UReason: Benchmarking Reasoning-to-Generation Alignment in Unified Multimodal Models

Cheng Yang[1,*] **Chufan Shi**[2,*] **Bo Shui**[3] **Yaokang Wu**[4] **Muzi Tao**[2]
Huijuan Wang[2] **Ivan Yee Lee**[1] **Yong Liu**[2] **Xuezhe Ma**[2] **Taylor Berg-Kirkpatrick**[1]
[1]University of California San Diego [2]University of Southern California
[3]University of Illinois Urbana-Champaign [4]Carnegie Mellon University
chy085@ucsd.edu, chufansh@usc.edu

Abstract

Unified multimodal models (UMMs) aim to integrate multimodal understanding and generation within a unified architecture, yet it remains unclear to what extent textual and visual modalities are aligned. To investigate this question, we use reasoning-guided image generation as a diagnostic task, where models produce textual reasoning first and then generate images. We introduce UReason, a benchmark for evaluating reasoning-to-generation alignment in this paradigm, consisting of $2,000$ human-curated and human-verified instances spanning five reasoning-intensive tasks: CODE, ARITHMETIC, SPATIAL, ATTRIBUTE and TEXT. To enable controlled analysis, we develop an evaluation framework that compares direct generation, reasoning-guided generation and decontextualized generation, which conditions only on the refined prompt extracted from reasoning. Across eight widely used open-source UMMs, while we find that reasoning-guided generation yields improvements over direct generation, somewhat surprisingly, decontextualized generation consistently outperforms reasoning-guided generation by a large margin. Our further analyses suggest that the intended visual semantics in textual reasoning are not reliably reflected in the generated images, despite their unified design and training. Overall, UReason serves as a practical litmus test for reasoning-to-generation alignment and provides a challenging benchmark for developing next-generation, more tightly aligned UMMs.

1 Introduction

The emergence of unified multimodal models has marked a significant milestone in artificial intelligence (Team, 2024; Xie et al., 2024; Zhou et al., 2024; Deng et al., 2025; Tong et al., 2026). These models integrate multimodal understanding and generation within a single architecture, aiming to learn a unified interface across different modalities. In doing so, UMMs bridge the long-standing divide between perception-oriented Vision-Language Models (Liu et al., 2023; Team, 2025a; Guo et al., 2025b; Huang et al., 2025b) and specialized Visual Generation Models (Sauer et al., 2023; Betker et al., 2023; Esser et al., 2024; Wu et al., 2025a). However, despite operating under a unified design, it remains unclear to what extent textual and visual modalities are aligned, especially when textual reasoning is expected to guide visual synthesis.

To investigate this question, we propose to study *reasoning-guided image generation* as a testbed for diagnosing reasoning-to-generation alignment in UMMs. Reasoning-guided image generation has been adopted in recent UMMs (Deng et al., 2025; Jin et al., 2025; Qin et al., 2026; Liang et al., 2026; Zhang et al., 2026); In this paradigm, the model first produces a textual reasoning, and then generates the image conditioned on that textual reasoning. This paradigm provides a concrete setting to study whether *textual reasoning* is faithfully transferred into *visual generation* in UMMs.

To this end, we introduce UReason, a benchmark that utilizes reasoning-guided image generation as a testbed for diagnosing reasoning-to-generation alignment in UMMs (Sec. 2). UReason focuses on reasoning-centric generation tasks and contains 2,000 human-curated and human-verified instances with verifiable evaluation criteria, spanning five task categories: CODE, ARITHMETIC, SPATIAL, ATTRIBUTE, and TEXT reasoning (Fig. 1). In each instance, models must first infer an implicit visual target through multi-step reasoning, determine the visual state implied by the prompt, and then synthesize an image that faithfully realizes that state.

To enable rigorous diagnosis, we develop the UReason Evaluation Toolkit (Sec. 3). Specifically, it compares three settings: *direct generation* from the original prompt, *reasoning-guided generation* with textual reasoning for image generation, and *de-*

[*]Equal Contribution. Project Page: https://ureason.github.io

Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing, pages 57269–57309
October 24-29, 2026 ©2026 Association for Computational Linguistics



Figure 1: Representative UReason instances covering CODE, ARITHMETIC, SPATIAL, ATTRIBUTE, and TEXT reasoning. Prompts specify implicit targets that must be derived via reasoning. Detailed task and subtask descriptions are listed in Appx. B.

contextualized generation, which conditions only on the extracted refined prompt part of textual reasoning (Fig. 2). In principle, the latter two settings preserve the same visual semantics intended by the model. This design provides a controlled framework to evaluate whether the intended visual semantics encoded in textual reasoning are faithfully reflected in visual generation.

We evaluate 8 widely used UMMs on UReason (Sec. 4). Our results reveal that translating implicit targets into pixel-level outputs remains challenging: while reasoning-guided generation generally improves performance over direct generation (e.g., +11.2% for Bagel). Surprisingly, decontextualized generation consistently outperforms reasoning-guided generation by a substantial margin (e.g., +44.8% for Bagel), suggesting that intended visual semantics encoded in textual reasoning is not reliably reflected in generation process.

Our analyses demonstrate that textual reasoning is beneficial for high-level planning (Sec. 5). Specifically, UMMs can often generate reasoning that correctly specifies target visual requirements. However, these intended visual semantics are not always faithfully reflected in the generated images, despite their unified architecture and training paradigm. Through further attention and error analyses, we find that contextual interference may weaken the transfer from intended visual semantics to generated images, such as distracting tokens in intermediate results.

Overall, we position UReason as a litmus test for assessing reasoning-to-generation alignment in UMMs, specifically whether generated images reflect the intended visual semantics in textual reasoning. Our results show that, despite unified design, current UMMs still struggle to transfer reasoning into pixels, leaving room for improvement.

2 The UReason Benchmark

UReason is designed by two complementary considerations. First, unlike traditional text-to-image benchmarks (Saharia et al., 2022; Lee et al., 2023; Huang et al., 2025a) that evaluate descriptive prompts with an emphasis on aesthetic fidelity, UReason shifts the paradigm from description to deduction: the target content is not stated verbatim and must be inferred from the input scenario, requiring multi-step reasoning such as state tracking and distractor suppression. Second, UReason is designed as a broad, fine-grained, and verifiable testbed for reasoning-to-generation failures. As shown in Tab. 1, UReason contains 2,000 human-curated and human-verified instances spanning 5 tasks: CODE, ARITHMETIC, SPATIAL, ATTRIBUTE, and TEXT reasoning (Fig. 1), with 30 fine-grained subcategories (Fig. 4). Compared with prior benchmarks, UReason provides broader task coverage, larger scale, and a finer-grained taxonomy, including the under-explored CODE domain.

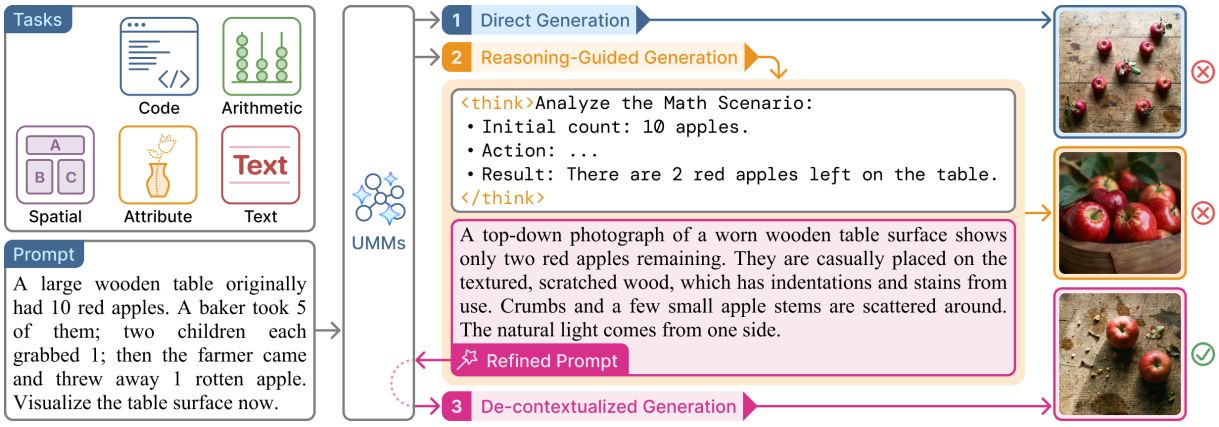


Figure 2: Overview of UReason evaluation framework. UReason compares 3 settings: 1 Direct Generation, 2 Reasoning-Guided Generation and 3 De-contextualized Generation.

These dataset properties, together with instance-level criteria, support our evaluation protocol and enable controlled diagnosis of failures (detailed in Appx. B).

2.1 Task

We now introduce the 5 tasks of UReason with representative instances presented in Fig. 1.

Code Reasoning. The Code Reasoning task evaluates whether UMMs can act as neural visual interpreters, mapping abstract code to concrete visual renderings. Given code snippets across eight languages (e.g., Python), the model must reason over execution logic and synthesize the corresponding visual outcome. Beyond syntactic recognition, this requires simulating program execution to determine the final visual state and converting it into a language description that guides image generation.

Arithmetic Reasoning. Arithmetic Reasoning evaluates UMMs' ability to perform arithmetic operations within sequential narratives across seven subcategories. Inputs describe quantity changes through events such as addition, and the model must generate a scene whose visible count matches the computed final state. This task requires models to move beyond keyword matching by explicitly deriving the final quantity through calculation and using it to constrain visual generation.

Spatial Reasoning. Spatial Reasoning assesses whether UMMs can interpret complex instructions and form structured visual arrangements across six subcategories. Unlike standard benchmarks with explicit spatial relations, our prompts include implicit cues such as swap operations and logical constraints. Models must resolve these high-level descriptions into coherent coordinate-based layouts before rendering, testing inter-object spatial reasoning beyond surface prompt alignment.

Attribute Reasoning. Attribute Reasoning evaluates whether UMMs can track and update object attributes under explicitly described state transitions and logical modifications (e.g., a hat being removed). The model must generate a scene where objects strictly exhibit the final attributes implied by the prompt. This task requires logical filtering: models must infer the terminal outcome rather than rendering intermediate states, suppressing the tendency to visualize irrelevant attributes.

Text Reasoning. Text Reasoning evaluates the model's ability to perform context-aware text rendering. The target text is not explicitly quoted, but must be inferred from contextual rules, such as identifying the "second" and "fourth" letters of a word. The key challenge lies in symbolic reasoning: the model must derive the correct answer from the context while suppressing irrelevant information, ensuring that only the final result is rendered.

2.2 Data Curation

UReason adopts a two-stage curation pipeline to ensure that each instance requires multi-step reasoning to determine a specific visual output. We begin with human-curated seed instances and then expand coverage through controlled LLM-assisted augmentation with human verification. Additional details about data annotation are provided in Appx. B.

Human-Curated Seed Data Construction. To ensure the quality and diversity of UReason, human

Benchmark	Problem Setting	Size	Cat./ Sub.	Task Category					Evaluation Setting	Metric Aspect	
				Code	Arith.	Spatial	Attr.	Text		Perf.	Diag.
Commonsense-T2I (Fu et al., 2024)	Text-to-Image	150	5/5	✗	✗	✓	✓	✗	1	✓	✗
WISE (Niu et al., 2025b)	Text-to-Image	1,000	3/25	✗	✗	✓	✓	✗	1	✓	✗
R2I-Bench (Chen et al., 2025a)	Text-to-Image	3,068	7/32	✗	✓	✓	✓	✓	1	✓	✗
OneIG-Bench (Chang et al., 2025)	Text-to-Image	2,440	6/26	✗	✓	✓	✓	✓	1 †	✓	✗
T2I-ReasonBench (Sun et al., 2025)	Text-to-Image	800	4/35	✗	✗	✗	✓	✓	1 †	✓	✗
GIR-Bench-T2I (Li et al., 2026b)	Text-to-Image	300	4/4	✗	✓	✓	✗	✓	1 †	✓	✗
KRIS-Bench (Wu et al., 2025b)	Image Editing	1,267	7/22	✗	✓	✓	✓	✓	1 †	✓	✗
RISEBench (Zhao et al., 2025)	Image Editing	360	4/16	✗	✓	✓	✓	✗	1	✓	✗
ROVER-IG (Liang et al., 2026)	Image Editing	908	4/17	✗	✓	✓	✓	✗	1 2	✓	✗
UReason (Ours)	Text-to-Image	2,000	5/30	✓	✓	✓	✓	✓	1 2 3	✓	✓

Table 1: Comparison of UReason with existing reasoning-driven image generation benchmarks. "Cat./Sub.": number of categories/subcategories. Evaluation settings: 1 Direct, 2 Reasoning-Guided, 3 De-contextualized. "Perf.": performance evaluation; "Diag.": controlled diagnostic comparison. "†" denotes benchmarks that run preliminary reasoning-chain experiments on specific models (e.g., Bagel) but primarily evaluate Direct Generation.

experts establish a fine-grained taxonomy over the 5 primary tasks, yielding 30 subcategories. These subcategories cover distinct reasoning and visual perspectives. Based on this hierarchy, experts manually construct seed instances, each consisting of a reasoning-intensive prompt and an evaluation criterion specifying the expected visual outcome. All seeds undergo strict validation, resulting in a foundational dataset of 500 high-quality seed instances.

LLM-Assisted Data Augmentation. After constructing the seed dataset, we scale UReason with a human-guided augmentation pipeline powered by Gemini-3-Pro (DeepMind, 2025). Annotators systematically vary key factors of each seed instance, including the number of reasoning steps, target visual entities, and narrative context, to generate diverse yet logically consistent variants. All candidates undergo multi-round human-LLM refinement and verification to ensure quality and correctness. This process expands UReason to 2,000 test instances, providing broad coverage of realistic and challenging reasoning scenarios.

Dataset Split. The full *test* set contains all 2,000 instances in UReason, while *testmini* is a 500-instance subset intended for rapid validation during model development. As reported in Appx. F, comparative experiments show consistent trends across the two evaluation sets. Unless otherwise stated, we report *testmini* results in the following sections.

3 Evaluation Framework

3.1 Evaluation Settings

UMMs can understand and generate multimodal information, typically achieved through a unified architecture to enable end-to-end training and inference. For image generation tasks, recent UMMs support reasoning-guided image generation, in which textual reasoning is produced before visual synthesis. As a result, image generation in UMMs differs from traditional text-to-image models in both architectural design and training paradigm.

As shown in Tab. 1, existing reasoning-driven image generation benchmarks are primarily evaluated under Direct Generation. To more systematically diagnose reasoning-to-generation alignment, we evaluate UMMs under the following three settings, including Reasoning-Guided and De-contextualized Generation.

Setting 1: Direct Generation. As illustrated in Fig. 2 (①), this setting evaluates a model's ability to generate an image directly from the original prompt without explicitly producing textual reasoning. Given a prompt P and a UMM M, the generated image I is:

$$I = M(P). \tag{1}$$

This setting provides a baseline by quantifying performance without additional textual reasoning.

Setting 2: Reasoning-Guided Generation. As illustrated in Fig. 2 (②), the model generates a reasoning trace R followed by the output image I, conditioned on the prompt P. Crucially, both R and I are produced within the same model and context window. Formally:

$$[R, I] = M(P). \tag{2}$$

This setting follows the chain-of-thought style adaptation for visual generation and measures the

net effect of textual reasoning (Deng et al., 2025). The reasoning trace is defined as $R = [R_t, R_p]$, where R_t denotes intermediate thoughts and R_p denotes a refined prompt that explicitly summarizes the intended visual specification.

Setting 3: De-contextualized Generation. As shown in Fig. 2 (③), after producing the reasoning trace $R = [R_t, R_p]$ in Setting 2, we discard the original prompt P and the intermediate thoughts R_t, and generate the image conditioned only on the refined prompt R_p:

$$I = M(R_p). \tag{3}$$

As a result, this setting serves as a controlled comparison to Setting 2: since the refined prompt R_p is extracted from R in Setting 2, both settings encode the same model-produced visual semantics, but in different contextual formats. In principle, if textual reasoning and visual generation are aligned, Setting 2 and 3 should lead to comparable performance.

3.2 Evaluation Metric

Each test instance in UReason specifies an instance-specific, verifiable ground-truth criterion, enabling objective and scalable evaluation across all tasks. We therefore report two complementary metrics. *Visual Verification Accuracy* measures whether a generated image satisfies the ground-truth criterion. *Performance Gain* measures accuracy differences between settings under our evaluation protocol.

Visual Verification Accuracy. For each test instance, UReason provides an instance-specific ground-truth criterion C that defines the expected visual outcome under correct reasoning. Given a generated image I, we define an indicator function $\mathbb{I}(I, C)$ that returns 1 if I satisfies C, and 0 otherwise. The overall visual verification accuracy over a dataset of N instances is:

$$\text{Accuracy} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(I_i, C_i). \tag{4}$$

For example, in ARITHMETIC reasoning, C specifies exact object count (Fig. 1). To implement $\mathbb{I}(\cdot, \cdot)$ at scale, we employ Qwen3-VL-235B-A22B (Team, 2025a) as our automated evaluator (Zhao et al., 2025; Niu et al., 2025b; Chen et al., 2025a; Wu et al., 2025b). We use prompts that ask the evaluator to perform focused verification against C. The template is provided in Appx. M, with human validation detailed in Appx. I.2.

