, and Pheng-Ann Heng. 2025c. Sciverse: Unveiling the knowledge comprehension and visual reasoning of lmms on multi-modal scientific problems. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 19683–19704.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, Guohai Xu, and Xing Yu. 2025. Deepeyesv2: Toward agentic multimodal model. *arXiv preprint arXiv:2511.05271*.

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

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

Dongzhi Jiang, Renrui Zhang, Ziyu Guo, Yanwei Li, Yu Qi, Xinyan Chen, Liuhui Wang, Jianhan Jin, Claire Guo, Shen Yan, Bo Zhang, Chaoyou Fu, Peng Gao, and Hongsheng Li. 2025. Mme-cot: Benchmarking chain-of-thought in large multimodal models for reasoning quality, robustness, and efficiency. *arXiv preprint arXiv:2502.09621*.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, and 1 others. 2023. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026.

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

Chengzu Li, Wenshan Wu, Huanyu Zhang, Yan Xia, Shaoguang Mao, Li Dong, Ivan Vulić, and Furu Wei. 2025a. Imagine while reasoning in space: Multimodal visualization-of-thought. *arXiv preprint arXiv:2501.07542*.

Kaican Li, Lewei Yao, Jiannan Wu, Tiezheng Yu, Jierun Chen, Haoli Bai, Lu Hou, Lanqing Hong, Wei Zhang, and Nevin L Zhang. 2025b. Insight-o3: Empowering multimodal foundation models with generalized visual search. *arXiv preprint arXiv:2512.18745*.

Ming Li, Jike Zhong, Shitian Zhao, Haoquan Zhang, Shaoheng Lin, Yuxiang Lai, Chen Wei, Konstantinos Psounis, and Kaipeng Zhang. 2025c. Tir-bench: A comprehensive benchmark for agentic thinking-with-images reasoning. *arXiv preprint arXiv:2511.01833*.

Xuchen Li, Xuzhao Li, Jiahui Gao, Renjie Pi, Shiyu Hu, and Wentao Zhang. 2025d. Look less, reason more: Rollout-guided adaptive pixel-space reasoning. *arXiv preprint arXiv:2510.01681*.

Zejun Li, Yingxiu Zhao, Jiwen Zhang, Siyuan Wang, Yang Yao, Runzhou Zhao, Jun Song, Bo Zheng, and Zhongyu Wei. 2025e. Mixture-of-visual-thoughts: Exploring context-adaptive reasoning mode selection for general visual reasoning. *arXiv preprint arXiv:2509.22746*.

Huawei Lin, Tong Geng, Zhaozhuo Xu, and Weijie Zhao. 2025a. Vtbench: Evaluating visual tokenizers for autoregressive image generation. *arXiv preprint arXiv:2505.13439*.

Zichuan Lin, Yicheng Liu, Yang Yang, Lvfang Tao, and Deheng Ye. 2025b. Adaptvision: Efficient vision-language models via adaptive visual acquisition. *arXiv preprint arXiv:2512.03794*.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. 2024. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, pages 216–233. Springer.

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

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

Ahmed Masry, Mohammed Saidul Islam, Mahir Ahmed, Aayush Bajaj, Firoz Kabir, Aaryaman Kartha, Md Tahmid Rahman Laskar, Mizanur Rahman, Shadikur Rahman, Mehrad Shahmohammadi, Megh Thakkar, Md Rizwan Parvez, Enamul Hoque, and Shafiq Joty. 2025. Chartqapro: A more diverse and challenging benchmark for chart question answering. *arXiv preprint arXiv:2504.05506*.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. 2021. Docvqa: A dataset for vqa on document images. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2200–2209.

OpenAI. 2025a. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*.

OpenAI. 2025b. Openai o3 and o4-mini system card. Technical report, OpenAI.

Runqi Qiao, Qiuna Tan, Guanting Dong, Minhui Wu, Chong Sun, Xiaoshuai Song, Jiapeng Wang, Zhuoma GongQue, Shanglin Lei, YiFan Zhang, Zhe Wei, Miaoxuan Zhang, Runfeng Qiao, Xiao Zong, Yida Xu, Peiqing Yang, Zhimin Bao, Muxi Diao, Chen Li, and Honggang Zhang. 2025a. 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.

Runqi Qiao, Qiuna Tan, Minghan Yang, Guanting Dong, Peiqing Yang, Shiqiang Lang, Enhui Wan, Xiaowan Wang, Yida Xu, Lan Yang, Chong Sun, Chen Li, and Honggang Zhang. 2025b. V-Thinker: Interactive thinking with images. *arXiv preprint arXiv:2511.04460*.

Chenrui Shi, Zedong Yu, Zhi Gao, Ruining Feng, Enqi Liu, Yuwei Wu, Yunde Jia, Liuyu Xiang, Zhaofeng He, and Qing Li. 2025. Gui knowledge bench: Revealing the knowledge gap behind vlm failures in gui tasks. *arXiv preprint arXiv:2510.26098*.

Zhaochen Su, Jincheng Gao, Hangyu Guo, Zhenhua Liu, Lueyang Zhang, Xinyu Geng, Shijue Huang, Peng Xia, Guanyu Jiang, Cheng Wang, Yue Zhang, Yi R. Fung, and Junxian He. 2026. Agentvista: Evaluating multimodal agents in ultra-challenging realistic visual scenarios. *arXiv preprint arXiv:2602.23166*.

Chaoyang Wang, Kaituo Feng, Dongyang Chen, Zhongyu Wang, Zhixun Li, Sicheng Gao, Meng Meng, Xu Zhou, Manyuan Zhang, Yuzhang Shang, and Xiangyu Yue. 2025a. Adatooler-v: Adaptive tool-use for images and videos. *arXiv preprint arXiv:2512.16918*.

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

Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, Wei Yu, and Dacheng Tao. 2025c. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 7907–7915.

Qianshan Wei, Yishan Yang, Siyi Wang, Jinglin Chen, Binyu Wang, Jiaming Wang, Shuang Chen, Zechen Li, Yang Shi, Yuqi Tang, Weining Wang, Yi Yu, Chaoyou Fu, Qi Li, and Yi-Fan Zhang. 2026. Agentic-mme: What agentic capability really brings to multimodal intelligence? *arXiv preprint arXiv:2604.03016*.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. 2025a. Webwalker: Benchmarking llms in web traversal. *arXiv preprint arXiv:2501.07572*.

Junfei Wu, Jian Guan, Kaituo Feng, Qiang Liu, Shu Wu, Liang Wang, Wei Wu, and Tieniu Tan. 2025b. Reinforcing spatial reasoning in vision-language models with interwoven thinking and visual drawing. *arXiv preprint arXiv:2506.09965*.

Penghao Wu and Saining Xie. 2024. V?: Guided visual search as a core mechanism in multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13084–13094.

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

Weiye Xu, Jiahao Wang, Weiyun Wang, Zhe Chen, Wengang Zhou, Aijun Yang, Lewei Lu, Houqiang Li, Xiaohua Wang, Xizhou Zhu, Wenhai Wang, Jifeng Dai, and Jinguo Zhu. 2025. Visulogic: A benchmark for evaluating visual reasoning in multi-modal large language models. *arXiv preprint arXiv:2504.15279*.

Dongchao Yang, Songxiang Liu, Disong Wang, Yuanyuan Wang, Guanglu Wan, and Helen Meng. 2025a. Omni-autothink: Adaptive multimodal reasoning via reinforcement learning. *arXiv preprint arXiv:2512.03783*.

Shuo Yang, Caren Han, Siwen Luo, and Eduard Hovy. 2025b. Magic-vqa: Multimodal and grounded inference with commonsense knowledge for visual question answering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 16967–16986.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, and 3 others. 2024. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9556–9567.

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

Xintong Zhang, Zhi Gao, Bofei Zhang, Pengxiang Li, Xiaowen Zhang, Yang Liu, Tao Yuan, Yuwei Wu, Yunde Jia, Song-Chun Zhu, and Qing Li. 2025a. Chain-of-focus: Adaptive visual search and zooming for multimodal reasoning via rl. *arXiv preprint arXiv:2505.15436*.

Yi-Fan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, Haonan Fan, Kaibing Chen, Jiankang Chen, Haojie Ding, Kaiyu Tang, Zhang Zhang, Liang Wang, Fan Yang, Tingting Gao, and Guorui Zhou. 2025b. Thyme: Think beyond images. *arXiv preprint arXiv:2508.11630*.

Yifan Zhang, Huanyu Zhang, Haochen Tian, Chaoyou Fu, Shuangqing Zhang, Junfei Wu, Feng Li, Kun Wang, Qingsong Wen, Zhang Zhang, Liang Wang, and Rong Jin. 2025c. Mme-realworld: Could your multimodal LLM challenge high-resolution real-world scenarios that are difficult for humans? In *The Thirteenth International Conference on Learning Representations (ICLR)*.

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

Shitian Zhao, Haoquan Zhang, Shaoheng Lin, Ming Li, Qilong Wu, Kaipeng Zhang, and Chen Wei. 2025. Pyvision: Agentic vision with dynamic tooling. *arXiv preprint arXiv:2507.07998*.

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

Xuanyu Zhu, Yuhao Dong, Rundong Wang, Yang Shi, Zhipeng Wu, Yinlun Peng, YiFan Zhang, Yihang Lou, Yuanxing Zhang, Ziwei Liu, Yan Bai, and Yuan Zhou. 2026. Vtc-bench: Evaluating agentic multimodal models via compositional visual tool chaining. *arXiv preprint arXiv:2603.15030*.

A Category Validity

This section clarifies the validity of our adaptive categories, especially the *Tool-Required* category. AdaptMMBench is specifically designed to study "thinking with images". Accordingly, the collected samples are mostly question-answering instances that require fine-grained visual evidence or visual enhancements, such as contrast adjustment and rotation correction. These samples are grounded in visually rich images where richer local evidence can change the reasoning outcome.

To verify this design, we further analyze cases where text-only reasoning fails and measure how often they are recovered under adaptive and oracle-visual reasoning. The oracle setting factors out errors caused by imperfect tool invocation and directly tests whether richer visual evidence is sufficient for solving the sample. As shown in Table 6, a substantial portion of text-only errors can be recovered once richer visual evidence is introduced, with Gemini-3-Pro reaching 100.00% recovery under oracle visual evidence.

At the same time, *Tool-Required* does not mean that every evaluated model is guaranteed to solve the sample after invoking tools. A sample is considered *Tool-Required* when the model should recognize that text-only reasoning is insufficient and acquire additional visual evidence through tools. This is an operational, capability-boundary-aware notion of reasoning-path selection, rather than a claim that tool use must fully resolve the task for every model.

In practice, text-only errors may also arise from limited commonsense, weak logical reasoning, missing knowledge, or output instability. Nevertheless, under our benchmark setting, invoking tools remains the more appropriate reasoning path when text-only reasoning fails, because visual operations provide the primary mechanism for introducing additional evidence. Richer visual details may reduce uncertainty and improve the reliability of the final answer. We therefore define these cases as *Tool-Required* not simply because text-only reasoning fails, but because tool use provides systematic and verifiable information gain.

B Evaluation Details

B.1 Supplementary Evaluation Settings

While the standard definitions of the Text-Reasoning and Adaptive Reasoning settings are described in Sec. 4, this section provides critical methodological details and supplementary context regarding the implementation and purpose of these settings.

Design and Implementation of the Oracle-Visual Setting. The Oracle-Visual setting is an idealized diagnostic control, not a realistic deployment scenario. Its purpose is not to estimate the performance that an autonomous tool-using model can directly achieve, but to isolate whether the relevant visual evidence is sufficient once it is made available. Concretely, the model is given the gold-standard visual evidence I_E, such as the annotated crop or the correctly transformed image region, and therefore does not need to perform target localization, tool selection, or parameter generation. This design deliberately removes errors from the evidence-acquisition stage and allows us to evaluate the model's reasoning and recognition ability conditioned on perfect visual evidence.

Under this interpretation, the Oracle-Visual result should be read as a diagnostic upper bound under perfect evidence acquisition. The Adaptive–Oracle gap is therefore not a claim of directly recoverable deployment performance. Instead, it quantifies the remaining challenge introduced by autonomous tool use, including noisy localization, incorrect transformation parameters, incomplete evidence, and multi-step visual-processing errors. A high oracle score together with a lower adaptive score indicates that the sample is solvable with appropriate visual evidence, while the current adaptive system fails to acquire or use that evidence reliably. Conversely, low oracle performance suggests that the bottleneck cannot be explained by tool execution alone and may also involve residual recognition or reasoning limitations. In this way, Oracle-Visual helps separate evidence sufficiency from tool-execution reliability and provides a controlled reference point for interpreting adaptive failures.

