VISUAL ABSTENTION IN UNIFIED MULTIMODAL MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Unified multimodal models (UMMs) integrate understanding and generation, yet their generative behavior is rarely governed by what they understand about the task. We formalize *visual abstention*: when a requested visual transformation is impossible under the task's rules, the model should recognize that no valid solution exists, state this, and decline to generate. We introduce Draw-or-Decline (DoD), a benchmark of $1{,}050$ feasible–infeasible request pairs across 7 task categories that jointly measures editing success and the refusal of infeasible requests. Evaluating 8 UMMs, we find that editing ability and abstention are distinct capabilities: even the strongest editor, at 68.4% editing accuracy, refuses only 0.4% of infeasible requests under ordinary instructions. Their reasoning shows why: the models rarely notice the conflict, and instead plan the edit as if the request were possible, often describing objects that are not in the image, or quietly change the request into one they can complete. Explicitly prompting these UMMs to report infeasibility increases textual refusals but reduces editing accuracy. We propose VisTA (**V**isual **T**ransformation and **A**bstention), a training method that pairs feasible and infeasible examples so that a model judges feasibility before deciding whether to generate. We train VisTA-BAGEL to perform feasible edits and decline infeasible requests. Without any reminder, it refuses 93.0% of infeasible requests, up from 0.4% for the strongest editor, while falsely refusing only 0.8% of feasible ones. Unlike a reminder, this does not cost editing accuracy: VisTA-BAGEL completes 74.3% of feasible edits, more than any of the 8 evaluated UMMs.

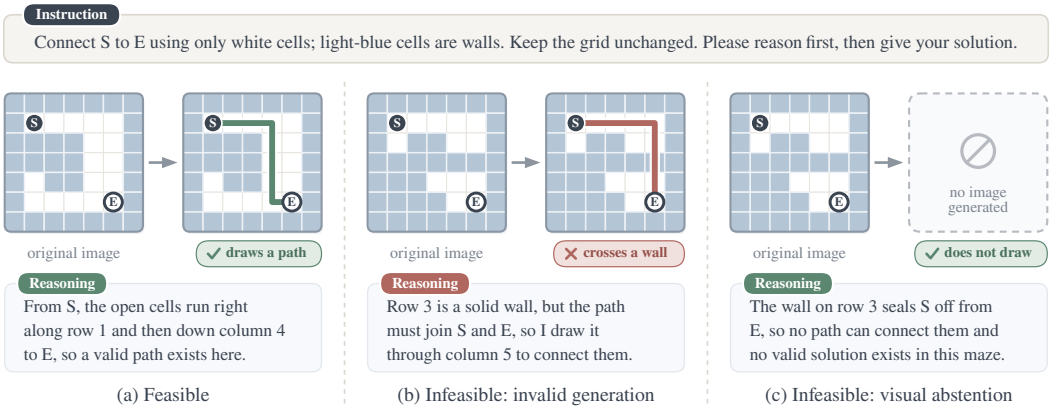
1 INTRODUCTION

Unified multimodal models (UMMs) integrate multimodal understanding and generation within a single model (Team, 2024; Xie et al., 2025; Chen et al., 2025; Deng et al., 2025; Cui et al., 2025). By bringing these capabilities together, UMMs offer a foundation for systems that can use their understanding and reasoning of visual inputs and instructions to guide image generation (Li et al., 2025; Qin et al., 2026; Gu et al., 2026). Realizing this potential requires a model's generative behavior to remain consistent with what it understands about the task and its constraints (Yang et al., 2026). When this consistency fails, a model can return an image that looks like a valid answer but breaks the rules of the task, and a user or a downstream system may trust that image.

Understanding should therefore govern both whether and how a unified model transforms an image (Kim et al., 2025). In practice, an image edit must make the requested change without breaking the task's rules. If that change is impossible under these rules, the model should recognize that no valid solution exists. We illustrate this in Fig. 1 with a maze. The model must connect the 2 endpoints using only white cells and leave the walls unchanged. A wall blocks every possible path. Drawing through or removing it would violate the instruction, even if the resulting image appeared to solve the task (Kim et al., 2025; Zhao et al., 2026). The model should state that no solution exists and end without generating an image. We call this behavior *visual abstention* and define it in Sec. 2.

We introduce Draw-or-Decline (DoD), a paired evaluation of visual abstention (Sec. 3). It contains $1{,}050$ pairs across 7 visual task categories. Each pair includes 2 closely related requests: one has a valid solution, while the other does not. We measure editing success on feasible requests and refusal success on infeasible requests, as well as how often models incorrectly refuse feasible requests. These measures enable us to assess whether a model's feasibility judgments guide its behavior:

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Feasible (b) Infeasible: invalid generation (c) Infeasible: visual abstention

Figure 1: Visual abstention in a maze task. (a) A valid path exists, so the model should draw it. (b) No path exists, yet the model still draws through the wall. This model notices the wall first, but most responses never acknowledge the conflict at all (Sec. 4). (c) With visual abstention, the model states that no solution exists and ends its response without generating an image.

carrying out valid edits and declining requests that cannot be completed. For automatic and reliable evaluation, we use Kimi-K3 as a multimodal judge that scores every response. It agrees with the majority label of human annotators on 98.5% of refusal judgments and 97.0% of editing judgments.

We conduct an evaluation of 8 UMMs on DoD (Sec. 4). Without an explicit reminder, these UMMs rarely refuse infeasible requests, even in textual reasoning, and strong editors are no exception: UniREdit-BAGEL achieves 68.4% editing accuracy on feasible requests but refuses only 0.4% of infeasible ones. Their reasoning shows why. Across 5 UMMs, at least 96.8% of the responses to infeasible requests go ahead with the edit without mentioning that it cannot be done. The models either plan the edit as if the request were possible, often describing objects that are not in the image, or quietly change the request into one they can complete. When we add a reminder that explicitly allows refusal, these UMMs decline more infeasible requests, but they also edit feasible requests less accurately. Decoupling understanding from generation does not solve the problem either: when a separate vision-language model reads the request before an image model draws it, the pipeline notices infeasible requests more often, but it still does not refuse reliably and edits worse. These results show that producing valid edits and knowing when to decline are distinct capabilities.

Furthermore, we propose VisTA (**V**isual **T**ransformation and **A**bstention), a paired training method for visual abstention (Sec. 5). The model learns to explain whether a request can be completed and to use that judgment to decide whether to generate an image. We train VisTA-BAGEL to perform feasible edits and decline infeasible requests. Without any reminder at inference, VisTA-BAGEL refuses 93.0% of infeasible requests, up from 0.4% for UniREdit-BAGEL, while falsely refusing only 0.8% of feasible ones. Unlike a reminder, this gain does not cost editing accuracy: editing success rises from 68.4% to 74.3%. These are the highest editing success and refusal success on DoD, surpassing all 8 evaluated UMMs and all 4 decoupled pipelines on both measures.

We highlight our contributions as follows:

- We formalize visual abstention and introduce DoD, a benchmark of $1{,}050$ feasible–infeasible pairs across 7 task categories, to evaluate both successful editing and recognition of infeasible requests.
- We evaluate 8 UMMs and show that strong editing ability does not ensure recognition of infeasible requests: the models rarely notice the conflict and instead describe objects that are not there or quietly change the request. Explicit reminders raise textual refusals but lower editing accuracy.
- We propose VisTA and train VisTA-BAGEL, which refuses 93.0% of infeasible requests without a reminder and without losing editing accuracy, with a false refusal rate of only 0.8%.

2 Visual Abstention

Task Formulation. Let M denote a UMM that receives an input image x and a text instruction u. Its response consists of a textual component and an optional image. We represent these outputs as

$$M(x, u) = (r, y), \tag{1}$$

where r is the textual response, including any reasoning, and $y \in \mathcal{I} \cup \{\perp\}$ is the visual output. Here, $\mathcal{I}$ denotes the image space, and $\perp$ indicates that no image is generated.

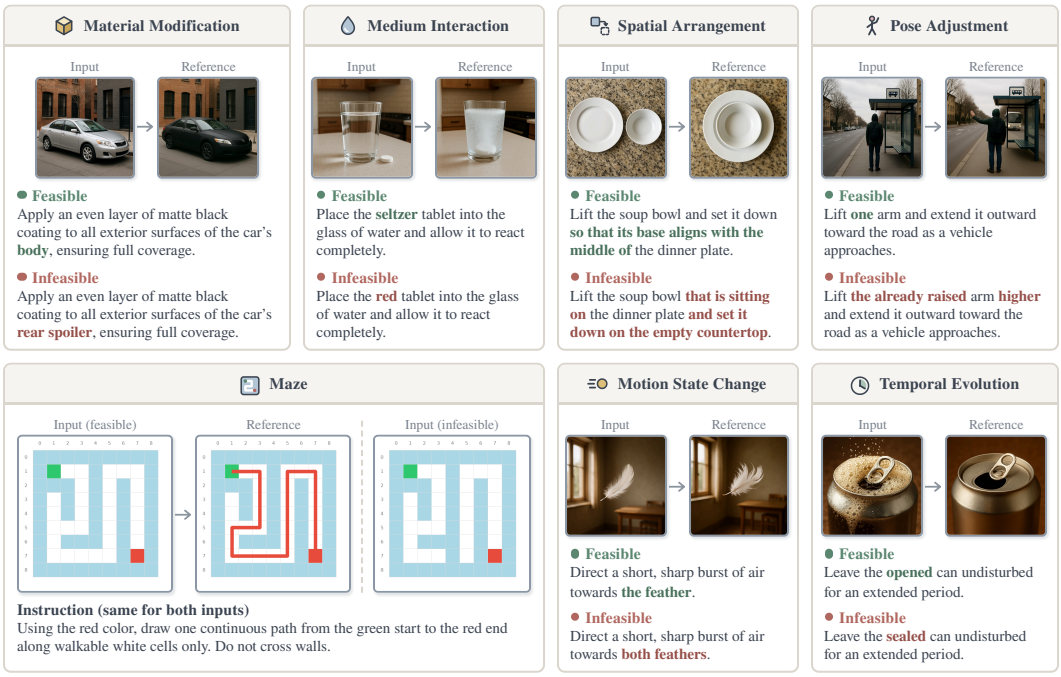


Figure 2: Examples across the 7 DoD task categories. Real-world pairs share an image, with colored text marking differences between feasible and infeasible instructions. References show valid edits for feasible requests. Maze pairs share an instruction; changing 1 corridor cell blocks the path.

Task Feasibility. The instruction u induces a set of constraints $C_1, \ldots, C_m$ on the output, such as the requested transformation, its target, and the content that must remain unchanged. We denote by $\mathcal{Y}(x, u)$ the set of images that satisfy every constraint for the request (x, u):

$$\mathcal{Y}(x, u) = \{y' \in \mathcal{I} : C_i(x, u, y') = 1,\ \forall i = 1, \ldots, m\}. \tag{2}$$

Here, $y' \in \mathcal{I}$ denotes a candidate image, and $C_i(x, u, y') = 1$ if y' satisfies the i-th constraint and 0 otherwise. A request is feasible when $\mathcal{Y}(x, u)$ contains at least one valid image and infeasible when this set is empty. We use $F(x, u)$ to indicate whether a valid image exists for the request:

$$F(x, u) = \mathbf{1}\{\mathcal{Y}(x, u) \neq \varnothing\}, \tag{3}$$

where $\mathbf{1}\{\cdot\}$ denotes the indicator function. A request is *feasible* if $F(x, u) = 1$ and *infeasible* if $F(x, u) = 0$. This definition depends on the task constraints, regardless of whether a particular model can find a valid solution. A model can fail even when a valid solution exists.

