OmniCache: A Trajectory-Oriented Global Perspective on Training-Free Cache Reuse for Diffusion Transformer Models

Huanpeng Chu*
Zhipu AI
chuhp@zju.edu.cn

Wei Wu*
Nanjing University
714031909@qq.com

Guanyu Feng†
Zhipu AI
guanyu.feng@zhipuai.cn

Yutao Zhang
Zhipu AI
z_zhangyt@163.com

Abstract

Diffusion models have emerged as a powerful paradigm for generative tasks such as image synthesis and video generation, with Transformer architectures further enhancing performance. However, the high computational cost of diffusion Transformers—stemming from a large number of sampling steps and complex per-step computations—presents significant challenges for real-time deployment. In this paper, we introduce OmniCache, a training-free acceleration method that exploits the global redundancy inherent in the denoising process.

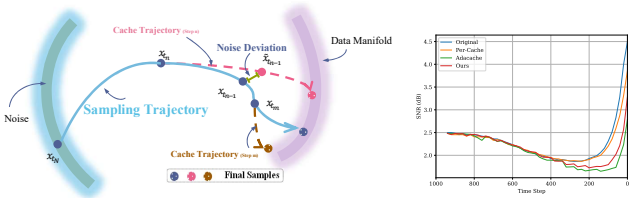
Unlike existing methods that determine caching strategies based on inter-step similarities and tend to prioritize reusing later sampling steps, our approach originates from the sampling perspective of DIT models. We systematically analyze the model's sampling trajectories and strategically distribute cache reuse across the entire sampling process. This global perspective enables more effective utilization of cached computations throughout the diffusion trajectory, rather than concentrating reuse within limited segments of the sampling procedure. In addition, during cache reuse, we dynamically estimate the corresponding noise and filter it out to reduce its impact on the sampling direction.

Extensive experiments demonstrate that our approach accelerates the sampling process while maintaining competitive generative quality, offering a promising and practical solution for efficient deployment of diffusion-based generative models.

1. Introduction

In recent years, diffusion models [15, 43, 44] have achieved remarkable performance as powerful image generation models [10, 38, 49]. Among the various backbone architectures for diffusion models, the Transformer [48] has demonstrated significant competitiveness, not only generating high-fidelity images [2, 37], but also excelling in video generation [3, 8, 28], text-to-speech synthesis [18, 26, 46], and 3D generation [4, 35]. However, while diffusion Transformers benefit from the outstanding scalability of the Transformer architecture, they also bring substantial efficiency challenges, including high deployment costs and slow inference speeds. As sampling costs scale proportionally with both the number of time steps and the size of the model at each step, current approaches to improving sampling efficiency naturally split into two directions: reducing the number of sampling steps [15, 20, 24, 42] or lowering the per-step inference cost [11, 55].

Methods for reducing the number of time steps include distilling the trajectory to fewer steps [30, 39, 45] and discretizing the reverse-time stochastic differential equation (SDE) or the ordinary differential equation (ODE) of probability flows [27, 42, 57]. On the other hand, the second approach primarily focuses on model compression [20, 24], such as using model quantization to transform the denoising



(a) Sampling (b) SNR Variation

Figure 1. a). The geometric sampling model during diffusion illustrates that every initial sample (drawn from the noise distribution) begins from a large sphere and converges to the final sample along a regular sampling trajectory. We perform cache reuse at different time steps x_{t_n} and x_{t_m}. We see early cache reuse allows subsequent sampling steps to filter introduced noise, correcting the trajectory. In contrast, late-stage cache reuse produces irrecoverable trajectory deviations. b). Real SNR curve during sampling phases under cache reuse conditions. Since model outputs show greater similarity in later sampling stages yet exhibit weaker denoising strength, cache reuse in later sampling steps leads to continuous SNR degradation.

*Both authors contributed equally to this research.
†Corresponding author.

network into lower-bit representations [13, 41].

A novel method for dynamic inference in diffusion models employs special caching mechanisms during the denoising process[19, 32, 33, 53, 59]. The starting point of these methods lies in the redundancy introduced by the diffusion model during sampling, which leads to high similarity between adjacent time steps. As a result, the computation for certain units can be reused in subsequent steps. However, as the noise is progressively filtered out, the more consistent inputs lead to higher similarity in denoising outputs. Therefore, existing caching strategies are more focused on later stages of the sampling process. However, output similarity is not an ideal metric for evaluating cache reuse, especially for current video generation models based on DIT.

Fig. 1 provides a visual illustration and experimental validation of our approach. As shown in Fig. 1a, when cache reuse is implemented during early sampling stages, subsequent sampling steps naturally filter out the cache-induced noise—a fundamental characteristic of diffusion models. This allows the overall sampling trajectory to be gradually corrected toward the proper direction. In contrast, concentrating cache reuse in later steps leaves the model with insufficient capacity to remediate the introduced noise, resulting in irrecoverable trajectory deviations. Fig. 1b further substantiates our hypothesis, given that model outputs exhibit higher similarity during later sampling stages, which corresponds to diminished denoising strength. Consequently, noise introduced through cache reuse during these later phases cannot be effectively eliminated, leading to a progressive deterioration in the signal-to-noise ratio. This empirical evidence strongly supports our approach of distributed cache reuse throughout the entire sampling process rather than focusing exclusively on later steps.