This diagnostic design allows us to systematically disentangle the different capability components of adaptive multimodal reasoning. Below, we provide further evaluation details and statistical evidence supporting the cross-model comparability of our Matthews Correlation Coefficient (MCC) metric, demonstrating that our results are robust against model-specific label prevalence, tool-use frequency, and sampling noise.

Table 6: Recovery of text-only errors under adaptive and oracle-visual reasoning.

Model	Text-Only Errors	Adaptive Correct	Adaptive Recovery	Oracle Correct	Oracle Recovery
Qwen3-vl-235B	237	95	40.08%	200	84.39%
GPT-5	391	230	58.82%	355	90.79%
Gemini-3-Pro	176	124	70.45%	176	100.00%

B.2 Prompt Robustness and Operational Delta Prompting

To ensure that the performance gaps between the Text-Reasoning and Adaptive settings are not artifacts of general prompt engineering variations, we employ a strict "delta prompting" protocol. Specifically, instead of authoring two entirely different system prompts, the Text-Reasoning prompt is generated by taking the model's standard Adaptive prompt and applying a minimal delta deletion: removing the JSON schemas of tool definitions, tool descriptions, and instructions regarding the search process, while keeping all core task instructions identical. This strict control ensures that performance differences are purely driven by the presence or absence of active tool-based perception, rather than varying levels of prompt detail or formatting constraints.

To empirically validate the sensitivity of our label construction to this delta prompting protocol, we compare two outputs for each sample: (i) the answer generated under the enforced Text-Reasoning prompt, and (ii) the answer generated in the Adaptive setting when the model autonomously selects text reasoning instead of invoking tools. The inconsistency rate is computed over this autonomous text-reasoning subset to measure whether the same reasoning path changes its final answer due to the prompt delta. Our evaluation shows that this inconsistency rate remains extremely low, generally within 0–5%, with an average of 2.66% across all evaluated models. This confirms that the model's text-reasoning behavior is highly stable under the minimal prompt delta, supporting the robustness of our capability-relative label construction.

B.3 Validity and Justification of the MCC Metric

B.3.1 Why MCC for Mode Selection

We use MCC as the primary metric for reasoning mode selection because this task is a binary decision problem with model-dependent class imbalance. As shown in Table 2, the confusion matrices are not formed under a fixed global label distribution: for each model, Tool-Required and Tool-Redundant labels are induced by its own text-reasoning capability. Table 7 makes this imbalance explicit by aggregating the TP/TN/FP/FN entries in Table 2 into the number of positive samples, $P = TP + FN$, and negative samples, $N = TN + FP$, for each model. The Tool-Required prevalence ranges from 13.5% for Gemini-3-Pro to 40.5% for Deepeyes and Deepeyes v2, showing that different models face substantially different capability-relative class distributions. Meanwhile, the decision marginals are also highly skewed for some systems: Deepeyes invokes tools on all 1,300 samples, while Deepeyes v2 invokes tools on 1,287 samples. These near-degenerate policies can appear superficially reasonable under imbalanced labels, but they do not demonstrate selective reasoning mode choice. A suitable metric should therefore reward correct decisions on both classes, penalize degenerate policies such as always invoking tools or never invoking tools, and remain meaningful when the class distribution changes.

Table 7: Model-specific class imbalance and tool-use prevalence derived from Table 2. $P = TP + FN$ denotes Tool-Required samples, and $N = TN + FP$ denotes Tool-Redundant samples.

Model	P	N	Tool-Required (%)	Tool-Use (%)
PixelReasoner	521	779	40.1	37.0
Deepeyes	526	774	40.5	100.0
Thyme	482	818	37.1	3.0
PyVision	512	788	39.4	68.5
Deepeyes v2	526	774	40.5	99.0
AdaptVision	508	792	39.1	59.7
Qwen3-vl-8B-Instruct	389	911	29.9	53.0
Qwen3-vl-32B-Instruct	265	1035	20.4	75.7
Qwen3-vl-235B-Instruct	237	1063	18.2	53.5
GPT-5	391	909	30.1	58.7
Gemini-3-Pro	176	1124	13.5	69.6

Accuracy is not sufficient for this setting because it is strongly affected by class prevalence. If most samples are Text-Sufficient for a strong text-reasoning model, a policy that rarely invokes tools may obtain a deceptively high accuracy without genuinely recognizing when tools are needed. Conversely, for models with many Tool-Required samples, frequent tool invocation can appear favorable even if it lacks item-level selectivity. Thus, ac-

curacy can conflate adaptive decision quality with the model-specific distribution of capability boundaries.

F1 score is also less appropriate because it emphasizes performance on a designated positive class, typically Tool-Required, while ignoring true negatives. In our evaluation, both types of correct decisions are important: invoking tools on Tool-Required samples captures the ability to seek missing evidence, while avoiding tools on Tool-Redundant samples captures the ability to prevent unnecessary computation and visual noise. A metric that downplays true negatives would underevaluate this second capability, which is central to adaptive reasoning.

AUC is designed for ranking or threshold-based evaluation from continuous confidence scores. Our benchmark evaluates the actual adaptive decision made by the model during inference, namely whether it invokes tools or remains in text reasoning mode. Many evaluated systems expose only this discrete behavior rather than calibrated confidence scores over tool necessity. Applying AUC to such binary decisions would either require additional assumptions about unavailable scores or reduce to a less informative ranking comparison.

MCC better matches our evaluation goal. It incorporates all four confusion-matrix entries, normalizes by the marginal frequencies of both the ground-truth labels and model decisions, and yields a balanced measure of agreement between tool necessity and tool invocation. As a result, MCC remains informative under class imbalance and assigns near-zero scores to non-selective strategies such as always-tool and never-tool policies. This makes MCC a more suitable metric for evaluating whether a model can recover its own capability-relative mode-selection boundary.

B.3.2 Cross-Model Comparability of MCC

This subsection explains why the MCC values in Table 2 can be compared across models, even though the adaptive categories are constructed relative to each model's capability boundary.

Although the Tool-Required and Tool-Redundant categories are defined relative to each model's text-only capability, the resulting MCC scores remain comparable across models. All models are evaluated on the same question set, with the same label-construction rule and the same decision objective. The only model-dependent component is the capability boundary induced by each model's own text-only performance.

A universal static label set would be less appropriate for adaptive mode selection, because tool necessity is inherently model-relative. We therefore construct a capability-relative Tool-Required / Tool-Redundant boundary for each model under a shared protocol, and evaluate how accurately the model recovers its own boundary. Thus, MCC measures model-relative decision quality under a normalized evaluation protocol, making cross-model comparison meaningful.

Proposition B.1 (Cross-model comparability of capability-relative MCC)**.** *Let $\mathcal{D}$ be the fixed evaluation set shared by all models. For each model m, let $Y_m \in \{0,1\}$ denote the capability-relative tool-necessity label induced by the same text-reasoning protocol, where $Y_m = 1$ indicates Tool-Required and $Y_m = 0$ indicates Tool-Redundant. Let $Z_m \in \{0,1\}$ denote the model's adaptive decision, where $Z_m = 1$ means invoking tools and $Z_m = 0$ means using text reasoning. If the label-construction rule and adaptive decision space are identical across models, and both classes have non-zero counts, then the MCC for model m equals the Pearson correlation $\mathrm{Corr}(Y_m, Z_m)$. Therefore, MCC measures the standardized agreement between a model's adaptive decision and its own capability boundary, making MCC values comparable as normalized capability-boundary recovery scores.*

Proof. For binary variables, the Pearson correlation between Y_m and Z_m can be written in terms of the confusion matrix entries:

$$\mathrm{Corr}(Y_m, Z_m) = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{6}$$

This is exactly the Matthews Correlation Coefficient. Since correlation is standardized by the marginal frequencies of both variables, it evaluates agreement after normalizing for class imbalance. Under our shared protocol, the only model-dependent component is the capability boundary Y_m; the decision objective remains the same for all models, namely to recover whether text reasoning is sufficient for that model. Thus, MCC values compare the same normalized quantity across models: the quality of capability-relative mode selection.

By-construction property versus empirical finding. By construction, MCC measures capability-relative alignment between a model's adaptive tool-use decisions and its empirically observed text-only

sufficiency, whereas final accuracy measures task success. The distinction between these objectives follows from the metric design; it is not itself an empirical discovery. The empirical finding is that the two metrics produce different model rankings: GPT-5 achieves a higher MCC than Gemini-3-Pro, while Gemini-3-Pro achieves substantially higher final adaptive accuracy. Thus, the observed ranking reversal shows that mode-selection alignment captures information not reflected by final accuracy alone.

Controlled comparison with shared labels. To test whether this ranking is driven by model-specific labels, we compare GPT-5 and Gemini-3-Pro on 1,005 questions for which both models induce the same reference labels. This shared-label set contains 136 Tool-Required and 869 Text-Sufficient questions. GPT-5 retains a higher MCC than Gemini-3-Pro (0.331 vs. 0.278). We further evaluate a class-balanced subset containing 136 Tool-Required and 136 Text-Sufficient questions. GPT-5 again obtains a higher MCC (0.518 vs. 0.466), whereas Gemini-3-Pro achieves substantially higher answer accuracy (82.88% vs. 68.64%). Therefore, the opposite MCC and accuracy rankings persist after controlling for the evaluated questions, reference labels, and class proportions.

Item-level bootstrap analysis. We additionally assess sampling uncertainty through an item-level paired bootstrap over the benchmark instances, recomputing each model's MCC and their difference for every resample. GPT-5 obtains an MCC of 0.424 with a 95% confidence interval of $[0.383, 0.464]$, while Gemini-3-Pro obtains 0.217 with a 95% confidence interval of $[0.183, 0.249]$. The paired MCC difference is 0.207 with a 95% confidence interval of $[0.159, 0.255]$ ($p < 0.0001$). These results support the stability of the MCC difference on the evaluated benchmark; they should not be interpreted as evidence of generalization beyond this evaluation distribution.

B.3.3 Metric Stability: Robustness to Tool Sets and Difficulty Thresholds

We further examine the stability of the Matthews Correlation Coefficient (MCC) under two potential confounding factors: the composition and scale of the available tool set, and the specific threshold used to define task difficulty. Our analysis demonstrates that MCC is a robust and highly stable evaluation metric for benchmarking reasoning mode selection, producing consistent and invariant model rankings.

Decoupling from Tool-Set Scale and Tool Types. A model's final adaptive accuracy is naturally influenced by the scale and composition of the available tool set, as larger tool sets introduce downstream execution complexities (e.g., redundant tool calls, execution failures, or noisy intermediate states). However, a key design strength of our evaluation framework is that the *MCC metric itself remains stable and unaffected by downstream tool-set variations*. Because the Tool-Required and Tool-Redundant ground-truth labels are induced solely by the model's baseline text-reasoning capability, the target decision boundary is completely determined by the model's intrinsic knowledge boundaries, independent of the external tool set. MCC evaluates the model's pure meta-cognitive decision of *whether* to seek external evidence, isolating this selective capability from the practical execution of specific tools. Consequently, while varying the tool set may change the final accuracy or tool-conditioned success rate, it does not alter the model's underlying capability boundary or its decision calibration. The MCC metric thus serves as a robust, invariant measure of reasoning mode selection, successfully decoupling the cognitive choice of tool invocation from downstream execution noise.

Robustness of Mode Selection Evaluation

We use temperature $= 0$ in the main evaluation, so decoding is deterministic and one decode per item is sufficient. This choice is made for both evaluation stability and efficiency. To further assess robustness, we additionally test a stochastic setting with temperature $= 0.7$ and report the resulting MCC variation in Table 8.

Robustness to Difficulty Thresholds and Rank Stability. In our main evaluation, the Tool-Required label is defined in a model-specific, capability-relative manner based on whether text-only reasoning fails for that specific model. To prove that our evaluation conclusions and model rankings are not sensitive to this specific definition, we perform a rigorous robustness check using a cross-model difficulty score:

$$d(x) = \frac{1}{M} \sum_{m=1}^{M} \mathbb{1}[\text{model } m \text{ fails under text-only}],$$

where M is the total number of evaluated models. We define thresholded difficult examples as

Table 8: Robustness of mode selection evaluation under stochastic decoding. Main MCC is computed with temperature $= 0$, while MCC variation is measured under temperature $= 0.7$.

