DDiT: Dynamic Patch Scheduling for Efficient Diffusion Transformers

Dahye Kim[1,2*] Deepti Ghadiyaram[1] Raghudeep Gadde[2]
[1]Boston University [2]Amazon
{dahye, dghadiya}@bu.edu raghudeep.g@gmail.com

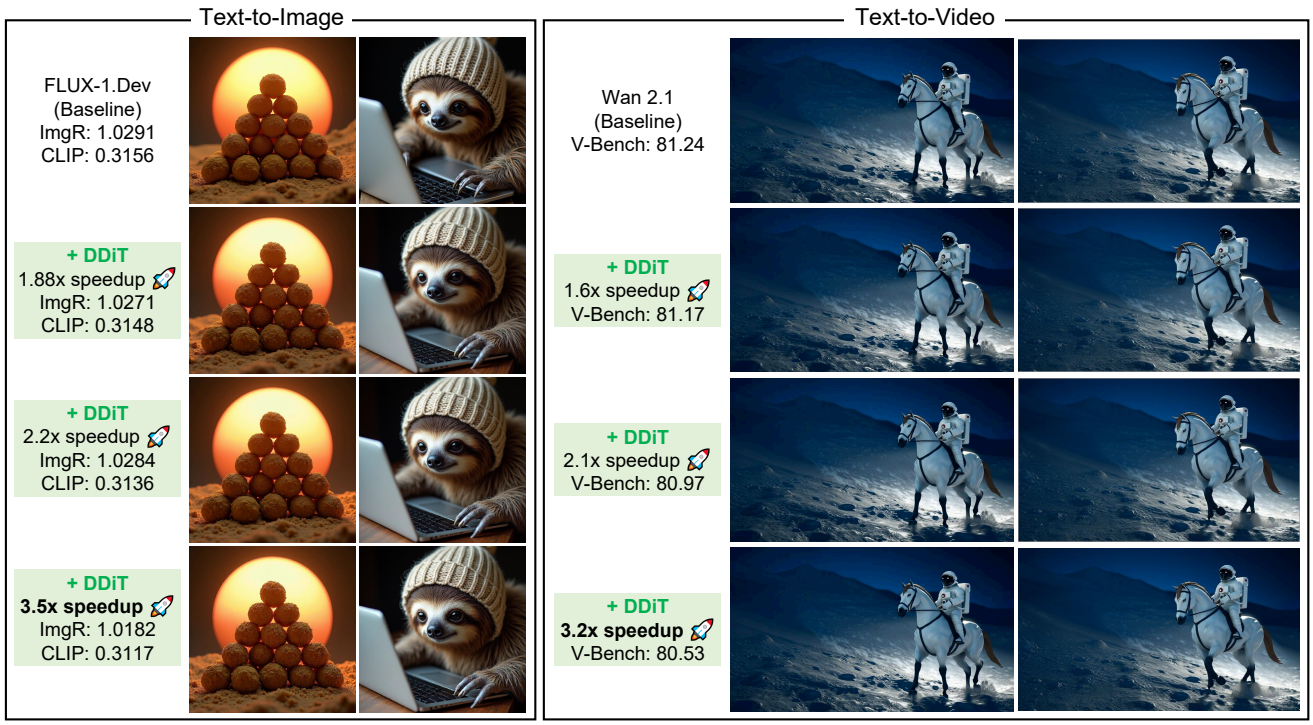


Figure 1. **DDiT dynamically selects the optimal patch size** at each denoising step at inference yielding significant computational gains at no loss of perceptual quality. Results are shown for FLUX-1.Dev [54] for text-to-image and Wan-2.1 [107] for text-to-video generation. The top panel denotes the baseline (original model), while the remaining panels illustrate outputs from DDiT at different acceleration rates. ImageReward [118], CLIP [83], and VBench [43] scores are reported (higher is better).

Abstract

Diffusion Transformers (DiTs) have achieved state-of-the-art performance in image and video generation, but their success comes at the cost of heavy computation. This inefficiency is largely due to the fixed tokenization process, which uses constant-sized patches throughout the entire denoising phase, regardless of the content's complexity. We propose dynamic tokenization*, an efficient* ***test-time strategy*** *that varies patch sizes based on content complexity and the denoising timestep. Our key insight is that early timesteps only require coarser patches to model global structure, while later iterations demand finer (smaller-sized) patches to refine local details. During inference, our method dynamically reallocates patch sizes across denoising steps for image and video generation and substantially reduces cost while preserving perceptual generation quality. Extensive experiments demonstrate the effectiveness of our approach: it achieves up to* $3.52\times$ *and* $3.2\times$ *speedup on FLUX-1.Dev and Wan* 2.1*, respectively, without compromising the generation quality and prompt adherence.*

1. Introduction

Diffusion transformers (DiTs) [8, 22, 53, 54, 81, 92, 117] have emerged as a dominant framework for content generation, producing high-quality and photorealistic results in both image and video synthesis. These advances have facilitated a wide range of applications, including image and video editing [7, 48], subject-driven generation [86, 113],

*Work done as an intern at Amazon.

and digital art creation [74]. However, this impressive performance comes with substantial computational cost – generating a *single* 5 second 720p video using Wan-2.1 [107] on an RTX 4090 takes 30 minutes! – significantly limiting the usage of these models in practice. Such high computational demands of generative models have catapulted the development of more efficient generation methods. Existing research has broadly focused on acceleration techniques such as feature caching [61, 62, 71, 128], feature pruning [6, 25, 108, 125], vector quantization [18, 94, 97, 104], and model distillation [49, 59, 91, 129].

Although these approaches show promise, they suffer from two key limitations. First, many methods [26, 40, 71, 115] typically employ a **hard, static reduction strategy**, such as removing a fixed amount of weights, operations, or tokens. Such static approach can lead to significant quality degradation, as computations critical to a specific output might be permanently discarded [18, 115]. Second, most existing methods [71, 72, 115] apply a rigid, one-size-fits-all strategy, that is agnostic to the input. This is problematic, as different prompts require varying levels of computational detail [73, 110]. A simple prompt like "a blue sky" should not require the same amount of computational resources compared to a prompt "a scene crowded with many zebras." The rigidity in all existing solutions prevents us from dynamically allocating resources where they are needed most.

In this work, we address the rigid, one-size-fits-all computation of existing methods. Our approach is based on a key observation: the visual content generated by a diffusion model evolves at varied levels of detail. Some denoising timesteps establish coarse scene structure, while others refine fine-grained visual details. Recent studies [50, 56, 99, 103] show that features generated at different timesteps of the denoising process encode different information, thus selecting the right timestep of diffusion features is important for successful downstream tasks such as classification [56], visual reasoning [50], visual correspondence [103], and semantic segmentation [99]. Furthermore, [80, 110] note that this information can also be used for image editing, catering to different levels of detail in an image [73, 80, 110], and for generating more prompt-aligned images by injecting different levels of prompt information at different timesteps [90].

This leads us to a critical question: **should every denoising step process the latent at the same granularity?** Or, could some steps operate on a coarser latent, thereby yielding computational benefits, while others use a finer latent to preserve detail? Thus, unlike prior works [6, 25, 108, 125] which approach efficiency by discarding weights or operations, we *dynamically allocate* it. Specifically, at every timestep, we adjust the patch size of the latent and adaptively use larger patches (coarser granularity) when less detail is required and smaller patches (finer granularity) when high fidelity is needed.

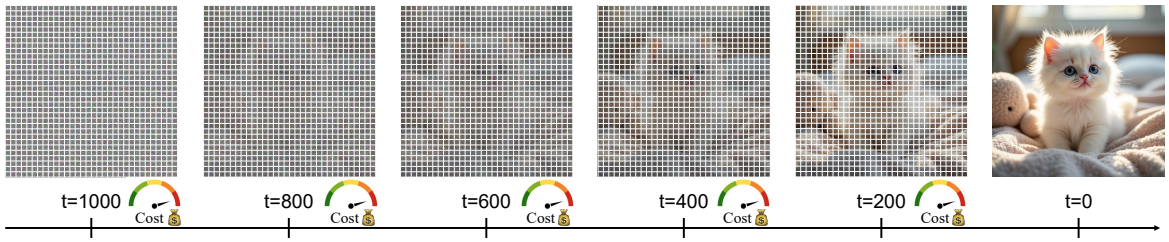
This, however, raises a new question: **how do we determine the optimal patch size at any given timestep and for any given prompt?** For this, we measure the *rate of change of the latent manifold* over time. We hypothesize that this rate correlates with the level of detail being generated. If the underlying latent evolves slowly within a short timestep window, we posit that coarse-grained details are being generated. Consequently, we divide the latent into coarser patches and process them, saving computational resources. Conversely, if the underlying latent evolves rapidly, we infer that fine-grained details are being generated and fall back to using finer-grained latent patches.

