Less is Enough: Training-Free Video Diffusion Acceleration via Runtime-Adaptive Caching

Xin Zhou, Dingkang Liang, Kaijin Chen, Tianrui Feng, Xiwu Chen, Hongkai Lin, Yikang Ding, Feiyang Tan, Hengshuang Zhao, Xiang Bai, *Senior Member, IEEE*

Abstract—Video generation models have demonstrated remarkable performance, yet their broader adoption remains constrained by slow inference speeds and substantial computational costs, primarily due to the iterative nature of the denoising process. Addressing this bottleneck is essential for democratizing advanced video synthesis technologies and enabling their integration into real-world applications. This work proposes EasyCache, a training-free acceleration framework for video diffusion models. EasyCache introduces a lightweight, runtime-adaptive caching mechanism that dynamically reuses previously computed transformation vectors, avoiding redundant computations during inference. Unlike prior approaches, EasyCache requires no offline profiling, pre-computation, or extensive parameter tuning. We conduct comprehensive studies on various large-scale video generation models, including OpenSora, Wan2.1, and HunyuanVideo. Our method achieves leading acceleration performance, reducing inference time by up to 2.1-3.3× compared to the original baselines while maintaining high visual fidelity with a significant up to 36% PSNR improvement compared to the previous SOTA method. This improvement makes our EasyCache a efficient and highly accessible solution for high-quality video generation in both research and practical applications. The code is available at https://github.com/H-EmbodVis/EasyCache.

Index Terms—Video Generation, Diffusion Models, Inference Acceleration.

1 Introduction

Video generation [1], [2], [3], [4] has become a central task in generative modeling, with applications in digital content creation and virtual environment interaction. Recently, Diffusion Transformers (DiTs) [5] have emerged as the dominant paradigm in this domain, credited for their remarkable representational capability and scalability. Nevertheless, the practical adoption of DiTs faces significant hurdles due to high computational costs and prolonged inference durations, primarily resulting from the iterative denoising steps required to model complex spatiotemporal interactions (as in Fig.1(a)). For example, generating a 5s, 720P video with HunyuanVideo [6] requires ∼2 H20 GPU hours. These computational demands substantially limit their feasibility in real-time scenarios and resource-constrained environments, thus restricting broader industrial and practical utilization.

The community has explored various acceleration approaches [7], [8], [9], [10] to mitigate computational barriers. While methods based on distillation or architectural changes can accelerate inference, they require extensive training datasets and significant resources. This motivates growing interest in training-free techniques. Among these, feature caching, which reuses intermediate results across denoising steps, has shown considerable promise. Some prior works [11], [12], [13], [14] relied on static caching schemes, reusing computations at fixed intervals (as depicted in Fig. 1(b)). However, such uniform caching lacks adaptability and often fails to align with the dynamic behavior of generative models, limiting cache effectiveness when output variations differ significantly across timesteps.

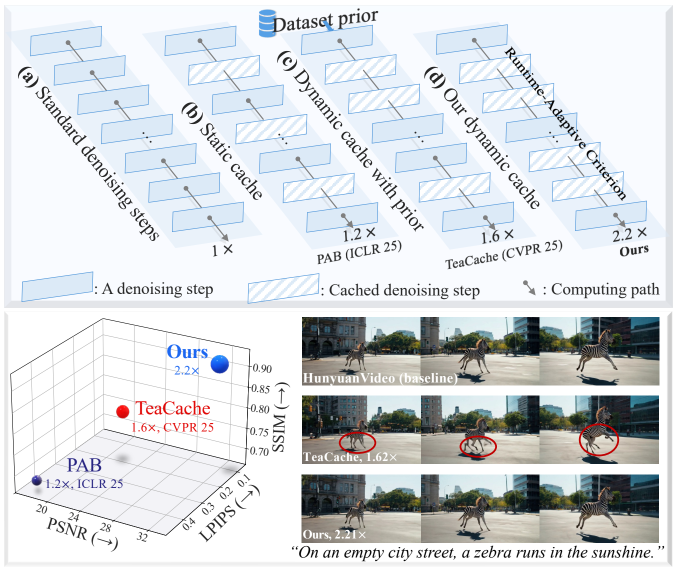


Fig. 1: The comparison between (a) the default iterative denoising, (b) static caching with fixed intervals, (c) dynamic cache with external "Dataset prior", and (d) our dynamic cache reuses computation by a runtime-adaptive criterion.

A recent pioneer work, TeaCache [15], shifts the focus to dynamic caching strategies, which leverage polynomial fitting and extensive offline profiling of dataset-specific priors to estimate suitable cache intervals (Fig. 1(c)). Although flexible, TeaCache suffers from practical limitations. Its reliance on extensive offline profiling and meticulous hyperparameter tuning makes it sensitive to dataset distributions,

- *X. Zhou, D. Liang, K. Chen, T. Feng, H. Lin, and X. Bai are with Huazhong University of Science and Technology. E-mail: (xzhou03, dkliang, xbai)@hust.edu.cn*
- *X. Chen, Y. Ding, and F. Tan are with MEGVII Technology.*
- *H. Zhao is with the University of Hong Kong.*
- *Xin Zhou and Dingkang Liang make equal contributions. The corresponding author is Xiang Bai (xbai@hust.edu.cn).*

undermining its generalizability. Moreover, the dependency on the dataset prior may misalign caching strategies with the model's internal dynamics, compromising generation fidelity, especially with aggressive feature reuse. These challenges naturally lead to a critical research question: *How can we design a runtime-adaptive caching framework that accelerates DiT-based video generation while preserving generation fidelity?*

To answer this question, we begin by analyzing the internal feature dynamics of DiTs during the iterative generation process, revealing a notable stability in model transformation rates (the ratio of the change in predicted noise and latent input) throughout denoising. Inspired by this stability, we posit that intermediate outputs can be reliably estimated using previously computed transformations. This motivates the design of our **EasyCache**, a simple yet effective feature caching framework that enables adaptive computation reuse for accelerating inference without any extra training.

Specifically, EasyCache monitors the relative transformation rate and detects local stability via a dynamic, runtime-adaptive thresholding scheme. When stability is estimated, it reuses previously computed transformation vectors to approximate future outputs, thus avoiding redundant full-model evaluations. Crucially, this approximation is governed by an accumulated local error indicator, which ensures that computation reuse is only triggered under reliable conditions. Unlike prior methods, which rely on costly offline statistics and fixed heuristics (e.g., polynomial fitting), EasyCache makes fully online decisions based on lightweight first-order dynamics, requiring no retraining, dataset-specific tuning, and architectural modification.

