This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Training-free Mixed-Resolution Latent Upsampling for Spatially Accelerated Diffusion Transformers

Wongi Jeong*[1] Kyungryeol Lee*[1] Hoigi Seo[1] Se Young Chun[1,2†]
[1]Dept. of Electrical and Computer Engineering, [2]INMC & IPAI
Seoul National University, Republic of Korea
{wg7139, kr.lee, seohoiki3215, sychun}@snu.ac.kr

Abstract

Diffusion transformers (DiTs) offer excellent scalability for high-fidelity generation, but their computational overhead poses a great challenge for practical deployment. Existing acceleration methods primarily exploit the temporal dimension, whereas spatial acceleration remains underexplored. In this work, we investigate spatial acceleration for DiTs via latent upsampling. We found that naïve latent upsampling for spatial acceleration introduces artifacts, primarily due to aliasing in high-frequency edge regions and mismatching from noise-timestep discrepancies. Then, based on these findings and analyses, we propose a training-free spatial acceleration framework, dubbed Region-Adaptive Latent Upsampling (RALU), to mitigate those artifacts while achieving spatial acceleration of DiTs by our mixed-resolution latent upsampling. RALU achieves artifact-free, efficient acceleration with early upsampling only on artifact-prone edge regions and noise-timestep matching for different latent resolutions, leading to up to 7.0× speedup on FLUX-1.dev and 3.0× on Stable Diffusion 3 with negligible quality degradation. Furthermore, our RALU is complementarily applicable to existing temporal acceleration methods and timestep-distilled models, leading to up to 15.9× speedup. Project page: https://ignoww.github.io/RALU_project/

1. Introduction

Diffusion models [16, 47, 48] have emerged as a dominant framework for generative modeling across diverse modalities such as image [5, 37, 40, 42, 57], video [2, 6, 18, 19, 31], and audio [15, 28, 43]. Convolutional U-Net architectures [41] have served as the backbone of relatively small diffusion models. Recently, diffusion transformers (DiTs) [1, 7, 31, 40] have been introduced, leveraging the scalability of transformers to achieve state-of-the-art results in visual synthesis.

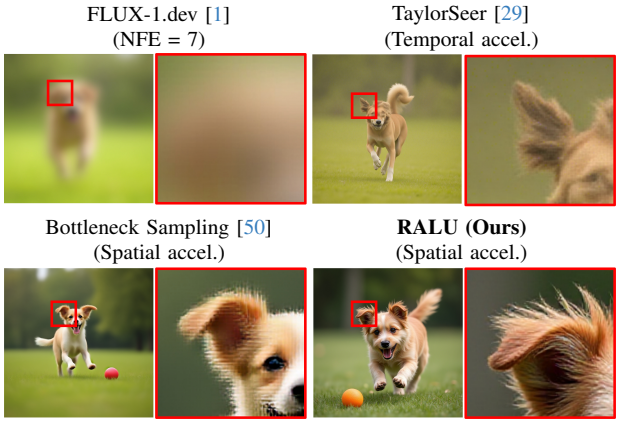


Figure 1. Generated 1024×1024 images using acceleration methods on FLUX-1.dev [1] for 7× speedups. While temporal acceleration methods struggle with aggressive speedups and Bottleneck Sampling [50] introduces artifacts, our RALU successfully accelerates while avoiding artifacts and maintaining high image quality.

Despite the superior performance of DiTs, they suffer from high inference latency. This bottleneck is inherent in their transformer-based architecture, whose complexity scales quadratically with the number of input tokens [33, 38]. As models scale to higher resolutions and process more tokens, computational cost and latency pose a great challenge for their deployment in real-world applications such as real-time interactive editing or on-device generation [25].

Distillation [12, 56] and post-training [34, 54, 55] have been explored to tackle this challenge, but they require additional training and substantial computational cost. Training-free temporal acceleration has emerged as an alternative by reducing computation across timesteps via feature caching [9, 32, 55, 61]. In contrast, spatial acceleration, which involves transitioning between resolutions for a quadratic reduction in computation, remains underexplored, particularly for training-free latent upsampling in DiT.

Prior works on spatial upsampling, often explored within the tasks such as super-resolution [58, 60] and higher-

* Authors contributed equally. † Corresponding author.

resolution synthesis [13, 23, 39], have focused on image space rather than latent space. There are only a few existing works for upsampling on the latent space, but mostly require extra training, limiting their applicability to existing generative models that were heavily trained. Pyramidal Flow [22] requires the unified training for its generative framework. Similarly, Latent-SR [21] achieves latent space super-resolution by proposing a learned noise addition module. Bottleneck Sampling [50] is a training-free framework for spatial acceleration, but suffers from substantial artifacts, as illustrated in Fig. 1 (bottom left).

Here, we investigate latent upsampling for DiTs. While late latent upsampling is beneficial for spatial acceleration, it introduces two types of artifacts: *aliasing artifacts* in high-frequency regions and *mismatching artifacts* from a noise-timestep discrepancy. We found that early upsampling mitigates aliasing artifacts, but requires more computation in high latent resolution. Then, we propose Region-Adaptive Latent Upsampling (RALU), a *training-free*, *spatial* acceleration approach for DiTs that efficiently and effectively exploits this trade-off. For RALU, we propose mixed-resolution latent upsampling that performs early upsampling only on artifact-prone regions and Noise and Timestep Matching (NT-Matching) that matches the noise distributions before and after upsampling. As illustrated in Fig. 1, RALU achieves computational gains by processing latents at low resolution mostly, while preserving high generation quality with significantly fewer artifacts by selectively performing early latent upsampling with NT-Matching. Moreover, RALU is complementarily applicable to existing temporal acceleration methods and timestep-distilled models, enabling additional speedups with negligible degradation in quality. The contributions of this work are summarized as follows:

- We investigate latent upsampling for DiTs as a means of spatial acceleration and explore the trade-off of upsampling between speedup and artifacts.
- We propose a training-free spatial acceleration method, RALU, that effectively exploits the trade-off by our mixed-resolution latent upsampling with early upsampling on artifact-prone edge regions and our NT-Matching for different latent resolutions, leading to up to 7.0× speedup with negligible quality degradation.
- We demonstrate that our RALU can be synergistically applicable to existing temporal acceleration methods and timestep-distilled models, leading to up to 15.9× speedup.

