Progress by Pieces: Test-Time Scaling for Autoregressive Image Generation

Supplementary Material

A. Related Works

Visual autoregressive models Visual autoregressive (AR) models have recently advanced rapidly as a natural extension of the language-modeling paradigm to vision [7, 24–26]. By adopting vector-quantized image representations, these methods discretize images into codebook indices, allowing the next-token prediction process to operate over image tokens. In practice, their generative quality is now competitive with strong diffusion baselines such as DALL·E 3 [2] and Stable Diffusion 3 [9].

Standard visual AR models decode in a raster-scan order (top-left to bottom-right), though several notable variants have been proposed. Masked autoregressive models (MAR [19]) offer a unified formulation of standard AR models and masked generative models, while VAR [26] applies next-scale prediction instead of next-token prediction. Nevertheless, the standard raster-scan AR formulation remains a competitive and versatile baseline for both image understanding and generation, and has also been adapted for image editing [21].

Test-time compute scaling While large language models (LLMs) have benefited significantly from test-time scaling techniques, such strategies remain underexplored in the context of visual autoregressive (AR) generation. In the language domain, increasing inference-time computation has proven to substantially improve model performance on challenging reasoning tasks such as mathematics [27] and coding [6], without any changes to model parameters. Two representative strategies include (i) chain-of-thought (CoT) prompting [16, 30], which extends generation length to encourage multi-step reasoning, and (ii) Best-of-N sampling with outcome reward models (ORMs) [23], which samples multiple outputs and selects the best one based on task-specific reward signals. Beyond outcome-based scoring, process reward models (PRMs) have also been introduced to evaluate intermediate reasoning steps, enabling more robust selection among sampled candidates. These methods demonstrate how allocating additional compute at inference can serve as a powerful tool to elicit more accurate or coherent outputs from pretrained models.

Extending test-time scaling strategies to visual AR models presents substantially different challenges. Unlike language models, which generate left-to-right sequences with clearly interpretable intermediate states, visual AR decoders expose only localized information at early steps due to their fixed raster-scan generation order. This structural constraint limits opportunities for selective computation allocation, mid-generation intervention, or step-wise verification. Some recent studies have explored test-time scaling within specific visual-generation pipelines [8, 34]. These works commonly observe that coarse early-stage outputs already contain meaningful global structure, enabling an approximate assessment of whether the generation is likely to successfully follow the given instruction. Although such findings highlight the usefulness of early structural cues, the corresponding methods rely on architecture-specific designs and do not provide a general test-time scaling mechanism for raster-scan AR token generation. Consequently, it remains unclear how to systematically allocate computation or incorporate structural feedback throughout the visual AR decoding process.

B. Rejection Analysis and Examples

We investigate the rejection ratio at each stage of *GridAR*'s grid-based progressive generation, where a grid is marked as *impossible* if rejected. In the first stage, one grid corresponded to one-quarter of the image ($R_1 = 4$), while in the second stage, each grid covered half of the image ($R_2 = 2$). Table 3 reports the proportion of rejected grids relative to the total number of generated grids at each stage, as well as the percentage of samples in which all grids were rejected. For Text-to-Image Generation, we used Janus-Pro, and for Image Editing, we used EditAR.

Across both tasks, rejections were rare in the first stage but occurred more frequently in the second stage. This pattern suggests that the verifier tended to pass grids when evidence for rejection was definitely uncertain. Figure 8 illustrates qualitative examples for partially rejected candidates.

Cases where all grids within a sample were deemed impossible were infrequent (especially for text-to-image generation, which was quite rare). In these situations, we continued the generation process rather than restarting from scratch, in order to maintain a fair comparison with Best-of-N search, which would otherwise be disadvantaged by additional token generation costs. Nevertheless, in practical deployments of *GridAR*, we recommend different strategies depending on application requirements: for time-sensitive settings, selecting the top-k grids may be more suitable, whereas in scenarios where performance is important, rolling back and regenerating from scratch may offer a more effective alternative.