Extensive experiments on several video generation models demonstrate that EasyCache achieves significant speedups and outstanding visual retention with original videos. On the computationally demanding HunyuanVideo [6], EasyCache improves the speedup over the SOTA caching strategy TeaCache [15] by 36% while boosting the PSNR by 37% and SSIM by 14%. Moreover, EasyCache is able to achieve continuous acceleration on the already accelerated efficient attention [16], pushing the speedup to 3.3$\times$ with only a slight 0.3 PSNR drop.

Our main contributions are summarized as follows: **1)** We reveal an exploitable relative stability in the transformation rates throughout the denoising process for Diffusion Transformers. **2)** We propose EasyCache, a training-free feature caching framework that exploits this stability through a lightweight, runtime-adaptive criterion, enabling substantial inference acceleration with minimal compromising model integrity. **3)** Extensive experiments across various video generation models demonstrate that our EasyCache achieves significant speedups while faithfully preserving visual fidelity, outperforming SOTA caching methods in both inference speed and visual retention. Moreover, EasyCache is orthogonally compatible with other acceleration strategies, further enhancing its practical applicability.

2 Related work

2.1 Diffusion Model

Diffusion models [5], [17] have revolutionized generative modeling across diverse domains, becoming a dominant paradigm for high-quality content synthesis. These probabilistic models have achieved remarkable success in fields including image generation [18], [19], [20], 3D object synthesis [21], [22], [23], and video creation [24], [25], [26], [27].

Within this landscape, video generation has garnered significant attention due to the growing demand for dynamic content creation. However, generating coherent and high-fidelity video presents significant challenges due to the complex temporal dynamics and the substantial computational overhead with long sequences. As a result, the field has witnessed a notable shift from traditional U-Net towards more scalable Diffusion Transformers (DiTs) [5], which have demonstrated strong capabilities in handling complex data modalities. DiT-based video generative models like Sora [3], HunyuanVideo [6], and Wan2.1 [4] have demonstrated exceptional capabilities in modeling complex spatiotemporal relationships. Despite their power, the iterative nature of these models leads to significant inference latency and remains a key adoption barrier, necessitating further research into efficient generation methods.

2.2 Training-free Diffusion Inference Acceleration

Training-free inference acceleration techniques [7], [10], [28], [29], [30], [31], [32], [33] are achieving significant attention as they speed up inference without resource-intensive retraining, thus preserving the integrity of large-scale pretrained models. Many methods are designed to reduce per-step costs. For example, efficient attention implementation like SVG [16] utilizes inherent sparsity in the 3D full attention, training-freely achieving promising inference speed amelioration.

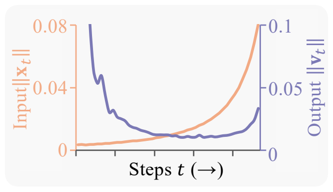
Another representative approach for reducing per-step cost is feature caching, which exploits computational redundancy in the iterative denoising process. Early static caching [13], [14] uses pre-determined rules but lacks the adaptability to non-uniform process dynamics. For example, PAB [12] employs a pyramid-style heuristic to broadcast attention at fixed intervals. Dynamic caching [15], [34] is then proposed to address this rigidity. The current SOTA, TeaCache [15], exemplifies a dynamic strategy that relies on an external dataset prior, modeling the relationship between timestep embeddings and output changes with a pre-fitted polynomial. While more adaptive, this reliance on offline profiling hinders its generalizability. Our work is motivated to develop a fully dynamic caching framework that eliminates the need for external priors by leveraging the intrinsic runtime dynamics of the inference process.

3 EasyCache

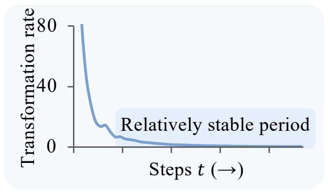
This section introduces our training-free runtime-adaptive dynamic caching framework for Diffusion Transformer (DiT)-based video generation.

3.1 A Revisit of Diffusion Modeling with Transformer

The Diffusion Transformer (DiT) [5] integrates Transformer architectures into diffusion models [17], [35], achieving superior scalability and performance for video generation. In this paradigm, noisy latent input $\mathbf{x}_t$ at t-th step is



(a) The input and output of each step



(b) The transformation rate of each step

Fig. 2: Analysis of feature dynamics. (a) The L1 norm of the input and output of each step. (b) The changes in the relative transformation rate between consecutive steps.

partitioned into non-overlapping patches, processed by sequential Transformer layers, and modulated by a learned time embedding $\mathbf{e}_t$ that reflects the noise level. During inference, a latent variable sampled from Gaussian noise is iteratively denoised over T steps to produce the final data sample. However, DiT-based video generation remains computationally intensive due to the need to model complex spatiotemporal dependencies and perform iterative denoising over many steps. For instance, generating a 5s, 720P video with HunyuanVideo [6] or Wan2.1-14B [4] requires $\sim$2 H20 hours. Such high computational demands significantly impede applications requiring user interaction or low latency. Although model-specific fine-tuning or retraining can improve efficiency, these methods demand substantial computational resources and large-scale datasets, limiting their accessibility. Therefore, it is critical to develop efficient, training-free acceleration techniques that can reduce inference costs while preserving generation quality.

3.2 Investigation of Feature Dynamics

To develop a reliable protocol for dynamic feature caching, we first systematically analyze the feature dynamics within diffusion transformers (DiTs). During inference, the network u_θ estimates the noise component $\mathbf{v}_t$ from the noisy latent $\mathbf{x}_t$ at each step t, conditioned on a prompt $\mathcal{T}$. The iteration at step t can be simply referred to $\mathbf{v}_t = u_\theta(\mathbf{x}_t \mid \mathcal{T})$.

Feature Dynamics Investigation. The most straightforward metric to investigate the correlation of DiT's internal feature is to evaluate feature dynamics as the norm of the input and output for each step, i.e., $\|\mathbf{x}_t\|$ and $\|\mathbf{v}_t\|$, where $\|\cdot\|$ denotes the L1 norm and an average operation. As shown in Fig. 2(a), $\|\mathbf{x}_t\|$ and $\|\mathbf{v}_t\|$ exhibit a growth trend and U-shaped curve over the diffusion process, respectively, showing a completely different changing pattern and making a difficult feature dynamic estimation.