Performance Gain. To further compare and analyze the contribution of different contextual information for the reasoning-driven image generation, we measure the performance improvement between consecutive settings. Let i and j denote two settings in our evaluation protocol with $i < j$. We define the performance gain from i to j as:

$$\Delta_{i \to j} = \text{Accuracy}_j - \text{Accuracy}_i \tag{5}$$

We focus on two transitions: $\Delta_{1 \to 2}$ quantifies the benefit of incorporating reasoning for image generation, while $\Delta_{2 \to 3}$ measures the gap between Setting 2 and Setting 3, which are designed to preserve the same intended visual semantics.

4 Experiments

4.1 Models

We benchmark 8 widely utilized open-source UMMs trained to perform reasoning-guided image generation: Bagel (Deng et al., 2025), SRUM (Jin et al., 2025), UniCoT, UniCoT-v2 (Qin et al., 2026), ThinkMorph (Gu et al., 2026), Bagel-Zebra-CoT (Li et al., 2026a), UniMoE2 (Li et al., 2025a) and T2I-R1 (Jiang et al., 2025). We do not evaluate closed-source systems such as Nano Banana Pro (Google, 2026), since it is unclear whether they are end-to-end UMMs, making our evaluation protocol difficult to apply (detailed in Appx. J).

4.2 Main Results

Tab. 2 reports the performance of all evaluated models under our 3 diagnostic settings.

Direct Prompting Performs Poorly on Implicit Targets. Direct Generation yields low accuracy across models and tasks, with overall performance ranging from 4.2% to 8.2%. This shows that direct text-to-image mapping is fundamentally insufficient to solve UReason, as the target visual content is intentionally implicit and must be derived through multi-step reasoning before visual generation.

Chain-of-Thought Reasoning Enhances Generation Capabilities. Comparing Direct Generation with Reasoning-Guided Generation, introducing explicit chain-of-thought reasoning improves overall performance for all eight models, with gains ranging from +0.4% (T2I-R1) to +19.2% (UniCoT-v2). These results indicate that CoT can effectively elicit reasoning behaviors that benefit unified multimodal generation. A concrete illustration of this

Model	Setting	Code		Arithmetic		Spatial		Attribute		Text		Overall	
		Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ
Bagel	1	10.0	-	5.0	-	3.0	-	6.0	-	9.0	-	6.6	-
	2	30.0	+20.0	14.0	+9.0	15.0	+12.0	19.0	+13.0	11.0	+2.0	17.8	+11.2
	3	**58.0**	**+28.0**	**51.0**	**+37.0**	**58.0**	**+43.0**	**60.0**	**+41.0**	**86.0**	**+75.0**	**62.6**	**+44.8**
UniCoT-v2	1	9.0	-	2.0	-	2.0	-	5.0	-	3.0	-	4.2	-
	2	27.0	+18.0	19.0	+17.0	29.0	+27.0	21.0	+16.0	21.0	+18.0	23.4	+19.2
	3	**65.0**	**+38.0**	**45.0**	**+26.0**	**46.0**	**+17.0**	**67.0**	**+46.0**	**87.0**	**+66.0**	**62.0**	**+38.6**
SRUM	1	11.0	-	4.0	-	3.0	-	6.0	-	7.0	-	6.2	-
	2	25.0	+14.0	8.0	+4.0	20.0	+17.0	24.0	+18.0	6.0	-1.0	16.6	+10.4
	3	**67.0**	**+42.0**	**50.0**	**+42.0**	**50.0**	**+30.0**	**50.0**	**+26.0**	**82.0**	**+76.0**	**59.8**	**+43.2**
Bagel-Zebra -CoT	1	7.0	-	7.0	-	2.0	-	10.0	-	5.0	-	6.2	-
	2	14.0	+7.0	10.0	+3.0	15.0	+13.0	23.0	+13.0	10.0	+5.0	14.4	+8.2
	3	**50.0**	**+36.0**	**43.0**	**+33.0**	**33.0**	**+18.0**	**48.0**	**+25.0**	**85.0**	**+75.0**	**51.8**	**+37.4**
ThinkMorph	1	9.0	-	1.0	-	4.0	-	3.0	-	10.0	-	5.4	-
	2	19.0	+10.0	12.0	+11.0	15.0	+11.0	26.0	+23.0	5.0	-5.0	15.4	+10.0
	3	**49.0**	**+30.0**	**37.0**	**+25.0**	**45.0**	**+30.0**	**55.0**	**+29.0**	**71.0**	**+66.0**	**51.4**	**+36.0**
UniCoT	1	12.0	-	3.0	-	6.0	-	12.0	-	8.0	-	8.2	-
	2	33.0	+21.0	18.0	+15.0	26.0	+20.0	21.0	+9.0	12.0	+4.0	22.0	+13.8
	3	**57.0**	**+24.0**	**42.0**	**+24.0**	**50.0**	**+24.0**	**42.0**	**+21.0**	**52.0**	**+40.0**	**48.6**	**+26.6**
T2I-R1	1	3.0	-	6.0	-	4.0	-	9.0	-	2.0	-	4.8	-
	2	6.0	+3.0	4.0	-2.0	2.0	-2.0	11.0	+2.0	3.0	+1.0	5.2	+0.4
	3	**20.0**	**+14.0**	**15.0**	**+11.0**	**12.0**	**+10.0**	**27.0**	**+16.0**	**47.0**	**+44.0**	**24.2**	**+19.0**
UniMoE2	1	5.0	-	4.0	-	2.0	-	10.0	-	4.0	-	5.0	-
	2	10.0	+5.0	3.0	-1.0	3.0	+1.0	12.0	+2.0	6.0	+2.0	6.8	+1.8
	3	**17.0**	**+7.0**	**13.0**	**+10.0**	**8.0**	**+5.0**	**21.0**	**+9.0**	**13.0**	**+7.0**	**14.4**	**+7.6**

Table 2: Model performance across three evaluation settings on UReason. Acc and Δ denote visual verification accuracy (%) and performance gain over the previous setting, respectively. 1, 2, and 3 represent Direct Generation, Reasoning-Guided Generation and De-contextualized Generation, respectively.

benefit emerges in the CODE task. In Direct Generation, models frequently fail to map raw syntax, such as HTML tags, into coherent visual layouts. In contrast, Reasoning-Guided Generation enables models to first translate the code into a description of the intended rendering, which provides a more interpretable conditioning signal for subsequent image synthesis. For example, Bagel improves from 10.0% to 30.0% on CODE, and UniCoT improves from 12.0% to 33.0%.

De-contextualized Generation Consistently Outperforms Reasoning-Guided Generation. While Reasoning-Guided Generation improves performance over Direct Generation, De-contextualized Generation yields an even larger gain. As shown in Tab. 2, accuracy increases from Reasoning-Guided Generation to De-contextualized Generation for every model, reaching +44.8% for Bagel, +43.2% for SRUM and +26.6% for UniCoT. This result is notable because the two settings preserve the same intended visual semantics. In principle, if textual reasoning is reflected in visual generation, these two settings should yield comparable performance. The consistent gap suggests that, despite a unified architecture and training in reasoning-guided image generation, current UMMs still exhibit a reasoning-to-generation alignment gap between textual reasoning and visual generation. In Sec. 5, we further examine possible factors associated with this gap, including reasoning quality, context length, error and attention analysis.

Full-Test Results Confirm the Main Trend. We additionally evaluate all eight models on the full test set. De-contextualized Generation outperforms Reasoning-Guided Generation for every model, with gains ranging from 8.6 to 44.4 percentage points. These gains are consistent in direction and magnitude with those observed on testmini, confirming that the main trend is not specific to the selected subset. Full per-task results for both evaluation sets are reported in Appx. F.

Model	Code	Arith.	Spatial	Attr.	Text	Overall
Bagel	93.0	94.0	88.0	96.0	96.0	93.4
SRUM	91.0	91.0	88.0	93.0	95.0	91.6
UniCoT	84.0	70.0	84.0	99.0	95.0	86.4
ThinkMorph	83.0	82.0	85.0	96.0	91.0	87.4
Bagel-Zebra-CoT	75.0	89.0	79.0	94.0	92.0	85.8

Table 3: Reasoning chain quality evaluation. We report the accuracy (%) of generated reasoning chains against ground-truth criteria across tasks.

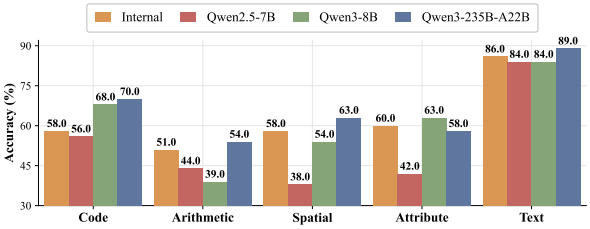


Figure 3: Comparison of internal (Bagel) and external (Qwen) prompt models across 5 tasks.

5 Discussion

Evaluating the Quality of Reasoning Chains. To localize the performance bottleneck, we ask whether failures originate from incorrect reasoning in intended visual semantics. We evaluate correctness of reasoning-chain R against ground-truth criteria using Qwen3-235B-A22B as an LLM-as-judge (see Appx. M for details). As shown in Tab. 3, models achieve consistently high reasoning accuracy, with Bagel reaching 93.4% overall. This suggests that UMMs can often infer the ground-truth visual target and produce coherent specifications. Therefore, the dominant challenge lies in faithfully realizing these specifications in pixels.

UMMs as Intrinsic Prompt Models. Modern online T2I systems employ an external prompt model to rewrite instructions before image synthesis (e.g., Qwen-Image (Team, 2025b)), highlighting prompt optimization as a practical component of T2I pipelines. A key advantage of UMMs is that they can perform this refinement end-to-end, without an external LLM. To quantify this capability, we compare Bagel's self-generated refined prompts with external prompt models based on Qwen2.5-7B, Qwen3-8B, and Qwen3-235B-A22B. As shown in Fig. 3, Bagel outperforms its LLM backbone Qwen2.5-7B and achieves performance comparable to stronger prompt models, including Qwen3-8B and Qwen3-235B-A22B. These results suggest that UMMs are native self-prompt models, offering a promising end-to-end alternative to the two-stage prompt-rewrite-then-generate pipeline.

Model	#R_p	Code	Arith.	Spatial	Attr.	Text	Overall
Bagel	1×	58.0	51.0	58.0	60.0	86.0	62.6
	4×	63.0	49.0	52.0	53.0	85.0	60.4
	8×	58.0	46.0	48.0	47.0	84.0	56.6
ThinkMorph	1×	49.0	37.0	45.0	55.0	71.0	51.4
	4×	46.0	34.0	37.0	55.0	73.0	49.0
	8×	44.0	33.0	34.0	51.0	73.0	47.0
Bagel-Zebra-CoT	1×	50.0	43.0	33.0	48.0	85.0	51.8
	4×	53.0	38.0	30.0	42.0	77.0	48.0
	8×	49.0	36.0	25.0	38.0	74.0	44.4

Table 4: Length-controlled ablation. 1×/4×/8× denotes the refined prompt R_p repeated once, four, or eight times, where 4× approximates the average token length of a full reasoning trace in our evaluation.

Ablating the Effect of Context Length. To determine whether the observed performance drop is due to longer contextual sequences alone, we conduct a length-controlled ablation. In this setup, the refined prompt R_p is repeated 4× or 8× without introducing any new semantic content. As shown in Tab. 4, artificially lengthening the context causes a relatively minor overall performance decline (at most -6.0% for Bagel at 8×), which is drastically smaller than the -44.8% gap observed between Reasoning-Guided Generation and Decontextualized Generation. Moreover, the evaluated UMMs are explicitly trained for reasoning-guided image generation. We further examine the training length distributions of representative models (Appx. G) and verify that the UReason evaluation sequences fall well within their typical training ranges, indicating no out-of-distribution length. We further conduct controlled experiments examining refined-prompt quality, prompt format, delimiters, testmini sampling, and the effect of preceding reasoning content with R_p held fixed. These results show that the examined factors cannot fully explain the gap and that preceding reasoning content still degrades performance when R_p is fixed; full results are provided in Appx. E.

Experiments on Alternative Evaluators. To further assess the robustness of our evaluation results, we employ two additional closed-source models, Gemini-2.5-Pro and Gemini-3.1-Pro, as alternative automated evaluators. We use these evaluators to assess UniCoT under all three settings. As shown in Table 5, minor absolute differences exist, possibly reflecting differences in the evaluators' baseline VQA capabilities (human agree-

Evaluator	Setting	Code		Arithmetic		Spatial		Attribute		Text		Overall	
		Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ
Qwen3-VL -235B-A22B	1	12.0	-	3.0	-	6.0	-	12.0	-	8.0	-	8.2	-
	2	33.0	+21.0	18.0	+15.0	26.0	+20.0	21.0	+9.0	12.0	+4.0	22.0	+13.8
	3	**57.0**	**+24.0**	**42.0**	**+24.0**	**50.0**	**+24.0**	**42.0**	**+21.0**	**52.0**	**+40.0**	**48.6**	**+26.6**
Gemini-2.5 -Pro	1	10.0	-	4.0	-	7.0	-	10.0	-	9.0	-	8.0	-
	2	30.0	+20.0	18.0	+14.0	23.0	+16.0	17.0	+7.0	9.0	0.0	19.4	+11.4
	3	**53.0**	**+23.0**	**40.0**	**+22.0**	**47.0**	**+24.0**	**43.0**	**+26.0**	**48.0**	**+39.0**	**46.2**	**+26.8**
Gemini-3.1 -Pro	1	10.0	-	3.0	-	6.0	-	10.0	-	8.0	-	7.4	-
	2	31.0	+21.0	16.0	+13.0	23.0	+17.0	15.0	+5.0	10.0	+2.0	19.0	+11.6
	3	**52.0**	**+21.0**	**41.0**	**+25.0**	**48.0**	**+25.0**	**40.0**	**+25.0**	**45.0**	**+35.0**	**45.2**	**+26.2**

Table 5: Alternative evaluator results on the UniCoT model. Acc and Δ denote visual verification accuracy (%) and performance gain over the previous setting, respectively. 1 , 2 , and 3 represent Direct Generation, Reasoning-Guided Generation and De-contextualized Generation, respectively.

Prompt	Code	Arith.	Spatial	Attr.	Text	Overall
Model R_p	58.0	51.0	58.0	60.0	86.0	62.6
Gold R_p^*	**72.0**	**58.0**	**66.0**	**61.0**	**91.0**	**69.6**

Table 6: Bagel performance (%) with its model-generated refined prompt R_p and the human-verified, answer-grounded gold refined prompt R_p^*.

Layers	P	R_t	R_p
1–7	3.98	5.50	9.79
8–14	3.11	4.22	8.24
15–21	0.20	0.30	0.79
22–28	0.12	0.21	0.58
1–28	**1.85**	**2.56**	**4.85**

Table 7: Average attention weights across layers for Bagel during image generation, scaled by 10^{-4}.

ment: 0.941 and 0.950 for Gemini-2.5-Pro and Gemini-3.1-Pro, respectively, compared with 0.924 for Qwen3-VL-235B-A22B); nevertheless, the relative performance trends across all three settings remain highly consistent. Gemini-2.5-Pro and Gemini-3.1-Pro replicate the experimental findings, showing substantial performance gains in De-contextualized Generation over Reasoning-Guided Generation. These consistent trends suggest that the observed reasoning-to-generation gap is not an artifact of a particular evaluator.

Gold Refined-Prompt Baseline. To separate generation fidelity from errors in a model's self-generated R_p, we construct an answer-grounded gold refined prompt R_p^* for each instance. Gemini-3.1-Pro receives the task and human-annotated ground-truth criterion and produces a self-contained visual description containing every required element and only the final state. Each description is human-verified against the criterion before use. As shown in Tab. 6, Bagel improves by 7.0 points over its own R_p, from 62.6% to 69.6%. This moderate gain is consistent with Bagel's high reasoning-chain accuracy and leaves substantial headroom in the generation stage, where the model must faithfully translate the intended visual semantics specified in the refined prompt into pixel-level realization.

Attention Analysis. To better understand why the same intended visual semantics can lead to different generation outcomes, we conduct an attention analysis on Bagel. During image generation, we extract average attention weights assigned to three token groups: P, R_t, and R_p. As shown in Table 7, the model allocates substantial attention to P and R_t, particularly in early to middle layers. Across layers, the attention paid to the intermediate reasoning trace R_t remains more than half of that paid to R_p. This suggests that image generation is highly sensitive to contextual interference, which may compete with the final visual specification and weaken the transfer from intended visual semantics to pixels. This suggests that contextual interference may partly contribute to the observed reasoning-to-generation alignment gap in UMMs. Appx. H provides further details, including analogous analyses on ThinkMorph and T2I-R1, which reveal similar attention patterns and further support the presence of contextual interference across different UMMs. We hope these findings motivate future work that better preserves intended visual semantics during reasoning-to-generation transfer.

