A Proofs

Theorem 1 (The correctness of speculative Jacobi decoding) The token sampled in each speculative Jacobi iteration satisfies $p_\theta(x|\boldsymbol{x}_{1:i-1}^{(j)})$, where x denotes a token, j denotes the index of iteration, i denotes the token index, and θ denotes the auto-regressive model parameters.

Proof. The main process of speculative Jacobi iteration is decomposed into two cases: (a) obtaining the token sampled in the previous iteration and then accepting it according to an acceptance probability; (b) rejecting the sampled token and resampling a new token according to a calibrated probability. Thus, to prove the correctness of speculative Jacobi decoding, we verify that the conditional probability of a token sampled following the above two cases, alongside the manually designed acceptance and resampling probability, remains $p_\theta(x|\boldsymbol{x}_{1:i-1}^{(j)})$.

For simplicity, by default, we omit the token index i and denote the token category of $\boldsymbol{x}_i^{(j)}$ as x. We denote the condition of token $\boldsymbol{x}_i^{(j)}$ at the j-th Jacobi iteration (*i.e.*, the tokens $\boldsymbol{x}_{1:i-1}^{(j)}$ and model weights θ) to $\mathcal{J}_j$. Thus, the condition of the $(j-1)$-th Jacobi iteration is denoted as $\mathcal{J}_{j-1}$. Thus, we can denote the probability $p_\theta(x|\boldsymbol{x}_{1:i-1}^{(j)})$ as $p(x|\mathcal{J}_j)$, and denote $p_\theta(x|\boldsymbol{x}_{1:i-1}^{(j-1)})$ as $p(x|\mathcal{J}_{j-1})$. We use a random boolean variable r to represent the acceptance. With these notations, the proof is as follows:

First, the acceptance probability on the token category x is manually set as follows:

$$p(r \text{ is true}|x, \mathcal{J}_j, \mathcal{J}_{j-1}) = \min\{1, \frac{p(x|\mathcal{J}_j)}{p(x|\mathcal{J}_{j-1})}\}, \tag{4}$$

and the calibrated resampling probability subsequent to the rejection is set as follows:

$$p(x|r \text{ is false}, \mathcal{J}_j, \mathcal{J}_{j-1}) = \frac{\max\{0, p(x|\mathcal{J}_j) - p(x|\mathcal{J}_{j-1})\}}{\sum_{x'} \max\{0, p(x'|\mathcal{J}_j) - p(x'|\mathcal{J}_{j-1})\}}. \tag{5}$$

Next, we make an assumption that $\mathcal{J}_j$ and x are conditionally independent given $\mathcal{J}_{j-1}$:

$$p(\mathcal{J}_j|x, \mathcal{J}_{j-1}) = p(\mathcal{J}_j|\mathcal{J}_{j-1}) \tag{6}$$

This assumption is reasonable due to the properties of the Jacobi iteration and the auto-regressive paradigm, *i.e.*, with the observation of the sequence $\boldsymbol{x}_{1:i-1}^{(j-1)}$, one of the tokens in $\boldsymbol{x}_{1:i-1}^{(j)}$ (denoted as $\boldsymbol{x}_k^{(j)}$) can be determined by $\boldsymbol{x}_k^{(j)} = f(\boldsymbol{x}_{1:k-1}^{(j-1)}, \theta)$ $(k < i)$ where the function f indicates the prediction-then-sampling of auto-regressive models, so the variable $\boldsymbol{x}_i^{(j)}$ is redundant as one of the conditions in the probability $p(\mathcal{J}_j|x, \mathcal{J}_{j-1})$. Thus, Equ. (6) is reasonable.

Then, with Bayes rule, Equ. (6) has the following equivalence:

$$p(\mathcal{J}_j|x, \mathcal{J}_{j-1}) = p(\mathcal{J}_j|\mathcal{J}_{j-1}) \quad \Leftrightarrow \quad p(x|\mathcal{J}_j, \mathcal{J}_{j-1}) = p(x|\mathcal{J}_{j-1}) \tag{7}$$