Thus, this dynamic strategy tailors the computation load to each timestep and each prompt, allocating more resources when needed and conserving them where possible. Ultimately, our approach gives us explicit control over the computational budget while generating the highest possible quality content given the computational budget (Fig. 1). In summary, our contributions are:

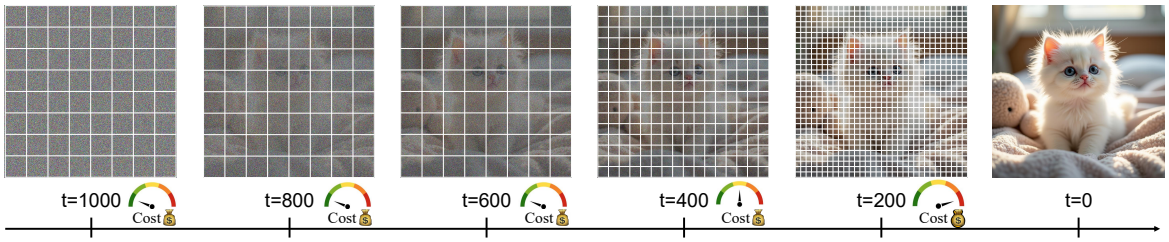
- We introduce a simple, intuitive, and low-cost strategy to **dynamically vary latent's granularity** in diffusion models, that requires minimal architectural changes (Fig. 2).
- We propose a test-time **Dynamic Patch Scheduler** that automatically determines the optimal patch size at each timestep, adapting the computational load based on generation complexity and the input prompt.
- We demonstrate through extensive experiments that our approach generalizes across both image and video diffusion transformers and achieves significant speedups – up to 3.52× on FLUX-1.Dev [54] and 3.2× speedups on Wan 2.1 [107], while maintaining high perceptual quality, photo-realism, and prompt alignment.
- We provide a detailed analysis of the rate of latent manifold evolution to generative complexity, offering a new perspective on the internal dynamics of diffusion models.

2. Related work

Efficient diffusion transformers. Diffusion transformers incur substantial computational costs due to their iterative denoising and attention operations. To address this challenge, a growing body of work has focused on improving the efficiency of these models through various algorithmic and architectural strategies. Fast sampling methods [29, 32, 63, 67, 68, 75, 77, 79, 91, 98, 120, 127, 132] reduce the number of sampling steps while preserving output quality. Caching-based methods [14, 27, 30, 42, 47, 58, 61, 61, 62, 64, 66, 70–72, 87, 93, 102, 116, 121, 124, 126, 128, 136] improve efficiency by reusing previously computed intermediate representations to avoid redundant computation. Pruning-based methods [3, 6, 9, 16, 25, 26, 40, 46, 51, 52, 55, 66, 69, 87, 95, 96, 100–102, 105, 108, 109, 119, 122, 125, 126, 131, 134] accelerate inference by removing redundant or less informative



(a) Denoising process of standard diffusion transformers (DiTs)



(b) Denoising process of DDiT (Ours)

Figure 2. **Main idea: dynamic tokenization during denoising**. Current methods use the same patch size for *all* denoising steps during inference time. Instead, **DDiT** adapts the patch size at each timestep according to the latent complexity, allocating fewer tokens for certain timesteps and more tokens for certain others. While DiT divides VAE latents into patches, for illustrative purposes, we use a real image in pixel space.

model weights, thereby reducing the number of operations. Quantization-based methods [12, 18, 20, 24, 57, 94, 97, 104] improve efficiency by converting model weights and activations from high-precision to low-precision representations, such as 8-bit integers [19]. Knowledge distillation methods [11, 28, 49, 59, 78, 91, 129, 135] achieve efficiency by compressing complex models into smaller version using distillation objectives [35]. Although these approaches have shown promising results in reducing computation, they typically rely on hard, predefined reduction rules that lack adaptivity to content complexity. Such hard constraints often discard essential details or oversimplify fine structures, ultimately degrading generation quality. In contrast, we dynamically allocate computation across timesteps for efficient yet high-quality generation.

Dynamic patch sizing for efficient transformers. Several prior works have explored using multiple patch sizes in transformer-based architectures. Methods such as [5, 10, 41, 111, 112, 133] train models capable of operating with different patch sizes across images in ViTs. To further enhance efficiency, subsequent approaches [1, 4, 13, 17, 85] enable adaptive patch sizes within a single image, allowing the model to allocate computation based on local content complexity. Similarly, several works have investigated using different patch sizes or resolutions in DiTs [2, 9, 15, 31, 37, 38, 45, 82, 88, 89]. However, all of these methods either 1) require training from scratch with sophisticated architectural designs, 2) are not generalizable to existing off-the-shelf pretrained DiTs, or 3) use a rigid and manually defined schedule for patch size during inference. We propose *DDiT*, a generic framework that dynamically adjusts patch sizes during test time for efficient generation.

3. Approach

Our goal is to achieve significant computational speedup at minimal loss of perceptual quality of image and video generations. We achieve this by *dynamically* varying the patch size of a latent at each denoising timestep based on the complexity of the underlying latent manifold. We first briefly introduce diffusion transformers (DiT) [81] in Sec. 3.1, motivate our approach to adapt DiT to dynamically process latent patches of different sizes in Sec. 3.2, and finally detail our novel approach to dynamically select the optimal latent patch size at every denoising step in (Sec. 3.3).

3.1. Preliminaries on Diffusion Transformers

Owing to the flexibility and scalability of the transformer architecture [106], DiTs have achieved wide adoption in content generation. Built upon the Vision Transformer (ViT) architecture [21], DiTs operate in the latent space of a pretrained variational autoencoder (VAE) [84]. Briefly, given an input image I[1], it is first encoded by the VAE into a latent representation $\mathbf{z} \in \mathbb{R}^{H \times W \times C}$, where H, W, and C denote the height, width, and channel dimensions of the latent feature map, respectively. The input to the transformer-based diffusion model is this latent $\mathbf{z}$. During training, Gaussian noise is gradually added to $\mathbf{z}$, and DiT is optimized to predict and remove this noise [36]. During inference, the model starts from pure noise and iteratively denoises it over T diffusion steps to recover a clean latent representation, which is then decoded by the VAE decoder to reconstruct the final image.

[1]For simplicity, we use image inputs, but our method is extensible to DiTs which process videos as we show in Sec. 4.

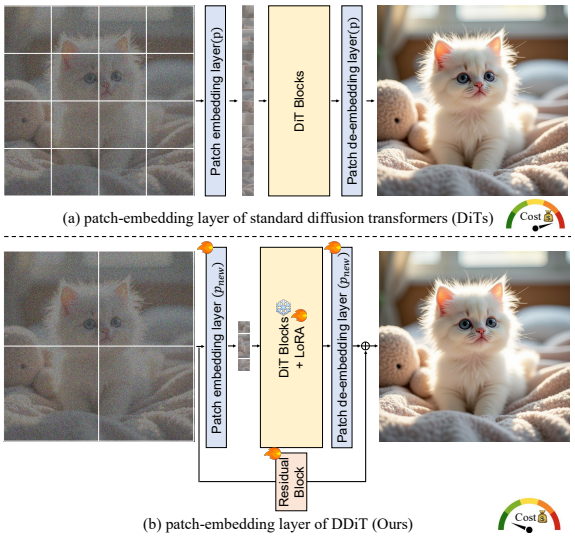


Figure 3. **Revised patch-embedding layer to support patches of varied resolutions.** We modify the standard patch-embedding layer, designed for a fixed patch size p, to additionally support patch sizes p_{new}.