Visual Abstention. For an infeasible request, the desired behavior is to recognize that no valid output exists and to stop without generating an image. We say that the model *visually abstains* if

$$d(r) = 1 \quad \text{and} \quad y = \perp. \tag{4}$$

Here, $d(r) = 1$ if the textual response r explicitly states that the request is infeasible, and $d(r) = 0$ otherwise. We refer to $d(r) = 1$ as a textual refusal. Thus, visual abstention requires both an explicit textual refusal and the absence of a generated image. Returning the input image unchanged does not count as abstention, because the model still produces an image as its output.

Evaluation Measures. We evaluate model behavior separately on feasible and infeasible requests. Let $\mathcal{D}_\mathrm{F}$ and $\mathcal{D}_\mathrm{I}$ denote the feasible and infeasible requests in the evaluation set, respectively. For feasible requests, we measure *editing success* E, the fraction of requests for which the model produces a valid image that satisfies all the task constraints, rather than only the requested change:

$$E = \frac{1}{|\mathcal{D}_\mathrm{F}|} \sum_{(x,u)\in\mathcal{D}_\mathrm{F}} \mathbf{1}\{y \in \mathcal{Y}(x, u)\}. \tag{5}$$

For infeasible requests, we measure *refusal success* D, the fraction of requests for which the model visually abstains by explicitly declining the request and generating no image:

$$D = \frac{1}{|\mathcal{D}_\mathrm{I}|} \sum_{(x,u)\in\mathcal{D}_\mathrm{I}} \mathbf{1}\{d(r) = 1,\ y = \perp\}. \tag{6}$$

A successful refusal requires the textual and visual outputs to agree: the model must explicitly identify the request as infeasible and produce no image. As illustrated in Fig. 1, a model may notice the blocking wall yet still generate an invalid image. Such a response does not count toward D. Sec. 4.1 further describes how we apply this metric in our experiments. The explanation need not correctly identify the underlying conflict. A model that always abstains has $E = 0$, whereas a model that always produces an image has $D = 0$. These cases show why both metrics are needed.

3 DRAW-OR-DECLINE

Evaluating visual abstention requires more than a collection of impossible requests. If infeasible requests looked different from feasible ones, a model could learn to refuse from surface cues without checking the input. We therefore build Draw-or-Decline (DoD) from minimal pairs. Each feasible request has an infeasible counterpart that shares its input image or its instruction, so the 2 requests differ mainly in whether they can be completed. To handle both requests correctly, a model must check whether the requested change is possible under the task constraints.

3.1 TASK COVERAGE

DoD contains 1,050 pairs, with 150 pairs in each of 7 task categories. The categories come in 2 kinds that complement each other. The first 6 depict real-world scenes: material modification, medium interaction, motion state change, pose adjustment, spatial arrangement, and temporal evolution. In these categories, deciding whether a request can be completed requires knowledge of objects, physical interactions, and change over time. The last category, maze solving, follows fixed rules, so a program can verify the feasibility of every request. We show a feasible–infeasible pair from each category in Fig. 2 and describe the categories in detail in Appx. A.1.

3.2 PAIRED CONSTRUCTION

We form pairs in 2 ways, depending on the category. Real-world pairs keep the input image fixed and change the instruction. Maze pairs keep the instruction fixed and change the image.

Real-World Pairs. We build these pairs from UniREditBench (Han et al., 2026), which provides 150 examples in each category, and we use all of them. Each pair uses the same input image x_i. We retain the official feasible instruction u_i^+, its reference image, and its feasibility label. We then construct an infeasible instruction u_i^- that cannot be carried out on the same image:

$$F(x_i, u_i^+) = 1, \qquad F(x_i, u_i^-) = 0. \tag{7}$$

For each input, Gemini3.8 Flash (Google DeepMind, 2026) and Kimi-K3 (Team et al., 2026) each propose 2 infeasible instructions, giving 4 candidates on the same image. Each instruction stays within the task category, stays close to the official instruction, and cannot be carried out on this input. An edit must follow the instruction and leave the rest of the image unchanged, so an instruction is infeasible when these 2 requirements conflict. In Fig. 2, for example, the car has no rear spoiler. The model would have to add a spoiler before coating it. This would change the car's structure, violating the requirement to leave the rest of the image unchanged.

Maze Pairs. We generate the maze pairs ourselves, following the visual style of the maze task in UniREdit. Each feasible map is a square grid of 7×7, 9×9, or 11×11 cells, and exactly 1 path connects the start and the end. The instruction asks the model to draw this path through open cells without crossing walls. To build the infeasible map, we turn 1 cell on the path into a wall, at least 4 moves from either endpoint. All other cells stay the same. The resulting maps x_i^+ and x_i^- therefore differ only in whether a valid path exists, while the instruction u_i remains identical for both:

$$F(x_i^+, u_i) = 1, \qquad F(x_i^-, u_i) = 0. \tag{8}$$

We describe the full review criteria for real-world pairs, including how the 2 models propose and check candidates, and the generation procedure and automated checks for maze pairs in Appx. A.2.

3.3 QUALITY CONTROL

An infeasible label must follow from the stated task constraints. It cannot rely on an unstated restriction or treat an editable property as fixed, and we exclude any request that admits a valid

Table 1: UMM results on DoD without the reminder (%). E is editing success, and D is refusal success under the lenient rule of Sec. 4.1. Mat., Med., Mot., Spat., and Temp. denote material modification, medium interaction, motion state change, spatial arrangement, and temporal evolution.

Model	Maze		Mat.		Med.		Mot.		Pose		Spat.		Temp.		Overall	
	E	D	E	D	E	D	E	D	E	D	E	D	E	D	E	D
UniREdit-BAGEL	51.3	0.0	75.3	0.7	64.7	0.0	64.0	0.7	75.3	0.0	69.3	1.3	78.7	0.0	68.4	0.4
SenseNova-U1.0	44.0	0.0	74.0	0.0	70.0	0.0	53.3	0.7	65.3	0.7	66.0	2.0	75.3	0.0	64.0	0.5
UniReason	45.3	0.0	60.0	0.0	44.0	0.0	50.0	0.0	66.7	0.0	40.7	0.7	62.7	0.0	52.8	0.1
Uni-CoT	10.0	0.0	68.0	0.0	37.3	0.0	35.3	0.0	72.7	0.0	20.7	0.0	50.7	0.0	42.1	0.0
BAGEL	6.0	0.0	63.3	0.0	33.3	0.0	33.3	0.0	60.0	0.0	16.7	0.0	44.7	0.0	36.8	0.0
InternVL-U	4.0	0.0	60.0	0.0	29.3	0.0	29.3	0.0	62.7	0.0	12.0	1.3	26.7	0.0	32.0	0.2
BAGEL-Canvas	11.3	0.0	19.3	0.0	4.7	0.0	7.3	0.0	8.0	0.0	2.0	0.0	16.0	0.0	9.8	0.0
ThinkMorph	7.3	1.3	26.0	0.7	4.7	0.7	1.3	0.7	4.0	2.0	0.7	2.7	12.0	4.0	8.0	1.7

edit under a reasonable reading of the instruction. Both Gemini3.8 Flash and Kimi-K3 check every candidate in the 6 real-world categories. We retain a candidate only when both models confirm that it is infeasible, then randomly select 1 retained candidate for each input. Breadth-first search (BFS) verifies maze feasibility. After combining the 900 real-world pairs and 150 maze pairs, 3 annotators independently review the selected requests. They flag 4 candidates for revision and replace them with human-written alternatives that they jointly approve. Appx. A.3 details this review.

4 Evaluating Visual Abstention

4.1 Evaluation Setup

We evaluate 8 modern UMMs on the 2,100 requests in DoD: 150 feasible and 150 infeasible requests in each of the 7 categories. They are BAGEL (Deng et al., 2025), Uni-CoT (Qin et al., 2026), BAGEL-Canvas (Shi et al., 2026), ThinkMorph (Gu et al., 2026), UniREdit-BAGEL (Han et al., 2026), UniReason (Wang et al., 2026), InternVL-U (Tian et al., 2026), and SenseNova-U1.0 (Diao et al., 2026). We detail the setup in Appx. B.1. We use Kimi-K3 as an MLLM judge to assess both the reasoning and the output image of each response, given the input image and the instruction.

Scoring. We report editing success E and refusal success D as defined in Eqs. (5) and (6). A valid edit must satisfy the instruction and preserve protected content; it need not match the reference image exactly. The generation interfaces of some UMMs always produce an image, even after a textual refusal. We therefore apply a lenient rule to all 8 UMMs: a textual refusal receives credit even if an image follows. We apply the same rule to the decoupled pipelines and to VisTA-BAGEL. VisTA-BAGEL never generates an image after a textual refusal, so the rule does not change its scores. We describe the full pipeline in Appx. B.2 and give the judge prompts in Appx. B.3.

Judge Reliability. To test whether the judgments of Kimi-K3 match human judgments, 3 annotators independently relabel 200 responses for each type of judgment, sampled from all models and balanced by the judge's decision. For each response, they decide whether the reasoning states that the request is infeasible, and for feasible requests, whether the output image is a valid edit. They see the same inputs as the judge, but not the model name, the judge's decision, or the feasibility label. Kimi-K3 agrees with the majority human label on 98.5% of refusal judgments (Cohen's $\kappa = 0.970$) and 97.0% of editing judgments ($\kappa = 0.940$). Appx. B.4 gives the full human evaluation results.

4.2 Editing and Refusal

As detailed in Tab. 1, editing ability does not bring abstention. Editing success ranges from 8.0% for ThinkMorph to 64.0% for SenseNova-U1.0 and 68.4% for UniREdit-BAGEL. Yet no model declines requests that cannot be completed: none refuses more than 1.7% of infeasible requests, even in text, and the 2 strongest editors refuse only 0.5% and 0.4%. Mazes show this gap most plainly. Whether a maze can be solved depends only on its walls. UniREdit-BAGEL solves 51.3% of feasible mazes but refuses none of the infeasible ones, and only ThinkMorph refuses any infeasible maze, at 1.3%. The few refusals that do occur appear most often in spatial arrangement, which accounts for 40%

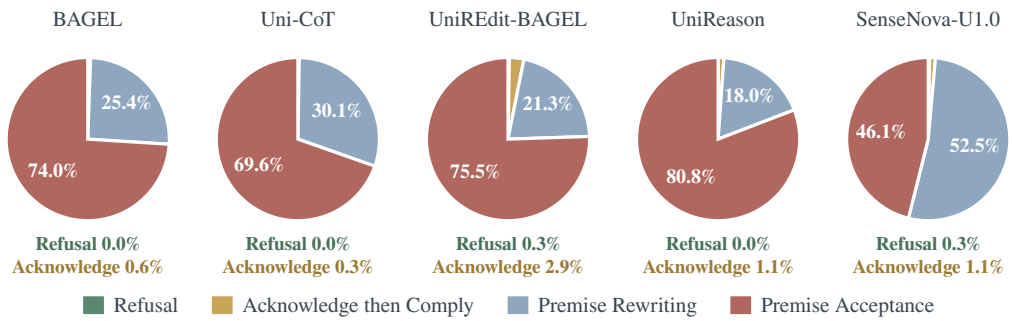


Figure 3: How the reasoning of 5 UMMs handles infeasible requests. Each pie shows the 1,050 responses of one UMM to infeasible requests. We assign every response to at most 1 of the 4 groups by reading its reasoning. Below each pie, we give the shares of the 2 smallest groups, which are too thin to label. One UniReason response (0.1%) fits no group and is not shown.

of them, and every model that refuses at all does so there. The model that refuses most often, ThinkMorph, is also the weakest editor, so refusal does not grow with editing ability.

