Parallel Jacobi Decoding for Fast Autoregressive Image Generation

Supplementary Material

A. Additional Quantitative Results

Additional Metrics. We additionally include MLLM-based metrics (VQAScore [25] and TIFA [14]) and human-preference-aligned metrics (ImageReward [60] and HPSv2 [58]). The results in Table 3 show that, even under these additional quality measurements, our method consistently delivers substantial inference acceleration while maintaining competitive generation quality.

Experiments on Janus-Pro 7B. We further conduct experiments on Janus-Pro 7B [5] using the MS-COCO [24] and PartiPrompt [62] datasets. As shown in Table 4, our method achieves the best acceleration performance among all decoding strategies while maintaining competitive image quality.

Compared with the results on Lumina-mGPT, the acceleration gains on Janus-Pro 7B are relatively smaller. This performance gap is mainly due to the lower output resolution (384×384) used by Janus-Pro, which reduces the number of spatial rows available for parallel decoding, thereby limiting the achievable acceleration. Nevertheless, under this more constrained setting, our method still achieves the best speed–quality trade-off across all methods. Notably, under the setting $c = 4$, our approach achieves nearly a 4.41× step reduction, while maintaining CLIP and IS scores comparable to those of the Vanilla AR baseline.

B. Additional Qualitative Results

Qualitative Results on Lumina-mGPT at Multiple Resolutions. In Figure 10, we present qualitative results generated by our method on Lumina-mGPT across different output resolutions, including 512×512, 768×768, and 1024×1024. Across all resolutions, our approach consistently produces visually compelling images with rich textures, coherent structures, and diverse artistic styles. Notably, even at higher resolutions where the generative process becomes more challenging due to the increased number of spatial tokens, our method maintains strong fidelity and detail preservation.

Comparison of Decoding Strategies on LlamaGen. We conduct qualitative comparisons using LlamaGen-XL [49] with four decoding strategies: Vanilla AR, SJD [52], GSD [43], and our proposed PJD method. All methods use the same sampling configuration, including top-$k = 1000$ and a CFG scale of 3.5, to ensure a fair evaluation across different prompts. For our method, we adopt the setting $c = 6$, which balances quality strength and sampling efficiency.

As shown in Figure 11, our method achieves the lowest sampling cost across all prompts, reducing the number of decoding steps by up to **5.4**× compared to the Vanilla AR. This reduction directly translates to significantly faster inference while requiring no modification to the underlying LlamaGen model. Despite the substantial reduction in sampling steps, the image quality of the generated images remains comparable to that of the other methods.

Comparison Between PJD and Vanilla AR on Janus-Pro. Figure 12 presents qualitative comparisons of 384×384 image generation on Janus-Pro across multiple prompts. Each pair shows the output of Vanilla AR (left) and our method (right). All methods use the same sampling configuration, including top-$k = 1000$ and a CFG scale of 5.0. For our method, we adopt the setting $c = 4$. Across all examples, our method produces images that are visually comparable to those generated by the full-step Vanilla AR model, despite requiring substantially fewer decoding steps.

Qualitative Results on Weak-Locality Images. We further provide qualitative examples on weak-locality images, as shown in Figure 13. These examples involve visual structures that rely more heavily on long-range dependencies and global consistency, such as symmetry and cross-region correspondence. Overall, our method is able to generate visually plausible weak-locality images with coherent global layouts. While the overall structure is usually maintained, fine-grained details may exhibit mild degradation. For example, subtle asymmetry or slight inconsistencies may appear in highly structured patterns, as in the rightmost example of Figure 13.

C. Ablation Study about attention mask

The ablation study evaluates the effectiveness of the proposed Row-Causal Mask (RCM). We compare two settings: using RCM and removing it during decoding. In the w/o RCM setting, each token in a row is allowed to attend to all previously generated tokens as well as all draft tokens produced in the current step. Experiments are conducted on both Lumina-mGPT and LlamaGen-XL using the MS-COCO dataset to assess the impact of RCM on decoding efficiency and image quality.

As shown in Table 5, disabling RCM consistently results in slower decoding and lower overall performance across both AR models. When RCM is enabled, the number of

Table 3. Quantitative comparison on MS-COCO with additional quality metrics.