2. Related Works

2.1. Flow matching

Flow matching [26] is a recent generative model that learns a deterministic transport map from a simple prior (*e.g.*, standard Gaussian noise) to a complex data distribution by integrating an ordinary differential equation (ODE). In particular, rectified flow [30] defines a linear interpolation path between the noise $\mathbf{x}_0$ and the data sample $\mathbf{x}_1$ by

$$\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1, \quad t \in [0,1], \tag{1}$$

with a constant velocity field $\mathbf{v}_t = \frac{d\mathbf{x}_t}{dt} = \mathbf{x}_1 - \mathbf{x}_0$. The learning objective is to train a neural network that takes the state $\mathbf{x}_t$ and time t as input to predict this ground-truth conditional velocity field by minimizing the difference between the predicted velocity and the true velocity.

2.2. Diffusion Transformer acceleration

DiTs are computationally expensive, especially when generating high-resolution images, as the cost of self-attention grows quadratically with the number of spatial tokens. To mitigate these bottlenecks, recent research has proposed various inference-time acceleration techniques, which can be broadly categorized into model compression, temporal acceleration, and spatial acceleration methods.

Model compression. Model compression techniques aim to reduce model size or computational complexity without retraining from scratch. Common approaches include quantization [8, 24, 46], distillation [12, 25, 56], and block pruning [11, 34, 45, 52]. In particular, block pruning methods skip transformer blocks that contribute less during inference, improving efficiency. However, they often require a healing strategy, as quality degradation occurs without fine-tuning.

Temporal acceleration. Temporal acceleration aims to reduce computation by skipping layers or reusing cached features across timesteps. Caching-based approaches have been extended to DiTs by storing internal activations such as block outputs [9, 27, 35, 61] or attention maps [55]. Other strategies move beyond reuse, instead forecasting future features by modeling their temporal trajectory [29]. Some works explore token-level pruning or selective execution [32], or introduce learnable token routers that dynamically decide which tokens to recompute and which to reuse [33, 54].

Spatial acceleration. Spatial acceleration refers to reducing computation by dynamically transitioning between latent resolutions. In DiTs, this is equivalent to adjusting the number of input patches (tokens). Recent studies [17, 22, 42, 49] have proposed cascaded diffusion frameworks that start from low-resolution and achieve high-resolution through upsampling during the denoising process. However, these frameworks require resource-intensive training.

To the best of our knowledge, Bottleneck Sampling [50] is the only existing training-free spatially accelerated image generation method. It achieves acceleration through two latent resolution changes via Lanczos resampling, but does not seem to correct for the skewed trajectory after upsampling, resulting in substantial artifacts. This highlights the need for a spatial acceleration method that can effectively mitigate such artifacts due to latent upsampling.

3. Challenges in Spatial Acceleration for DiTs

Multi-resolution approaches have been popular for many vision tasks, leading to spatial acceleration. However, while latent upsampling for DiTs has a great potential for spatial acceleration, cutting computational cost, it has not been well-investigated. In this section, we argue that naïve latent upsampling for acceleration introduces two types of artifacts in DiTs and describe our key findings on how each type of artifact can be effectively mitigated.

3.1. Aliasing artifacts due to late upsampling

We observe that various latent upsampling methods (*e.g.*, nearest-neighbor, bilinear, bicubic, Lanczos) for DiT-based generations introduce aliasing artifacts predominantly in high-frequency regions such as semantic edges when upsampled at late timesteps. Since the low-resolution latent space may not have enough representation power for high-frequency details such as sharp boundaries, late upsampling often leads to visual distortions. Fig. 2a visualizes this phenomenon: while fidelity is maintained within smooth interior regions, aliasing artifacts appear prominently near boundaries when upsampled at the mid-timestep among 18 steps.

We found that these artifacts can be avoided by upsampling in earlier diffusion timesteps when the generated semantic structure is still coarse. To empirically validate this, we conducted the experiment with two-stage processes: denoising diffusion in low-resolution latent space and restoring full-resolution in various timesteps t_{up}. Fig. 2b illustrates that upsampling in the early timestep ($t_{up} \leq 0.3$) does not introduce aliasing artifacts, while late upsampling ($t_{up} \geq 0.5$) leads to a significantly higher aliasing artifact ratio (i.e., the proportion of images exhibiting visible aliasing) and their undesirable edge energy (i.e., the average edge intensity quantified by a Canny edge detector).

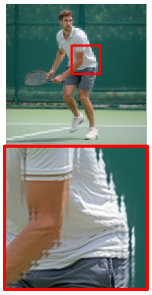
Remark 1. *Aliasing artifacts in DiTs occur mostly in edge regions when upsampling latents at later timesteps, but do not occur when upsampling latents at earlier timesteps.*

However, naïvely upsampling all latents in the early stages would sacrifice the computational benefits at a lower resolution. Due to this trade-off, mixed-resolution latent upsampling is desirable, which performs early upsampling only on the edge-region latents and late upsampling on the rest of the latents for mitigating aliasing artifacts and spatially accelerating image generation in DiTs, respectively.

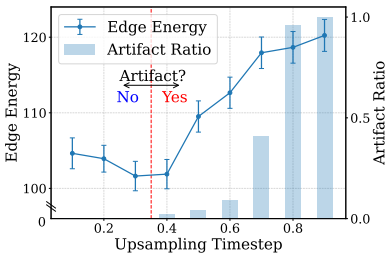
3.2. Distribution mismatching artifacts

Latent upsampling for flow-based generations introduces distribution mismatching artifacts without properly injecting noise and matching timestep distribution (i.e., how frequently timesteps are sampled during inference [59]).

Correlated noise after upsampling. After upsampling, latent distribution deviates from the original flow trajectory.



(a) Aliasing.

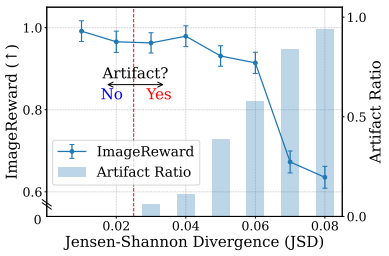


(b) Analysis on upsampling timestep and artifact.

Figure 2. (a) An example of aliasing artifacts generated using FLUX-1.dev (prompt: *"A man on the tennis court is about to use his racket"*) with 9 low-resolution steps, 2× upsampling, and 9 full-resolution steps. (b) Edge energy and aliasing artifact ratio over image vs. upsampling timestep, averaged over 100 images.



(a) Mismatching.



(b) Analysis on JSD and performance.

Figure 3. (a) An example of mismatching artifacts generated using FLUX-1.dev (prompt: *"A group of people standing on top of a snow covered ski slope"*) with early upsampling ($t_{up} = 0.3$) and noise injection. (b) ImageReward [53] score and mismatching artifact ratio vs. JSD, averaged over 100 images.