Error Analysis. To understand failure modes, error cases from Bagel under Setting 2 are analyzed, identifying four error types: (1) *Reasoning Errors* (5.8%; see Fig. 14): the model produces an incorrect reasoning chain or derives an incorrect final state, such as miscalculating object counts; (2) *Instruction Misinterpretation* (10.6%; see Fig. 15): the model misunderstands the fundamental intent of the instruction, such as rendering code text itself rather than visualizing the object specified by the code; (3) *Concept Hallucination* (8.2%; see Fig. 16): the generated image introduces objects or visual concepts that are not specified in the input prompt; and (4) *Task-Specific Errors* (75.4%; see Figs. 17–21): the model correctly reasons about the intended visual outcome but fails to faithfully realize specific task requirements in the generated image, such as object counts, spatial arrangements, attributes, or rendered text. The dominance of task-specific errors highlights the reasoning-to-generation alignment gap: even when models derive the correct visual intent in textual reasoning, they often fail to faithfully realize it in the generated image. This result demonstrates UReason's effectiveness as a diagnostic benchmark for fine-grained failure analysis. Representative error cases are provided in Appx. N.

6 Related Work

Unified Multimodal Models. Unified multimodal models aim to support multimodal understanding and generation within a single model, typically by mapping text and images into a shared representational interface. Recent approaches span diffusion-based models (Li et al., 2025c; Swerdlow et al., 2025; Shi et al., 2025), autoregressive models (Team, 2024; Wang et al., 2024; Wu et al., 2025a; Chen et al., 2025c; Tong et al., 2025), and hybrids that combine both mechanisms (Zhou et al., 2024; Xie et al., 2024; Deng et al., 2025). Despite these advances, characterizing cross-modal interactions and the interplay between understanding and generation remains an active research area (Yan et al., 2026; Liang et al., 2026; Niu et al., 2025a; Zhang et al., 2025). UReason complements this line of work by providing a diagnostic evaluation of reasoning-to-generation alignment, specifically how textual reasoning influences visual generation.

Chain-of-Thought. Chain-of-thought reasoning has emerged as a powerful technique to enhance the capabilities of LLMs (Wei et al., 2022; Chen et al., 2025b) and MLLMs (Li et al., 2025b; Luo et al., 2025). Recent large reasoning models (OpenAI, 2024, 2025; Guo et al., 2025a) further demonstrate that test-time scaling enables more accurate outcomes. UMMs integrate processing for language and image within a single architecture, which provides a natural foundation for reasoning-guided image generation. Consequently, recent works (Jiang et al., 2025; Deng et al., 2025; Jin et al., 2025; Yang et al., 2025; Qin et al., 2026; Liang et al., 2026; Gao et al., 2026) have adopted explicit reasoning chains to plan via natural language before synthesizing images. Despite the appeal of this reasoning-guided paradigm, the actual alignment between reasoning and visual generation quality remains underexplored, motivating our investigation of reasoning-to-generation alignment in this area.

T2I Benchmarks. T2I benchmarks have progressed from evaluating explicit prompt adherence to probing implicit reasoning capabilities. Prior work focuses on generation quality via text-image alignment (Saharia et al., 2022; Ghosh et al., 2023; Lee et al., 2023), compositional generation (Huang et al., 2023), and safety constraints (Schramowski et al., 2023; Seshadri et al., 2024). More recent benchmarks target reasoning-driven scenarios that require commonsense and world knowledge (Fu et al., 2024; Niu et al., 2025b; Chen et al., 2025a; Sun et al., 2025; Chang et al., 2025). UReason complements this line of work with probing cross-modality alignment between textual reasoning and visual generation in UMMs.

7 Conclusion

We introduce UReason, a diagnostic benchmark for reasoning-guided image generation in UMMs, with 5 tasks, 2,000 instances and a controlled framework comparing direct, reasoning-guided, and decontextualized generation. Across 8 open-source models, we observe that reasoning improves over direct prompting, the intended visual semantics expressed in textual reasoning are not always faithfully reflected in the generated images, and conditioning only on the refined prompt often performs best, showing the reasoning-to-generation alignment gap. These findings suggest that advancing UMMs requires not only stronger reasoning capabilities, but also more robust mechanisms to ensure that inferred visual semantics can be reliably carried through the image generation process.

Limitations

UReason focuses on single-turn reasoning-guided image generation, which allows us to cleanly attribute each outcome to a specific reasoning-to-generation (i.e., from textual reasoning to visual generation) process. We do not explicitly evaluate image editing tasks. Nevertheless, we expect similar reasoning-to-generation tensions to arise in editing pipelines that condition on verbose reasoning or edit histories. Extending UReason and our diagnostic protocol to multi-turn interactions remains an important direction, as interference effects may become more pronounced when reasoning traces accumulate over turns.

Acknowledgements

This research was supported in part by the NSF under grants 2146151 and 2422214. We thank the reviewers, Area Chairs, Senior Area Chairs, and Program Chairs for their constructive feedback and suggestions during the review process.

References

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, and 1 others. 2023. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8.

Zongsheng Cao, Yangfan He, Anran Liu, Jun Xie, Feng Chen, and Zepeng Wang. 2025. Purifygen: A risk-discrimination and semantic-purification model for safe text-to-image generation. In *Proceedings of the 33rd ACM International Conference on Multimedia*.

Jingjing Chang, Yixiao Fang, Peng Xing, Shuhan Wu, Wei Cheng, Rui Wang, Xianfang Zeng, Gang YU, and Hai-Bao Chen. 2025. OneIG-bench: Omni-dimensional nuanced evaluation for image generation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Kaijie Chen, Zihao Lin, Zhiyang Xu, Ying Shen, Yuguang Yao, Joy Rimchala, Jiaxin Zhang, and Lifu Huang. 2025a. R2i-bench: Benchmarking reasoning-driven text-to-image generation. *arXiv preprint arXiv:2505.23493*.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025b. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*.

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. 2025c. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*.

DeepMind. 2025. Introducing gemini 3. https://blog.google/products/gemini/gemini-3-collection/. Accessed: 2025-12-29.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, and 1 others. 2025. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, and 1 others. 2024. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*.

Xingyu Fu, Muyu He, Yujie Lu, William Yang Wang, and Dan Roth. 2024. Commonsense-t2i challenge: Can text-to-image generation models understand commonsense? *arXiv preprint arXiv:2406.07546*.

Xin Gao, Cheng Yang, Chufan Shi, and Taylor Berg-Kirkpatrick. 2026. Do text edits generalize to visual generation? benchmarking cross-modal knowledge editing in umms. *arXiv preprint arXiv:2606.00477*.

Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. 2023. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152.

Google. 2026. Nano banana 2 - gemini ai image generator and photo editor. https://gemini.google/overview/image-generation/.

Jiawei Gu, Yunzhuo Hao, Huichen Will Wang, Linjie Li, Michael Qizhe Shieh, Yejin Choi, Ranjay Krishna, and Yu Cheng. 2026. Thinkmorph: Emergent properties in multimodal interleaved chain-of-thought reasoning. In *The Fourteenth International Conference on Learning Representations*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, and 1 others. 2025b. Seed1. 5-vl technical report. *arXiv preprint arXiv:2505.07062*.

Kaiyi Huang, Chengqi Duan, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. 2025a. T2i-compbench++: An enhanced and comprehensive

benchmark for compositional text-to-image generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. 2023. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *Advances in Neural Information Processing Systems*, 36:78723–78747.

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

Dongzhi Jiang, Ziyu Guo, Renrui Zhang, Zhuofan Zong, Hao Li, Le Zhuo, Shilin Yan, Pheng-Ann Heng, and Hongsheng Li. 2025. T2i-r1: Reinforcing image generation with collaborative semantic-level and token-level cot. *arXiv preprint arXiv:2505.00703*.

Weiyang Jin, Yuwei Niu, Jiaqi Liao, Chengqi Duan, Aoxue Li, Shenghua Gao, and Xihui Liu. 2025. Srum: Fine-grained self-rewarding for unified multimodal models. *arXiv preprint arXiv:2510.12784*.

Tony Lee, Michihiro Yasunaga, Chenlin Meng, Yifan Mai, Joon Sung Park, Agrim Gupta, Yunzhi Zhang, Deepak Narayanan, Hannah Teufel, Marco Bellagente, and 1 others. 2023. Holistic evaluation of text-to-image models. *Advances in Neural Information Processing Systems*, 36:69981–70011.

Ang Li, Charles L. Wang, Deqing Fu, Kaiyu Yue, Zikui Cai, Wang Bill Zhu, Ollie Liu, Peng Guo, Willie Neiswanger, Furong Huang, Tom Goldstein, and Micah Goldblum. 2026a. Zebra-cot: A dataset for interleaved vision-language reasoning. In *The Fourteenth International Conference on Learning Representations*.

Hongxiang Li, Yaowei Li, Bin Lin, Yuwei Niu, Yuhang Yang, Xiaoshuang Huang, Jiayin Cai, Xiaolong Jiang, Yao Hu, and Long Chen. 2026b. GIR-bench: Versatile benchmark for generating images with reasoning. In *The Fourteenth International Conference on Learning Representations*.

Yunxin Li, Xinyu Chen, Shenyuan Jiang, Haoyuan Shi, Zhenyu Liu, Xuanyu Zhang, Nanhao Deng, Zhenran Xu, Yicheng Ma, Meishan Zhang, and 1 others. 2025a. Uni-moe-2.0-omni: Scaling language-centric omnimodal large model with advanced moe, training and data. *arXiv preprint arXiv:2511.12609*.

Yunxin Li, Zhenyu Liu, Zitao Li, Xuanyu Zhang, Zhenran Xu, Xinyu Chen, Haoyuan Shi, Shenyuan Jiang, Xintong Wang, Jifang Wang, and 1 others. 2025b. Perception, reason, think, and plan: A survey on large multimodal reasoning models. *arXiv preprint arXiv:2505.04921*.

Zijie Li, Henry Li, Yichun Shi, Amir Barati Farimani, Yuval Kluger, Linjie Yang, and Peng Wang. 2025c. Dual diffusion for unified image generation and understanding. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 2779–2790.

Yongyuan Liang, Wei Chow, Feng Li, Ziqiao Ma, Xiyao Wang, Jiageng Mao, Jiuhai Chen, Jiatao Gu, Yue Wang, and Furong Huang. 2026. ROVER: Benchmarking reciprocal cross-modal reasoning for omnimodal generation. In *The Fourteenth International Conference on Learning Representations*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916.

Ruilin Luo, Chufan Shi, Yizhen Zhang, Cheng Yang, Songtao Jiang, Tongkun Guan, Ruizhe Chen, Ruihang Chu, Peng Wang, Mingkun Yang, Yujiu Yang, Junyang Lin, and Zhibo Yang. 2026. From narrow to panoramic vision: Attention-guided cold-start reshapes multimodal reasoning. In *International Conference on Learning Representations*.

Ruilin Luo, Zhuofan Zheng, Lei Wang, Yifan Wang, Xinzhe Ni, Zicheng Lin, Songtao Jiang, Yiyao Yu, Chufan Shi, Ruihang Chu, and 1 others. 2025. Unlocking multimodal mathematical reasoning via process reward model. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Yuwei Niu, Weiyang Jin, Jiaqi Liao, Chaoran Feng, Peng Jin, Bin Lin, Zongjian Li, Bin Zhu, Weihao Yu, and Li Yuan. 2025a. Does understanding inform generation in unified multimodal models? from analysis to path forward. *arXiv preprint arXiv:2511.20561*.

Yuwei Niu, Munan Ning, Mengren Zheng, Weiyang Jin, Bin Lin, Peng Jin, Jiaqi Liao, Chaoran Feng, Kunpeng Ning, Bin Zhu, and 1 others. 2025b. Wise: A world knowledge-informed semantic evaluation for text-to-image generation. *arXiv preprint arXiv:2503.07265*.

OpenAI. 2024. Learning to reason with llms.

OpenAI. 2025. OpenAI o3-mini.

Luozheng Qin, GONG JIA, Yuqing Sun, Tianjiao Li, Haoyu Pan, Mengping Yang, Xiaomeng Yang, Chao Qu, Zhiyu Tan, and Hao Li. 2026. Uni-cot: Towards unified chain-of-thought reasoning across text and vision. In *The Fourteenth International Conference on Learning Representations*.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, and 1 others. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494.

Axel Sauer, Tero Karras, Samuli Laine, Andreas Geiger, and Timo Aila. 2023. Stylegan-t: Unlocking the power of gans for fast large-scale text-to-image synthesis. In *International conference on machine learning*, pages 30105–30118. PMLR.

Patrick Schramowski, Manuel Brack, Björn Deiseroth, and Kristian Kersting. 2023. Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22522–22531.

Preethi Seshadri, Sameer Singh, and Yanai Elazar. 2024. The bias amplification paradox in text-to-image generation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6367–6384.

Chufan Shi, Cheng Yang, Yaokang Wu, Linghao Jin, Bo Shui, Taylor Berg-Kirkpatrick, and Xuezhe Ma. 2026. Are vlms seeing or just saying? uncovering the illusion of visual re-examination. *arXiv preprint arXiv:2605.15864*.

Chufan Shi, Haoran Yang, Deng Cai, Zhisong Zhang, Yifan Wang, Yujiu Yang, and Wai Lam. 2024. A thorough examination of decoding methods in the era of LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8601–8629. Association for Computational Linguistics.

Qingyu Shi, Jinbin Bai, Zhuoran Zhao, Wenhao Chai, Kaidong Yu, Jianzong Wu, Shuangyong Song, Yunhai Tong, Xiangtai Li, Xuelong Li, and 1 others. 2025. Muddit: Liberating generation beyond text-to-image with a unified discrete diffusion model. *arXiv preprint arXiv:2505.23606*.

Kaiyue Sun, Rongyao Fang, Chengqi Duan, Xian Liu, and Xihui Liu. 2025. T2i-reasonbench: Benchmarking reasoning-informed text-to-image generation. *arXiv preprint arXiv:2508.17472*.

Alexander Swerdlow, Mihir Prabhudesai, Siddharth Gandhi, Deepak Pathak, and Katerina Fragkiadaki. 2025. Unified multimodal discrete diffusion. *arXiv preprint arXiv:2503.20853*.

Raphael Tang, Linqing Liu, Akshat Pandey, Zhiying Jiang, Gefei Yang, Karun Kumar, Pontus Stenetorp, Jimmy Lin, and Ferhan Türe. 2023. What the daam: Interpreting stable diffusion using cross attention. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5644–5659.

Muzi Tao, Chufan Shi, Huijuan Wang, Shengbang Tong, and Xuezhe Ma. 2026. Asymmetric idiosyncrasies in multimodal models. In *Conference on Language Modeling*.

Chameleon Team. 2024. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*.

Qwen Team. 2025a. Qwen3-vl: Sharper vision, deeper thought, broader action. *Qwen Blog. Accessed*, pages 10–04.

Qwen-Image Team. 2025b. Qwen image. https://huggingface.co/spaces/Qwen/Qwen-Image. Accessed: 2026-05-23.

Shengbang Tong, David Fan, Jiachen Li, Yunyang Xiong, Xinlei Chen, Koustuv Sinha, Michael Rabbat, Yann LeCun, Saining Xie, and Zhuang Liu. 2025. Metamorph: Multimodal understanding and generation via instruction tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17001–17012.

Shengbang Tong, David Fan, John Nguyen, Ellis Brown, Gaoyue Zhou, Shengyi Qian, Boyang Zheng, Théophane Vallaeys, Junlin Han, Rob Fergus, and 1 others. 2026. Beyond language modeling: An exploration of multimodal pretraining. *arXiv preprint arXiv:2603.03276*.

Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, and 1 others. 2024. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, and 1 others. 2025a. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*.

Yongliang Wu, Zonghui Li, Xinting Hu, Xinyu Ye, Xianfang Zeng, Gang YU, Wenbo Zhu, Bernt Schiele, Ming-Hsuan Yang, and Xu Yang. 2025b. KRIS-bench: Benchmarking next-level intelligent image editing models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. 2024. Show-o: One single transformer to unify multimodal understanding and generation. *arXiv preprint arXiv:2408.12528*.

Zhiyuan Yan, Kaiqing Lin, Zongjian Li, Junyan Ye, Hui Han, Haochen Wang, Zhendong Wang, Bin Lin, Hao Li, Xinyan Xiao, Jingdong Wang, Haifeng Wang, and Li Yuan. 2026. Unified multimodal models as auto-encoders. *Preprint*, arXiv:2509.09666.

Cheng Yang, Chufan Shi, Yaxin Liu, Bo Shui, Junjie Wang, Mohan Jing, Linran Xu, Xinyu Zhu, Siheng Li, Yuxiang Zhang, Gongye Liu, Xiaomei Nie, Deng

Cai, and Yujiu Yang. 2025. Chartmimic: Evaluating lmm's cross-modal reasoning capability via chart-to-code generation. In *International Conference on Learning Representations*, volume 2025, pages 26590–26646.

Jaehong Yoon, Shoubin Yu, Vaidehi Patil, Huaxiu Yao, and Mohit Bansal. 2025. Safree: Training-free and adaptive guard for safe text-to-image and video generation. In *International Conference on Learning Representations*.