Hence, according to Equ. (4) and Equ. (7), the probability that a token category x is sampled in the previous iteration and subsequently accepted can be computed as:

$$\begin{aligned} p(r \text{ is true}, x|\mathcal{J}_j, \mathcal{J}_{j-1}) &= p(x|\mathcal{J}_j, \mathcal{J}_{j-1}) \cdot p(r \text{ is true}|x, \mathcal{J}_j, \mathcal{J}_{j-1}) \\ &= p(x|\mathcal{J}_{j-1}) \cdot \min\{1, \frac{p(x|\mathcal{J}_j)}{p(x|\mathcal{J}_{j-1})}\} \\ &= \min\{p(x|\mathcal{J}_j), p(x|\mathcal{J}_{j-1})\} \end{aligned} \tag{8}$$

With Equ. (8), we can calculate the probability of rejection with the law of total probability on the token categories:

$$\begin{aligned} p(r \text{ is false}|\mathcal{J}_j, \mathcal{J}_{j-1}) &= 1 - p(r \text{ is true}|\mathcal{J}_j, \mathcal{J}_{j-1}) \\ &= 1 - \sum_{x'} p(r \text{ is true}, x'|\mathcal{J}_j, \mathcal{J}_{j-1}) \\ &= \sum_{x'} p(x'|\mathcal{J}_j) - \min\{p(x'|\mathcal{J}_j), p(x'|\mathcal{J}_{j-1})\} \\ &= \sum_{x'} \max\{0, p(x'|\mathcal{J}_j) - p(x'|\mathcal{J}_{j-1})\}. \end{aligned} \tag{9}$$

Figure 10: The images generated by Lumina-mGPT (Liu et al., 2024b) with our acceleration method.

Then, with Equ. (5) and Equ. (9), we get the following equation:

$$\begin{aligned} & p(x|r \text{ is false}, \mathcal{J}_j, \mathcal{J}_{j-1}) \cdot p(r \text{ is false}|\mathcal{J}_j, \mathcal{J}_{j-1}) \\ & = \frac{\max\{0, p(x|\mathcal{J}_j) - p(x|\mathcal{J}_{j-1})\}}{\sum_{x'} \max\{0, p(x'|\mathcal{J}_j) - p(x'|\mathcal{J}_{j-1})\}} \cdot \sum_{x'} \max\{0, p(x'|\mathcal{J}_j) - p(x'|\mathcal{J}_{j-1})\} \\ & = \max\{0, p(x|\mathcal{J}_j) - p(x|\mathcal{J}_{j-1})\}. \end{aligned} \tag{10}$$

Since

$$\forall a \in \mathbb{R}, b \in \mathbb{R}, \quad a = \min\{a, b\} + \max\{0, a - b\}, \tag{11}$$

we can decompose $p(x|\mathcal{J}_j)$ as follows:

$$p(x|\mathcal{J}_j) = \min\{p(x|\mathcal{J}_j), p(x|\mathcal{J}_{j-1})\} + \max\{0, p(x|\mathcal{J}_j) - p(x|\mathcal{J}_{j-1})\}. \tag{12}$$

With Equ. (8), Equ. (10) and Equ. (12), we can compute:

$$\begin{aligned} p(x|\mathcal{J}_j) &= \min\{p(x|\mathcal{J}_j), p(x|\mathcal{J}_{j-1})\} + \max\{0, p(x|\mathcal{J}_j) - p(x|\mathcal{J}_{j-1})\} \\ &= p(x|\mathcal{J}_{j-1}) \cdot p(r \text{ is true}|x, \mathcal{J}_j, \mathcal{J}_{j-1}) \\ &\quad + p(r \text{ is false}|\mathcal{J}_j, \mathcal{J}_{j-1}) \cdot p(x|r \text{ is false}, \mathcal{J}_j, \mathcal{J}_{j-1}). \end{aligned} \tag{13}$$

According to Equ. (13), the conditional distribution $p(x|\mathcal{J}_j)$ can exactly represent (a) obtaining the token sampled in the previous iteration and then accepting it according to an acceptance probability; (b) rejecting the sampled token and resampling a new token according to a calibrated probability. In conclusion, the token sampled in each speculative Jacobi iteration satisfies $p_\theta(x|\boldsymbol{x}_{1:i-1}^{(j)})$.