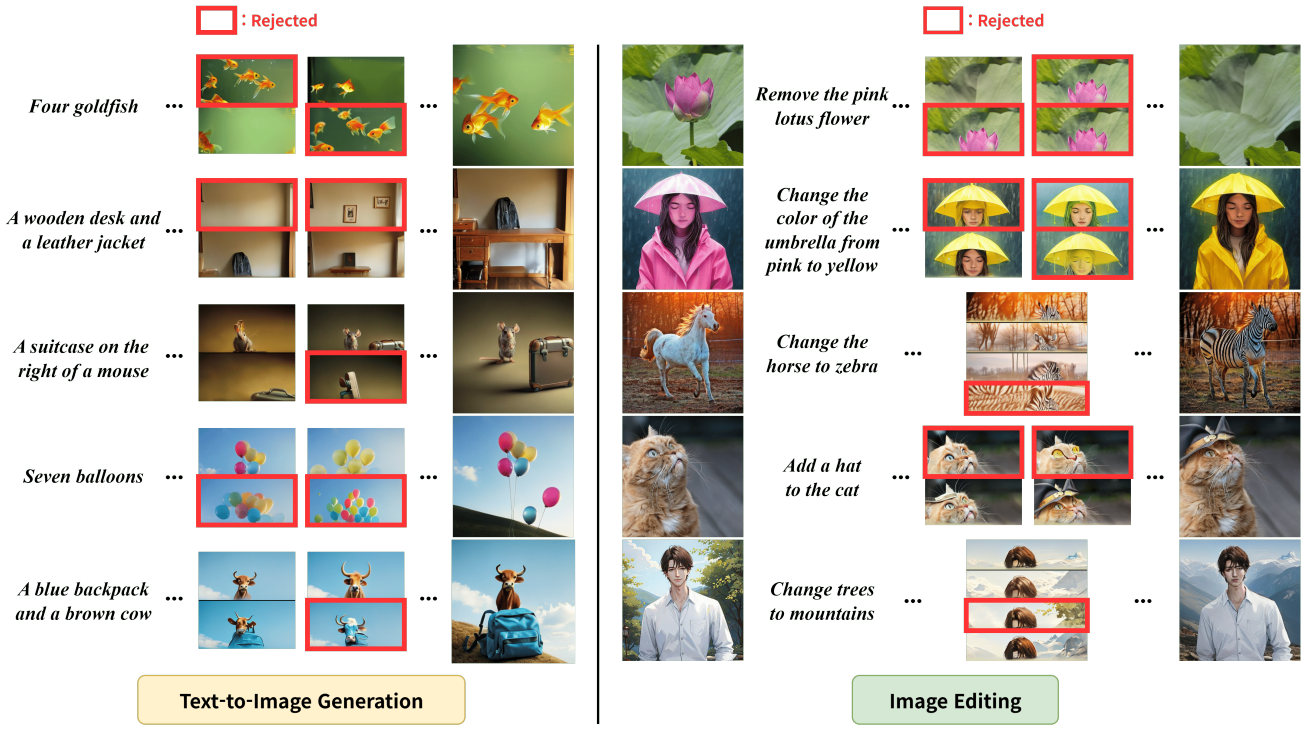


Figure 8. Examples of rejected samples.

Table 3. Rejection statistics for different models across stages, reporting both grid-level rejection rates and fully rejected sample rates.

Task	Model	Stage	Grid-level rejection rate (%)	Fully rejected sample rate (%)
Text-to-Image Generation	**Janus-Pro**	First-stage (R_1=4)	2.545	0.870
		Second-stage (R_2=2)	21.406	3.709
Text-to-Image Generation	**LlamaGen**	First-stage (R_1=4)	10.753	1.271
		Second-stage (R_2=2)	34.305	7.167
Image Editing	**EditAR**	First-stage (R_1=4)	28.545	20.429
		Second-stage (R_2=2)	43.252	21.286

C. Verifier Accuracy and Its Impact on Overall Performance

In this section, we examine whether vision-language models can effectively serve as zero-shot verifiers ψ to assess partial image candidates within the *GridAR* framework. Specifically, we aim to determine whether these models are capable of identifying cases where a partial image is already insufficient to fulfill the given prompt. To support this evaluation, we construct a human-annotated benchmark comprising 720 partial images: 480 candidates from the R_1-grid and 240 from the R_2-grid. Each sample is labeled to indicate whether the prompt can still be satisfied given the visible content. Annotation examples are shown in Figure 9. We evaluate four vision-language models as verifiers in this setting: GPT-4.1, GPT-4o-mini, MiniCPM-V 4.5 [32] (open-source), and GLM-4V [12] (open-source).

As confirmed in Table 4, these models demonstrate notable verification capabilities. In particular, MiniCPM-V 4.5 and GLM-4V, both in the 8B to 9B parameter range, achieve F1 scores between 71.97 and 80.41. As expected, verifiers show higher discriminative performance on the R_2-row grid, which provides richer visual content. We also report final image quality of each verifier in the T2I-CompBench++ benchmark, when adopted in *GridAR* (N=4). Image quality is measured as the average score across all evaluation dimensions. Even when using the relatively lower-performing MiniCPM-V 4.5 verifier, *GridAR* outperforms the Best-of-N baseline (N = 8), showing an improvement of 11.7%. We also observe a trend: as

Table 4. F1 scores of each verifier on human-annotated partial image candidates, and corresponding final image quality when applied to *GridAR* (N=4). Improvements are reported relative to the Best-of-N baseline (N=8).

Verifier ψ	R_1-row grid (R_1= 4) F1 score	R_2-row grid (R_2= 2) F1 score	**Final Image Quality**
MiniCPM-V 4.5	71.97	76.40	0.5519 (+11.7%)
GPT-4.1	90.17	88.06	0.5654 (+14.4%)
GPT-4o-mini	74.22	83.29	0.5647 (+14.2%)
GLM-4V	72.96	80.41	0.5603 (+13.4%)

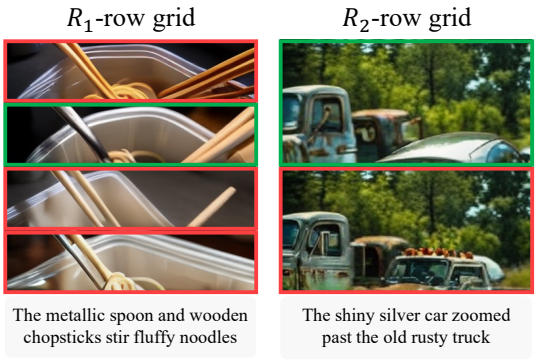


Figure 9. Annotation examples of partial image candidates.

verifier accuracy improves, the final image quality increases accordingly. Given the rapid progress in vision-language models, we expect *GridAR* to evolve into a more powerful framework in the near future. An analysis of failure cases, where verifiers make incorrect predictions, is provided in Appendix E.