Xinchen Zhang, Bowei Liu, Jiale Liu, Chufan Shi, Yizhen Zhang, Junhong Liu, Youliang Zhang, Zhiheng Li, Yujiu Yang, and Ling Yang. 2026. Omniverifier-m1: Multimodal meta-verifier with explicit structured recalibration. *arXiv preprint arXiv:2605.28805*.

Xinchen Zhang, Xiaoying Zhang, Youbin Wu, Yanbin Cao, Renrui Zhang, Ruihang Chu, Ling Yang, and Yujiu Yang. 2025. Generative universal verifier as multimodal meta-reasoner. *arXiv preprint arXiv:2510.13804*.

Xiangyu Zhao, Peiyuan Zhang, Kexian Tang, Xiaorong Zhu, Hao Li, Wenhao Chai, Zicheng Zhang, Renqiu Xia, Guangtao Zhai, Junchi Yan, Hua Yang, Xue Yang, and Haodong Duan. 2025. Envisioning beyond the pixels: Benchmarking reasoning-informed visual editing. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Chunting Zhou, Lili Yu, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. 2024. Transfusion: Predict the next token and diffuse images with one multi-modal model. *arXiv preprint arXiv:2408.11039*.

Appendix

A LLM Disclosure **15**

B Data Annotation **15**
B.1 Task Taxonomy 15
B.2 Annotation Pipeline 16
B.3 Data Statistics 17
B.4 Details on Comparison with Existing Benchmarks 17

C Details on Evaluation Settings **17**

D Discussion on the Scope of Reasoning **18**

E Additional Controlled Analyses **18**

F Consistency Between the Test and Testmini Sets **18**

G Context Length Statistics for Training and Evaluation **19**

H Details on Attention Analysis **19**
H.1 Attention Weight Computation . . 19
H.2 Qualitative Token Influence Analysis 19
H.3 Attention Analysis on ThinkMorph and T2I-R1 20
H.4 Potential Mitigation Directions . . 20

I Discussion on Evaluation Metrics **20**
I.1 Details on MLLM-as-a-Judge . . . 20
I.2 Details on Human Evaluation . . . 20

J Discussion on Closed-Source Systems **21**

K Model Repositories **22**

L Reproducibility **22**

M Prompts **22**

N Case Study **22**
N.1 Error Cases 22
N.2 More Qualitative Results 24

O Ethical Statement **24**

A LLM Disclosure

We use Gemini-3-Pro to conduct LLM-Assisted Data Augmentation as detailed in Section B. LLMs are used to assist with drafting and polishing paper text. No LLMs are used to originate research ideas.

B Data Annotation

B.1 Task Taxonomy

To guarantee the quality and diversity of UReason, human experts first establish a fine-grained taxonomy spanning 30 subcategories under the five primary tasks (Fig. 4). Each subcategory is designed to isolate a distinct reasoning capability or visual aspect, intentionally focusing on fundamental operations and visual characteristics. This granular design enables convenient and precise error localization and diagnostic analysis. Importantly, these subcategories are not mutually exclusive and can be combined to construct test cases when needed.

CODE REASONING UReason defines eight subcategories for Code Reasoning based on programming languages, covering object-oriented languages (Python, C#, C++, Java), front-end technologies (HTML, CSS, JavaScript), SQL for data querying, and various programming paradigms. Models are required to understand language-specific syntax, control flow, data structures, and computational logic to simulate code execution and determine the final visual rendering. This comprehensive coverage ensures that models must possess broad code comprehension capabilities across different programming ecosystems. Notably, code snippets are designed such that their visual outputs involve arithmetic counts, spatial layouts, attribute constraints, and text rendering.

ARITHMETIC REASONING UReason defines seven subcategories for Arithmetic Reasoning covering different operational complexities and object configurations. Single-Type (St) and Multi-Type (Mt) subcategories distinguish scenarios based on object diversity, while Add (Add), Subtract (Sub), Multiply (Mul), Divide (Div), and Transfer (Trf) subcategories focus on specific arithmetic operations. Models are required to perform sequential numerical reasoning through narrative events, tracking quantity changes dynamically and ensuring that the final visual output strictly reflects the calculated object count. This demands models to transcend superficial keyword matching and function as quantitative reasoners.

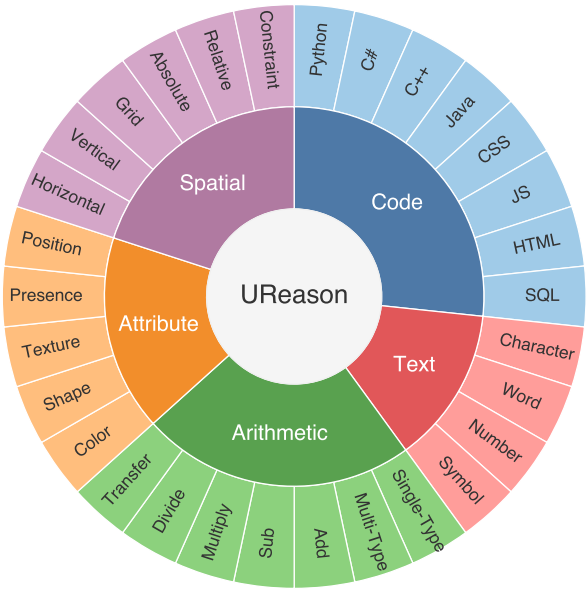


Figure 4: Taxonomy of UReason tasks. The benchmark contains 5 task categories with 30 fine-grained subcategories covering diverse reasoning and visual generation challenges.

SPATIAL REASONING UReason defines six subcategories for Spatial Reasoning covering different spatial arrangement paradigms: Horizontal (Hor), Vertical (Ver), Grid (Grd), Absolute (Abs), Relative (Rel), and Constraint (Cst). These subcategories assess models' capacity to resolve high-level semantic descriptions into structured coordinate-based layouts. Unlike standard benchmarks where spatial relationships are explicitly stated, models must infer spatial configurations from implicit cues, logical constraints, and relational reasoning. The Constraint subcategory particularly challenges models to act as spatial reasoners that interpret and satisfy predefined placement rules and restrictions—such as "object A cannot be adjacent to object B" or "all red objects must be on the left side"—before determining the final spatial arrangement that complies with all specified constraints.

ATTRIBUTE REASONING UReason defines five subcategories for Attribute Reasoning based on different object properties: Color (Clr), Shape (Shp), Texture (Tex), Presence (Prs) and Position (Pos). These subcategories evaluate models' capability to track and update object attributes through state transitions and logical modifications described in the text. Models must perform logical filtering to derive the terminal outcome rather than rendering initial or intermediate states, effectively suppressing visual biases toward irrelevant attributes men-

tioned in the prompt. This requires maintaining attribute consistency throughout complex state evolution narratives.

TEXT REASONING UReason defines four subcategories for Text Reasoning based on the granularity of textual elements: Character (Chr), Word (Wrd), Number (Num), and Symbol (Sym). These subcategories assess models' ability to perform context-aware text inference at different semantic levels. Models must derive the target text through contextual rules, linguistic transformations, mathematical operations, or symbolic reasoning, rather than directly rendering explicitly quoted strings. This requires models to act as symbolic reasoners that suppress irrelevant information and render only the logically derived final result.

B.2 Annotation Pipeline

In this section, we provide comprehensive details about the data curation process of UReason, including the human-curated seed data construction, LLM-assisted augmentation pipeline, and data statistics.

B.2.1 Human-Curated Seed Data Construction

Annotator Background. Our annotation team consists of 5 expert annotators with professional backgrounds in computer vision and natural language processing. All annotators possess at least a Master's degree in computer science, with an average of 3 years of experience in AI research.

Taxonomy Design Process. The design of our fine-grained taxonomy follows a systematic, multi-stage approach. We first identify five primary reasoning dimensions—Code, Arithmetic, Spatial, Attribute, and Text—based on a literature review of reasoning capabilities in language models and an analysis of real-world visual generation requirements. For each primary category, we conduct brainstorming sessions with human annotators to identify representative subcategories, with selection criteria emphasizing coverage of diverse reasoning patterns, distinctiveness in targeting specific skills, verifiability for objective evaluation and practical relevance to real-world use cases. We then conduct a pilot study with 10 instances per subcategory to verify feasibility. After iterative refinement, we establish the final taxonomy comprising 30 subcategories across 5 main tasks, as illustrated in Fig. 4.

Seed Instance Construction Details. Expert annotators manually construct seed instances following specific design principles:

As shown in Fig. 1, for prompt design, each prompt necessitates intermediate reasoning steps to derive the visual target. While the target remains implicit, the prompt provides sufficient information for a unique, deterministic solution. For evaluation criterion design, each criterion specifies a concrete, verifiable aspect of the generated image, such as exact object counts, specific text strings, or precise spatial relationships. We specify exact values rather than ranges for quantitative attributes and define clear verification rules for qualitative attributes. Different task categories employ tailored criterion formats: Arithmetic tasks specify exact object counts after all operations, Spatial tasks define precise positional relationships or arrangements, Attribute tasks indicate final resolved object attributes, Text tasks require exact text strings to be rendered, and Code tasks encompass all four of the above aspects.

Notably, our early pilot experiments reveal that excessively large quantities or lengthy text strings pose significant challenges for current UMMs, prompting us to adjust the difficulty levels accordingly in our annotation process.

B.2.2 LLM-Assisted Data Augmentation

We use Gemini-3-Pro (DeepMind, 2025) to generate 10 candidate variants for each of the 500 seed instances, yielding 5,000 candidates. We systematically vary three dimensions while preserving a unique answer: reasoning complexity, target visual entities, and narrative context. Complexity is controlled by adding or removing reasoning steps and distractors; visual variation includes object types, quantities, colors, shapes, sizes, and textures; narrative variation covers linguistic styles and everyday or scientific scenarios. Gemini is used only to propose candidates; all inclusion decisions and final labels are made by human annotators. We also constrain quantities and text lengths based on pilot studies so that the resulting criteria remain visually verifiable.

Human-LLM Interaction Workflow. Each of the 5,000 candidates is independently labeled by three expert annotators as accepted, requiring revision, or rejected. Inter-annotator agreement is substantial (Fleiss' $\kappa = 0.78$). Disagreements are resolved through discussion, and annotators retain

the best three candidates per seed, corresponding to a 30% adoption rate; the remaining 70% are not adopted.

Candidates requiring revision enter a multi-round human–LLM refinement loop. Every retained candidate then undergoes unanimous final human verification for logical correctness, label correctness, diversity, stylistic variation, and compliance with the evaluation criterion. Together with the 500 manually authored seeds, all 1,500 retained variants pass this final verification to form the 2,000-instance benchmark. We additionally inspect candidates for shortcut cues and superficial patterns that could reveal the target without the intended reasoning.

B.3 Data Statistics

Tab. 8 presents the statistics of UReason. The benchmark comprises $2,000$ instances evenly distributed across five task categories (400 each), spanning 30 fine-grained subcategories. Subcategory counts reflect each domain's scope: CODE encompasses 8 programming languages, while ARITHMETIC, SPATIAL, ATTRIBUTE, and TEXT contain 7, 6, 5, and 4 subcategories respectively. Prompt lengths are measured using the Qwen3-8B tokenizer.

B.4 Details on Comparison with Existing Benchmarks

Tab. 1 compares UReason with existing reasoning-driven image generation benchmarks. Compared with prior text-to-image and image-editing benchmarks, UReason provides a larger and more fine-grained diagnostic testbed, containing 2,000 instances across 5 task categories and 30 subcategories. This scale and taxonomy allow UReason to cover a broader range of reasoning-to-generation challenges, including CODE, ARITHMETIC, SPATIAL, ATTRIBUTE, and TEXT reasoning. In particular, UReason introduces the under-explored CODE reasoning domain, where models must interpret code snippets across multiple programming languages, reason over execution logic or state changes, and synthesize the corresponding visual outcomes.

Beyond dataset coverage, UReason differs from prior benchmarks in how it measures reasoning-to-generation alignment. Most existing benchmarks primarily evaluate final-image performance under direct generation, and some only run preliminary reasoning-chain experiments on specific models. In contrast, UReason evaluates whether the image generated in the same inference process faithfully reflects the model's own textual reasoning. By further comparing Reasoning-Guided Generation with De-contextualized Generation, UReason provides a controlled way to measure whether intermediate reasoning context helps or interferes with visual synthesis. This formulation differs from benchmarks that infer alignment or consistency indirectly from final task performance across separate multimodality understanding and generation tasks (Li et al., 2026b). Therefore, UReason contributes both a broad benchmark with new task coverage and a controlled protocol for analyzing reasoning-to-generation alignment in UMMs.

C Details on Evaluation Settings

Our evaluation settings are designed to diagnose whether visual generation in reasoning-guided image generation faithfully reflects the intended visual semantics expressed in textual reasoning, thereby providing a controlled lens on reasoning-to-generation alignment in UMMs.

UMMs typically support two generation modes: *Direct Generation*, which generates an image directly from the user instruction, and *Reasoning-Guided Generation*, which generates textual reasoning for image generation. We additionally introduce *De-contextualized Generation*, which conditions image generation only on the model's intended visual specification[1].

A key design choice is how to obtain this intended visual specification. Rather than relying on an external model or human annotation to summarize the reasoning trace, we explicitly require the evaluated model to output a refined prompt, R_p, at the end of the reasoning trace. This refined prompt serves as the model's own textual summary of the final visual target, i.e., the intended visual semantics that the model aims to realize in the image.

This design is important for fairness and comparability. In principle, one could attempt to infer the intended visual semantics directly from the textual reasoning trace. However, doing so would typically require an additional summarization step, often involving an external model, to convert reasoning trace into a concise visual specification. Such a multi-stage pipeline would introduce additional

[1] In this paper, we use *intended visual semantics* to refer to the final visual target in the abstract representational space, and *intended visual specification* to refer to its expression in concrete textual form.

Category	Code	Arithmetic	Spatial	Attribute	Text	Overall
Count	400	400	400	400	400	2000
Subcategories	8	7	6	5	4	30
Prompt Length (AVG.)	113.14	131.44	169.27	97.78	79.66	118.26
Prompt Length (STD.)	51.61	13.11	35.90	18.79	8.33	42.97

Table 8: Statistics of UReason across different tasks.

model bias and cumulative error, making it harder to attribute performance differences to the evaluated UMM itself. By instead requiring the model to explicitly produce R_p, we ensure that the decontextualized setting uses the model's own final visual specification while avoiding confounding effects from external summarization.

This formulation also brings practical benefits. Because R_p is explicitly expressed in text, it provides an interpretable representation of the intended visual semantics for downstream analysis. This supports not only a fair comparison between Reasoning-Guided Generation and Decontextualized Generation, but also subsequent analyses of reasoning chains and attention behavior discussed in the paper.

D Discussion on the Scope of Reasoning

In this work, reasoning refers to the free-form, linear solution traces produced by existing reasoning-guided UMMs before image synthesis. Self-generated reflection also does not necessarily induce renewed visual grounding (Shi et al., 2026). We do not study structured reasoning frameworks that rely on explicit search, path selection, or process-level verification, such as Tree of Thoughts or Monte Carlo tree search. Existing UMMs are trained primarily on free-form reasoning data, so imposing a structured format only at inference time could introduce a train–test mismatch. Extending unified models with structured and verifiable reasoning remains an important direction for future work.

E Additional Controlled Analyses

Prompt-Format and Delimiter Controls. We test whether the Setting-2 gap is an artifact of the output template by changing only its format to an equation-style representation, `thinking="..."`, `refined_prompt="..."`. We separately prepend empty `<think></think>` delimiters to the same R_p in Setting 3. Seeds, samplers, and all sampling parameters are fixed within each comparison. Tab. 9 shows that the format changes overall accuracy by at most 0.8 points and the delimiters by at most 2.4 points, both far smaller than the Setting-2-to-Setting-3 gaps.

Holding the Refined Prompt Fixed. Starting from Bagel's Setting-2 output, we parse the trace into $[R_t, R_p]$, retain the same R_p and generation format, and replace R_t with a trace sampled from an unrelated instance generated by the same model. Tab. 10 compares R_p alone, the original $R_t + R_p$, and the unrelated $R_t + R_p$. Adding the original trace reduces accuracy from 62.6% to 21.0%; an unrelated trace reduces it further to 16.4%. This intervention isolates the effect of preceding reasoning content while controlling for the refined prompt, delimiters, format, and generation mode.

Subsampling Stability. We randomly draw 100 instances per task from the full 2,000-instance benchmark and recompute overall accuracy on each resulting 500-instance subset, repeating the procedure 2,000 times. For Bagel, the subsampled accuracies are $64.1 \pm 2.3\%$ under Setting 3 and $19.5 \pm 2.1\%$ under Setting 2. This variability is much smaller than the observed 44.8-point gap, showing that the result is stable to the particular testmini sample.

F Consistency Between the Test and Testmini Sets

Tab. 11 reports detailed performance for all eight unified multimodal models across all three evaluation settings on both the full test set and its testmini subset. The results demonstrate strong consistency between the full test set and testmini. The consistent trends and minimal performance gaps suggest that testmini effectively mirrors the full test set, serving as a reliable and efficient evaluation subset for model development, particularly for researchers with limited computational resources.