To address these issues, we propose OmniCache, a trajectory-oriented global perspective on training-free cache reuse for Diffusion Transformer Models. First, we simplify and estimate the sampling trajectories of various diffusion models, revealing a pattern that is independent of the input sample. Leveraging the inherent regularity of these trajectory shapes, we introduce a curvature-based approach to determine the optimal nodes for cache reuse. Additionally, we estimate and correct the noise induced by cache reuse. Considering that diffusion models primarily rely on low-frequency signals during the early stages of sampling to establish the overall structure and basic semantics, and incorporate high-frequency signals in the later stages to capture finer details, we apply either high-pass or low-pass filtering to the estimated noise, thereby preventing unnecessary alterations to the hidden states across different sampling stages. Our method achieves a 2–2.5× speedup on models such as Opensora and Latte with virtually no loss in performance metrics. Moreover, on more challenging or less redundant distilled models (e.g., CogVideoX-5b-I2V-distill [56], 16steps), our approach yields a 1.45× speedup without performance degradation, whereas existing methods tend to lead to model collapse.

The contributions of our paper are as follows:

- We analyzed the influence of cache reuse on the sampling process of DIT models and pioneered a novel cache reuse methodology based on model sampling trajectories. To the best of our knowledge, this work represents the first systematic investigation that optimizes cache reuse strategies by explicitly considering the inherent characteristics of the diffusion sampling path.
- We discovered that noise introduced by cache reuse in the current step exhibits remarkably high correlation with the previous step. By leveraging this characteristic, we further estimate and filter out cache-induced nosie. In addition, we incorporate both high-pass and low-pass filtering on estimated noise artifacts to mitigate the impact on irrelevant signals.
- Our experimental results demonstrate that our approach delivers a 2–2.5× speedup on models such as Opensora and Latte without any performance degradation. Furthermore, we are able to further exploit the potential of cache reuse, achieving a 1.45× speedup on distilled models with extremely low redundancy.

2. Related Work

2.1. Efficient diffusion models

Diffusion models have broken the long-time dominance of generative adversarial networks (GANs [12]) in the challenging task of image synthesis. However, their extremely slow generation speed and large memory footprint constrain their widespread use for downstream tasks.

A great deal of recent work has focused on speeding up the sampling process while also improving the quality of the resulting samples. These efficient sampling methods can be classified into two main categories [47, 52]: Efficient Process Strategies (EPS), which suggest ways to improve the efficiency of diffusion models or speed up the sampling process, and Efficient Design Strategies (EDS), which recommend modifications to the design of baseline diffusion models. Konwledge distillation [16, 29], analytical trajectory estimation [1], early-stop strategy [31], and differential equation (DE) [44] are designed as EPS to decrease the discretization step size. For EDS, SnapFusion [24] obtains an efficient UNet structure by network pruning, which accelerates the inference process of the diffusion model. The model class Latent Diffusion Models (LDMs) [38] provide efficient image generation from the latent space with a single network pass. Model acceleration techniques—such as model quantization [9] and sparse attention [54]—have also been incorporated into diffusion models.

However, the cache reuse method lies somewhere in be-

tween these two approaches. It leverages the similarity between consecutive steps in the diffusion process by storing intermediate results from the denoising network modules in cache to reduce computation.

2.2. Cache Mechanisms

In large language models and multimodal models, KV caching is widely used [21, 36]. This method caches the key and value components from the attention mechanism so that they can be reused in subsequent attention computations, thereby accelerating inference. Additionally, prefix caching stores computed results for a fixed prefix, which can then be directly reused during generation without redundant calculations.

There are also several cache-based approaches in diffusion model generation. For example, Deepcache [32] expedites computation across adjacent steps in UNet networks by caching the output feature maps of upsampling blocks. Δ-DIT [7] stores the residuals between several blocks in the diffusion model to accelerate computation. AdaCache [19] dynamically adjusts the intensity of cache reuse for each video being generated, while TOCA [50] applies cache reuse along the token dimension in DIT. Furthermore, TGATE [6] accelerates the cross-attention module in the later stages of sampling by caching output feature maps.

However, most existing methods are based on the similarity between adjacent steps and lack a comprehensive study on the impact of cache reuse on the sampling process. Our approach analyzes the effects of cache reuse based on the sampling trajectory and correct the noise induced by cache reuse in sampling stage.

3. Method

In Section 3.1, we introduce the principles of the diffusion model's sampling process as well as the logic behind cache reuse. In Section 3.2, we model the sampling trajectory, analyze the impact of cache reuse on the trajectory, and propose a curvature-based cache reuse strategy. In Section 3.3, we decouple the noise induced by cache reuse and present a noise correction module. Finally, in Section 3.4, we provide a detailed description of our OmniCache method, covering both calibration stage and sampling stage.

3.1. Preliminaries

Diffusion models are a family of probabilistic generative models that progressively destruct real data by injecting noise, then learn to reverse this process for generation, represented notably by denosing diffusion probabilistic models (DDPMs [15]). DDPM is composed of two chains: a forward chain that perturbs data to noise, and a reverse chain that converts noise back to data. The former is usually designed by hand and its goal is to convert any data distribution into a simple prior distribution (e.g., a standard Gaussian distribution).

In the denoising process, with a Gaussian noise x_T, the diffusion model can generate samples by iterative sampling x_{t-1} from $p_\theta(x_{t-1}|x_t)$ until obtaining x_0, where the Gaussian distribution $p_\theta(x_{t-1}|x_t)$ is a simulation of the unavailable real distribution $q(x_{t-1}|x_t)$. The mean value $\mu_\theta(x_{t-1}|x_t)$ of $p_\theta(x_{t-1}|x_t)$ is calculated from the noise prediction network ϵ_θ (usually the UNet model or DiT model). Therefore, the sampling process of x_{t-1} is shown as follows:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha_t}}}\epsilon_\theta(x_t, t)\right) + \sigma_t z, \quad z \in N(0, I) \tag{1}$$

The diffusion model continuously evokes the noise prediction network to acquire the noise $\epsilon_\theta(x_t, t)$ and filter it out. The complexity of the noise prediction network ϵ_θ and the huge hidden states especially for video-generation make the sampling of diffusion models expensive.