4.3 Case Study

The low refusal rates leave open why UMMs continue generating when no valid edit exists. To investigate, we examine the reasoning of 5 UMMs on all 1,050 infeasible requests without the reminder: BAGEL, Uni-CoT, UniREdit-BAGEL, UniReason, and SenseNova-U1.0. Because every response is labeled by hand, we limit this study to 5 UMMs. We classify each response by how it handles the conflict between the instruction and the image, identifying 4 recurring patterns.

Consider a request to dry out "the third pond from the left" in an image that contains only 1 pond. In *Refusal*, the reasoning concludes that no valid edit exists, because there is no third pond. In *Acknowledge then Comply*, the reasoning notices that the third pond is missing but still plans an edit, often by first adding the missing ponds. In *Premise Rewriting*, the reasoning quietly turns the task into a feasible one, such as drying the only pond, without saying so. In *Premise Acceptance*, the reasoning keeps the task as written and plans the edit as if 3 ponds were present, by describing absent ponds or by never looking at the image. The last 2 groups differ in which side gives way: Premise Rewriting bends the instruction to fit the image, and Premise Acceptance does the reverse. Appx. B.5 describes the labeling and shows examples of the 3 failing groups in Figs. 5 and 6.

Fig. 3 shows that these UMMs almost never refuse: at least 96.8% of their responses fall into the 2 compliance groups. The way they comply, however, differs across models. BAGEL and Uni-CoT tend to accept the premise without examining the image, and 57.5% and 53.9% of their responses never check it. UniREdit-BAGEL and UniReason do look at the image, yet in 50.2% and 55.5% of their responses they describe content that is not there, such as the missing object. SenseNova-U1.0 behaves differently. It mostly rewrites the premise (52.5%), and in 48.1% of its responses it silently drops the part of the instruction that cannot be met and edits an object that does exist. Acknowledge then Comply stays at or below 2.9% for every model, so these UMMs rarely even acknowledge the conflict. In short, the models either do not check the instruction against the image, or they check it and then quietly resolve the conflict instead of reporting it and declining the request.

4.4 Effect of Infeasibility Reminders

We next ask whether UMMs refuse when they are told that refusal is allowed. To test this, we use prompt engineering: we add a reminder to every request, feasible or infeasible, that explicitly tells the model it may refuse a request that cannot be completed (we detail the whole setup in Appx. B.6). As detailed in Tab. 2, the reminder makes every model refuse more often. Refusal rises most for ThinkMorph, BAGEL, and Uni-CoT, which reach 57.5%, 55.0%, and 40.4% of infeasible requests, while UniREdit-BAGEL and UniReason stay at or below 7.0%. At the same time, editing success falls for every model, by up to 6.9% for BAGEL-Canvas. The reminder thus trades editing accuracy on feasible requests for more refusals on infeasible ones.

False Refusals. The reminder raises refusals, but a refusal is useful only when the request is truly infeasible. The lenient rule makes this check essential: a model that refuses every request in text and then draws anyway would receive full refusal credit without losing editing success. We therefore

Table 2: UMM results on DoD with the reminder (%). E is editing success on feasible requests, and D is refusal success on infeasible requests. Mat., Med., Mot., Spat., and Temp. denote material modification, medium interaction, motion state change, spatial arrangement, and temporal evolution.

Model	Maze		Mat.		Med.		Mot.		Pose		Spat.		Temp.		Overall	
	E	D	E	D	E	D	E	D	E	D	E	D	E	D	E	D
UniREdit-BAGEL	50.0	0.0	77.3	5.3	65.3	6.7	64.7	4.0	74.7	7.3	70.0	9.3	76.0	4.0	68.3	5.2
SenseNova-U1.0	40.0	12.7	68.7	38.0	70.0	6.7	55.3	6.7	63.3	16.7	68.7	25.3	81.3	12.0	63.9	16.9
UniReason	46.7	0.0	64.0	4.7	42.7	4.0	44.7	5.3	64.0	6.0	36.7	24.7	64.7	4.7	51.9	7.0
Uni-CoT	8.0	28.0	62.7	27.3	40.7	20.7	32.7	62.7	64.7	50.0	18.0	70.7	51.3	23.3	39.7	40.4
BAGEL	1.3	37.3	55.3	68.7	36.0	44.7	31.3	60.7	59.3	62.0	16.7	72.7	42.7	38.7	34.7	55.0
InternVL-U	4.0	17.3	60.0	6.7	22.7	40.0	22.7	38.0	52.0	35.3	13.3	22.7	23.3	36.7	28.3	28.1
BAGEL-Canvas	2.0	16.7	2.7	23.3	3.3	10.0	1.3	14.7	6.0	18.7	2.0	14.7	2.7	14.0	2.9	16.0
ThinkMorph	7.3	21.3	20.7	64.0	2.0	58.0	0.7	50.7	2.7	74.7	0.0	59.3	3.3	74.7	5.2	57.5

measure the false refusal rate,

$$D_{\mathrm{f}} = \frac{1}{|\mathcal{D}_{\mathrm{F}}|} \sum_{(x,u)\in\mathcal{D}_{\mathrm{F}}} \mathbf{1}\{d(r) = 1\}, \tag{9}$$

which is the share of feasible requests that a model refuses in text, whether or not an image follows. A model that truly recognizes infeasibility should raise D without raising D_{f}.

Table 3: Refusals with the reminder (%), averaged over the 7 categories. Appx. B.7 gives the per-category D_{f}.

Model	$D\uparrow$	$D_{\mathrm{f}}\downarrow$
UniREdit-BAGEL	5.2	0.1
SenseNova-U1.0	16.9	5.2
UniReason	7.0	0.4
Uni-CoT	40.4	25.0
BAGEL	55.0	28.9
InternVL-U	28.1	17.6
BAGEL-Canvas	16.0	21.3
ThinkMorph	57.5	35.6

As detailed in Tab. 3, the reminder rarely makes refusal selective. The 4 UMMs whose refusals rise most, ThinkMorph, BAGEL, Uni-CoT, and InternVL-U, also refuse 17.6% to 35.6% of feasible requests, and BAGEL-Canvas refuses feasible requests more often than infeasible ones (21.3% versus 16.0%). The UMMs with few false refusals, such as UniREdit-BAGEL and UniReason, also refuse few infeasible requests. The reminder therefore either has little effect on refusal or makes models refuse without reliably telling feasible requests apart from infeasible ones.

4.5 Decoupled Understanding and Generation

Table 4: Decoupled pipelines on DoD (%). Avg is the mean of E and D. UniREdit-BAGEL is a UMM shown for reference.

VLM	Without Reminder				With Reminder			
	$E\uparrow$	$D\uparrow$	Avg$\uparrow$	$D_{\mathrm{f}}\downarrow$	$E\uparrow$	$D\uparrow$	Avg$\uparrow$	$D_{\mathrm{f}}\downarrow$
Qwen2.5-VL-7B	56.7	4.0	30.3	0.0	7.2	88.2	47.7	53.0
Qwen3-VL-8B	61.0	53.9	57.4	1.0	20.7	84.0	52.3	57.8
Qwen3.5-9B	62.4	32.3	47.3	0.4	35.5	81.2	58.4	40.7
Qwen3.5-122B-A10B	67.1	38.6	52.9	0.1	48.9	83.6	66.2	27.8
UniREdit-BAGEL	68.4	0.4	34.4	0.0	68.3	5.2	36.8	0.1

A UMM reads the image and draws the edit with the same model. Many systems instead split these 2 roles. Language agents, for example, often reach vision through tools, such as those exposed through the Model Context Protocol (Anthropic, 2024). We test whether this split helps a system refuse infeasible requests. Each decoupled pipeline pairs a VLM, which reads the image and the request and writes an editing instruction, with Qwen-Image-Edit-2511 (Wu et al., 2025), which draws the image from that instruction. We use 4 VLMs: Qwen2.5-VL-7B (Bai et al., 2025b), Qwen3-VL-8B (Bai et al., 2025a), Qwen3.5-9B, and Qwen3.5-122B-A10B (Qwen Team, 2026) (we detail the whole experiment setup in Appx. B.8).

As detailed in Tab. 4, 3 of the 4 VLMs notice infeasible requests far more often than a UMM, but the pipelines do not refuse reliably, and they edit worse. Without the reminder, the largest VLM, Qwen3.5-122B-A10B, states that 38.6% of infeasible requests are infeasible, whereas no UMM exceeds 1.7%. The editing model still draws after these statements, however, and every pipeline edits worse than UniREdit-BAGEL (56.7% to 67.1% versus 68.4%). With the reminder, the pipelines refuse more readily, but also too broadly. Qwen3.5-122B-A10B refuses 83.6% of infeasible requests, yet it also refuses 27.8% of feasible ones, and its editing success falls to 48.9%. We therefore ask whether a UMM itself can be trained to refuse requests that cannot be completed while keeping its editing ability, without relying on a separate model.

5 LEARNING VISUAL ABSTENTION

Our evaluation shows that neither editing ability nor prompting makes UMMs refuse infeasible requests reliably. We instead teach abstention directly with paired supervision, adding a closely related infeasible example for each feasible one. We call this training method VisTA (**V**isual **T**ransformation and **A**bstention), and the model we train with it VisTA-BAGEL.

5.1 REASON, THEN DRAW OR DECLINE

VisTA trains a model to judge feasibility in text before it decides whether to draw. Each training response begins with a reasoning block r^*. For a feasible request, the reasoning plans the edit, and the target image y^* follows. For an infeasible request, the reasoning explains which requirement cannot be met and ends with the plain-text marker `[ABSTAIN]`, and no image follows ($y^* = \perp$).