Configuration	Latency (↓)	Step (↓)	VQAScore (↑)	TIFA (↑)	ImageReward (↑)	HPSv2 (↑)
Lumina-mGPT	197.16s	2357	0.8501	0.8545	0.7035	0.2834
w. SJD	52.97s	1056	0.8440	0.8471	0.7004	0.2835
w. GSD	34.36s	698	0.8396	0.8469	0.6854	0.2821
Ours (c=11)	24.78s	371	0.8399	0.8517	0.6924	0.2825
Ours (c=9)	22.62s	328	0.8414	0.8551	0.6782	0.2819
LlamaGen-XL	49.58s	1024	0.6976	0.7317	-0.2951	0.2629
w. SJD	27.55s	631	0.7192	0.7461	-0.2723	0.2636
w. GSD	19.31s	383	0.6950	0.7274	-0.3230	0.2627
Ours (c=8)	13.80s	248	0.7063	0.7458	-0.2947	0.2633
Ours (c=6)	11.84s	213	0.7059	0.7423	-0.3053	0.2633

Table 4. Quantitative comparison of decoding methods on Janus-Pro across the MS-COCO and PartiPrompt datasets. We report the latency and number of sampling steps required to generate a single 384×384 image, together with acceleration factors relative to Vanilla AR.

Dataset	Configuration	Latency (↓)	Step (↓)	Acceleration (↑)		FID (↓)	CLIP-Score (↑)	IS (↑)
				Latency	Step			
MS-COCO	Vanilla AR	17.29s	576	1.00×	1.00×	33.83	32.04	32.32
	w. SJD	8.68s	299	1.99×	1.94×	34.17	32.07	32.58
	w. GSD	7.18s	247	2.41×	2.33×	34.08	32.06	32.18
	Ours (c=6)	6.89s	174	2.51×	3.30×	33.94	32.14	32.76
	Ours (c=4)	**5.92s**	**149**	**2.92×**	**3.86×**	33.96	32.15	32.48
PartiPrompt	Vanilla AR	17.05s	576	1.00×	1.00×	-	32.25	19.47
	w. SJD	8.56s	287	1.99×	2.01×	-	32.28	19.33
	w. GSD	7.02s	235	2.43×	2.45×	-	32.30	19.43
	Ours (c=6)	6.72s	162	2.54×	3.56×	-	32.17	19.51
	Ours (c=4)	**5.83s**	**139**	**2.92×**	**4.41×**	-	32.17	19.48

sampling steps is reduced, leading to clear and consistent improvements in decoding efficiency. At the same time, image quality also improves: FID, CLIP score, and IS are consistently better than their w/o RCM counterparts. This confirms that enforcing row-wise causal dependencies provides more reliable token predictions and reduces error propagation during decoding.

We also show qualitative comparisons on Lumina-mGPT in Figure 14, where the top row corresponds to results with RCM and the bottom row corresponds to results without RCM. The results with RCM are generally more natural and visually coherent, while removing RCM tends to produce less reasonable structures and less consistent details.

D. Analysis on Effectiveness of PJD

Figure 15 visualizes the draft and accepted token counts per decoding step for 768×768 image generation. For SJD, the draft token count is fixed to 50 at every step (blue line), whereas our PJD method adaptively adjusts the number of drafted tokens (red line). Importantly, PJD keeps the draft length below 50 across all steps, ensuring that no excessive number of tokens is processed at once. However, the number of accepted tokens per step differs significantly between the two methods. The blue points indicate that SJD accepts only a small fraction of the 50 drafted tokens at each step, whereas the red points show that PJD consistently accepts many more tokens per step. This suggests that our adaptive drafting strategy makes much more effective use of the computation, resulting in a higher ratio of accepted tokens to drafted tokens.

