Are Reasoning Vision-Language Models Robust to Semantic Visual Distractions?

Yizheng Sun[1], Mochuan Zhan[1], Yanan Ma[2], Jia Tong See[1], Yifan Wang[1], Ziyi Wang[3], Hao Li[4], Yang Cui[1], Wenhao Cai[1], Jingyu Sun[1], Chenghua Lin[1], Riza Batista-Navarro[1], Jingyuan Sun[1,*]

[1]University of Manchester, [2]Mayo Clinic, [3]Marex, [4]Imperial College London,

***Correspondence:** jingyuan.sun@manchester.ac.uk

Abstract

Reasoning Vision-Language Models (VLMs) achieve strong performance on complex multimodal tasks, but reliable real-world application requires handling visual inputs that are messier than clean, curated benchmarks. Existing works mainly evaluate such reliability of VLMs through input corruptions, such as noise, blur and weather effects, which make visual evidence harder to perceive. This leaves a critical reliability failure mode underexplored: a model may perceive the evidence correctly, yet reason from plausible but irrelevant and distracting evidence and propagate this mistake to its final answer. To address this gap, we introduce **Distract-Bench**, a benchmark for evaluating VLM robustness to **semantic visual distractions**, defined as meaningful but task-irrelevant visual cues added to inputs while preserving the ground-truth answer. We comprehensively evaluate eight leading open-source and two closed-source VLMs across conventional vision corruptions and Distract-Bench. Our results show that Distract-Bench exposes a robustness failure distinct from vision corruptions: reasoning VLMs largely track their non-reasoning base models under perceptual degradation, but show consistently lower robustness to semantic distractions. Further analysis shows that these distractions often enter the reasoning process of VLMs, are treated as evidence, and lead to incorrect answers. Together, these findings reframe robustness evaluation for reasoning VLMs, shifting the focus from degraded perception to distractions for reliable real-world visual reasoning. Our data and code are available at https://github.com/Yizheng-Sun/Distract-Bench.

1 Introduction

Large Vision-Language Models (VLMs) have advanced rapidly, achieving strong performance across a wide range of vision-language tasks (Liu et al., 2023; Team, 2025c; OpenAI, 2024). More recently, inspired by reasoning-centric language models trained with reinforcement learning (Guo et al., 2025), a growing line of work has equipped VLMs with explicit long-form reasoning abilities (Yang et al., 2025; Huang et al., 2025; Leng et al., 2025; Meng et al., 2026). These reasoning-enhanced VLMs have shown substantial gains on multimodal mathematical reasoning (Lu et al., 2024; Wang et al., 2024), scientific understanding, and general visual question answering (Yue et al., 2024; Chen et al., 2024). This progress suggests that stronger reasoning is becoming a central path toward more capable multimodal systems.

Despite the strong performance of reasoning VLMs, reliable real-world application requires handling visual inputs that are messier than clean, curated benchmarks (Taori et al., 2020; Qiu et al., 2024). Existing visual robustness studies for VLMs mainly evaluate their reliability through visual corruptions, such as noise, blur, weather effects, and compression artifacts, as shown in Figure 1(A) (Hendrycks and Dietterich, 2019; Chen et al., 2023; Saxena et al., 2026). These visual corruptions do not introduce new semantic content. They primarily measure *perceptual robustness*: whether task-relevant evidence remains accessible when the input is degraded. They leave a reasoning-specific failure mode underexplored: a model may clearly perceive the relevant content, yet still attend to plausible but irrelevant visual information, use it as evidence during reasoning, and propagate this mistake to its final answer.

This failure mode is especially important for reasoning VLMs. Real multimodal inputs often contain extra objects, irrelevant text, misleading annotations, or contextual cues unrelated to the question (Selvaraju et al., 2019; Li et al., 2023b; Shi et al., 2023). Such distractors do not necessarily hide the correct evidence; instead, they compete with it during evidence selection. Once a model attends to the wrong cue, long-form reasoning can

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 2159–2176
November 6-10, 2026 ©2026 Association for Computational Linguistics

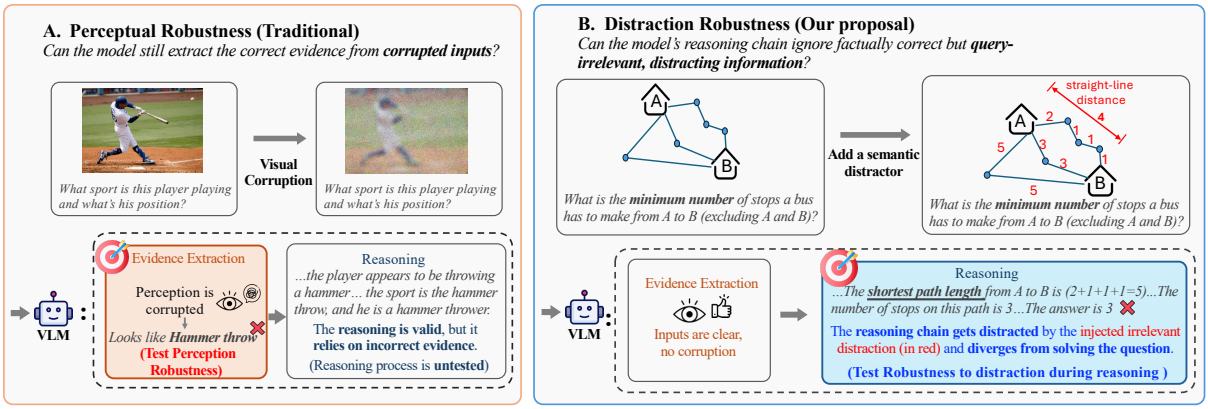


Figure 1: **From perceptual robustness to semantic-distraction robustness.** Traditional robustness evaluation tests whether VLMs can still perceive evidence from visually corrupted inputs, where failures mainly reflect degraded perception. Distract-Bench targets a different failure mode: the image remains clear and the ground-truth answer is unchanged, but a factually correct, query-irrelevant semantic distractor is injected. This setting tests whether reasoning VLMs can select the right evidence and ignore plausible but irrelevant visual cues, or whether such distractors enter the reasoning chain and lead to incorrect answers.

amplify the mistake, turning an initially irrelevant detail into a seemingly coherent but incorrect reasoning path as illustrated in Figure 1(B) (Shi et al., 2023; Turpin et al., 2023).

To study this failure mode, we introduce **Distract-Bench**, a human-verified benchmark for evaluating robustness under answer-preserving semantic distraction. Distract-Bench adds plausible but task-irrelevant semantic distractors to multimodal samples while preserving the original ground-truth answer. Because the relevant evidence remains available and the answer does not change, clean-to-distracted failures can be attributed to breakdowns in evidence selection rather than to changed task semantics, ordinary task difficulty, or degraded perceptual access.

This controlled setting allows us to address the central question of this paper: *How robust are reasoning VLMs to answer-preserving semantic visual distractions?* We decompose this question into three connected research questions. **RQ1:** Does answer-preserving semantic distraction expose a robustness failure mode distinct from conventional vision corruption, or is it merely another form of vision corruption? **RQ2:** Can reasoning VLMs maintain reliable performance under semantic distraction, given their strong reasoning capability? **RQ3:** When models fail under semantic distraction, do the added distractors enter the reasoning process and contribute to incorrect answers?

Our experiments answer these questions with three main findings:

- **Semantic distraction exposes a failure mode distinct from vision corruption.** Under vision corruptions, reasoning VLMs largely inherit their base models' robustness patterns, whereas Distract-Bench produces sharply different degradation behaviors.

- **Reasoning VLMs remain highly sensitive to semantic distraction.** Despite their strong clean performance, reasoning VLMs show clear performance degradation on Distract-Bench, indicating that strong reasoning capability does not eliminate sensitivity to task-irrelevant semantic cues.

- **Distractors often enter the reasoning process in failure cases.** Output-level analysis shows that VLMs frequently refer to distracting visual cues in wrong answers, often treating them as evidence and propagating them toward incorrect conclusions.

Together, these findings show that Distract-Bench captures a critical robustness problem beyond perception: whether VLMs can resist the misleading but answer-preserving semantic distractions. This makes distraction-robustness testing essential for developing reliable real-world VLMs, especially as long-form reasoning can amplify the impact of irrelevant evidence.

2 Related Work

2.1 Reasoning Vision-Language Models

Vision-Language Models (VLMs) have evolved from perception and instruction-following systems toward models with explicit multimodal reasoning capabilities. Early and widely used VLMs such as BLIP (Li et al., 2023a), LLaVA (Liu et al., 2023), Qwen3-VL (Team, 2025c) and GPT-4o (OpenAI, 2024) demonstrate strong visual understanding, grounding, and open-ended interaction over image-text inputs. More recently, inspired by the success of reasoning-centric language models (Guo et al., 2025; Team, 2025a,b), models such as Vision-R1 (Huang et al., 2025), R1-OneVision (Yang et al., 2025), OpenVLThinker (Deng et al., 2025), MMR1 (Leng et al., 2025), and MM-Eureka (Meng et al., 2026) explicitly enhance model reasoning capability over their base models by introducing supervised reasoning traces, distillation, or reinforcement learning to improve performance on complex multimodal tasks. This progression marks a shift from VLMs that primarily perceive and describe visual content to VLMs with enhanced reasoning ability over multimodal evidence. However, existing studies largely focus on benchmark accuracy gains, leaving it unclear whether such reasoning mechanisms also make VLMs more robust under distribution shifts or semantically distracting information.

