RETHINKING TIMESTEP-LEVEL CACHE IN DIFFUSION: A MINIMALIST FRAMEWORK FOR EFFECTIVE STEP SCHEDULING AND ERROR RECTIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Timestep-level caching has emerged as a prominent technique for accelerating diffusion transformers due to its simplicity and strong empirical performance. Although existing approaches are predominantly formulated as residual caching (i.e., model output minus input), we show through first-order Taylor expansion that these formulations are equivalent to direct output caching and can therefore be interpreted as enlarging the numerical integration step size. This interpretation recasts timestep-level caching as a numerical integration problem, where generation quality is determined by both when larger integration steps are taken and how the resulting truncation errors are controlled. Consequently, we identify two fundamental factors governing the effectiveness of timestep-level caching: the allocation of cached steps and the rectification of cache-induced errors. Motivated by these observations, we propose a minimalist yet highly effective acceleration framework that jointly addresses these two challenges. Specifically, we introduce a quality-driven scheduler to determine cache step allocation and a lightweight neural compensator to correct truncation errors along the denoising trajectory. Unlike existing approaches that rely on heuristic schedules and static linear corrections, our framework allocates cache steps according to their impact on final image quality and employs a neural compensator that better adapts to complex denoising trajectories, thereby achieving superior quality-acceleration trade-offs, all with minimal training overhead. Extensive experiments on FLUX.1-DEV demonstrate that our method substantially outperforms existing baselines, achieving an LPIPS of 0.1914 at 1024×1024 resolution while delivering a 2.77× inference speedup. As an architecture-agnostic framework, it further generalizes to video diffusion models, achieving an LPIPS of 0.0779 with a 2.91× speedup on COGVIDEOX-2B. Notably, across all evaluated models, the entire pre-processing stage requires less than 30 minutes, highlighting the efficiency and practicality of our approach.

1 INTRODUCTION

Diffusion models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020; Dhariwal & Nichol, 2021), particularly Diffusion Transformers (DiTs) (Peebles & Xie, 2023), have achieved unprecedented success in high-fidelity image (Rombach et al., 2022) and video (Blattmann et al., 2023) synthesis. However, their superior generation quality comes at the cost of substantial computation, as deep transformer networks must be repeatedly evaluated across dozens of denoising timesteps.

To mitigate this inference bottleneck, cache strategies have emerged by caching and reusing features with strong temporal redundancy across consecutive denoising timesteps. Existing methods exploit such redundancy at different granularities. Some perform fine-grained reuse of intermediate representations, such as layer-level caching in ΔDiT (Chen et al., 2024), L2C (Ma et al., 2024a) and token-level redundancy exploitation in ToCa (Zou et al., 2025), ClusCa (Zheng et al., 2025). While these methods often achieve favorable performance, they are typically difficult to optimize, computationally expensive, and highly dependent on specific model architectures. In contrast, timestep-level methods, such as TeaCache (Liu et al., 2025a), ERTACache (Peng et al., 2025) and SeaCache (Chung et al., 2026), adopt a much simpler strategy by reusing timestep-wise model features. Specifically, they often cache the model residual (i.e., output-input difference). These approaches are lightweight,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

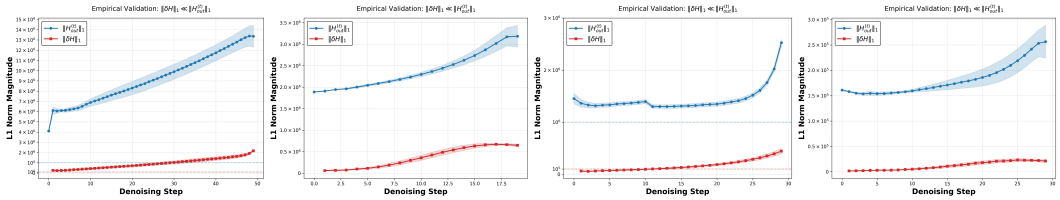

Figure 1: Empirical validation of feature scales across different DiTs (left to right: CogVideoX-2b, Latte-1, FLUX.1-dev, PixArt-XL-2-512×512). The norm of the input hidden states H_{in} is consistently negligible compared to the output H_{out}, with an average scale gap exceeding $10\times$.

easy to deploy, and introduce minimal optimization overhead. However, despite their efficiency, existing timestep-level methods still lack a clear understanding of the fundamental nature of caching. Moreover, most of them rely heavily on heuristic rules or manually designed thresholds, leading to compromised performance. We argue that timestep-level caching remains substantially underexplored. Its formulation is conceptually simple and does not require intrusive modifications to the model internals. More importantly, with proper optimization, timestep-level caching has the potential to outperform many sophisticated fine-grained caching strategies.