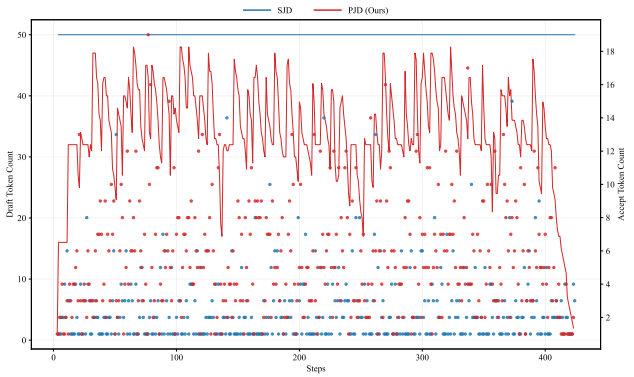

Figure 15. Draft token count (lines) and accepted token count (dots) per decoding step for SJD and PJD. PJD uses fewer draft tokens than SJD while achieving a higher accepted token count at each step.

Table 5. Ablation study on the effectiveness of the Row-Causal Mask (RCM) on the MS-COCO dataset. We compare decoding performance on both Lumina-mGPT and LlamaGen-XL, evaluating the impact of removing RCM. Incorporating RCM consistently improves sampling efficiency while also providing better image quality.

Model	Configuration	Latency (↓)	Step (↓)	Acceleration (↑)		FID (↓)	CLIP-Score (↑)	IS (↑)
				Latency	Step			
Lumina-mGPT	Vanilla AR	197.16s	2357	1.00×	1.00×	30.79	31.31	32.81
	Ours ($c = 11$, w/o RCM)	26.63s	388	7.40×	6.07×	32.65	31.32	31.02
	Ours ($c = 11$)	24.78s	371	7.96×	6.35×	32.38	31.53	31.23
LlamaGen-XL	Vanilla AR	49.58s	1024	1.00×	1.00×	45.02	28.59	22.11
	Ours ($c = 6$, w/o RCM)	12.78s	227	3.88×	4.41×	45.67	28.50	21.26
	Ours ($c = 6$)	11.84s	213	4.19×	4.81×	45.12	28.67	22.07

Figure 10. Qualitative results generated by our method on Lumina-mGPT [27] at different image resolutions. From top to bottom, the rows show images generated at 512×512, 768×768, and 1024×1024 resolutions, respectively. Our method produces high-quality and diverse images across all resolutions, demonstrating strong generative capability and scalability.

E. Evaluation hardware and VRAM usage

All efficiency results are measured on a single RTX 4090 (48GB). We report the peak VRAM usage and the incremental memory cost (ΔM), defined as the maximum additional VRAM during a forward pass. Table 6 shows that our method accelerates inference with comparable VRAM usage.

Table 6. **VRAM usage comparison.**

Configuration	VRAM	ΔM
Lumina-mGPT	16.98G	0.10G
w. SJD	17.63G	0.12G
Ours ($c = 11$)	17.77G	0.14G

F. Prompt Details

For completeness and reproducibility, we provide the text prompts used to generate the four images shown in Figure 1 of the main paper. The corresponding prompts, ordered from left to right and from top to bottom, are given as follows:

1. *Close-up portrait in the style of Van Gogh, thick brush strokes, vivid colors.*
2. *Close-up portrait of a red fox with glowing orange fur, richly textured hair strands, warm sunlight reflections, sharp amber eyes, and high-contrast colors creating a vivid, striking appearance.*
3. *Pixar-quality 3D close-up portrait of a smiling girl, highly detailed hair strands, luminous skin, expressive*