2.2 Robustness Evaluation for Vision-Language Models

As VLMs are increasingly applied beyond curated benchmarks, robustness evaluation has become an important direction for understanding their reliability under distribution shifts. Existing studies largely follow the tradition of visual robustness and VQA robustness, testing whether models remain accurate under image corruptions (Hendrycks and Dietterich, 2019; Schiappa et al., 2022) and adversarial visual changes (Li et al., 2021). More recent benchmarks extend this evaluation to modern VLMs at larger scale: Chen et al. (2023) study the robustness of adaptation methods for pre-trained VLMs under both visual and textual corruptions. Sun et al. (2025) investigate how input corruption affects model acceleration methods on VLMs. VLM-RobustBench systematically evaluates current VLM families under a broad suite of visual distortions with varying severities but covers a limited set of benchmarks (Saxena et al., 2026). These works provide valuable evidence that VLM performance can be fragile under input-level shifts. However, they primarily stress the perception or surface-form processing of multimodal inputs, rather than the reliability of the reasoning process itself. In contrast, our work studies how reasoning-enabled VLMs behave under both conventional corruptions and semantically meaningful distractors that directly challenge the model's reasoning chain.

3 Distract-Bench Dataset Curation

This section describes the curation process of Distract-Bench as visualized in Figure 2. Distract-Bench is designed to evaluate whether VLMs can maintain a correct answer when the input contains factually correct, query-irrelevant, and distracting information. Its construction follows a four-stage pipeline: (i) raw sample collection, (ii) expert-designed seed distractors, (iii) scalable synthetic editing, and (iv) human quality verification.

Raw sample collection. We first collect raw image-question samples from established VLM benchmarks to reduce dataset-specific bias and ensure that the original tasks are well formed. Since distraction robustness primarily concerns failures in the reasoning process, we focus on reasoning-heavy benchmarks, including MathVision and MathVista, and further include samples from the general-domain benchmark MMMU to improve coverage and diversity (Wang et al., 2024; Lu et al., 2024; Yue et al., 2024). In total, we collect 800 raw samples, each consisting of an image and its corresponding textual query.

Expert-designed seed distractors. To define the target distraction carefully, human annotators first construct a set of seed examples. For each selected sample, annotators inspect both the image and the question, then add a semantic distractor into the image. The distractor is required to fulfil four criteria: (1) contain obvious semantic information; (2) be factually correct and visually plausible; (3) be irrelevant to the query and therefore not affect the correct answer; and (4) optionally lead to dead ends or plausible wrong answers. Annotators also provide a textual explanation describing what information is injected, why it is irrelevant to the original answer, and how it may potentially mislead the model's reasoning. This process yields 100 expert-designed seed examples, which serve as demonstrations for large-scale data generation.

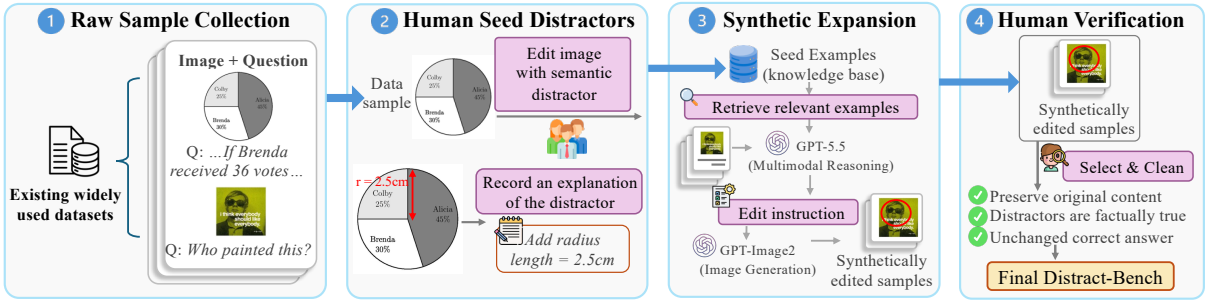


Figure 2: **Distract-Bench dataset curation pipeline.** We collect raw image-question samples from widely used VLM benchmarks (Yue et al., 2024; Lu et al., 2024; Wang et al., 2024), construct expert-designed seed distractors, scale the process with retrieval-guided synthetic editing, and finally conduct human expert verification to retain high-quality samples whose distractors are factually correct, query-irrelevant, and answer-preserving.

Scalable distractor synthesis. Following prior work on LLM-driven synthetic data generation and instruction-guided image editing (Wang et al., 2023; Long et al., 2024; Brooks et al., 2023), we use the expert-designed examples as seed patterns for scalable synthetic curation. For each seed, we leverage GPT-5.5 (OpenAI, 2026b) to summarize the original image, edited image, and distractor-induced reasoning effect, forming a knowledge base of human-verified edit patterns. For each remaining raw sample, GPT-5.5 describes the image, retrieves three relevant seed examples as in-context demonstrations, and generates a detailed edit instruction specifying the distractor content, location, visual style, and layout constraints. The detailed full prompt is shown in Appendix F.1. The original image and edit instruction are then passed to GPT-Image2 (OpenAI, 2026c) to synthesize the edited image while preserving all original content except the injected distractor. The edit instruction and generated image examples are shown in Appendix F.2. In parallel, GPT-5.5 produces a concise distractor description for later analysis of whether VLM outputs explicitly reference the distractor.

Human quality verification. To ensure that Distract-Bench isolates reasoning robustness rather than artifacts from image editing, human experts examine every generated sample. Each sample is checked against the four criteria as introduced above. Samples that violate any criterion are removed. After this verification and cleaning process, we obtain 506 high-quality samples that satisfy all requirements. These samples constitute Distract-Bench, a benchmark specifically designed to test the robustness of VLM reasoning under semantically meaningful but query-irrelevant distractions. Data consent and usage terms for Distract-Bench are discussed in Appendix A.1.

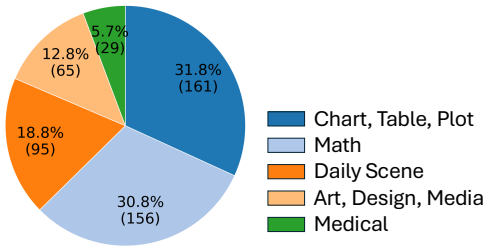


Figure 3: Category distribution of Distract-Bench samples. The benchmark contains 506 samples spanning diverse visual domains, including charts, tables, plots, mathematics, daily scenes, art/design/media, and medical images.

4 Experimental Design

Our experiments are designed to answer the central question posed above: *How robust are reasoning VLMs to answer-preserving semantic visual distractions?* We evaluate ten VLMs, including reasoning-enhanced models and corresponding non-reasoning reference models, under clean inputs, conventional vision corruptions, and Distract-Bench semantic distraction.

4.1 Setup

Tasks and Datasets. We evaluate models on a diverse suite of eight benchmarks covering complementary multimodal abilities. These include ChartQA (Masry et al., 2022), HallusionBench (Guan et al., 2024), InfoVQA (Mathew et al., 2022), MMMU (Yue et al., 2024), MMStar (Chen et al., 2024), TextVQA (Singh et al., 2019), MathVision (Wang et al., 2024), and MathVista (Lu et al., 2024). For analysis, we group MathVision and MathVista as *reasoning* benchmarks, while the remaining six datasets are grouped as *general* bench-

marks. Details about these datasets are discussed in Appendix A. The input corruption methods applied to these datasets are explained in Appendix B.

Models. We evaluate a broad set of VLMs, including open-source reasoning models, their non-reasoning base models, and closed-source frontier models. For controlled paired comparisons, we select models whose reasoning and non-reasoning variants share the same underlying architecture. Specifically, for Qwen2.5-VL (Bai et al., 2025), we use Qwen2.5-VL-7B (Bai et al., 2025) as the non-reasoning reference model and compare it with five reasoning-style variants built upon the same Qwen2.5-VL backbone: R1-OneVision (Yang et al., 2025), Vision-R1 (Huang et al., 2025), MMR1 (Leng et al., 2025), MM-Eureka (Meng et al., 2026), and OpenVL-Thinker (Deng et al., 2025). We further include Qwen3-VL-8B and its thinking counterpart, Qwen3-VL-8B-Thinking (Team, 2025c), as an additional paired comparison. Because models within each pair share the same architecture, these comparisons reduce architectural confounds and allow us to directly compare robustness between reasoning-style models and their corresponding non-reasoning references. To broaden the evaluation beyond these paired open-source models, we also evaluate GPT-4o and GPT-5 (OpenAI, 2024, 2026a) on Distract-Bench as closed-source reference points, providing additional validation that the benchmark reveals semantic-distraction effects beyond the controlled paired setting. The hyperparameter settings we use are listed in Appendix C.

Metrics. We report standard accuracy on clean, corrupted, and distracted inputs, and use Relative Robustness (RR) (Chen et al., 2023) to measure the fraction of clean performance preserved after perturbation, i.e., $\mathrm{RR} = P_O/P_I$, where P_I and P_O denote clean and perturbed performance. For sample-level analysis, we further report retention, defined as whether a model remains correct after corruption on samples it answered correctly in the clean setting. We also measure error overlap, measured by the Jaccard similarity between perturbation-induced error sets of a base–reasoning model pair. For Distract-Bench output analysis, we introduce two distractor-oriented metrics: Distraction Reference Rate (DRR), which measures how often the injected distractor appears in the model output, and Harmful Reference Rate (HFR), which measures how often such distractor reference is associated with an incorrect answer. We also report distractor-reference ratios among correct and wrong outputs to distinguish incidental mentions from failure-related distractor uptake. Detailed definitions of all metrics are provided in Appendix D.

		Reasoning			General		
Model	Type	Clean Acc. (%)	Visual Corrupted Acc. (%)	Relative Robust. (%) ↑	Clean Acc. (%)	Visual Corrupted Acc. (%)	Relative Robust. (%) ↑
Qwen2.5-VL family							
Qwen2.5	Base	46.6	40.2	89.9	73.0	60.7	84.5
R1-One	Reas.	46.8	40.0	86.6	70.3	56.3	81.6
Eureka	Reas.	50.4	45.2	91.6	72.4	60.4	84.9
MMR1	Reas.	48.9	42.4	89.4	70.2	57.4	83.1
Vision-R1	Reas.	59.5	51.8	87.5	73.7	60.8	84.0
OpenVLT	Reas.	47.6	41.2	88.1	74.1	61.8	84.8
Qwen3-VL family							
Qwen3-I	Base	70.4	64.7	**92.4**	74.9	63.9	**86.3**
Qwen3-T	Reas.	71.4	65.2	92.0	79.9	66.1	83.4

Table 1: Robustness evaluation results under conventional visual corruption across reasoning and general benchmarks. Clean accuracy, corrupted accuracy, and relative robustness (RR) are reported for base and reasoning VLMs. Models within the same family show similar robustness levels.