Revisiting Timestep-level Caching. In this work, we re-examine timestep-level caching from the perspective of first-order Taylor expansion. Let t denote the current denoising timestep, and Δt be the integration step size. We formalize the forward pass of a deep transformer network $\mathcal{F}$ as $H_{out}^{(t)} = \mathcal{F}(H_{in}^{(t)}, t)$, where $H_{in}^{(t)}$ and $H_{out}^{(t)}$ represent the high dimension latent input and output states of t step, respectively. To execute caching during inference, the objective is to approximate the output of the subsequent step $\tilde{H}_{out}^{(t-1)}$ without executing the complete network $\mathcal{F}$. Assuming the denoising trajectory governed by $\mathcal{F}$ is sufficiently smooth over local temporal intervals, the true state $H_{out}^{(t-1)}$ can be expanded around $(H_{in}^{(t)}, t)$ as follows:

$$H_{out}^{(t-1)} \approx H_{out}^{(t)} + \mathcal{J}_H \cdot (H_{in}^{(t-1)} - H_{in}^{(t)}) + \mathcal{J}_t \cdot \Delta t, \tag{1}$$

where $\mathcal{J}H = \frac{\partial \mathcal{F}}{\partial Hin}$ and $\mathcal{J}t = \frac{\partial \mathcal{F}}{\partial t}$ denote the input-feature Jacobian matrix and temporal derivative, respectively. Defining $\delta H = H_{in}^{(t-1)} - H_{in}^{(t)}$, the expression simplifies to:

$$H_{out}^{(t-1)} \approx H_{out}^{(t)} + \mathcal{J}_H \delta H + \mathcal{J}_t \Delta t. \tag{2}$$

To contextualize this theoretical framework, we analyze the prevalent residual caching formulations together with the simpler direct output caching formulation as examples.

- ▷ **Example 1: Residual Caching.** TeaCache, ERTACache and SeaCache are typical methods, which instead cache the feature increment $H_{out}^{(t)} - H_{in}^{(t)}$ at timestep t. By assuming that this increment is temporally invariant, the approximated state satisfies $\tilde{H}_{out}^{(t-1)} - H_{in}^{(t-1)} = H_{out}^{(t)} - H_{in}^{(t)}$, which yields $\tilde{H}_{out}^{(t-1)} = H_{out}^{(t)} + \delta H$. The resulting truncation error is thus $E_{res} \approx \|(\mathcal{J}_H - I)\delta H + \mathcal{J}_t \Delta t\|$, where I denotes the identity matrix. This strategy essentially relies on the assumption that $\mathcal{J}_H \approx I$ and $\mathcal{J}_t \approx 0$.
- ▷ **Example 2: Output Caching.** This formulation is more straightforward, directly caching the output $H_{out}^{(t)}$ at timestep t. Under the assumption that the output of the network remains constant across the local interval, the approximated state is formulated as $\tilde{H}_{out}^{(t-1)} = H_{out}^{(t)}$. Consequently, the truncation error is given by $E_{out} \approx \|\mathcal{J}_H \delta H + \mathcal{J}_t \Delta t\|$. This formulation implicitly presumes that the network satisfies $\mathcal{J}_H \approx 0$ and $\mathcal{J}_t \approx 0$.