Starting from an initial noise $\mathbf{x}_0 \sim \mathcal{N}(0, \mathbf{I})$, flow matching aims to learn a mapping to the target $\mathbf{x}_1$ by following Eq. (1) through a denoising process. Then, the conditional distribution of $\hat{\mathbf{x}}_t$ at timestep t is:

$$\hat{\mathbf{x}}_t | \mathbf{x}_1 \sim \mathcal{N}\left(t\mathbf{x}_1, (1-t)^2\mathbf{I}\right). \tag{2}$$

After upsampling, the distribution becomes:

$$\mathrm{Up}(\hat{\mathbf{x}}_t) | \mathbf{x}_1 \sim \mathcal{N}\left(t\mathrm{Up}(\mathbf{x}_1), (1-t)^2\mathbf{\Sigma}\right), \tag{3}$$

where $\mathrm{Up}(\cdot)$ is a upsampling function and $\mathbf{\Sigma}$ is the covariance matrix after upsampling. For any interpolation-based upsampler, $\mathrm{Up}(\hat{\mathbf{x}}_t)$ no longer matches the original trajectory's distribution due to non-isotropic $\mathbf{\Sigma}$. To enforce $\mathrm{Up}(\hat{\mathbf{x}}_t)$ stay in the original trajectory, correlated noise should be injected, making the upsampled covariance matrix isotropic.

Timestep distribution matching. Noise injection alters noise level, which in turn skews timestep distribution. Models can be trained for this modification [22, 49], but additional massive computation is required for each model. The mismatch between the pretrained model's original timestep

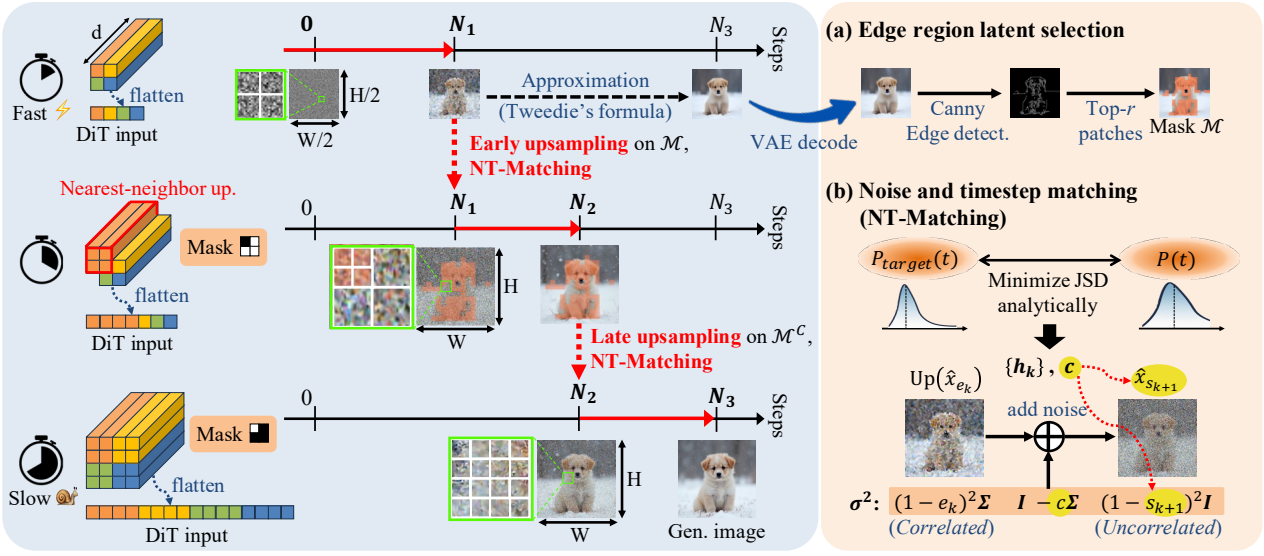


Figure 4. Overview of the proposed RALU framework. RALU consists of three different resolution processes: (1) low-resolution sampling for early denoising, (2) mixed-resolution sampling by upsampling edge region latents, and (3) full-resolution refinement by upsampling all remaining latents. (a) We select the top r fraction of patches with the strongest edge signals from the decoded image and upsample them early. (b) We add correlated noise to the upsampled latents and design a corresponding timestep schedule. (See Sec. 4.2 for more details).

distribution and the modified distribution seems to cause oversampling certain intervals and a frequency imbalance, resulting in visual artifacts. Fig. 3a visualizes this: early upsampling avoids aliasing, but the lack of distribution matching creates global mismatching artifacts.

We found that these artifacts can be mitigated by aligning the upsampling-induced timestep distribution with the original model's. To verify this, we constructed a two-stage framework with fixed upsampling timestep to $t_{up} = 0.3$ (to prevent aliasing) and applied our corrective noise and timestep distribution (detailed in Sec. 4.2) while varying its parameters to induce a range of Jensen-Shannon divergence (JSD) values. Fig. 3b shows that as JSD decreases, mismatching artifacts diminish (disappearing at JSD < 0.03) and the ImageReward [53] score improves ($p < 0.001$).

Remark 2. *Mismatching artifacts in flow-based DiTs occur when naïvely adding noise after upsampling, but can be mitigated by aligning the new sampling distribution with the original timestep distribution of the pretrained model.*

4. Region-Adaptive Latent Upsampling

To break the fundamental trade-off between acceleration and artifacts identified in Sec. 3, we propose Region-Adaptive Latent Upsampling (RALU) approach.

4.1. Mixed-resolution latent upsampling

As illustrated in Fig. 4, our approach comprises mixed-resolution latent upsampling. The denoising process begins at *low-resolution* to accelerate. We reduce the latent resolution by a factor of 2 along width and height, thereby reducing the number of latent tokens to only 1/4.

Next, we perform an intermediate *edge region upsampling* step to prevent aliasing artifacts, which typically occur in high-frequency regions as discussed in Sec. 3.1. We identify artifact-prone regions by 1) estimating the clean latent $\mathbf{x}_0$ via Tweedie's formula from the low-resolution stage, 2) decoding it with the VAE, and 3) performing Canny edge detection. We then select and upsample the top-r ratio of latent patches corresponding to these edges. The variable r controls the trade-off between acceleration and artifact and we chose it in the range of 20-30% of the whole latents, making the mixed-resolution latents.