Pair	N	Agreement (%)	Retention gap @ 90% CI	Error overlap (Jaccard)
Q2.5→R1-One	10.1k	86.4	+5.8 [+5.2,+6.4]	.566
Q2.5→Eureka	11.4k	96.9	−0.1 [−0.4,+0.1]	.870
Q2.5→MMR1	10.7k	91.3	+2.7 [+2.2,+3.2]	.681
Q2.5→V-R1	10.8k	90.9	+0.8 [+0.3,+1.3]	.663
Q2.5→OVT	11.1k	92.1	−0.6 [−1.1,−0.2]	.695
Q3-I→Q3-T	11.2k	89.2	+2.7 [+2.2,+3.2]	.578
Mean	–	91.1	+1.9	.675

Table 2: Matched-sample robustness similarity between reasoning VLMs and their base models under vision corruption. N counts the correct samples under clean condition shared by each base–reasoning pair. We report sample-wise agreement, base-minus-reasoning retention ratio gap with 90% confidence interval (CI), and Jaccard overlap of corruption-induced errors; positive gaps indicate higher base-model retention. Most pairs show highly aligned sample-wise patterns.

5 Result Analysis

This section presents the empirical answers to the three research questions. Section 5.1 shows that semantic distraction produces a robustness pattern distinct from conventional vision corruption: reasoning VLMs largely follow their corresponding base models under vision corruptions, but exhibit different degradation behavior on Distract-Bench. Section 5.2 examines whether reasoning VLMs maintain reliable performance under semantic distraction, showing that despite strong clean perfor-

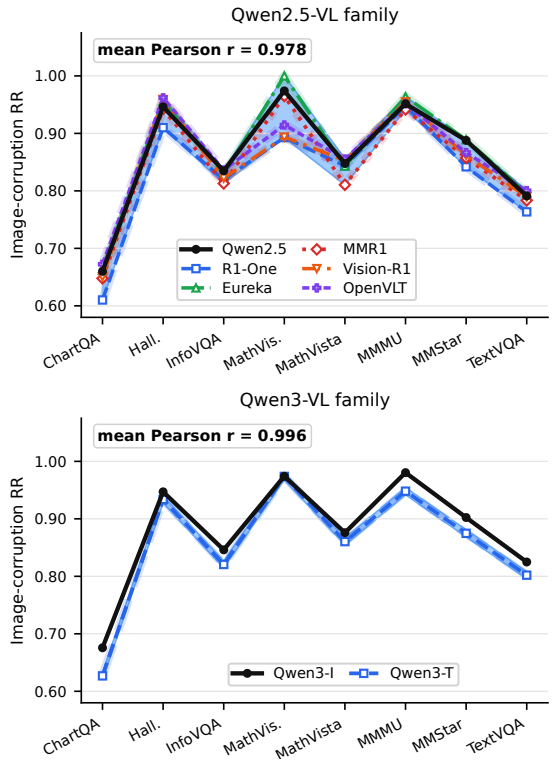


Figure 4: Per-benchmark relative robustness (RR) under vision corruption. Reasoning models closely track their base models across benchmarks, indicating highly inherited perceptual robustness within each family.

mance, they remain sensitive to task-irrelevant semantic cues. Section 5.3 further analyzes model outputs, showing that distractors often appear in failure cases, are treated as evidence, and are propagated toward incorrect answers.

5.1 Semantic Distraction Exposes a Failure Mode Distinct from Vision Corruption

Vision corruption robustness is largely inherited from the base model. We first examine conventional visual corruption as the perceptual-robustness baseline. As shown in Table 1, reasoning models remain close to their base models under visual corruption. Figure 4 further shows highly similar per-benchmark robustness fingerprints within each family, with mean Pearson correlations of 0.978 for Qwen2.5-VL and 0.996 for Qwen3-VL. The full detailed results for each model on each benchmark are shown in Appendix E. Table 2 confirms this similarity at the sample level: base–reasoning pairs agree on retain/loss outcomes for 91.1% of clean-correct samples on average, with substantial corruption-error overlap (mean Jaccard 0.675) and only a small average retention gap of +1.9 pp. Together, these results indicate that conventional image-corruption robustness is largely inherited from the base model.

Model	Type	N	Clean Acc. (%)	Semantically Distracted Acc. (%)	Drop Acc. (pp) ↓	Relative Robust. (%) ↑
Qwen2.5-VL family						
Qwen2.5	Base	506	80.0	73.7	6.3	92.1
R1-One	Reas.	506	71.9	61.3	10.7	85.2
Eureka	Reas.	506	81.6	71.9	9.7	88.1
MMR1	Reas.	506	83.4	75.3	8.1	90.3
Vision-R1	Reas.	506	77.1	66.0	11.1	85.6
OpenVLT	Reas.	506	82.2	69.4	12.8	84.4
Qwen3-VL family						
Qwen3-VL-I.	Base	506	84.8	83.6	1.2	**98.6**
Qwen3-VL-T.	Reas.	506	85.0	81.0	4.0	95.3
Closed-source Models						
GPT-4o	Closed-s.	506	76.7	69.2	7.5	90.2
GPT-5	Closed-s.	506	81.6	78.1	3.6	**95.6**

Table 3: Distract-Bench robustness evaluation results under answer-preserving semantic visual distractions. We report clean and distracted accuracy, accuracy drop in percentage points, and relative robustness (RR). Reasoning models are consistently more sensitive to distractions than their base models.

Distract-Bench breaks the inherited robustness pattern. If semantic distraction were simply another form of vision corruption, we would expect reasoning-tuned models to remain similarly aligned with their base models. Table 3 shows a different pattern. On Distract-Bench, the Qwen2.5-VL base model retains 92.1% relative robustness, while its reasoning-tuned variants drop to 84.4%–90.3%. The same trend appears in the Qwen3-VL family: Qwen3-I is nearly unaffected by distraction, with 98.6% relative robustness, whereas Qwen3-T drops to 95.3%. Thus, unlike vision corruption, semantic distraction creates clearer base–reasoning robustness differences.

To rule out differences in sample composition, we further compare vision corruption and Distract-Bench distraction on the same 506 original samples in Table 4. Under vision corruption, the mean relative-robustness gap between base and reasoning models is only -1.3 pp, and the signed retention gap is nearly zero at -0.3 pp. Under Distract-Bench distraction, these gaps increase to 5.0 pp and $+5.9$ pp, respectively, showing a stronger and more consistent base-model advantage. These results answer RQ1: Distract-Bench exposes a distinct distraction-robustness axis where visible but distracting semantic cues change both robustness gaps and failure patterns.

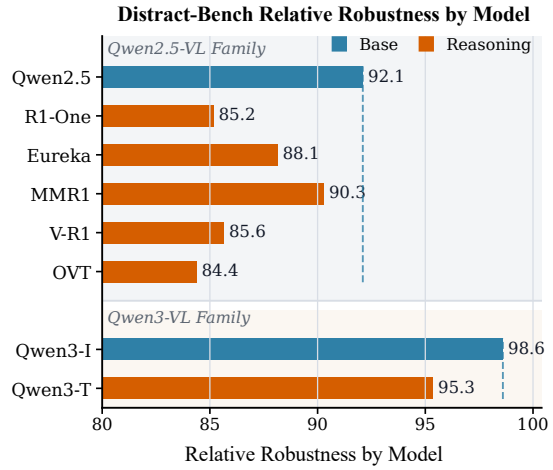


Figure 5: Distract-Bench relative robustness for base and reasoning-tuned models. Dashed lines mark the corresponding base-model robustness within each family; reasoning models consistently fall below their base models under semantic distraction.

	Vision Corruption				Distract-Bench			
Pair	RR gap	Agr. (%)	Ret. gap	Err. Jac.	RR gap	Agr. (%)	Ret. gap	Err. Jac.
Qwen2.5-VL →								
R1-One	-1.1	79.3	+3.9	0.289	+6.9	77.4	+14.2	0.247
Eureka	-1.9	90.7	-1.9	0.562	+4.0	82.0	+2.8	0.247
MMR1	-0.2	80.4	+2.6	0.330	+1.8	82.4	+3.1	0.267
V-R1	-3.7	83.3	-3.0	0.322	+6.5	81.6	+7.1	0.279
OVT	-2.9	83.8	-1.9	0.344	+7.7	80.1	+4.8	0.228
Qwen3-VL-Instruct →								
Qwen3-VL-T.	+2.0	89.3	-1.2	0.430	+3.3	90.1	+3.3	0.204
Mean	-1.3	84.5	-0.3	0.380	+5.0	82.3	+5.9	0.246

Table 4: Controlled 506-sample comparison of base and reasoning model pairs under Visual Corruption and semantic distractions. RR gap measures the base-minus-reasoning robustness-rate difference, Agr. is matched-sample agreement, Ret. gap is the correctness-retention difference, and Err. Jac. is the Jaccard overlap of induced errors. Distract-Bench demonstrates sharply distinct sample-wise patterns compared with vision corruption.

5.2 Reasoning VLMs Remain Sensitive to Semantic Distraction

Table 3 reports Distract-Bench results on the evaluated open-source VLMs, together with GPT-4o and GPT-5 as closed-source reference points. Although reasoning VLMs generally achieve strong clean performance, they still show clear degradation when answer-preserving semantic distractors are added. As illustrated in Figure 5, in the Qwen2.5-VL family, the non-reasoning reference model retains 92.1% relative robustness, while the reasoning VLMs retain 84.4%–90.3%. Their absolute accuracy drops are also larger, increasing from 6.3 pp for the reference model to 8.1–12.8 pp for reasoning VLMs. A similar pattern appears in the Qwen3-VL family.