D. Additional Experimental Results

D.1. Human Evaluation Results

To evaluate the effectiveness of *GridAR*, we conduct a user study and observe a strong correlation with human preference. Each participant is asked to assess images generated by Janus-Pro. For every sample, three images are presented: two produced by Best-of-N strategy (N=1 and N=8) and one generated by *GridAR* (N=8). In that study, a total of 20 participants are instructed to select the image that best aligns with the given prompt. We test 50 randomly selected samples, drawn from T2I-CompBench++, ensuring a balanced distribution of 4-6 samples per category. Figure 10 reports the proportion of times each method is chosen as the best. *GridAR* demonstrates a markedly stronger human preference compared to the Best-of-N baselines under the same N setting.

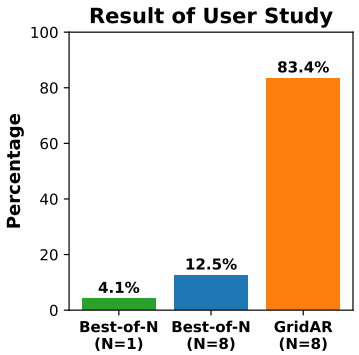


Figure 10. User study results showing selection proportions across three generation methods. *GridAR* (N=8) is strongly preferred over the Best-of-N baselines.

D.2. Image Editing Quantitative Results

To evaluate *GridAR* in the image editing setting, we extend the candidate selection process to incorporate both the source image and the editing instruction. A grid is accepted only when it simultaneously satisfies instruction-following fidelity and background-preservation, consistent with established evaluation protocols in prior image editing evaluation practice [17]. Additionally, we apply a single prompt reformulation at the generation halfway point to enhance instruction consistency, after which selection continues under the same dual-criterion rule.

Table 5 presents the full quantitative results on PIE-Bench across all baselines and autoregressive test-time scaling variants. We report standard background-preservation metrics (PSNR, LPIPS, MSE, SSIM), the structure distance metric used by DINO, and CLIP-based semantic similarity score for both whole-image and edited-region image.

Across all baselines evaluated under a unified Best-of-N configuration, *GridAR* outperforms both diffusion- and autoregressive-based baselines on every metric, demonstrating superior structural fidelity, stronger background preservation, and competitive or superior CLIP similarity. Under comparable CLIP similarity to the stronger Best-of-N baseline (N=8), *GridAR* delivers substantially better structural consistency and background preservation, achieving the lowest Structure Distance (36.632) and the strongest reconstruction metrics (PSNR 21.950, LPIPS 101.700, MSE 103.896, SSIM 75.815). This indicates that *GridAR* provides a more efficient and effective test-time scaling strategy, improving semantic and structural fidelity without relying on increased sampling budgets.

Table 5. Quantitative image-editing results on PIE-Bench. EditAR + *GridAR* (N=4) achieves the strongest overall performance, obtaining the lowest structure distance, best background preservation, and competitive CLIP similarity while outperforming or matching larger Best-of-N baselines.

Method	Structure Distance (↓)	Background Preservation				CLIP Similarity	
		PSNR (↑)	LPIPS (↓)	MSE (↓)	SSIM (↑)	Whole (↑)	Edited (↑)
Diffusion Models							
InstructPix2Pix + BoN (N=4)	54.903	21.054	146.089	216.340	77.361	24.917	22.271
InstructDiffusion + BoN (N=4)	76.023	21.066	131.994	308.671	77.375	24.760	22.126
MGIE + BoN (N=4)	70.417	21.502	135.859	304.363	78.190	24.792	22.405
Auto-Regressive Models							
EditAR (N=1)	41.580	20.721	125.214	205.538	74.079	24.813	22.092
EditAR + BoN (N=2)	40.890	21.065	118.810	166.652	74.648	25.103	22.247
EditAR + BoN (N=4)	40.948	21.368	113.829	140.533	75.043	25.386	22.436
EditAR + BoN (N=8)	42.837	21.464	111.252	135.404	74.979	**25.628**	**22.626**
EditAR + GridAR (N=4)	**36.632**	**21.950**	**101.700**	**103.896**	**75.815**	25.532	22.501

D.3. Ablation Study

In Table 6 and Table 7, we present a stepwise evaluation to quantify the contribution of each component in *GridAR*. Specifically, the Grid Generation module denotes our progressive, grid-based image generation process, and the Prompt Reformulation module refers to revising the prompt based on intermediate results during generation. As shown in both tables, each component yields clear improvements, with the full combination achieving the best performance.

Table 6. Ablation study: Component-wise evaluation of *GridAR* (N= 4) on the T2I-CompBench++ benchmark with Janus-Pro.

Grid Generation	Prompt Reformul.	Attribute Binding			Object Relationship			Numeracy	Complex
		Color	Shape	Texture	2D Spatial	3D Spatial	Non-Spatial		
		0.6781	0.4026	0.5305	0.2218	0.3024	0.7830	0.4958	0.3971
✓		0.7260	0.4660	0.6030	0.2305	0.3105	0.7830	0.5183	0.3969
	✓	0.7819	0.5685	0.7020	0.2747	0.3390	0.7850	0.5582	0.3987
✓	✓	0.7930	0.5807	0.7275	0.2968	0.3594	0.7867	0.5531	0.4009

Table 7. Ablation study: Component-wise evaluation of *GridAR* (N= 4) on the PIE-Bench with EditAR.