To mitigate this issue, TeaCache [15] estimates output dynamics for feature caching using polynomial fitting to determine scaling factors between input and output relative differences. However, this approach requires heuristic factor fitting and separate fitting for different settings (e.g., resolution changes).

Transformation Rate Stability. To establish a criterion for adaptive computation reuse, we quantify the stability of the DiT's output with respect to input changes. Inspired by the concept of function derivatives, we approximate the "*directional derivative*" of the model u_θ with respect to input latent $\mathbf{x}_t$ using a first-order approximation:

$$\frac{\partial u_\theta}{\partial \mathbf{x}_t} \approx \frac{u_\theta(\mathbf{x}_t \mid \mathcal{T}) - u_\theta(\mathbf{x}_{t-1} \mid \mathcal{T})}{\mathbf{x}_t - \mathbf{x}_{t-1}} = \frac{\mathbf{v}_t - \mathbf{v}_{t-1}}{\mathbf{x}_t - \mathbf{x}_{t-1}}. \tag{1}$$

This represents the ratio of the output change to the input change between consecutive steps. To better access and simplify this "derivative", we adopt the L1 norm and average to reduce the derivative at step t to a number by:

$$k_t = \frac{\|\mathbf{v}_t - \mathbf{v}_{t-1}\|}{\|\mathbf{x}_t - \mathbf{x}_{t-1}\|}. \tag{2}$$

While k_t is not a formal derivative, it serves a similar purpose, and we define it as the relative transformation rate.

Surprisingly, as shown in Fig. 2(b), k_t rapidly stabilizes and remains relatively stable after an initial period of sharp fluctuation. We hypothesize that this reflects the DiT model's evolving role: early steps focus on establishing the global layout with a non-linear mapping, while later steps refine local details with a stable, approximately linear relationship. This largely relative linear relationship across a wide range of the denoising process motivates our method, suggesting reliable output estimation and enabling runtime-adaptive criterion of redundant computations.

3.3 Runtime-Adaptive Criterion

Capitalizing on the observed stability of the transformation rate, we propel forward with introducing a simple yet effective adaptive computation reuse framework, denoted as EasyCache. It operates on a core principle, i.e., approximating the model's output using previously computed transformations during stable diffusion phases, employing a lightweight criterion to ensure approximation accuracy.

Efficient Output Approximation. The foundation of EasyCache lies in the observation from Sec. 3.2 that the transformation of $\mathbf{x}_t$ into predicted noise $\mathbf{v}_t$ is expected to exhibit local consistency during a relatively stable phase. We define the transformation vector at step t as $\Delta_t := \mathbf{v}_t - \mathbf{x}_t$.

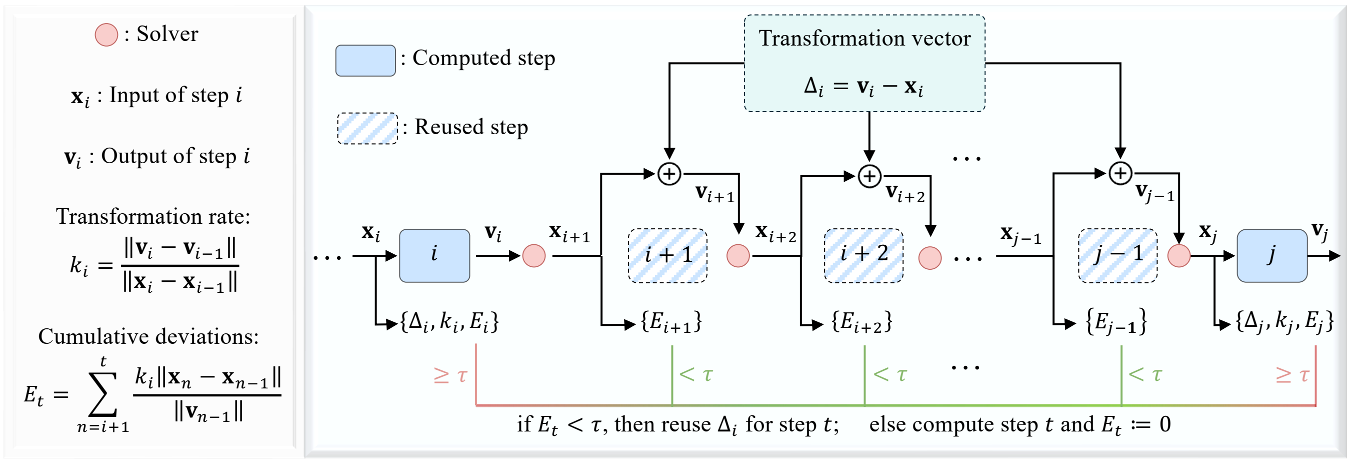
Our central hypothesis is that in the relatively stable regions of the diffusion process, the current transformation vector Δ_t can be well approximated by one from a recent fully computed step $i < t$ by $\Delta_t \approx \Delta_i$. Thus, for steps deemed skippable, we approximate the model output $\mathbf{v}_t$ using this cached transformation vector Δ_i:

$$\hat{\mathbf{v}}_t = \mathbf{x}_t + \Delta_i, \quad \text{where } \Delta_i := \mathbf{v}_i - \mathbf{x}_i. \tag{3}$$

This approximation incurs negligible computational cost compared to a full forward pass $u_\theta(\mathbf{x}_t|\mathcal{T})$. The critical challenge, addressed next, is to establish a lightweight yet reliable criterion for determining when this reuse is appropriate without sacrificing generation quality.

Adaptive Caching Criterion. The approximation in Eq. 3 relies on an adaptive criterion to determine whether the transformation vector remains nearly unchanged, i.e., whether $||\Delta_t - \Delta_i||$ $(t > i \geq 1)$ is small. However, directly calculating this difference at every step is non-trivial. To address this, we introduce a proxy measure informed by the inherent dynamics of the diffusion process.