Note that the latent z is pre-processed before feeding to the diffusion transformer. Specifically, z is first divided into non-overlapping patches of size $p \times p$. Following this, each patch is *tokenized* by passing through a patch embedding layer parameterized by weights $\mathbf{w}^{\text{emb}} \in \mathbb{R}^{p \times p \times C \times d}$ and bias $\mathbf{b}^{\text{emb}} \in \mathbb{R}^d$. This layer projects each patch into an embedding space of dimension d. The resulting embeddings from each patch are then processed by L stacked transformer blocks comprising a series of attention and feed-forward layers [106]. The attention mechanism learns to attend to relevant patches by computing pairwise dependencies among all $N = \frac{HW}{p^2}$ patches. Thus, attention operation has a computational complexity proportional to $\mathcal{O}(N^2)$.

Naturally, a smaller p increases the number of tokens N, leading to an expensive attention operation, thereby higher computational cost per layer. Further, since denoising is an iterative operation, using a small patch size p is even more computationally prohibitive. Moreover, prior studies have shown that not all denoising steps need the same level of granularity [50, 73, 80, 99, 103, 110]. These factors motivate us to vary the patch size p dynamically across timesteps to balance efficiency and generation quality.

3.2. Dynamic Patching and Tokenization

We aim to modify a pre-trained DiT to seamlessly operate under different patch sizes with minimal architectural modifications (Fig. 3). To this end, we adapt the patch embedding layer, originally operating on patch size p, to also handle new patch sizes p_{new} and allow input latents of varying spatial resolutions. We define p_{new} a positive integer multiple of p, *i.e.*, $\{p, 2p, 4p, ...\}$.

Modifications to the patch embedding layer. To generalize DiT to p_{new}, we introduce patch-specific embedding layers for each patch size we wish to support. Recall from Sec. 3.1 that C denotes the number of latent channels and d represents the embedding dimension. Let $\mathbf{w}^{\text{emb}}_{p_{\text{new}}} \in \mathbb{R}^{p_{\text{new}} \times p_{\text{new}} \times C \times d}$ and $\mathbf{b}^{\text{emb}}_{p_{\text{new}}} \in \mathbb{R}^d$ denote the weight matrix and bias vector of the patch embedding layer corresponding to p_{new}. Each patch of size p_{new} is linearly projected into an embedding of dimension d using this newly added embedding layer. This results in a total of $N_{p_{\text{new}}} = \frac{HW}{p^2_{\text{new}}}$ patches. Since $N_{p_{\text{new}}}$ is smaller than N by a factor of $(p_{\text{new}}/p)^2$, DiT now processes fewer patches and yields significant computational gains. As shown in Fig. 4, increasing the patch size from p to $2p$ yields a $3\times$ computational gain!

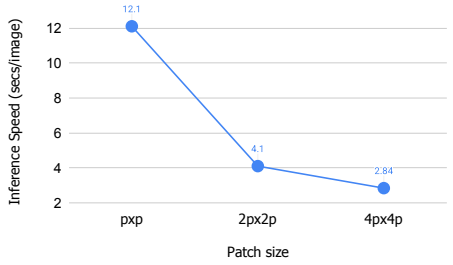


Figure 4. **Inference speed vs. patch size.** Inference speed measured over 50 denoising steps for generating 1024×1024 images using FLUX-1.Dev [54], where every timestep uses a fixed patch size. As the patch size increases from $p \rightarrow 2p \rightarrow 4p$, the number of tokens decreases quadratically ($4096 \rightarrow 1024 \rightarrow 256$), resulting in approximately $3\times$ and $4\times$ faster inference for $2p$ and $4p$, respectively, compared to p.

To minimize the training cost, we retain the base model originally trained on the latent patch size p and introduce a Low-Rank Adaptation (LoRA) branch [39] into each transformer block in DiT. This LoRA branch serves as an *adaptive pathway* and enables the model to process patches of different sizes. Additionally, as shown in Fig. 3, we add a residual connection from *before* the patch embedding layer to *after* the patch de-embedding block. This helps strike a balance between the base latent manifold and the new manifold being learnt by LoRA for p_{new}.

We reuse the learnt positional embeddings of the original patch size p for p_{new} by bilinearly interpolating them for the new patch size. We also introduce a learnable patch embedding (a d-dimensional vector) added to all tokens akin to positional embeddings. This serves as a patch-size identifier and helps the model distinguish which patch size is being used at each timestep. At test time, we use the learned patch-size embedding as is.

Finally, to distill the knowledge from the frozen base model to the LoRA-augmented model, we fine-tune the LoRA branch with a distillation loss. Let ϵ_{θ_L} and ϵ_{θ_T} denote the predicted noise from the LoRA-fine-tuned and frozen base models respectively. The distillation loss is:

$$\mathcal{L} = ||\epsilon_{\theta_L}(\mathbf{z}_t^{p_{\text{new}}}, t) - \epsilon_{\theta_T}(\mathbf{z}_t^{p}, t)||_2^2. \quad (1)$$

These minor architectural tweaks allow us to dynamically support larger patch sizes while maintaining the base

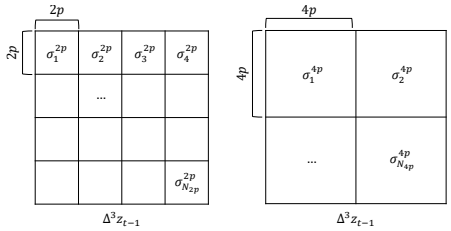


Figure 5. Given $\Delta^{(3)}\mathbf{z}_{t-1}$, we divide it into patches of size $p_i \times p_i$, compute within-patch standard deviation $\boldsymbol{\sigma}_{t-1}^{p_i}$ of the acceleration.

model's perceptual output quality. We stress and empirically show in Sec. 4 that these changes are seamlessly extensible to any diffusion-based image or video models.

3.3. Dynamic Patch Scheduling

Now that we enabled processing of multiple size input patches, how do we learn *when* to adapt to a larger patch-size (*i.e.*, coarser token) and when to switch back to a smaller patch-size (*i.e.*, fine-grained token)? To this end, we introduce a dynamic patch scheduling mechanism to determine the appropriate patch size at each diffusion timestep. Since different timesteps correspond to different levels of generative detail [73, 80, 90, 99, 103, 110], selecting the proper patch size at each stage is crucial for maintaining both efficiency and quality. We hypothesize that:

- large patches may reasonably capture coarse scene structures without significant compromises of visual quality while yielding computational speedups.
- smaller patches may be pertinent to capture fine-grained details to retain all visual intricacies.

We automate this intuition in a highly light-weight manner and design a **training-free dynamic scheduler** that adaptively selects the patch size based on the *rate of evolution of the latent representations* within a window of timesteps.

Latent evolution estimation. We employ finite-difference approximations of increasing order to quantify how latent representations evolve during the denoising process. Let $\mathbf{z}_t$ denote the latent at timestep t. The first-order finite difference captures the displacement of latent features between consecutive timesteps:

$$\Delta\mathbf{z}_t = \mathbf{z}_t - \mathbf{z}_{t+1}. \tag{2}$$

Similarly, the second-order difference describes the *rate* of change of this displacement, representing the local velocity of the denoising trajectory, defined by

$$\Delta^{(2)}\mathbf{z}_{t-1} = \Delta\mathbf{z}_{t-1} - \Delta\mathbf{z}_t. \tag{3}$$

Finally, the third-order finite difference quantifies the variation in this velocity. This can be interpreted as a measure of *acceleration* of a latent's evolution during denoising within a short temporal window.

$$\Delta^{(3)}\mathbf{z}_{t-1} = \Delta^{(2)}\mathbf{z}_{t-1} - \Delta^{(2)}\mathbf{z}_t = 2\left(\frac{\Delta\mathbf{z}_{t-1} + \Delta\mathbf{z}_{t+1}}{2} - \Delta\mathbf{z}_t\right), \tag{4}$$

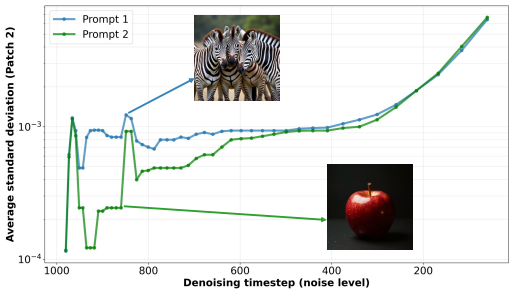


Figure 6. **Visualization of $\boldsymbol{\sigma}_{t-1}^{2p,(\rho)}$ for two prompts (log scale).** *Prompt 1: "Several zebras are standing together behind a fence." Prompt 2: "A simple red apple on a black background."* Prompts requiring different levels of spatial granularity exhibit distinct $\boldsymbol{\sigma}_{t-1}^{2p,(\rho)}$ patterns across timesteps. For the fine-grained zebra pattern, $\boldsymbol{\sigma}_{t-1}^{2p,(\rho)}$ remains higher, indicating higher detail sensitivity, whereas for the simpler apple scene, $\boldsymbol{\sigma}_{t-1}^{2p,(\rho)}$ is lower, thus we can seamlessly use larger patch sizes during generation.