Through extensive empirical investigation across multiple configurations, we observe a critical property: The magnitude of the variation term δH is consistently negligible compared to that of the output feature H_{out} (i.e., $\|\delta H\| \ll \|H_{\text{out}}\|$), indicating that δH contributes only a higher-order perturbation (see Figure 1). Under this observation, the higher-order terms in Equation 2 vanish in practice, revealing that residual caching formulations are equivalent to direct output caching. We further validate this conclusion empirically. Table S1 presents a comparison between residual and

output caching using the same schedule in ERTACache (Peng et al., 2025) on MSCOCO (Lin et al., 2014) with FLUX.1-DEV. The two variants share identical configurations, differing only in the caching formulation. Their generation quality is nearly indistinguishable, providing strong empirical evidence for the close correspondence predicted by our theoretical analysis.

Key Design Principles. Our analysis and empirical evidence reveal that timestep-level caching is fundamentally an output reuse mechanism and can therefore be interpreted as enlarging the numerical integration step size. From this perspective, timestep-level caching becomes a numerical integration problem, where generation quality is jointly determined by when larger integration steps are introduced and how the resulting truncation errors are controlled. Consequently, we identify two fundamental factors governing the effectiveness of timestep-level caching: **(1) cache step allocation along the denoising trajectory** and **(2) rectification of cache-induced truncation errors**.

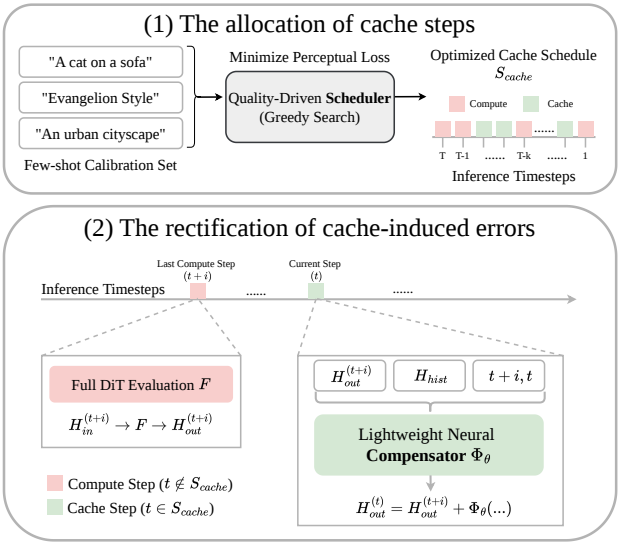

Figure 2: The overall architecture of our proposed framework, comprising the quality-driven Scheduler for step selection and the Compensator for dynamic trajectory alignment.

Existing methods address these two factors only heuristically. For step allocation, current methods typically adopt manually designed schedules (Peng et al., 2025) or empirical thresholds (Liu et al., 2025a), neither of which directly targets the final generation quality. For error rectification, existing methods mainly rely on simplified closed-form linear compensation (Peng et al., 2025), which are insufficient for modeling cache-induced errors along nonlinear denoising trajectories.

In this paper, we propose a minimalist yet highly effective cache framework. First, to address step allocation, we propose a quality-driven **Scheduler**, Operating on an extremely small calibration set, the scheduler employs a greedy search algorithm to directly minimize the final perceptual loss of the generated outputs under a given budget, converging rapidly to an effective schedule. Second, to address error rectification, we design a highly lightweight **Compensator**. This module is optimized to minimize the L_1 discrepancy between the cached representations and the actual current timestep features. The combined architecture is exceptionally lightweight and incurs minimal pre-processing overhead. It consistently outperforms previous state-of-the-art methods across multiple mainstream diffusion models and various acceleration ratios. These results provide strong empirical support for our theoretical perspective that the effectiveness of timestep-level caching is jointly governed by step allocation and cache-induced error rectification.

In summary, our main contributions are listed as follows:

- ▷ **Theoretical Analysis:** We provide a first-order Taylor expansion analysis revealing that previous residual caching are equivalent to direct output caching and can thus be viewed as enlarging the numerical integration step size.
- ▷ **Minimalist Framework:** We show that the effectiveness of timestep-level caching is fundamentally governed by two factors: cache step allocation and cache-induced error rectification. Motivated by this, we propose a new framework including a scheduler and a compensator, leading to substantial improvements over existing methods.
- ▷ **Experimental Validation:** Extensive experiments across multiple mainstream image and video diffusion models demonstrate state-of-the-art acceleration performance and strong generalization capability. Notably, our framework requires only a remarkably lightweight training process, incurring minimal data and computational overhead.