Grid Generation	Prompt Reformul.	Structure Distance (↓)	Background Preservation				CLIP Similarity	
			PSNR (↑)	LPIPS (↓)	MSE (↓)	SSIM (↑)	Whole (↑)	Edited (↑)
		40.948	21.368	113.829	140.533	75.043	25.386	22.436
✓		37.949	21.782	106.464	108.935	75.452	25.533	22.551
	✓	37.978	21.766	105.909	116.746	75.400	25.507	22.451
✓	✓	36.632	21.950	101.700	103.896	75.815	25.532	22.501

D.4. Grid Configuration Analysis

Our method employs a row-partitioned progressive generation strategy that explores multiple partial candidates in parallel at early stages and retains only those that remain plausible under the verifier. While the main experiments adopt the configuration $(R_1, R_2) = (4, 2)$ for its balance between information and computational cost, here we investigate how the framework behaves under different grid decompositions, including both fewer and additional stages. To examine how the method behaves under varying degrees of decomposition, we evaluate three configurations: (2) as a single step, $(4, 2)$ as the default two-stage setup, and $(8, 4, 2)$ as a three-stage extension obtained by applying the same framework once more. All experiments use MiniCPM-V 4.5 as the verifier.

As shown in Table 8, all configurations outperform the BoN baseline. The single-step configuration (2) achieves an average score of 0.5548, the default two-stage configuration $(4, 2)$ improves this to 0.5686, and the three-stage configuration $(8, 4, 2)$ further increases performance to 0.5706. This monotonic trend $(2) < (4, 2) < (8, 4, 2)$ reflects the benefit of finer decomposition. However, when verification is applied too early - as in the $(8, 4, 2)$ configuration, where verification occurs already at the first $1/8$ of the image - the structural signal available to the verifier is limited, resulting in only modest gains relative to the additional computational cost introduced by deeper configurations. Overall, these results confirm that our approach generalizes across a wide range of grid configurations and offers a controllable trade-off between verification granularity and compute overhead.

Table 8. Evaluation of *GridAR* (N=8) with Janus-Pro on the T2I-CompBench++ benchmark under different grid configurations, using MiniCPM-V 4.5 as the verifier.

Grid Config.	Attribute Binding			Object Relationship			Numeracy	Complex
	Color	Shape	Texture	2D Spatial	3D Spatial	Non-Spatial		
None	0.7235	0.4177	0.5600	0.2430	0.3165	0.7853	0.5068	0.4013
(2)	0.8006	0.5810	0.7126	0.2715	0.3389	0.7870	0.5491	0.3975
(4, 2)	0.8195	0.5901	0.7363	0.3021	0.3630	0.7823	0.5546	0.4008
(8, 4, 2)	0.8099	0.5953	0.7458	0.3084	0.3627	0.7840	0.5622	0.3963

D.5. Comparison of Prompt Reformulation Strategies

In Table 9, we compare two prompt reformulation strategies: prompt replacement and our proposed three-way CFG. As shown in the table, *GridAR* (N=4) with the prompt replacement strategy achieves an average score of 0.5623, while *GridAR* (N=4) with the three-way CFG achieves 0.5654. The proposed three-way CFG can therefore be effectively adopted within the layout-specified prompt reformulation. However, both strategies outperform the Best-of-N (N=8) baseline, indicating that either approach can be used successfully in practice.

Table 9. Evaluation of *GridAR* (N=4) with Janus-Pro on the T2I-CompBench++ benchmark under different prompt reformulation strategies, using GPT-4.1 as the verifier.

Reformulation Strategy	Attribute Binding			Object Relationship			Numeracy	Complex
	Color	Shape	Texture	2D Spatial	3D Spatial	Non-Spatial		
Three-way CFG	0.8050	0.6014	0.7268	0.2833	0.3503	0.7887	0.5684	0.3992
Prompt Replacement	0.7930	0.5807	0.7275	0.2968	0.3594	0.7867	0.5531	0.4009

D.6. Detailed Cost-Performance Analysis

We provide a more detailed breakdown of the wall-clock time in our framework to better understand the cost-performance trade-off. In Table 10, wall-clock time is measured on RTX 3090 GPUs and averaged over eight processed batches. To obtain an accurate measurement, we employ the open-source verifier MiniCPM-V 4.5 rather than an API-based verifier. Although our framework incurs additional computation overhead in the verifier and decoding stages, the results indicate that *GridAR* offers a more favorable cost-performance Pareto frontier when comparing GridAR (*N*=4) against the Best-of-*N* (*N*=8) baseline.

D.7. Comparison with Other AR-Series Test-Time Scaling Methods

Recently, test-time scaling approaches have been proposed that are designed for other types of autoregressive (AR) models - such as TTS-VAR [8] for next-scale prediction AR models and PARM [34] for masked-modeling AR models. Since these approaches are non-trivial to apply to raster-scan decoding AR models, we do not include them in the main set of comparisons. However, to more comprehensively evaluate our framework, we additionally compare our framework against these test-time scaling baselines on the T2I-Compbench++ and GenEval benchmarks (Table 11). Notably, our test-time scaling approach achieves the best image quality in most cases and demonstrates consistently strong performance across diverse aspect

Table 10. Detailed wall-clock time breakdown and final image quality comparison between Best-of-*N* baselines and our GridAR framework with a MiniCPM-V 4.5 verifier.