Control	Model/Condition	Code	Arith.	Spatial	Attr.	Text	Overall
Prompt format	Bagel, original	30.0	14.0	15.0	19.0	11.0	17.8
	Bagel, equation format	29.0	17.0	13.0	16.0	12.0	17.4
	UniCoT-v2, original	27.0	19.0	29.0	21.0	21.0	23.4
	UniCoT-v2, equation format	25.0	19.0	27.0	20.0	22.0	22.6
Delimiter	Bagel, R_p only	58.0	51.0	58.0	60.0	86.0	62.6
	Bagel, empty delimiters $+R_p$	54.0	49.0	58.0	61.0	83.0	61.0
	UniCoT-v2, R_p only	65.0	45.0	46.0	67.0	87.0	62.0
	UniCoT-v2, empty delimiters $+R_p$	63.0	43.0	43.0	64.0	85.0	59.6

Table 9: Prompt-format and delimiter controls on testmini. The equation format uses `thinking="..."`, `refined_prompt="..."`; the delimiter control prepends `<think></think>` to the same R_p. Seeds, samplers, and sampling parameters are held fixed within each comparison.

Context	Code	Arith.	Spatial	Attr.	Text	Overall
R_p only	58.0	51.0	58.0	60.0	86.0	62.6
Original $R_t + R_p$	35.0	15.0	20.0	21.0	14.0	21.0
Unrelated $R_t + R_p$	31.0	13.0	15.0	15.0	8.0	16.4

Table 10: Controlled intervention on Bagel (accuracy, %). The refined prompt and generation format are fixed while the preceding reasoning content is varied.

G Context Length Statistics for Training and Evaluation

The UMMs evaluated in our experiments are all explicitly post-trained for reasoning-guided image generation, which helps avoid the concern that the reasoning-style context or their lengths are out of distribution for the models. To further verify that the performance degradation observed in Reasoning-guided Generation is not attributable to context lengths, we compare the token lengths of open-source reasoning-guided image generation training data for ThinkMorph (Gu et al., 2026) and Bagel-Zebra-CoT (Li et al., 2026a), both of which explicitly include reasoning-guided image generation data, against the average textual context length encountered during UReason evaluation. As shown in Tab. 12, UReason evaluation sequences are consistently shorter than the models' training distribution across both models, confirming that the models are not exposed to unprecedented sequence lengths.

H Details on Attention Analysis

H.1 Attention Weight Computation

Attention allocation has also been used to diagnose multimodal reasoning (Luo et al., 2026). To explore the causal mechanisms, we conduct an attention analysis on Bagel (Deng et al., 2025) in Sec. 5. Here, we present more details on the attention analysis. To analyze attention during image generation, we consider three semantic groups in the input sequence: the original prompt P, the intermediate reasoning trace R_t, and the refined prompt R_p. At each diffusion timestep, for each layer ℓ, the attention weight matrix $\mathbf{A}^{(\ell)} \in \mathbb{R}^{H \times L_q \times L_k}$ captures the normalized attention each query token assigns to every key token, where H is the number of attention heads, and L_q, L_k denote the query and key sequence lengths, respectively. The query tokens correspond to the visual generation tokens produced during image synthesis, while the key tokens span the full input context including P, R_t, and R_p. The attention received by each token group is obtained by averaging over the corresponding key positions and attention heads:

$$\bar{a}_{\mathcal{G}}^{(\ell)} = \frac{1}{H} \sum_{h=1}^{H} \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{A}_{h,:,j}^{(\ell)} \tag{6}$$

where $\mathcal{G} \in \{P,\ R_t,\ R_p\}$ denotes the index set of tokens belonging to each group. The final reported values are averaged across all evaluated samples, all diffusion timesteps, and all query tokens within each of the four contiguous layer groups (layers 1–7, 8–14, 15–21, and 22–28).

H.2 Qualitative Token Influence Analysis

To provide a more intuitive understanding of contextual interference, we further visualize token influence maps following DAAM (Tang et al., 2023). For each target noun token within R_t, we isolate its attention weights from $\mathbf{A}^{(\ell)}$ and aggregate them across diffusion timesteps and layers, producing maps that highlight the pixel-level regions most strongly influenced by each token. As shown in Figure 5, even when the logical flow within R_t correctly deduces that "cream" should be excluded, the image still contains cream. Moreover, the token

"cream" is primarily associated with the cream region in the generated image through cross-attention. To further corroborate this, we manually select a local region in the generated image where cream appears, and compute the attention scores from the corresponding visual generation tokens to all textual tokens. As shown in Figure 8, "cream" ranks among the top-10 most attended textual tokens for that region, excluding special tokens (e.g., start/end of sentence token). Together, these findings suggest that visual synthesis can be influenced by the mere presence of a token in the conditioning context. More broadly, these findings suggest that semantic contextual interference may be one factor associated with the observed reasoning-to-generation alignment gap.

Additional qualitative cases are provided in Figures 6 and 7, with their corresponding attention score analyses detailed in Figures 9 and 10.

H.3 Attention Analysis on ThinkMorph and T2I-R1

To further validate our findings beyond a single model, we additionally conduct the same attention analysis on ThinkMorph (Gu et al., 2026) and T2I-R1 (Jiang et al., 2025), which generates images in an autoregressive manner. Specifically, we extract the average attention weights assigned to the three token groups P, R_t, and R_p following the same procedure described above. The results are consistent with our observations on Bagel: for UMMs, visual generation remains highly sensitive to surrounding context, which may compete with the final visual specification for attention and weaken the transfer from intended visual semantics to pixels. We use the term contextual interference to distinguish this phenomenon from general prompt sensitivity and compression: here, the issue stems from interfering information within the conditioning context, rather than ordinary variations in prompt wording or formatting. Related work similarly finds that textual distinctions can weaken during image generation (Tao et al., 2026).

H.4 Potential Mitigation Directions

Training-free suppression methods developed for safe image generation, such as SAFREE (Yoon et al., 2025) and PurifyGen (Cao et al., 2025), suggest possible mechanisms for selectively suppressing interfering content while reinforcing the final visual specification. Adapting these methods to reasoning-guided UMMs is non-trivial because the concepts to suppress are instance-specific and must be inferred from the model's own reasoning, rather than selected from a predefined concept set. Such methods would also need to operate within a unified transformer context rather than a separate text-encoder conditioning pathway. We leave this adaptation as a direction for future work.

I Discussion on Evaluation Metrics

I.1 Details on MLLM-as-a-Judge

Unlike evaluations that test open-ended and subjective alignment, UReason evaluates strictly verifiable ground-truth criteria, including exact object counts and specific text strings. By relying on these deterministic criteria, we frame the evaluation as a binary Visual Question Answering (VQA) problem (i.e., Yes/No judgment) rather than an open-ended assessment of overall image quality. Following prior analyses of LLM decoding sensitivity (Shi et al., 2024), we set the temperature to 0.1 for all automated MLLM/LLM evaluators. This formulation substantially reduces subjectivity and evaluator dependence, following a methodology widely adopted when evaluating images against deterministic targets (Zhao et al., 2025; Niu et al., 2025b; Chen et al., 2025a; Wu et al., 2025b).

I.2 Details on Human Evaluation

To validate the automated metrics described in Sec. 3.2, we conduct human evaluation on Uni-CoT, Bagel, and T2I-R1. UniCoT is evaluated over all 500 testmini instances, and we extend the same evaluation protocol to Bagel and T2I-R1. For each evaluated instance, annotators assess images from all three diagnostic settings and the reasoning trace from Setting 2.

Evaluators. Our evaluation panel consists of three graduate students, all holding Master's degrees in Computer Science with research experience in computer vision or natural language processing.

Annotation Task and Interface. A screenshot of the annotation interface is provided in Figure 11. Each generated image and reasoning trace is paired with its corresponding ground-truth criterion C, which specifies a concrete and verifiable target outcome. The annotation task is framed as an objective binary judgment problem: given a generated image or reasoning trace alongside the criterion, the evaluator judges whether it satisfies C, selecting

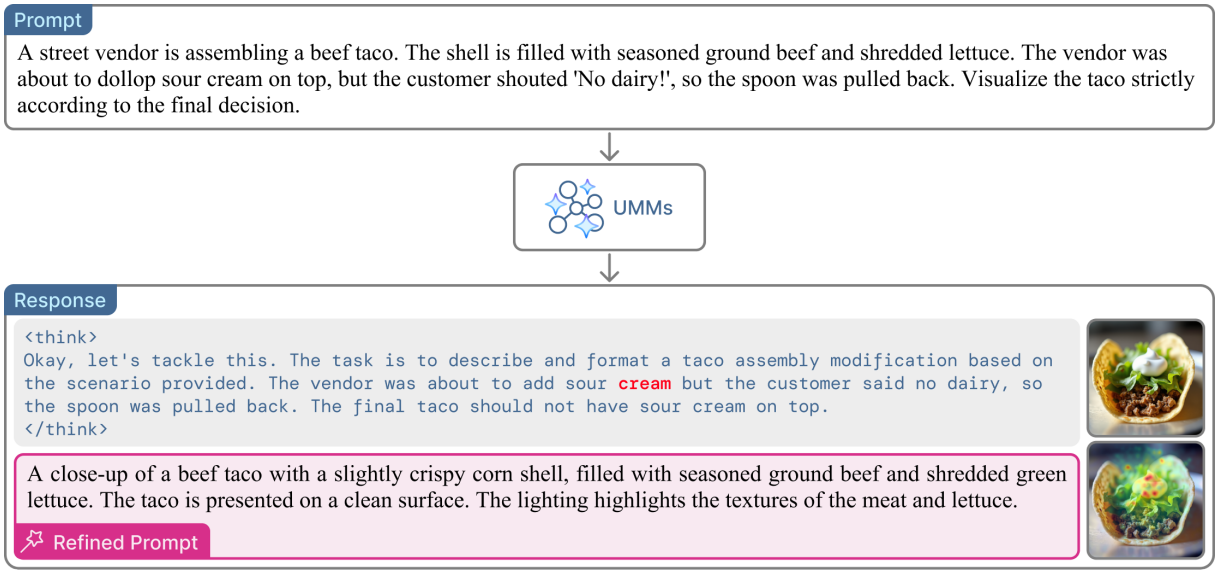


Figure 5: Qualitative token influence maps following DAAM (Tang et al., 2023). Although R_t correctly deduces that sour cream should be excluded, the token "cream" (highlighted in red) still induces strong localized attention in the attention layers, correlating with the region where sour cream is generated. This suggests that token presence can influence visual synthesis independently of the surrounding logical context, bypassing the constraints of the refined prompt R_p.

either Yes or No. This binary formulation minimizes subjectivity and aligns directly with our automated metric. Each evaluator annotated all image–criterion and reasoning-trace–criterion pairs without access to the judgments of others.

Disagreement Resolution. In cases where all three evaluators agree, the consensus label is directly adopted as the ground-truth annotation. In cases of disagreement, the three evaluators convene in a discussion session to jointly review the image or reasoning trace alongside the criterion and reach a final unanimous decision. The resolved label is recorded only after full consensus is achieved, ensuring that every ground-truth annotation is unambiguous.

Agreement Computation. We treat the final human-annotated labels after disagreement resolution as ground-truth binary scores and compare them against the binary scores produced by our automated evaluators, Qwen3-VL-235B-A22B and Qwen3-235B-A22B. Agreement is measured as the proportion of instances on which the automated evaluator and the human panel reach the same binary label. As reported in Table 15, image agreement ranges from 0.924 to 0.931 across the three evaluated models, while reasoning-chain agreement ranges from 0.962 to 0.972. For UniCoT, agreement also remains high across all five task categories. These results support the automated pipeline as a scalable proxy for human judgment on UReason's criterion-grounded binary tasks.

J Discussion on Closed-Source Systems

We acknowledge the emergence of proprietary reasoning-supported image generation systems, such as Nano Banana. As a large closed-box system, Nano Banana does not disclose whether its image generation pipeline is implemented as an end-to-end UMM or as a system composed of multiple specialized modules. It also does not provide sufficient implementation details to determine whether additional hidden reasoning modules, prompt-refinement modules, or internal strategies analogous to our de-contextualized generation setting are involved. In addition, according to its technical documentation [2], Nano Banana employs a mandatory reasoning-based iterative inference paradigm: it first performs a reasoning phase, synthesizes a preliminary visual draft, refines it through further reasoning, and then produces the final output. However, the current API encapsulates these intermediate reasoning phases [3], preventing access to the explicit reasoning chain. Consequently, we are unable to apply our diagnostic protocol to Nano Banana or isolate the effect of

[2] https://ai.google.dev/gemini-api/docs/image-generation

[3] https://ai.google.dev/gemini-api/docs/thinking

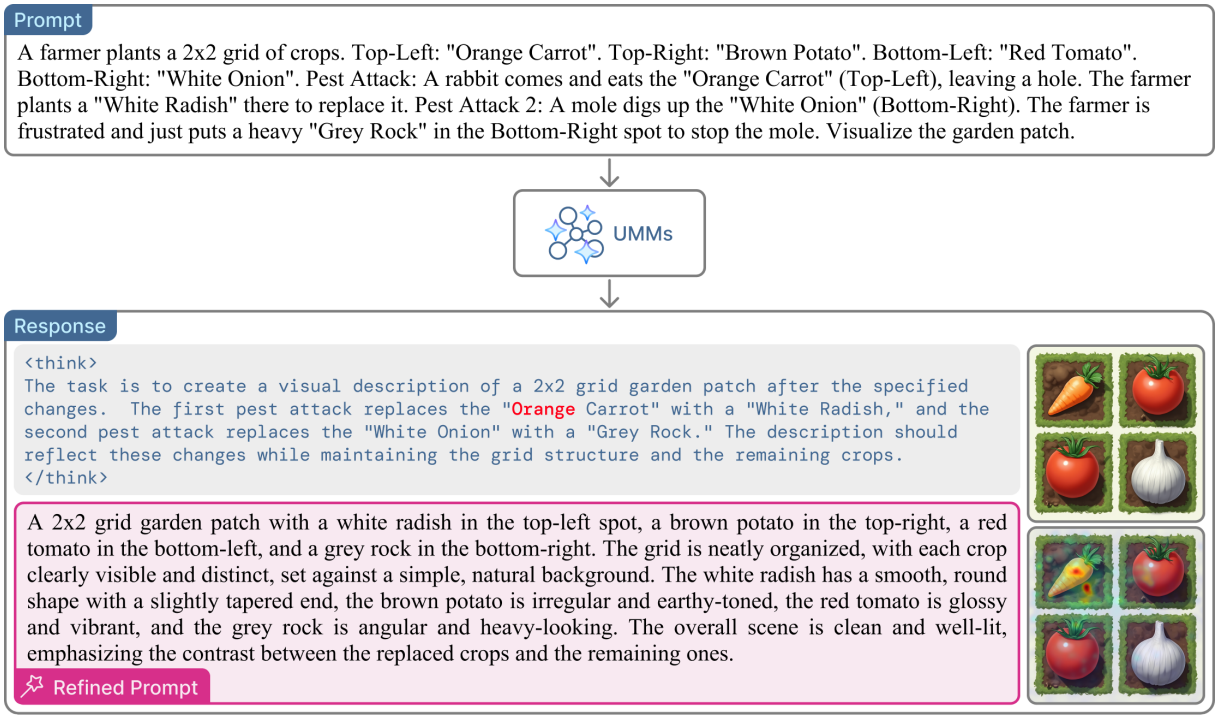


Figure 6: Qualitative token influence maps following DAAM (Tang et al., 2023). Although R_t correctly deduces that the "Orange Carrot" in the top-left should be replaced by a radish, the mere presence of the "Orange" (highlighted in red) token still induces strong localized attention in the attention layers. This correlates directly with the top-left quadrant where an orange carrot is erroneously generated, suggesting that token presence can influence visual synthesis independently of the surrounding logical context, bypassing the constraints of the refined prompt R_p.

reasoning traces in its generation process.

However, the open-source models we evaluate represent the mainstream models adopting the reasoning-guided image generation paradigm. Our experimental conclusions reveal consistent bottlenecks in how these UMMs handle alignment between textual reasoning and visual generation. Therefore, we believe our benchmark and findings provide a contribution to the open-source community, offering guidance for future improvements.

K Model Repositories

Tab. 16 summarizes the models we use and their Hugging Face repositories. For all evaluated models, we use the same generation hyperparameters across the three settings, such as decoding temperature, sampling strategy, scheduler steps, and guidance scale when applicable.

L Reproducibility

We release all 2,000 UReason instances, their evaluation criteria, the judge prompts, and the complete evaluation code. For reproducibility, Tab. 16 reports the exact model identifiers used in our experiments.

M Prompts

We present the evaluation prompts used in our automated evaluation. Fig. 12 shows the visual verification accuracy prompt, and Fig. 13 shows the reasoning chain evaluation prompt.

N Case Study

N.1 Error Cases

In this section, we provide a detailed qualitative analysis of the four failure modes identified in Sec. 5 for Bagel.

Reasoning Errors. This category involves failures where the model's intermediate reasoning process is incorrect. As illustrated in Fig. 14, the model is tasked with tracking the location and number of oranges. The target picture requires exactly "2 oranges on the wooden lid and 2 oranges on the grass." However, the model's thought process erroneously concludes that the final count is 4 oranges on the lid.