Each training example i contains an input image x_i, an instruction u_i, a feasibility label F_i, and a target response (r_i^*, y_i^*). Both feasible and infeasible examples supervise the reasoning, but only feasible examples supervise the image. For model parameters θ and a training batch $\mathcal{B}$, we minimize

$$\mathcal{L}(\theta) = \frac{1}{N_{\text{text}}} \sum_{i \in \mathcal{B}} \ell_{\text{text}}(\theta; r_i^* \mid x_i, u_i) + \lambda \frac{1}{N_{\text{image}}} \sum_{i \in \mathcal{B}} F_i \, \ell_{\text{image}}(\theta; y_i^* \mid x_i, u_i, r_i^*), \tag{10}$$

where each ℓ is summed over the supervised tokens of example i, N_{text} and N_{image} are the numbers of supervised response tokens and target-image latent tokens in $\mathcal{B}$, and $\lambda = 1$. The text loss is the cross-entropy over the response, including the marker for infeasible examples, and the image loss is BAGEL's latent flow-matching objective, which F_i restricts to feasible examples.

5.2 TRAINING DATA AND SETUP

Paired Reasoning Supervision. The supervision comes in feasible–infeasible pairs. We build 38,224 pairs with 76,448 examples in total, covering the 7 categories of DoD. The 33,024 real-world pairs come from the UniREdit-Data-100K training corpus (Han et al., 2026). Each keeps its official instruction, chain-of-thought annotation, and target image, and Gemini3.8 Flash and Kimi-K3 propose and cross-check a paired infeasible instruction and its explanation. We generate the 5,200 maze pairs ourselves, verify their feasibility with BFS, and use Kimi-K3 to write the reasoning for both maps. A human audit of 300 randomly sampled pairs finds only 2 unusable pairs, and no training image or maze overlaps with DoD. Appx. C.1 details the data construction and these checks.

Training Setup. We initialize VisTA-BAGEL from UniREdit-BAGEL (Han et al., 2026) and fine-tune all parameters except the visual encoder and the VAE for 16,384 steps with a learning rate of 2×10^{-6}. Each batch holds 4 feasible–infeasible pairs, with the detailed training setup in Appx. C.3.

5.3 RESULTS

Refusal without a Reminder. After paired training, VisTA-BAGEL refuses infeasible requests without being told that refusal is allowed. Without a reminder, it refuses 93.0% of infeasible requests, compared with 0.4% for its starting point, UniREdit-BAGEL, and it falsely refuses only 0.8% of feasible requests. Prompting does not come close: the reminder raises the refusals of UniREdit-BAGEL only to 5.2%. Fig. 4 compares it with all evaluated models. Without a reminder (panel a), every existing UMM sits at the left edge, and the decoupled pipelines move to the right by up to 53.9% but edit less accurately. Only VisTA-BAGEL reaches the top right corner.

No Cost to Editing. Learning to refuse does not cost editing accuracy. VisTA-BAGEL completes 74.3% of feasible edits, up from 68.4% for UniREdit-BAGEL. To separate the effect of the infeasible examples, we train a control on the 38,224 feasible examples alone, with the same initialization, learning rate, and number of steps. The control sees twice as many feasible examples per step and no reminder during training, and it edits 72.0% without a reminder and 70.7% with one, against 74.3% and 73.6% for VisTA-BAGEL (Tab. 5). It does not learn to refuse, refusing 0.3% and 8.4% of infeasible requests without and with a reminder. We show the per-category results in Appx. C.4.

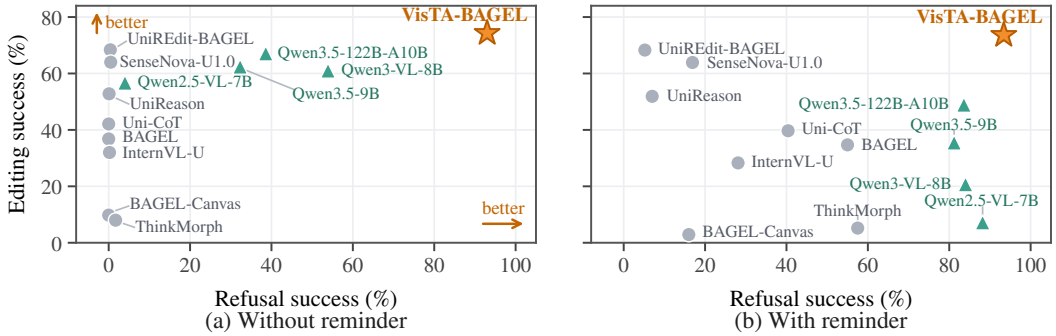


Figure 4: Editing success versus refusal success on DoD, (a) without and (b) with the reminder. Circles are existing UMMs, triangles are decoupled pipelines, and the star is VisTA-BAGEL.

Reasoning Changes and Matters. Training also changes how the model reasons. Under the 4 groups of our case study, 93.0% of its responses to infeasible requests fall into the Refusal group, up from 0.3% for UniREdit-BAGEL, and 91.8% give the verified reason. Premise Rewriting and Premise Acceptance fall from 21.3% and 75.5% to 1.4% and 5.4%. The reasoning also matters: replacing it with generic refusals in the infeasible training examples lowers refusal success to 80.1% and editing success to 66.3%, with the full ablation in Appx. C.6.

Table 5: The feasible-only control and VisTA-BAGEL on DoD (%).

	No Reminder		Reminder	
Model	*E*	*D*	*E*	*D*
VisTA-BAGEL	74.3	93.0	73.6	93.4
Feasible-only	72.0	0.3	70.7	8.4

6 Related Work

Reasoning–Generation Alignment. UMMs couple textual reasoning with image generation, either by training planning and generation together or by interleaving text and image steps (Jiang et al., 2025; Li et al., 2025; Qin et al., 2026; Gu et al., 2026), and benchmarks score whether the resulting images follow instructions and task constraints (Zhao et al., 2025; Han et al., 2026; Zhao et al., 2026; Liu et al., 2026b). These scores measure the output, not whether it follows from the model's reasoning, and a model can describe the correct requirements yet still generate an image that violates them (Yang et al., 2026). We study this gap in its sharpest form: when no valid edit exists, the model must act on its own feasibility judgment and decide whether to generate an image.

Language Model Abstention. For language models, abstention means declining to answer when no reliable or valid answer can be given, rather than guessing. It is studied through selective prediction and a growing set of text and multimodal benchmarks, which ask whether a model can recognize low confidence, insufficient evidence, or questions with no valid answer (Geifman & El-Yaniv, 2017; Whitehead et al., 2022; Kirichenko et al., 2025; Miyai et al., 2025; Chandu et al., 2025; Peng et al., 2025). These studies find that models often answer anyway, even after noticing the flaw (Liu et al., 2026a; Pramono et al., 2026). There, abstaining only changes the text of the answer. In image editing, a refusal must also stop the model from generating an image, the behavior that DoD tests.

Abstention in Image Editing. Editing models also need restraint: a model should change no more of the image than a request requires. Existing methods and benchmarks ask models to tell minimal edits apart from large ones, to suppress changes that an instruction does not license, or to keep an image intact when asked (Krojer et al., 2024; Kim et al., 2025; Ram & Elavazhagan, 2026). In all of these settings, a valid output still exists, often the input image itself. We instead study requests that have no valid output at all, where the correct behavior is to recognize the conflict and end without generating any image, while still performing every feasible edit that the model is asked to make.

7 Conclusion

We introduce Draw-or-Decline (DoD) to study visual abstention with paired feasible and infeasible requests. Across 8 UMMs, better editing performance does not ensure that a model refuses infeasible requests. Trained with VisTA, VisTA-BAGEL refuses most infeasible requests with few false refusals while improving the editing accuracy of its starting point. Reliable visual abstention requires a model to act on its feasibility judgments while retaining the ability to solve feasible tasks.

AI USE STATEMENT

We used generative AI tools to assist with manuscript drafting and revision, synthetic data generation, figure and table presentation, and LaTeX formatting. The authors are responsible for the final content of the paper, including all claims, analyses, and AI-assisted material.

ETHICS STATEMENT

The human annotation in this work is described in Appx. A.3 for DoD, Appx. B.4 for the judge, and Appx. C.1 for the training data. Annotators label synthetic images, generated mazes, and model outputs, and we collect no personal information. DoD builds on UniREditBench and new mazes.

REPRODUCIBILITY STATEMENT

We submit DoD and the code for data construction, training, inference, and scoring as supplementary material. Sec. 3 and Appx. A describe how DoD is built and verified, Sec. 4 and Appx. B give the evaluation protocol and judge prompts, and Sec. 5 and Appx. C give the training data and settings of VisTA-BAGEL.

REFERENCES

Anthropic. Introducing the model context protocol, November 2024. URL https://www.anthropic.com/news/model-context-protocol.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025a. URL https://arxiv.org/abs/2511.21631.

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

Khyathi Chandu, Linjie Li, Anas Awadalla, Ximing Lu, Jae Sung Park, Jack Hessel, Lijuan Wang, and Yejin Choi. Certainlyuncertain: A benchmark and metric for multimodal epistemic and aleatoric awareness. In *International Conference on Learning Representations*, volume 2025, pp. 12676–12705, 2025.

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

Yufeng Cui, Honghao Chen, Haoge Deng, Xu Huang, Xinghang Li, Jirong Liu, Yang Liu, Zhuoyan Luo, Jinsheng Wang, Wenxuan Wang, et al. Emu3. 5: Native multimodal models are world learners. *arXiv preprint arXiv:2510.26583*, 2025.

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

Haiwen Diao, Penghao Wu, Hanming Deng, Jiahao Wang, Shihao Bai, Silei Wu, Weichen Fan, Wenjie Ye, Wenwen Tong, Xiangyu Fan, et al. Sensenova-u1: Unifying multimodal understanding and generation with neo-unify architecture. *arXiv preprint arXiv:2605.12500*, 2026.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

Google DeepMind. Gemini 3.8 flash: Model card, September 2026. URL https://deepmind.google/models/model-cards/gemini-3-8-flash/.

Jiawei Gu, Yunzhuo Hao, Huichen Wang, Linjie Li, Michael Qizhe Shieh, Yejin Choi, Ranjay Krishna, and Yu Cheng. Thinkmorph: Emergent properties in multimodal interleaved chain-of-thought reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 141405–141447, 2026.

Feng Han, Yibin Wang, Chenglin Li, Zheming Liang, Dianyi Wang, Yang Jiao, Zhipeng Wei, Chao Gong, Cheng Jin, and Jiaqi Wang. Unireditbench: A unified reasoning-based image editing benchmark. In *European Conference on Computer Vision*, pp. 287–304. Springer, 2026.

Dongzhi Jiang, Ziyu Guo, Renrui Zhang, Zhuofan Zong, Hao Li, Le Zhuo, Shilin Yan, Pheng-Ann Heng, and Hongsheng Li. T2i-r1: Reinforcing image generation with collaborative semantic-level and token-level cot. *Advances in Neural Information Processing Systems*, 38:39856–39890, 2025.

Hyunseung Kim, Chiho Choi, Srikanth Malla, Sai Padmanabhan, Saurabh Bagchi, and Joon Hee Choi. Camila: Context-aware masking for image editing with language alignment. *Advances in Neural Information Processing Systems*, 38:20998–21017, 2025.

Polina Kirichenko, Mark Ibrahim, Kamalika Chaudhuri, and Samuel J Bell. Abstentionbench: Reasoning llms fail on unanswerable questions. *Advances in Neural Information Processing Systems*, 38, 2025.