The noise prediction network ϵ_θ, taking the current mainstream video generation model architecture DIT (Latent Diffusion Transformers) as an example, is composed of multiple layers stacked together (for instance, Latte-1 consists of 48 layers with identical structures). Each layer comprises an attention module (Atten) and a linear layer (MLP) arranged in a residual format. The essence of cache reuse is to store the output of the previous step for use in the current step (using the attention module as an example):

$$\widetilde{f}_t^l = \begin{cases} f_t^l + Atten(f_t^l) & OriginalForward \\ f_t^l + AttenCache & CacheReuse \end{cases} \tag{2}$$

For the input feature f_t^l at the current step, the cache reuse strategy employs the previously stored $AttenCache$ as the output of the attention module, thereby saving the corresponding computation. Typically, $AttenCache$ is obtained by storing the output of the attention module from the previous step $t-1$ or even earlier steps. The discrepancy between $AttenCache$ and the real output $Atten(f_t^l)$ accumulates through subsequent layers, ultimately introducing unavoidable noise in the model's final output.

$$\begin{aligned} \widetilde{x}_{t-1} &= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha_t}}}\widetilde{\epsilon}_\theta(x_t, t)\right) + \sigma_t z, \quad z \in N(0, I) \\ &= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha_t}}}\big(\epsilon_\theta(x_t, t) + q_\theta(x_t, t)\big)\right) + \sigma_t z. \end{aligned} \tag{3}$$

From the perspective of complete sampling process, these cache noises accumulate to form cache noise $q_\theta(\widetilde{x}_t, t)$ which further accumulates in the current output $\widetilde{x}_{t-1}$, thus affecting subsequent sampling processes.

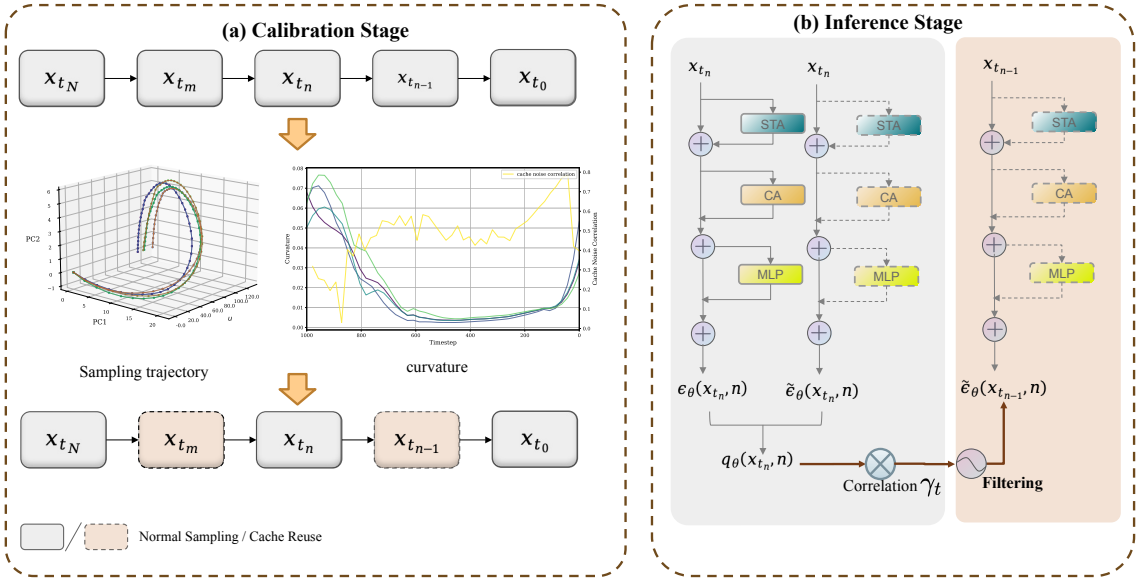


Figure 2. The diagram of our OmniCache. In the calibration stage, we store the states at different time steps x_{t_n} to obtain the corresponding sampling trajectory, while simultaneously computing the curvature and the correlation of cache-induced noise. Based on the curvature, we determine the appropriate time steps for cache reuse. In the actual sampling stage, we estimate the noise induced by cache reuse in real time, and correct the denoising model's output based on the noise correlation and high-pass/low-pass filtering.

3.2. Sampling Trajectory of Diffusion Models

Diffusion models generate samples that conform to the true data distribution by injecting noise into the data and subsequently removing it via a reverse process. This process is typically implemented by solving the reverse diffusion equation (e.g., PF-ODE). During the sampling phase, the process of generating a sample from the noise space can be regarded as a trajectory between the noise distribution and the data distribution, i.e., the sampling trajectory. Closely related to this is the denoising trajectory, which is produced at each denoising step and implicitly influences the curvature and shape of the sampling trajectory. Specifically, the denoising trajectory determines the rotation direction at each sampling point, and this rotational information reflects the curvature of the sampling trajectory.