Instruction Misinterpretation. These errors occur when the model fails to grasp the fundamental modality or semantic intent of the prompt. A representative example is illustrated in Fig. 15: when

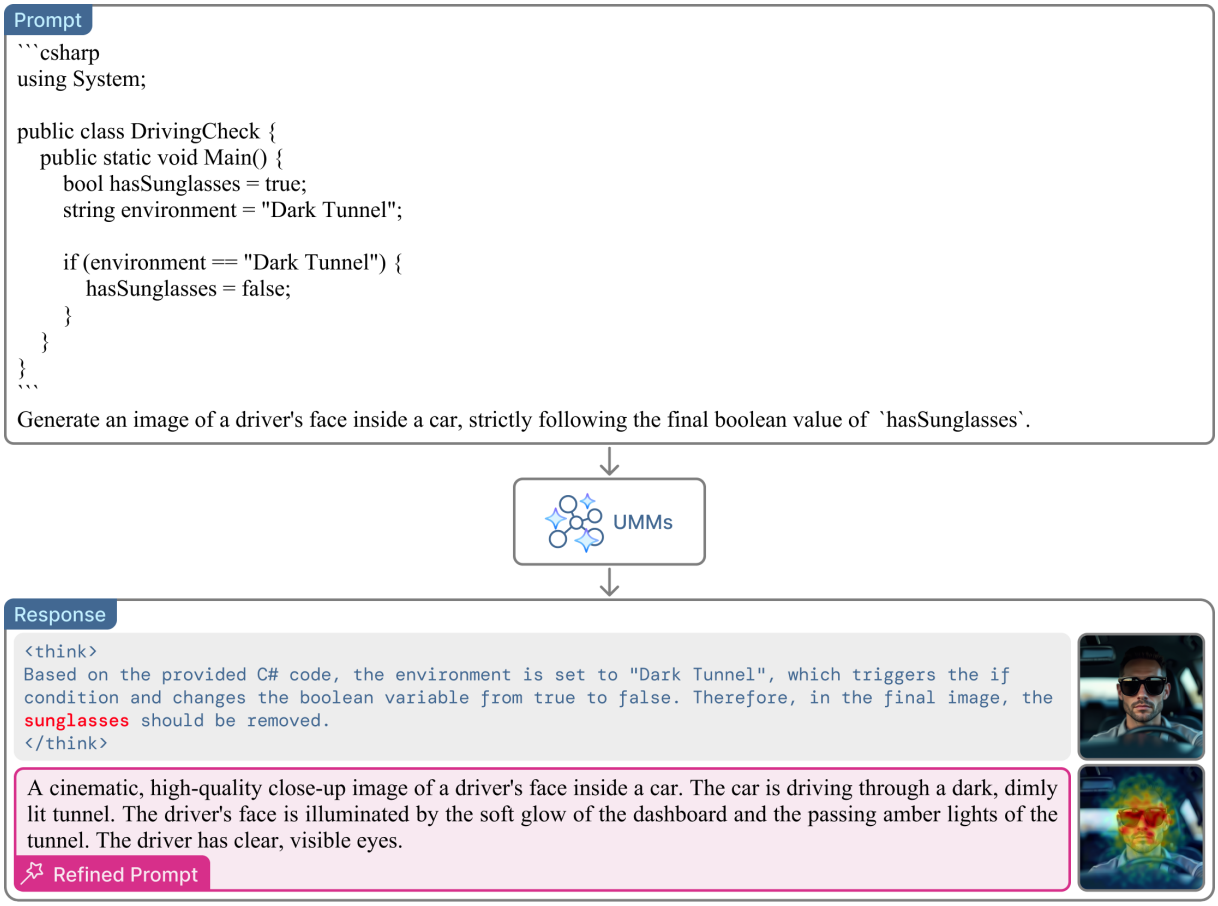


Figure 7: Qualitative token influence maps following DAAM (Tang et al., 2023). Although R_t correctly deduces from the C# code that sunglasses should be removed, the token "sunglasses"(highlighted in red) still induces strong localized attention in the attention layers, correlating with the region where sunglasses are erroneously generated. This suggests that token presence can influence visual synthesis independently of the surrounding logical context, bypassing the constraints of the refined prompt R_p.

asked to "visualize a jewelry item based on the code", the model occasionally renders the code text itself as an image rather than compiling the code into a visual object.

Concept Hallucination. This error type refers to the generation of objects that appear nowhere in the input prompt. For instance, as illustrated in Fig. 16 in a scene describing a simple garden, the model might spontaneously generate "yellow roses" despite them never being mentioned. This suggests an over-reliance on training priors rather than strict adherence to the prompt constraints.

Task-Specific Errors. This category accounts for the majority of failures. In these instances, the model successfully avoids the pitfalls of incorrect reasoning, instruction misinterpretation, and concept hallucination, yet still fails to produce a correct output. Crucially, these execution failures occur precisely in the dimensions UReason is designed to diagnose. Our fine-grained task design enables analysis of how reasoning affects arithmetic counts, spatial layouts, attribute consistency, and text rendering. This allows researchers to identify specific failure modes in the reasoning-to-generation pipeline. We analyze representative examples across the five tasks below:

- CODE: In Fig. 17, the prompt defines a specific HTML table layout for four fashion items. While the model correctly infers the 2×2 grid structure, it fails to map the specific items (Dress, Jeans, Jacket, Sneakers) to their designated table cells, resulting in misalignment despite the structural information being present in the refined prompt. This demonstrates a failure in binding semantic content to structural positions.

- ARITHMETIC: As shown in Fig. 18, the refined prompt clearly specifies the final state:

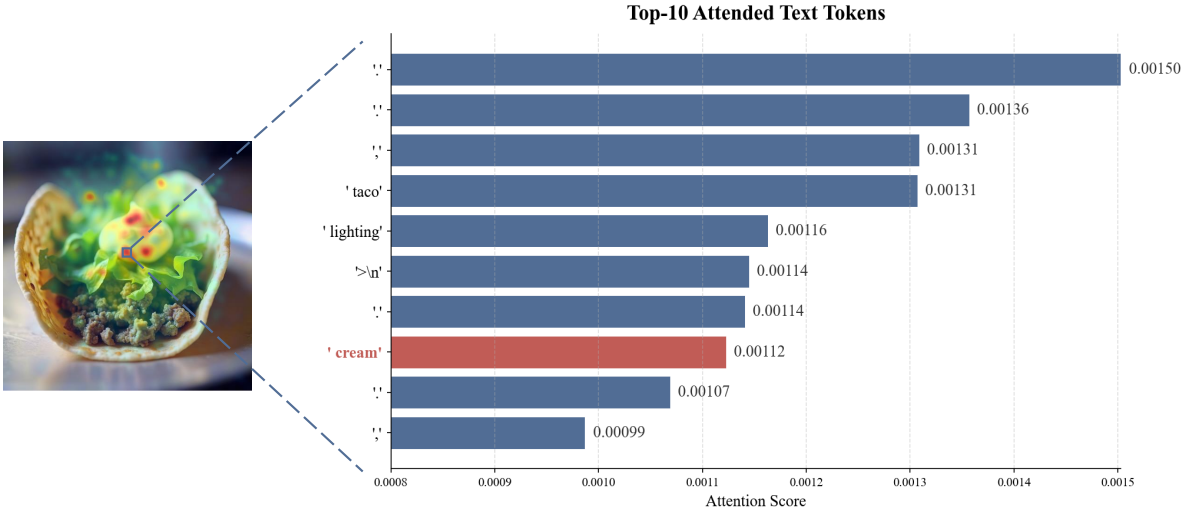


Figure 8: Top-10 attended textual tokens for the cream region in the generated image, measured by aggregating attention scores from the corresponding visual generation tokens to all textual tokens. The token "cream" appears three times in the full context: once in P, once in R_t (highlighted in red in Figure 5), and once in R_p. We compute the attention scores for the highlighted occurrence in R_t, which ranks among the top-10 most attended textual tokens for the cream region, excluding special tokens. The other two occurrences of "cream" in P and R_p rank 16 and 48, respectively.

"two green apples placed on it and a white bowl containing one green apple." The reasoning trace is concise and correct. Nevertheless, the generated image displays three apples on the table and one in the bowl, violating the count constraint. This highlights that even with a correct execution plan, current UMMs struggle to translate precise quantitative specifications into exact object counts, particularly when conditioned on verbose reasoning context.

- SPATIAL: In Fig. 19, the prompt specifies four quadrants with distinct toppings. While the model generates a pizza, it fails to maintain strict boundary separation and correct topping distribution for each quadrant, instead blending the instructions into a generic pizza image. Examination of the reasoning trace reveals lengthy intermediate steps with detailed visual descriptions. This excessive context likely acts as noise, causing long-context interference that distracts the model from adhering to strict spatial layout constraints.
- ATTRIBUTE: As shown in Fig. 20, the refined prompt explicitly describes the cup as "empty except for the ice." However, the generated image contains a brown, coffee-like liquid. This error could stem from contextual interference: the reasoning trace explicitly mentions "brown iced coffee" to describe the initial state, and this description likely acted as noise, causing the model to erroneously render the initial configuration instead of the final empty state specified in the refined prompt.
- TEXT: As seen in Fig. 21, the prompt requests the text "FIRE," but the model generates "EIME". Despite the refined prompt containing the correct string, the visual generator fails to render the characters accurately. This likely reflects the difficulty of precisely controlling character-level generation when conditioned on verbose reasoning traces, where irrelevant token associations may interfere with accurate text rendering.

N.2 More Qualitative Results

In addition to the failure modes analyzed above, we provide comprehensive qualitative comparisons across the five tasks defined in UReason: CODE, ARITHMETIC, SPATIAL, ATTRIBUTE, and TEXT reasoning. Fig. 22 to 26 showcase generated samples from representative UMMs across the three evaluation settings.

O Ethical Statement

This work introduces UReason, a benchmark for evaluating image generation. Our dataset is human-curated and human-verified, with LLM-assisted

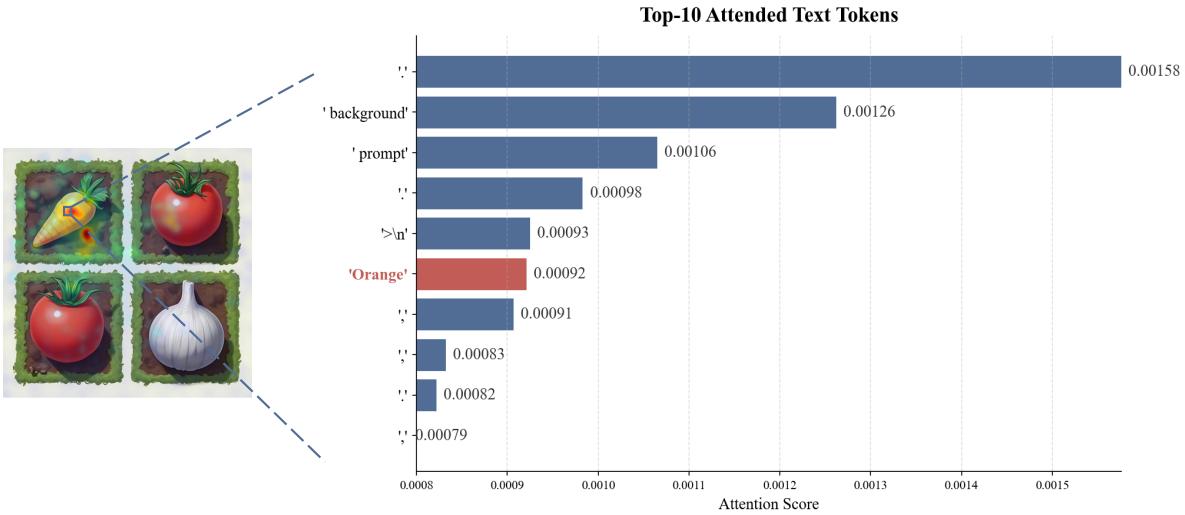


Figure 9: Top-10 attended textual tokens for the erroneously generated carrot region in the top-left quadrant of the image, measured by aggregating attention scores from the corresponding visual generation tokens to all textual tokens. We compute the attention scores for the occurrence of the token "Orange" (from "Orange Carrot") within R_t (highlighted in red in Figure 6). This specific instance ranks as the highest attended object-specific token for the region. The notably high attention score suggests that the mere presence of the "Orange" token within the reasoning trace contributes strongly to the erroneous visual manifestation of the carrot in the generated image, overriding the intended spatial execution logic.

candidate generation, and contains only synthetic text prompts and program or code snippets; it does not include personal data, private information, or images of real individuals. The benchmark is designed for capability evaluation rather than deployment. The primary systems evaluated as generation models are publicly available open-source models; closed-source models are used only for data augmentation and evaluation robustness analyses. Its positive impact is to make reasoning-to-generation failures measurable and thereby support the development of more reliable multimodal systems. Potential negative impacts include benchmark gaming, overinterpreting benchmark scores as evidence of general generation safety or capability, and using improved generation fidelity for harmful content creation. Automated evaluators may also inherit model biases despite their high agreement with human judgments. Finally, because the benchmark currently uses English prompts and synthetic scenarios, its concepts may reflect culturally narrow assumptions and should not be treated as globally representative. We will release the data for research use with documentation of its intended diagnostic scope.

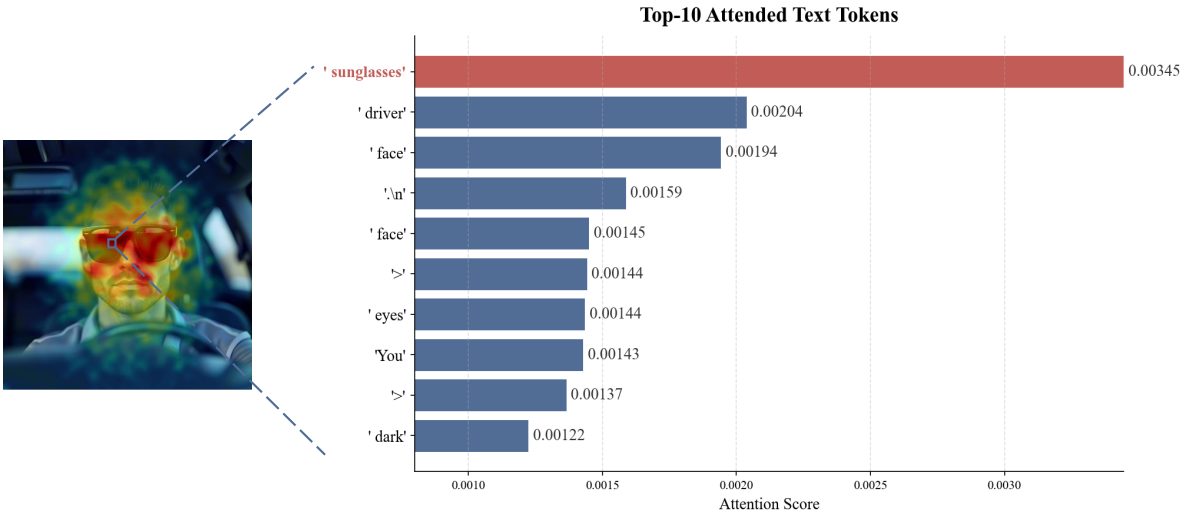


Figure 10: Top-10 attended textual tokens for the sunglasses region in the generated image, measured by aggregating attention scores from the corresponding visual generation tokens to all textual tokens. We compute the attention scores for the occurrence of the token "sunglasses" within R_t (highlighted in red in Figure 7). This specific instance ranks as the highest attended textual token for the sunglasses region, excluding special tokens. The disproportionately high attention score suggests that the mere presence of the "sunglasses" token within the reasoning trace contributes most strongly to the erroneous visual manifestation of sunglasses in the generated image, overriding the intended execution logic.