Finally, the remaining low-resolution latent tokens are upsampled to full-resolution to generate a complete high-resolution image. This final step ensures the consistency between the early and late upsampled regions in the final output by matching position embeddings. As demonstrated in Fig. 5, late upsampling (left) results in aliasing, while early upsampling (right) effectively avoids this artifact.

4.2. Noise and timestep matching

As discussed in Sec. 3.2, we design appropriate correlated noise injection and timestep distribution matching after latent upsampling to mitigate mismatching artifacts in DiTs.

Noise injection. Starting from Eq. (3), at the last timestep of the k-th stage e_k, the conditional distribution of the latent after 2× nearest-neighbor upsampling is:

$$\mathrm{Up}(\hat{\mathbf{x}}_{e_k})|\mathbf{x}_1 \sim \mathcal{N}\left(e_k \mathrm{Up}(\mathbf{x}_1), (1-e_k)^2\boldsymbol{\Sigma}\right), \quad (4)$$

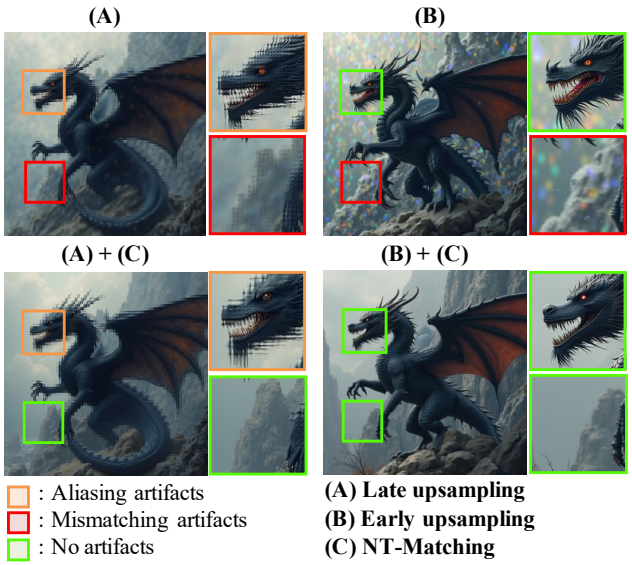


Figure 5. Resolving artifacts from naïve latent upsampling. Aliasing artifacts are avoided by (B) early upsampling, while Mismatching artifacts are mitigated by (C) noise and timestep matching.

where $\mathbf{\Sigma}$ has a block-diagonal structure with the 4×4 blocks filled with ones. As discussed in Sec. 3.2, since $\mathbf{\Sigma}$ is non-isotropic, proper correlated noise $\mathbf{z} \sim \mathcal{N}(0, \mathbf{\Sigma}')$ should be added to put it back onto the original trajectory's distribution:

$$a\mathrm{Up}(\hat{\mathbf{x}}_{e_k}) + b\mathbf{z} = \hat{\mathbf{x}}_{s_{k+1}},\ \mathbf{z} \sim \mathcal{N}(0, \mathbf{\Sigma}') \tag{5}$$

$$\hat{\mathbf{x}}_{s_{k+1}} | \mathbf{x}_1 \sim \mathcal{N}\left(s_{k+1}\mathrm{Up}(\mathbf{x}_1), (1-s_{k+1})^2\mathbf{I}\right). \tag{6}$$

where s_{k+1} is the starting timestep of stage $k+1$ and a, b are scalar values, and $\mathbf{\Sigma}'$ is the covariance matrix of $\mathbf{z}$. The parameters s_{k+1}, a, and b are derived by matching the distribution in Eq. (6) under the constraint $\mathbf{\Sigma}' = \mathbf{I} - c\mathbf{\Sigma}$. By defining the composite term δ_k as $\delta_k \equiv (1-e_k)/\sqrt{c}$, the rescheduling parameters can be expressed as:

$$s_{k+1} = \frac{e_k}{\delta_k + e_k},\ a = \frac{1}{\delta_k + e_k},\ \text{and}\ b = \frac{\delta_k}{\delta_k + e_k}. \tag{7}$$

Detailed derivations are provided in the supplementary material. This noise injection method modifies the approach from the training-based method of [22]. While [22] can learn the noise and timestep parameters during its unified training, our *training-free* setting necessitates a different solution, as these parameters cannot be arbitrarily determined for a pre-trained model. Therefore, we introduce a timestep distribution matching that is compatible with pretrained models.

Timestep distribution matching. After noise injection at the timestep e_k, the diffusion process resumes at s_{k+1}, meaning reusing the original schedule can oversample the interval $[s_{k+1}, e_k]$. As mentioned in Sec. 3.2, this misalignment causes artifacts. Noise and Timestep Matching (NT-Matching) resolves these artifacts by matching the timestep distribution with the original scheduler, such as the non-uniform sampling employed by flow matching [1, 10]. Their probability density function (PDF) and truncated PDF are:

$$f_h(t) = \frac{h}{(1+(h-1)t)^2}\ ,0 \le t \le 1, \tag{8}$$

$$f_{h,s,e}(t) = \frac{f_h(t)}{F_h(e) - F_h(s)}\ ,s \le t \le e, \tag{9}$$

where h is a shifting parameter and $F_h(t)$ is a cumulative distribution function (CDF) of $f_h(t)$. Our method injects noise at the end of each stage, which requires additional denoising over the overlapping interval $[s_{k+1}, e_k]$. Therefore, timestep sampling within intervals $[0,1], [s_2, e_1], \ldots, [s_K, e_{K-1}]$ should follow $f_h(t)$. The overall sampling distribution can be written as a weighted sum of truncated PDFs:

$$P_{\text{target}}(t) = \frac{f_{h_{\text{ori}},0,1}(t) + \sum_{k=1}^{K-1}(e_k - s_{k+1}) f_{h_{\text{ori}},s_{k+1},e_k}(t)}{1 + \sum_{k=1}^{K-1}(e_k - s_{k+1})}$$

where h_{ori} is the shifting parameter of the base model and $K = 3$ is the number of stages. However, the actual sampling intervals are $[0, e_1], [s_2, e_2], \ldots, [s_K, 1]$. Therefore, the target distribution should also be adjusted accordingly. We use a stage-wise shifting parameter h_k to control the PDF in each interval. Assuming we sample $N_k - N_{k-1}$ timesteps in the k-th interval according to $f_{h_k,s_k,e_k}(t)$, the modified timestep distribution $P(t)$ is:

$$P(t) = \frac{\sum_{k=1}^{K}(N_k - N_{k-1}) f_{h_k,s_k,e_k}(t)}{N_K}. \tag{10}$$

We minimize the Jensen-Shannon divergence between the target distribution $P_{\text{target}}(t)$ and the actual distribution $P(t)$ via numerical search. We analytically determine c and $\{h_k\}$, and these values then dictate the noise degree, correlation, and full timestep schedule. Fig. 5 shows the effectiveness of NT-Matching: while the image with only early upsampling (top right) suffers from mismatching artifacts, adding NT-Matching (bottom right) successfully mitigates them.