2 Related Works

Cache-based acceleration leverages temporal redundancy across denoising steps to reduce computation without altering the model backbone. Existing methods can be generally categorized by caching granularity, including block-/layer-level caching and timestep-level caching. **Block-/layer-level** methods reuse intermediate hidden representations within the network (DeepCache (Ma et al., 2024b), FORA (Selvaraju et al., 2024) and ΔDiT (Chen et al., 2024)). These methods cache and reuse fine-grained internal features to mitigate cache-induced errors during diffusion. In contrast, **timestep-level** methods operate at a coarser granularity by directly reusing timestep-wise predictions, residuals, or trajectory information across neighboring denoising steps. Representative examples include ERTACache (Peng et al., 2025), SeaCache (Chung et al., 2026), and EasyCache (Zhou et al., 2025). Due to their architecture-agnostic design, these methods can be readily applied to a wide range of diffusion backbones. Our proposed method also belongs to this category.

Beyond cache-based acceleration, various orthogonal methods for efficient diffusion have been studied, including pruning, quantization, distillation, and solver techniques. Details in Appendix A.

3 Methodology

In this section, we detail the proposed minimalist framework for diffusion acceleration. As discussed earlier, the efficacy of timestep-level caching is fundamentally constrained by two primary factors: the allocation of cache steps across the denoising trajectory, and the rectification of cache-induced truncation errors. The core objective of this work is not to engineer an overly complex architectural intervention, but rather to empirically validate that lightweight optimizations on these two dimensions can unlock the potential of timestep-level caching. To this end, we propose two simple modules: a quality-driven **scheduler** and a neural **compensator**.

3.1 Quality-Driven Step Allocation

Existing methods typically rely on heuristic schedules or thresholds derived from adjacent feature differences. However, intermediate feature variations exhibit a weak correlation with the final perceptual quality. To strictly align the caching schedule with optimal visual fidelity, we formulate step allocation as a discrete optimization problem and solve it via a quality-driven greedy search.

Operating on an extremely small calibration set, the scheduler systematically identifies the sequence of cache steps that yields the lowest error, thereby achieving the highest perceptual quality in the final outputs. Let N_{target} denote the target number of cached steps (i.e., total steps minus computed steps). The algorithm initializes an empty cache schedule S_{cache} and an available candidate pool S_{avail}. During each optimization round, the algorithm independently evaluates the addition of each candidate $c \in S_{avail}$ into the current schedule. This is achieved by executing the full diffusion generation process and computing the Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018) metric against the un-cached ground-truth trajectory.

The candidates are subsequently ranked in ascending order of their resulting LPIPS scores. To ensure temporal diversity and prevent the schedule from collapsing into dense, adjacent clusters, the algorithm selects the top-K candidates under the strict constraint that no two selected candidates within the same round are temporally adjacent. The selected candidates are appended to S_{cache} and removed from S_{avail}. This iterative process terminates once the schedule length reaches N_{target}. The complete procedure is detailed by Algorithm 1 in Appendix B.

3.2 Lightweight Neural Compensator

Direct output caching assumes that the cached output remains sufficiently accurate over multiple subsequent timesteps. However, this approximation error grows as the cache interval increases, leading to noticeable performance degradation at high acceleration ratios. Existing linear compensation strategies can partially reduce this error (for example, closed-form compensation), but their capacity is limited when the underlying denoising dynamics become highly non-linear. To better rectify the accumulated deviation, we introduce a lightweight neural compensator, denoted as Φ_θ.

Rather than directly predicting the target state $H_{out}^{(t)}$, the compensator is designed to estimate the difference between the target state and the most recently computed non-cached state. Specifically, it learns $H_{out}^{(t)} - H_{out}^{(t+i)}$, where $H_{out}^{(t+i)}$ denotes the output feature from the nearest preceding timestep at which a full model evaluation is performed. This difference formulation substantially simplifies the learning objective by focusing on the cache-induced deviation rather than the full feature representation. The corrected output is computed as:

$$H_{out}^{(t)} = H_{out}^{(t+i)} + \Phi_\theta\Big(H_{out}^{(t+i)}, H_{hist}, t+i, t\Big), \tag{3}$$

where H_{hist} represents a collection of hidden states from several previous timesteps, providing temporal context for estimating the evolution of the denoising trajectory.