Model	**Main MCC** ($T = 0$)	**MCC Variation** ($T = 0.7$)
PixelReasoner	0.02	± 0.006
Deepeyes	0.00	± 0
Thyme	0.01	± 0.004
PyVision	0.19	± 0.007
Deepeyes v2	0.01	± 0.001
AdaptVision	0.16	± 0.006
Qwen3-vl-8B-Instruct	0.10	± 0.005
Qwen3-vl-32B-Instruct	0.15	± 0.006
Qwen3-vl-235B-Instruct	0.23	± 0.008
GPT-5	0.43	± 0.006
Gemini-3-Pro	0.27	± 0.008

$Y_\tau(x) = \mathbb{1}[d(x) \geq \tau]$ and sweep the threshold τ from 0.2 to 0.8 to recompute each model's MCC. This setup evaluates how well each model's adaptive decisions align with a static, consensus-based difficulty definition instead of its own boundary.

As shown in Tab. 9, our core conclusions are remarkably robust across all thresholds. MCC maintains a consistently strong positive correlation with the final adaptive accuracy gain, with Pearson correlation coefficients ranging from 0.715 (at $\tau = 0.2$) to 0.501 (at $\tau = 0.8$). The correlation remains highly stable, only slightly decreasing at extreme thresholds ($\tau = 0.8$) where the subset of difficult samples becomes very small and represents only the most challenging edge cases.

Crucially, Tab. 10 demonstrates the extreme stability of our model rankings under varying difficulty thresholds. The Spearman rank correlation between the threshold-induced MCC rankings and our main capability-relative MCC ranking is exceptionally high, ranging from 0.927 to 0.987 (with Kendall Tau between 0.801 and 0.929). Furthermore, the set of Top-5 models remains completely identical ($5/5$ overlap) across all swept thresholds, and the Top-3 models exhibit a perfect 100% overlap ($3/3$). While minor rank swaps can occur among models with nearly identical MCC values, the overall model rankings and our primary evaluation findings remain absolutely stable and invariant to the choice of difficulty threshold. This mathematically confirms that MCC is a robust and reliable metric for benchmarking reasoning mode selection in VLMs.

In summary, our robustness analysis across both tool-set configurations and difficulty thresholds demonstrates that MCC is a highly stable and reliable metric. It successfully decouples decision quality from downstream tool-execution complexities and is robust to how task difficulty is defined, ensuring consistent and invariant VLM benchmarking.

B.4 Key Step Coverage

Eq. 4 measures how far the model's reasoning progresses along the annotated key-step sequence. For each annotated step k_i, GPT-5 semantically judges whether its content is covered by the complete generated trajectory $\{s_1, \ldots, s_n\}$, either explicitly or implicitly. The maximization identifies the furthest covered step, and the score is normalized by the total number of annotated key steps $|K|$. If k_3 is the furthest covered step, for example, the trajectory receives $3/|K|$ even when k_1 or k_2 is not separately verbalized, because preceding content may be merged, implicit, or bypassed by a valid derivation. KCoverage therefore permits different wording and concise reasoning without requiring exact textual matching or explicit coverage of every preceding step.

Alternative valid reasoning paths. To assess whether a single reference sequence systematically misses valid alternatives, we manually reviewed 300 randomly sampled trajectories from GPT-5 and 300 from Gemini-3-Pro. For each model, we sampled 150 mathematical trajectories, where alternative derivations are more likely, and 150 trajectories from the other domains. We found one substantially different but valid path for GPT-5 (0.33%) and two for Gemini-3-Pro (0.67%), including one mathematical case and one calculation-based knowledge case. Overall, 3 of the 600 reviewed trajectories (0.50%) followed a substantially different valid path. Because mathematical cases were deliberately over-sampled, this analysis is a targeted stress test rather than an estimate of benchmark-wide prevalence. These cases are rare, but a single reference sequence may still underestimate a valid trajectory that diverges substantially from the annotated path, as acknowledged in the Limitations section.

B.5 Tool Execution Effectiveness

Eq. 5 evaluates whether the model's tool calls are valid and useful for the reasoning process. For each tool invocation t_i, GPT-5 judges its validity conditioned on the corresponding reasoning step s_i. A tool call is considered valid when it matches the

Table 9: Sensitivity of MCC–adaptive-gain correlation to the difficulty threshold.

τ	Difficult Count	Difficult Fraction	Mean MCC	Pearson with Adaptive Gain	Spearman with Adaptive Gain
0.2	645	49.6%	0.159	0.715	0.533
0.3	551	42.4%	0.154	0.710	0.533
0.4	467	35.9%	0.149	0.674	0.497
0.5	383	29.5%	0.149	0.680	0.524
0.6	289	22.2%	0.145	0.641	0.419
0.7	218	16.8%	0.137	0.567	0.369
0.8	146	11.2%	0.132	0.501	0.314

Table 10: Rank stability of threshold-induced MCC rankings compared with the main MCC ranking. The paper Top-3 models are GPT-5, Qwen3-VL-235B-Instruct, and Gemini-3-Pro.

τ	Spearman Rank Corr. vs. Main MCC	Kendall Tau vs. Main MCC	Paper Top-3 Overlap	Paper Top-5 Overlap	**Threshold Top-3 Models**
0.2	0.927	0.801	3/3	5/5	GPT-5, Gemini-3-Pro, Qwen3-VL-235B-Instruct
0.3	0.927	0.801	3/3	5/5	GPT-5, Gemini-3-Pro, Qwen3-VL-235B-Instruct
0.4	0.950	0.865	3/3	5/5	GPT-5, Gemini-3-Pro, Qwen3-VL-235B-Instruct
0.5	0.987	0.929	3/3	5/5	GPT-5, Gemini-3-Pro, Qwen3-VL-235B-Instruct
0.6	0.955	0.865	3/3	5/5	GPT-5, Gemini-3-Pro, Qwen3-VL-235B-Instruct
0.7	0.937	0.828	3/3	5/5	GPT-5, Gemini-3-Pro, Qwen3-VL-235B-Instruct
0.8	0.937	0.828	3/3	5/5	GPT-5, Gemini-3-Pro, Qwen3-VL-235B-Instruct

Table 11: Per-model MCC under representative difficulty thresholds.

Model	$\tau = 0.3$	$\tau = 0.5$	$\tau = 0.7$
PixelReasoner	0.029	0.067	0.112
DeepEyes	0.000	0.000	0.000
Thyme	0.008	0.025	0.026
PyVision	0.196	0.180	0.180
DeepEyes v2	0.016	0.025	0.018
AdaptVision	0.223	0.189	0.163
Qwen3-VL-8B-Instruct	0.100	0.085	0.062
Qwen3-VL-32B-Instruct	0.090	0.114	0.091
Qwen3-VL-235B-Instruct	0.195	0.216	0.233
GPT-5	0.400	0.333	0.364
Gemini-3-Pro	0.277	0.306	0.228

model's stated intent, avoids redundant invocations, uses appropriate tool arguments, and does not suffer from execution errors. The metric averages these binary validity judgments over all N_{tool} tool invocations in the trajectory. Therefore, a higher Tool Execution Effectiveness score indicates that the model not only decides to use tools, but also executes them in a way that is consistent with its reasoning goal.

B.6 Auxiliary-Line Generation Analysis

We further evaluate four models with explicit visual-generation capabilities on the 120 auxiliary-line problems: Visual Sketchpad (Hu et al., 2024), ViLaSR (Wu et al., 2025b), V-Thinker (Qiao et al., 2025b), and CodePlot-CoT (Duan et al., 2025). Table 12 separates the effects of textual construction guidance, tool execution, and visual auxiliary evidence.

Gold auxiliary text provides modest gains, while allowing tool use based on that guidance yields further improvements, especially for Visual Sketchpad and CodePlot-CoT. Gold auxiliary images produce the largest gains, and adding textual descriptions provides a smaller complementary benefit. The gap between gold textual guidance with tool use and a gold auxiliary image indicates that current models remain imperfect at executing visual constructions. Moreover, misleading auxiliary evidence reduces accuracy below the original baseline for all four models, showing that models are sensitive to auxiliary content and that incorrect constructions can actively interfere with downstream reasoning.

C Correlation Analysis of Experimental Results

We further analyze how different diagnostic metrics relate to final adaptive accuracy. This analysis is intended to be descriptive rather than causal: model-level correlations can be affected by model family, scale, base text-reasoning capability, and task composition. Nevertheless, the trends help clarify why mode selection, process quality, and final accuracy should be evaluated as complementary dimensions.

MCC is not redundant with final accuracy. As shown in Fig. 7, after excluding frontier closed-source models and the largest Qwen models, MCC has only a weak association with adaptive accuracy (Pearson $r = 0.15$). This supports our view that MCC is not a proxy for final task accuracy.

Table 12: Accuracy (%) of visual-generation models under different auxiliary-line conditions on 120 problems.

Setting	Input / Constraint	Visual Sketchpad	ViLaSR	V-Thinker	CodePlot-CoT
Original baseline	Original image; the model freely selects whether and which tools to use	29.17	25.00	20.83	32.50
Gold auxiliary text, no tool use	Gold textual description of the auxiliary construction; visual tools are disabled	32.50	28.33	24.17	36.67
Gold auxiliary text + tool use	Gold textual description of the auxiliary construction; the model may invoke visual tools	35.83	29.17	25.00	40.00
Gold auxiliary image	Gold auxiliary image is provided	41.67	36.67	26.67	45.00
Gold auxiliary image + text	Gold auxiliary image and its textual description are provided	42.50	37.50	29.17	46.67
Misleading auxiliary	Random auxiliary text and a random auxiliary image are provided	20.00	18.33	15.00	21.67

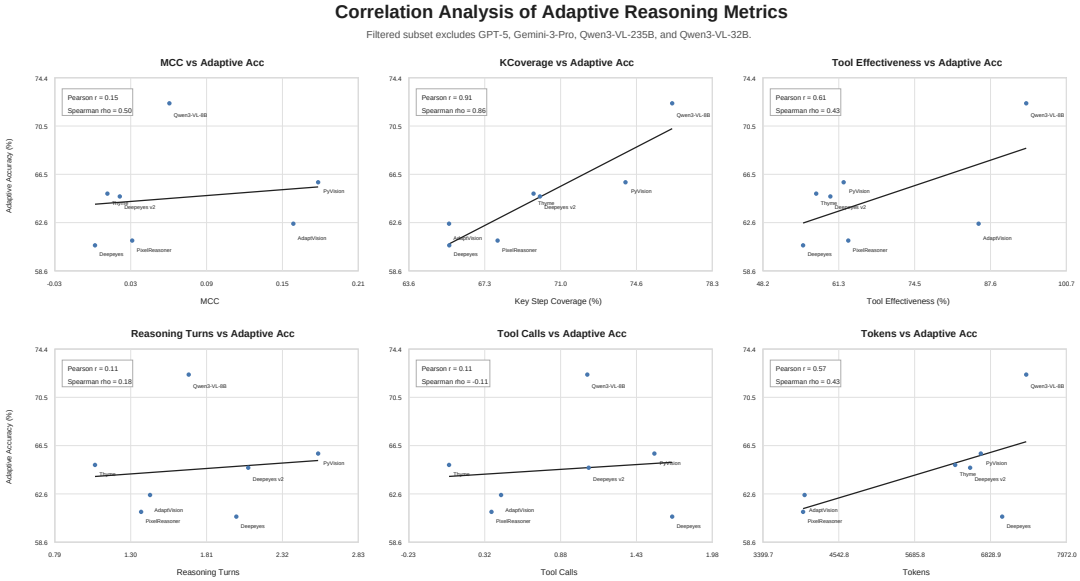


Figure 7: Correlation analysis between diagnostic metrics and adaptive accuracy on a filtered model subset, excluding GPT-5, Gemini-3-Pro, Qwen3-VL-235B, and Qwen3-VL-32B. This subset removes frontier closed-source models and the largest Qwen models to reduce scale-driven effects.