Method	**Total Wall-Clock Time**	**Wall-Clock Time**			**Final Image Quality**
		Decoding time	Verifier processing time	ORM Processing time	
Best-of-*N* (*N*=4)	23.85 s	21.97 s	0 s	1.88 s	0.4764
Best-of-*N* (*N*=8)	49.05 s	45.14 s	0 s	3.91 s	0.4943
GridAR + MiniCPM-V 4.5 (*N*=4)	35.53 s	27.48 s	6.15 s	1.90 s	0.5519
GridAR + MiniCPM-V 4.5 (*N*=8)	68.41 s	54.66 s	10.01 s	3.74 s	0.5686

prompts, except in a few dimensions where large backbone-quality differences dominate the outcomes. Taken together, these results demonstrate that raster-scan decoding AR models are fully capable of effective and competitive test-time scaling.

Table 11. Comparison with other AR test-time scaling approaches on T2I-Compbench++ and GenEval. The GenEval "Overall" score represents the average across all evaluation dimensions.

Method	**T2I-Compbench++**								**GenEval**			
	Color	Shape	Texture	2D Spatial	3D Spatial	Non-Spatial	Numeracy	Complex	Overall	Count	Position	Color Attr.
Janus-Pro	0.5388	0.3476	0.4357	0.1607	0.2806	0.7733	0.4467	0.3870	0.79	0.59	0.77	0.65
+ Best-of-*N* (*N*=8)	0.7235	0.4177	0.5600	0.2430	0.3165	0.7853	0.5068	0.4013	0.86	0.76	0.86	0.72
+ **GridAR** (*N*=8)	**0.8172**	**0.6174**	**0.7408**	**0.3214**	0.3587	**0.7930**	0.5932	**0.4050**	**0.88**	**0.79**	**0.92**	**0.73**
Infinity (*next-scale prediction*)	0.7421	0.4557	0.6034	0.2279	0.4023	0.7820	0.5479	–	0.69	0.59	0.20	0.62
+ Best-of-*N* (*N*=8)	0.7950	0.5439	0.6886	0.2545	0.4205	0.7870	0.6090	–	0.74	0.68	0.21	0.67
+ TTS-VAR (*N*=8)	0.8073	0.5914	0.7121	0.2644	**0.4302**	0.7880	**0.6340**	–	0.75	0.74	0.22	0.68
Show-o (*masked modeling*)	–	–	–	–	–	–	–	–	0.53	0.49	0.11	0.28
+ PARM (*N*=20)	–	–	–	–	–	–	–	–	0.67	0.68	0.29	0.45

D.8. Impact of Outcome Reward Model

In our experiments, we employ Qwen2.5-VL-7B [1] as the outcome reward model (ORM), which provides a fair comparison between our method and existing test-time scaling baselines. Nevertheless, because GridAR incorporates an additional internal verifier, we further examine whether the observed performance improvements could simply stem from the ORM being weaker than the internal verifier. To isolate this factor, we analyze the effect of ORM scaling, as summarized in Table 12.

Even when the ORM is scaled from 7B to 72B, the performance gain is marginal - a 0.3% improvement in the Best-of-N (N=4) setting and a 0.1% improvement for GridAR + MiniCPM-V 4.5 (N=4). Moreover, the relative ordering across methods remains unchanged. These results confirm that our performance gains do not arise merely from the internal verifier being stronger than the ORM.

E. Failure Case Analysis and Future Directions

GridAR may potentially fail in certain scenarios during grid-based progressive generation and prompt reformulation processes. To understand such cases, we analyze verifier errors identified during the accuracy evaluation presented in Appendix C. We also observe cases where prompt reformulation does not lead to natural image generation. These observations motivate several future directions, which we outline in this section.

During grid-based progressive generation, the verifier occasionally encounters partial candidates whose visible region consists solely of background, particularly in R_1-row grid (Figure 11 (a)). These cases are not inherently erroneous within the *GridAR* framework; in both image editing and image generation settings, such background-only regions may still align with prompts describing atmosphere, environment, or other contextual details, and thus remain valid assessment targets. However, when prompts lack explicit references to such background elements, we observe that verifiers sometimes reject these candidates as unrelated. Stronger models, such as GPT-4.1, tend to infer that foreground content may appear in subsequent lower-grid regions and therefore avoid premature rejection. In contrast, open-source vision-language models often over-reject these background-only candidates. Although such behavior is not common in our framework - as shown in the verifier quantitative evaluation - it remains one of the failure modes observed when weaker verifiers are used.

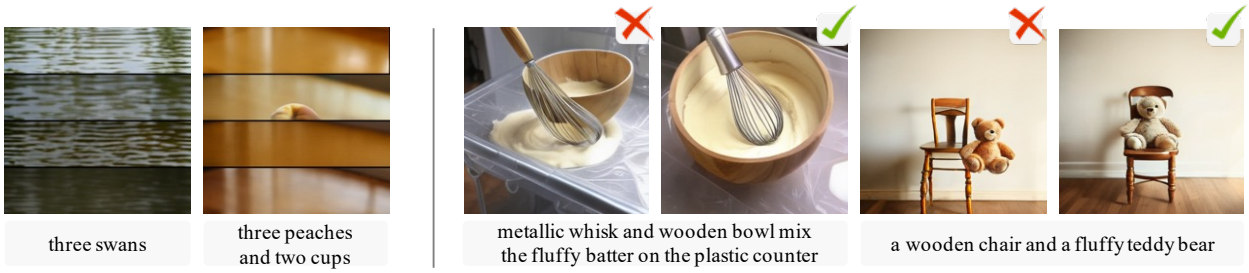
In the prompt reformulation stage, we occasionally observe cases, illustrated in Figure 11 (b), where attempts to satisfy the prompt while considering the intermediate layout lead to implausible scene layouts. Although the reformulated prompt remains faithful to the original intent, the resulting layout can reflect spatial arrangements that are unnatural or physically

Table 12. Impact of ORM scale on the performance of *GridAR* (*N*=4) with Janus-Pro on the T2I-CompBench++ benchmark, assessed using MiniCPM-V 4.5 as the verifier.