As shown in Fig. 3, our goal is to propose an adaptive caching criterion that governs feature reuse for any step t following the most recent fully computed step, denoted as i. Our central hypothesis is that if the expected change in the model output $\mathbf{v}_t$ relative to $\mathbf{v}_{t-1}$ is small, the process is considered locally relatively stable. This stability, in turn, implies that the transformation vector Δ_t also remains



: Solver
$\mathbf{x}_i$: Input of step i
$\mathbf{v}_i$: Output of step i
Transformation rate:
$k_i = \frac{\|\mathbf{v}_i - \mathbf{v}_{i-1}\|}{\|\mathbf{x}_i - \mathbf{x}_{i-1}\|}$
Cumulative deviations:
$E_t = \sum_{n=i+1}^{t} \frac{k_i\|\mathbf{x}_n - \mathbf{x}_{n-1}\|}{\|\mathbf{v}_{n-1}\|}$
: Computed step
: Reused step
Transformation vector
$\Delta_i = \mathbf{v}_i - \mathbf{x}_i$
$\{\Delta_i, k_i, E_i\}$
$\{E_{i+1}\}$
$\{E_{i+2}\}$
$\{E_{j-1}\}$
$\{\Delta_j, k_j, E_j\}$
$\geq \tau$
$< \tau$
if $E_t < \tau$, then reuse Δ_i for step t; else compute step t and $E_t := 0$

Fig. 3: The overall of our method. For simplicity, we start from a computed step i. A runtime-adaptive criterion evaluates each subsequent step, reusing the cached transformation vector Δ_i while the accumulated deviation E_t remains below a threshold τ. A full computation is performed when the threshold is exceeded, as exemplified in step j.

nearly constant, thus justifying the safe reuse of the cached transformation vector Δ_i from step i.

To estimate the output for step t, we approximate $||\mathbf{v}_t - \mathbf{v}_{t-1}||$ as $k_i||\mathbf{x}_t - \mathbf{x}_{t-1}||$ by making a constant assumption for k_i. Based on this, we define the local stability indicator (output change rate) ε_t as the percentage of the estimated output change at step t relative to the latest output $\mathbf{v}_{t-1}$:

$$\varepsilon_t = \frac{||\mathbf{v}_t - \mathbf{v}_{t-1}||}{||\mathbf{v}_{t-1}||} \approx \frac{k_i||\mathbf{x}_t - \mathbf{x}_{t-1}||}{||\mathbf{v}_{t-1}||} \times 100\%. \tag{4}$$

A small ε_t indicates that the output is evolving smoothly and predictably with respect to input variations. To account for cumulative deviations since i, we sum these indicators:

$$E_t = \sum_{n=i+1}^{t} \varepsilon_n. \tag{5}$$

The transformation Δ_i is reused for step t if this accumulated stability indicator E_t remains below a predefined tolerance threshold τ. Thus, $E_t < \tau$ is the practical condition for deeming the process stable enough for approximation. A larger τ allows for greater tolerance to cumulative deviations, enabling more aggressive feature reuse, while a smaller τ potentially promotes visual fidelity through conservative reuse. As a key hyperparameter, τ can be easily tuned for a trade-off between speed and quality.

The Integrated EasyCache Framework. Finally, the complete EasyCache integrates the runtime-adaptive criterion and the approximation strategy into a unified workflow. The update rule for $\mathbf{v}_t$ at any given step t is as follows:

$$\mathbf{v}_t = \begin{cases} u_\theta(\mathbf{x}_t|\mathcal{T}), & \text{if } E_t \geq \tau \text{ or } t \in [0, R-1] \cup \{T-1\} \\ \mathbf{x}_t + \Delta_i, & \text{otherwise} \end{cases}. \tag{6}$$

Here, R is the number of initial warm-up steps where full computation is mandatory to capture the highly changing initial diffusion phase. For any step t, if $E_t \geq \tau$, $t < R$ or $t = T-1$, a full computation $u_\theta(\mathbf{x}_t|\mathcal{T})$ is performed. Subsequently, the cache is updated by storing the new transformation vector Δ_t, the reference step is set to $i := t$, and the cumulative deviation is reset to $E_t := 0$. Otherwise, the cached vector Δ_i is reused. Our EasyCache ensures that computational resources are dynamically allocated, focusing on steps that introduce significant changes.

3.4 An Intuitive Perspective on the Relative Stability

We analyze the stability of the transformation rate k (Eq. 2) during video generation, using widely used Ordinary Differential Equation (ODE)-based sampler in video generation field, i.e., flow matching [36]. Typically, the current $\mathbf{x}_t$ is updated via an Euler step: $\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{v}_{t-1}\Delta s_t$, where Δs_t represents the continuous time step from step $t-1$ to t, and $\mathbf{v}_{t-1}$ is the predicted velocity field. Therefore, substituting this into Eq. 2, we can obtain:

$$k_t = \frac{\|\mathbf{v}_t - \mathbf{v}_{t-1}\|}{\|\mathbf{x}_{t-1} + \mathbf{v}_{t-1}\Delta s_t - \mathbf{x}_{t-1}\|} = \frac{\|\mathbf{v}_t - \mathbf{v}_{t-1}\|}{\|\mathbf{v}_{t-1}\Delta s_t\|}. \tag{7}$$

The stability can be understood as follows: 1) The training process encourages a linear velocity field that smoothly transforms noise into target data. In the later stages of generation, the transformation from $\mathbf{v}_{t-1}$ to $\mathbf{v}_t$ becomes near-linear, approximating $\mathbf{v}_t \approx c\mathbf{v}_{t-1}$, where c is a scalar or a near-scalar transformation that varies slowly during the stable phase. Thus, $k_t \approx \frac{|c-1|}{\Delta s_t}$; 2) Given that Δs_t is typically determined by a fixed scheduler (potentially constant or slowly varying), k_t quickly stabilizes after an initial period of fluctuation during non-linear structure formation. This relative stability enables the training-free, runtime-adaptive identification of redundant computation steps, allowing selective reuse of transformation vectors with minimal output sacrifice. As a result, our EasyCache substantially reduces inference cost while maintaining strong visual fidelity since approximations are applied when the model's behavior is intrinsically predictable.

4 Experiments

4.1 Experiment Setup

Evaluation Metrics. We evaluate our EasyCache on text-to-video generation and its generalization to text-to-image, focusing on inference efficiency (latency and speedup of the DiT modules) and generation quality. For the primary task of text-to-video generation, we use default prompts in VBench [40] to assess visual retention. Specifically, we measure pixel-level fidelity, structural similarity, and perceptual consistency using PSNR, SSIM [41], and LPIPS [42]

TABLE 1: The comparison between other acceleration strategies and our EasyCache. We report inference latency, speedup, and visual quality metrics (PSNR, SSIM, LPIPS, and VBench) on representative video generation models.