B More Qualitative Results

In Fig. 10, we showcase more generated images with Lumina-mGPT accelerated by our method. These results illustrate that our method functions well on the image contents including humans,

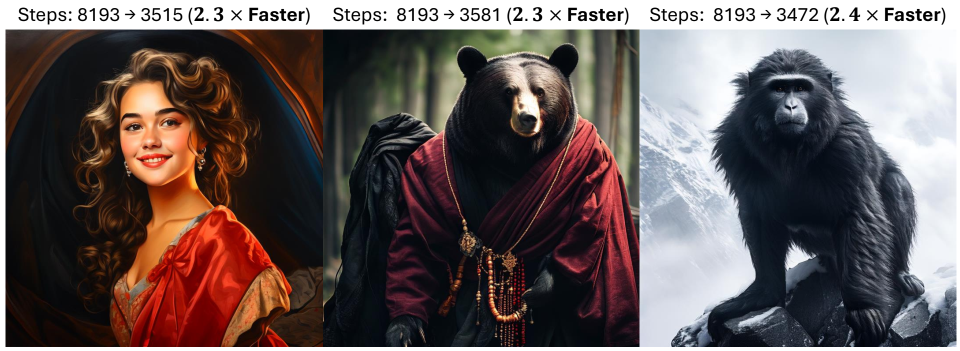

Figure 11: The images generated by Emu3 (BAAI, 2024) with our acceleration method.

animals, and landscapes. Recently, a new powerful auto-regressive model, Emu3 (BAAI, 2024), has been released. We also explore our method on Emu3 for text-to-image generation, and we find it still leads to great step compression, shown in Fig. 11. We leave the quantitative results of Emu3 for future work.

We have included additional qualitative results for Lumina-mGPT and Anole in the supplementary material of the revised paper, specifically in Fig. 15 and Fig. 16, and we report both the steps and latency. According to the reported latency and step compression in these figures, our SJD outperforms other decoding methods while maintaining visual quality. Furthermore, spatial token initialization can further enhance the acceleration of our SJD. Additionally, we observe that Anole exhibits significantly higher image diversity compared to Lumina-mGPT. Despite the fixed random seed, it remains challenging for Anole to generate similar images due to the differences among the decoding methods.

C Inference Latency

In addition to reporting the step compression ratio, we also report the practical latency of SJD on servers. We set the batch size as 1 for testing, and report the latency of the accelerated Lumina-mGPT 7B excluding the pre- and post-processing operations. For 768×768 image generation (the number of generated tokens is at least 2357), we perform the experiments on one RTX 4090 GPU. For 1024×1024 image generation (the number of generated tokens is at least 4165), we perform the experiments on one A100 GPU. In these settings, the latency of Lumina-mGPT with and without our method is presented in Fig. 12. Our method significantly accelerates the auto-regressive image generation.

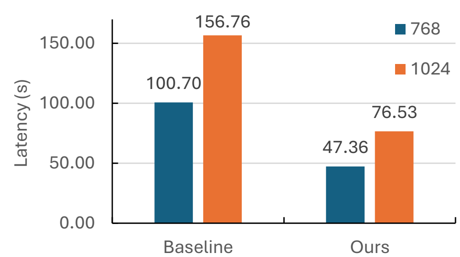

Figure 12: The latency of Lumina-mGPT on generating 768×768 and 1024×1024 images without or with our method.

D More Quantitative Results

Results on more models. In addition to Anole (Chern et al., 2024) and Lumina-mGPT (Liu et al., 2024b), we evaluate our method with the text-to-image LlamaGen (Sun et al., 2024a). This model adopts a two-stage training strategy: (a) stage1: LlamaGen is first trained on a subset of LAION-COCO (LAION, 2022) (50M 256×256 images); (b) stage2: it is then fine-tuned on 10M high aesthetic quality internal data with a resolution of 512×512. In Tab. 3, we evaluate our method with the two versions of LlamaGen. The results show that our method can still accelerate this model without sacrificing the visual quality. However, in comparison to the experiments conducted on Lumina-mGPT and Anole, the acceleration ratios on LlamaGen are lower. We hypothesize that this discrepancy is attributed to the model size (LlamaGen has 3.1B model parameters while Lumina-mGPT has 7B parameters), as some existing works for multi-token prediction demonstrate that the

Table 3: The evaluation of LlamaGen (Sun et al., 2024a) with or without our method on MSCOCO2017 (Lin et al., 2014) and Parti-prompt (Yu et al., 2022).