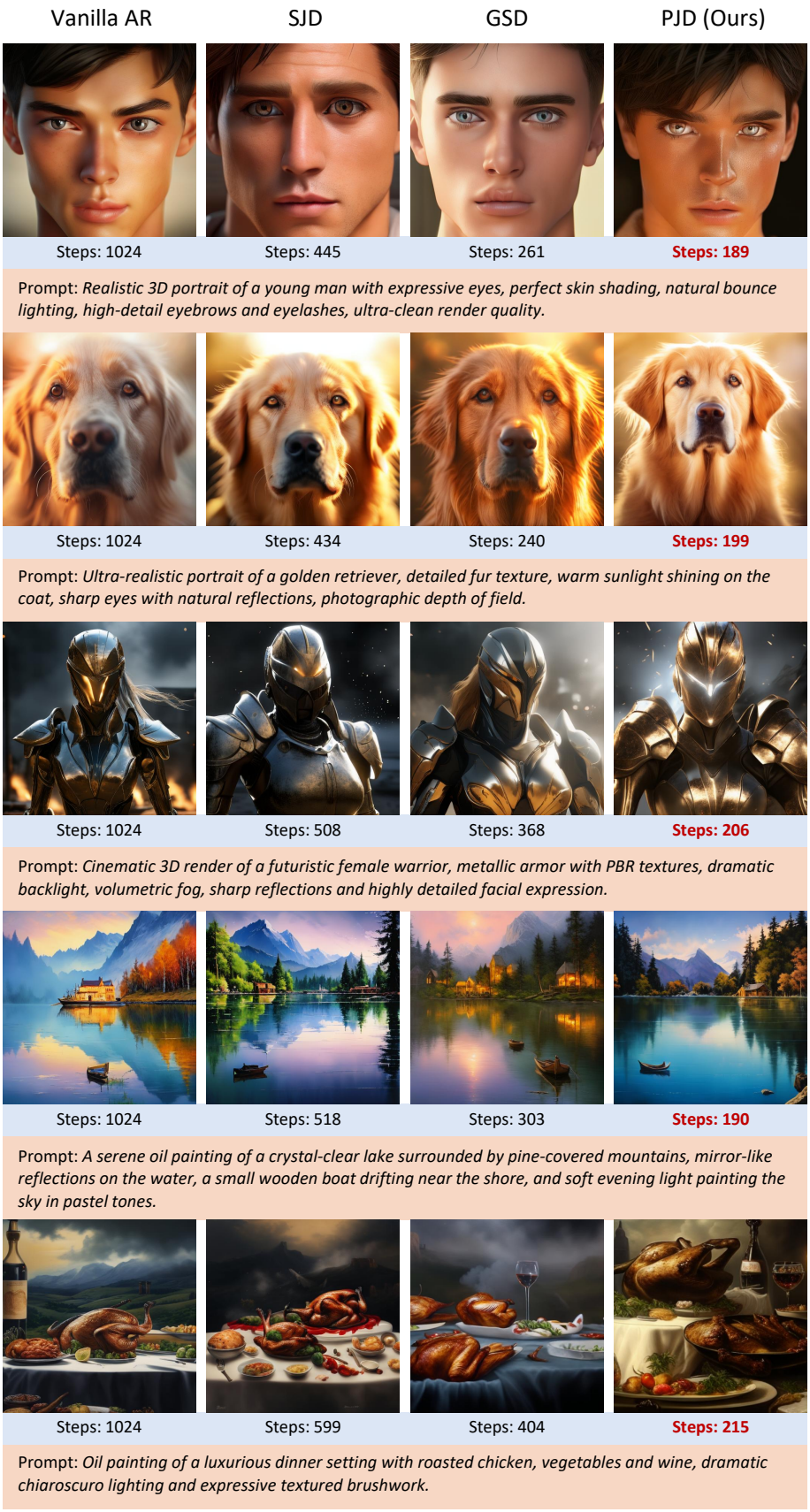

Figure 11. Qualitative comparison of 512×512 image generation results on LlamaGen-XL [49] using four decoding strategies: Vanilla AR, SJD, GSD, and our PJD method. Across all prompts, our approach achieves the fastest generation with the fewest sampling steps.