Methods	Reference	Efficiency		Visual Retention			VBench (%) ↑
		Latency (s) ↓	Speedup ↑	PSNR ↑	SSIM ↑	LPIPS ↓	
			Open-Sora 1.2 [1] (51 frames, 848×480)				
Open-Sora 1.2 ($T = 30$)	-	44.90	1×	-	-	-	**79.40**
+ 50% steps	-	22.71	1.98×	15.82	0.6961	0.3363	77.36
+ Random 0.5	-	22.68	1.98×	16.51	0.7037	0.3264	76.78
+ Static cache	-	24.19	1.86×	15.73	0.6961	0.3382	77.37
+ T-GATE [37]	TMLR 25	39.70	1.13×	19.55	0.6927	0.2612	75.42
+ PAB [12]	ICLR 25	31.57	1.42×	23.58	0.8220	0.1743	76.95
+ TeaCache [15]	CVPR 25	28.92	1.55×	23.56	0.8433	0.1318	79.27
+ **EasyCache (ours)**	-	**21.21**	**2.12**×	**23.95**	**0.8556**	**0.1235**	78.74
			Wan2.1-1.3B [38] (81 frames, 832×480)				
Wan2.1 ($T = 50$)	-	175.35	1×	-	-	-	**81.30**
+ 40% steps	-	70.10	2.50×	14.50	0.5226	0.4374	80.30
+ Random 0.4	-	71.69	2.45×	11.92	0.4204	0.5911	78.68
+ Static cache	-	71.54	2.45×	14.18	0.5007	0.4789	79.58
+ PAB [12]	ICLR 25	102.03	1.72×	18.84	0.6484	0.3010	77.60
+ TeaCache [15]	CVPR 25	87.77	2.00×	22.57	0.8057	0.1277	81.04
+ **EasyCache (ours)**	-	**69.11**	**2.54**×	**25.24**	**0.8337**	**0.0952**	80.49
			HunyuanVideo [6] (129 frames, 960×544)				
HunyuanVideo ($T = 50$)	-	1124.30	1×	-	-	-	82.20
+ 50% steps	-	566.17	1.99×	18.79	0.7101	0.3319	81.78
+ Random 0.5	-	578.83	1.94×	19.85	0.7201	0.3214	81.04
+ Static cache	-	573.38	1.96×	18.74	0.7081	0.3309	81.76
+ PAB [12]	ICLR 25	958.23	1.17×	18.58	0.7023	0.3827	76.98
+ TeaCache [15]	CVPR 25	674.04	1.67×	23.85	0.8185	0.1730	**82.32**
+ SVG [16]	ICML 25	802.70	1.40×	26.57	0.8596	0.1368	81.97
+ **EasyCache (ours)**	-	**507.97**	**2.21**×	**32.66**	**0.9313**	**0.0533**	82.01

TABLE 2: Generalization to text-to-image generation.

Method	Efficiency		Visual quality	
	Latency (s) ↑	Speedup ↑	FID-30k ↓	CLIP Score↑
FLUX.1-dev [39]	25.5	1×	25.8	26.0
+ Teacache [15]	7.8	3.27×	24.5	**26.2**
+ **EasyCache (ours)**	**5.5**	**4.64**×	**23.2**	26.1

against the original videos. We also report the VBench score for human perceptual alignment. For image generation, we generate 30,000 samples from COCO-2017 [43] captions, reporting FID-30k for perceptual quality and CLIP Score [44] for semantic alignment.

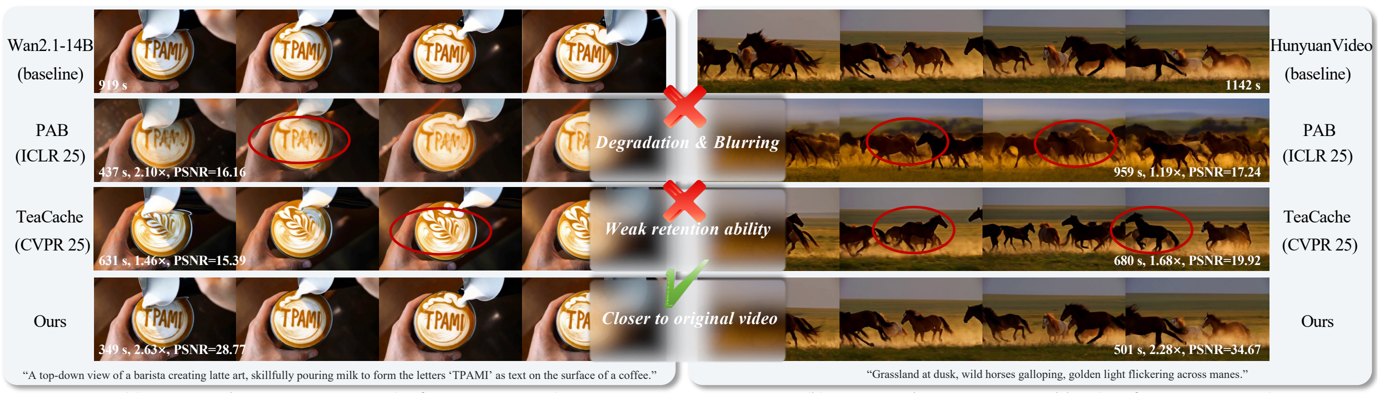
Implementation Details. Our EasyCache only requires a few hyperparameter tuning. To achieve a desirable trade-off between inference efficiency and generation quality, we set (τ, R) as (10%, 5) for Open-Sora 1.2 [1], (5%, 10) for Wan2.1 [38], and (2.5%, 5) for HunyuanVideo [6]. For the principal quantitative evaluations reported using the VBench default prompts, we generate 5 video samples with varying seeds per prompt on NVIDIA A800 GPUs. To expedite ablation studies, we generate a single video per prompt with seed 0, using Wan2.1-1.3B as the baseline.

4.2 Main Results

We conduct experiments on leading video generation models, including Open-Sora 1.2 [1], Wan2.1-1.3B [38], and HunyuanVideo [6], to rigorously evaluate the acceleration and visual retention of EasyCache.

We first compare EasyCache with several naive training-free acceleration strategies, including direct step reduction (e.g., using only 50% of the denoising steps), probabilistic caching (Random 0.5), and static interval caching. While these straightforward methods are easy to implement and can be tuned to provide a similar speedup as EasyCache, their lack of adaptability to the evolving dynamics of the diffusion process leads to significant quality degradation. As shown in Tab. 1, static cache and direct step reduction result in over 40% lower PSNR and more than 20% reduction in SSIM compared to EasyCache on HunyuanVideo. This substantial gap shows that static caching fails to accommodate the non-uniform temporal variation inherent in video generation. In contrast, our EasyCache employs a runtime-adaptive criterion to selectively reuse computation during locally relatively stable phases in a training-free manner, thereby achieving both superior acceleration and markedly better visual fidelity across all evaluated baselines.