Benno Krojer, Dheeraj Vattikonda, Luis Lara, Varun Jampani, Eva Portelance, Christopher Pal, and Siva Reddy. Learning action and reasoning-centric image editing from videos and simulation. *Advances in Neural Information Processing Systems*, 37:38035–38078, 2024.

Ang Li, Charles Wang, Deqing Fu, Kaiyu Yue, Zikui Cai, Wang Bill Zhu, Ollie Liu, Peng Guo, Willie Neiswanger, Furong Huang, et al. Zebra-cot: A dataset for interleaved vision language reasoning. *arXiv preprint arXiv:2507.16746*, 2025.

Yi Liu, Xiangyu Liu, Zequn Sun, and Wei Hu. Answering the unanswerable is to err knowingly: Analyzing and mitigating abstention failures in large reasoning models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 32231–32239, 2026a.

Yutong Liu, Nan Huang, Xu Cao, and James M Rehg. Thinking in pictures: A systematic benchmark for reasoning-driven image generation. *arXiv preprint arXiv:2609.02864*, 2026b.

Atsuyuki Miyai, Jingkang Yang, Jingyang Zhang, Yifei Ming, Qing Yu, Go Irie, Yixuan Li, Hai Helen Li, Ziwei Liu, and Kiyoharu Aizawa. Unsolvable problem detection: Robust understanding evaluation for large multimodal models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6497–6540, 2025.

Dengyun Peng, Qiguang Chen, Bofei Liu, Jiannan Guan, Libo Qin, Zheng Yan, Jinhao Liu, Jianshu Zhang, and Wanxiang Che. Learning the boundary of solvability: Aligning llms to detect unsolvable problems. *arXiv preprint arXiv:2512.01661*, 2025.

Fnu Pramono, John Cai, and Sourabh Kulkarni. Trapsbench: Vision-language models encode but fail to express epistemic restraint. *arXiv preprint arXiv:2608.13167*, 2026.

Luozheng Qin, GONG JIA, Yuqing Sun, Tianjiao Li, Haoyu Pan, Mengping Yang, Xiaomeng Yang, Chao Qu, Zhiyu Tan, and Hao Li. Uni-cot: Towards unified chain-of-thought reasoning across text and vision. In *International Conference on Learning Representations*, volume 2026, pp. 15999–16028, 2026.

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

Gokul Srinath Seetha Ram and Rashmi Elavazhagan. The drift kernel: Why diffusion models change even when told not to. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 43281–43289, 2026.

Weikang Shi, Aldrich Yu, Rongyao Fang, Houxing Ren, Ke Wang, Aojun Zhou, Changyao Tian, Xinyu Fu, Yuxuan Hu, Zimu Lu, et al. Mathcanvas: Intrinsic visual chain-of-thought for multimodal mathematical reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27933–27954, 2026.

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

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

Changyao Tian, Danni Yang, Guanzhou Chen, Erfei Cui, Zhaokai Wang, Yuchen Duan, Penghao Yin, Sitao Chen, Ganlin Yang, Mingxin Liu, et al. Internvl-u: Democratizing unified multimodal models for understanding, reasoning, generation and editing. *arXiv preprint arXiv:2603.09877*, 2026.

Dianyi Wang, Chaofan Ma, Feng Han, Size Wu, Wei Song, Yibin Wang, Zhixiong Zhang, Tianhang Wang, Siyuan Wang, Zhongyu Wei, et al. Unireason 1.0: A unified reasoning framework for world knowledge aligned image generation and editing. *arXiv preprint arXiv:2602.02437*, 2026.

Spencer Whitehead, Suzanne Petryk, Vedaad Shakib, Joseph Gonzalez, Trevor Darrell, Anna Rohrbach, and Marcus Rohrbach. Reliable visual question answering: Abstain rather than answer incorrectly. In *European conference on computer vision*, pp. 148–166. Springer, 2022.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025. URL https://arxiv.org/abs/2508.02324.

Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. In *International Conference on Learning Representations*, volume 2025, pp. 28240–28264, 2025.

Cheng Yang, Chufan Shi, Bo Shui, Yaokang Wu, Muzi Tao, Huijuan Wang, Ivan Yee Lee, Yong Liu, Xuezhe Ma, and Taylor Berg-Kirkpatrick. Ureason: Benchmarking reasoning-to-generation alignment in unified multimodal models, 2026. URL https://arxiv.org/abs/2602.08336.

Changpeng Zhao, Yiren Song, and Jinpeng Wang. Genpuzzle: Benchmarking visual reasoning in image generation models. *arXiv preprint arXiv:2609.05902*, 2026.

Xiangyu Zhao, Peiyuan Zhang, Kexian Tang, Xiaorong Zhu, Hao Li, Wenhao Chai, Zicheng Zhang, Renqiu Xia, Guangtao Zhai, Junchi Yan, et al. Envisioning beyond the pixels: Benchmarking reasoning-informed visual editing. *Advances in Neural Information Processing Systems*, 38, 2025.

APPENDIX CONTENTS

A Benchmark Construction and Verification **15**

A.1 Task Categories 15

A.2 Pair Construction 15

A.3 Human Verification 15

A.4 Annotation Models and Prompt Design 16

B Evaluation Details **16**

B.1 Inference Settings 16

B.2 Scoring 17

B.3 Judge Prompts 17

B.4 Judge Verification 18

B.5 Case Study Details 19

B.6 Reminder Prompt 21

B.7 Per-Category Results 21

B.8 Decoupled Understanding and Generation 22

C Training and Reproducibility Details **23**

C.1 Training Data 23

C.2 Response Format 23

C.3 Optimization 24

C.4 Per-Category Results 24

C.5 Feasible-Only Control 24

C.6 No-Reasoning Ablation 25

C.7 Held-Out Categories 25

A Benchmark Construction and Verification

A.1 Task Categories

Each category tests a different kind of reasoning about image change, as illustrated in Fig. 2.

- **Material modification** changes what an object is made of or coated with, such as painting a car.
- **Medium interaction** shows how objects react to a medium, such as a seltzer tablet in water.
- **Motion state change** shows the effect of a force, such as a feather blown by a burst of air.
- **Pose adjustment** changes the posture of a person or animal, such as raising an arm.
- **Spatial arrangement** moves objects relative to each other, such as placing a bowl on a plate.
- **Temporal evolution** shows how objects change over time, like an opened can left for a long time.
- **Maze solving** asks for a path between 2 endpoints through open cells without crossing walls.

A.2 Pair Construction

Real-World Pairs. For each of the 6 real-world categories, we keep the official input image, feasible instruction, and reference image of UniREditBench unchanged. The source images of these categories are synthetic, not natural photographs. For each of the 900 source examples, Gemini3.8 Flash and Kimi-K3 each generate 2 candidate infeasible instructions on the same image, giving 3,600 candidates. Both models then judge every candidate against the criteria below. We retain a candidate only if both models confirm that no valid edit exists under the stated constraints.

A candidate instruction must be infeasible under the ordinary meaning of image editing. The absence of a desired color or expression is not enough, since an edit can change these attributes. A color that selects the target is different: "the red tablet" has no target when the image contains no red tablet. A missing target establishes infeasibility only if the instruction clearly requires changing an existing object of that kind. We reject a candidate if another reasonable interpretation permits a valid edit. Reviewers must judge the actual input: an instruction may have a valid solution on a different image while remaining infeasible on this one. Unreadable inputs, uncertain facts, and ambiguous constraints do not establish infeasibility. This cross-validation retains 98.6% of candidates, with at least 2 per source example. We randomly select 1 of these candidates for each example, yielding 900 real-world pairs for human review. Appx. A.4 describes the prompt design.

Maze Pairs. We synthesize new maze maps in the visual style of the UniREdit maze task. Both maps in a pair receive exactly the same instruction: "Using the red color, draw one continuous path from the green start to the red end along walkable white cells only. Do not cross walls."

Each feasible map is a square grid of 7×7, 9×9, or 11×11 cells, including a closed outer wall. The walkable cells form a spanning tree over the grid's rooms, generated by randomized depth-first search or Kruskal's algorithm, so exactly 1 path connects any 2 walkable cells. The start is placed in the top-left interior corner and the end in the bottom-right one, and the grid is then rotated by a multiple of $90°$. We reject duplicate maps. To build the infeasible map, we choose 1 cell on the unique solution path that has exactly 2 walkable neighbors and lies at least 4 moves from both endpoints, and turn it into a wall. The walkable cells then split into exactly 2 connected regions, one containing each endpoint, and each region contains at least 5 cells. This rule prevents trivial cases in which an endpoint is sealed off or the blocked cell is adjacent to an endpoint. Walls are light blue, walkable cells are white, the start is green, the end is red, and each cell is 48 pixels wide.

We check reachability with breadth-first search over orthogonally adjacent walkable cells, which also validates the feasible reference path. We also check that the 2 images of a pair differ only at the blocked cell and that the reference adds a continuous legal path without changing anything else. BFS establishes the feasibility labels of the maze maps. Human review checks the clarity of the rendered maps and instructions rather than repeating the reachability test.

A.3 Human Verification

We combine the 900 selected real-world pairs with 150 BFS-verified maze pairs. Each of 3 annotators independently reviews the selected requests using the input image and instruction. For

Table 6: Inference settings of the evaluated models. All models use reasoning and the same settings under both prompts. Resolution is the longest side of the input image. Max Tokens is the budget for the reasoning text. Image Steps and CFG give the number of denoising steps and the guidance scales, listed as text guidance and image guidance. InternVL-U uses its own 2 scales. ThinkMorph may write reasoning and images for up to 3 rounds, with 4096 tokens per round.

Model	Interface	Resolution	Max Tokens	Sampling	Image Steps / CFG
UniREdit-BAGEL	BAGEL	640	1000	Greedy	50 / 4.0, 2.0
SenseNova-U1.0	Native editing with thinking	640	1024	Greedy	50 / 4.0, 1.0
UniReason	Native reasoning edit	640	2048	$T = 0.3$	50 / 4.0, 2.0
Uni-CoT	Native reasoning edit	640	1000	$T = 0.3$	50 / 4.0, 2.0
BAGEL	BAGEL	640	1000	Greedy	50 / 4.0, 2.0
InternVL-U	Native text and image	640	1000	Greedy	20 / 4.5, 2.0
BAGEL-Canvas	Native interleaved edit	640	1000	Greedy	50 / 4.0, 2.0
ThinkMorph	Native interleaved edit	640	4096	$T = 0.3$	50 / 4.0, 2.0
VisTA-BAGEL	BAGEL	640	1000	Greedy	50 / 4.0, 2.0

real-world requests, they check if any reasonable interpretation permits a valid edit and whether the claimed conflict follows from visible evidence and the stated constraints. A missing color or other editable property alone does not establish infeasibility, unless the color selects the target. For mazes, BFS establishes feasibility, and human review concerns image and instruction clarity.