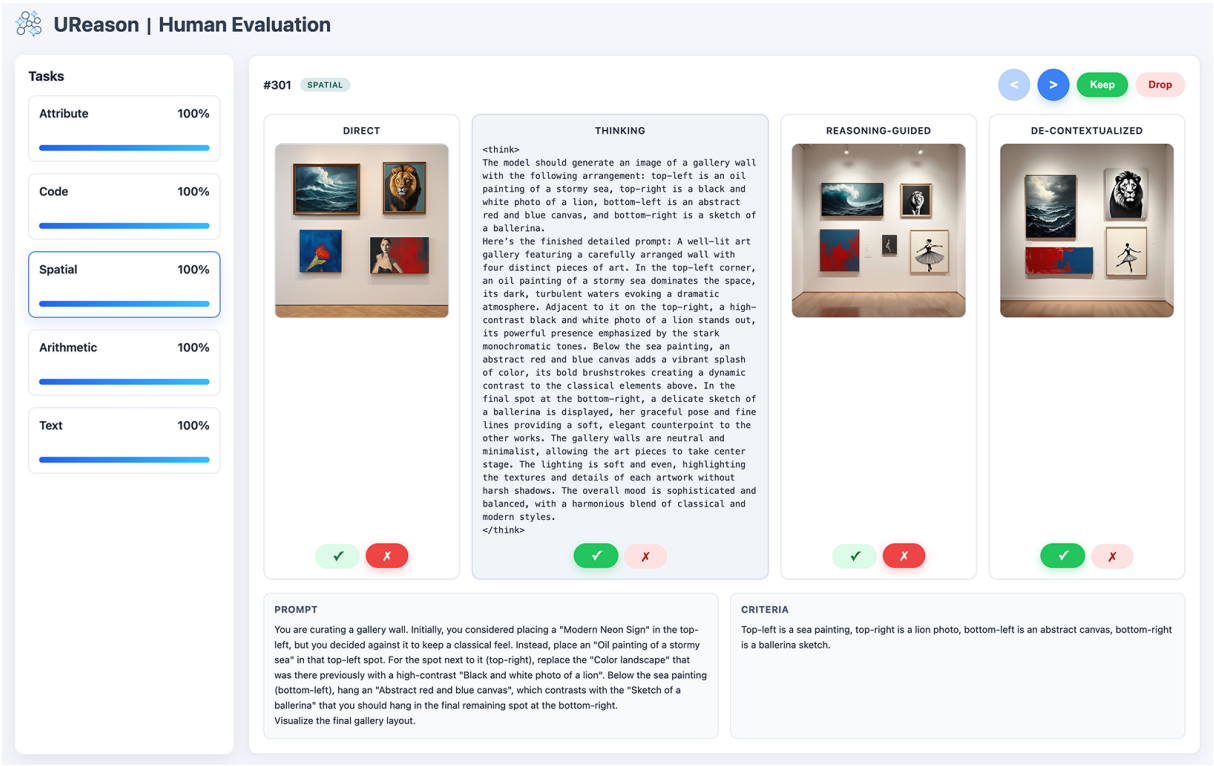


Figure 11: Screenshot of the interface used for human evaluation.

Model	Test Set	Setting	Code		Arithmetic		Spatial		Attribute		Text		Overall	
			Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ
Bagel	testmini	1	10.0	-	5.0	-	3.0	-	6.0	-	9.0	-	6.6	-
		2	30.0	+20.0	14.0	+9.0	15.0	+12.0	19.0	+13.0	11.0	+2.0	17.8	+11.2
		3	**58.0**	**+28.0**	**51.0**	**+37.0**	**58.0**	**+43.0**	**60.0**	**+41.0**	**86.0**	**+75.0**	**62.6**	**+44.8**
	test	1	12.0	-	8.0	-	6.0	-	5.0	-	10.0	-	8.2	-
		2	33.0	+21.0	18.0	+10.0	16.0	+10.0	21.0	+16.0	13.0	+3.0	20.2	+12.0
		3	**60.0**	**+27.0**	**54.0**	**+36.0**	**59.0**	**+43.0**	**63.0**	**+42.0**	**87.0**	**+74.0**	**64.6**	**+44.4**
UniCoT	testmini	1	12.0	-	3.0	-	6.0	-	12.0	-	8.0	-	8.2	-
		2	33.0	+21.0	18.0	+15.0	26.0	+20.0	21.0	+9.0	12.0	+4.0	22.0	+13.8
		3	**57.0**	**+24.0**	**42.0**	**+24.0**	**50.0**	**+24.0**	**42.0**	**+21.0**	**52.0**	**+40.0**	**48.6**	**+26.6**
	test	1	13.0	-	5.0	-	9.0	-	13.0	-	9.0	-	9.8	-
		2	37.0	+24.0	18.0	+13.0	31.0	+22.0	24.0	+11.0	15.0	+6.0	25.0	+15.2
		3	**57.0**	**+20.0**	**46.0**	**+28.0**	**54.0**	**+23.0**	**42.0**	**+18.0**	**57.0**	**+42.0**	**51.2**	**+26.2**
SRUM	testmini	1	11.0	-	4.0	-	3.0	-	6.0	-	7.0	-	6.2	-
		2	25.0	+14.0	8.0	+4.0	20.0	+17.0	24.0	+18.0	6.0	-1.0	16.6	+10.4
		3	**67.0**	**+42.0**	**50.0**	**+42.0**	**50.0**	**+30.0**	**50.0**	**+26.0**	**82.0**	**+76.0**	**59.8**	**+43.2**
	test	1	10.0	-	5.0	-	6.0	-	10.0	-	9.0	-	8.0	-
		2	27.0	+17.0	10.0	+5.0	23.0	+17.0	28.0	+18.0	10.0	+1.0	19.6	+11.6
		3	**69.0**	**+42.0**	**49.0**	**+39.0**	**53.0**	**+30.0**	**55.0**	**+27.0**	**85.0**	**+75.0**	**62.2**	**+42.6**
ThinkMorph	testmini	1	9.0	-	1.0	-	4.0	-	3.0	-	10.0	-	5.4	-
		2	19.0	+10.0	12.0	+11.0	15.0	+11.0	26.0	+23.0	5.0	-5.0	15.4	+10.0
		3	**49.0**	**+30.0**	**37.0**	**+25.0**	**45.0**	**+30.0**	**55.0**	**+29.0**	**71.0**	**+66.0**	**51.4**	**+36.0**
	test	1	12.0	-	3.0	-	5.0	-	5.0	-	13.0	-	7.6	-
		2	23.0	+11.0	16.0	+13.0	18.0	+13.0	27.0	+22.0	9.0	-4.0	18.6	+11.0
		3	**48.0**	**+25.0**	**35.0**	**+19.0**	**46.0**	**+28.0**	**53.0**	**+26.0**	**68.0**	**+59.0**	**50.0**	**+31.4**
UniCoT-v2	testmini	1	9.0	-	2.0	-	2.0	-	5.0	-	3.0	-	4.2	-
		2	27.0	+18.0	19.0	+17.0	29.0	+27.0	21.0	+16.0	21.0	+18.0	23.4	+19.2
		3	**65.0**	**+38.0**	**45.0**	**+26.0**	**46.0**	**+17.0**	**67.0**	**+46.0**	**87.0**	**+66.0**	**62.0**	**+38.6**
	test	1	10.0	-	4.0	-	3.0	-	6.0	-	4.0	-	5.4	-
		2	29.0	+19.0	20.0	+16.0	31.0	+28.0	23.0	+17.0	24.0	+20.0	25.4	+20.0
		3	**65.0**	**+36.0**	**48.0**	**+28.0**	**41.0**	**+10.0**	**68.0**	**+45.0**	**84.0**	**+60.0**	**61.2**	**+35.8**
Bagel-Zebra -CoT	testmini	1	7.0	-	7.0	-	2.0	-	10.0	-	5.0	-	6.2	-
		2	14.0	+7.0	10.0	+3.0	15.0	+13.0	23.0	+13.0	10.0	+5.0	14.4	+8.2
		3	**50.0**	**+36.0**	**43.0**	**+33.0**	**33.0**	**+18.0**	**48.0**	**+25.0**	**85.0**	**+75.0**	**51.8**	**+37.4**
	test	1	8.0	-	8.0	-	3.0	-	11.0	-	6.0	-	7.2	-
		2	16.0	+8.0	11.0	+3.0	17.0	+14.0	26.0	+15.0	13.0	+7.0	16.6	+9.4
		3	**52.0**	**+36.0**	**45.0**	**+34.0**	**35.0**	**+18.0**	**50.0**	**+24.0**	**86.0**	**+73.0**	**53.6**	**+37.0**
T2I-R1	testmini	1	3.0	-	6.0	-	4.0	-	9.0	-	2.0	-	4.8	-
		2	6.0	+3.0	4.0	-2.0	2.0	-2.0	11.0	+2.0	3.0	+1.0	5.2	+0.4
		3	**20.0**	**+14.0**	**15.0**	**+11.0**	**12.0**	**+10.0**	**27.0**	**+16.0**	**47.0**	**+44.0**	**24.2**	**+19.0**
	test	1	4.0	-	6.0	-	5.0	-	10.0	-	3.0	-	5.6	-
		2	7.0	+3.0	7.0	+1.0	3.0	-2.0	13.0	+3.0	5.0	+2.0	7.0	+1.4
		3	**22.0**	**+15.0**	**16.0**	**+9.0**	**14.0**	**+11.0**	**29.0**	**+16.0**	**49.0**	**+44.0**	**26.0**	**+19.0**
UniMoE2	testmini	1	5.0	-	4.0	-	2.0	-	10.0	-	4.0	-	5.0	-
		2	10.0	+5.0	3.0	-1.0	3.0	+1.0	12.0	+2.0	6.0	+2.0	6.8	+1.8
		3	**17.0**	**+7.0**	**13.0**	**+10.0**	**8.0**	**+5.0**	**21.0**	**+9.0**	**13.0**	**+7.0**	**14.4**	**+7.6**
	test	1	6.0	-	5.0	-	3.0	-	11.0	-	5.0	-	6.0	-
		2	11.0	+5.0	3.0	-2.0	4.0	+1.0	13.0	+2.0	7.0	+2.0	7.6	+1.6
		3	**19.0**	**+8.0**	**14.0**	**+11.0**	**10.0**	**+6.0**	**23.0**	**+10.0**	**15.0**	**+8.0**	**16.2**	**+8.6**

Table 11: Performance comparison on the full test set and its testmini subset for all eight models. Acc and Δ denote visual verification accuracy (%) and performance gain over the previous setting, respectively. 1, 2, and 3 represent Direct Generation, Reasoning-Guided Generation, and De-contextualized Generation, respectively.

Model	Training Length (# tokens)	UReason Eval Length (# tokens)
ThinkMorph	490.8 (276.7)	316.3 (154.8)
Bagel-Zebra-CoT	476.4 (474.4)	256.7 (123.4)

Table 12: Token length comparison between model training data and UReason evaluation contexts. Mean (Std.) reported. Prompt lengths are measured using the Qwen3-8B tokenizer.

Layers	P	R_t	R_p
1–7	2.45	4.57	6.93
8–14	1.71	3.32	5.33
15–21	0.12	0.21	0.43
22–28	0.08	0.12	0.34
1–28	**1.09**	**2.06**	**3.26**

Table 13: Average attention weights across layers for ThinkMorph during image generation, scaled by 10^{-4}.

Layers	P	R_t	R_p
1–10	2.57	3.56	4.01
11–20	1.17	2.19	2.73
21–30	1.08	1.83	4.21
1–30	**1.61**	**2.53**	**3.65**

Table 14: Average attention weights across layers for T2I-R1 during image generation, scaled by 10^{-4}.

Model	Image	Reasoning
UniCoT	0.924	0.962
Bagel	0.929	0.968
T2I-R1	0.931	0.972

Task (UniCoT)	Image	Reasoning
Code	0.913	0.950
Arithmetic	0.920	0.960
Spatial	0.937	0.940
Attribute	0.940	0.970
Text	0.910	0.990

Table 15: Agreement rates between automated evaluators and human judgments, reported across models (left) and across task categories for UniCoT (right).

Evaluation Prompt for Visual Verification Accuracy

You are an objective image evaluator. Your goal is to verify if the image content matches the provided text description.

Target Description: "`{description}`"

Please think step by step:

1. Analyze the image content carefully.
2. Compare the visual elements with the "Target Description".
3. Determine if the image strictly meets the requirements.

Finally, output your judgment in the following format:
If it matches, output <answer>Yes</answer>.
If it does not match, output <answer>No</answer>.

Figure 12: Evaluation prompt for visual verification accuracy.

Evaluation Prompt for the Quality of Reasoning Chain

You are an objective reading comprehension evaluator.
I will provide you with a "User Prompt", a model's "Thought Process" (which includes its intermediate thoughts and final refined prompt) and "Target Criteria".
Your task is to judge whether the final state or conclusion described in the model's thought process contains and satisfies a specific "Target Criteria".

=== User Prompt ===
`{user_prompt}`

=== Model Thought Process ===
`{model_thought}`

=== Target Criteria ===
`{criteria}`

=== Instruction ===
Please think step by step:

1. Analyze the "Target Criteria" to understand the specific visual or logical constraints required.
2. Read the entire "Thought Process" carefully.
3. Determine whether the "Target Criteria" is successfully met or clearly present in the outcome of the thought process.
    - Note: The text does not need to match the criteria word-for-word, but the specific semantic meaning and target state must be unambiguously present. Do not guess or assume unstated information.

Finally, output your judgment in the following format:
If the target criteria is clearly met or present in the text, output <answer>Yes</answer>.
If the criteria is missed or contradicted, output <answer>No</answer>.

Figure 13: Evaluation prompt for assessing the reasoning chain against target criteria.

Prompt

A wooden crate lid (flat surface) held 4 oranges. The farmer moved 3 oranges onto the grass next to the lid. Then, he picked up 1 orange from the grass and put it back on the lid. Visualize the wooden lid and the grass.

UMMs

Response

<think>Reasoning: The wooden crate lid initially holds 4 oranges. The farmer moves 3 oranges to the grass, leaving 1 orange on the lid. Then, he picks up the orange from the grass and puts it back on the lid. Therefore, the final count is 4 oranges on the lid.</think>



A wooden crate lid with 4 oranges resting on it, placed on a flat surface.

Refined Prompt

1. Analyze the "Target Criteria":
The core requirement is that, in the final state, there should be 2 oranges on the wooden lid and 2 oranges on the grass. This is a specific distribution of the oranges after all actions.

2. Read the "Thought Process" carefully:
- Initially, 4 oranges are on the lid.
- The farmer moves 3 oranges to the grass → lid: 1 orange, grass: 3 oranges.
- Then, he picks up 1 orange from the grass and puts it back on the lid → lid: 2 oranges, grass: 2 oranges.
However, the model's reasoning incorrectly states: "he picks up the orange from the grass" (singular, implying only one was on the grass), and concludes that the final count is 4 oranges on the lid, which is factually wrong.
The model says: after moving 3 to the grass (leaving 1 on lid), he "picks up the orange from the grass" — this phrasing suggests he picks up one orange from the grass (so grass now has 2, lid has 2), but then says he puts it back on the lid, which would make lid: 2, grass: 2.
But then the model concludes: "final count is 4 oranges on the lid" — which contradicts the actions described.
So, the model's logic is inconsistent: the actions described would lead to 2 on lid and 2 on grass, but the conclusion says 4 on the lid, which is only possible if all oranges were returned, which didn't happen.

3. Determine if the thought process covers the "Target Criteria":
The correct final state (2 on lid, 2 on grass) is not acknowledged in the reasoning. The model incorrectly concludes 4 on the lid, which directly contradicts the target criteria. Even if the intermediate step might accidentally align, the final reasoning does not reflect or support the target distribution.
Therefore, the thought process does not imply or cover the target criteria.

<answer>No</answer>

Evaluation

Figure 14: **An Illustrative Example of Reasoning Error**. This case shows where the model's intermediate reasoning process is incorrect. The incorrect reasoning steps are highlighted in red.

Prompt

```
<div id="display_case">
    <div id="tl">Gold Necklace</div>
    <div id="tr">Silver Ring</div>
    <div id="bl" onclick= "this. innerText='Pearl Bracelet'"> Diamond Earrings</div>
    <div id="br">Ruby Brooch</div>
</div>
<script>
    // Scenario: A customer buys the earrings, triggering a restock.
    // The click event changes the text content of the element.
    document.getElementById("bl").click();
</script>
```

Visualize a jewelry display box with four compartments (2x2).



Response

<think>The task is to create a visual description of a jewelry display case with four compartments arranged in a 2x2 grid. The compartments are as follows: the top-left is "Gold Necklace," the top-right is "Silver Ring," the bottom-left is "Diamond Earrings" (which changes to "Pearl Bracelet" after a click event), and the bottom-right is "Ruby Brooch." The JavaScript code provided indicates that clicking on the bottom-left compartment triggers a change in its content to "Pearl Bracelet." The visual description should reflect this setup, showing the compartments in their initial state and then the updated state after the click event. The image should be a simple, clean representation of the jewelry case, with each compartment clearly rendered.</think>

A 2x2 grid jewelry display case with four compartments. The top-left compartment is "Gold Necklace," the top-right is "Silver Ring," the bottom-left is "Pearl Bracelet," and the bottom-right is "Ruby Brooch."

Refined Prompt



Figure 15: **An Illustrative Example of Instruction Misinterpretation**. This case demonstrates a failure to grasp the semantic intent of the prompt. The model erroneously renders the code text itself rather than visualizing the target object described by the code.

Prompt

A gardener is planning a square bed. He follows a rule: "Root vegetables go Top-Right, and Leafy greens go Bottom-Left." Based on this, place "Rows of orange carrots" in the Top-Right and "Large green cabbages" in the Bottom-Left. The remaining corners are for flowers: put "Blooming yellow sunflowers" in the Top-Left to catch the morning sun, and "Red tulip flowers" in the last empty spot (Bottom-Right). Do not mix the vegetables with the flowers.
Visualize the garden bed from above.

UMMs

Response

<think>Alright, let's tackle this gardening plan. The gardener has a square bed with specific rules for where different plants go. First, I need to visualize the square and divide it into four corners. The top-right corner should be for root vegetables, and the bottom-left for leafy greens. Then the top-left corner is for sunflowers, and the bottom-right for tulips.