The sample-matched results in Table 4 provide additional context for this sensitivity. On the same 506 Distract-Bench source samples, the mean base-minus-reasoning relative-robustness gap is +5.0 pp, and the mean retention gap is +5.9 pp. These positive gaps indicate that, in the evaluated paired comparisons, reasoning VLMs less often preserve correct clean predictions after semantic distraction. Together, these results answer RQ2: reasoning VLMs are sensitive to answer-preserving semantic distractions despite their strong reasoning capability. This suggests that reliable reasoning under clean inputs is not sufficient to ensure robust evidence selection when plausible but task-irrelevant visual cues are present.

5.3 Distractors Often Derail the Reasoning Process Itself

Reasoning models reference distractors more often. To determine whether Distract-Bench failures are caused by the injected semantic distractors, we analyze model outputs for explicit distractor reference. For a fair comparison, we use the same open-ended prompting for all models, without instructing either base or reasoning models to produce short or direct answers. Table 5 shows that reasoning-enhanced models generally refer to distractors more often than their corresponding base models. In the Qwen2.5-VL family, the base model has a DRR of 19.0%, while most reasoning variants are higher. More importantly, as shown in Figure 6B, harmful reference ratios are consistently higher for reasoning models than for their base models. This suggests that failures under semantic distraction are not merely unexplained answer flips. Instead, irrelevant visual cues are often incorporated into the model's reasoning process and treated as supporting evidence.

Distractor reference is concentrated in wrong answers. Across all models, wrong outputs reference the distractor much more often than correct outputs, as visualized in Figure 6A. The same trend also appears for GPT-4o and GPT-5, whose wrong-output reference ratios are substantially higher than their correct-output reference ratios. This pattern indicates that distractor reference is not merely incidental: when models fail, they are substantially

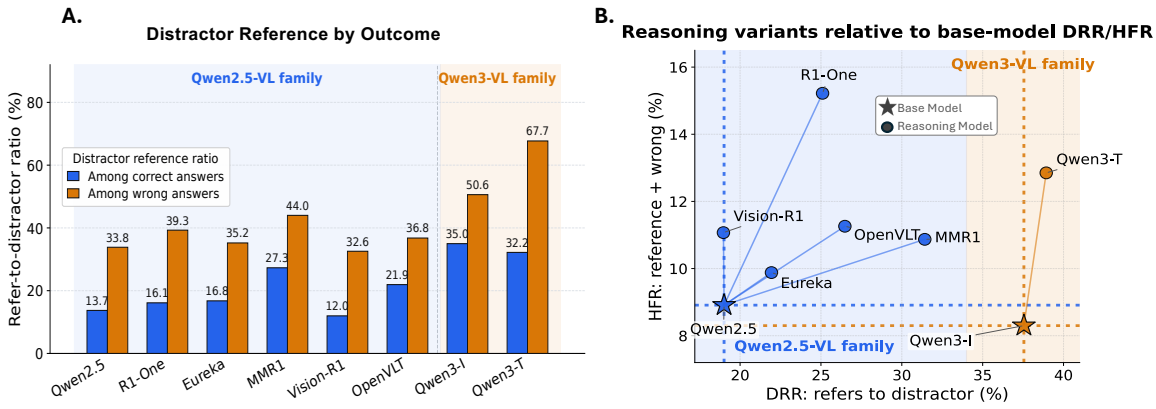


Figure 6: **VLM distractor-reference behavior on Distract-Bench.** (A) Distractor-reference ratios among correctly and incorrectly answered samples, grouped by model family. (B) Harmful Reference Rate (HFR), measuring distractor mentions in incorrect answers, for reasoning-tuned models relative to the corresponding base model within each family.

Model	Type	DRR (%)	HFR (%) ↓	Correct outputs ref. ratio (%)	Wrong outputs ref. ratio (%)
Qwen2.5-VL family					
Qwen2.5	Base	19.0	8.9	13.7	33.8
R1-One	Reas.	25.1	15.2	16.1	39.3
Eureka	Reas.	21.9	9.9	16.8	35.2
MMR1	Reas.	31.4	10.9	27.3	44.0
Vision-R1	Reas.	19.0	11.1	12.0	32.6
OpenVLT	Reas.	26.5	11.3	21.9	36.8
Qwen3-VL family					
Qwen3-I	Base	37.5	**8.3**	35.0	50.6
Qwen3-T	Reas.	38.9	12.8	32.2	67.7
Closed-source Models					
GPT-4o	Closed-s.	22.5	13.1	13.5	43.8
GPT-5	Closed-s.	18.4	11.3	9.1	53.4

Table 5: Distractor-reference analysis on Distract-Bench. DRR measures how often outputs mention the injected distractor, while HFR counts harmful mentions that occur in incorrect Distract-Bench answers. Correct-output and wrong-output reference ratios measure how often the distractor is mentioned among correctly and incorrectly answered samples, respectively.

more likely to have brought the irrelevant cue into the reasoning process. Figure 6B complements this observation by showing that reasoning-tuned models not only mention distractors more often, but also more often mention them in harmful failure cases relative to their base models. Figure 7 provides a qualitative illustration, showing that an injected cue can be ignored or mentioned harmlessly, but in failure cases it is treated as evidence and propagated to the wrong answer. More qualitative examples are included in Appendix G.

These output-level results answer RQ3: semantic distractors do not simply cause unexplained answer flips. They often enter the model's reasoning, are treated as relevant evidence, and are carried forward to incorrect conclusions. This explains why Distract-Bench is especially revealing for reasoning VLMs: Reasoning can amplify an irrelevant distraction once it is selected as part of the evidence chain.

6 Conclusion

This work identifies semantic visual distraction as a critical robustness challenge for reasoning-enhanced VLMs and introduces Distract-Bench. Unlike conventional visual corruptions, which mainly test whether models can perceive degraded evidence, Distract-Bench tests whether models can ignore clear but query-irrelevant semantic distractions. Our results show that stronger reasoning does not necessarily bring stronger reliability: reasoning VLMs largely follow their base models under perceptual degradation, but often show lower robustness under semantic distraction. Further analysis shows that distractors can enter the reasoning process, be treated as evidence, and lead to incorrect answers. These findings motivate a broader view of VLM robustness: reliable visual reasoning should be evaluated not only under degraded perception, but also under misleading semantic distractions.

7 Limitations

Our study has two main limitations. First, Distract-Bench uses human-verified semantic distractors injected through controlled image edits under laboratory conditions; while this enables clean evaluation and helps isolate answer-preserving distraction from perception degradation, it may not cover all

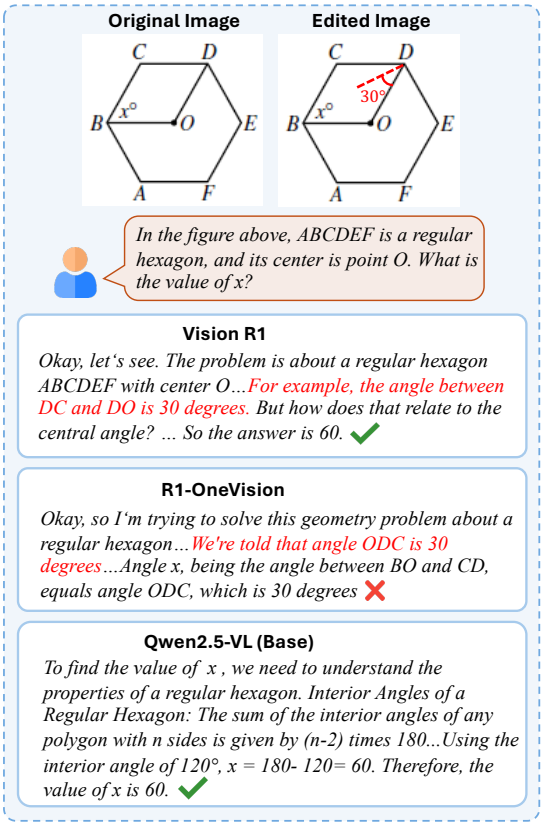


Figure 7: **Qualitative example of distraction-induced reasoning failure.** The injected irrelevant cue may be ignored or mentioned harmlessly, but in failure cases it is treated as evidence and propagated to an incorrect answer.

types of real-world distractions, which are highly varied. Collecting and curating distractions from naturally occurring real-world samples is therefore a promising direction for future work. Second, our work is primarily diagnostic: we reveal robustness risks introduced by reasoning enhancement and show that distractors can enter the reasoning process, but we do not propose a mitigation method. Developing VLMs that are both reasoning-capable and robust remains an important future direction, potentially through improved evidence selection, distraction-aware training, or inference-time verification.

References

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. *CoRR*, abs/2502.13923.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. 2023. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, pages 18392–18402. IEEE.

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? In *NeurIPS*.

Shuo Chen, Jindong Gu, Zhen Han, Yunpu Ma, Philip H. S. Torr, and Volker Tresp. 2023. Benchmarking robustness of adaptation methods on pre-trained vision-language models. In *NeurIPS*.

Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. 2025. Openvlthinker: Complex vision-language reasoning via iterative SFT-RL cycles. In *NeurIPS*.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. 2024. Hallusionbench: An advanced diagnostic suite for entangled language hallucination and visual illusion in large vision-language models. In *CVPR*, pages 14375–14385. IEEE.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nat.*, 645(8081):633–638.

Dan Hendrycks and Thomas G. Dietterich. 2019. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR (Poster)*. OpenReview.net.

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. *CoRR*, abs/2503.06749.

Sicong Leng, Jing Wang, Jiaxi Li, Hao Zhang, Zhiqiang Hu, Boqiang Zhang, Yuming Jiang, Hang Zhang, Xin Li, Lidong Bing, Deli Zhao, Wei Lu, Yu Rong, Aixin Sun, and Shijian Lu. 2025. MMR1: enhancing multimodal reasoning with variance-aware sampling and open resources. *CoRR*, abs/2509.21268.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. 2023a. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 19730–19742. PMLR.