Compared to advanced training-free acceleration methods such as PAB [12] and TeaCache [15], our EasyCache demonstrates clear advantages in both efficiency and visual quality. As shown in Tab. 1, on Wan2.1-1.3B, EasyCache achieves a 2.54× speedup, substantially faster than TeaCache (2.0×) and PAB (1.7×), while maintaining superior visual fidelity with a 11.8% improvement in PSNR and a 25.5% reduction in LPIPS relative to the best-competing method TeaCache. On more time-consuming HunyuanVideo, EasyCache delivers a significant 36.9% higher PSNR and a 69.2% lower LPIPS compared to TeaCache and a notable 6.09dB PSNR and 57.9% speedup improvement over the leading efficient attention implementation SVG [16], resulting in markedly superior perceptual similarity. We observe that the VBench score [40] exhibits a negligible



(a) Comparation on Wan2.1-14B (81frames, 832×480) (b) Comparation on HunyuanVideo (129frames, 960×544)

Fig. 4: Qualitative comparison of EasyCache with baseline and prior acceleration methods [12], [15]. On (a) Wan2.1-14B [4] and (b) HunyuanVideo [6]. Our approach consistently produces results that are closer to the original video.

drop (e.g., -0.19% for HunyuanVideo). This can be attributed to the evaluation methodology of certain sub-dimensions within VBench, such as its sensitivity to minor structural variations that are not perceptible to human observers. Importantly, all observed VBench deviations remain below 1%, indicating that the visual retention achieved by EasyCache is well within acceptable bounds for practical use.

Unlike PAB's pyramid-style attention broadcast at fixed intervals and TeaCache's offline heuristics coming from the dataset prior, our EasyCache dynamically identifies and exploits locally relatively stable phases. This enables EasyCache to achieve higher acceleration with less perceptual quality sacrifice, setting a new standard for training-free diffusion model inference.

To further validate the broader applicability of our acceleration strategy, we finally evaluate its performance on text-to-image generation using the 50-step FLUX.1-dev [39]. As shown in Tab. 2, EasyCache achieves approximately 40% speedup over TeaCache and markedly reduces latency compared to the FLUX.1-dev baseline. Notably, our approach maintains or even improves generation quality, outperforming FLUX.1-dev by 10% in FID and 0.1 in CLIP score. These strong performances highlight the intrinsic generalizability of our runtime-adaptive criterion, enabling adequate acceleration and visual quality across both video and image generation.

4.3 Qualitative Results

We present qualitative comparisons in Fig. 4 with prior caching strategies [12], [15] on two computationally intensive baselines, Wan2.1-14B [4] and HunyuanVideo [6]. Here, we present results for single-sample, noting that such case studies may exhibit more variability than the aggregated metrics in Tab. 1. As shown in Fig. 4(a), the less adaptive caching in PAB [12] results in notable degradation and blurring, rendering text and fine details illegible. While dynamic approaches like TeaCache [15], which depend on offline dataset priors, frequently fail to retain intricate patterns unique to each sample, as in Fig. 4(b). This limitation emphasizes the difficulty of generalizing to the diverse instance-specific dynamics encountered during inference. In comparison, our EasyCache, driven by a runtime-adaptive criterion, closely matches the outputs of the original baseline even under aggressive acceleration (e.g., 2.63× speedup on Wan2.1-14B versus 1.46× with TeaCache). This superior performance is attributed to leveraging the intrinsic transformation stability of the diffusion process and eliminating the need for offline profiling, thereby ensuring both fidelity and adaptability across models and settings. Additional qualitative results are provided at https://H-EmbodVis.github.io/EasyCache.

TABLE 3: Ablation on the error threshold τ.

τ	Efficiency		Visual Retention		
	Latency (s) ↓	Speedup ↑	PSNR ↑	SSIM ↑	LPIPS ↓
2%	109.18	1.61×	30.73	0.9342	0.0356
3%	89.51	1.96×	27.58	0.8882	0.0633
5%	69.11	2.54×	24.74	0.8041	0.1121
7%	63.57	2.76×	23.22	0.7328	0.1541
10%	56.77	3.09×	21.67	0.6428	0.2159

4.4 Ablation Study

We conduct ablation studies on the VBench prompts, each generating one video with Wan2.1-1.3B [4]. Note that the primary objective is to achieve an optimal trade-off between computational efficiency and visual fidelity of the generated videos. The default settings are marked in gray.

The effect of tolerance threshold τ. We first analyze the effect of the tolerance threshold τ, which dictates the aggressiveness of computation reuse in Tab. 3. A smaller τ (e.g., 2%) yields better visual retention (PSNR 30.73) but substantially lower speedup (1.61×). While a higher $\tau = 10\%$ significantly boosts speedup with a 21.7% improvement over $\tau = 5\%$, but at the cost of considerable degradation in visual quality. Our default $\tau = 5\%$ achieves a compelling trade-off, providing a 2.54× speedup while maintaining reasonable visual fidelity (PSNR 24.74), effectively balancing efficiency gains with quality preservation.

The effect of initial warm-up steps R. The warm-up steps R controls the number of fully computed initial steps. As discussed in Sec. 3.2, the initial diffusion phase exhibits rapid and unstable changes in transformation rate changes, and Tab. 4 shows that fully computing initial steps is beneficial. Reducing R slightly increases speedup but also causes visual quality degradation. Increasing R to 15 improves visual metrics but reduces the speedup to 2.10× due to more full computations. We find that $R = 10$ in

TABLE 4: Ablation on initial warm-up steps number R.

R	Efficiency		Visual Retention		
	Latency (s) ↓	Speedup ↑	PSNR ↑	SSIM ↑	LPIPS ↓
2	66.73	2.63×	23.39	0.7665	0.1389
5	67.21	2.61×	23.45	0.7689	0.1374
10	69.11	2.54×	24.74	0.8041	0.1121
15	83.49	2.10×	26.48	0.8372	0.0874

TABLE 5: Ablation on criteria for adaptive reuse.

Adaptive Criterion	Efficiency		Visual Retention		
	Latency (s) ↓	Speedup ↑	PSNR ↑	SSIM ↑	LPIPS ↓
Probabilistic	70.39	2.49×	22.46	0.6965	0.1846
Output-relative	69.01	2.54×	23.14	0.7669	0.1422
w/o Re-compute	72.47	2.42×	17.70	0.3984	0.4876
Ours	69.11	2.54×	24.74	0.8041	0.1121

TABLE 6: Ablation on update strategies for k.