Starting with the top-right corner: root vegetables like carrots. So I'll place rows of orange carrots there. Next, the bottom-left corner needs leafy greens. Cabbages are a good choice, so I'll put large green cabbages there. Then the top-left corner is for sunflowers. They need to catch the morning sun, so that makes sense. Finally, the bottom-right corner is for tulips. Red tulips will look nice there.</think>

A square garden bed divided into four equal sections. The top-right section contains rows of orange carrots, the bottom-left section has large green cabbages, the top-left section is filled with blooming yellow sunflowers, and the bottom-right section features red tulip flowers. Each plant type is distinctly separated in its designated corner, adhering to the gardener's planting plan.

Refined Prompt



Figure 16: **An Illustrative Example of Concept Hallucination**. This case illustrates the generation of unprompted objects. The model spontaneously renders “yellow roses” despite them being absent from the input.

Prompt

```
<table border="1">
    <tr>
    <td>Dress</td>
    <td>Jeans</td>
    </tr>
    <tr>
    <td>Jacket</td>
    <td>Sneakers</td>
    </tr>
</table>
```

Visualize a fashion store window divided into four glass panes. Render the items defined inside the HTML `<td>` tags.

UMMs

Response

<think>The task is to transform a simple HTML table into a visually engaging shop window display. The table has two rows and two columns, so the window should be divided into four glass panes. Each pane will showcase one item from the table. The items are "Dress," "Jeans," "Jacket," and "Sneakers." The description should detail the layout and presentation of these items in the window, ensuring clarity and alignment with the given HTML structure.</think>

A modern fashion store window divided into four glass panes, each showcasing one item from a simple HTML table. The top left pane displays a stylish dress, the top right shows a pair of jeans, the bottom left features a sleek jacket, and the bottom right presents a pair of sneakers. The items are arranged neatly, with clean lines and a minimalist aesthetic, reflecting the structured layout of the HTML table. The window is framed with a subtle metallic border, and the overall design is bright and inviting, emphasizing the items' details and textures.

Refined Prompt



Figure 17: **An Illustrative Example of Task-Specific Error (CODE).** Although the model correctly identifies the 2x2 grid structure from the HTML prompt, it fails to map the specific items (Dress, Jeans, Jacket, Sneakers) to their corresponding cells, resulting in generation errors.

Prompt

A flat wooden table surface held 3 green apples. A person moved 2 apples into a white bowl sitting on the table. Then, they took 1 apple out of the bowl and placed it back on the table surface.
Visualize the table surface and the white bowl.

UMMs

Response

<think>Initially, there are three green apples on the table. The person moves two apples into the white bowl, leaving one apple on the table. Then, they take one apple from the bowl and place it back on the table. This results in two apples on the table and one apple in the bowl. The final count is two apples on the table and one apple in the bowl.</think>

A flat wooden table surface with two green apples placed on it and a white bowl containing one green apple.

Refined Prompt



Figure 18: **An Illustrative Example of Task-Specific Error (ARITHMETIC).** The refined prompt specifies "two apples on the table and one apple in the bowl". Although the reasoning trace is correct, the generated image displays three apples on the table, highlighting the challenge of translating precise quantitative tokens into exact object counts.

Prompt

A chef is making a pizza divided into four quarters (2x2). The customer wants a specific layout. Top-Left: "Pepperoni". Top-Right: "Mushrooms". Bottom-Left: The customer originally asked for "Pineapple", but changed their mind to "Olives" at the last minute. Bottom-Right: "Green Peppers".
Visualize the pizza toppings.

UMMs

Response

<think>Alright, let's tackle this pizza customization scenario. The chef has divided the pizza into four equal quarters, arranged in a 2x2 grid. The customer initially requested "Pineapple" for the bottom-left section but changed their mind to "Olives" at the last minute. The other sections are set: top-left is "Pepperoni," top-right is "Mushrooms," and bottom-right is "Green Peppers."

First, I need to visualize each quadrant clearly. The top-left quadrant should have pepperoni slices, which are typically round and red, scattered across the area. The top-right quadrant is for mushrooms, which would be depicted as small, brown, and irregularly shaped pieces. The bottom-left quadrant, originally pineapple but now olives, should show black olives, round and glossy, spread out. The bottom-right quadrant is green peppers, which are green, sliced into small, triangular pieces.

Now, considering the pizza's overall appearance, it should have a golden-brown crust with a slightly charred edge, indicating it's been baked. The cheese should be melted and slightly browned in spots, with a smooth texture. The toppings should be evenly distributed but not overcrowded, leaving some visible cheese in between. The colors of the toppings should contrast well with the cheese and crust, making each section distinct.</think>

A freshly baked pizza divided into four equal 2x2 quadrants, each featuring distinct toppings. The top-left quadrant is generously covered with round, red pepperoni slices, the top-right quadrant is dotted with small, brown mushrooms, the bottom-left quadrant showcases black olives spread evenly, and the bottom-right quadrant is filled with green, triangular pepper slices. The pizza has a golden-brown crust with slightly charred edges, melted cheese with a smooth, slightly browned surface, and a subtle sheen. The background is a neutral wooden table, and the lighting highlights the textures and colors of the toppings, creating an appetizing and visually appealing image.

Refined Prompt



Figure 19: **An Illustrative Example of Task-Specific Error (SPATIAL).** While the prompt specifies four quadrants with distinct toppings, the model fails to maintain strict boundary separation. It blends the instructions into a generic pizza image rather than distributing the toppings correctly across the requested regions.

Prompt

A clear plastic cup was filled with brown iced coffee and ice cubes. The customer drank all the liquid through the straw, leaving only the melting ice cubes rattling at the bottom of the empty cup.
Visualize the cup strictly based on the finished drink.

UMMs

Response

<think>Alright, let's tackle this. The task is to visualize a clear plastic cup filled with brown iced coffee and ice cubes, then describe and render the cup after the customer drank all the liquid, leaving only the melting ice cubes at the bottom.

I need to focus on the cup itself. The ice cubes should be scattered at the bottom of the cup, with some of them partially melted, creating a small pool of water. The melting process should be subtle but noticeable, indicating that the ice has been there for a while after the drink was finished. The cup should be upright, and the lighting should highlight the transparency of the plastic and the wetness of the ice cubes.</think>

A clear plastic cup with a transparent, slightly glossy finish, standing upright. Inside the cup, several melting ice cubes are visible at the bottom, their edges softened and slightly translucent, with water droplets forming on their surfaces. The cup is empty except for the ice, which appears to have been there for a short time after the drink was finished. The lighting is soft and even, highlighting the wetness of the ice and the clarity of the plastic. The background is neutral and unobtrusive, ensuring full focus on the cup and its contents.

Refined Prompt



Figure 20: **An Illustrative Example of Task-Specific Error (ATTRIBUTE).**Despite the prompt explicitly specifying the cup as "empty except for the ice", the model generates a brown liquid. This error likely stems from contextual interference, where the model's priors dilute the strict attribute constraint.

Prompt

A technician is labeling a glass case for an emergency tool. The source code is 'FIREWALL', but the safety standard requires only the first four letters to be printed on the glass.
Visualize a red emergency glass box.

UMMs

Response

<think>Okay, let's tackle this task. The user wants me to generate an image of a glass case with an emergency label that truncates the source code 'FIREWALL' to the first four letters. The safety standard requires only the first four letters to be printed on the glass.

First, I need to understand the requirements clearly. The source code is 'FIREWALL', and we need to display the first four letters, which are 'FIRE'. The glass case should be red, as that's a common color for emergency equipment. The label should be printed on the glass, not on a separate sticker or label. The glass case itself might be a box or a cabinet, but the exact shape isn't specified, so I can choose a simple rectangular box for clarity.</think>

A red glass emergency box with a transparent front panel displaying the word "FIRE" in bold, uppercase letters centered on the glass. The box has a simple rectangular design with a slightly reflective surface, and the red color is vibrant and uniform. The label is printed directly on the glass, ensuring high visibility and clarity. The overall scene is clean and professional, with no additional elements or distractions.

Refined Prompt



Figure 21: **An Illustrative Example of Task-Specific Error (TEXT).** The prompt requests the text "FIRE", but the model generates "EIME". Despite the correct string being present in the refined prompt, the UMM fails to render the characters accurately.



Figure 22: Qualitative examples for CODE.

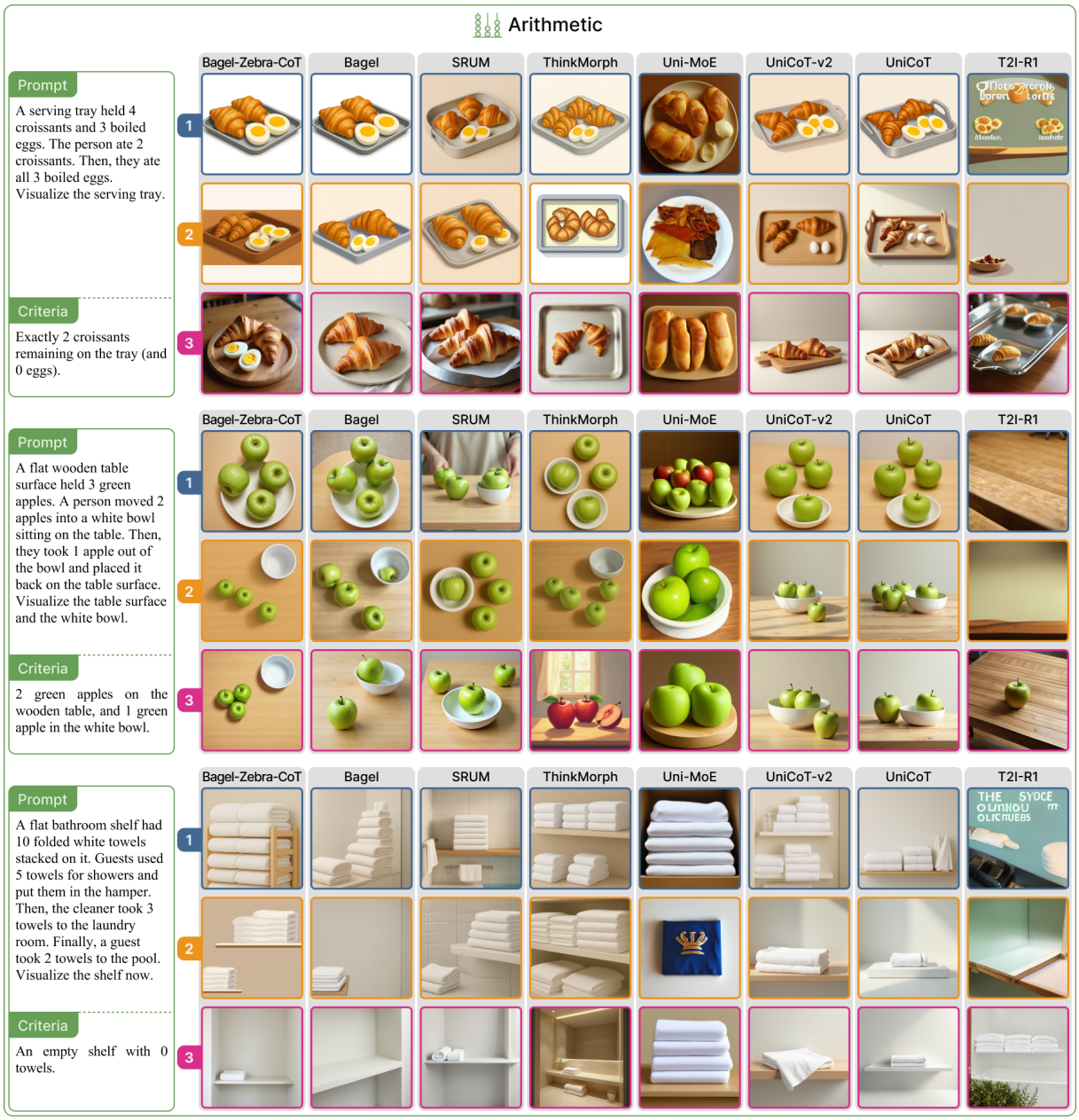


Figure 23: Qualitative examples for ARITHMETIC



Figure 24: Qualitative examples for SPATIAL



Figure 25: Qualitative examples for ATTRIBUTE.

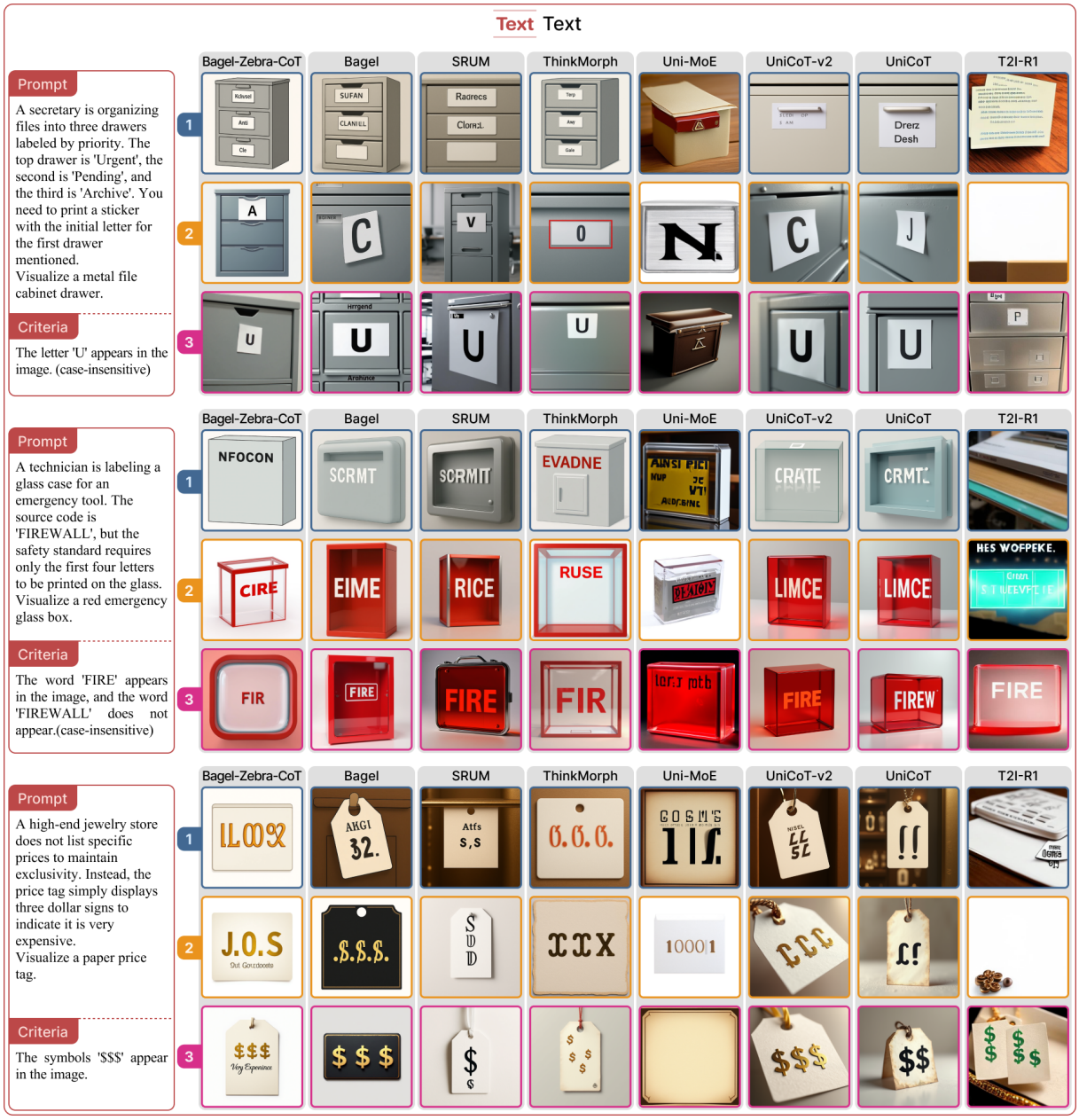


Figure 26: Qualitative examples for TEXT.

Model Name	Hugging Face Repository
Bagel	https://huggingface.co/ByteDance-Seed/BAGEL-7B-MoT
UniCoT	https://huggingface.co/Fr0zencr4nE/UniCoT-7B-MoT
UniCoT-v2	https://huggingface.co/Fr0zencr4nE/UniCoT-7B-MoT-v0.2
SRUM	https://huggingface.co/Wayne-King/SRUM_BAGEL_7B_MoT
Bagel-Zebra-CoT	https://huggingface.co/multimodal-reasoning-lab/Bagel-Zebra-CoT
ThinkMorph	https://huggingface.co/ThinkMorph/ThinkMorph-7B
T2I-R1	https://huggingface.co/CaraJ/T2I-R1
UniMoE2	https://huggingface.co/HIT-TMG/Uni-MoE-2.0-Image
Qwen2.5-7B	https://huggingface.co/Qwen/Qwen2.5-7B-Instruct
Qwen3-8B	https://huggingface.co/Qwen/Qwen3-8B
Qwen3-235B-A22B	https://huggingface.co/Qwen/Qwen3-235B-A22B-Instruct-2507
Qwen3-VL-235B-A22B	https://huggingface.co/Qwen/Qwen3-VL-235B-A22B-Instruct

Table 16: List of models and their Hugging Face repositories.