MCC evaluates whether a model selects the appropriate reasoning mode relative to its own capability boundary, whereas final accuracy also depends on visual recognition, language reasoning, tool execution, and robustness to multi-step visual context. This filtered analysis is complemented by the full-model ranking: Gemini-3-Pro achieves the highest final adaptive accuracy, but GPT-5 obtains the highest MCC. Thus, a model can answer more accurately overall while still being less calibrated in deciding when tool-augmented reasoning is necessary.

Mode selection improves with model scale. Although MCC is not redundant with final accuracy, it is not random. Separately from the filtered correlation analysis, the full Qwen3-VL family shows a monotonic within-family trend: MCC increases from 0.06 for Qwen3-VL-8B to 0.14 for Qwen3-VL-32B and 0.22 for Qwen3-VL-235B. We interpret this as suggestive evidence that adaptive mode selection may benefit from stronger model capacity, rather than as a controlled scaling-law claim. Parameter count is not isolated from other factors such as training data, post-training recipes, instruction-following ability, and tool-use alignment. Nevertheless, the trend is consistent with the view that mode selection is a higher-level metacognitive capability: to choose the correct reasoning mode, a model must estimate whether its current visual-textual evidence is sufficient, anticipate whether additional visual operations may provide useful information, and avoid redundant tool invocation when text reasoning is already adequate.

Process quality is more closely associated with final accuracy. Within the same filtered subset, Key Step Coverage and Tool Effectiveness show stronger correlations with adaptive accuracy than MCC. Key Step Coverage is highly correlated with

adaptive accuracy (Pearson $r = 0.91$), and Tool Effectiveness also shows a clear positive association (Pearson $r = 0.61$). This indicates that final success is more closely associated with whether the model traverses the necessary reasoning steps and whether its tool calls provide useful visual evidence.

Process efficiency shows mixed associations with final accuracy. In contrast to process quality, efficiency indicators such as the number of reasoning turns, tool invocation counts, and total token usage show mixed and less interpretable associations with final adaptive accuracy. Reasoning turns and tool-call counts have weak correlations, while token usage shows a moderate positive association in this subset. This should not be interpreted as evidence that using more tokens causes higher accuracy, since token consumption is also affected by model configuration, image resolution, and tool-interface design. A concise trajectory with fewer steps or tool calls is only beneficial if the model still secures the necessary evidence to solve the task; otherwise, premature termination leads to reasoning failure. Conversely, an over-elaborate trajectory with redundant tool calls represents high computational cost without guaranteeing correctness. Therefore, process efficiency evaluates resource consumption rather than cognitive correctness, highlighting the necessity of assessing efficiency as a separate dimension to guide deployment trade-offs.

MCC correlates with adaptive recovery. Figure 8 shows the relationship between MCC and recovery, defined as the fraction of oracle-visual gains recovered by adaptive reasoning:

$$\text{Recovery} = \frac{\text{Acc}_{\text{adaptive}} - \text{Acc}_{\text{text}}}{\text{Acc}_{\text{oracle}} - \text{Acc}_{\text{text}}}.$$

The results suggest two observations:

- **MCC is positively associated with adaptive recovery.** Models with higher MCC tend to achieve stronger recovery, indicating that accurate mode selection is important for translating tool availability into adaptive accuracy gains. For example, GPT-5 achieves the strongest mode-selection quality and a large adaptive improvement over text-only reasoning, showing that accurate tool-invocation decisions can translate into substantial gains.
- **MCC is not the only factor determining recovery or efficiency.** The correlation is moderate (Pearson $r = 0.29$) because recovery is also shaped by downstream reasoning quality, tool execution quality, and the model's underlying text and visual reasoning capabilities. For example, AdaptVision has relatively strong MCC among open-source models, but its weak key-step coverage limits the benefit of adaptive reasoning, resulting in only a small adaptive gain. Similarly, comparing Table 2 with Table 3 shows that higher MCC does not necessarily imply better efficiency. MCC measures whether the model makes the correct mode-selection decision, whereas efficiency further depends on what happens after tool invocation, including the length of the reasoning trajectory, the number of tool calls, the maximum-pixel configuration, and model-specific training recipes and parameter scales.

D More Data Details

D.1 Data Source Distribution

Natural scene VQA. We target high-resolution natural scene by leveraging VisualProbe (Lai et al., 2025) for small-object search and a custom SA-1B (Kirillov et al., 2023) subset for large-scale object reasoning. Queries are explicitly designed to evaluate attributes, spatial, counting, physical state and text-recognition across distinct scales. Moreover, statistics of bounding box sizes are presented in Fig. 9.

Text-Rich VQA. This domain covers diverse charts, tables, and documents. We aggregate standard samples from ChartQA (Masry et al., 2022) and DocVQA (Mathew et al., 2021) with high-resolution challenges from ChartQA-Pro (Masry et al., 2025), MM-RealWorld (Zhang et al., 2025c), and Insight-o3 (Li et al., 2025b) to demand precise visual inspection and deep reasoning.

Math VQA. To assess mathematical reasoning in visual contexts, we consolidate high-quality samples from a spectrum of established benchmarks including MathVista (Lu et al., 2023), MathVerse (Zhang et al., 2024), We-Math (Qiao et al., 2025a), LogicVista (Xiao et al., 2024), Visulogic (Xu et al., 2025), AuxSolidMath, and VT-Bench (Lin et al., 2025a).

GUI VQA. We construct a cross-platform suite covering iOS, Android, Web, macOS, Windows, and Linux. This is achieved by integrating generic datasets like GUI-Knowledge-Bench (Shi et al., 2025) and MMBench-GUI (Liu et al., 2024) with

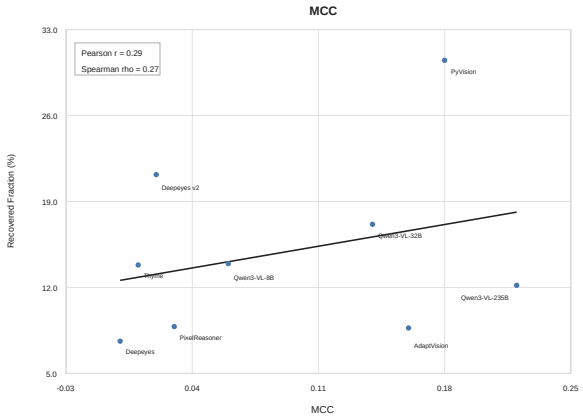


Figure 8: Correlation between MCC and adaptive recovery. Recovery measures the fraction of oracle-visual gains over text-only reasoning recovered by adaptive reasoning.

domain-specific samples from WebWalker (Wu et al., 2025a).

Knowledge VQA. This category is sourced from disciplinary benchmarks across Physics, Chemistry, and Biology. Specifically, we incorporate expert-level samples from **MMMU** (Yue et al., 2024) and **SciVerse** (Guo et al., 2025c) to evaluate the models' ability to integrate specialized domain knowledge with visual reasoning.

D.2 Data Construction Pipeline

The construction workflow of AdaptBench is depicted in Figure 10. Initially, raw data is partitioned based on reasoning complexity, specifically separating tasks that necessitate external tool intervention from those amenable to text-only inference. To further challenge model adaptability, we augment the visual inputs with diverse transformations, thereby mandating fine-grained perception. Finally, we implement a multi-stage verification pipeline involving expert annotation of transformation logic, key reasoning steps, and rigorous human review, ensuring a high-fidelity ground truth for the final benchmark.

D.3 Dataset Difficulty

We further analyze the difficulty stratification of AdaptMMBenchto verify that the benchmark covers samples with different levels of empirical solvability. **This analysis is intended only to characterize the dataset composition and is not used as a difficulty label for adaptive mode-selection evaluation.** Specifically, we identify samples that can be solved by Qwen2.5-VL-7B-Instruct under text-only reasoning as the relatively easy subset, which accounts for 44.23% of the benchmark. The remaining 55.77% of samples form the relatively hard subset. As shown in Table 13, we report the adaptive reasoning accuracy of each evaluated model on the two subsets. All evaluated models achieve substantially higher adaptive reasoning accuracy on the easy subset than on the hard subset, indicating that this stratification captures a meaningful difference in sample difficulty.

Table 13: Accuracy on the relatively easy and hard subsets of AdaptMMBench. The split is used only for dataset-level difficulty analysis and is not used as an evaluation label.

Model	Easy	Hard
Thyme	93.2	42.5
Qwen3-vl-235B	96.9	73.0
Qwen3-vl-32B	96.0	71.2
Qwen3-vl-8B	91.8	56.8
PyVision	82.6	52.6
PixelReasoner	87.5	40.1
GPT-5	95.0	77.8
Deepeyes v2	92.0	43.0
Deepeyes	87.0	39.9
AdaptVision	92.2	38.9

D.4 Annotation Details

This section addresses two potential concerns about the annotation process: subjectivity in visual-enhancement parameters and reliability of human-provided annotations.

Enhancement parameters such as brightness adjustment and rotation are not subjective annotation targets. Instead, they are manually specified perturbation parameters applied to normal images during

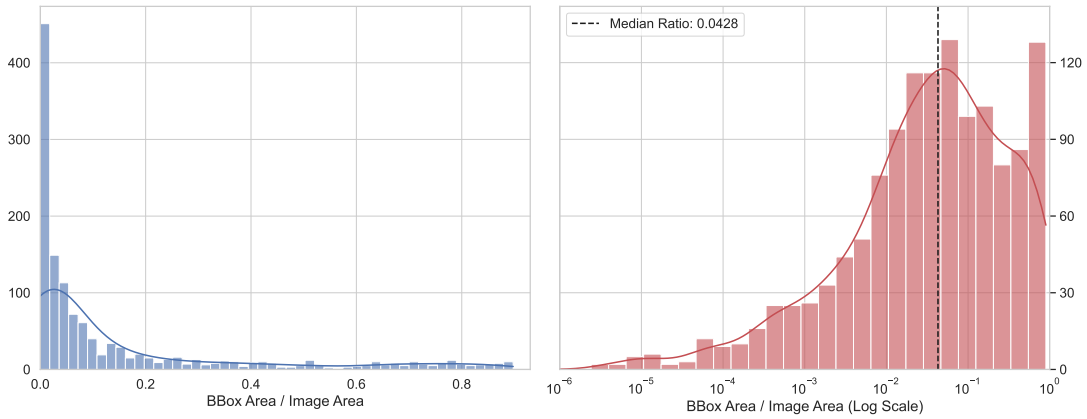


Figure 9: Statistics of bounding box sizes in AdaptMMBench.

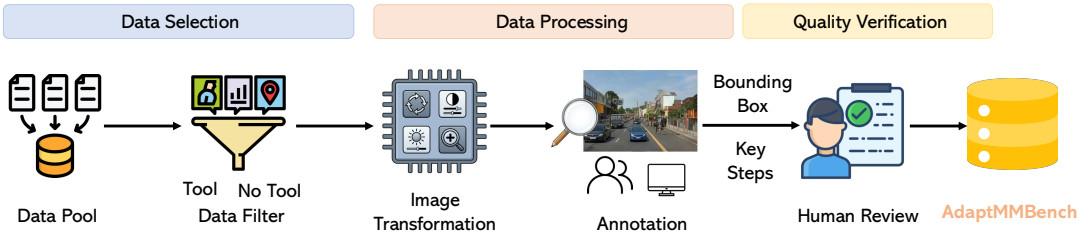


Figure 10: Overview of the data curation and annotation process.

benchmark construction. Therefore, these values should be understood as controlled data-generation settings rather than human-labeled semantic annotations.

For human-provided annotations, including answer-region bounding boxes and key reasoning steps, we conduct multi-annotator verification on the 1,300 main samples. The three annotators achieve 97% agreement on answer-region boxes and 91% agreement on key-step annotations. Cases with inconsistent annotations are further reviewed and re-annotated before inclusion in the final benchmark.

E LLM-Judge Reliability

We validate the reliability of the LLM-based process evaluator from two complementary perspectives: inter-judge consistency across frontier models and human agreement on sampled predictions.

E.1 Robustness to Decoding Randomness

The main results in Table 3 use temperature $= 0$ for deterministic evaluation. To test sensitivity to decoding stochasticity, we additionally run open-source models at temperature $= 0.7$ and report mean and standard deviation in Table 14. The scores fluctuate moderately, as stochastic decoding can change reasoning trajectories and tool-use behavior, but the main process-quality ranking remains stable. Efficiency metrics, especially token usage, show larger variation because sampled reasoning paths may differ in length and tool-call patterns.