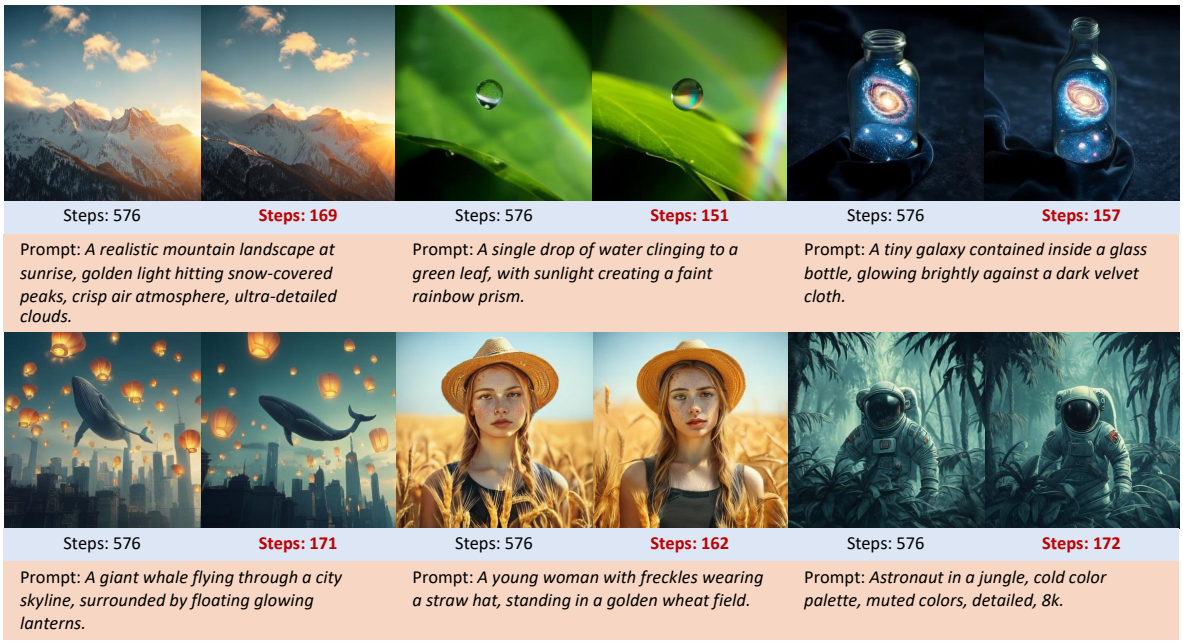

Figure 12. Qualitative comparisons of 384×384 image generation on Janus-Pro [5] across multiple prompts. For each pair, the left image is generated by Vanilla AR and the right image is generated by our method. Our approach significantly reduces the number of sampling steps while preserving comparable image quality.

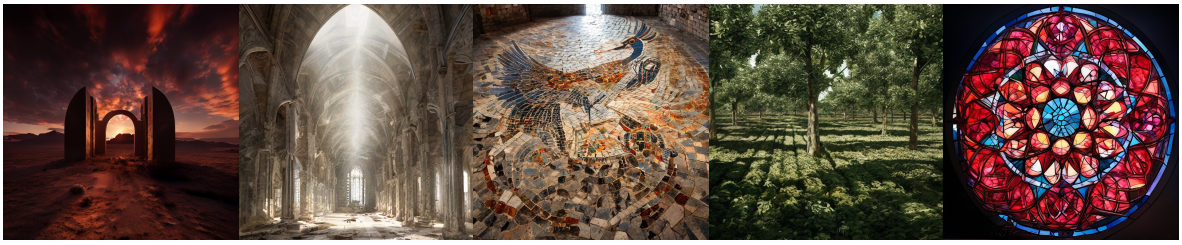

Figure 13. Qualitative weak-locality image generation results of our method on Lumina-mGPT.

eyes with detailed reflections, soft rim lighting, premium animation-film rendering, ultra high resolution.

4. *Baroque-style oil painting portrait of a young woman with voluminous curly hair, glowing soft light on her face, silky skin texture, rich golden highlights, elegant earrings, luxurious fabric with detailed folds, dramatic dark backdrop, ultra-detailed, masterful brushstroke realism, museum-quality portrait.*

G. Discussion and Future Work

While PJD significantly accelerates autoregressive image generation, several limitations suggest promising directions for future work. First, the current acceptance rule is inherited from speculative decoding and does not explicitly account for 2D spatial structure. Designing convergence rules that incorporate local spatial consistency or region-level stability may further improve distributional faithfulness. Second, although PJD dynamically adjusts the number of draft tokens, extremely large draft regions may challenge GPU parallelism or memory capacity. Exploring hardware-aware scheduling or adaptive token grouping could help ensure scalability at very high resolutions. Overall, PJD highlights the benefits of aligning autoregressive decoding with image structure, and further investigation into acceptance rules, hardware scalability, and generalized 2D generation strategies may enable even faster generation while preserving high image quality. More broadly, recent multimodal reasoning studies [8, 9, 15] suggest that structured visual understanding and learning-based optimization may also provide useful perspectives for future extensions of efficient visual generation.

Figure 14. Qualitative comparison with and without the proposed Row-Causal Mask (RCM). The top row shows results with RCM, while the bottom row shows results without RCM.