Dataset	**Configuration**	Acceleration (↑) Latency	Acceleration (↑) Step	FID (↓)	CLIP-Score (↑)
COCO	LlamaGen-stage1	1.00×	1.00×	28.54	30.87
	LlamaGen-stage1 + Ours	1.56×	1.63×	29.00	30.82
	LlamaGen-stage2	1.00×	1.00×	56.21	28.26
	LlamaGen-stage2 + Ours	1.54×	1.63×	57.02	28.33
Parti	LlamaGen-stage1	1.00×	1.00×	-	30.22
	LlamaGen-stage1 + Ours	1.57×	1.73×	-	30.29
	LlamaGen-stage2	1.00×	1.00×	-	28.14
	LlamaGen-stage2 + Ours	1.62×	1.69×	-	28.16

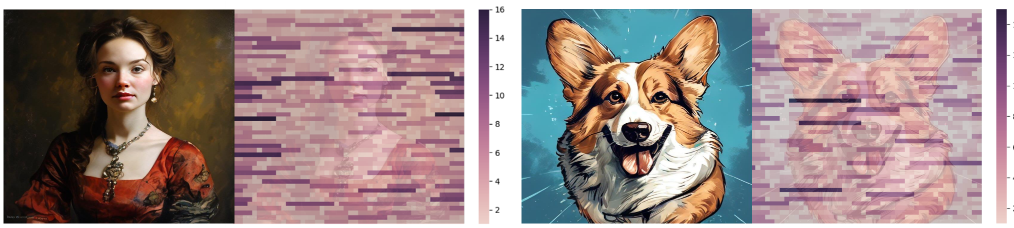

Figure 13: The visualization of the accelerated tokens on 2D space.

model size has a great influence on the effectiveness of acceleration (Gloeckle et al., 2024). We leave this investigation to future work.

We further compare SJD to other decoding methods on Anole (Chern et al., 2024). As shown in Tab. 4 and Tab. 5, consistent with the results on Lumina-mGPT, SJD with spatial token initialization can create larger acceleration ratios than other decoding methods on Anole, and the cost of visual quality is small.

More results about visual quality. We take the CLIP-Score and the human preference score (HPSv2) (Wu et al., 2023) as the metrics for evaluating the visual quality for our ablation studies (the step compression ratios are reported in Sec. 5.4). We present the results in Tab. 7, Tab. 8, Tab. 9, and Tab. 10. From Tab. 7, given any K values in the top-K sampling strategies, we can observe that the human preferences are also not much different among the original auto-regressive decoding, the original Jacobi decoding, and our SJD.

Perplexity. We also compare the perplexities between SJD and other decoding methods on Lumina-mGPT, as detailed in Tab. 6. Since the perplexities are influenced by the sampling strategies (Hu et al., 2023), we report the perplexities under various K values. Given an identical K value, the perplexities between our method and other decoding methods are close. Furthermore, we note that $K = 2000$ results in a perplexity higher than that of large language models (Gu & Dao, 2023; Yang et al., 2024b) on language processing tasks. Despite this high value, the text-to-image auto-regressive model can still generate high-quality images. This indicates that image generation can tolerate a wide range of image tokens.

Statistics of model outputs. We compute the statistics of the logarithm of the token probability for both auto-regressive decoding and our method. The average and standard deviation of all image tokens are presented in Tab. 11. The results demonstrate that the image tokens accepted by our method exhibit similar statistics to those accepted by the original auto-regressive decoding. Consequently, our method generally does not mistakenly accept tokens with lower probabilities.

E VISUALIZATION OF ACCELERATION IN 2D SPACE

We visualize the impact of multi-token prediction in a 2D space. As illustrated in Fig. 13, the color of each long strip area represents the length of accepted tokens from that area, with darker colors indicating longer sequences of accepted tokens, *i.e.*, higher acceleration. We observe that high acceleration tends to occur in the background, particularly on the left and right sides of images. Additionally, while some high acceleration is observed on foreground objects, it is relatively sparse in 2D space.

F Limitation and future work

Since our speculative Jacobi decoding is training-free, the accelerated model itself is still not specialized for multi-token prediction. Therefore, the acceleration ratio has the potential to be further improved. In the future, we believe that fine-tuning the auto-regressive models for fast image generation is a promising direction. Also, acceleration is important for long-sequence generation, like video generation. Since videos contain more redundancy than images, the initialization of candidate tokens should be carefully designed if applying our speculative Jacobi decoding to video generation.