Existing methods [5] provide an approach to approximate the sampling trajectory by confidently using a 3-D subspace. First, we compute the unit vector between the starting and ending points:

$$u = \frac{x_{t_N} - x_{t_0}}{\|x_{t_N} - x_{t_0}\|} \tag{4}$$

Next, we project the trajectory onto the orthogonal complement of u. We perform PCA on this projection and extract two principal directions, w_1 and w_2. Each sampling point is then approximated in the 3-D subspace spanned by u, w_1, and w_2:

$$x_{t_i}^{\text{proj}} = (x_{t_i} \cdot u)u + (x_{t_i} \cdot w_1)w_1 + (x_{t_i} \cdot w_2)w_2. \tag{5}$$

These two additional principal components allow us to more accurately approximate the actual trajectory, with deviations from the true trajectory being almost indistinguishable. This enables a better understanding of the geometric structure of high-dimensional sampling trajectories.

3.2.1. Sampling Trajectories Exhibit a Boomerang Shape

1. As shown in Fig. 3, the scale along the $x_{t_N} - x_{t_0}$ axis is significantly larger than that of the other two principal components. Consequently, the trajectory is very close to the straight line connecting its endpoints. 2. Trajectories from different initial starting points all exhibit a consistent boomerang shape, demonstrating strong regularity that is independent of the specific generated content and inherent solely to the diffusion model itself. 3. The spacing between adjacent steps along the sampling trajectory gradually decreases. The intervals between trajectory points essentially reflect the differences between adjacent steps. Therefore, similarity-based cache reuse methods tend to occur more frequently in the later stages of sampling.

3.2.2. Impact of Cache Reuse on Sampling Trajectory

In order to more intuitively demonstrate the impact of cache reuse on the sampling trajectory, we conducted an analysis on the CogVideoX-5b-I2V-distill model. Since CogVideoX-5b-I2V-distill utilizes only 16 sampling steps, the effects of cache reuse become even more pronounced. In Fig. 3a, we applied cache reuse during an early stage of sampling ($x_{t_{12}} \rightarrow x_{t_{11}}$). At this point, it is evident that cache reuse itself introduces additional noise, which mis-

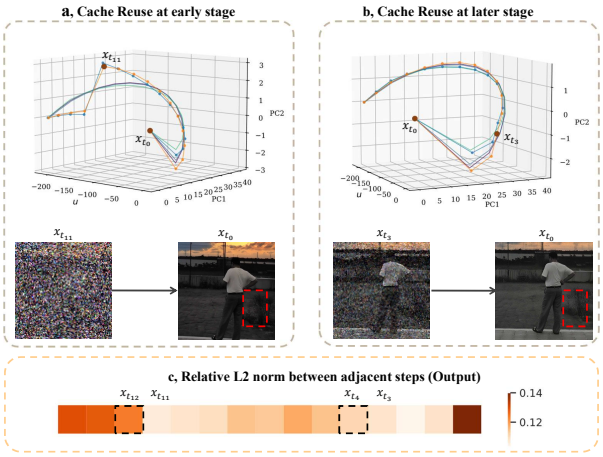


Figure 3. We visualized the sampling trajectories of the distilled version of CogVideoX-5b-I2V-distill [56] (a total of 16 steps). The unmarked trajectory in a and b represents the normal sampling process. Additionally, we applied cache reuse at one step in the early stage (left panel) and one step in the later stage (right panel), and visualized the corresponding intermediate outputs along with the final output results. In c, we visualized the relative L2 error between the output noise at each step and that of the previous step.

leads the sampling direction of the diffusion model. However, the diffusion model has an inherent self-correction capability. During the early sampling phase, when cache reuse is applied, subsequent sampling steps treat the noise introduced as part of the denoising process, thereby pulling the sampling direction back on track. As shown in Fig. 3a, the hidden states $x_{t_{11}}$ are dominated by noise, so the additional noise introduced can be effectively ignored. In contrast, during the later sampling stage ($x_{t_4} \rightarrow x_{t_3}$), the noise induced by cache reuse is much harder to disregard, and there are not enough subsequent sampling steps to correct it, making the change in sampling direction irreversible.

Furthermore, we visualized the time steps at which cache reuse was applied along with the final generated outputs. It can be observed that when cache reuse is applied at x_{t_4}, the generated results exhibit **ripple-like noise regions** (highlighted by red boxes), leading to poorer quality compared to applying cache reuse at $x_{t_{12}}$. This phenomenon is consistent with the SNR plot in Fig. 1b: introducing additional noise into the noise-dominated hidden states has little effect during the early sampling stages, whereas the impact is pronounced in the later stages.

However, as shown in Fig. 3c, the relative L2 norm of the output noise at x_{t_4} is lower than that at $x_{t_{12}}$. Therefore, similarity-based cache reuse strategies tend to select x_{t_4} for cache reuse (as in AdaCache). Consequently, the locally based similarity metric of output features diverges from the truly optimal selection strategy. We, instead, explore a more comprehensive cache reuse strategy based on the sampling trajectory.

3.2.3. Trajectory Curvature-Based Cache Reuse

Our analysis reveals that sampling steps with higher similarity predominantly emerge in later stages of the sampling process, precisely when cache reuse introduces irreversible perturbations to the sampling trajectory. Based on this insight, we propose a novel caching strategy grounded in the global properties of sampling trajectories. We identify that trajectory curvature serves as an effective indicator for optimal cache deployment: regions of minimal curvature signify stable directional progression, making these ideal candidates for cache reuse without compromising generation quality. Conversely, segments exhibiting pronounced curvature represent critical directional transitions where standard sampling processes should be preserved. Notably, our comprehensive experiments demonstrate remarkable structural consistency across diverse sampling trajectories, with nearly identical geometric patterns emerging regardless of generation contents. This consistency enables our curvature-based caching methodology to generalize effectively across various video generation tasks.