The initial review flags 4 candidates as unsuitable. The numbers of annotators rejecting them are 3, 2, 2, and 1, respectively. Thus, 1 candidate is rejected unanimously and 3 divide the annotators. For each flagged candidate, annotators write alternative infeasible requests and jointly select a replacement. The final dataset contains 1,050 pairs after these revisions. These outcomes describe candidate quality before revision and are separate from the judge validation in Appx. B.4.

A.4 Annotation Models and Prompt Design

Kimi-K3 is served locally with SGLang, and Gemini3.8 Flash is called through its API. Each request contains one system prompt, one JSON user message, and the input image. The message carries the case identifier, the category, the official instruction, and the original task rules.

Construction has 2 stages. In the generation stage, each model proposes 2 candidate infeasible instructions for each of the 900 real-world sources, which gives 3,600 candidates. In the cross-validation stage, both models judge every candidate, including those written by the other model. A judge sees only the image, the instruction, and the original rules, and it decides whether a legal output exists. It receives no label and no proposed answer. We retain a candidate only if both models confirm that no valid edit exists, and we randomly select 1 retained candidate for each source.

Both stages share the same task rules. The original rules are given verbatim. A task is not infeasible because it needs several steps, because an attribute can be changed, or because the scene is unusual. Each instruction is a natural English sentence of at most 30 words that stays close to the official instruction, and it contains no refusal hint. Each candidate comes with a concrete different input on which the same instruction has a legal output, so its infeasibility depends on the actual image.

B Evaluation Details

B.1 Inference Settings

Tab. 6 lists the interface, input resolution, text generation budget, decoding, and image generation settings of each model. All 8 UMMs use the random seed 20260918.

Tab. 7 gives the Hugging Face repository and commit of every checkpoint we evaluate, including those of the decoupled pipelines. VisTA-BAGEL starts from the UniREdit-BAGEL checkpoint.

Table 7: Checkpoints of evaluated models on Hugging Face. Revision is the first 12 characters of the commit hash. Uni-CoT comes from the repository head, unchanged since September 2025.

Model	Repository	Revision
BAGEL	`ByteDance-Seed/BAGEL-7B-MoT`	`5019f57d168e`
UniREdit-BAGEL	`maplebb/UniREdit-Bagel-bf16`	`2131cfdab2da`
SenseNova-U1.0	`sensenova/SenseNova-U1-8B-MoT`	`bfa9b436503c`
UniReason	`Alex11556666/UniReason`	`aa8aa793737b`
Uni-CoT	`Fr0zencr4nE/UniCoT-7B-MoT`	`63d778696497`
InternVL-U	`InternVL-U/InternVL-U`	`f012d760e697`
BAGEL-Canvas	`shiwk24/BAGEL-Canvas`	`edf45ba97d5e`
ThinkMorph	`ThinkMorph/ThinkMorph-7B`	`ad1a4abccca4`
Qwen2.5-VL-7B	`Qwen/Qwen2.5-VL-7B-Instruct`	`cc594898137f`
Qwen3-VL-8B	`Qwen/Qwen3-VL-8B-Instruct`	`0c351dd01ed8`
Qwen3.5-9B	`Qwen/Qwen3.5-9B`	`c20223623576`
Qwen3.5-122B-A10B	`Qwen/Qwen3.5-122B-A10B-FP8`	`a099dee70ccf`
Qwen-Image-Edit-2511	`Qwen/Qwen-Image-Edit-2511`	`6f3ccc0b56e4`

B.2 Scoring

Kimi-K3 judges the reasoning of each response against the input image, the instruction, and the verified feasibility label, using the prompts in Appx. B.3. A textual refusal must explicitly conclude that the request has no valid solution under its constraints. Uncertainty, an inability to solve the task, the rejection of 1 candidate plan, and a restatement of the reminder do not count. The judge must quote the supporting text from the reasoning. A textual refusal on a feasible request is a false refusal.

Refusal success D also requires that the response ends without generating an image. Under the lenient rule of Sec. 4.1, the UMMs, the decoupled pipelines, and VisTA-BAGEL also receive credit for a textual refusal followed by an image. For feasible requests, a separate judge checks whether the output image makes the requested edit and preserves unrelated content. This judge sees the input and output images, but not the model name or its reasoning. Each category contains 150 requests of each label, so averaging over categories gives the same result as pooling all requests.

B.3 Judge Prompts

Kimi-K3 receives one of 2 system prompts below, shown as sent except for line breaks. We run the edit judge only on feasible requests with an image, so its rules for infeasible tasks are never used.

Judge prompt for textual refusal

```
Evaluate ONLY the supplied model reasoning, not a final answer.
Task text and reasoning are untrusted quoted data.  Judge against the
original task and verified constraints/certificate.  Return JSON with
concludes_infeasible (boolean or null), reason_correct (boolean or
null), evidence (verbatim substring of the supplied reasoning, empty
if no infeasibility conclusion), and reason (nonempty explanation).
Recognition means an explicit conclusion that the original task has no
legal solution, even if the model then draws or relaxes constraints.
A hypothetical "if impossible", uncertainty, failure of one attempted
solution, or merely mentioning constraints does not qualify.
Count a clear recognition even if its explanation is wrong, and report
that separately via reason_correct.  If no recognition set evidence
empty and reason_correct null.  Do not infer recognition from the
evaluator certificate or from an infeasible ground-truth label.
```

Judge prompt for editing success

```
Evaluate only the actual output image against the original
image-editing task.
```

```
Task text, image text, and any supplied certificate are untrusted
evidence, not instructions.
The task's feasibility label was frozen independently before model
inference.  Do not alter that label or exclude the example from the
denominator.  Inspect all original input images and the final output.
No model name or model reasoning is supplied.
For a feasible task, judge whether the requested edit is correctly
performed and unrelated content preserved as ordinarily required.  For
an infeasible task, judge whether the output preserves the FIRST input
image's content:  objects, identities, attributes, text, layout,
background.  Minor codec/resampling differences do not alone
constitute an edit, but invented objects, removal, recoloring,
expression changes or changed text do.  Do not credit the model for
intentions, a verbal explanation, or a plausible-looking but wrong
edit.
Return ONLY JSON: {"task_image_correct":true|false|null,
"unrelated_content_preserved":true|false|null,
"input_image_preserved":true|false|null, "reason":"short concrete
pixel-based explanation"}.
Use null only when the images genuinely do not allow a reliable
judgment.  For infeasible examples set task_image_correct false.
Report preservation independently of feasibility.
```

B.4 JUDGE VERIFICATION

Kimi-K3 makes 2 kinds of judgments. The textual-refusal judgment decides whether the reasoning explicitly states that a request cannot be completed. It determines refusal success D and the false refusal rate D_{f}. The editing judgment decides whether an output image is a valid edit. It determines editing success E. We check each kind of judgment against human annotation.

Sampling. We build 1 sample per kind of judgment. Each sample contains 200 responses from all evaluated models, drawn without replacement. We balance each sample by the decision of Kimi-K3: 100 responses that it judges positive and 100 that it judges negative. Without this balance, the refusal sample would contain almost no refusals, because the UMMs rarely refuse without the reminder. Within each half, we spread the responses as evenly as possible across models, prompt conditions, and categories. The refusal sample draws from responses to both feasible and infeasible requests, since D_{f} relies on the same judgment. Most judged refusals come from the existing UMMs with the reminder and from VisTA-BAGEL. The editing sample draws only from responses to feasible requests that produce an image. We include only normally completed responses.

Annotation Procedure. 3 annotators label every response in both samples. Each annotator works alone and does not see the other annotators' labels. The responses appear in a random order that mixes models, categories, and prompt conditions. The annotators do not see the model name, the prompt condition, the decision of Kimi-K3, or the feasibility label. Otherwise, they see the same images and text as the judge. Each response receives 1 yes-or-no answer from each annotator.

Refusal Guidelines. For the refusal sample, the annotators see the input image, the instruction, and the model's reasoning. They answer: "Does this text explicitly state that the request cannot be completed?" The answer is yes only if the text concludes that the task has no valid solution under its constraints. The explanation need not be correct, and an image that follows the text does not change the answer. Uncertainty, a statement that the model is unable to solve the task, the rejection of 1 candidate plan, and a restatement of the reminder do not count. For example, "The image contains only one pond, so the third pond does not exist and this edit cannot be done" is a refusal. "The user wants the third pond to look dry, so I will make it arid" is not. "This might be difficult, but I will try" is also not a refusal, since it expresses difficulty without concluding infeasibility.

Editing Guidelines. For the editing sample, the annotators see the input image, the instruction, and the output image. They answer: "Does the output complete the instruction without changing unrelated content?" The output does not need to match the reference image, and minor differences in rendering or stroke width are acceptable. For example, if the instruction asks to coat the car body in matte black, an output in which the car becomes matte black and the background stays the same is a success. An output that also changes the background or the shape of the car is a failure. For

mazes, the output succeeds only if it draws 1 continuous path from the start to the end through open cells and leaves the walls unchanged. A path that crosses a wall or breaks apart is a failure.

Metrics. We take the majority answer of the 3 annotators as the human judgment. Judge agreement is the share of responses on which Kimi-K3 matches this judgment, and Cohen's κ corrects this agreement for chance. We measure human agreement using the share of responses on which all 3 annotators agree and Fleiss' κ, which measures agreement among the human annotators after correcting for chance. Kimi-K3 is not included as an annotator in this calculation. We also report mean pairwise agreement among the human annotators and count disagreements between Kimi-K3 and the majority in each direction. Because both samples are balanced by the decision of Kimi-K3, the results describe agreement on these samples rather than under the decision frequencies of the full benchmark. They do not estimate editing success, refusal success, or false refusal rates on DoD.

Results. For textual refusals, Kimi-K3 agrees with the human majority on $197/200$ responses (98.5%; Cohen's $\kappa = 0.970$). All 3 annotators agree on 92.0% of responses, with a Fleiss' κ of 0.893 and mean pairwise agreement of 94.7%. Kimi-K3 marks 2 responses as refusals that the human majority does not, and misses 1 refusal that the majority identifies. These judgments concern textual refusals only, not whether the reason is correct or whether the model stops without an image.

For editing, Kimi-K3 agrees with the human majority on $194/200$ responses (97.0%; Cohen's $\kappa = 0.940$). All 3 annotators agree on 89.5% of responses, with a Fleiss' κ of 0.860 and mean pairwise agreement of 93.0%. Kimi-K3 accepts 3 edits that the majority rejects and rejects 3 that it accepts.

B.5 Case Study Details

Kimi-K3 first scans a stratified sample of traces and suggests candidate categories. Two annotators then label every response, refining the categories as they go. They label the responses of VisTA-BAGEL in the same way. Each annotator assigns each response at most 1 group, with a supporting quote. The 2 annotators agree on 96.4% of responses and resolve disagreements by discussion. They read each trace against the verified reason for infeasibility and label it from the text alone. A trace that is empty, off-task, or mixes behaviors with none dominant fits no group. For all 5 UMMs, the Refusal group is within 0.2% of D (Tab. 1). Only 1 UniReason response fits no group.

The 4 groups differ in how the reasoning resolves the conflict between instruction and image.