G Broader Impacts

Image generation offers extensive utility in helping users, designers, and artists produce fantastic content. Nonetheless, these models could be exploited to create deceptive content. Thus, it is crucial for the users including researchers and developers to acknowledge the potential negative social impact of image generation models.

H Analysis on the Effectiveness of our method

This section analyzes the acceleration mechanism of our speculative Jacobi decoding in image generation. *We empirically find that this acceleration stems from the resampling of unaccepted tokens.* Specifically, some tokens are *continuously resampled* (*i.e.*, ***their positions within the entire sequence are reused for multiple forward passes***) according to Equ. (3) over iterations until they are accepted. For clarity and simplicity, we refer to this process of a token being continuously resampled by Equ. (3) (except the possible rejection resampling) as *refinement*. Consequently, Equ. (3) is the main operation of every *refinement step*. In the following paragraphs, we explore the influences of this refinement.

The acceleration originates from the refinement of unaccepted tokens. In our verification phase, there are *three treatments* for the tokens: *acceptance*, *rejection*, and *refinement*, corresponding to Equ. (1), Equ. (2), and Equ. (3), respectively. We empirically find that ***only the first two treatments*** are ***insufficient*** to support acceleration. We conduct the following experiment to demonstrate that our method makes it hard to achieve acceleration without refinement: *when we deactivate the refinement (i.e., using the newly initialized tokens to replace the unaccepted tokens as the draft tokens in the next iterations), we observe that the model requires over two thousand forward passes to generate images rather than one thousand forward passes. Although our token initializations with spatial prior (e.g., horizontal repeat) are slightly better than the random token initialization in replacing the unaccepted tokens, its performance is still much worse than directly refining the unaccepted tokens.* The examples of the generated images under such setting are shown in Fig. 14. This phenomenon illustrates that the acceleration of our method originates from refining unaccepted tokens.

I Qualitative Analysis of image randomness on our method

Like Fig. 2, we also examine the image randomness with both the auto-regressive decoding and our speculative Jacobi decoding. As shown in Fig. 17, first, we find that SJD does introduce some randomness into image generation (the random variable r in Equ. (1)), so the images generated with auto-regressive decoding cannot exactly align those generated with SJD, even when the random seed is fixed. Therefore, in Fig. 17, given a column, two images with the same K value cannot be exactly identical.

However, the diversity of the set of images is not observed to be influenced. In Fig. 17, we present the images generated based on three textual prompts. Given the same prompt and K value from top-K sampling, the model with different decoding methods generates images with many similarities. For example, when $K = 2000$, for the first prompt “an apple of a strange color”, the images in the identical columns show the apples with similar color patterns and styles. Also, for the third prompt