E.2 Inter-Judge Consistency

We additionally evaluate the same model outputs with Gemini-3-Pro and Claude Opus 4.8 under the identical protocol. Although absolute scores differ moderately across judges, all three judges produce identical model rankings for both key-step coverage and tool effectiveness. This consistency suggests that our process-level conclusions are not artifacts of GPT-5-specific preferences.

E.3 Human Agreement with GPT-5 Judgments

We conduct human validation with three annotators on 300 randomly sampled predictions, with 100 predictions from each of PyVision, AdaptVision, and Qwen3-VL-235B. Annotators make binary judgments on step coverage and tool effectiveness. As shown in Table 16, GPT-5 achieves high agreement with human judgments, supporting its reliability as an evaluator for these process metrics.

We further analyze disagreement cases. Tool-judgment disagreements mainly fall into three cat-

Table 14: Robustness of process-level evaluation under stochastic decoding. We report mean and standard deviation over repeated runs at temperature $= 0.7$ for key step coverage, tool effectiveness, and efficiency metrics.

Model	Key Step Cov. (%) ↑	Tool Effect. (%) ↑	Efficiency		
			Steps ↓	Tools ↓	Tokens ↓
PixelReasoner	69.26 ± 2.25	65.38 ± 3.28	1.43 ± 0.19	0.41 ± 0.09	4492.61 ± 688.87
Deepeyes	66.94 ± 3.43	56.72 ± 2.23	2.08 ± 0.15	1.72 ± 0.08	7179.48 ± 191.18
Thyme	71.03 ± 3.77	59.91 ± 4.06	1.11 ± 0.09	0.07 ± 0.08	5948.37 ± 671.49
PyVision	75.84 ± 2.35	64.44 ± 3.15	2.48 ± 0.14	1.49 ± 0.10	6532.15 ± 605.53
Deepeyes v2	71.72 ± 2.09	61.96 ± 3.90	2.15 ± 0.10	1.13 ± 0.03	6278.44 ± 242.17
AdaptVision	66.61 ± 3.39	88.18 ± 3.51	1.49 ± 0.15	0.48 ± 0.05	4621.83 ± 434.67
Qwen3-vl-8B	77.86 ± 3.31	93.34 ± 2.02	1.76 ± 0.07	1.03 ± 0.07	7338.92 ± 515.28
Qwen3-vl-32B	80.21 ± 2.86	92.78 ± 2.33	2.31 ± 0.07	1.32 ± 0.03	7068.54 ± 491.93
Qwen3-vl-235B	83.57 ± 2.84	95.12 ± 2.29	1.97 ± 0.05	0.94 ± 0.04	6721.36 ± 533.06

Table 15: Inter-judge consistency across GPT-5, Gemini-3-Pro, and Claude Opus 4.8. Rank triplets follow GPT/Gemini/Claude order.

Model	KSC			KSC Rank	TE			TE Rank
	GPT	Gem.	Claude	G/Gem./C	GPT	Gem.	Claude	G/Gem./C
PixelReasoner	67.91	74.70	72.10	7/7/7	62.99	63.50	62.10	5/5/5
Deepeyes	65.57	74.20	70.80	8/8/8	55.17	49.80	49.60	9/9/9
Thyme	69.65	75.40	73.60	6/6/6	57.44	56.90	55.40	8/8/8
PyVision	74.10	78.20	74.90	4/4/4	62.19	62.80	61.30	6/6/6
Deepeyes v2	69.96	76.90	74.20	5/5/5	59.89	57.20	56.80	7/7/7
AdaptVision	65.56	71.80	67.30	9/9/9	85.55	80.30	77.80	4/4/4
Qwen3-VL-8B	76.36	79.40	77.60	3/3/3	93.79	90.30	88.10	2/2/2
Qwen3-VL-32B	81.47	84.40	80.10	2/2/2	91.42	89.60	85.70	3/3/3
Qwen3-VL-235B	82.27	85.90	83.70	1/1/1	93.93	93.40	91.60	1/1/1

Table 16: Human agreement with GPT-5 process judgments. Tools and Steps denote the number of evaluated tool-use and key-step judgments. Agreement is reported as overall agreement, with tool/step agreement shown in parentheses.

Model	Tools	Steps	Agreement (Tool / Step)
PyVision	172	273	96.0 (97.1 / 94.9)
AdaptVision	45	145	92.0 (95.6 / 90.3)
Qwen3-VL-235B	106	203	94.0 (94.3 / 94.1)

egories: crops contain relevant information but include background noise; the main purpose is achieved but incomplete; and the target is correct but the crop is blurry. The corresponding disagreement rates are PyVision (1.74 / 0.58 / 0.58), AdaptVision (4.40 / 0 / 0), and Qwen3-VL-235B (2.83 / 0.94 / 1.89). Humans tend to count these cases as matched or effective, while GPT-5 applies a stricter criterion.

For key-step judgments, disagreements mainly come from two boundary cases: steps are implicitly expressed rather than explicit, or the wording is uncertain. The corresponding rates are PyVision (4.76 / 0.37), AdaptVision (8.28 / 1.38), and Qwen3-VL-235B (5.42 / 0.49). Since our evaluation is based on maximum matched-step correctness, these minor boundary discrepancies have limited impact on the final judgment.

F Transform Results

Tab. 17 serves as a supplementary detailed analysis to the main experimental results presented in Tab. 4. While Tab. 4 reports the model performance on the original dataset (denoted as "w/o enh.") and the aggregated dataset ("All"), it does not explicitly isolate the performance on the transformed data. To provide a comprehensive view of model robustness against data variations, Tab. 17 exclusively presents the accuracy results for the **Transformed** subset across all five domains. The highlighting scheme follows the same convention as the main table to facilitate direct comparison of the autonomous decision-making capabilities in the Adaptive mode.

Table 17: Main experimental results on the **Transform** subset of AdaptBench. We report accuracy (%) across five specific domains and overall aggregates. Performance for **Text** and **Oracle** modes is displayed in deep gray to prioritize adaptive results. * indicates that the model supports enhancement operations.

Model	Mode	Natural scene	OCR	GUI	Knowledge	Math	Overall Accuracy
Open-Source Models							
PixelReasoner	Text	46.67	50.00	30.00	56.67	21.67	41.15
	Adaptive	61.67	62.50	42.50	66.67	41.67	55.38
	Oracle	66.67	75.00	70.00	83.33	55.00	69.62
Deepeyes	Text	56.67	40.00	35.00	55.00	41.67	46.92
	Adaptive	56.67	37.50	32.50	60.00	48.33	48.85
	Oracle	73.33	77.50	65.00	81.67	63.33	72.31
Thyme*	Text	48.33	35.00	32.50	55.00	38.33	43.08
	Adaptive	45.00	20.00	42.50	53.33	46.67	43.08
	Oracle	68.33	80.00	62.50	78.33	63.33	70.38
PyVision*	Text	50.00	30.00	15.00	53.33	30.00	37.69
	Adaptive	55.00	27.50	15.00	63.33	33.33	41.54
	Oracle	70.00	72.50	35.00	85.00	45.00	62.69
Deepeyes v2*	Text	55.00	45.00	27.50	58.33	40.00	46.54
	Adaptive	50.00	30.00	45.00	50.00	36.67	43.08
	Oracle	73.33	67.50	67.50	88.33	70.00	74.23
AdaptVision	Text	43.33	52.50	42.50	56.67	31.67	45.00
	Adaptive	60.00	30.00	27.50	63.33	35.00	45.38
	Oracle	76.67	77.50	70.00	95.00	56.67	75.38
Qwen3-vl -8B-Instruct	Text	40.00	40.00	30.00	55.00	25.00	38.46
	Adaptive	35.00	42.50	32.50	55.00	31.67	39.62
	Oracle	60.00	82.50	80.00	86.67	55.00	71.54
Qwen3-vl -32B-Instruct	Text	50.00	50.00	35.00	58.33	31.67	45.38
	Adaptive	40.00	57.50	40.00	71.67	33.33	48.46
	Oracle	51.67	90.00	77.50	88.33	60.00	71.92
Qwen3-vl -235B-Instruct	Text	45.00	60.00	32.50	65.00	3.33	45.00
	Adaptive	51.67	57.50	40.00	71.67	25.00	49.23
	Oracle	68.33	97.50	82.50	93.33	51.67	76.92
Closed-Source Models							
GPT-5*	Text	55.00	32.50	17.50	60.00	41.67	43.85
	Adaptive	71.67	60.00	47.50	75.00	40.00	59.62
	Oracle	73.33	85.00	52.50	88.33	48.33	69.62
Gemini-3-Pro*	Text	70.00	75.00	50.00	85.00	30.00	61.92
	Adaptive	81.67	97.50	62.50	90.00	48.33	75.38
	Oracle	86.67	100.00	80.00	95.00	51.67	81.54

G Category Results

In this section, we provide a fine-grained analysis of model performance across all specific categories defined in our benchmark. To ensure legibility and accommodate the wide range of sub-domains, the detailed accuracy results are presented in two separate tables:

- **Tab. 18** reports the performance metrics for the **GUI** and **Realworld** domains.
- **Tab. 19** covers the **Knowledge**, **Math**, and **OCR** domains.

In both tables, the first row (labeled as *N*) denotes the number of test samples available for each corresponding category. All accuracy values are reported in decimal format.

Table 18: Detailed Accuracy (Part 1/2): GUI and Realworld domains. * indicates that the model supports enhancement operations.

		GUI						Realworld				
Model	**Mode**	And.	Lin.	Mac.	Web	Win.	iOS	Attr.	Count	Integ.	Phys.	Spat.
N	(Count)	83	22	39	75	51	30	118	16	121	10	35
PixelReasoner	Text	0.51	0.45	0.31	0.52	0.53	0.47	0.45	0.12	0.35	0.50	0.34
	Adaptive	0.51	0.50	0.41	0.61	0.57	0.50	0.55	0.50	0.49	0.70	0.43
	Oracle	0.59	0.68	0.36	0.63	0.67	0.57	0.68	0.75	0.69	0.50	0.63
Deepeyes	Text	0.59	0.45	0.36	0.63	0.45	0.57	0.55	0.38	0.44	0.30	0.51
	Adaptive	0.48	0.36	0.44	0.68	0.49	0.53	0.58	0.44	0.50	0.30	0.57
	Oracle	0.67	0.50	0.46	0.75	0.63	0.63	0.65	0.69	0.69	0.40	0.69
Thyme*	Text	0.53	0.45	0.36	0.60	0.41	0.50	0.57	0.62	0.51	0.40	0.34
	Adaptive	0.54	0.55	0.41	0.64	0.45	0.57	0.64	0.75	0.54	0.40	0.51
	Oracle	0.61	0.68	0.56	0.80	0.55	0.60	0.73	0.75	0.79	0.60	0.60
PyVision*	Text	0.81	0.64	0.82	0.47	0.76	0.77	0.51	0.44	0.24	0.40	0.40
	Adaptive	0.83	0.59	0.82	0.51	0.80	0.90	0.58	0.44	0.36	0.50	0.51
	Oracle	0.92	0.77	0.92	0.79	0.88	0.83	0.87	0.88	0.80	0.70	0.94
Deepeyes v2*	Text	0.46	0.45	0.46	0.68	0.59	0.57	0.60	0.50	0.52	0.70	0.46
	Adaptive	0.49	0.45	0.44	0.71	0.53	0.63	0.58	0.69	0.55	0.50	0.49
	Oracle	0.64	0.68	0.51	0.77	0.76	0.77	0.72	0.81	0.79	0.80	0.63
AdaptVision	Text	0.48	0.32	0.28	0.59	0.49	0.53	0.48	0.25	0.40	0.60	0.40
	Adaptive	0.51	0.36	0.41	0.68	0.55	0.60	0.54	0.25	0.40	0.60	0.46
	Oracle	0.65	0.50	0.51	0.76	0.71	0.63	0.69	0.69	0.75	0.80	0.60
Qwen3-vl -8B-Instruct	Text	0.52	0.45	0.49	0.64	0.51	0.53	0.55	0.44	0.47	0.70	0.40
	Adaptive	0.61	0.64	0.46	0.63	0.57	0.67	0.58	0.50	0.44	0.70	0.57
	Oracle	0.70	0.59	0.51	0.77	0.67	0.60	0.70	0.88	0.84	0.80	0.77
Qwen3-vl -32B-Instruct	Text	0.81	0.77	0.56	0.75	0.57	0.77	0.56	0.56	0.50	0.50	0.34
	Adaptive	0.76	0.77	0.51	0.68	0.67	0.83	0.64	0.50	0.54	0.80	0.46
	Oracle	0.80	0.68	0.62	0.83	0.73	0.77	0.78	0.81	0.86	0.70	0.83
Qwen3-vl -235B-Instruct	Text	0.80	0.77	0.79	0.65	0.88	0.73	0.61	0.50	0.50	0.40	0.34
	Adaptive	0.83	0.77	0.87	0.76	0.78	0.77	0.60	0.50	0.57	0.50	0.46
	Oracle	0.92	0.91	0.95	0.88	0.94	0.90	0.81	0.75	0.83	0.60	0.83
GPT-5*	Text	0.86	0.82	0.87	0.47	0.75	0.90	0.62	0.44	0.31	0.40	0.46
	Adaptive	0.89	0.86	0.90	0.75	0.88	0.90	0.77	0.62	0.55	0.60	0.57
	Oracle	0.96	0.95	0.82	0.84	0.80	0.90	0.92	0.88	0.82	0.80	0.97
Gemini-3-Pro*	Text	0.90	0.86	0.92	0.79	0.86	0.90	0.58	0.38	0.50	0.50	0.54
	Adaptive	0.93	0.86	0.95	0.84	0.90	0.93	0.82	0.69	0.69	0.90	0.60
	Oracle	0.94	0.91	0.97	0.91	0.90	0.93	0.83	0.75	0.78	0.70	0.86

Table 19: Detailed Accuracy (Part 2/2): Knowledge, Math, and OCR domains. * indicates that the model supports enhancement operations.