The architecture of Φ_θ is intentionally lightweight to ensure negligible additional inference overhead. To capture the temporal evolution between the cached state and the target timestep, temporal information is encoded through sinusoidal embeddings of both $t+i$ and t. Spatial and channel-wise correlations are then modeled by a *LocalAdaptiveBlock*, which combines a depthwise convolution with a Squeeze-and-Excitation (SE) attention mechanism. The resulting features are further modulated through Adaptive Layer Normalization (AdaLN) conditioned on the temporal embeddings, enabling timestep-aware correction. Finally, a SwiGLU transformation and a final projection generate the difference compensation in the original hidden space.

The compensator is optimized using a time-normalized Smooth L1 objective to ensure stable gradients across highly variable timestep intervals. The loss function is defined as:

$$\mathcal{L} = \frac{1}{i}\mathrm{L1}\left(\Phi_\theta(H_{out}^{(t+i)}, H_{hist}, t+i, t), H_{out}^{(t)} - H_{out}^{(t+i)}\right). \tag{4}$$

Training the compensator requires only a small amount of offline-generated calibration data and can be completed rapidly. Combined with the lightweight architecture and difference learning formulation, the compensator effectively corrects cache-induced truncation errors while maintaining strong generalization under different acceleration settings.

4 Experiments

4.1 Image Synthesis Experiments

Implementation Settings. To optimize the cache scheduler and compensator, we construct a synthetic prompt set consisting of 2,000 captions generated by ChatGPT-5.5 (details in Appendix D). We intentionally employ synthetic prompts rather than real datasets to reduce the risk of overfitting to a particular data distribution. This design encourages the error compensator to learn a more transferable correction mechanism, enabling robust error correction and consistent performance under data distribution shifts. For training, we use a mixed-resolution dataset consisting of 30% samples at 256 and 70% samples at 512. This design is essential for achieving robust cross-resolution generalization: models trained on a single resolution exhibit limited transferability to unseen resolutions, whereas mixed-resolution substantially improves out-of-distribution performance. Unless otherwise specified, all methods are evaluated on 1,000 images sampled from `MSCOCO-val-2014` (Lin et al., 2014) dataset. All experiments are conducted on a single NVIDIA GeForce RTX 4090 48GB GPU.

Discussion. Qualitative visualization examples are shown in Figure 3, with additional results provided in Appendix G. The quantitative results in Table 1a further reveal several important observations regarding the effectiveness and robustness of the proposed framework.

The proposed method consistently surpasses recent state-of-the-art caching approaches, including the timestep-level method ERTACache (Peng et al., 2025), SeaCache (Chung et al., 2026) and the block-level method TaylorSeer (Liu et al., 2025c). For instance, at the 1024×1024 resolution with a comparable $\sim 2.78\times$ inference speedup, our method (G3-top3-10++) achieves an LPIPS of 0.1914, establishing a substantial quantitative margin over ERTACache (0.2725) and TaylorSeer (0.3230). Compared with SeaCache, our method also achieves a significantly higher SSIM of 0.8312 versus 0.8175 and lower LPIPS of 0.1914 versus 0.2084. These substantial performance gains further validate our analysis that the effectiveness of timestep-level caching fundamentally depends on both cache-step allocation and the rectification of cache-induced errors. More importantly, it reveals that