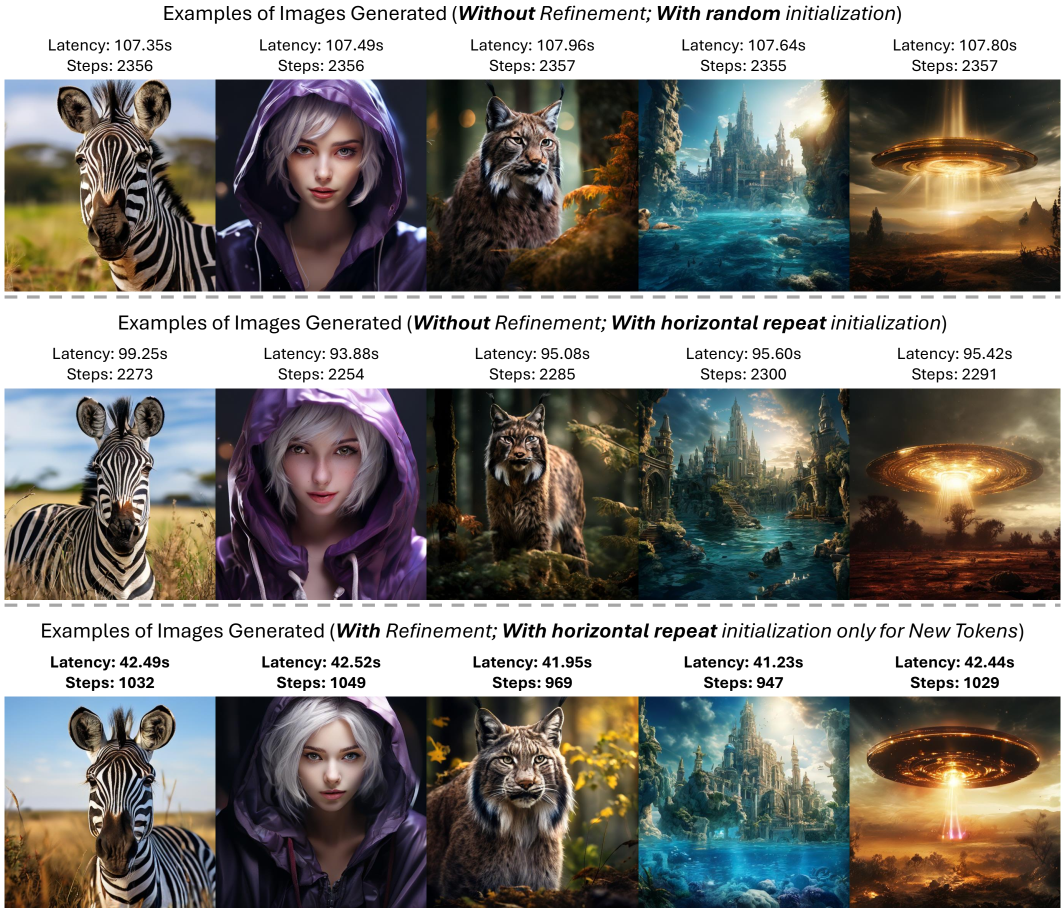

Figure 14: **Ablation studies on acceleration mechanism**: *examples of images generated by our SJD without or with refining unaccepted tokens.* When the refinement defined by Equ. (3) is ***NOT*** applied (*i.e.*, using the newly initialized tokens to replace the unaccepted tokens as the draft tokens in the next iterations), there is almost no acceleration (though one of our token initializations with spatial prior, horizontal repeat, can slightly reduce the steps in these images). This illustrates that ***refining*** unaccepted tokens is essential to the acceleration mechanism in SJD.

"pumpkin on the table", the frequency of faces carved on the pumpkins is consistent for these two decoding methods.

Moreover, the K value in top-K sampling still dominates the image randomness about texture, color, and local structure details. With larger K, the image details about textures, colors, and local structures increase. Such image randomness still largely comes from the random token sampling.

J Analysis on Failure Cases

As shown in Fig. 18, when generating images with exquisite details, although auto-regressive decoding can produce artifacts, SJD seems to generate continuous tokens that cause the artifacts, as highlighted by the red boxes in this figure. The pre-trained auto-regressive model is not sufficiently robust to handle such complex images. Consequently, it may mistakenly accept a sequence of draft tokens that contain artifacts.

Table 4: The evaluation of Anole on the validation set of MSCOCO2017. JD: Jacobi decoding. ISP: initialization with spatial prior. SJD: Speculative Jacobi decoding.

Configuration		Average Latency (↓)	Acceleration (↑) Latency	Step	FID (↓)	CLIP-Score (↑)
A	Anole (Chern et al., 2024)	48.96s	1.00×	1.00×	28.87	30.59
B	***w.*** JD (Song et al., 2021)	47.60s	1.03×	1.06×	29.34	30.64
C	***w.* SJD**	27.08s	1.81×	1.94×	29.04	30.54
D	***w.* SJD (ISP)**	**26.18s**	**1.87×**	**1.97×**	29.14	30.61

Table 5: The evaluation of Anole on the validation set of Parti-prompt. JD: Jacobi decoding. ISP: initialization with spatial prior. SJD: Speculative Jacobi decoding.

Configuration		Average Latency (↓)	Acceleration (↑) Latency	Step	CLIP-Score (↑)
A	Anole (Chern et al., 2024)	48.24s	1.00×	1.00×	30.46
B	***w.*** JD (Song et al., 2021)	44.65s	1.08×	1.14×	30.57
C	***w.* SJD**	26.77s	1.80×	2.00×	30.55
D	***w.* SJD (ISP)**	**25.12s**	**1.92×**	**2.11×**	30.48

Table 6: The comparison of perplexity on Lumina-mGPT.