		Knowledge				**Math**				**OCR**			
Model	**Mode**	Bio.	Chem.	Geo.	Phys.	Alg.	Geo.	Log.	Stat.	Chart	Diag.	Doc.	Tab.
N	(Count)	57	58	10	75	64	75	13	48	171	40	55	34
PixelReasoner	Text	0.68	0.47	0.70	0.51	0.44	0.44	0.38	0.42	0.57	0.70	0.53	0.62
	Adaptive	0.70	0.45	0.50	0.63	0.52	0.49	0.38	0.52	0.65	0.62	0.55	0.62
	Oracle	0.82	0.62	0.60	0.61	0.69	0.69	0.46	0.54	0.72	0.88	0.73	0.79
Deepeyes	Text	0.61	0.38	0.70	0.44	0.45	0.48	0.31	0.29	0.49	0.52	0.51	0.62
	Adaptive	0.65	0.40	0.70	0.44	0.47	0.59	0.31	0.48	0.58	0.62	0.55	0.53
	Oracle	0.79	0.66	1.00	0.60	0.66	0.64	0.54	0.58	0.63	0.78	0.64	0.76
Thyme*	Text	0.65	0.40	0.70	0.47	0.53	0.52	0.31	0.40	0.56	0.55	0.55	0.76
	Adaptive	0.67	0.41	0.60	0.45	0.52	0.53	0.54	0.42	0.61	0.57	0.56	0.65
	Oracle	0.75	0.66	0.90	0.60	0.64	0.73	0.38	0.52	0.69	0.75	0.62	0.74
PyVision*	Text	0.51	0.16	0.60	0.33	0.28	0.33	0.38	0.29	0.59	0.62	0.58	0.68
	Adaptive	0.72	0.36	0.60	0.49	0.25	0.39	0.31	0.27	0.72	0.75	0.64	0.71
	Oracle	0.56	0.74	0.70	0.57	0.48	0.52	0.46	0.65	0.74	0.85	0.87	0.97
Deepeyes v2*	Text	0.60	0.43	0.30	0.44	0.48	0.40	0.15	0.31	0.58	0.68	0.51	0.62
	Adaptive	0.65	0.48	0.50	0.49	0.50	0.57	0.31	0.40	0.61	0.55	0.51	0.53
	Oracle	0.84	0.53	0.50	0.63	0.66	0.69	0.31	0.62	0.74	0.80	0.73	0.71
AdaptVision	Text	0.63	0.47	0.70	0.51	0.41	0.59	0.15	0.35	0.59	0.70	0.58	0.71
	Adaptive	0.61	0.55	0.80	0.45	0.45	0.52	0.46	0.33	0.63	0.65	0.64	0.68
	Oracle	0.84	0.67	0.90	0.67	0.70	0.72	0.46	0.69	0.70	0.98	0.78	0.76
Qwen3-vl -8B-Instruct	Text	0.75	0.62	0.40	0.65	0.55	0.47	0.31	0.56	0.65	0.75	0.36	0.76
	Adaptive	0.75	0.71	0.70	0.69	0.69	0.56	0.38	0.69	0.68	0.70	0.53	0.71
	Oracle	0.91	0.79	0.60	0.77	0.86	0.77	0.38	0.88	0.78	0.98	0.69	0.91
Qwen3-vl -32B-Instruct	Text	0.84	0.78	1.00	0.68	0.67	0.68	0.31	0.79	0.81	0.82	0.62	0.82
	Adaptive	0.82	0.76	0.90	0.77	0.67	0.77	0.38	0.85	0.83	0.90	0.75	0.85
	Oracle	0.95	0.95	0.90	0.96	0.95	0.84	0.38	0.96	0.91	1.00	0.91	0.91
Qwen3-vl -235B-Instruct	Text	0.81	0.84	0.60	0.75	0.73	0.75	0.23	0.83	0.79	0.78	0.67	0.91
	Adaptive	0.91	0.79	1.00	0.87	0.66	0.83	0.77	0.81	0.80	0.92	0.73	0.79
	Oracle	0.96	0.98	0.90	0.96	0.94	0.96	0.69	0.98	0.89	1.00	0.89	0.94
GPT-5*	Text	0.63	0.45	0.60	0.55	0.38	0.43	0.15	0.42	0.75	0.72	0.67	0.76
	Adaptive	0.88	0.83	0.80	0.88	0.67	0.64	0.23	1.00	0.87	0.90	0.78	0.94
	Oracle	0.93	0.98	0.90	0.93	0.80	0.84	0.46	0.96	0.86	0.98	0.93	0.97
Gemini-3-Pro*	Text	0.86	0.86	0.90	0.77	0.72	0.76	0.46	0.88	0.86	0.95	0.82	0.94
	Adaptive	0.93	0.95	1.00	0.92	0.86	0.91	0.31	0.98	0.86	0.98	0.89	1.00
	Oracle	0.95	0.91	0.90	0.96	0.94	0.91	0.62	0.96	0.89	1.00	0.98	0.97

H Reasoning Mode Prompt

Here we provide the detailed prompts used in our experiments.

I Error Analysis

In this section, we provide detailed visualizations of the failure modes discussed in the *Error Analysis* (Sec. 5.5 of the main paper. By examining the intermediate reasoning steps, we offer concrete examples across different categories of tool-related errors.

Visual Reasoning Failures. As noted in the main text, 42.3% of errors stem from the model's inability to correctly manipulate or locate visual information. We present two representative scenarios:

- **Wrong Image Transformations:** Figure 11 illustrates a case where the model repeatedly fails to correct the image orientation. This visual reasoning failure propagates to the OCR stage, causing the model to misread "831K" as "83K" and producing an incorrect prediction.
- **Incorrect Region Selection:** Figure 12 demonstrates a spatial grounding failure in a dense document. The model zooms into an incorrect region (Question 235 instead of Question 238), leading to reasoning that is logically valid but based on irrelevant visual evidence.

Context Noise in Multi-step Reasoning. Figure 13 depicts the specific error type (accounting for 7.3% of cases) where visual perception is initially correct but overridden by context noise. In this example, the model successfully enhances the image and identifies the correct number of objects ("two") in the intermediate step. However, distracted by the accumulated visual and textual context from the multi-step process, it becomes overly cautious and hallucinates a negation, resulting in a failure.

Correction via Forced Tool Invocation. Figure 14 illustrates a specific scenario (representative of the 7.0% of corrected errors) where forcing tool invocation rectifies an initial estimation failure. In this example, the model originally relies on imprecise visual intuition, incorrectly identifying "Cerulean Blue" as the answer. However, when forced to invoke tools, it bypasses the typical spatial zooming approach and adopts a creative programmatic strategy: using Python to perform a **pixel-level RGB count**. By rigorously verifying consistency across multiple tolerance thresholds, the model successfully overrides its initial hallucination and derives the correct answer based on quantitative data.

Performance Degradation due to Incorrect Mode Selection. Figure 15 exemplifies the 8.3% of cases caused by incorrect mode selection, where the model unnecessarily invokes tools for tasks solvable by direct visual inspection. In this example, accurate icon counting is achievable via standard OCR or visual recognition (as seen in the Text-CoT mode). However, in the adaptive mode, the model complicates the task by adopting an **unreliable engineering approach**: using OpenCV edge detection to count squares. This strategy proves fragile, as the model struggles with parameter tuning—first detecting excessive noise and then over-filtering actual targets—ultimately leading to a hallucinated final count due to the confused tool outputs.

Text Reasoning Setting Prompts

[Multiple Choice]
Question: {question}
Options: {options}
Please think step-by-step and give the final answer following the format: <think> reasoning process </think> <answer> the option's letter </answer>

[Short Answer]
Question: {question}
Please think step-by-step and give the final answer following the format: <think> reasoning process </think> <answer> a single word or phrase </answer>

Adaptive Reasoning Setting Prompts

[Multiple Choice]
Question: {question}
Options: {options}
[Original adaptive prompt] Please think step-by-step and give the final answer following the format: <think> reasoning process </think> <answer> the option's letter </answer>

[Short Answer]
Question: {question}
[Original adaptive prompt] Please think step-by-step and give the final answer following the format: <think> reasoning process </think> <answer> a single word or phrase </answer>

Oracle-Visual Setting Prompts

[Zoom-in Setting]
The first image is the global view, and the second image is the key region (zoomed-in) to help answer the question.
Question: {question}
Please think step-by-step and give the final answer following the format: <think> reasoning process </think> <answer> a single word or phrase </answer>

[Transformed Setting]
The first image is the original input which might be distorted (rotated or dark), and the second image has been corrected and enhanced to show the true content.
Question: {question}
Please think step-by-step and give the final answer following the format: <think> reasoning process </think> <answer> a single word or phrase </answer>

[Auxiliary Line Setting]
The first image is the original image, and the second image is the same image with auxiliary lines to help solve the problem.
Question: {question}
Please think step-by-step and give the final answer following the format: <think> reasoning process </think> <answer> a single word or phrase </answer>

LLM Judge Prompt

I will give you a question related to the image, the ground truth answer, and the model predicted answer. Your task is to determine whether the model's predicted answer and the ground truth answer are consistent, and output the Judgement.
Note that [Model Predicted Answer] is consistent with [Ground Truth Answer] whenever they are essentially the same. If the meaning is expressed in the same way, it is considered consistent, for example, 'pink' and 'it is pink'.
If they are consistent, the Judgement is 1; if they are different, the Judgement is 0.
Output Format: Just output the Judgement and don't output anything else.
[Question]: {question}
[Ground Truth Answer]: {ground_truth}
[Model Predicted Answer]: {prediction}
[Judgement]:

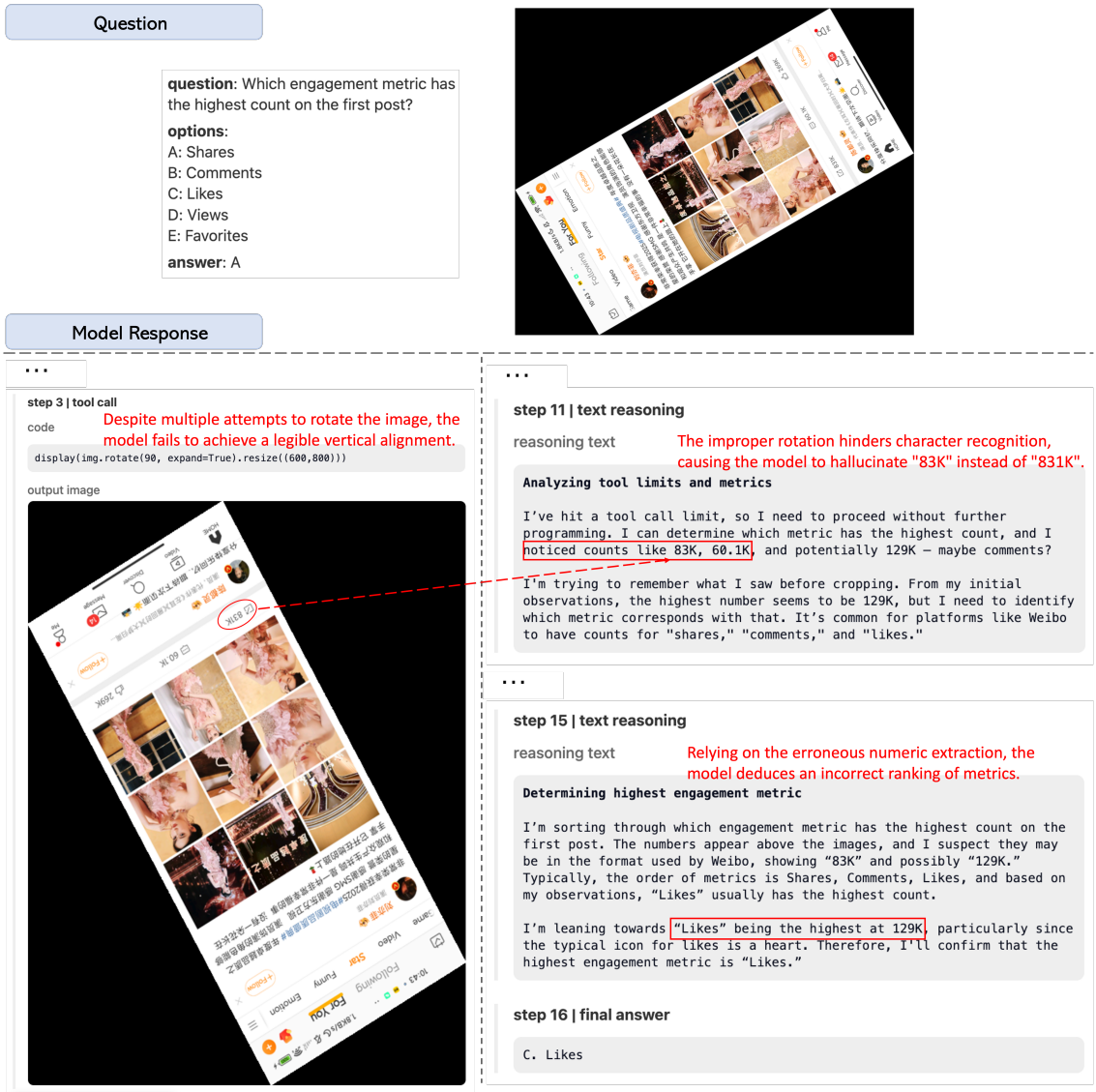


Figure 11: A failure case caused by incorrect image transformation (rotation) leading to OCR errors.

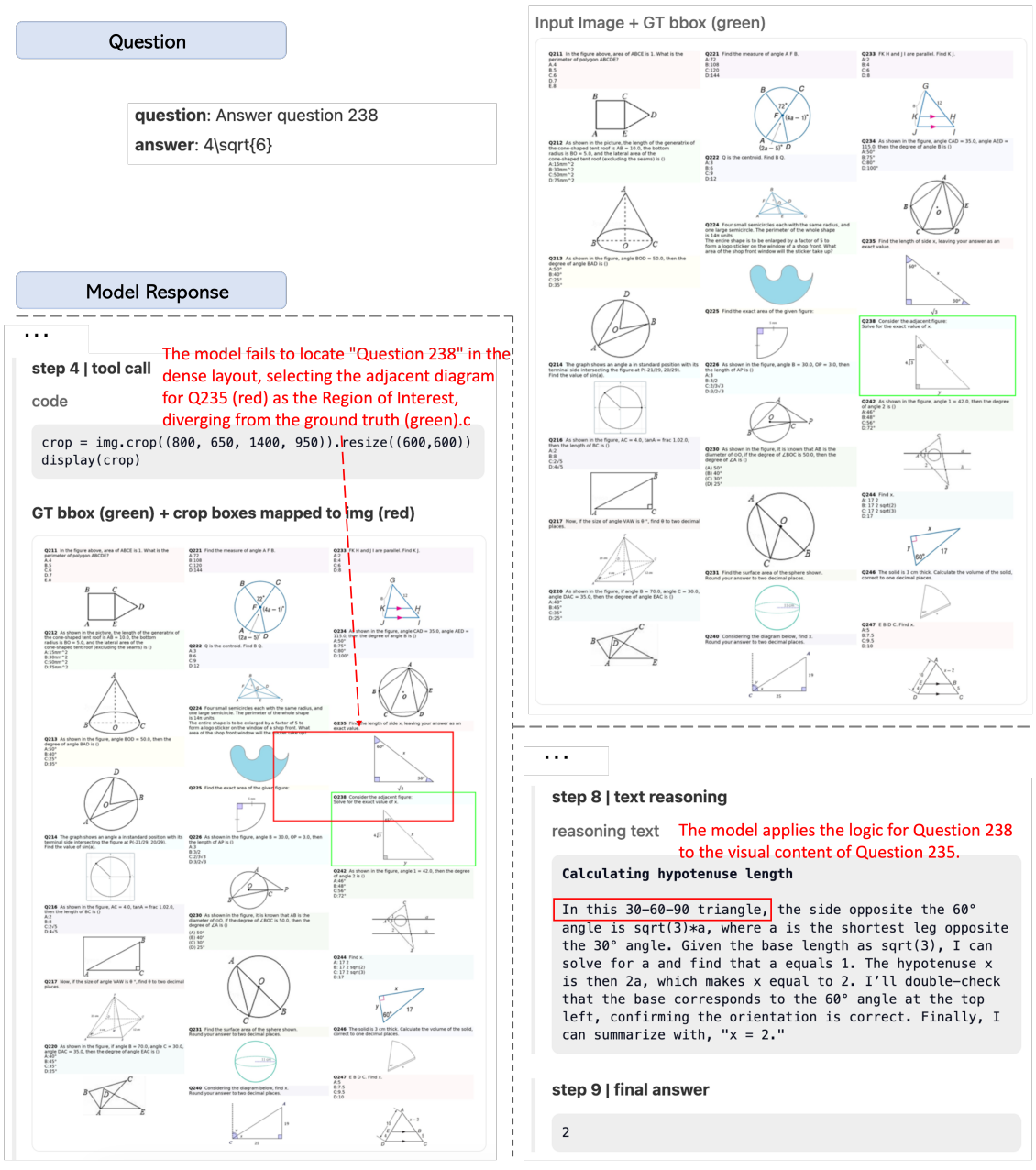


Figure 12: A failure case caused by zooming into an incorrect region (spatial misalignment).

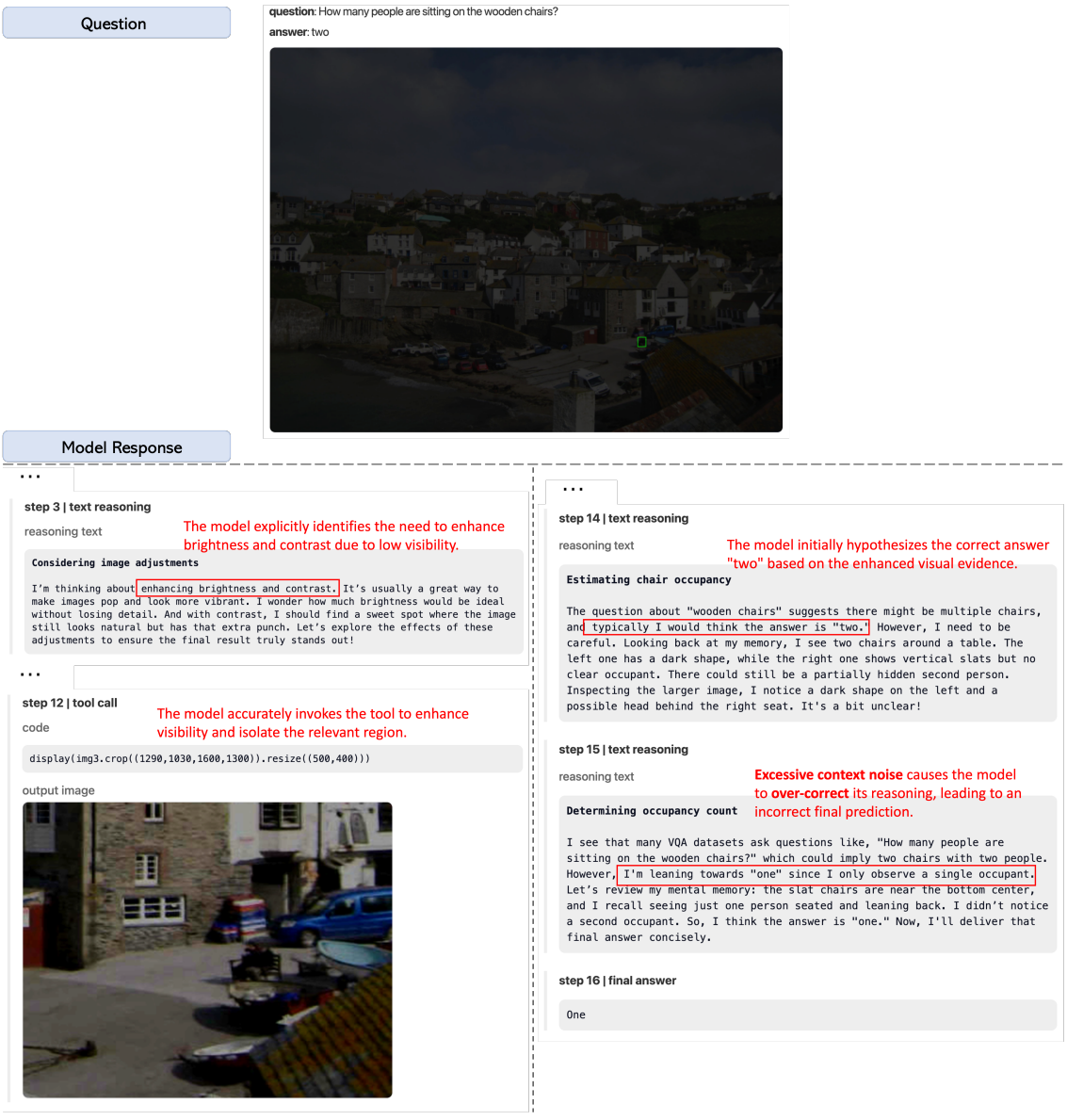
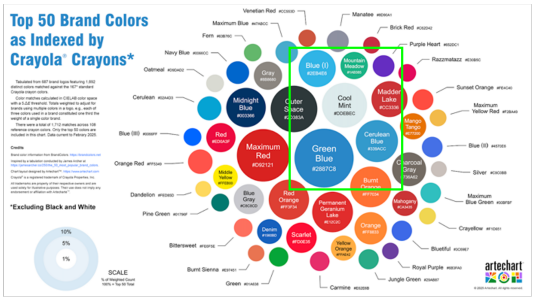


Figure 13: A failure case where correct intermediate visual grounding is overridden by context noise.