3.3. Cache-induced Noise Correction

The noise induced by a single step of cache reuse in Eqn. 3 can be computed as follows:

$$q_\theta(x_t, t) = \widetilde{\epsilon}_\theta(x_t, t) - \epsilon_\theta(x_t, t) \tag{6}$$

When we perform a Taylor expansion on cache-induced noise $q_\theta(x_t, t)$, we obtain the following formula:

$$\begin{aligned} q_\theta(x_{t-1}, t-1) &\approx q_\theta(x_t, t) - \frac{d\, q_\theta(x_t, t)}{dt} + \mathcal{O}(\Delta t^2) \\ &\approx \gamma_{t-1}\, q_\theta(x_t, t), \end{aligned} \tag{7}$$

where γ_{t-1} is the correlation of noise.

This indicates that when we perform normal forward sampling in the preceding step, we can simultaneously perform cache reuse to obtain the corresponding cache reuse noise $q_\theta(x_t, t)$ for estimating $q_\theta(x_{t-1}, t-1)$. In order to estimate the noise induced by cache reuse more accurately, we need to minimize the second term in Eqn. 7. When we further decouple the second term, we obtain:

$$\begin{aligned} -\frac{d\, q_\theta(x_t, t)}{dt} &= \frac{d\, \epsilon_\theta(x_t, t)}{dt} - \frac{d\, \widetilde{\epsilon}_\theta(x_t, t)}{dt} \\ &= \frac{\sqrt{\alpha_t - \alpha_t \overline{\alpha_t}}}{\beta_t} \left(\frac{d\, \widetilde{x}_{t-1}}{dt} - \frac{d\, x_{t-1}}{dt} \right) \\ &\approx \frac{\sqrt{\alpha_t - \alpha_t \overline{\alpha_t}}}{\beta_t} \left(\frac{d\, x_t}{dt} - \frac{d\, x_{t-1}}{dt} \right) \\ &\approx \frac{\sqrt{\alpha_t - \alpha_t \overline{\alpha_t}}}{\beta_t} \frac{d^2 x_t}{dt^2} \end{aligned} \tag{8}$$

The first derivative of $q_\theta(x_t, t)$ is correlated to the second derivative of the sampling trajectory. That is, when we select a time step with minimal curvature for cache reuse, the

first derivative of the cache noise is minimized, and at this point the actual cache noise $q_\theta(x_{t-1}, t-1)$ is most strongly correlated with the cache noise $q_\theta(x_t, t)$ from the previous step. As shown in Fig. 2, we visualize the variation of curvature and noise correlation γ_t over time steps in the Latte model. It can be observed that curvature and γ_t are negatively correlated, which is consistent with our previous predictions. When the curvature is low, the sampling trajectory tends to continue along the previous direction, resulting in a stronger correlation in the noise introduced by cache reuse.

Therefore, we can use the estimated cache noise $\gamma_{t-1}\ q_\theta(x_t, t)$ to correct the current step. Besides, our trajectory curvature-based cache reuse and the correction of cache-induced noise are inherently coupled and mutually reinforcing.

Furthermore, for diffusion models with high redundancy (for example, 50-step diffusion models such as Latte), there can be consecutive time steps where cache reuse is applied. In this case, the estimated noise can be approximated as the product of multiple correlation coefficients:

$$\begin{aligned} q_\theta(x_{t-1}, t-1) &\approx \gamma_{t-1}\ q_\theta(x_t, t) \\ &\approx \gamma_{t-1} * \gamma_t\ q_\theta(x_{t+1}, t+1) \end{aligned} \tag{9}$$

Therefore, to ensure the effectiveness of the noise correction module, we impose an additional constraint that prohibits performing cache reuse for three consecutive steps.

At the same time, considering that diffusion models process different signals at different sampling stages—focusing on global information and low-frequency signals in the early stages, and on fine details and high-frequency signals in the later stages. We apply low-pass filtering to the estimated noise during the early sampling stages to prevent it from introducing additional noise into the high-frequency components. Conversely, in the later stages, we apply high-pass filtering to prevent low-frequency noise from degrading the underlying information.

3.4. Summary of Methods

As shown in the Fig. 2, our method consists of the following steps:

- During the **calibration stage**, we obtain the input state x_t of diffusion model at each step, output $\epsilon_\theta(x_t, t)$ by the denoising model. Then, we reduce the sampling trajectory to three dimensions and calculate the curvature at each step. Besides, we must ensure that no three consecutive time steps are selected for cache reuse, so as to guarantee the effectiveness of subsequent noise correction.
- During the **actual sampling stage**, when a predetermined cache reuse time step is reached, we replace the original computed results with the cached attention outputs and MLP outputs, thereby saving computational cost. Based on the noise correlation scale γ_t, we estimate the cache-induced noise, then apply either high-pass or low-pass filtering to correct the output of the denoising model.

4. Experiment

4.1. Implementation details

Model Configurations We conducted experiments across four widely-used DIT models for various generation tasks. In the image generation domain, DIT-XL [2] was employed for class-conditional image generation. For video generation scenarios, we implemented OpenSora [61] (480p -2s at 30steps) and Latte [34] (512×512 -2s at 50steps) for text-to-video generation and CogVideoX-5b-I2V-distill [56] for image-to-video conversion. All experiments were performed on NVIDIA 4090 GPU hardware.

Evaluation Metrics For class-conditional image generation, we uniformly sampled 1000 categories from ImageNet and generated 50000 samples at a resolution of 256 × 256 pixels. We then evaluated their FID [14] and Precision and Recall metrics [22]. For text-to-video generation, we test our approach based on the VBench framework [17], generating 5 videos for each of the 950 benchmark prompts under different random seeds, resulting in a total of 4,750 videos. The generated videos are comprehensively evaluated across 16 aspects proposed in VBench. Besides, we also report reference-based PSNR, SSIM [51] and LPIPS [58] as quality metrics. For the image-to-video generation task, we randomly generated 100 videos and evaluated them using metrics such as Q-Align [23]. In addition, we conducted human evaluations to analyze and compare the videos across two major dimensions: consistency and video quality.