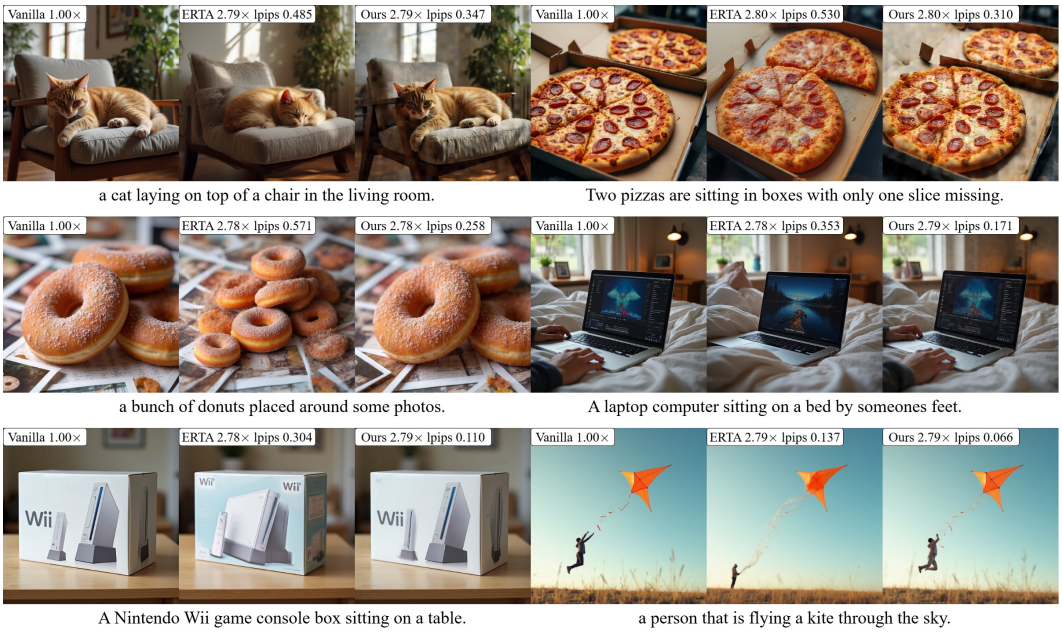

Figure 3: Visualization examples of FLUX.1-DEV at resolutions of 512, 768, and 1024 (from top to bottom). Compared with ERTACache (Peng et al., 2025), the proposed method more faithfully preserves the visual characteristics of the original outputs across different resolutions, yielding results that remain consistently closer to the reference generations.

the capability of timestep-level caching remains largely underexplored; when properly exploited, it can even surpass more complex block-level caching strategies like TaylorSeer.

When comparing the baseline version of our method (pure scheduler) with its compensator-enhanced variant (denoted by "++") clears the contributions of the two proposed components. The results show that incorporating the lightweight compensator consistently yields further improvements across all evaluation metrics. (e.g., SSIM from 0.8084 to 0.8312 and LPIPS from 0.2207 to 0.1914 at 1024×1024). This indicates that the compensator effectively mitigates the deviations along the non-linear denoising trajectory without introducing prohibitive computational latency.

Crucially, the empirical data demonstrates the strong out-of-distribution (OOD) generalization capabilities of the trained compensator. While it was optimized exclusively on a mixed-resolution dataset comprising 256 and 512 spatial dimensions, it maintains robust error-rectification performance on unseen high-resolution generations of 768 and 1024. At these OOD resolutions, the "++" variant continues to yield a clear statistical advantage over the base schedule.

Finally, our framework exhibits a more stable degradation curve under extreme acceleration regimes compared to existing approaches. As the computational budget is further constrained (from $t = 10$ down to $t = 7$ at 1024×1024), our method manages to deliver a $3.78\times$ speedup with an LPIPS of 0.3204. In contrast, TaylorSeer (N=7, O=1) deteriorates significantly to an LPIPS of 0.5488 at a lower $3.63\times$ speedup. While performance inevitably declines at highly sparse numerical integrations, our method provides a more reliable quality-acceleration trade-off for practical deployment.

4.2 Video Synthesis Experiments

Model-agnosticism and performance. A fundamental advantage of our method is its model-agnostic design. Because the method optimizes timestep scheduling and error rectification rather than relying on specific structural priors, it can be naturally extended to different models and modalities with minimal code modifications. We validate this adaptability by applying the framework to video synthesis using CogVideoX-2b (Yang et al., 2025) and OpenSora-1.2 (Zheng et al., 2024). Qualitative results are shown in Figure 4, additional visualization examples can be found in Appendix H. As shown in Table 1b and 1c, our method consistently outperforms existing accel-