Method	ORM	Attribute Binding			Object Relationship			Numeracy	Complex
		Color	Shape	Texture	2D Spatial	3D Spatial	Non-Spatial		
Best-of-*N* (*N*=4)	Qwen2.5-VL-7B	0.6781	0.4026	0.5305	0.2218	0.3024	0.7830	0.4958	0.3971
	Qwen2.5-VL-72B	0.6819	0.4114	0.5314	0.2216	0.2964	0.7873	0.4983	0.3958
GridAR + MiniCPM-V 4.5 (*N*=4)	Qwen2.5-VL-7B	0.7959	0.5681	0.7265	0.2655	0.3522	0.7817	0.5302	0.3947
	Qwen2.5-VL-72B	0.8007	0.5707	0.7306	0.2630	0.3474	0.7833	0.5300	0.3957

unlikely, as the model pushes the layout to align with the prompt. Such cases are infrequent but highlight a failure mode in which prompt reformulation comes at the expense of scene plausibility.

Future directions To address these issues, *GridAR* could be extended in a more adaptive manner. For instance, if the prompt suggests that key objects are likely to appear in the lower part of the image, reducing the R_1 value or initiating progressive generation from a later stage may allow the verifier to access a broader and more meaningful context. Furthermore, we plan to explore more advanced prompt reformulation strategies that not only reason over the layout but also assess whether the resulting scenes are physically plausible and determine when reformulating the prompt would result in infeasible generations.



(a) Grid-based generation: Background-only grids (b) Prompt reformulation: Implausible layout issue

Figure 11. Examples of failure cases. (a) Cases where some verifiers may prematurely reject candidates when only background regions are visible in the early grids. (b) Unrealistic layouts that may be induced when attempting to satisfy the prompt while being aware of the intermediate layout.

F. Detailed Experimental Setup

Baselines We primarily compare against a simple Best-of-N (BoN) baseline driven by an outcome reward model (ORM). To contextualize text-to-image performance, we report results alongside state-of-the-art diffusion systems (SDXL [22], PixArt-α [5], DALL·E 3 [2], Stable Diffusion 3 [9], and FLUX.1 [18]) and contemporary autoregressive generators (Lumina-mGPT [20], Emu3 [28], Show-o [31], and Infinity [13]). For instruction-guided image editing, we benchmark diffusion-based editors - InstructPix2Pix [3], InstructDiffusion [11], MGIE [10] - and an autoregressive editor, EditAR [21], which generates tokens sequentially.

Experimental environment We conduct all experiments on RTX 3090 GPUs. The base AR model and the ORM run on 2× RTX 3090 GPUs, with additional GPUs allocated to the verifiers - one extra GPU for MiniCPM-V 4.5 and two extra GPUs for GLM-4V. The batch size is fixed to 2, and for fair comparison with the Best-of-N baseline, the first row is generated identically across all methods. Unless otherwise noted, the sampling temperature is 1.0.

Evaluation protocols Image editing is evaluated on PIE-Bench [15], which comprises 700 images across nine editing scenarios. Each sample includes a source image and a natural-language instruction, together with auxiliary annotations (source prompt, target prompt, and a ground-truth edit mask). Instruction following is measured via CLIP similarity [14] between the edited image and the target prompt, computed over the full image and, separately, within the annotated edit region. Source preservation is assessed using a structure-aware distance derived from DINO-ViT [4] features ("structure distance") and standard fidelity/perceptual metrics - PSNR, SSIM [29], LPIPS [33], and MSE.

G. Qualitative Results on Image Generation and Editing

Figure 12. Additional qualitative results for text-to-image generation.

Figure 13. Additional qualitative results for text-to-image Generation.