Comparison Targets For benchmarking purposes, given that our proposed method is a training-free inference acceleration strategy, we selected several comparable approaches as baselines. The comparison methods include FORA [40], Δ-DiT [7], T-GATE [59], PAB [60], AdaCache [19], TeaCache [25] and ToCA [50]. These methods represent the current state-of-the-art techniques for accelerating diffusion model inference without requiring additional training.

4.2. Performance On Video Generation

Text-to-video Generation Quantitative comparison between our OmniCache with other training-free DiT acceleration methods is shown in Tab. 1. In the table, OmniCache-slow and OmniCache-fast perform cache reuse on 50% and 60% of the time steps, respectively. For the 30-step Latte model, OmniCache-slow and OmniCache-fast save 15 and 18 steps of computation, respectively. Specifically, our OmniCache-slow achieves a 2× speedup with only a 0.16% drop in the VBench metric compared to the baseline, outperforming other training-free, cache-based DiT acceleration methods. Moreover, OmniCache-fast further accelerates the process, striking an excellent balance between per-

Method	VBench (%) ↑	PSNR ↑	LPIPS ↓	SSIM ↑	FLOPs (T)	Speedup
Open-Sora	79.22	–	–	–	3230.24	1.00×
+ Δ-DiT	78.21	11.91	0.5692	0.4811	3166.47	–
+ T-GATE	77.61	15.50	0.3495	0.6760	2818.40	1.10×
+ PAB-fast	76.95	23.58	0.1743	0.8220	2558.25	1.34×
+ PAB-slow	78.51	27.04	0.0925	0.8847	2657.70	1.20×
+ ToCa(R = 85%)	78.34	–	–	–	1394.03	2.36×
+ TeaCache-fast	78.48	19.10	0.2511	0.8415	1640.00	2.25×
+ OmniCache-slow	78.83	22.37	0.1553	0.8180	1615.12	2.00×
+ OmniCache-fast	78.50	21.27	0.1841	0.7930	1292.10	2.50×
Latte	77.40	–	–	–	3439.47	1.00×
+ Δ-DiT	52.00	8.65	0.8513	0.1078	3437.33	–
+ T-GATE	75.42	19.55	0.2612	0.6927	3059.02	1.11×
+ PAB-fast	73.13	17.16	0.3903	0.6421	2576.77	1.33×
+ PAB-slow	76.32	19.71	0.2699	0.7014	2767.22	1.24×
+ AdaCache-fast	76.26	17.70	0.3522	0.6659	1010.33	2.74×
+ AdaCache-fast (w/ MoReg)	76.47	18.16	0.3222	0.6832	1187.31	2.46×
+ AdaCache-slow	77.07	22.78	0.1737	0.8030	2023.65	1.59×
+ TeaCache-fast	76.69	18.62	0.3133	0.6678	1120.00	3.28 ×
+ OmniCache-slow	77.24	22.48	0.1955	0.7903	1719.74	2.00×
+ OmniCache-fast	77.09	21.06	0.2463	0.7575	1375.79	2.50×

Table 1. Quantitative evaluation of text-to-video generation quality. Here, we compare our method, OmniCache, with several training-free, cache-based DiT acceleration approaches on multiple video baselines. It can be observed that our OmniCache-fast scheme strikes a favorable balance between the acceleration ratio and actual performance metrics, incurring only minimal performance degradation with a 2.5× speedup. Corresponding visualizations can be found in Appendix.

Figure 4. The image-to-video generation results on the CogVideoX-5b-I2V-distill model are shown. The generated video comprises 80 frames, and we extracted the 10th, 40th, and 80th frames for visualization.

Method	Aesthetic Quality ↑	Q-Align↑	Speedup
CogVideoX-5b-I2V-distill	0.59	0.79	1.00×
+AdaCache*	0.57	0.77	1.33×
+OmniCache	0.621	0.792	1.45×

Table 2. Quantitative evaluation of image-to-video generation.

formance and acceleration—with a 2.5× speedup and only a 0.31% reduction in VBench.

Image-to-video Generation We generated 100 videos using the CogVideoX-5b-I2V-distill model to evaluate the effectiveness of our method. This model is a distilled version with 16 sampling steps, and the resolution of the generated videos is consistent with that of the input images, reaching up to 1080P (768×1360). Due to extensive reduction in redundancy through model distillation, existing cache reuse methods such as Δ-DiT, learn to cache, and PAB fail to generate proper videos, often resulting in model collapse. Shown in Tab. 2, we adapted AdaCache for this model, which performs cache reuse on an average of 4 sampling steps, leading to a 2% decrease in the Q-Align metric. In contrast, our OmniCache, which accelerates 5 steps (an ac-

Method	FLOPs(T)	Speed	sFID↓	FID↓	Precision↑	Recall↑
DiT-XL/2-G (cfg = 1.50)	118.68	1.00×	4.98	2.31	0.82	0.58
33% steps	39.40	3.01×	6.31	2.76	0.81	0.57
37% steps	44.15	2.69×	6.04	2.64	0.81	0.58
+FORA (N = 3)	39.95	2.97×	6.21	2.80	0.80	0.59
+FORA (N = 2.8)	43.36	2.74×	6.13	2.80	0.80	0.59
+ToCa (N = 4, R = 93%)	43.22	2.75×	5.81	2.68	0.80	0.57
+OmniCache-fast	47.47	2.50×	5.86	2.69	0.81	0.56

Table 3. Quantitative comparison on class-to-image generation on ImageNet with DiT-XL/2.