Linjie Li, Jie Lei, Zhe Gan, and Jingjing Liu. 2021. Adversarial VQA: A new benchmark for evaluating the robustness of VQA models. In *ICCV*, pages 2022–2031. IEEE.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023b. Evaluating object hallucination in large vision-language models. In *EMNLP*, pages 292–305. Association for Computational Linguistics.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *NeurIPS*.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. In *ACL (Findings)*, volume ACL 2024 of *Findings of ACL*, pages 11065–11082. Association for Computational Linguistics.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. 2024. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *ICLR*. OpenReview.net.

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq R. Joty, and Enamul Hoque. 2022. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. In *ACL (Findings)*, volume ACL 2022 of *Findings of ACL*, pages 2263–2279. Association for Computational Linguistics.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and C. V. Jawahar. 2022. Infographicvqa. In *WACV*, pages 2582–2591. IEEE.

Fanqing Meng, Lingxiao Du, Zongkai Liu, Zhixiang Zhou, Quanfeng Lu, Tiancheng Han, Daocheng Fu, Kaipeng Zhang, Ping Luo, Yu Qiao, Jiaheng Zhang, Michael Qizhe Shieh, Qiaosheng Zhang, and Wenqi Shao. 2026. Mm-eureka: Toward stable multimodal reasoning via rule-based reinforcement learning with policy drift control. *Trans. Mach. Learn. Res.*, 2026.

OpenAI. 2024. Gpt-4o system card. *CoRR*, abs/2410.21276.

OpenAI. 2026a. Openai GPT-5 system card. *CoRR*, abs/2601.03267.

OpenAI. 2026b. Gpt-5.5 system card. https://openai.com/index/gpt-5-5-system-card/. Accessed: 2026-09-22.

OpenAI. 2026c. Image generation. https://developers.openai.com/api/docs/guides/image-generation. Accessed: 2026-05-19.

Jielin Qiu, Yi Zhu, Xingjian Shi, Florian Wenzel, Zhiqiang Tang, Ding Zhao, Bo Li, and Mu Li. 2024. Benchmarking robustness of multimodal image-text models under distribution shift. *J. Data-centric Mach. Learn. Res.*, 1:(2):1–56.

Rohit Saxena, Alessandro Suglia, and Pasquale Minervini. 2026. Vlm-robustbench: A comprehensive benchmark for robustness of vision-language models. *CoRR*, abs/2603.06148.

Madeline Schiappa, Shruti Vyas, Hamid Palangi, Yogesh S. Rawat, and Vibhav Vineet. 2022. Robustness analysis of video-language models against visual and language perturbations. In *NeurIPS*.

Ramprasaath Ramasamy Selvaraju, Stefan Lee, Yilin Shen, Hongxia Jin, Shalini Ghosh, Larry P. Heck, Dhruv Batra, and Devi Parikh. 2019. Taking a HINT: leveraging explanations to make vision and language models more grounded. In *ICCV*, pages 2591–2600. IEEE.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 31210–31227. PMLR.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards VQA models that can read. In *CVPR*, pages 8317–8326. Computer Vision Foundation / IEEE.

Yizheng Sun, Hao Li, Chang Xu, Hongpeng Zhou, Chenghua Lin, Riza Batista-Navarro, and Jingyuan Sun. 2025. Does acceleration cause hidden instability in vision language models? uncovering instance-level divergence through a large-scale empirical study. In *EMNLP*, pages 8442–8456. Association for Computational Linguistics.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. 2020. Measuring robustness to natural distribution shifts in image classification. In *NeurIPS*.

Kimi Team. 2025a. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599.

Qwen Team. 2025b. Qwen3 technical report. *CoRR*, abs/2505.09388.

Qwen Team. 2025c. Qwen3-vl technical report. *CoRR*, abs/2511.21631.

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

Ke Wang, Junting Pan, Weikang Shi, Zimu Lu, Houxing Ren, Aojun Zhou, Mingjie Zhan, and Hongsheng Li. 2024. Measuring multimodal mathematical reasoning with math-vision dataset. In *NeurIPS*.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *ACL (1)*, pages 13484–13508. Association for Computational Linguistics.

Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, Bo Zhang, and Wei Chen. 2025. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization. In *ICCV*, pages 2376–2385. IEEE.

Xiang Yue, Yuansheng Ni, Tianyu Zheng, Kai Zhang, 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 *CVPR*, pages 9556–9567. IEEE.

A Benchmark and Task Details

Benchmark	Split	Number of Samples
ChartQA	Test	2,500
HallusionBench	With Image	951
InfoVQA	Validation	2,801
MMMU	Validation	900
MMStar	Validation	1,500
TextVQA	Validation	5,000
MathVision	Test-Mini	304
MathVista	Test-Mini	1,000
Total	–	**14,956**

Table 6: Summary of benchmark datasets, evaluation splits, and sample sizes used in our robustness evaluation.

We conduct evaluation on eight established multimodal benchmarks, as summarized in Table 6. The selected datasets span a broad range of VLM capabilities, including chart and document understanding (ChartQA (Masry et al., 2022), InfoVQA (Mathew et al., 2022), TextVQA (Singh et al., 2019)), hallucination and visual consistency evaluation (HallusionBench (Guan et al., 2024)), general multimodal reasoning (MMMU (Yue et al., 2024), MMStar (Chen et al., 2024)), and multimodal mathematical reasoning (MathVision (Wang et al., 2024), MathVista (Lu et al., 2024)). We use the standard validation or test splits when available, and adopt the commonly used mini-test splits for MathVision and MathVista. Overall, the evaluation contains 14,956 samples, enabling a comprehensive assessment across both perception-centric and reasoning-intensive tasks.

A.1 Data Consent

Distract-Bench is constructed from existing open-source academic benchmarks, including MMMU, MathVision, and MathVista, and is intended solely for academic research. We use these source datasets only to study and evaluate the robustness of VLM reasoning under controlled semantic distractions. The constructed samples are released for non-commercial academic use, with the goal of supporting reproducible evaluation and future research on reliable multimodal reasoning. All released data will follow the usage terms of the original datasets, and Distract-Bench should not be used for commercial deployment, user profiling, or any application beyond research and evaluation.

B Input Corruption Methods

To evaluate perceptual robustness, we apply a suite of 20 visual corruption methods inspired by methods from the work by Chen et al. (2023). These corruptions are designed to simulate natural distribution shifts while preserving the underlying task semantics, allowing us to test whether VLMs can maintain stable predictions when the input signal is degraded. The corruption suite is summarized in Table 7. We adopt a broad set of ImageNet-C-style perturbations covering noise, blur, weather and environmental effects, color and appearance changes, compression artifacts, geometric distortion, and complete image removal (Chen et al., 2023; Hendrycks and Dietterich, 2019). Most image corruptions are applied with severity levels from 1 to 5, where higher severity corresponds to stronger degradation. These perturbations primarily stress the visual perception stage of VLM inference.

Category	Vision Perturbation Method
Noise	Blank Image Gaussian Noise Shot Noise Impulse Noise Speckle Noise
Blur	Gaussian Blur Glass Blur Defocus Blur Motion Blur Zoom Blur
Weather / Env.	Fog Frost Snow Spatter
Appearance	Contrast Brightness Saturate JPEG Compression Pixelate Elastic Transform

Table 7: Summary of the 20 vision corruption methods used for perceptual robustness evaluation.

C Hyper-parameter Settings

For all evaluations, we use deterministic decoding to ensure reproducibility and fair comparison across models. Unless otherwise specified, we set the maximum number of output tokens to 4096 for each benchmark. For reasoning-intensive mathematical benchmarks, namely MathVision (Wang et al., 2024) and MathVista (Lu et al., 2024), we increase the maximum number of output tokens to 16384, allowing models sufficient space to generate long-form reasoning chains before producing the final answer.

For Distract-Bench and the output-level distractor-reference analysis, we use the same open-ended prompt for all models and do not add any instruction that asks models to produce short or direct answers. This allows both base and reasoning models to generate intermediate reasoning when needed before giving the final answer.

For MathVista, we follow common evaluation practice and report results under three prompt settings: `cot`, `format`, and `solution` (Lu et al., 2024). The `cot` prompt uses a step-by-step style, presenting the original question and choices while encouraging the model to reason before placing the final answer at the end. The `format` prompt is the most constrained setting, asking the model to output only the required answer format, such as an option letter, integer, float, or Python list, which facilitates answer extraction but discourages verbose reasoning. The `solution` prompt appends `Solution:` before generation, explicitly nudging the model to produce a worked solution followed by the final answer. Thus, `cot` is reasoning-friendly, `format` is answer-only and format-focused, and `solution` frames the response as a solution write-up.

D Metrics Details

We define the following indicators for each sample i: $c_i = 1$ if the model prediction is correct and 0 otherwise; $r_i = 1$ if the model output explicitly refers to the injected distractor and 0 otherwise. When needed, superscripts I and O denote the clean input and perturbed input, respectively.

Accuracy. Accuracy (Acc.) measures standard task performance under a given input condition:

$$\mathrm{Acc} = \frac{1}{N}\sum_{i=1}^{N} c_i.$$

We report accuracy on clean inputs, visually corrupted inputs, and Distract-Bench distracted inputs.

Relative Robustness. Relative Robustness (RR) (Chen et al., 2023) measures the fraction of clean performance preserved after perturbation:

$$\mathrm{RR} = \frac{P_O}{P_I},$$

where P_I is clean accuracy and P_O is accuracy under the perturbed condition. A higher RR indicates stronger robustness, with $\mathrm{RR} = 1$ meaning that the model fully preserves its clean performance.

Retention. Retention is used in matched sample-level analysis. It measures how often a model remains correct after perturbation among samples it answered correctly on the clean input:

$$\mathrm{Retention} = \frac{1}{|\mathcal{C}|}\sum_{i\in\mathcal{C}} c_i^O, \quad \mathcal{C} = \{i : c_i^I = 1\}.$$

For base–reasoning comparisons, the retention gap is reported as base retention minus reasoning-model retention; positive values indicate higher base-model retention.