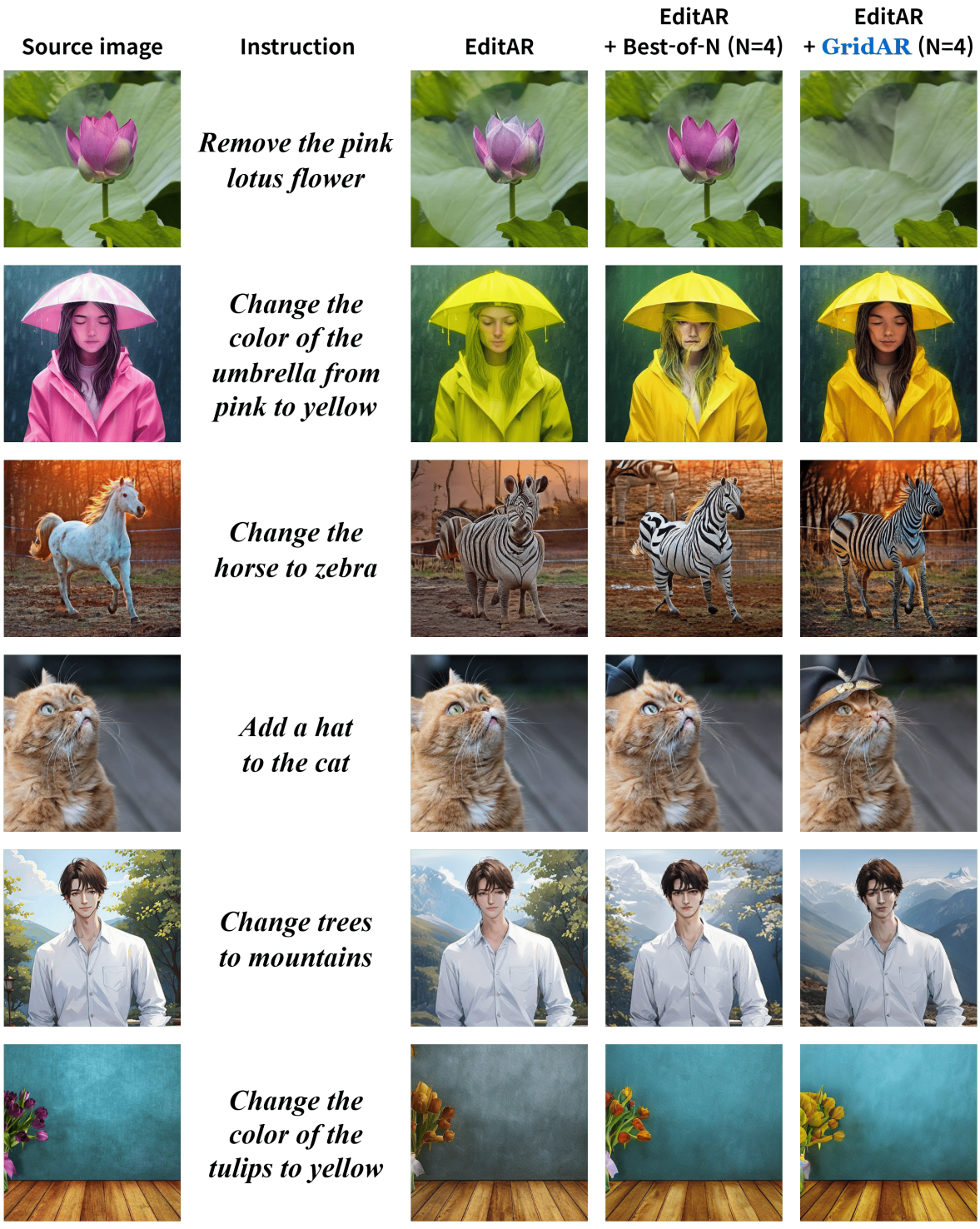


Figure 14. Additional qualitative results for image editing.

H. Prompt Templates

Full prompts for image generation and editing tasks are available in our source code repository.

H.1. Prompts for Text-to-Image Generation

Prompt for Selection of First Rows (Quarters) in Text-to-Image Generation

You are given a single image consisting of 4 contiguous horizontal quarters (from top to bottom: quarter 1, quarter 2, quarter 3, quarter 4).

Each quarter shows the top quarter (upper 1/4 crop) of a different full image generated from the same text prompt. The lower three-quarters of each full image are not shown in this composite. Since only the top part is visible, some quarters may show only the background without any objects. In other cases, objects may appear only partially, with the rest continuing into the unseen lower part of the image.

The text prompt is: "{}".

For each of the 4 quarters, answer strictly with either "possible" or "impossible" (lowercase, no punctuation).
Output must contain exactly 4 words, separated by commas, in order: quarter 1, quarter 2, quarter 3, and quarter 4.
Example format: possible, impossible, possible, impossible

Say "impossible" for a quarter only if it is certain that the prompt cannot be satisfied.

If there is any reasonable way the prompt could still be satisfied, say "possible".

Prompt for Layout-Specified Prompt Reformulation in Text-to-Image Generation

Rewrite the prompt "{}" considering the given partially generated images, so that it fully describes the final image layout with the correct total number of objects and accurately satisfies the original prompt, helping the model complete the remaining half.

OUTPUT
- Output exactly one sentence: either the refined prompt or the unchanged original.
- Begin with the original text prompt; when refining, append the clause after a comma.

EXAMPLES
Original: "A photo of eight bears"; Visible: three bears on the top →
Output: "A photo of eight bears, three on the top and five on the bottom."

Original: "A photo of one chicken"; Visible: only the upper half of the same chicken →
Output: "A photo of one chicken" (unchanged; continuation only)

...

Prompt for Outcome Reward Model in Text-to-Image Generation [34]

This image is generated by a prompt: {}. Does this image accurately represent the prompt? Please answer yes or no without explanation.

H.2. Prompts for Image Editing

Prompt for Selection of First Rows (Quarters) in Image Editing

You are given two images:
(A) the source image to be edited, and
(B) a composite image consisting of 4 contiguous horizontal quarters (from top to bottom: quarter 1, quarter 2, quarter 3, quarter 4).
Each quarter in (B) shows the top quarter (upper 1/4 crop) of a different full image produced by applying the same editing instruction to the source image. The lower three-quarters of each full image are not shown. Since only the top part is visible, some quarters may show only the background without any objects, while others may show objects partially, with the rest continuing into the unseen lower part of the image.