Method	Aesthetic Quality↑	Q-Align↑
CogVideoX-5b-I2V-distill	0.59	0.79
+ OmniCache (Cache Reuse)	0.58	0.778
+ OmniCache (Cache Reuse + Noise Correct)	0.593	0.788
+ OmniCache (Cache Reuse + Noise Correct/Filtering)	0.621	0.792

Table 4. The effect of different modules of OmniCache with CogVideoX-5b-I2V-distill.

celeration ratio of 1.45), is able to maintain a stable qalign metric. In addition, we conducted human evaluations to compare the baseline and our OmniCache. The win rates for visual quality and textual instruction compliance were **5%** and **-3%**, respectively. This indicates that our method produces results that are nearly identical to the original model, with almost no degradation in generation quality at an acceleration ratio of 1.45.

We visualized the intermediate frames of the generated videos in Fig. 4. It can be observed that the similarity-based AdaCache introduces additional noise, which cannot be filtered out in subsequent sampling steps, resulting in some blurry regions that degrade the video quality. In contrast, although the final generated videos from our method differ significantly from the original, the overall image quality remains much closer to the original—almost **lossless**.

4.3. Performance on Image Generation

Class-Conditional Image Generation Quantitative comparison between OmniCache with other training-free DiT acceleration methods is shown in Tab. 3. DiT-XL/2, based on a DDPM sampler, generates images in 250 steps. Due to the extremely high similarity between adjacent steps, the magnitude of noise induced by cache reuse is lower compared to that in the video generation task, making it challenging for our noise correct module. Nevertheless, our method still achieves results comparable to those of the contemporaneous SOTA method, ToCa.

4.4. Ablation Study

Our method, OmniCache uses the sampling trajectory to determine the cache reuse time steps (CacheReuse). And noise correction and filtering module (Noise Correct/Filtering) occurs during sampling stage. The methods listed in Tab. 4 are consistent with those in Tab. 2 , all methods perform cache reuse on 5 sampling steps, corresponding to an acceleration ratio of 1.45. It can be observed that our noise correction module is capable of rectifying cache-induced noise, thereby improving the quality of generated videos. Moreover, when we further apply high-pass and low-pass filtering to the estimated noise to focus on specific information (Filtering), the corresponding video metrics are further enhanced, resulting in nearly lossless performance compared to the original model.

5. Conclusion and Limitation

Unlike previous cache reuse methods based on similarity or other local features, our OmniCache provides an novel perspective on diffusion model acceleration through two key innovations. First, we analyze sampling trajectory geometry to determine optimal cache reuse opportunities based on trajectory curvature patterns, enabling informed decisions about when cache reuse will least impact generation quality. Second, we address the noise introduced during cache reuse through dynamic estimation of its correlation with previous sampling steps, while simultaneously applying high-pass and low-pass filtering at different sampling stages to effectively remove interfering signals while preserving essential components. Our method proves highly effective in video generation tasks with relatively few sampling steps (30–50 steps), achieving nearly lossless performance with a 2× acceleration. Furthermore, for distilled models with low redundancy where current cache reuse methods almost fail, our approach still delivers video quality that is nearly lossless compared to the original model at an acceleration ratio of 1.45.

OmniCache's main limitation is that cache reuse cannot be applied for three consecutive steps. We introduced this constraint to ensure the reliability of our cache-induced noise estimation, which may limit its maximum acceleration capability.

5.1. Acknowledgment

OmniCache was carried out at Zhipu AI. We gratefully acknowledge the CogVideo team and other collaborating groups for their generous support and assistance during research and development. We also thank the inference infrastructure team for the insightful technical discussions that helped shape this project.

References

[1] Fan Bao, Chongxuan Li, Jiacheng Sun, Jun Zhu, and Bo Zhang. Estimating the optimal covariance with imperfect mean in diffusion probabilistic models. *arXiv preprint arXiv:2206.07309*, 2022. 2

[2] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22669–22679, 2023. 1, 6

[3] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. *Unpublished*, 2024. 1

[4] Ziang Cao, Fangzhou Hong, Tong Wu, Liang Pan, and Ziwei Liu. Large-vocabulary 3d diffusion model with transformer. *arXiv preprint arXiv:2309.07920*, 2023. 1

[5] Defang Chen, Zhenyu Zhou, Can Wang, Chunhua Shen, and Siwei Lyu. On the trajectory regularity of ode-based diffusion sampling. In *International Conference on Machine Learning*, pages 7905–7934. PMLR, 2024. 4

[6] Michael Chen, Ning Zhao, and Wei Li. Tgate: Accelerating cross-attention in diffusion models via token-level cache reuse. *Advances in Neural Information Processing Systems*, 37, 2024. 3

[7] 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. 3, 6

[8] Shoufa Chen, Mengmeng Xu, Jiawei Ren, Yuren Cong, Sen He, Yanping Xie, Animesh Sinha, Ping Luo, Tao Xiang, and Juan-Manuel Perez-Rua. Gentron: Delving deep into diffusion transformers for image and video generation. *arXiv preprint arXiv:2312.04557*, 2023. 1

[9] Huanpeng Chu, Wei Wu, Chengjie Zang, and Kun Yuan. Qncd: Quantization noise correction for diffusion models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 10995–11003, 2024. 2

[10] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021. 1

[11] Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 1

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 2

[13] Yefei He, Luping Liu, Jing Liu, Weijia Wu, Hong Zhou, and Bohan Zhuang. Ptqd: Accurate post-training quantization for diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[14] 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. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017. 6