Question



question: Among the given brand color usages, which color—Cerulean Blue, Green Blue, or Blue (I)—has the lowest utilization?

answer: Blue (I)

Model Response

...

step 7 | tool call

code

The model is leveraging NumPy to calculate the exact pixel count for each target RGB color to quantify area size.

```
import numpy as np

def count_near_color(arr, target_rgb, tol=25):
    diff = np.linalg.norm(arr - np.array(target_rgb), axis=2)
    mask = diff <= tol
    return mask.sum(), mask

targets = {
    "Cerulean Blue": (51,154,204),
    "Blue (I)": (46,180,230),
    "Green Blue": (40,135,200),
}

results = {}
masks = {}
for name, rgb in targets.items():
    count, mask = count_near_color(arr, rgb, tol=22)
    results[name] = count
    masks[name] = mask

results
```

output text

```
{'Cerulean Blue': 55894, 'Blue (I)': 12450, 'Green Blue': 54521}
```

...

step 9 | tool call

The model iteratively adjusts tolerance thresholds to ensure the ranking remains consistent across different sensitivity levels.

code

```
def count_near_color_tol(arr, target_rgb, tol):
    diff = np.linalg.norm(arr - np.array(target_rgb), axis=2)
    return (diff <= tol).sum()

for tol in [15,20,25,30,35]:
    print(tol, {name: count_near_color_tol(arr, rgb, tol) for name, rgb in
targets.items()})
```

output text

```
15 {'Cerulean Blue': 16598, 'Blue (I)': 12193, 'Green Blue': 51197}
20 {'Cerulean Blue': 18353, 'Blue (I)': 12382, 'Green Blue': 53167}
25 {'Cerulean Blue': 65275, 'Blue (I)': 12555, 'Green Blue': 69457}
30 {'Cerulean Blue': 72982, 'Blue (I)': 12650, 'Green Blue': 70489}
35 {'Cerulean Blue': 75116, 'Blue (I)': 12960, 'Green Blue': 71434}
```

step 10 | text reasoning

reasoning text

The model explicitly bypasses zooming, correctly identifying the answer via global color analysis instead.

```
Finalizing image analysis

I've concluded that Blue (I) has the lowest utilization by far compared to the other
options. While we didn't focus on zooming, we still did color analysis, which should
be good enough. However, to ensure compliance, we could consider cropping or
computing the bounding area. For the final answer, I'll keep it concise: Blue (I) has
the lowest utilization, and the code analysis shows that the Blue (I) circle is the
smallest.
```

step 11 | final answer

```
Blue (I).
```

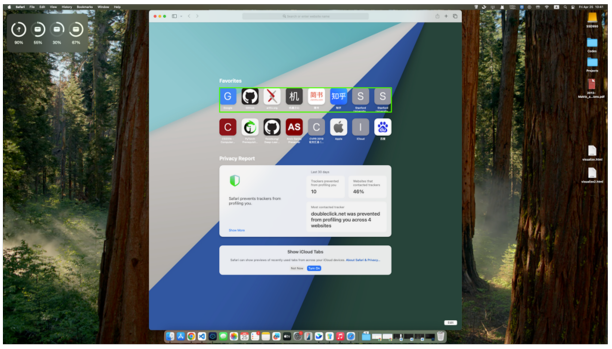
Figure 14: Example of correcting visual estimation errors via code-based pixel analysis.

Question

question: In the Favorites section, how many icons are shown in the first row of the start page?

options:
A: 6
B: 7
C: 8
D: 9
E: 10

answer: C



Model Response

...

step 7 | text reasoning

reasoning text

Instead of simple visual recognition, the model opts for complex OpenCV-based square detection.

```
Working on square detection

I'm considering using OpenCV to detect squares. First, I'll convert the cropped image to grayscale, then apply Canny edge detection. After that, I'll find contours, focusing on the count of square-like shapes while setting a size threshold. This process should help in isolating the squares effectively. Let's implement this step by step and see how well it performs!
```

step 8 | tool call

code

```
import numpy as np, cv2
crop_cv = cv2.cvtColor(np.array(crop), cv2.COLOR_RGB2BGR)
gray = cv2.cvtColor(crop_cv, cv2.COLOR_BGR2GRAY)
edges = cv2.Canny(gray, 80, 200)
# find contours
contours, hierarchy = cv2.findContours(edges, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
len(contours)
```

outputs

```
202
```

...

step 10 | tool call

code

```
rects = []
for cnt in contours:
    x,y,w1,h1 = cv2.boundingRect(cnt)
    area = w1*h1
    if area>1500 and area<50000 and abs(w1-h1)/max(w1,h1)<0.2:
        rects.append((x,y,w1,h1))
len(rects), rects[:10]
```

outputs

Initial detection is overly sensitive, yielding 17 candidate squares (noisy false positives).

```
(17,
 [(566, 292, 42, 48),
  ......])
```

...

step 12 | tool call

code

```
icon_rects = [r for r in rects if 52 <= r[2] <= 64 and 52 <= r[3] <= 64 and r[1] > 170]
len(icon_rects), sorted(icon_rects, key=lambda r:r[0])
```

outputs

Subsequent adjustment is too strict, reducing the count to 6 and missing target icons.

```
(6,
 [(332, 227, 59, 58),
  (408, 227, 58, 58),
  (482, 227, 59, 58),
  (557, 227, 59, 58),
  (708, 227, 58, 58),
  (782, 227, 58, 58)])
```

step 13 | text reasoning

reasoning text

```
Adjusting crop dimensions

I've found 6 squares in my current crop, but it seems like there might be more than 6 in the first row. However, my crop might be covering only part of that row, especially with the right side being truncated. So I think I should extend the crop further to the right to capture more of that first row. Let's explore this adjustment and see how it looks!
```

...

step 16 | text reasoning

reasoning text

Confused by conflicting tool outputs, the model hallucinates additional icons and fails to recognize the visible count.

```
Analyzing icon labels

I'm checking the small labels below the icons. The leftmost 'G' is for Google, followed by 'GitHub', 'arXiv.org', '机器之心', '简书', '知乎', and '京东'. There are also two 'S' icons which could be for Stack Overflow or Stanford. The first row seems to have 10 icons in total, but my partial detection originally found only 6. It looks like 4 more icons were cropped out, confirming that the first row indeed has 10 icons. I'll finalize on Option E.
```

step 17 | final answer

```
E. 10
```

Figure 15: Example of performance degradation caused by unnecessary tool usage in a simple visual task.

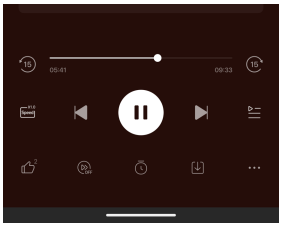
J Process Evaluation Example

To better understand our evaluation protocol, we present detailed cases of process reasoning quality assessment in Figure 16 and Figure 17.

K Process Evaluation Prompt

To ensure a reproducible and standardized assessment, we leverage LLM-based judges with specialized prompts for process auditing. Specifically, we employ the tool invocation effectiveness prompt (detailed in Figure 18) is used to audit the functional correctness and intent alignment of each tool call within the adaptive process. Subsequently, the key step coverage Prompt (detailed in Figure 19) to verify the logical completeness of the model's reasoning trajectory against annotated ground truth.

Question: Based on the playback interface, what is the current status of the audio content?
A: The audio is currently playing
B: The audio is paused
C: The audio is stopped
D: "The audio is buffering
Answer: A



Qwen-3-VL-235B: Thinking: The playback interface shows a large pause button (two vertical bars) in the center, which is the standard icon for indicating that audio or video content is currently paused. If it were playing, it would typically show a triangle pointing to the right. The progress bar also indicates a position at 05:41 out of 09:33, suggesting the content has been started and is not at the beginning or stopped. Therefore, the correct status is paused. Answer: B. The audio is paused"

Judgements:

Key step1: Identify the playback interface on the screen.
Judgment: Matched
Reason: The solution explicitly identifies the playback interface.

Key step2: Locate the central play/pause control button.
Judgment: Matched
Reason: It locates and describes the central play/pause control button.

Key step3: Observe the button shows a pause icon.
Judgment: Matched
Reason: It observes and notes that the button shows a pause icon.

Key step4: Apply UI convention: a pause icon indicates the media is currently playing.
Judgment: Unmatched
Reason: The solution incorrectly claims a pause icon means paused, contradicting the UI convention that it indicates currently playing.

Key step5: Map this function to option A.
Judgment: Unmatched
Reason: The solution maps the status to option B, while the ground truth maps it to option A.

Rate: 3/5

Figure 16: Illustration of key step coverage evaluation.

Question: What's written on the Aunt left chest?
Correct Answer: M·5

Qwen-3-VL-235B:



Step1: Thinking: I need to get a closer look at the Aunt's left chest to read the text on her shirt. I will zoom in on that specific area for a clearer view.
Tool1: { "name": "image_zoom_in_tool", "parameter": "{"label": "Aunt wearing a striped shirt and a straw hat", "bbox_2d": [730, 331, 1000, 854], "img_idx": 0}" }



Step2: Thinking: Now I see the Aunt's shirt more clearly. There is a small, rectangular, brownish patch on her left chest. It appears to have some white text or a logo on it. The text looks like "M'5" or something similar, but it's still a bit blurry. I need to zoom in even closer on just that patch to read it accurately.
Tool2: { "name": "image_zoom_in_tool", "parameter": "{"label": "close-up of the patch on the Aunt's left chest", "bbox_2d": [437, 350, 550, 386], "img_idx": 0}"}



Step3: Thinking: I apologize, but it seems there was an error in the zooming process. The image I received is a close-up of a grey, textured fabric with some small specks, which does not match the description of the Aunt's striped shirt or the patch on her chest. This suggests the tool may have focused on the wrong area. I will now try to zoom in again on the correct region, specifically targeting the patch on the Aunt's left chest to get a clear view of the text.
Tool3: { "name": "image_zoom_in_tool", "parameter": "{"label": "close-up of the patch on the Aunt's left chest", "bbox_2d": [437, 350, 550, 386], "img_idx": 0}"}

Step4: Thinking: I see that the zoom tool is consistently returning an image of a grey, textured fabric, which is not the Aunt's shirt. This indicates a persistent error in the tool's targeting. Since I cannot rely on the tool to get a clear view, I will have to make my best assessment from the original, wider image. In the original photo, the patch on her left chest is visible. It is a small, rectangular, brownish patch with white lettering. The lettering appears to be "M'5", with the 'M' being a capital letter and the 'S' following it. The 'S' might have a small apostrophe or design element before it, but "M'5" is the most legible interpretation.
Answer: M5

Judgements:

Correct Tools: [Tool1] Brief Analysis: Tool1 correctly zooms in on the woman on the right, matching the intent to focus on the aunt's area. Tools 2 and 3 aimed to zoom the patch on her left chest but instead targeted the man's shirt area, returning a grey fabric region; therefore, they did not satisfy the intended focus.

Rate: 1/3

Figure 17: Illustration of tool effectiveness evaluation.

Tool Invocation Effectiveness Prompt

The above content contains the model's multi-step reasoning process. You are a top-tier visual reasoning audit expert. You possess strong logical analysis skills and can accurately evaluate whether a model's tool usage aligns with its stated reasoning intent.
All Available Tools:
{tools}
Each step in the reasoning process includes:

- [Step] A textual reasoning step.
- [Tool] A tool invocation (All Available Tools above).
- [Tool Execution Output] The execution output (image, text, or highlighted region).

Evaluation Criteria:

1. Tool Invocation Correctness: Check if the invocation is valid, properly formatted, and consistent with the tool's definition and the corresponding Step ID.
2. Tool Execution Output Validation: Check if the output satisfies the intent of the corresponding Step ID and Tool ID. The output only needs to fulfill the specific step's purpose, not the overall problem. Evaluate based on the global image.

Attention:

- Only steps satisfying ALL criteria are listed as correct.
- Do not analyze for errors unless the output explicitly indicates a failure.
- Ignore image resolution/size info; assume the tool operates on the original source image.

Output Format (JSON-like):
Correct Tools: [Tool1, Tool2, Tool3...]
Brief Analysis: [Briefly explain why these tool invocations are correct and why others are incorrect]

Figure 18: Prompt used for evaluating tool invocation effectiveness.

Key Step Coverage Prompt

You are an expert system for verifying solutions to image-based problems. Your task is to: 1. Segment the provided solution into logical reasoning steps. 2. Match each ground truth middle step with the solution steps.
INPUT FORMAT: 1. Problem: The original question/task. 2. Solution: A continuous paragraph containing the model's reasoning. 3. Ground Truth: Essential steps required for a correct answer.
TASK:
Step 1: Segment the Solution
Divide the solution paragraph into distinct logical reasoning steps. Each step should represent a coherent reasoning unit.
Step 2: Match Ground Truth Steps
For each ground truth step, determine if it is matched in any of the solution steps based on the following:

- Content Match: Must match specific values and details.
- Tool Integrity: If tool execution has errors, the step is "Unmatched".
- Implicit Logic: Reasonable logical skips (like identifying obvious objects or using background knowledge) are permissible.

OUTPUT FORMAT (JSON):

```
{
  "solution_steps": ["Step 1...", "Step 2..."],
  "judgments": [
    {
      "step_index": <integer>,
      "judgment": "Matched" | "Unmatched",
      "reason": "Brief explanation."
    }
  ]
}
```

ADDITIONAL RULES: 1. Only output the JSON object with no extra text. 2. Judge each ground truth step in order without omission.
DATA:

- [Problem] {question}
- [Answer] {answer}
- [Solution] {solution}
- [Ground Truth Information] {gt_annotation}

Figure 19: Prompt used for key step coverage evaluation.