The editing instruction is: "{}".

For each of the 4 quarters, make two independent judgments:
1) instruction_following — does the visible part allow the edited image to plausibly satisfy the instruction?
2) source_preservation — do the visible regions unrelated to the instruction look reasonably consistent with the source image?

OUTPUT FORMAT (STRICT):
Return a single-line JSON object with exactly two keys:
{{"instruction_following": "<q1,q2,q3,q4>", "source_preservation": "<q1,q2,q3,q4>"}}
- For each key, the value is a lowercase string of exactly four words, each either "pass" or "fail", separated by a comma and a single space, in order: quarter 1, quarter 2, quarter 3, quarter 4.
- Example: {{"instruction_following": "pass, fail, pass, fail", "source_preservation": "pass, pass, fail, pass"}}
- No extra keys, punctuation, notes, or explanations.

Prompt for Layout-Specified Prompt Reformulation in Image Editing

Rewrite the editing instruction "{}" into an extremely concise command that guides the model to finish the edit in the lower half of the image, based on the visual cues from the upper half.

OUTPUT
- Output exactly one sentence.
- If unchanged, output the original instruction verbatim.
- Otherwise, output a single rewritten sentence (no quotes) that integrates visible cues and preserves counts/scope.

EXAMPLES
Original: "Change the animal from a cat to a dog"; Visible: dog head visible →
Output: "Draw the dog's body."

Original: "Change the color of the horse from white to golden"; Visible: golden head visible →
Output: "Make the rest of the horse golden."

...

Prompt for Outcome Reward Model in Image Editing

The first image is the source image, and the second image is the edited image produced by applying the following instruction: "{}". Does the edited image correctly follow the instruction while preserving the rest of the source image? Please answer yes or no without explanation.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 6

[2] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023. 1, 7

[3] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18392–18402, 2023. 7

[4] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 7

[5] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis, 2023. 7

[6] Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. In *The Twelfth International Conference on Learning Representations*. 1

[7] 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. 1

[8] Zhekai Chen, Ruihang Chu, Yukang Chen, Shiwei Zhang, Yujie Wei, Yingya Zhang, and Xihui Liu. TTS-VAR: A test-time scaling framework for visual auto-regressive generation. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. 1, 5

[9] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, pages 12606–12633. PMLR, 2024. 1, 7

[10] Tsu-Jui Fu, Wenze Hu, Xianzhi Du, William Yang Wang, Yinfei Yang, and Zhe Gan. Guiding instruction-based image editing via multimodal large language models. In *The Twelfth International Conference on Learning Representations*, 2024. 7

[11] Zigang Geng, Binxin Yang, Tiankai Hang, Chen Li, Shuyang Gu, Ting Zhang, Jianmin Bao, Zheng Zhang, Houqiang Li, Han Hu, et al. Instructdiffusion: A generalist modeling interface for vision tasks. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 12709–12720, 2024. 7

[12] Team GLM, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, et al. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *arXiv preprint arXiv:2406.12793*, 2024. 2

[13] Jian Han, Jinlai Liu, Yi Jiang, Bin Yan, Yuqi Zhang, Zehuan Yuan, Bingyue Peng, and Xiaobing Liu. Infinity: Scaling bitwise autoregressive modeling for high-resolution image synthesis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15733–15744, 2025. 7

[14] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021. 7

[15] Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Pnp inversion: Boosting diffusion-based editing with 3 lines of code. *International Conference on Learning Representations (ICLR)*, 2024. 7

[16] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022. 1

[17] Max Ku, Dongfu Jiang, Cong Wei, Xiang Yue, and Wenhu Chen. Viescore: Towards explainable metrics for conditional image synthesis evaluation. *arXiv preprint arXiv:2312.14867*, 2023. 3

[18] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024. 7

[19] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37:56424–56445, 2024. 1

[20] Dongyang Liu, Shitian Zhao, Le Zhuo, Weifeng Lin, Yi Xin, Xinyue Li, Qi Qin, Yu Qiao, Hongsheng Li, and Peng Gao. Lumina-mgpt: Illuminate flexible photorealistic text-to-image generation with multimodal generative pretraining. *arXiv preprint arXiv:2408.02657*, 2024. 7

[21] Jiteng Mu, Nuno Vasconcelos, and Xiaolong Wang. Editar: Unified conditional generation with autoregressive models. *arXiv preprint arXiv:2501.04699*, 2025. 1, 7

[22] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *The Twelfth International Conference on Learning Representations*, 2024. 7

[23] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024. 1

[24] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024. 1

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

[26] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024. 1

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

[28] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024. 7

[29] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 7

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

[31] 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 *The Thirteenth International Conference on Learning Representations*. 7

[32] Tianyu Yu, Zefan Wang, Chongyi Wang, Fuwei Huang, Wenshuo Ma, Zhihui He, Tianchi Cai, Weize Chen, Yuxiang Huang, Yuanqian Zhao, et al. Minicpm-v 4.5: Cooking efficient mllms via architecture, data, and training recipe. *arXiv preprint arXiv:2509.18154*, 2025. 2

[33] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 7

[34] Renrui Zhang, Chengzhuo Tong, Zhizheng Zhao, Ziyu Guo, Haoquan Zhang, Manyuan Zhang, Jiaming Liu, Peng Gao, and Hongsheng Li. Let's verify and reinforce image generation step by step. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 28662–28672, 2025. 1, 5, 11