5. Experiments

5.1. Quantitative results

Base models and metrics. We adopt FLUX.1-dev [1] (FLUX) and Stable Diffusion 3 Medium [10] (SD3) as base models. We assessed image quality and text alignment with ImageReward [53], CLIP-IQA [51], NIQE [36], T2I-CompBench [20] and GenEval [14]. We measured acceleration using latency and FLOPs on a single A100 GPU. More details are provided in the supplementary materials.

Table 1. Quantitative comparisons of RALU with the baselines on (Top) FLUX.1-dev [1] (FLUX) and (Bottom) Stable Diffusion 3 [10] (SD3). Performance is evaluated with CLIP-IQA [51] and NIQE [36] for image quality, T2I-CompBench [20] and GenEval [14] for image-text alignment, and ImageReward [53] for both. The number in parentheses next to FLUX indicates the total number of inference steps. ↑ / ↓ denotes that a higher / lower metric is favorable. Speedup (Speed.) is calculated relative to the base model FLOPs (floating-point operations). **T** and **S** denote the temporal and spatial acceleration, respectively. Additional computational metrics are reported in the supplementary material.

Method	Accel.	Latency (s) ↓	TFLOPs ↓	Speed. ↑	Overall	Image quality		Text alignment	
					ImageReward ↑	CLIP-IQA ↑	NIQE ↓	T2I-Comp. ↑	GenEval ↑
FLUX (50)	-	25.1	2991.0	1.00×	1.095	0.707	6.75	0.634	0.698
FLUX (10)	T	5.18	610.02	4.90×	0.981	0.679	6.93	0.618	0.647
Δ-DiT [9]	T	7.42	772.10	3.87×	0.102	0.487	9.60	0.306	0.397
ToCa [61]	T	15.5	601.12	4.98×	-1.827	0.253	10.6	0.259	0.137
TeaCache [27]	T	5.23	610.59	4.90×	0.944	0.665	7.92	0.620	0.647
TaylorSeer [29]	T	9.34	556.72	5.37×	0.972	0.684	6.77	0.594	0.619
Bottleneck [50]	S	5.37	571.23	5.24×	0.889	0.661	9.16	0.620	0.687
RALU (Ours)	S	**5.04**	**540.47**	**5.53**×	1.022	0.700	6.43	0.626	0.652
FLUX (7)	T	3.79	431.45	6.93×	0.920	0.660	8.25	0.594	0.583
TeaCache [27]	T	4.21	431.83	6.93×	0.733	0.623	13.7	0.599	0.594
TaylorSeer [29]	T	7.00	431.74	6.83×	0.660	0.646	9.43	0.514	0.446
Bottleneck [50]	S	3.78	431.52	6.93×	0.792	0.631	8.71	0.605	0.672
RALU (Ours)	S	**3.75**	**426.01**	**7.02**×	0.999	0.681	6.87	0.633	0.682

Method	Accel.	Latency (s) ↓	TFLOPs ↓	Speed. ↑	Overall	Image quality		Text alignment	
					ImageReward ↑	CLIP-IQA ↑	NIQE ↓	T2I-Comp. ↑	GenEval ↑
SD3 (28)	-	4.04	351.66	1.00×	0.971	0.692	6.09	0.667	0.703
SD3 (14)	T	2.14	183.30	1.92×	0.888	0.667	6.06	0.651	0.628
Δ-DiT [9]	T	2.41	183.46	1.92×	0.875	0.667	5.87	0.660	0.640
ToCa [61]	T	2.62	181.51	1.94×	0.885	0.663	5.72	0.655	0.653
RAS [32]	T	2.07	185.14	1.90×	0.706	0.574	4.85	0.649	0.583
TaylorSeer [29]	T	2.40	183.48	1.92×	0.913	0.664	5.53	0.654	0.666
Bottleneck [50]	S	2.14	186.21	1.89×	0.890	0.636	5.70	0.645	0.644
RALU (Ours)	S	**2.04**	**181.09**	**1.94**×	0.971	0.684	5.17	0.671	0.647
SD3 (9)	T	1.46	123.17	2.86×	0.565	0.619	6.10	0.627	0.561
RAS [32]	T	1.40	118.99	2.96×	0.423	0.529	5.70	0.613	0.470
TaylorSeer [29]	T	1.87	123.49	2.85×	0.059	0.521	7.44	0.561	0.572
Bottleneck [50]	S	1.44	127.35	2.76×	0.628	0.566	5.58	0.629	0.528
RALU (Ours)	S	**1.38**	**116.61**	**3.02**×	0.798	0.645	5.44	0.661	0.599

Table 2. (Top) Quantitative results of integrating temporal acceleration methods into RALU under a 5× speedup setting on FLUX. (Bottom) Quantitative results of adapting RALU on timestep-distilled models (FLUX.1-schnell, SD3.5L-Turbo). Speedups are measured relative to FLUX.1-dev and Stable Diffusion 3.5 Large, respectively. **D** denotes the timestep-distilled model. The W value of TaylorSeer denotes the number of warm-up steps.

Method	Accel.	TFLOPs ↓	Speed. ↑	Overall	Image quality		Text alignment	
				ImageReward ↑	CLIP-IQA ↑	NIQE ↓	T2I-Comp. ↑	GenEval ↑
RALU (5×)	S	540.47	5.53×	1.022	0.700	6.43	0.626	0.652
+ Δ-DiT	S + T	422.56	7.08×	0.827	0.619	7.65	0.560	0.555
+ ToCa	S + T	409.74	7.30×	0.914	0.522	6.94	0.620	0.652
+ TaylorSeer ($W = 3$)	S + T	410.70	7.28×	0.959	0.708	6.43	0.631	0.680
+ TaylorSeer ($W = 2$)	S + T	331.38	9.03×	0.926	0.691	6.00	0.590	0.586
FLUX.1-schnell (4)	D	252.88	11.83×	1.055	0.686	7.46	0.640	0.688
+ RALU	D + S	187.92	15.91×	0.992	0.650	6.88	0.621	0.636
SD3.5L-Turbo (4)	D	107.93	6.31×	0.840	0.670	7.20	0.596	0.744
+ RALU	D + S	82.23	8.28×	0.789	0.628	6.77	0.607	0.696