We hypothesize that if the acceleration is slow at a given timestep, there is a relatively minor difference in the underlying latent manifold in the local temporal window. On the other hand, **a high acceleration value suggests a larger difference in the structure of the underlying manifold.** We use this measure as a proxy to identify transition points where the generative process intrinsically shifts between generating coarse to fine structures or vice versa.

Empirically, we find that the third-order difference captures this variation more effectively and remains more stable, while the first- and second-order differences fail to do so, likely because they capture relatively short-term temporal changes (Sec. 4.4). This observation is consistent with [121], which shows that the difference between neighboring noise predictions is explicitly related to the third-order finite difference.

Spatial variance estimation. We use latent z_t in the above formulation for simplicity, but in practice, z_t is always divided into patches of size $p_{\text{new}} \times p_{\text{new}}$ (Sec. 3.2). Our final task now is to select the right patch size at each latent manifold. This requires quantifying and aggregating the acceleration at which the latent patches evolve. Thus, we divide z_{t-1} into patches of size $p_i \times p_i$, where $p_i \in p_{\text{new}}$. Then, we compute the **standard deviation** $\boldsymbol{\sigma}_{t-1}^{p_i}$ of the acceleration (defined in Eqn. 4) within each patch (Fig. 5). We hypothesize that if the per-(latent) pixel standard deviation within a latent patch is high, then the denoising process is focusing on generating finer-grained details. On the other hand, if the standard deviation is low, then the underlying evolving latent is smooth. As shown in Fig. 6, prompts with different levels of granularity exhibit distinct variance profiles across timesteps.

Given $\boldsymbol{\sigma}_{t-1}^{p_i}$, our goal is to determine the appropriate patch size at each timestep. A straightforward way to do this is to aggregate $\boldsymbol{\sigma}_{t-1}^{p_i}$ by taking their mean across patches, which provides a simple measure of the overall latent vari-

Table 1. **Quantitative comparison of text-to-image generation performance with state-of-the-art methods** on COCO, DrawBench, and PartiPrompts. If not specified, all results are reported using 50 inference steps by default. Each color (**Yellow** , **Blue**) indicates methods operating at similar inference speeds. As highlighted in **Blue** , our method achieves the best overall image quality, evidenced by the lowest FID scores, strong prompt alignment (CLIP and ImageReward), and high perceptual similarity (SSIM and LPIPS). **Bold**: best. Underline: second-best.

Model	Speed↓ (secs/image)	COCO		DrawBench				PartiPrompts	
		FID↓	CLIP↑	CLIP↑	ImgR↑	SSIM↑	LPIPS↓	CLIP↑	ImgR↑
FLUX-1.Dev (50 steps)	12.0	33.07	0.314	0.3156	1.0291	–	–	0.3197	1.192
FLUX-1.Dev (28 steps)	6.72	33.35	0.312	0.3140	1.0107	0.739 ± 0.155	0.175 ± 0.126	0.3118	1.115
FLUX-1.Dev (15 steps)	3.6	34.02	0.311	0.3121	0.9865	0.591 ± 0.155	0.298 ± 0.128	0.3072	0.9613
TeaCache ($\delta = 0.6$)	6.0	34.95	0.303	0.3071	0.9968	$\mathbf{0.654 \pm 0.145}$	$\mathbf{0.241 \pm 0.114}$	0.3018	0.9701
TeaCache ($\delta = 0.8$)	5.33	35.57	0.303	0.3075	0.9780	0.612 ± 0.146	0.279 ± 0.114	0.2991	0.9699
TaylorSeer ($N = 3, O = 2$)	6.0	34.74	0.303	0.3085	0.9721	0.632 ± 0.173	0.271 ± 0.149	0.3142	0.9813
TaylorSeer ($N = 6, O = 1$)	3.5	35.02	0.302	0.3040	0.9535	0.583 ± 0.130	0.284 ± 0.106	0.3004	0.9714
DDiT	5.5	**33.42**	**0.317**	**0.3136**	**1.0284**	0.635 ± 0.183	0.264 ± 0.190	**0.3192**	**1.189**
DDiT + Teacache ($\delta = 0.4$)	3.4	33.60	0.315	0.3117	1.0182	0.592 ± 0.118	0.267 ± 0.120	0.3172	1.062

ation at that timestep. However, this fails to effectively capture the generative dynamics occurring at that timestep. For example, when generating an image containing both a uniform white background and a highly textured region, averaging might smoothen the higher standard deviation values, leading to the scheduler choosing larger patches and thus overlooking fine details in the textured area. To better capture such spatial heterogeneity in the underlying latent manifold, we instead take the ρ-th percentile of the per-patch variances, denoted as $\sigma_{t-1}^{p_i,(\rho)}$. This percentile-based aggregation allows us to capture meaningful information across patches without averaging out important signals, while also avoiding bias toward a few high-variance outliers.

Concretely, we compare $\sigma_{t-1}^{p_i,(\rho)}$ against a predefined variance threshold τ. For each timestep, we select the largest patch size whose corresponding variance is below the threshold (τ). If no such patch satisfies this condition, it defaults to the smallest patch size, which is 1. We formulate this **patch size scheduling** as follows:

$$p_t = \begin{cases} \max(p_i), & \text{if } \sigma_{t-1}^{p_i,(\rho)} < \tau, \\ 1, & \text{otherwise.} \end{cases} \tag{5}$$

Controlling τ gives us explicit control over the speed: if users prefer faster generation, a higher τ can be selected; otherwise, a smaller τ can be used for higher quality with less speed gain. We select τ and ρ empirically and balance generation stability and visual quality. In Sec. 4, we also show how such adaptive scheduling enables the generation process to allocate computational resources efficiently and strategically, while maintaining the overall generation fidelity.

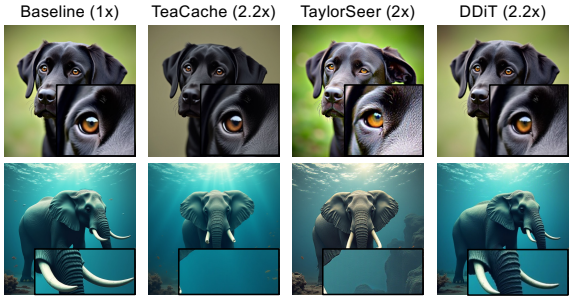


Figure 7. Qualitative comparisons with the base model [54], TeaCache [61], TaylorSeer [62], and DDiT under similar speedups on DrawBench. **DDiT** effectively preserves fine-grained details, pose, spatial layout, and overall color distribution of the generated images.