Table 1: Quantitative results on FLUX.1-DEV, COGVIDEOX-2B, and OPENSORA-1.2. For FLUX.1-DEV, results are evaluated on 1,000 MSCOCO samples at resolutions of 256×256, 512×512, 768×768, and 1024×1024. For COGVIDEOX-2B, results are evaluated using VBench on 48-frame videos at 480P (480×720). For OPENSORA-1.2, results are evaluated using VBench on 51-frame videos at 480P (480×848). Notation Gn-top$k$-t denotes greedy search with n calibration samples, retaining the top-k candidate timesteps at each iteration and targeting a t-step computation. "++" denotes inference with the compensator. Rows highlighted in gray indicate our proposed method. Bold indicates the best within each group. "Lat" denotes latency, measured in seconds.

(a) FLUX.1-DEV

Settings	PSNR ↑	SSIM ↑	LPIPS ↓	Speedup ↑	Lat ↓
Resolution: 256 × 256					
Vanilla (30 steps)	–	–	–	1.00×	3.28
ERTACache	19.49	0.7089	0.2135	2.78×	1.18
Ours (G3-top3-10)	22.03	0.7703	0.1549	2.78×	1.18
Ours (G3-top3-10++)	**22.83**	**0.8073**	**0.1284**	2.78×	1.18
Resolution: 512 × 512					
Vanilla (30 steps)	–	–	–	1.00×	5.82
ERTACache	19.95	0.7480	0.2406	2.76×	2.11
Ours (G3-top3-10)	22.39	0.7998	0.1825	2.76×	2.11
Ours (G3-top3-10++)	**23.10**	**0.8273**	**0.1536**	2.76×	2.11
Resolution: 768 × 768 (OOD Resolution)					
Vanilla (30 steps)	–	–	–	1.00×	11.52
ERTACache	20.92	0.7711	0.2524	2.76×	4.17
Ours (G3-top3-10)	23.13	0.8186	0.1951	2.76×	4.17
Ours (G3-top3-10++)	**23.58**	**0.8369**	**0.1709**	2.76×	4.18
Resolution: 1024 × 1024 (OOD Resolution)					
Vanilla (30 steps)	–	–	–	1.00×	19.53
Vanilla (10 steps)	15.88	0.6628	0.4251	2.79×	7.00
UniformCache (I=3)	15.89	0.6591	0.4270	2.79×	7.00
ERTACache	20.50	0.7700	0.2725	2.79×	6.99
TaylorSeer (N=4,O=1)	17.91	0.7106	0.3230	2.72×	7.18
SeaCache (δ=0.6)	22.82	0.8175	0.2084	2.75×	7.10
Ours (G3-top3-10)	22.63	0.8084	0.2207	2.79×	7.01
Ours (G3-top3-10++)	**23.22**	**0.8312**	**0.1914**	2.77×	7.05
TaylorSeer (N=5,O=1)	16.69	0.6854	0.3660	2.98×	6.56
Ours (G3-top3-9)	21.77	0.7922	0.2437	3.06×	6.39
Ours (G3-top3-9++)	**22.17**	**0.8096**	**0.2185**	3.04×	6.42
TaylorSeer (N=6,O=1)	15.72	0.6408	0.4358	3.27×	5.97
Ours (G3-top3-8)	21.42	0.7581	0.2817	3.38×	5.78
Ours (G3-top3-8++)	**21.78**	**0.7758**	**0.2690**	3.37×	5.81
TaylorSeer (N=7,O=1)	14.85	0.5532	0.5488	3.63×	5.38
Ours (G3-top3-7)	19.98	0.7229	0.3329	3.80×	5.14
Ours (G3-top3-7++)	**20.25**	**0.7373**	**0.3204**	3.78×	5.17