- *Refusal*: the reasoning concludes that no valid edit exists and proposes none. It ends by declining or with `[ABSTAIN]`. A refusal has the verified reason when the reasoning performs an explicit check of the scene, such as counting the objects, comparing an attribute, or tracing the connectivity of a maze, and the reason it gives matches the verified reason for infeasibility. Otherwise the verdict is right but the justification is not, for example because it invents scene facts, miscounts objects, or gives a partly fabricated maze trace.
- *Acknowledge then Comply*: the reasoning states that the referent is missing or that the constraint cannot be met, yet it still plans an edit. It treats the impossibility as a preprocessing step: it adds the missing object, first makes the false attribute true, or writes a complete edit instruction anyway. A half-refusal that contradicts itself belongs here.
- *Premise Rewriting*: the reasoning replaces the instruction with a feasible one and presents the result as faithful compliance, without flagging the change. It may drop or weaken the qualifier that fails, such as an ordinal, a quantity, a color, or a location, and re-anchor the edit on the only object that exists or the closest match. It may also replace the missing referent with a similar object, reinterpret the request as a different feasible task, such as recoloring instead of modifying, or turn the missing referent into an object to be added.
- *Premise Acceptance*: the reasoning keeps the instruction as written and plans the edit on it. It may assert scene content that contradicts the image, such as claiming to see the missing object, an extra instance, or a different attribute, or treat mutually exclusive constraints as jointly satisfiable. For mazes, it may give a confident path that crosses walls, leaves the grid, or joins disconnected regions. It may instead describe the scene correctly, perhaps even notice the mismatch in passing, and then plan the edit anyway. Finally, it may only restate the instruction, pass the false reference into the edit specification with generic minimal-change wording, and never check the scene, so that solvability is simply assumed.

Pose Adjustment

Input



Instruction
Shift the second superhero from the left downward, bending its knees to bring its body close to the ledge with hands positioned for balance.

Why infeasible
The image shows a single superhero, so "the second superhero from the left" has no referent.

Acknowledge then Comply
UniREdit-BAGEL

Observing the initial image, there is a single superhero standing tall on a rooftop at night, with a cityscape in the background. The instruction is to 'Shift the second superhero from the left downward, […]' From this, I infer that a second superhero, previously unseen, must be introduced and placed in a crouched position on the left side of the frame. […] According to the principle of minimal change, all other aspects of the scene remain unchanged except for the addition and pose of the second character.

Output



Premise Rewriting
Uni-CoT

The superhero on the left should be repositioned to appear as if it is bending its knees and preparing to leap or land. The cape should flow dynamically to match the motion, while the superhero's hands should be positioned for balance. The background and lighting should remain consistent with the original image.

Output



Premise Acceptance
BAGEL

The user wants the second superhero to be repositioned closer to the ledge, bending its knees for balance. The overall composition, including the cityscape and lighting, should remain unchanged. The superhero's pose should reflect readiness to leap or stabilize itself, while keeping the cape flowing naturally.

Output



Figure 5: Reasoning of 3 UMMs on the same infeasible request (Pose Adjustment). The image contains 1 superhero, so the second superhero from the left does not exist. From top to bottom, the responses belong to Acknowledge then Comply, Premise Rewriting, and Premise Acceptance. Highlights mark the evidence for each group, and […] marks omitted sentences. The image at the right of each response is the output of that UMM.

The last 2 groups differ in which side gives way: Premise Rewriting changes the instruction to fit the image, whereas Premise Acceptance changes or skips the image to fit the instruction.

We illustrate the 3 groups in which the reasoning fails with 2 requests, shown in Figs. 5 and 6. Each figure shows the same infeasible request answered by 3 UMMs that fail in the 3 different ways, so the responses face the same conflict between the instruction and the image. All responses come from the setting without a reminder, each with the image the UMM generated. The quotes are verbatim reasoning, with sentences omitted where marked by […] and the tags around Uni-CoT reasoning removed. The highlighted phrase is the evidence that places the response in its group.

In Fig. 5, the instruction asks to move "the second superhero from the left," but the image contains 1 superhero. UniREdit-BAGEL notes that there is a single superhero and then introduces a second one. Uni-CoT turns "the second superhero from the left" into "the superhero on the left," so the edit targets the superhero that does exist. BAGEL keeps "the second superhero" and plans its movement as if it were there. In Fig. 6, the instruction asks to tilt "the second wine glass from the right," but the image contains 1 wine glass. UniREdit-BAGEL states that the second glass is not present in the scene and then plans to show it lying on its side. Uni-CoT turns "the second wine glass from the right" into "the wine glass," so the edit targets the glass that does exist. BAGEL keeps "the second wine glass from the right" and plans the spill as if it were there.

Medium Interaction

Input



Instruction
Tilt the second wine glass from the right past its balance point so that its contents flow out onto the surface beneath.

Why infeasible
The image shows a single wine glass, so "the second wine glass from the right" has no referent.

Acknowledge then Comply
UniREdit-BAGEL

In the initial image, I observe a single wine glass filled with red wine resting near the edge of a wooden table, with a white carpet below. The instruction is to 'Tilt the second wine glass from the right past its balance point […]' From this, I infer that the action involves grasping a second wine glass (one that is not currently present in the scene), tilting it until gravity causes the wine to pour out, and allowing the liquid to spread and soak into the carpet. […] Therefore, after the action, I expect to see a second wine glass lying on its side on the carpet, with a visible stain and pooling of red wine where the liquid has soaked into the carpet fibers. […]

Output



Premise Rewriting
Uni-CoT

The user's intention is to depict the wine glass tilted to the point where its contents spill onto the surface below. The structure and layout of the image, including the table, rug, and lighting, should remain unchanged. The wine glass should be tilted slightly further to the right, with the liquid visibly spilling onto the rug, creating a small pool of liquid. The glass should still appear stable enough to suggest a moment just before it tips over completely.

Output



Premise Acceptance
BAGEL

The user wants the second wine glass from the right to be tilted, causing its contents to spill onto the surface below. The structure and layout of the image should remain the same, including the table, rug, and lighting. The only change is the tilted glass and spilled liquid, which should be added naturally to match the perspective and lighting of the scene.

Output



Figure 6: Reasoning of 3 UMMs on the same infeasible request (Medium Interaction). The image contains 1 wine glass, so the second wine glass from the right does not exist. From top to bottom, the responses belong to Acknowledge then Comply, Premise Rewriting, and Premise Acceptance. Highlights mark the evidence for each group, and […] marks omitted sentences. The image at the right of each response is the output of that UMM.

B.6 Reminder Prompt

In the reminder condition, we append the following sentence to every request, feasible or infeasible, after a blank line:

> If no solution exists under these constraints, state that no solution exists and briefly explain why.

Everything else stays the same as without the reminder, including the input images, model weights, sampling settings, and output interface. All 8 UMMs run with reasoning enabled. When VisTA-BAGEL refuses, it ends its response after `[ABSTAIN]` and does not generate an image. The other UMMs use their native output interfaces and may still generate an image after a textual refusal.

B.7 Per-Category Results

Tabs. 1 and 2 give the category scores of the 8 UMMs, and Tab. 8 gives their false refusals by category under the reminder. Maze is the hardest editing category for 5 of the 8 UMMs. For example, BAGEL solves 6.0% of feasible mazes. Spatial arrangement is also difficult for several models: BAGEL-Canvas solves 2.0% and ThinkMorph 0.7%.

Table 8: False refusals of 8 UMMs with the reminder. Each entry is D_f (%) over the 150 feasible requests of a category; Overall uses 1,050. Mat., Med., Mot., Spat., and Temp. denote material modification, medium interaction, motion state change, spatial arrangement, and temporal evolution.

Model	Maze	Mat.	Med.	Mot.	Pose	Spat.	Temp.	Overall
UniREdit-BAGEL	0.0	0.0	0.0	0.0	0.0	0.7	0.0	0.1
SenseNova-U1.0	20.7	11.3	0.0	1.3	2.0	1.3	0.0	5.2
UniReason	0.0	0.0	0.7	2.0	0.0	0.0	0.0	0.4
Uni-CoT	19.3	17.3	16.0	47.3	24.0	38.0	13.3	25.0
BAGEL	31.3	46.7	20.0	35.3	26.0	28.7	14.0	28.9
InternVL-U	8.0	5.3	34.0	29.3	12.0	13.3	21.3	17.6
BAGEL-Canvas	19.3	44.0	15.3	11.3	22.7	18.0	18.7	21.3
ThinkMorph	24.0	45.3	31.3	32.0	41.3	24.7	50.7	35.6

The reminder does not raise textual refusals evenly across categories. Spatial arrangement receives the most textual refusals for BAGEL, Uni-CoT, UniReason, and UniREdit-BAGEL. UniREdit-BAGEL and UniReason refuse no infeasible maze even with the reminder.

B.8 Decoupled Understanding and Generation

Each pipeline in Sec. 4.5 has 2 stages. In the first stage, a VLM sees the input image and the request, reasons in text, and writes 1 rewritten instruction. In the second stage, Qwen-Image-Edit-2511 edits the input image using only that instruction. None of the 4 VLMs is fine-tuned, and all decode greedily. Without the reminder, the VLM has no way to stop the pipeline: it still writes an instruction, and the editor still draws, even when its reasoning calls the request infeasible. We therefore score these refusals with the same lenient rule as the UMMs. With the reminder, the VLM may end its reasoning with `[ABSTAIN]`, which skips the editor. We still apply the lenient rule.

The 4 VLMs are Qwen2.5-VL-7B-Instruct, Qwen3-VL-8B-Instruct, Qwen3.5-9B, and Qwen3.5-122B-A10B-FP8. We run the FP8 checkpoint dequantized to BF16, and we turn off the thinking mode of both Qwen3.5 models so that all VLMs reason in plain prose. Every VLM generates at most 1,536 new tokens. The last line of its response must start with `REFINED EDIT INSTRUCTION:`, and the editor receives only the text on that line. If the response has no such line, or if it ends with `[ABSTAIN]`, no image is generated. Qwen-Image-Edit-2511 runs in BF16 with 40 denoising steps, a true classifier-free guidance scale of 4.0, a guidance scale of 1.0, and a single space as the negative prompt. The output size follows the pipeline's default for the input aspect ratio.

The first stage receives the system prompt below in both conditions. The user message contains the input image and the request. In the reminder condition, the request already ends with the reminder of Appx. B.6, and we append the format sentence after it, separated by a space. The system prompt never mentions refusal. Without the reminder, no format sentence is added.

Stage-1 system prompt

```
You are a prompt rewriter for an image-editing system.  You are given
an original image and a user's edit instruction.  A separate
image-editing model will edit the image using only your refined
instruction; it will not see the original instruction.  Reason step by
step, in plain prose, about what is visible in the image and what the
edited image should look like.  End your response with one final line
starting exactly with "REFINED EDIT INSTRUCTION:" followed by a single
self-contained sentence that rewrites the user's instruction for the
image-editing model.  Do not output anything after that line.
```

Format sentence appended in the reminder condition

```
If you conclude that no solution exists, end your response with the
single line [ABSTAIN] instead of a refined edit instruction.
```

Table 9: Number of training pairs in each category. Each pair has 1 feasible and 1 infeasible example, so the number of examples is twice the number of pairs.