4. Experiments

4.1. Setup

Implementation details. We use FLUX-1.dev [54] and Wan-2.1 1.3B [107] as base models for the text-to-image (T2I) and text-to-video (T2V) experiments, respectively. To support new patch sizes p_{new}, we introduce corresponding patch embedding and de-embedding layers for the patchify operation, along with corresponding patch positional embeddings. We also add LoRA parameters [39] with a rank of 32 into the feed-forward layers of each transformer block and a single residual block, which are then fine-tuned along with the newly introduced components. For both T2I and T2V models, we support patch sizes $p_{\text{new}} = 2p, 4p$, although our method in principle can be extended to any size patches. For both T2I and T2V tasks, we use 50 inference steps for comparison, but our method can be applied with any number of inference steps. The T2I model is finetuned on the T2I-2M dataset [44], a synthetic dataset generated using the base model, and the T2V model is trained on synthetic videos generated by the base model using prompts from the Vchitect-T2V-Dataverse [23]. We use Prodigy [76], an optimizer that automatically finds the optimal learning rate without requiring manual tuning, with a learning rate of 1.0 for T2I. For the T2V model, we employ AdamW [65] with a learning rate of 1×10^{-4}. We initialize the patch-embedding weights using the pseudo-inverse of the bilinear-interpolation projection following [5], which helps preserve the base model's functional behavior. To balance visual fidelity and computational efficiency, we set $\tau = 0.001$ and $\rho = 0.4$ for all experiments.

Figure 8. **Qualitative comparison on DrawBench** with the baseline and TaylorSeer [62]. Our method remains robust even for complex prompts that require a deeper understanding of semantic content.

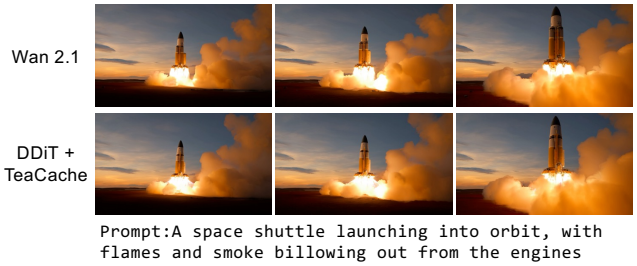


Figure 9. **Qualitative comparison of text-to-video generation between DDiT and the baseline.** DDiT produces videos with comparable visual quality to the baseline while achieving significant speedup.

Evaluation setup. For evaluation, we generate images at a resolution of 1024×1024 using 50 inference steps and a guidance scale of 3.5 for the text-to-image task, and videos at 480×832 resolution with 81 frames using 50 inference steps for the text-to-video task. As commonly done, for text-to-image evaluation, we use the COCO dataset [60] to compute CLIP [33, 83] and FID [34] scores against real images, assessing overall visual quality. We additionally evaluate on DrawBench [88] and PartiPrompts [123] datasets using CLIP score and ImageReward [118] to measure text–image alignment, and SSIM [114] and LPIPS [130] to assess structural similarity with the base model. For text-to-video evaluation, we adopt VBench [43] and follow the evaluation protocol proposed in their work.

4.2. Text-to-Image Generation

We first evaluate the effectiveness of our approach on the text-to-image (T2I) generation task and compare it with state-of-the-art acceleration methods. Our evaluation focuses on measuring both efficiency and perceptual quality, as reducing inference cost often leads to a loss in fine-grained visual details or degraded text–image alignment. We use FLUX-1.dev [54] as the base model and vary the number of inference steps to simulate different computational budgets. Our approach is compared against TeaCache [61] and TaylorSeer [62], two state-of-the-art caching-based acceleration methods, under multiple configurations to ensure a fair comparison.

Table 2. **Quantitative results on V-Bench [43].** Comparison of DDiT under different threshold settings (τ) and its combination with TeaCache [61].

Model	Speed (↑)	VBench (↑)
Wan-2.1 (Baseline)	1.0×	81.24
Ours (τ=0.004)	1.6×	81.17
Ours (τ=0.001)	2.1×	80.97
Ours (τ=0.001) + TeaCache (δ=0.05)	3.2×	80.53

As shown in Table 1 and Fig. 7, our method achieves substantial improvements in inference speed while maintaining high generation quality. Compared to the base model, our approach achieves comparable FID (only a 0.35 difference) and CLIP scores, while delivering a $\mathbf{2.18}\times$ speedup. This demonstrates that dynamically adjusting patch sizes across denoising steps enables more efficient computation without compromising perceptual fidelity. Under similar inference speeds (rows 4, 6, and 8), our method consistently outperforms prior approaches. As shown in Fig. 8, our model seamlessly handles complex cases that require a deeper understanding of the semantic content of the prompt. Moreover, our method preserves the overall perceptual similarity to the base model's output while substantially reducing inference cost.

Combining with TeaCache [61]: Furthermore, our approach is complementary to existing acceleration strategies such as caching. When combined with TeaCache (row 9), our method achieves a $\mathbf{3.52}\times$ speedup over the baseline! This **surpasses all existing state-of-the-art approaches** in both efficiency and generation quality. These results confirm that our dynamic patch scheduling strategy effectively balances computation and quality, offering a simple yet powerful mechanism for efficient diffusion generation.

4.3. Text-to-Video Generation

We further evaluate the effectiveness of our method in the text-to-video (T2V) generation setting and compare it with the base model [107]. Our method dynamically adjusts the patch size across denoising steps, allowing the model to allocate computation adaptively according to the complexity of spatial structures.

As shown in Table 2, our approach significantly reduces inference time while maintaining competitive video quality, as reflected by the VBench score [43]. Qualitative results in Fig. 9 show that our method preserves motion consistency and fine-grained frame details even at accelerated inference speeds. Additional results are in Appendix.

4.4. Analysis

In this section, we conduct an extensive analysis of our method to better understand our method.

Effect of speedup on visual quality: a user study. To assess whether humans can distinguish between DDiT and

Table 3. **Effect of the n-th order difference on generation quality.** Higher-order terms capture more informative temporal dynamics, improving both FID and CLIP scores. The third-order term ($n = 3$) achieves the best overall performance.

Method	FID↓	CLIP↑	ImageReward↑
DDiT ($n = 1$)	34.71	0.2927	0.9782
DDiT ($n = 2$)	34.28	0.3082	1.0128
DDiT ($n = 3$)	**33.42**	**0.3136**	**1.0284**

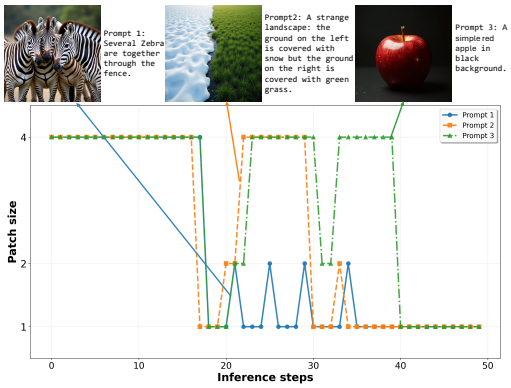


Figure 10. **Sample patch schedules** for 3 different prompts. Our dynamic patch scheduler seamlessly adapts to each prompt's complexity and detail, thereby allocating more computation (aka higher percentage of smaller patch sizes) to images with highly detailed textures compared to simpler ones, thereby balancing efficiency and visual quality.

baseline generations, we conduct a user study on visual preference. Raters were shown image pairs (DDiT vs. baseline) presented side by side in random order and asked to select the image with higher visual quality. We find that generations from DDiT are visually as pleasing and photo-realistic as DiT $\mathbf{61}\%$ of the time, while DiT generations are preferred over DDiT 22% of the time. Surprisingly, we find that DDiT generations are preferred over DiT baseline 17% times, even though this was not our main goal. These results clearly demonstrate that DDiT achieves visual quality on par with the baseline while providing substantial speedup.

Effect of n-th order difference equation. Table 3 shows the impact of employing different n-th order terms in our latent variation estimation. As n increases, both FID and CLIP scores consistently improve, suggesting that higher-order differences capture richer and more informative temporal dynamics of the latent space throughout the denoising process. In particular, the third-order term ($n = 3$) achieves the best overall performance, producing the lowest FID and the highest CLIP and ImageReward scores.