Update of k	Efficiency		Visual Retention		
	Latency (s) ↓	Speedup ↑	PSNR ↑	SSIM ↑	LPIPS ↓
k_{avg}	107.40	1.63×	27.18	0.8935	0.0624
k_{EMA}	73.14	2.40×	24.94	0.8111	0.1073
k_{local}	69.11	2.54×	24.74	0.8041	0.1121

Wan2.1 provides a good balance, ensuring a more stable transformation rate k for reliable subsequent computation reuse, preserving visual quality while achieving a solid 2.54× speedup.

The effect of adaptive reuse criterion. We further evaluate different adaptive reuse criteria in Tab. 5, using $R = 10$ initial steps and similar speedups (∼2.5×) for fair comparison. The probabilistic caching baseline, which reuses computations with a fixed probability, shows suboptimal visual retention, indicating the limitations of static strategies. The output-relative method [34], which measures reuse based on the relative output change $\|\mathbf{v}_{t-1} - \mathbf{v}_{t-2}\|/\|\mathbf{v}_{t-1}\|$, considers recent output history and achieves moderate performance but is less effective due to the U-shaped change pattern observed in Fig. 2(a). We also evaluate a without re-computing variant, where the first 20 steps are fully computed, and all subsequent steps directly reuse the transformation vector without any correction. Although the changes in later steps are relatively stable, this still leads to the most degradation in visual quality due to accumulated errors. In contrast, our method dynamically adjusts reuse based on the relationship between input and output changes, effectively correcting potential errors and achieving the best balance between efficiency and fidelity.

The effect of transformation rate k update strategy. Accurate estimation of the transformation rate k is essential for our effective adaptive caching. As shown in Tab. 6, we compare three update strategies: 1) history averaging (k_{avg}), which averages all k values after the R steps warm-up; 2) an EMA-like approach (k_{EMA}), utilizing an exponential moving average as $k_{\text{EMA}}^{(i)} = 0.9k_{\text{EMA}}^{(i-1)} + 0.1k^{(i)}$; and 3) our default local update (k_{local}), which uses the most recent fully computed interval from Eq. 2. While k_{avg} achieves the highest visual quality, it sacrifices speedup due to slow response to local dynamics. k_{EMA} slightly improves PSNR over k_{local}, but at the cost of reduced efficiency. Our k_{local} method achieves the best balance, providing the 2.54× speedup and competitive visual quality by promptly adapting to diffusion dynamics.

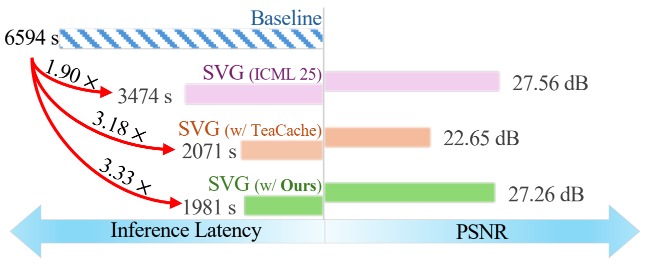


Fig. 5: The compatibility with other acceleration techniques (SVG) [16] on HunyuanVideo (129frames, 1280×720).

TABLE 7: Speedup comparison on GPU architectures.

Methods	Ampere (A800)	Hopper (H20)
HunyuanVideo [6]	3064 s	6594 s
+ SVG [16]	1959 s (1.6×)	3474 s (1.9×)
+ Ours	**1335 s** (2.3×)	**2897 s** (2.3×)

4.5 Compatibility with Other Acceleration Techniques

This subsection analyzes the flexibility of our approach by evaluating its compatibility with additional acceleration strategies on the time-consuming HunyuanVideo [6]. Experiments are performed on 35 prompts uniformly sampled across seven dimensions of the T2V-CompBench [45] due to the high computational cost. As shown in Fig. 5, generating a 5s 720P video takes 2 hours, even on an expensive H20 GPU. While the advanced, efficient attention method (SVG) [16] alone already achieves a 1.90× speedup with a PSNR of 27.56, integrating EasyCache pushes the total speedup to an impressive 3.33×, reducing the generation time for a high-resolution video from nearly 2 hours to just 33 minutes, with only a marginal 1.1% PSNR drop. Although SVG combined with TeaCache [15] achieves a 3.18× speedup, it leads to a significant 17.8% reduction in PSNR. We argue that TeaCache may fail to precisely identify the steps that are more important for visual retention. Notably, as shown in Tab. 7, our method results in consistent acceleration across both Hopper (H20) and Ampere (A800) architectures, whereas SVG's gains are more hardware-dependent, making EasyCache particularly suitable for diverse academic and practical environments.

5 Conclusion

This paper aims to alleviate the high computational cost of Diffusion Transformer-based video generation in a training-free manner. Our key insight is the relative stability of the transformation rate during iterative denoising, which enables dynamic computation reuse. Building on this, our EasyCache employs a lightweight, runtime-adaptive criterion that leverages this intrinsic stability to reuse computations dynamically, eliminating the need for the costly offline profiling and external dataset priors that constrain existing methods. Extensive experiments show that EasyCache achieves state-of-the-art acceleration while preserving visual fidelity. We believe our work offers a new perspective and a practical, efficient solution for accelerating diffusion models, facilitating their broader application.

Limitation. Although EasyCache achieves an impressive 2.1–3.3× speedup, a significant gap remains from real-time generation. This is a common challenge mainly due to the need for large-scale models to generate high-quality videos. Enabling truly real-time video generation is an important direction for future research.

REFERENCES

[1] Z. Zheng, X. Peng, T. Yang, C. Shen, S. Li, H. Liu, Y. Zhou, T. Li, and Y. You, "Open-sora: Democratizing efficient video production for all," *arXiv preprint arXiv:2412.20404*, 2024.

[2] X. Ma, Y. Wang, G. Jia, X. Chen, Z. Liu, Y.-F. Li, C. Chen, and Y. Qiao, "Latte: Latent diffusion transformer for video generation," *Trans. Mach. Learn. Research*, 2024.

[3] T. Brooks *et al.*, "Video generation models as world simulators," *OpenAI Blog*, vol. 1, p. 8, 2024.