Error Overlap. Error overlap measures whether a base model and its reasoning-tuned counterpart fail on the same samples after perturbation. Let $\mathcal{E}_b$ and $\mathcal{E}_r$ denote the perturbation-induced error sets of the base and reasoning models:

$$\mathrm{Jaccard} = \frac{|\mathcal{E}_b \cap \mathcal{E}_r|}{|\mathcal{E}_b \cup \mathcal{E}_r|}.$$

A higher value indicates more shared failure cases, while a lower value indicates more divergent perturbation-induced failures.

Distraction Reference Rate. Distraction Reference Rate (DRR) is our proposed metric for measuring how often the model explicitly refers to the injected semantic distractor:

$$\mathrm{DRR} = \frac{1}{N}\sum_{i=1}^{N} r_i.$$

DRR captures how frequently irrelevant semantic evidence enters the model's reasoning or final response.

Model	Category	Clean Results (Acc@1 %)									
		ChartQA	Hallusion Bench	InfoVQA	MMMU	MMStar	TextVQA	MathVision	MathVista		
									cot	*format*	*solution*
Qwen2.5-VL Family											
Qwen2.5-VL-7B (Base)	**Non-Thinking**	81.76	68.35	81.51	52.67	61.2	92.28	25.33	69.3	70.7	67.8
R1-Onevision	**Thinking**	82.2	66.56	78.9	51.22	58.2	85	27.63	62.9	64.9	66
MM-Eureka		81.84	64.98	81.72	52.11	61.2	92.34	29.61	72	71.9	71.2
MMR1		76.88	69.3	79.4	45.67	58.87	91	28.29	68.8	60.6	69.6
Vision-R1		84.72	66.67	82.54	54.11	63.6	90.28	46.38	73.5	72.4	72.6
OpenVLThinker		84.48	68.98	85.29	52.89	59.4	93.46	26.97	69.6	68.2	68.2
Qwen3-VL Family											
Qwen3-VL-8B (Base)	**Non-Thinking**	79.92	73.4	83.04	56.56	64.13	92.28	63.49	77	69.8	77.4
Qwen3-VL-8B-Thinking	**Thinking**	86.6	73.08	87.36	70.67	73.47	88.5	62.83	80.2	80.3	79.9

Table 8: Clean accuracy results across all evaluated models and benchmarks. Results are reported as Acc@1 (%).

Model	Category	Image Corrupted Results (Acc@1 %)									
		ChartQA	Hallusion Bench	InfoVQA	MMMU	MMStar	TextVQA	MathVision	MathVista		
									cot	*format*	*solution*
Qwen2.5-VL Family											
Qwen2.5-VL-7B (Base)	**Non-Thinking**	53.96	64.67	68.05	50.11	54.33	73	24.67	59	61.5	55.7
R1-Onevision	**Thinking**	50.16	60.57	64.66	48.56	49.07	64.9	24.67	53.5	54.6	55.4
MM-Eureka		54.16	62.15	67.73	50.22	54.4	73.52	31.25	60.6	61.5	59.2
MMR1		49.8	65.3	64.55	43	50.47	71.3	27.3	56	48.1	57.4
Vision-R1		55.12	63.93	67.87	51.67	54.8	71.3	41.45	62.4	61.8	62.1
OpenVLThinker		56.8	66.35	71.33	50.33	51.47	74.8	24.67	59.4	59	57.8
Qwen3-VL Family											
Qwen3-VL-8B (Base)	**Non-Thinking**	54	69.51	70.26	55.44	57.87	76.12	61.84	67.6	61.2	67.5
Qwen3-VL-8B-Thinking	**Thinking**	54.28	68.24	71.69	67	64.27	70.96	61.18	67.8	69.7	69.2

Table 9: Accuracy results under visual corruptions across all evaluated models and benchmarks. Results are reported as Acc@1 (%).

Harmful Reference Rate. Harmful Reference Rate (HFR) is our proposed metric for measuring how often distractor reference is associated with an incorrect answer:

$$\mathrm{HFR} = \frac{1}{N}\sum_{i=1}^{N} r_i(1-c_i).$$

Unlike DRR, which counts all distractor references, HFR focuses on cases where the model both mentions the distractor and gives an incorrect answer.

Correct and Wrong Output Reference Ratios. To distinguish incidental distractor mentions from failure-related distractor uptake, we separately compute distractor-reference ratios among correct and wrong outputs:

$$\mathrm{Ref}_{\mathrm{correct}} = \frac{\sum_i r_i c_i}{\sum_i c_i},$$

$$\mathrm{Ref}_{\mathrm{wrong}} = \frac{\sum_i r_i(1-c_i)}{\sum_i (1-c_i)}.$$

The correct-output reference ratio measures how often the distractor is mentioned despite a correct answer, while the wrong-output reference ratio measures how often distractors appear in failed outputs.

E Full Results under Vision Corruption

This section provides the full per-benchmark results for the vision-corruption experiments discussed in Section 5.1. Table 8 reports clean Acc@1 for all models and benchmarks, Table 9 reports Acc@1 after applying vision corruptions, and Table 10 reports the corresponding Relative Robustness (RR). These tables complement the aggregate analysis in the main paper by showing that the similarity between base and reasoning-tuned models is not driven by a single benchmark, but is broadly observed across chart understanding, document VQA, multimodal knowledge, text-rich VQA, and visual reasoning tasks.

The clean results in Table 8 show that reasoning-tuned models often change task performance, sometimes improving clean accuracy on reasoning-

Model	Category	Robustness to Image Input Perturbation (RR%) ↑										
		ChartQA	Hallusion Bench	InfoVQA	MMMU	MMStar	TextVQA	MathVision	MathVista			Average
									cot	*format*	*solution*	
Qwen2.5-VL Family												
Qwen2.5-VL-7B (Base)	**Non-Thinking**	66	94.62	83.49	95.15	88.78	79.11	97.4	84.97	86.99	82.32	85.9
R1-Onevision		61.02 $_{-4.98}$	91.00 $_{-3.62}$	81.95 $_{-1.54}$	94.79 $_{-0.36}$	84.19 $_{-4.59}$	76.35 $_{-2.76}$	89.29 $_{-8.11}$	85.06 $_{+0.09}$	84.13 $_{-2.86}$	83.86 $_{+1.54}$	83.2 $_{-2.7}$
MM-Eureka		66.18 $_{+0.18}$	95.63 $_{+1.01}$	82.87 $_{-0.62}$	96.38 $_{+1.23}$	88.89 $_{+0.11}$	79.62 $_{+0.51}$	100.00 $_{+2.60}$	84.17 $_{-0.80}$	85.62 $_{-1.37}$	83.15 $_{+0.83}$	**86.3** $_{+0.4}$
MMR1	**Thinking**	64.78 $_{-1.22}$	94.23 $_{-0.39}$	81.29 $_{-2.20}$	94.16 $_{-0.99}$	85.73 $_{-3.05}$	78.35 $_{-0.76}$	96.51 $_{-0.89}$	81.40 $_{-3.57}$	79.37 $_{-7.62}$	82.39 $_{+0.07}$	83.8 $_{-2.1}$
Vision-R1		65.06 $_{-0.94}$	95.90 $_{+1.28}$	82.22 $_{-1.27}$	95.48 $_{+0.33}$	86.16 $_{-2.62}$	78.98 $_{-0.13}$	89.36 $_{-8.04}$	84.90 $_{-0.07}$	85.36 $_{-1.63}$	85.62 $_{+3.30}$	84.9 $_{-1.0}$
OpenVLThinker		67.23 $_{+1.23}$	96.19 $_{+1.57}$	83.63 $_{+0.14}$	95.17 $_{+0.02}$	86.64 $_{-2.14}$	80.03 $_{+0.92}$	91.46 $_{-5.94}$	85.26 $_{+0.29}$	86.51 $_{-0.48}$	84.84 $_{+2.52}$	85.7 $_{-0.2}$
Qwen3-VL Family												
Qwen3-VL-8B (Base)	**Non-Thinking**	67.57	94.7	84.61	98.04	90.23	82.49	97.41	87.79	87.68	87.3	**87.8**
Qwen3-VL-8B-Thinking	**Thinking**	62.68 $_{-4.89}$	93.38 $_{-1.32}$	82.06 $_{-2.55}$	94.81 $_{-3.23}$	87.48 $_{-2.75}$	80.20 $_{-2.29}$	97.38 $_{-0.03}$	84.62 $_{-3.17}$	86.80 $_{-0.88}$	86.69 $_{-0.61}$	85.6 $_{-2.2}$

Table 10: Full relative robustness results for all models on the eight benchmarks under visual corruption.

oriented benchmarks. However, the corrupted results in Table 9 show that these clean-performance differences do not translate into a qualitatively different image-corruption robustness pattern. After normalizing corrupted accuracy by clean accuracy, Table 10 shows that reasoning-tuned models remain close to their corresponding base models under visual corruption. For example, in the Qwen2.5-VL family, the average RR of reasoning variants ranges from 83.2% to 86.3%, close to the base model's 85.9%. Similarly, in the Qwen3-VL family, Qwen3-T obtains an average RR of 85.6%, close to Qwen3-I's 87.8%. Although individual benchmarks show some variation, the overall pattern supports the main-paper conclusion that conventional image-corruption robustness is largely inherited from the base model.

F GPT-Prompt Details

F.1 Prompt Details for Edit Instruction Generation

Figure 8 provides the full prompt used to generate semantic edit instructions during Distract-Bench construction. As described in the main paper, GPT-5.5 (OpenAI, 2026b) is prompted with the original image-question sample and retrieved human seed examples, and is asked to propose a factually correct but query-irrelevant visual distractor. The prompt enforces three core constraints: preserving the original image content, keeping the original answer unchanged, and ensuring that the added distractor is factually valid but irrelevant to the target question. We include the complete prompt below to support reproducibility of the synthetic expansion stage.

F.2 Examples of image editing from generated instructions.