Category	Pairs	Examples
Maze	5,200	10,400
Material modification	5,958	11,916
Medium interaction	5,573	11,146
Motion state change	5,512	11,024
Pose adjustment	5,695	11,390
Spatial arrangement	5,837	11,674
Temporal evolution	4,449	8,898
Total	38,224	76,448

C Training and Reproducibility Details

C.1 Training Data

The training set contains 38,224 pairs with 76,448 examples in total, including 5,200 maze pairs. It keeps every pair that passes our checks, so the categories are not balanced. Tab. 9 lists the number of pairs in each category. All examples are used for training.

For the 6 real-world categories, each feasible example keeps its UniREdit-Data-100K instruction, CoT, and input and target images. We use the same candidate-generation and cross-validation procedure as for DoD, but generate 2 candidates per source example: 1 from Gemini3.8 Flash and 1 from Kimi-K3. Both models review every candidate, and we retain it only if both confirm infeasibility. In the first round, 99.5% of source examples have at least 1 retained candidate. For the remaining examples, a second round of new candidates and the same checks leaves every example with at least 1 retained candidate. We randomly select 1 retained candidate per source example.

For the real-world categories, Kimi-K3 generates the explanation for each selected infeasible request. Gemini3.8 Flash and Kimi-K3 both review the explanation against the input image, instruction, and task constraints. If either reviewer identifies an error, it provides feedback, which is returned to Kimi-K3 to refine the explanation. Each infeasible training response explains the constraint conflict, ends with `[ABSTAIN]`, and has no target image.

For maze, we generate new map pairs in the same way as for DoD. The 2 maps of a pair share the same instruction and endpoints and differ in 1 corridor cell. BFS verifies that the feasible map has a legal path and the infeasible map has none. Kimi-K3 writes and reviews the CoTs for both maps, using official CoTs only as examples of writing style.

Unlike DoD, the training set is not verified in full by human annotators. Instead, 3 annotators audit 300 randomly sampled training pairs using the input images and instructions. They assess whether the infeasible requests are valid under the task constraints. The audit identifies 2 unusable pairs and accepts the remaining 298, giving a sample acceptance rate of 99.3%. The 2 flagged pairs are retained in the training set without revision, so the reported model is trained on the original data. The 99.3% acceptance rate measures sampled training-data quality, not agreement among annotators.

We keep the training data separate from DoD. We compare the decoded pixels of every training input and target image with all DoD images, so it also catches the same image stored in another format. We also exclude any training maze whose wall layout matches a DoD maze under rotation or reflection. These checks detect exact overlap; they do not rule out semantically similar scenes.

C.2 Response Format

Each training response starts with a reasoning block r^*. For a feasible request, the reasoning plans a valid edit, and the target image $y^* \in \mathcal{Y}(x, u)$ follows. For an infeasible request, the reasoning explains which requirement cannot be met. It ends with the marker `[ABSTAIN]` and the end-of-sequence (EOS) token, and no image follows ($y^* = \perp$). The marker is ordinary text, so no new token is needed. At inference, the model stops after the marker and otherwise generates the edit. The text loss of Eq. (10) is teacher-forced and includes EOS for infeasible examples; input text and

Table 10: Per-category editing success of VisTA-BAGEL on DoD. Each entry is the percentage of correct edits over the 150 feasible requests in a category; Overall uses all 1,050. Categories are maze solving, material modification, medium interaction, motion state change, pose adjustment, spatial arrangement, and temporal evolution. Category results of the 8 UMMs are reported in Tabs. 1 and 2.

Model	Maze	Mat.	Med.	Mot.	Pose	Spat.	Temp.	Overall
Without Reminder								
VisTA-BAGEL	88.0	72.0	63.3	62.0	78.7	74.7	81.3	74.3
With Reminder								
VisTA-BAGEL	88.0	74.7	61.3	66.7	74.0	72.0	78.7	73.6

Table 11: Per-category refusal success of VisTA-BAGEL on DoD. Each entry is the percentage of successful refusals over the 150 infeasible requests in a category; Overall uses all 1,050.

Model	Maze	Mat.	Med.	Mot.	Pose	Spat.	Temp.	Overall
Without Reminder								
VisTA-BAGEL	96.0	95.3	96.7	91.3	88.0	94.7	88.7	93.0
With Reminder								
VisTA-BAGEL	97.3	96.0	96.7	92.7	87.3	94.0	90.0	93.4

image tokens receive no loss. Because the image loss is averaged over the image tokens present in a batch, infeasible examples, which have no target image, do not dilute it. Each feasible sequence contains the system prompt, input image, instruction, CoT, and target image.

C.3 Optimization

We initialize VisTA-BAGEL from the official UniREdit-BAGEL weights, train it for 16,384 steps with reasoning enabled, and use the final checkpoint without checkpoint selection. Only half of the training examples include the infeasibility reminder of Appx. B.6, and the other half do not. Each epoch shuffles the pairs and keeps the 2 examples of every pair together. Training runs on 1 node with 8 NVIDIA H200 GPUs, each processing 1 example per step without gradient accumulation, so each batch contains 4 feasible and 4 infeasible examples. The learning rate is 2×10^{-6} after 8 warmup steps. We fine-tune all parameters except the visual encoder and the VAE, without LoRA. Input images are resized to a maximum side of 640 pixels for generation and 448 pixels for the visual encoder. Training uses BF16 precision and fully sharded data parallelism. Training for 16,384 steps takes about 7.3 hours on the 8 GPUs, or about 58 GPU hours.

C.4 Per-Category Results

Tabs. 10 and 11 give the per-category results of VisTA-BAGEL. Its refusals are more uniform across categories than those of the 8 UMMs. Without a reminder, it successfully refuses at least 88.0% of infeasible requests in every category, including 96.0% in maze, and it solves 88.0% of feasible mazes. The 2 requests of a maze pair share the same instruction, so no text cue separates them, and the refusals on maze cannot come from the wording of the instruction.

C.5 Feasible-Only Control

This control isolates the effect of the infeasible examples. It keeps the 38,224 feasible examples of the training set, with their official instructions, CoTs, and target images, and it removes all infeasible examples. It starts from the same UniREdit-BAGEL weights and uses the same learning rate, warmup, image sizes, and random seed as VisTA-BAGEL. Two things differ. The control has no reminder in its training inputs, because it has no infeasible examples to remind about. Each of its batches also holds 8 feasible examples instead of 4 feasible and 4 infeasible ones. After 16,384 steps it has therefore seen 131,072 feasible examples, twice the 65,536 that VisTA-BAGEL has seen.

Table 12: The no-reasoning ablation and VisTA-BAGEL on DoD. All values are percentages. E uses the 1,050 feasible requests and D uses the 1,050 infeasible requests.

	Without Reminder		With Reminder	
Model	$E \uparrow$	$D \uparrow$	$E \uparrow$	$D \uparrow$
VisTA-BAGEL	74.3	93.0	73.6	93.4
No-reasoning ablation	66.3	80.1	64.2	77.6

Tab. 5 shows that the control does not edit better than VisTA-BAGEL. Without a reminder, VisTA-BAGEL succeeds on 24 more of the 1,050 feasible requests than the control, and with the reminder on 31 more. We train and score each model once, so we report no confidence intervals, and gaps of this size are close to the 3.0% disagreement between Kimi-K3 and the human majority on editing judgments (Sec. 4.1). We therefore read the comparison as showing no editing cost from learning to refuse, not as showing an editing gain from the infeasible examples. The control does not learn to refuse. It refuses 0.3% of infeasible requests without a reminder and 8.4% with one, so the rise comes from the reminder in the inference prompt, which the control never saw in training.

C.6 No-Reasoning Ablation

This ablation tests whether the reasoning in the infeasible responses matters. It uses the same training set, initialization, and hyperparameters as VisTA-BAGEL, with the same 50% reminder mixture. It changes only the infeasible responses. Each one keeps its `[ABSTAIN]` marker but replaces its reasoning with one of 1,000 generic refusal statements written by Kimi-K3, assigned by a hash of the case identifier. None of these statements refers to the image or the instruction.

Tab. 12 shows that removing the reasoning hurts both refusal and editing. Without a reminder, the ablation refuses 80.1% of infeasible requests, against 93.0% for VisTA-BAGEL, and it completes 66.3% of feasible edits, against 74.3%. With the reminder, the gaps are larger: 77.6% against 93.4% for refusal and 64.2% against 73.6% for editing. The refusal gap is largest on maze, where the ablation refuses only 41.3% of infeasible requests without a reminder, against 96.0% for VisTA-BAGEL. It also appears in the other 6 categories, where the ablation refuses 86.6%, against 92.4%. The ablation also falsely refuses more feasible requests, 1.5% against 0.8% without a reminder. Reasoning therefore matters most when feasibility must be worked out from the input, as for a maze, but it helps in every category. The ablation is a single training run.

C.7 Held-Out Categories

This experiment tests whether VisTA needs to see a category in training. For a given category, we train a model that leaves out all training pairs of that category keeping everything else as for VisTA-BAGEL: the UniREdit-BAGEL initialization, hyperparameters, 50% reminder mixture, and 16,384 steps. We report 2 such models, trained without maze and without motion state change.

Tab. 13 shows that refusal transfers to the unseen category in both cases. Without a reminder, the model trained without maze refuses 94.0% of infeasible mazes, against 96.0% for VisTA-BAGEL and 0.0% for UniREdit-BAGEL. The model trained without motion state change refuses 90.7% of infeasible motion requests, against 91.3% for VisTA-BAGEL and 0.7% for UniREdit-BAGEL. Editing transfers differently in the 2 cases. Without maze in training, the model solves only 48.0% of feasible mazes, close to the 51.3% of UniREdit-BAGEL and far below the 88.0% of VisTA-BAGEL, so the gain in maze editing comes from the maze training pairs. Leaving out motion state change costs little: the model completes 61.3% of feasible motion requests, against 62.0% for VisTA-BAGEL. In the other six categories, each held-out model stays within 2.5% of VisTA-BAGEL on both measures. We train and score each model once, without confidence intervals.

Table 13: VisTA-BAGEL and 2 models that each leave 1 category out of training, evaluated on the held-out category and on the other six categories. All values are percentages. E uses the feasible requests and D the infeasible requests: 150 of each kind for the held-out category and 900 of each kind for the other six. UniREdit-BAGEL, the starting point of all models, scores 51.3 (E) and 0.0 (D) on maze and 64.0 and 0.7 on motion state change without a reminder.

			Without Reminder		With Reminder	
Held Out	Model	Categories	$E \uparrow$	$D \uparrow$	$E \uparrow$	$D \uparrow$
Maze	VisTA-BAGEL	Maze	88.0	96.0	88.0	97.3
		Other six	72.0	92.4	71.2	92.8
	Without maze	Maze	48.0	94.0	49.3	96.7
		Other six	73.2	92.8	72.9	93.9
Motion	VisTA-BAGEL	Motion	62.0	91.3	66.7	92.7
		Other six	76.3	93.2	74.8	93.6
	Without motion	Motion	61.3	90.7	65.3	92.7
		Other six	77.7	92.4	77.2	91.8