Effect of the patch schedule across different prompts. We examine how our dynamic patch scheduling mechanism adapts to different text prompts with varying levels of complexity. As illustrated in Fig. 10, our method automatically adjusts the patch schedule based on the semantic and structural richness of each prompt, effectively reallocating computational resources throughout the denoising process. For prompts describing complex scenes that involve fine-grained textures, the scheduler assigns more denoising steps with finer patches to capture detailed visual information. Conversely, for simpler prompts that depict minimal structures or uniform backgrounds, the model adaptively switches to coarser patches, thereby reducing redundant computation and accelerating inference. This adaptive behavior allows the model to balance efficiency and quality on a per-prompt basis, ensuring that computational effort is concentrated where it contributes most to perceptual fidelity. Overall, this demonstrates that our patch scheduling strategy not only accelerates generation but also enables content-aware allocation of computation, leading to improved scalability and robustness across diverse prompt distributions.

Table 4. **Effect of the threshold τ on DrawBench.** Higher τ values yield faster inference at very mild dip in generation quality.

Method	Speed (×)	CLIP	ImageReward
DDiT (τ=0.004)	1.88	0.3148	1.0271
DDiT (τ=0.001)	2.18	0.3136	1.0284
DDiT (τ=0.01)	3.52	0.3082	1.0124

Effect of the threshold on patch scheduling. We analyze the impact of varying the threshold τ used in our patch scheduling mechanism, which determines when to switch between coarse and fine patch sizes during denoising. As shown in Table 4, increasing τ results in slightly lower visual quality across all metrics, including FID, CLIP, and ImageReward scores. This trend can be attributed to the patch size scheduler becoming less sensitive to temporally local variations of the latent manifold at higher thresholds, leading to the premature selection of coarser patches and loss of fine-grained details. Nevertheless, the degradation remains small, confirming the robustness of our scheduling strategy. To balance visual fidelity and computational efficiency, we set $\tau = 0.001$ for all experiments.

5. Conclusion and Future Work

We present an intuitive and highly-computationally efficient method, *DDiT*, to adapt diffusion transformers to patches of different sizes during denoising while maintaining visual quality. *DDiT* demonstrates a critical insight: not all timesteps require the underlying latent space to be equally fine-grained. Building on this insight, we dynamically select the optimal patch size at every timestep and achieve significant computational gains, with no loss in perceptual visual quality. Our approach requires just adding a simple plug-and-play LoRA adapter to make the patch-embedding (and de-embedding) blocks amenable to varied input patch sizes. This minimal architectural tweak allows any DiT-based model to benefit from fast inference. Notably, it can also be applied to long-video generation, allowing the model to generate longer videos with the same amount of compute. In our current design, for a given timestep, we use a fixed patch-size, but vary patch-sizes across timesteps. A natural future research would involve investigating varied patch sizes *within* a given timestep, for further efficiency.

References

[1] Xiaoqi An, Lin Zhao, Chen Gong, Nannan Wang, Di Wang, and Jian Yang. Sharpose: Sparse high-resolution representation for human pose estimation. In *AAAI*, 2024. 3

[2] Yuval Atzmon, Maciej Bala, Yogesh Balaji, Tiffany Cai, Yin Cui, Jiaojiao Fan, Yunhao Ge, Siddharth Gururani, Jacob Huffman, Ronald Isaac, et al. Edify image: High-quality image generation with pixel space laplacian diffusion models. *arXiv preprint arXiv:2411.07126*, 2024. 3

[3] Omri Avrahami, Or Patashnik, Ohad Fried, Egor Nemchinov, Kfir Aberman, Dani Lischinski, and Daniel Cohen-Or. Stable flow: Vital layers for training-free image editing. In *CVPR*, 2025. 2

[4] Zhuhua Bai, Weiqing Li, Guolin Yang, Fantong Meng, Renke Kang, and Zhigang Dong. A coarse-to-fine framework for point voxel transformer. In *CSCWD*, 2024. 3

[5] Lucas Beyer, Pavel Izmailov, Alexander Kolesnikov, Mathilde Caron, Simon Kornblith, Xiaohua Zhai, Matthias Minderer, Michael Tschannen, Ibrahim Alabdulmohsin, and Filip Pavetic. Flexivit: One model for all patch sizes. In *CVPR*, 2023. 3, 6

[6] Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *CVPR*, 2023. 2

[7] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 1

[8] Qi Cai, Jingwen Chen, Yang Chen, Yehao Li, Fuchen Long, Yingwei Pan, Zhaofan Qiu, Yiheng Zhang, Fengbin Gao, Peihan Xu, et al. Hidream-i1: A high-efficient image generative foundation model with sparse diffusion transformer. *arXiv preprint arXiv:2505.22705*, 2025. 1

[9] Shuning Chang, Pichao Wang, Jiasheng Tang, and Yi Yang. Flexdit: Dynamic token density control for diffusion transformer. *arXiv preprint arXiv:2412.06028*, 2024. 2, 3

[10] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *ICCV*, 2021. 3

[11] Jierun Chen, Dongting Hu, Xijie Huang, Huseyin Coskun, Arpit Sahni, Aarush Gupta, Anujraaj Goyal, Dishani Lahiri, Rajesh Singh, Yerlan Idelbayev, et al. Snapgen: Taming high-resolution text-to-image models for mobile devices with efficient architectures and training. In *CVPR*, 2025. 3

[12] 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. In *CVPR*, 2025. 3

[13] Mengzhao Chen, Mingbao Lin, Ke Li, Yunhang Shen, Yongjian Wu, Fei Chao, and Rongrong Ji. Cf-vit: A general coarse-to-fine method for vision transformer. In *AAAI*, 2023. 3

[14] 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 preprint arXiv:2406.01125*, 2024. 2

[15] Shoufa Chen, Chongjian Ge, Shilong Zhang, Peize Sun, and Ping Luo. Pixelflow: Pixel-space generative models with flow. *arXiv preprint arXiv:2504.07963*, 2025. 3

[16] Xinle Cheng, Zhuoming Chen, and Zhihao Jia. Cat pruning: Cluster-aware token pruning for text-to-image diffusion models. *arXiv preprint arXiv:2502.00433*, 2025. 2

[17] Rohan Choudhury, JungEun Kim, Jinhyung Park, Eunho Yang, László A Jeni, and Kris M Kitani. Accelerating vision transformers with adaptive patch sizes. *arXiv preprint arXiv:2510.18091*, 2025. 3

[18] Juncan Deng, Shuaiting Li, Zeyu Wang, Hong Gu, Kedong Xu, and Kejie Huang. Vq4dit: Efficient post-training vector quantization for diffusion transformers. In *AAAI*, 2025. 2, 3

[19] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In *NeurIPS*, 2023. 3

[20] Zhenyuan Dong and Sai Qian Zhang. Ditas: Quantizing diffusion transformers via enhanced activation smoothing. In *WACV*, 2025. 3

[21] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3

[22] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *ICLR*, 2024. 1

[23] Weichen Fan, Chenyang Si, Junhao Song, Zhenyu Yang, Yinan He, Long Zhuo, Ziqi Huang, Ziyue Dong, Jingwen He, Dongwei Pan, et al. Vchitect-2.0: Parallel transformer for scaling up video diffusion models. *arXiv preprint arXiv:2501.08453*, 2025. 6

[24] Zichen Fan, Steve Dai, Rangharajan Venkatesan, Dennis Sylvester, and Brucek Khailany. Sq-dm: Accelerating diffusion models with aggressive quantization and temporal sparsity. *arXiv preprint arXiv:2501.15448*, 2025. 3

[25] Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. In *NeurIPS*, 2023. 2

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

[27] Haipeng Fang, Sheng Tang, Juan Cao, Enshuo Zhang, Fan Tang, and Tong-Yee Lee. Attend to not attended: Structure-then-detail token merging for post-training dit acceleration. In *CVPR*, 2025. 2

[28] Weilun Feng, Chuanguang Yang, Zhulin An, Libo Huang, Boyu Diao, Fei Wang, and Yongjun Xu. Relational diffusion distillation for efficient image generation. In *ACM MM*, pages 205–213, 2024. 3

[29] Daniel Gallo Fernández, Răzvan-Andrei Matişan, Alejandro Monroy Muñoz, Ana-Maria Vasilcoiu, Janusz Partyka, Tin Hadži Veljković, and Metod Jazbec. Duodiff: Accelerating diffusion models with a dual-backbone approach. *arXiv preprint arXiv:2410.09633*, 2024. 2