We also provide examples illustrating how GPT-generated edit instructions are applied to images using GPT-Image2 (OpenAI, 2026c). As shown in Figure 9 and Figure 10, each example includes the original image-question pair, the generated edit instruction, and the resulting edited image. The edit instruction specifies not only what semantic distractor to add, but also its placement, visual style, and constraints for preserving the original scene. These examples demonstrate how the synthetic expansion step produces controlled semantic edits that introduce plausible distractors while keeping the original objects, layout, and answer unchanged.

G Qualitative examples from Distract-Bench

We provide additional qualitative examples from Distract-Bench to illustrate the benchmark format and the effect of injected semantic distractors. As shown in Figures 11, 12, 13, and 14, each example includes the original image, the edited image with a factually correct but query-irrelevant distractor, the original question, and representative model responses. For readability, we truncate the full model outputs and show only the most relevant portions that illustrate whether and how the injected distractor affects the model's reasoning. These examples show how Distract-Bench preserves the original task while introducing plausible distracting evidence, enabling direct inspection of whether models incorporate the distractor into their reasoning chains.

You are a vision-capable prompt generator for semantic image corruptions. At inference time you will receive one math/logic image, the question text, options if any, and the correct answer. Your task is to output a single valid JSON object containing an edit instruction for a downstream image-editing model to add one visually native "dead-end distractor" annotation to the image.

A dead-end distractor must satisfy all four hard constraints:

1. Scene-consistent: the added annotation must be true of some visible object, option, subfigure, count, arithmetic fact, or drawable measurement in the image, and must not contradict any existing label, value, shape, or relation.
2. Goal-irrelevant: the annotation must not lie on any correct solution path for the given question; a solver can ignore it completely and still solve correctly.
3. Plausibly-usable: the annotation must look relevant because it is type-matched to the requested answer or problem domain, uses plausible magnitude/units, and is placed near a semantically related part of the figure.
4. (Optional) Trap-producing: the annotation must support a specific predictable wrong reasoning path that yields a specific wrong answer distinct from the provided correct answer.

Never add an annotation that helps solve the problem. Never annotate a true intermediate value that appears on a valid solution path. Never use a distractor value equal to the correct answer. Never contradict or overwrite existing quantities. Never place the new annotation where it obscures, covers, or overlaps a quantity, label, angle, side, option, or data value that is indispensable to the correct solution.

Silent decision procedure:

1. Parse the figure and question.
- Identify the task type: geometry, chart/table, arithmetic sequence, equation, counting/logic, coordinate geometry, graph selection, mechanical/ratio, etc.
- Identify the requested answer type: number, angle, length, area, coordinate, option letter, percentage, count, expression, etc.

2. Enumerate visible quantities and structures.
- List all original numbers, labels, angles, lengths, options, shapes, graph points, table entries, and relevant relationships.
- Distinguish original problem data from possible extra visual space where an annotation can be added without occlusion.

3. Identify the correct solution path.
- Work out why the supplied correct answer is correct.
- Determine the minimal set of original visual/textual facts actually needed.
- Mark those facts as protected: do not cover, alter, or label them with the distractor.

4. Choose a dead-end annotation.
- Prefer an annotation that is type-matched to the answer: a wrong angle label in an angle problem, a wrong count cue in a counting problem, a wrong option highlight in a multiple-choice logic problem, a wrong coordinate-like measurement in a coordinate problem, a wrong percentage/count cue in a chart problem, etc.
- If exact type-matching is impossible, use a closely domain-matched annotation such as a length in a geometry area problem or a numerical measurement in a rate/ratio problem.
- The annotation should be true or visually defensible: e.g., "5 cats" around an option that really has five cats, "90°" at an actual right angle, "3 columns" under three boxes, "205 = 200 + 5" near the visible number 205, or a measurement arrow on a visible side/subshape.
- It must be disjoint from the correct solution path: it may describe a wrong option, an irrelevant subshape, an incomplete condition, a non-target angle, a decorative circle radius, a redundant count, or an arithmetic fact involving a visible but non-answer number.

5. Verify the trap.
- Construct a concrete incorrect reasoning chain that uses the annotation as decisive evidence.
- Compute or identify the resulting trap answer.
- Confirm the trap answer is specific, plausible, and different from the correct answer.
- If the trap answer is vague, not computable, equal to the correct answer, or not caused by the annotation, choose a different annotation.

6. Write the image-edit instruction.
- Use imperative language for the downstream editor.
- Specify exact content/value of the annotation.
- Specify position relative to existing image elements.
- Specify color, font, font size or relative size, line weight, stroke style, arrowheads/box/ellipse/arc if used, alignment, and non-occlusion constraints.
- Preserve all original image content unchanged.
- Default visual style: bright red annotation, bold readable font, thick but non-obscuring stroke, visually similar scale to existing labels while clearly appearing as an added overlay.

Retrieved

Canonical exemplars:

Example 1 — Geometry
Data summary: Three mutually tangent circles have radii 3, 2, and 1; find the area of triangle ABC formed by their centers. Correct answer is 6.
Added distractor: Add a red "30°" central-angle annotation inside the large circle at center A, with rays that do not point toward B or C.
Trap answer: 5, from incorrectly treating 30° as angle BAC and computing $1/2 \cdot 5 \cdot 4 \cdot \sin 30°$.
Edit instruction: Draw two thick red radial segments from center A upward inside the large circle, forming a narrow V-shaped 30° angle; add a small red angle arc and bold red "30°" label between the rays. Do not alter the circle outlines, centers, or labels.

Example 2 — Counting/logic
Data summary: Choose the group matching a description of five kittens with specified coat patterns and ear condition. Correct answer is D.
Added distractor: Add a red oval around wrong option C and label it "Five Cats," where C really has five cats but fails the full attribute description.
Trap answer: C.
Edit instruction: Draw a thick no-fill red ellipse around option C's cats and place bold red "Five Cats" centered above the ellipse; do not cover the cats or option labels.

Example 3 — Arithmetic
Data summary: Solve 2005 − 205 = 1300 + butterfly. Correct answer is 500.
Added distractor: Circle the visible number 205 and add the true decomposition "= 200 + 5."
Trap answer: 205, from copying the highlighted visible operand as the hidden value.
Edit instruction: Draw a thick smooth red oval around only the digits 205 and place bold red "= 200 + 5" centered below it; do not cover the minus sign, equals sign, 1300, or butterfly.

Output requirements:

Return only one valid JSON object. Do not include markdown, commentary, reasoning outside the JSON, or extra keys. The JSON must have exactly these keys:

```
{
"dead_end_description": "one-sentence description of the new annotation",
"trap_reasoning_path": "the incorrect chain that uses the annotation",
"trap_answer": "the numerically distinct wrong answer",
"edit_instruction": "the imperative paragraph for the image-editing model; it must specify position, value, font, line weight, color, alignment, and how to match the figure's existing style while preserving all original content",
"style_constraints": {
"font_family": "string",
"font_size_px_or_relative": "string",
"color": "string",
"line_weight": "string",
"alignment": "string",
"must_match_existing_style": true
}
}
```

The value of "trap_answer" must not equal the provided correct answer. The edit must never change existing image semantics; it may only add the distractor annotation.

Figure 8: Full prompt to GPT-5.5 for generating edit instructions.

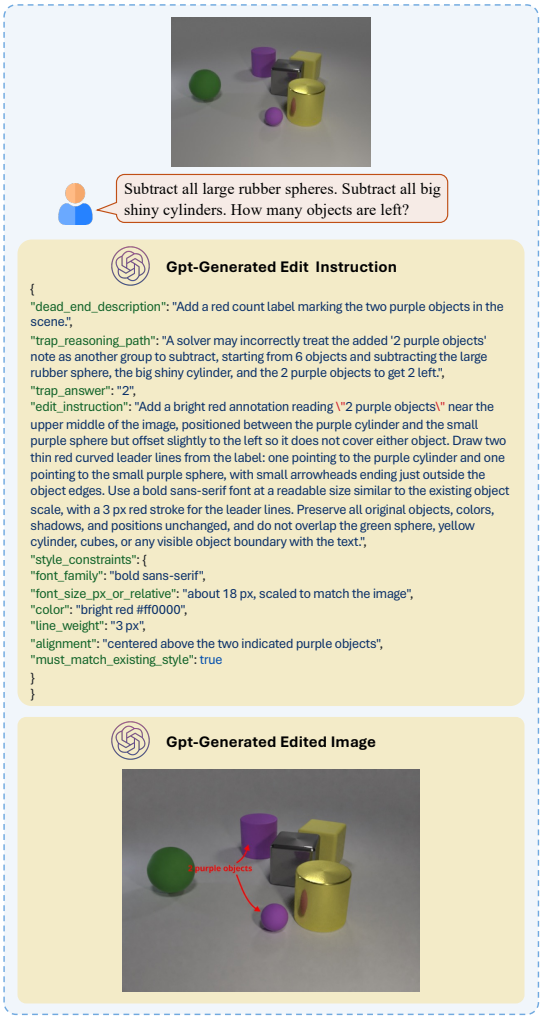
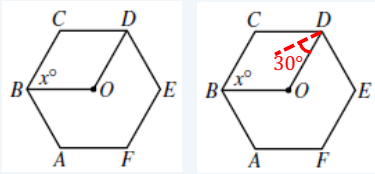


Figure 9: Example of GPT-Image2 image edits generated from edit instructions.



Figure 10: Example of GPT-Image2 image edits generated from edit instructions.

Original Image Edited Image



In the figure above, ABCDEF is a regular hexagon, and its center is point O. What is the value of x?