T2I generation performance comparison. We compare RALU with existing temporal acceleration methods such as Δ-DiT [9], ToCa [61], TeaCache [27], RAS [32], TaylorSeer [29], and the spatial acceleration method Bottleneck Sampling [50]. Tab. 1 presents the results on FLUX (top) and SD3 (bottom). Temporal acceleration methods strug-

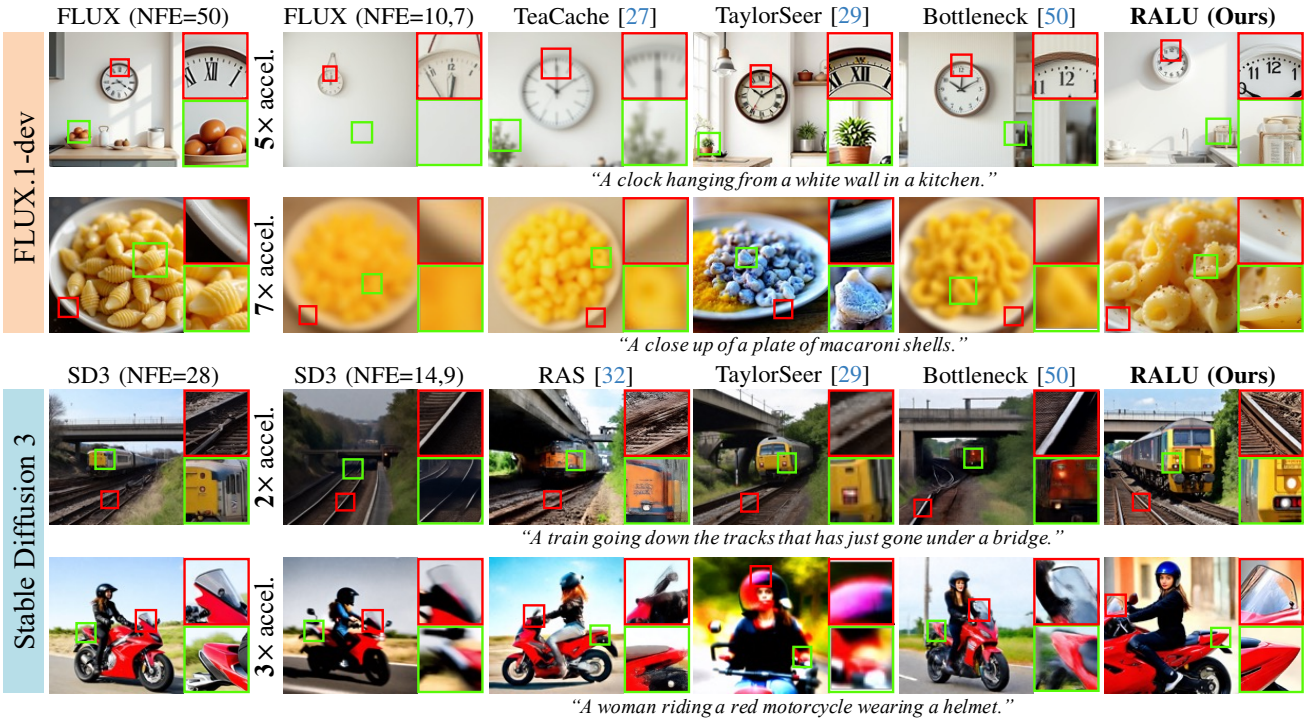


Figure 6. Qualitative comparison of images generated by baseline methods and RALU on FLUX and SD3 under various speedups. For FLUX, we compare at 5× and 7× speedups; for SD3, at 2× and 3×. NFE (number of function evaluations) refers to the number of inference steps. Zoomed-in regions on the right highlight that RALU preserves fine-grained details and avoids artifacts more effectively than the other baselines, even under high speedups. More results are provided in the supplementary material. Best viewed in zoom.

gle to deliver strong performance in both image quality and text alignment. The spatial acceleration baseline, Bottleneck Sampling, also suffers from degradation in image quality. In contrast, RALU achieves higher image fidelity and text alignment compared to other baselines.

5.2. Qualitative results

As shown in Fig. 6, base models with reduced inference steps and temporal acceleration methods tend to produce images with noticeable blur or artifacts. Bottleneck Sampling also introduces aliasing or mismatching artifacts. In contrast, RALU outperforms both temporal and spatial acceleration baselines, maintaining superior visual fidelity and semantic alignment with minimal artifacts.

5.3. Integrating RALU with other accelerations

As mentioned in Sec. 1, RALU is complementary to temporal acceleration and can be combined. Tab. 2 (Top) presents the quantitative results of incorporating temporal acceleration methods (*e.g.*, caching, forecasting) into the RALU framework. This integration yields additional improvements in speed up to 9.03× without any training.

Furthermore, RALU is also compatible with timestep-distilled models. Unlike temporal acceleration, which is applied at inference without retraining, timestep distillation represents a training-based acceleration strategy that fine-tunes the model to operate efficiently in fewer steps. Despite this difference, RALU can be seamlessly integrated with such distilled models to provide further inference-time speedups. As demonstrated in Tab. 2 (Bottom), this combination achieves up to a 15.91× total speedup over the base model with only minimal degradation in generation quality. We also provide qualitative comparisons for these integrations in the supplementary material.

Remark 3. *RALU can be integrated with temporal acceleration method or timestep-distilled model to achieve further training-free speedup while incurring only minimal quality degradation.*

5.4. Artifact comparison

Fig. 7 shows the artifact ratio according to computational cost (TFLOPs). Bottleneck Sampling [50] suffers from a high artifact ratio, and a naïve upsampling approach also results in a considerable artifact ratio. In contrast, our proposed RALU achieves a substantially lower artifact ratio than both baselines at similar TFLOPs for various speedup settings, demonstrating its superior artifact mitigation.