[30] Kaifeng Gao, Jiaxin Shi, Hanwang Zhang, Chunping Wang, Jun Xiao, and Long Chen. Ca2-vdm: Efficient autoregres-

sive video diffusion model with causal generation and cache sharing. *arXiv preprint arXiv:2411.16375*, 2024. 2

[31] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Joshua M Susskind, and Navdeep Jaitly. Matryoshka diffusion models. In *ICLR*, 2023. 3

[32] Matthew Gwilliam, Han Cai, Di Wu, Abhinav Shrivastava, and Zhiyu Cheng. Accelerate high-quality diffusion models with inner loop feedback. *arXiv preprint arXiv:2501.13107*, 2025. 2

[33] 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

[34] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 2017. 7

[35] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[36] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 3

[37] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 3

[38] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *JMLR*, 2022. 3

[39] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022. 4, 6

[40] Taihang Hu, Linxuan Li, Joost van de Weijer, Hongcheng Gao, Fahad Shahbaz Khan, Jian Yang, Ming-Ming Cheng, Kai Wang, and Yaxing Wang. Token merging for training-free semantic binding in text-to-image synthesis. In *NeurIPS*, 2024. 2

[41] Youbing Hu, Yun Cheng, Anqi Lu, Zhiqiang Cao, Dawei Wei, Jie Liu, and Zhijun Li. Lf-vit: Reducing spatial redundancy in vision transformer for efficient image recognition. In *AAAI*, 2024. 3

[42] Yushi Huang, Zining Wang, Ruihao Gong, Jing Liu, Xinjie Zhang, Jinyang Guo, Xianglong Liu, and Jun Zhang. Harmonica: Harmonizing training and inference for better feature caching in diffusion transformer acceleration. *arXiv preprint arXiv:2410.01723*, 2024. 2

[43] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *CVPR*, 2024. 1, 7

[44] jackyhate. text-to-image-2m: A high-quality, diverse text-to-image training dataset. https://huggingface.co/datasets/jackyhate/text-to-image-2M, 2024. 6

[45] 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. *arXiv preprint arXiv:2410.05954*, 2024. 3

[46] Chen Ju, Haicheng Wang, Zeqian Li, Xu Chen, Zhonghua Zhai, Weilin Huang, and Shuai Xiao. Turbo: Informativity-driven acceleration plug-in for vision-language models. *arXiv preprint arXiv:2312.07408*, 2023. 2

[47] Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. In *ICCV*, 2025. 2

[48] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *CVPR*, 2023. 1

[49] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. Bk-sdm: A lightweight, fast, and cheap version of stable diffusion. In *ECCV*, 2024. 2, 3

[50] Dahye Kim, Xavier Thomas, and Deepti Ghadiyaram. Revelio: Interpreting and leveraging semantic information in diffusion models. In *ICCV*, 2025. 2, 4

[51] Geonung Kim, Beomsu Kim, Eunhyeok Park, and Sunghyun Cho. Diffusion model compression for image-to-image translation. In *ACCV*, pages 2105–2123, 2024. 2

[52] Minchul Kim, Shangqian Gao, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. Token fusion: Bridging the gap between token pruning and token merging. In *WACV*, 2024. 2

[53] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024. 1

[54] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024. 1, 2, 4, 6, 7

[55] Youngwan Lee, Kwanyong Park, Yoorhim Cho, Yong-Ju Lee, and Sung Ju Hwang. Koala: Empirical lessons toward memory-efficient and fast diffusion models for text-to-image synthesis. In *NeurIPS*, 2024. 2

[56] Alexander C Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *ICCV*, 2023. 2

[57] Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. Svdquant: Absorbing outliers by low-rank components for 4-bit diffusion models. *arXiv preprint arXiv:2411.05007*, 2024. 3

[58] Senmao Li, Taihang Hu, Fahad Shahbaz Khan, Linxuan Li, Shiqi Yang, Yaxing Wang, Ming-Ming Cheng, and Jian Yang. Faster diffusion: Rethinking the role of unet encoder in diffusion models. In *CoRR*, 2023. 2

[59] 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. In *NeurIPS*, 2023. 2, 3

[60] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 7

[61] 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. In *CVPR*, 2025. 2, 6, 7

[62] Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with taylorseers. *arXiv preprint arXiv:2503.06923*, 2025. 2, 6, 7

[63] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 2

[64] 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. 2

[65] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization (2017). *arXiv preprint arXiv:1711.05101*, 2017. 6

[66] 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. 2

[67] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In *NeurIPS*, 2022. 2

[68] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 2025. 2

[69] Wenbo Lu, Shaoyi Zheng, Yuxuan Xia, and Shengjie Wang. Toma: Token merge with attention for diffusion models. *arXiv preprint arXiv:2509.10918*, 2025. 2

[70] Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K Wong. Fastercache: Training-free video diffusion model acceleration with high quality. *arXiv preprint arXiv:2410.19355*, 2024. 2

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

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

[73] Shweta Mahajan, Tanzila Rahman, Kwang Moo Yi, and Leonid Sigal. Prompting hard or hardly prompting: Prompt inversion for text-to-image diffusion models. In *CVPR*, 2024. 2, 4, 5

[74] Marian Mazzone and Ahmed Elgammal. Art, creativity, and the potential of artificial intelligence. In *Arts*. MDPI, 2019. 2

[75] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *CVPR*, 2023. 2

[76] Konstantin Mishchenko and Aaron Defazio. Prodigy: An expeditiously adaptive parameter-free learner. *arXiv preprint arXiv:2306.06101*, 2023. 6

[77] Trong-Tung Nguyen, Quang Nguyen, Khoi Nguyen, Anh Tran, and Cuong Pham. Swiftedit: Lightning fast text-guided image editing via one-step diffusion. In *CVPR*, 2025. 2

[78] Geon Yeong Park, Sang Wan Lee, and Jong Chul Ye. Inference-time diffusion model distillation. In *ICCV*, 2025. 3

[79] Yong-Hyun Park, Chieh-Hsin Lai, Satoshi Hayakawa, Yuhta Takida, and Yuki Mitsufuji. *Jump Your Steps*: Optimizing sampling schedule of discrete diffusion models. *arXiv preprint arXiv:2410.07761*, 2024. 2

[80] Or Patashnik, Daniel Garibi, Idan Azuri, Hadar Averbuch-Elor, and Daniel Cohen-Or. Localizing object-level shape variations with text-to-image diffusion models. In *ICCV*, 2023. 2, 4, 5

[81] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023. 1, 3

[82] Pablo Pernias, Dominic Rampas, Mats L Richter, Christopher J Pal, and Marc Aubreville. Würstchen: An efficient architecture for large-scale text-to-image diffusion models. *arXiv preprint arXiv:2306.00637*, 2023. 3

[83] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 1, 7

[84] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 3

[85] Tomer Ronen, Omer Levy, and Avram Golbert. Vision transformers with mixed-resolution tokenization. In *CVPR*, 2023. 3

[86] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *CVPR*, 2023. 1

[87] Omid Saghatchian, Atiyeh Gh Moghadam, and Ahmad Nickabadi. Cached adaptive token merging: Dynamic token reduction and redundant computation elimination in diffusion model. *arXiv preprint arXiv:2501.00946*, 2025. 2

[88] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022. 3, 7

[89] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *TPAMI*, 2022. 3

[90] Ketan Suhaas Saichandran, Xavier Thomas, Prakhar Kaushik, and Deepti Ghadiyaram. Progressive prompt detailing for improved alignment in text-to-image generative models. *arXiv preprint arXiv:2503.17794*, 2025. 2, 5

[91] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 2, 3

[92] Team Seedream, Yunpeng Chen, Yu Gao, Lixue Gong, Meng Guo, Qiushan Guo, Zhiyao Guo, Xiaoxia Hou, Weilin Huang, Yixuan Huang, et al. Seedream 4.0: Toward next-generation multimodal image generation. *arXiv preprint arXiv:2509.20427*, 2025. 1

[93] Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. Fora: Fast-forward caching in diffusion transformer acceleration. *arXiv preprint arXiv:2407.01425*, 2024. 2