Configuration		Perplexity with Top-K sampling $K = 10$	$K = 100$	$K = 2000$
A	Lumina-mGPT (Liu et al., 2024b)	7.31	43.37	204.06
B	***w.*** JD (Song et al., 2021)	7.20	43.85	197.64
C	***w.* SJD**	7.34	43.87	217.96
D	***w.* SJD (ISP)**	7.26	44.03	199.70

Table 7: CLIP-Score of various decoding methods on Lumina-mGPT with different top-K values. The image qualities for Jacobi Decoding and our method correspond to Fig. 6. The image qualities for Auto-regression are only for the comparison in this table. Note that the image quality score with greedy sampling is extremely poor, as this setting leads to meaningless images for a lot of prompts (analyzed in Fig. 2).

Decoding Methods	Sampling	CLIP-Score	HPSv2
Auto-regression	Top-1 Sampling	26.40	0.1976
Auto-regression	Top-10 Sampling	32.83	0.2950
Auto-regression	Top-100 Sampling	32.41	0.3020
Auto-regression	Top-2000 Sampling	32.00	0.2965
Jacobi Decoding	Top-1 Sampling	26.34	0.1413
Jacobi Decoding	Top-10 Sampling	32.75	0.2960
Jacobi Decoding	Top-100 Sampling	32.46	0.3089
Jacobi Decoding	Top-2000 Sampling	31.68	0.3103
Ours	Top-1 Sampling	26.16	0.1695
Ours	Top-10 Sampling	32.27	0.2942
Ours	Top-100 Sampling	32.65	0.2977
Ours	Top-2000 Sampling	31.83	0.3020

Figure 15: The qualitative comparison of different decoding methods on Lumina-mGPT.

Figure 16: The qualitative comparison of different decoding methods on Anole. Considering the high image diversity of Anole, although the random seed is fixed, it is still hard for Anole to generate similar images with different decoding methods.

Table 8: CLIP-Scores on Lumina-mGPT with various resolutions. The image qualities of our method under different settings correspond to Fig. 7. The image qualities for Auto-regression are only for the comparison in this table.

Decoding Methods	Resolutions	CLIP-Score	HPSv2
Auto-regression	512	29.49	0.2503
Auto-regression	768	32.00	0.2965
Auto-regression	1024	31.41	0.2961
Ours	512	29.69	0.2558
Ours	768	31.83	0.3020
Ours	1024	31.11	0.2935

Table 9: CLIP-Score of our method on Lumina-mGPT with various Jacobi window sizes. The image qualities correspond to Fig. 8.

Jacobi Window Size	CLIP-Score	HPSv2
1	32.00	0.2965
4	31.91	0.3046
16	31.83	0.3020
32	31.55	0.3045

Figure 17: Comparing our method to the original auto-regressive decoding on the image randomness. First, considering the random variable in SJD, given a column, two images with the same K value cannot be exactly identical. Second, changing the decoding method from auto-regression to SJD has little influence on the image diversity for each prompt (*e.g.*, given $K = 2000$ for each decoding method, the color patterns and styles of the generated apples are similar, and the frequency of the carved faces on pumpkins is also similar). Third, the top-K sampling still dominates the image randomness about texture, color, and local structure details. The images in each column share a single random seed.

Figure 18: **Failure Cases**. In complex image scenarios, our method generates some continuous tokens that result in artifacts, as highlighted by the red boxes. The pre-trained model inaccurately accepts a large sequence of the tokens that cause the artifacts.

Table 10: CLIP-Score of our method on Lumina-mGPT with various token initialization when generating images with simple patterns. The image qualities correspond to Fig. 9.

Token Initialization	CLIP-Score	HPSv2
Horizontal Sample	31.52	0.2567
Vertical Sample	30.91	0.2622
Horizontal Repeat	31.17	0.2616
Vertical Repeat	31.15	0.2651
Random	31.37	0.2681

Table 11: The comparison of token statistics on Lumina-mGPT.

Decoding Methods	Logarithm of Token Probability	
	Average	Standard Deviation
Auto-regression	-4.8950	2.3457
Ours	-4.9007	2.3275