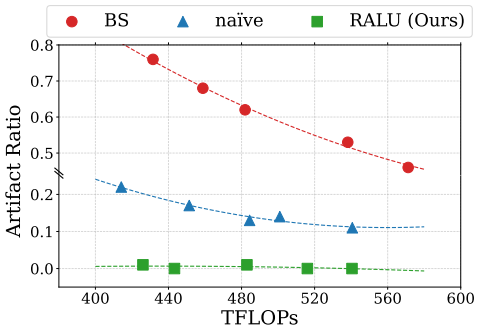
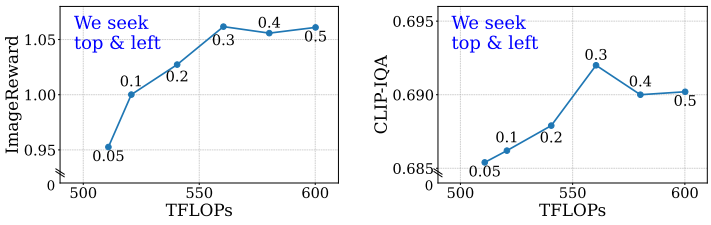


Figure 7. RALU has a drastically lower artifact ratio than Bottleneck Sampling (BS) and a naïve upsampling for a similar TFLOPs. Trendlines represent quadratic fits.



(a) ImageReward vs. TFLOPs. (b) CLIP-IQA vs. TFLOPs.

Figure 8. Trade-off between (a) ImageReward and (b) CLIP-IQA against TFLOPs for different upsampling ratios (0.05–0.5). Quality generally improves with the upsampling ratio but peaks around 0.3.

Table 3. Effect of NT-Matching. Results are on the FLUX 7× setting. The optimized parameters yield JSD= 0.026, while other parameters were adjusted to obtain higher JSDs for comparison.

JSD	ImageReward ↑	NIQE ↓	T2I-CompBench ↑
0.026 (optimized)	**0.999**	**6.51**	**0.633**
0.030	0.972	6.60	0.571
0.035	0.981	6.53	0.569
0.040	0.966	6.80	0.565

5.5. Edge region selection

Image vs. latent space. For edge region selection, we apply a Canny edge detector in the image space. An alternative is to apply the Sobel operator directly in the latent space. As detailed in the supplementary material, we found that image-space detection yields more accurate edge maps and slightly better generative performance. We adopted the Canny detector as the additional VAE decoder overhead is modest (2.48 TFLOPs for FLUX and SD3).

Fixed vs. adaptive upsampling ratio. We utilize a fixed top-r ratio to select edge regions but also experimented with a strategy that adjusts this ratio dynamically. Results in the supplementary material show that, for a similar average FLOPs, the adaptive method did not noticeably improve average image fidelity over the fixed ratio. Given that the fixed ratio ensures consistent computational costs without sacrificing quality, we adopted it for its stability.

5.6. Ablation study

Effect of early upsampling ratio. In Stage 2 of RALU, increasing the amount of top-r ratio of latent upsampling results in a more robust avoidance of aliasing artifacts. We define the upsampling ratio as the fraction of top-r ratio of latents selected for early upsampling. Fig. 8 highlights the impact of the upsampling ratio on the generated image quality. We note a trade-off where higher upsampling ratios improve image quality but lead to an increase in FLOPs. Therefore, we adopt ratios in the 0.2-0.3 range, which offers the best compromise between quality and efficiency.

Effect of NT-Matching. We analyzed the effectiveness of NT-Matching by measuring metrics under various noise level and denoising timestep scenarios. As shown in Tab. 3, it is evident that optimizing JSD yields the best image quality and text alignment, highlighting the superiority of NT-Matching.

6. Discussion

The precise mechanism by which diffusion models handle simultaneous sampling or denoising across varying resolutions, particularly how self-attention weights are computed in a multi-resolution setup, remains an area for further investigation. However, the observed robustness of these models to changes in input structure, such as the successful application of methods like token merging [3, 4, 44], where multiple tokens are compressed into a single token without major performance drops, suggests an inherent capacity for resolution-agnostic or adaptive representation learning.

7. Conclusion

In this work, we proposed *Region-Adaptive Latent Upsampling* (RALU), a training-free framework to accelerate Diffusion Transformers. RALU resolves the trade-off between acceleration and aliasing artifacts in latent upsampling through its three-stage process: low-resolution denoising for acceleration, edge-selective upsampling to avoid aliasing, and full-resolution refinement. Furthermore, to prevent mismatching artifacts, we introduced a noise-timestep matching scheme. Experiments show that RALU achieves speedup with negligible quality loss and artifacts, and complements temporal acceleration methods for further gains, offering an effective solution for efficient DiT inference.

Acknowledgements

This work was supported in part by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [NO.RS-2021-II211343, Artificial Intelligence Graduate School Program (Seoul National University)], the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (Nos. RS-2025-02263628, RS-2022-NR067592), the BK21 FOUR program of the Education and Research Program for Future ICT Pioneers, Seoul National University and Samsung Electronics MX Division.

References

[1] Black Forest Labs. FLUX. https://github.com/black-forest-labs/flux, 2024. 1, 5, 6

[2] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. *CVPR*, 2023. 1

[3] Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. *CVPR*, 2023. 8

[4] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. *ICLR*, 2023. 8

[5] Huiwen Chang, Han Zhang, Jarred Barber, Aaron Maschinot, Jose Lezama, Lu Jiang, et al. Muse: Text-to-image generation via masked generative transformers. *ICML*, 2023. 1

[6] Haoxin Chen, Yong Zhang, Xiaodong Cun, Menghan Xia, Xintao Wang, Chao Weng, and Ying Shan. Videocrafter2: Overcoming data limitations for high-quality video diffusion models. *CVPR*, 2024. 1

[7] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-σ: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. *ECCV*, 2024. 1

[8] Lei Chen, Yuan Meng, Chen Tang, Xinzhu Ma, Jingyan Jiang, Xin Wang, Zhi Wang, and Wenwu Zhu. Q-dit: Accurate post-training quantization for diffusion transformers. *CVPR*, 2025. 2

[9] Pengtao Chen, Mingzhu Shen, Peng Ye, Jianjian Cao, Chongjun Tu, Christos-Savvas Bouganis, Yiren Zhao, and Tao Chen. Δ-DiT: A training-free acceleration method tailored for diffusion transformers. *arXiv:2406.01125*, 2024. 1, 2, 6

[10] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, et al. Scaling rectified flow transformers for high-resolution image synthesis. *ICML*, 2024. 5, 6

[11] Gongfan Fang, Kunjun Li, Xinyin Ma, and Xinchao Wang. Tinyfusion: Diffusion transformers learned shallow. *CVPR*, 2025. 2

[12] Weilun Feng, Chuanguang Yang, Zhulin An, Libo Huang, Boyu Diao, Fei Wang, and Yongjun Xu. Relational diffusion distillation for efficient image generation. *ACM MM*, 2024. 1, 2