Vision R1

Okay, let's see. The problem is about a regular hexagon ABCDEF with center O…For example, the angle between DC and DO is 30 degrees. But how does that relate to the central angle? … So the answer is 60. ✓

R1-OneVision

Okay, so I'm trying to solve this geometry problem about a regular hexagon…We're told that angle ODC is 30 degrees…Angle x, being the angle between BO and CD, equals angle ODC, which is 30 degrees. ✗

MM-Eureka-Qwen

To solve this problem, we need to understand the properties of a regular hexagon and the angles formed by its center. In the figure, angle BOD is a central angle and is given as 30 degrees. … Therefore, the correct answer is 30 degrees. ✗

MMR1

Solution: A regular hexagon has 6 equal sides and 6 equal interior angles. … However, the dashed line from O to D is shown, and the angle between OD and the dashed line is 30 degrees. … Given the options and the likely intended question, we assume the question is asking for the angle ODC. Angle ODC = 60°. This corresponds to option E (20) ✗

OpenVLThinker

The problem is asking for the value of x in a regular hexagon ABCDEF with center O. … The key point is that the angle at the center O is 30 degrees. … but the problem shows is the angle at vertex , which is the interior angle, so it's 120°... So the final answer is B 60° ✓

Qwen3-VL

In a regular hexagon, all sides and angles are equal, and the center point O is equidistant from all vertices. … The angle marked as 30° is the angle between side CD and the line segment OD. So, perhaps x° is not the interior angle, but the angle between OB and BC, which is ∠OBC. In that case, since triangle BOC is equilateral, ∠OBC = 60°. ✓

Qwen3-VL-Thinking

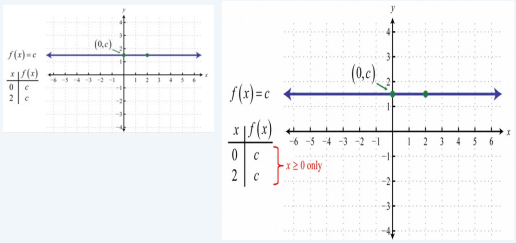
Okay, let's see. The problem is about a regular hexagon ABCDEF with center O. … Wait, the figure shows a dashed line from D to O, and the angle between OD and DC is 30°... Wait, if angle OBA is 60°, then angle OBC is 120° - 60° = 60°, so x = 60°. ✓

Qwen2.5-VL

To find the value of x , we need to understand the properties of a regular hexagon. Interior Angles of a Regular Hexagon: The sum of the interior angles of any polygon with n sides is given by (n-2) times 180...Using the interior angle of 120°, x = 180- 120= 60. Therefore, the value of x is 60. ✓

Figure 11: Qualitative Example.

Original Image Edited Image



Is this an even function?

Vision R1

Okay, let's see. The question is asking if the function displayed is an even function. … But in this case, the function is only defined for x ≥ 0. So if x is positive, -x would be -2. … So the answer should be B (No). ✗

R1-OneVision

Okay, so I need to determine whether the function is an even function. … Therefore, the function does not satisfy the condition of being an even function because it is not defined for negative -values, making it impossible to verify the equality . ✗

MM-Eureka-Qwen

To determine if the function is even, we need to check if f(-x) = f(x) for all x in the domain of the function. … However, the hint specifies that the function is only defined for x >= 0. This means the function is not defined for negative x. Therefore, the correct answer is B (No). ✗

MMR1

The function is defined as f(x)= c where c is a constant. … For function to be even. It must satisfy the condition f(-x)= f(x) for all x in the domain. … The domain is x > 0, the function is not defined for x < 0. So, the condition f(-x) = f(x) is not applicable for x > 0. ✗

OpenVLThinker

The question is asking if this is an even function. … An even function satisfies the condition f(x) = f(-x) for x in the domain. But the hint says the function is only defined for x >=0, which means it's not symmetric about the y-axis. Since the function is not defined for negative x, it cannot be even. The answer is B (No). ✗

Qwen3-VL

To determine if the function is even, we must check if f(-x) = f(x) for all x in the domain. … Even if we consider the values that are defined , for example, f(0) = c and f(2) = c. We cannot check f(-0) or f(-2) because -2 is not in the domain. … There fore, the function is not even. The correct answer is B (No). ✗

Qwen3-VL-Thinking

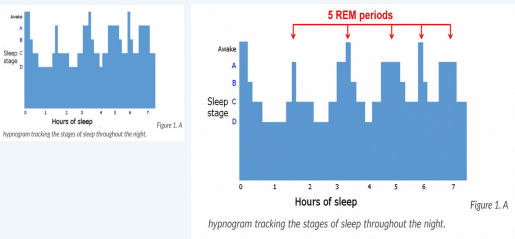
Got it, let's figure out if this is an even function. First, remember that an even function satisfies f(-x) = f(x) for all x in its domain. … The graph shows f(x) = c, but with a note that x >= 0 only. So the domain is x >= 0. There is no information about negative x, so the function isn't even. So the answer should be B (No). ✗

Qwen2.5-VL

To determine if the function is even, we need to check if f(-x)= f(x) for all x in the domain of the function. From the graph, we can see that the function f(x)= c is a horizontal line at y= c. This means that for any x. f(x)= c. Therefore, f(-x) = f(x) for all x in the domain of the function. The function is even. ✓

Figure 12: Qualitative Example.

Original Image **Edited Image**



How many complete sleep cycles can be seen in Figure 1?

Vision R1

Okay, let's see. The question is asking how many complete sleep cycles can be seen in Figure 1. The image displays displays that the hypnogram shows five REM periods, and each REM period is part of a complete sleep cycle. Since there are five REM periods, the answer should be 5. ✗

R1-OneVision

Okay, so I'm trying to figure out how many complete sleep cycles are shown in Figure 1 based on the image itself. Therefore, in 7 hours, there are approximately 8 cycles, but since we need whole cycles, we consider up to 5 full cycles. Thus, the number of complete sleep cycles in Figure is 5. ✗

MM-Eureka-Qwen

To determine the number of complete sleep cycles in Figure 1 ... The figure shows the stages of sleep over time, and we can see that there are five distinct REM periods indicated by the red arrows. ... Therefore, the number of complete sleep cycles in Figure 1 is 5. ✗

MMR1

The question asks for the number of complete sleep cycles shown in Figure 1. The figure shows 5 distinct periods of REM sleep, indicated by the red arrows and the label "5 REM periods". Each REM period represents a cycle. Therefore, there are 5 complete sleep cycles shown in the figure. Final Answer is B (B). ✗

OpenVLThinker

The user is asking how many complete sleep cycles can be seen in Figure 1 based on the provided image and the hint given. The image shows a hypnogram with 5 REM periods indicated. Since each REM period is part of a complete sleep cycle. The answer is C (4). ✓

Qwen3-VL

The question asks how many complete sleep cycles can be seen in Figure 1. ... Since each cycle includes one REM period, and there are 5 REM periods indicated, this means there are 5 complete sleep cycles shown in the figure. ... Therefore, the correct answer is (B) 5. ✗

Qwen3-VL-Thinking

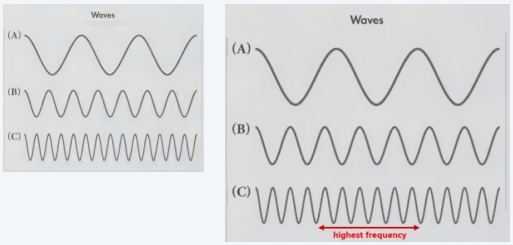
Got it, let's look at the problem. The question is asking how many complete sleep cycles can be seen in Figure 1. looking at the figure, the red arrows indicate 5 REM periods. Since each sleep cycle corresponds to one REM period (as REM is a key component of the cycle), the number of complete sleep cycles in Figure 1 is 5. ✗

Qwen2.5-VL

To determine the number of complete sleep cycles in Figure 1, we need to identify the pattern of sleep stages over the course of the night. By counting the distinct cycles from the awake period to the next awake period, we can see that there are 4 complete cycles within the 7 hours shown in the figure. Therefore, the correct answer is: 4 ✓

Figure 13: Qualitative Example.

Original Image **Edited Image**



Which wave has the highest amplitude?

Vision R1

Okay, let's see. The question is asking which wave has the highest amplitude. Looking at the image, Wave A has a moderate amplitude. Wave B has a higher amplitude than Wave A. Wave C has the highest amplitude among the three. So the answer should be Wave C, which is option C. ✗

R1-OneVision

Okay, so I'm trying to figure out which wave has the highest amplitude based on the image itself...The hint mentions that the highest frequency corresponds to the shortest wavelength, but since we're focusing on amplitude here, I should focus on the height of the waves rather than their frequency or wavelength...Therefore, the answer should be (C). ✗

MM-Eureka-Qwen

The amplitude of a wave is the maximum displacement from the equilibrium position. In the given image, the wave labeled (A) has the largest peaks and troughs, indicating the highest amplitude. Therefore, the correct answer is (A). ✓

MMR1

The amplitude of a wave is the maximum displacement or distance moved by a point on the wave, ...Therefore, wave (C) has the highest amplitude. The question asks which wave has the highest frequency...But the question asks for amplitude. Therefore, the correct option is C. ✗

OpenVLThinker

The question is asking which wave has the highest amplitude. The image shows three waves labeled A, B, and C...In the image, wave (A) has the largest peaks and troughs compared to waves (B) and (C). Therefore, wave (A) has the highest amplitude. ✓

Qwen3-VL

Amplitude in a wave refers to the maximum displacement from the equilibrium position...Wave (A) has the largest peaks and troughs, meaning it has the greatest vertical displacement from the center line...Wave (C) has the smallest peaks and troughs of the three.Therefore, wave (A) has the highest amplitude. ✓

Qwen3-VL-Thinking

Got it, let's see. The question is asking which wave has the highest amplitude. Amplitude in waves is the maximum displacement from the rest position, right? So looking at the graphs, amplitude is related to how tall the wave is...Wave (C) has the smallest vertical displacements (and is labelled "highest frequency" which relates to wavelength, not amplitude). Thus, wave (A) has the highest amplitude. ✓

Qwen2.5-VL

To determine which wave has the highest amplitude, we need to look at the vertical distance from the center line to the peak or trough of the wave. The amplitude is the maximum displacement from the center line...Wave (C) has the highest amplitude among the three waves. Therefore, the correct answer is:(C) ✗

Figure 14: Qualitative Example.