[4] A. Wang *et al.*, "Wan: Open and advanced large-scale video generative models," *arXiv preprint arXiv:2503.20314*, 2025.

[5] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2023, pp. 4195–4205.

[6] W. Kong *et al.*, "Hunyuanvideo: A systematic framework for large video generative models," *arXiv preprint arXiv:2412.03603*, 2024.

[7] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, "Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps," in *Proc. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 5775–5787.

[8] V. A. Korthikanti, J. Casper, S. Lym, L. McAfee, M. Andersch, M. Shoeybi, and B. Catanzaro, "Reducing activation recomputation in large transformer models," *Proc. Mach. Learn. Syst.*, vol. 5, pp. 341–353, 2023.

[9] C. Meng, R. Rombach, R. Gao, D. Kingma, S. Ermon, J. Ho, and T. Salimans, "On distillation of guided diffusion models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 14 297–14 306.

[10] J. Zhang, J. Wei, P. Zhang, J. Zhu, and J. Chen, "Sageattention: Accurate 8-bit attention for plug-and-play inference acceleration," in *Proc. Int. Conf. Learn. Representations*, 2025.

[11] M. Xu, M. Zhu, Y. Liu, F. X. Lin, and X. Liu, "Deepcache: Principled cache for mobile deep vision," in *Proc. Annual Int. conf. Mobile Comput. and Network.*, 2018, pp. 129–144.

[12] X. Zhao, X. Jin, K. Wang, and Y. You, "Real-time video generation with pyramid attention broadcast," in *Proc. Int. Conf. Learn. Representations*, 2025.

[13] P. Chen, M. Shen, P. Ye, J. Cao, C. Tu, C.-S. Bouganis, Y. Zhao, and T. Chen, "δ-dit: A training-free acceleration method tailored for diffusion transformers," *arXiv preprint arXiv:2406.01125*, 2024.

[14] P. Selvaraju, T. Ding, T. Chen, I. Zharkov, and L. Liang, "Fora: Fast-forward caching in diffusion transformer acceleration," *arXiv preprint arXiv:2407.01425*, 2024.

[15] F. Liu, S. Zhang, X. Wang, Y. Wei, H. Qiu, Y. Zhao, Y. Zhang, Q. Ye, and F. Wan, "Timestep embedding tells: It's time to cache for video diffusion model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 7353–7363.

[16] H. Xi *et al.*, "Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity," in *Proc. Int. Conf. Mach. Learn.*, 2025.

[17] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Neural Inf. Process. Syst.*, 2020, pp. 6840–6851.

[18] C. Saharia *et al.*, "Photorealistic text-to-image diffusion models with deep language understanding," in *Proc. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 36 479–36 494.

[19] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10 684–10 695.

[20] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," in *Proc. Int. Conf. Learn. Representations*, 2024.

[21] Z. Zhao *et al.*, "Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation," *arXiv preprint arXiv:2501.12202*, 2025.

[22] W. Li *et al.*, "Step1x-3d: Towards high-fidelity and controllable generation of textured 3d assets," *arXiv preprint arXiv:2505.07747*, 2025.

[23] Z. Chen *et al.*, "3dtopia-xl: Scaling high-quality 3d asset generation via primitive diffusion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 26 576–26 586.

[24] D. Zhou, W. Wang, H. Yan, W. Lv, Y. Zhu, and J. Feng, "Magicvideo: Efficient video generation with latent diffusion models," *arXiv preprint arXiv:2211.11018*, 2022.

[25] A. Blattmann, R. Rombach, H. Ling, T. Dockhorn, S. W. Kim, S. Fidler, and K. Kreis, "Align your latents: High-resolution video synthesis with latent diffusion models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 22 563–22 575.

[26] A. Blattmann *et al.*, "Stable video diffusion: Scaling latent video diffusion models to large datasets," *arXiv preprint arXiv:2311.15127*, 2023.

[27] W. Hong, M. Ding, W. Zheng, X. Liu, and J. Tang, "Cogvideo: Large-scale pretraining for text-to-video generation via transformers," in *Proc. Int. Conf. Learn. Representations*, 2023.

[28] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proc. Int. Conf. Learn. Representations*, 2021.

[29] T. Karras, M. Aittala, T. Aila, and S. Laine, "Elucidating the design space of diffusion-based generative models," in *Proc. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 26 565–26 577.

[30] D. Bolya and J. Hoffman, "Token merging for fast stable diffusion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 4599–4603.

[31] H. Wang, D. Liu, Y. Kang, Y. Li, Z. Lin, N. K. Jha, and Y. Liu, "Attention-driven training-free efficiency enhancement of diffusion models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 16 080–16 089.

[32] E. Zhang, J. Tang, X. Ning, and L. Zhang, "Training-free and hardware-friendly acceleration for diffusion models via similarity-based token pruning," in *Proc. AAAI Conf. Artif. Intell.*, 2025.

[33] C. Zou, X. Liu, T. Liu, S. Huang, and L. Zhang, "Accelerating diffusion transformers with token-wise feature caching," in *Proc. Int. Conf. Learn. Representations*, 2025.

[34] F. Wimbauer *et al.*, "Cache me if you can: Accelerating diffusion models through block caching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 6211–6220.

[35] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proc. Int. Conf. Learn. Representations*, 2021.

[36] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *Proc. Int. Conf. Learn. Representations*, 2023.

[37] H. Liu, W. Zhang, J. Xie, F. Faccio, M. Xu, T. Xiang, M. Z. Shou, J.-M. Perez-Rua, and J. Schmidhuber, "Faster diffusion via temporal attention decomposition," *Trans. Mach. Learn. Research*, 2025.

[38] A. Wang *et al.*, "Wan: Open and advanced large-scale video generative models," *arXiv preprint arXiv:2503.20314*, 2025.

[39] B. F. Labs, "Flux," https://github.com/black-forest-labs/flux, 2024.

[40] Z. Huang *et al.*, "Vbench: Comprehensive benchmark suite for video generative models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 21 807–21 818.

[41] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[42] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 586–595.

[43] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.

[44] J. Hessel, A. Holtzman, M. Forbes, R. Le Bras, and Y. Choi, "Clipscore: A reference-free evaluation metric for image captioning," in *Proc. Conf. Empirical Methods in Natural Language Process.*, 2021, pp. 7514–7528.

[45] K. Sun, K. Huang, X. Liu, Y. Wu, Z. Xu, Z. Li, and X. Liu, "T2v-compbench: A comprehensive benchmark for compositional text-to-video generation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 8406–8416.