[13] Peng Gao, Le Zhuo, Dongyang Liu, Ruoyi Du, Xu Luo, Longtian Qiu, et al. Lumina-t2x: Scalable flow-based large diffusion transformer for flexible resolution generation. *ICLR*, 2025. 2

[14] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *NeurIPS*, 2023. 5, 6

[15] Zhifang Guo, Jianguo Mao, Rui Tao, Long Yan, Kazushige Ouchi, Hong Liu, and Xiangdong Wang. Audio generation with multiple conditional diffusion model. *AAAI*, 2024. 1

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 1

[17] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *JMLR*, 23(47): 1–33, 2022. 2

[18] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *NeurIPS*, 2022. 1

[19] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *ICLR*, 2023. 1

[20] Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *NeurIPS*, 2023. 5, 6

[21] Jinho Jeong, Sangmin Han, Jinwoo Kim, and Seon Joo Kim. Latent space super-resolution for higher-resolution image generation with diffusion models. *CVPR*, 2025. 2

[22] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. *ICLR*, 2025. 2, 3, 5

[23] Gwanghyun Kim, Hayeon Kim, Hoigi Seo, Dong Un Kang, and Se Young Chun. Beyondscene: Higher-resolution human-centric scene generation with pretrained diffusion. *ECCV*, 2024. 2

[24] Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. Svdqunat: Absorbing outliers by low-rank components for 4-bit diffusion models. *ICLR*, 2025. 2

[25] Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. *NeurIPS*, 2023. 1, 2

[26] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *ICLR*, 2023. 2

[27] Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. *CVPR*, 2025. 2, 6, 7

[28] Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. Audioldm: Text-to-audio generation with latent diffusion models. *ICML*, 2023. 1

[29] Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with taylorseers. *ICCV*, 2025. 1, 2, 6, 7

[30] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *ICLR*, 2022. 2

[31] Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, et al. Sora: A review on background, technology, limitations, and opportunities of large vision models. *arXiv preprint arXiv:2402.17177*, 2024. 1

[32] Ziming Liu, Yifan Yang, Chengruidong Zhang, Yiqi Zhang, Lili Qiu, Yang You, and Yuqing Yang. Region-adaptive sampling for diffusion transformers. *arXiv preprint arXiv:2502.10389*, 2025. 1, 2, 6, 7

[33] Jinming Lou, Wenyang Luo, Yufan Liu, Bing Li, Xinmiao Ding, Weiming Hu, Jiajiong Cao, Yuming Li, and Chenguang Ma. Token caching for diffusion transformer acceleration. *arXiv preprint arXiv:2409.18523*, 2024. 1, 2

[34] Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. *NeurIPS*, 2024. 1, 2

[35] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. *CVPR*, 2024. 2

[36] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal processing letters*, 20(3):209–212, 2012. 5, 6

[37] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *ICML*, 2022. 1

[38] William Peebles and Saining Xie. Scalable diffusion models with transformers. *CVPR*, 2023. 1

[39] 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. *ICLR*, 2024. 2

[40] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *CVPR*, 2022. 1

[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *MICCAI*, 2015. 1

[42] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 2022. 1, 2

[43] Flavio Schneider, Ojasv Kamal, Zhijing Jin, and Bernhard Schölkopf. Moûsai: Efficient text-to-music diffusion models. *ACL*, 2024. 1

[44] Hoigi Seo, Junseo Bang, Haechang Lee, Joohoon Lee, Byung Hyun Lee, and Se Young Chun. Geometrical properties of text token embeddings for strong semantic binding in text-to-image generation. *arXiv preprint arXiv:2503.23011*, 2025. 8

[45] Hoigi Seo, Wongi Jeong, Jae-sun Seo, and Se Young Chun. Skrr: Skip and re-use text encoder layers for memory efficient text-to-image generation. *ICML*, 2025. 2

[46] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. *CVPR*, 2023. 2

[47] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *ICML*, 2015. 1

[48] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ICLR*, 2021. 1

[49] Jiayan Teng, Wendi Zheng, Ming Ding, Wenyi Hong, Jianqiao Wangni, Zhuoyi Yang, and Jie Tang. Relay diffusion: Unifying diffusion process across resolutions for image synthesis. *ICLR*, 2024. 2, 3

[50] Ye Tian, Xin Xia, Yuxi Ren, Shanchuan Lin, Xing Wang, Xuefeng Xiao, Yunhai Tong, Ling Yang, and Bin Cui. Training-free diffusion acceleration with bottleneck sampling. *arXiv preprint arXiv:2503.18940*, 2025. 1, 2, 6, 7

[51] Jianyi Wang, Kelvin CK Chan, and Chen Change Loy. Exploring clip for assessing the look and feel of images. *AAAI*, 2023. 5, 6

[52] Enze Xie, Junsong Chen, Yuyang Zhao, Jincheng Yu, Ligeng Zhu, Yujun Lin, et al. Sana 1.5: Efficient scaling of training-time and inference-time compute in linear diffusion transformer. *ICML*, 2025. 2

[53] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *NeurIPS*, 2023. 3, 4, 5, 6

[54] Haoran You, Connelly Barnes, Yuqian Zhou, Yan Kang, Zhenbang Du, Wei Zhou, et al. Layer-and timestep-adaptive differentiable token compression ratios for efficient diffusion transformers. *CVPR*, 2025. 1, 2

[55] Zhihang Yuan, Hanling Zhang, Lu Pu, Xuefei Ning, Linfeng Zhang, Tianchen Zhao, Shengen Yan, Guohao Dai, and Yu Wang. Ditfastattn: Attention compression for diffusion transformer models. *NeurIPS*, 2024. 1, 2

[56] Linfeng Zhang and Kaisheng Ma. Accelerating diffusion models with one-to-many knowledge distillation. *arXiv preprint arXiv:2410.04191*, 2024. 1, 2

[57] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *ICCV*, 2023. 1

[58] Qingping Zheng, Ling Zheng, Yuanfan Guo, Ying Li, Songcen Xu, Jiankang Deng, and Hang Xu. Self-adaptive reality-guided diffusion for artifact-free super-resolution. *CVPR*, 2024. 1

[59] Tianyi Zheng, Peng-Tao Jiang, Ben Wan, Hao Zhang, Jinwei Chen, Jia Wang, and Bo Li. Beta-tuned timestep diffusion model. *ECCV*, 2024. 3

[60] Shangchen Zhou, Peiqing Yang, Jianyi Wang, Yihang Luo, and Chen Change Loy. Upscale-a-video: Temporal-consistent diffusion model for real-world video super-resolution. *CVPR*, 2024. 1

[61] Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. *ICLR*, 2025. 1, 2, 6