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33(6):6840–6851, 2020. 1, 3

[16] Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021. 2

[17] Ziqi Huang, Yinan He, Jiashuo Yu, Chenyang Zhang, et al. Vbench: A comprehensive benchmark for video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 6

[18] Xin Jing, Yi Chang, Zijiang Yang, Jiangjian Xie, Andreas Triantafyllopoulos, and Bjoern W. Schuller. U-dit tts: U-diffusion vision transformer for text-to-speech. *Proceedings of the ITG Conference on Speech Communication*, pages 56–60, 2023. 1

[19] 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. *arXiv preprint arXiv:2411.02397*, 2024. 2, 3, 6

[20] Bo-Kyeong Kim, Hyoung-Kyu Song, Thibault Castells, and Shinkook Choi. Bk-sdm: Architecturally compressed stable diffusion for efficient text-to-image generation. *Proceedings of the Workshop on Efficient Systems for Foundation Models at ICML2023*, 2023. 1

[21] Eric Kim and Frank Lee. Kv caching in transformers for efficient inference. *Proceedings of the International Conference on Learning Representations*, 2020. 3

[22] Tuomas Kynkänniemi, Tero Karras, Samuli Laine, and Timo Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, 2019. 6

[23] Xi Li, Yifan Zhang, and Jie Chen. Q-align: A quality alignment metric for image-to-video generation. *arXiv preprint arXiv:2303.XXXX*, 2023. 6

[24] 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. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 2

[25] 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. *arXiv preprint arXiv:2411.19108*, 2024. 6

[26] Huadai Liu, Rongjie Huang, Xuan Lin, Wenqiang Xu, Maozong Zheng, Hong Chen, Jinzheng He, and Zhou Zhao. Vit-tts: Visual text-to-speech with scalable diffusion transformer. *arXiv preprint arXiv:2305.12708*, 2023. 1

[27] 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. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022. 1

[28] Haoyu Lu, Guoxing Yang, Nanyi Fei, Yuqi Huo, Zhiwu Lu, Ping Luo, and Mingyu Ding. Vdt: General-purpose video diffusion transformers via mask modeling. *International Conference on Learning Representations*, 2023. 1

[29] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021. 2

[30] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023. 1

[31] Zhaoyang Lyu, Xudong Xu, Ceyuan Yang, Dahua Lin, and Bo Dai. Accelerating diffusion models via early stop of the diffusion process. *arXiv preprint arXiv:2205.12524*, 2022. 2

[32] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. *arXiv preprint arXiv:2312.00858*, 2023. 2, 3

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

[34] Xin Ma, Yaohui Wang, Gengyun Jia, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024. 6

[35] Shentong Mo, Enze Xie, Ruihang Chu, Lanqing Hong, Matthias Niessner, and Zhenguo Li. Dit-3d: Exploring plain diffusion transformers for 3d shape generation. *Advances in Neural Information Processing Systems*, 36, 2024. 1

[36] Helen Nguyen and Isaac Patel. Advances in kv caching techniques for large-scale language models. *Advances in Neural Information Processing Systems*, 34, 2021. 3

[37] William Peebles and Saining Xie. Scalable diffusion models with transformers. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 1

[38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 1, 2

[39] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022. 1

[40] 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. 6

[41] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1972–1981, 2023. 2

[42] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 1

[43] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[44] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 1, 2

[45] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023. 1

[46] Zhenxiong Tan, Xinyin Ma, Gongfan Fang, and Xinchao Wang. Litefocus: Accelerated diffusion inference for long audio synthesis. *arXiv preprint arXiv:2407.10468*, 2024. 1

[47] Anwaar Ulhaq, Naveed Akhtar, and Ganna Pogrebna. Efficient diffusion models for vision: A survey, 2022. 2

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017. 1

[49] Xierui Wang, Siming Fu, Qihan Huang, Wanggui He, and Hao Jiang. Ms-diffusion: Multi-subject zero-shot image personalization with layout guidance. *arXiv preprint arXiv:2406.07209*, 2024. 1

[50] Y. Wang, X. Chen, J. Li, et al. Toca: Accelerating diffusion transformers with token-wise feature caching. *arXiv preprint arXiv:2408.XXXX*, 2024. 3, 6

[51] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 6

[52] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-gan: Training gans with diffusion. *arXiv preprint arXiv:2206.02262*, 2022. 2

[53] 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. *arXiv preprint arXiv:2312.03209*, 2023. 2

[54] Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, et al. Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. *arXiv preprint arXiv:2502.01776*, 2025. 2

[55] Xingyi Yang, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Diffusion probabilistic model made slim. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22552–22562, 2023. 1

[56] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Xiaotao Gu, Yuxuan

Zhang, Weihan Wang, Yean Cheng, Ting Liu, Bin Xu, Yuxiao Dong, and Jie Tang. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 2, 5, 6

[57] Qinsheng Zhang, Molei Tao, and Yongxin Chen. Gddim: Generalized denoising diffusion implicit models. *arXiv preprint arXiv:2206.05564*, 2022. 1

[58] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. 6

[59] Wentian Zhang, Haozhe Liu, Jinheng Xie, Mike Zheng, and Jürgen Schmidhuber. T-gate: Accelerating cross-attention in diffusion models via cache reuse. *arXiv preprint arXiv:2404.02747*, 2024. 2, 6

[60] Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Pab: Pyramid attention broadcast for real-time video generation. *arXiv preprint arXiv:2408.12588*, 2024. 6

[61] Zangwei Zheng et al. Opensora: Democratizing efficient video production for all, 2024. GitHub repository, 2024. 6