[94] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *CVPR*, 2023. 2, 3

[95] Jaskirat Singh, Lindsey Li, Weijia Shi, Ranjay Krishna, Yejin Choi, Pang Wei Koh, Michael F Cohen, Stephen Gould, Liang Zheng, and Luke Zettlemoyer. Negative token merging: Image-based adversarial feature guidance. *arXiv preprint arXiv:2412.01339*, 2024. 2

[96] Ethan Smith, Nayan Saxena, and Aninda Saha. Todo: Token downsampling for efficient generation of high-resolution images. *arXiv preprint arXiv:2402.13573*, 2024. 2

[97] Junhyuk So, Jungwon Lee, Daehyun Ahn, Hyungjun Kim, and Eunhyeok Park. Temporal dynamic quantization for diffusion models. In *NeurIPS*, 2023. 2, 3

[98] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2

[99] Nick Stracke, Stefan Andreas Baumann, Kolja Bauer, Frank Fundel, and Björn Ommer. Cleandift: Diffusion features without noise. In *CVPR*, 2025. 2, 4, 5

[100] Sitong Su, Jianzhi Liu, Lianli Gao, and Jingkuan Song. F^3-pruning: a training-free and generalized pruning strategy towards faster and finer text-to-video synthesis. In *AAAI*, 2024. 2

[101] Wenhao Sun, Rong-Cheng Tu, Jingyi Liao, Zhao Jin, and Dacheng Tao. Asymrnr: Video diffusion transformers acceleration with asymmetric reduction and restoration. *arXiv preprint arXiv:2412.11706*, 2024.

[102] Wenzhang Sun, Qirui Hou, Donglin Di, Jiahui Yang, Yongjia Ma, and Jianxun Cui. Unicp: A unified caching and pruning framework for efficient video generation. *arXiv preprint arXiv:2502.04393*, 2025. 2

[103] Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. In *NeurIPS*, 2023. 2, 4, 5

[104] Shilong Tian, Hong Chen, Chengtao Lv, Yu Liu, Jinyang Guo, Xianglong Liu, Shengxi Li, Hao Yang, and Tao Xie. Qvd: Post-training quantization for video diffusion models. In *ACM MM*, 2024. 2, 3

[105] Yuchuan Tian, Zhijun Tu, Hanting Chen, Jie Hu, Chao Xu, and Yunhe Wang. U-dits: Downsample tokens in u-shaped diffusion transformers. In *NeurIPS*, 2024. 2

[106] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3, 4

[107] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 1, 2, 6, 7

[108] Hongjie Wang, Difan Liu, Yan Kang, Yijun Li, Zhe Lin, Niraj K Jha, and Yuchen Liu. Attention-driven training-free efficiency enhancement of diffusion models. In *CVPR*, 2024. 2

[109] Kafeng Wang, Jianfei Chen, He Li, Zhenpeng Mi, and Jun Zhu. Sparsedm: Toward sparse efficient diffusion models. *arXiv preprint arXiv:2404.10445*, 2024. 2

[110] Luozhou Wang, Shuai Yang, Shu Liu, and Ying-cong Chen. Not all steps are created equal: Selective diffusion distillation for image manipulation. In *ICCV*, 2023. 2, 4, 5

[111] Yulin Wang, Rui Huang, Shiji Song, Zeyi Huang, and Gao Huang. Not all images are worth 16x16 words: Dynamic transformers for efficient image recognition. In *NeurIPS*, 2021. 3

[112] Yunke Wang, Bo Du, Wenyuan Wang, and Chang Xu. Multi-tailed vision transformer for efficient inference. *Neural Networks*, 2024. 3

[113] Yilin Wang, Haiyang Xu, Xiang Zhang, Zeyuan Chen, Zhizhou Sha, Zirui Wang, and Zhuowen Tu. Omnicontrolnet: Dual-stage integration for conditional image generation. In *CVPR*, 2024. 1

[114] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. In *IEEE TIP*, 2004. 7

[115] Zhendong Wang, Yifan Jiang, Huangjie Zheng, Peihao Wang, Pengcheng He, Zhangyang Wang, Weizhu Chen, Mingyuan Zhou, et al. Patch diffusion: Faster and more data-efficient training of diffusion models. In *NeurIPS*, 2023. 2

[116] Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, et al. Cache me if you can: Accelerating diffusion models through block caching. In *CVPR*, 2024. 2

[117] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025. 1

[118] 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. In *NeurIPS*, 2023. 1, 7

[119] Yu Xu, Fan Tang, Juan Cao, Yuxin Zhang, Xiaoyu Kong, Jintao Li, Oliver Deussen, and Tong-Yee Lee. Headrouter: A training-free image editing framework for mm-dits by adaptively routing attention heads. *arXiv preprint arXiv:2411.15034*, 2024. 2

[120] Shenghao Yang and Kimon Fountoulakis. Weighted flow diffusion for local graph clustering with node attributes: An algorithm and statistical guarantees. In *ICML*, 2023. 2

[121] Hancheng Ye, Jiakang Yuan, Renqiu Xia, Xiangchao Yan, Tao Chen, Junchi Yan, Botian Shi, and Bo Zhang. Training-free adaptive diffusion with bounded difference approximation strategy. In *NeurIPS*, 2024. 2, 5

[122] Haoran You, Connelly Barnes, Yuqian Zhou, Yan Kang, Zhenbang Du, Wei Zhou, Lingzhi Zhang, Yotam Nitzan, Xiaoyang Liu, Zhe Lin, et al. Layer-and timestep-adaptive differentiable token compression ratios for efficient diffusion transformers. In *CVPR*, 2025. 2

[123] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022. 7

[124] Zhihang Yuan, Yuzhang Shang, Hanling Zhang, Tongcheng Fang, Rui Xie, Bingxin Xu, Yan Yan, Shengen Yan, Guohao Dai, and Yu Wang. E-car: Efficient continuous autoregressive image generation via multistage modeling. *arXiv preprint arXiv:2412.14170*, 2024. 2

[125] Dingkun Zhang, Sijia Li, Chen Chen, Qingsong Xie, and Haonan Lu. Laptop-diff: Layer pruning and normalized distillation for compressing diffusion models. *arXiv preprint arXiv:2404.11098*, 2024. 2

[126] Evelyn Zhang, Bang Xiao, Jiayi Tang, Qianli Ma, Chang Zou, Xuefei Ning, Xuming Hu, and Linfeng Zhang. Token pruning for caching better: 9 times acceleration on stable diffusion for free. *arXiv preprint arXiv:2501.00375*, 2024. 2

[127] Hui Zhang, Zuxuan Wu, Zhen Xing, Jie Shao, and Yu-Gang Jiang. Adadiff: Adaptive step selection for fast diffusion. *arXiv preprint arXiv:2311.14768*, 2023. 2

[128] Hui Zhang, Tingwei Gao, Jie Shao, and Zuxuan Wu. Blockdance: Reuse structurally similar spatio-temporal features to accelerate diffusion transformers. In *CVPR*, 2025. 2

[129] Linfeng Zhang and Kaisheng Ma. Accelerating diffusion models with one-to-many knowledge distillation. *arXiv preprint arXiv:2410.04191*, 2024. 2, 3

[130] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 7

[131] Wangbo Zhao, Yizeng Han, Jiasheng Tang, Kai Wang, Yibing Song, Gao Huang, Fan Wang, and Yang You. Dynamic diffusion transformer. *arXiv preprint arXiv:2410.03456*, 2024. 2

[132] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Dpm-solver-v3: Improved diffusion ode solver with empirical model statistics. *NeurIPS*, 2023. 2

[133] Qiqi Zhou and Yichen Zhu. Make a long image short: Adaptive token length for vision transformers. In *ECML PKDD*, 2023. 3

[134] Haowei Zhu, Dehua Tang, Ji Liu, Mingjie Lu, Jintu Zheng, Jinzhang Peng, Dong Li, Yu Wang, Fan Jiang, Lu Tian, et al. Dip-go: A diffusion pruner via few-step gradient optimization. In *NeurIPS*, 2024. 2

[135] Yuanzhi Zhu, Hanshu Yan, Huan Yang, Kai Zhang, and Junnan Li. Accelerating video diffusion models via distribution matching. *arXiv preprint arXiv:2412.05899*, 2024. 3

[136] Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. *arXiv preprint arXiv:2410.05317*, 2024. 2