(b) COGVIDEOX-2B

Settings	VBench ↑	PSNR ↑	SSIM ↑	LPIPS ↓	Speed ↑	Lat ↓
Vanilla (50 steps)	80.49%	–	–	–	1.00×	78.67
ΔDiT (N_c=4)	79.61%	17.93	0.6612	0.3319	1.08×	–
ΔDiT (N_c=8)	79.31%	16.69	0.6277	0.3822	1.15×	–
ΔDiT (N_c=12)	79.09%	16.15	0.6126	0.4053	1.26×	–
PAB	79.76%	28.04	0.8978	0.0860	1.35×	–
FasterCache	79.83%	28.93	0.9066	0.0766	1.62×	–
ERTACache (λ=0.08)	79.30%	32.77	0.9394	0.0368	1.62×	–
Ours (G1-top5-30)	80.13%	36.04	0.9599	0.0194	1.66×	47.25
Ours (G1-top5-30++)	**80.16%**	**36.41**	**0.9620**	**0.0185**	1.65×	47.74
Vanilla (17 steps)	73.69%	14.24	0.5754	0.4796	2.91×	27.05
UniformCache (I=3)	73.36%	13.95	0.5655	0.4865	2.92×	27.00
TeaCache	79.00%	20.97	0.7614	0.2057	2.92×	–
ERTACache (λ=0.3)	78.79%	26.44	0.8702	0.1012	2.93×	–
Ours (G1-top5-17++)	**79.28%**	**29.81**	**0.8906**	**0.0779**	2.91×	27.05
Ours (G1-top5-12++)	79.17%	22.19	0.7761	0.1845	3.73×	21.07

(c) OPENSORA-1.2

Method	VBench ↑	PSNR ↑	SSIM ↑	LPIPS ↓	Speed ↑	Lat ↓
Vanilla (30 steps)	79.40%	–	–	–	1.00×	52.28
T-GATE	75.42%	19.55	0.6927	0.2612	1.13×	–
PAB	76.95%	23.58	0.8220	0.1743	1.42×	–
TeaCache	79.27%	23.56	0.8433	0.1318	1.55×	–
Ours (G1-top3-18)	79.74%	32.11	0.9198	0.0465	1.65×	31.66
Ours (G1-top3-18++)	**79.83%**	**32.18**	**0.9217**	**0.0424**	1.64×	31.96
Vanilla (15 steps)	77.36%	15.82	0.6961	0.3363	1.98×	–
Random 0.5	76.78%	16.51	0.7037	0.3264	1.98×	–
Static Cache	77.37%	15.73	0.6961	0.3382	1.86×	–
EasyCache	78.74%	23.95	0.8556	0.1235	2.12×	–
Ours (G2-top3-14)	79.13%	28.39	0.8886	0.0786	2.11×	24.78
Ours (G2-top3-14++)	**79.44%**	**28.43**	**0.8927**	**0.0696**	2.11×	24.82
Ours (G2-top3-12)	78.41%	26.45	0.8510	0.1270	2.49×	20.98
Ours (G2-top3-12++)	78.94%	25.89	0.8543	0.1081	2.48×	21.11
Ours (G1-top3-10)	78.24%	24.60	0.8280	0.1469	2.96×	17.64
Ours (G1-top3-10++)	78.37%	25.02	0.8259	0.1320	2.96×	17.66

eration baselines. For methods whose latency is reported as "–", the quantitative results are taken from the ERTACache paper. Since the hardware configuration differs only in GPU, absolute latency values are not directly comparable. Therefore, we compare methods primarily in terms of speedup.

For COGVIDEOX-2B, we compare against representative acceleration methods, including ΔDiT (Chen et al., 2024), PAB (Zhao et al., 2025), FasterCache (Lv et al., 2025), TeaCache (Liu et al., 2025a) and ERTACache (Peng et al., 2025), covering both timestep-level and block-level caching paradigms. Notably, under the G1-top5-17++ configuration, our method achieves a 2.91× speedup while maintaining an LPIPS of only 0.0779, establishing a clear performance margin over both ERTACache (0.1012) and TeaCache (0.2057) at comparable acceleration levels.

On OPENSORA-1.2, the framework similarly maintains higher structural similarity and lower perceptual degradation (e.g., an LPIPS of 0.1320 at a 2.96× speedup) compared to recent video caching methods like T-GATE (Liu et al., 2025b) and EasyCache (Zhou et al., 2025).

Pre-processing cost. The performance gains achieved by our method do not introduce prohibitive preparation costs. Table S2 details the time required for the two stages: the greedy search for cache step allocation and the training of the compensator. Across the three primary experiments, the total

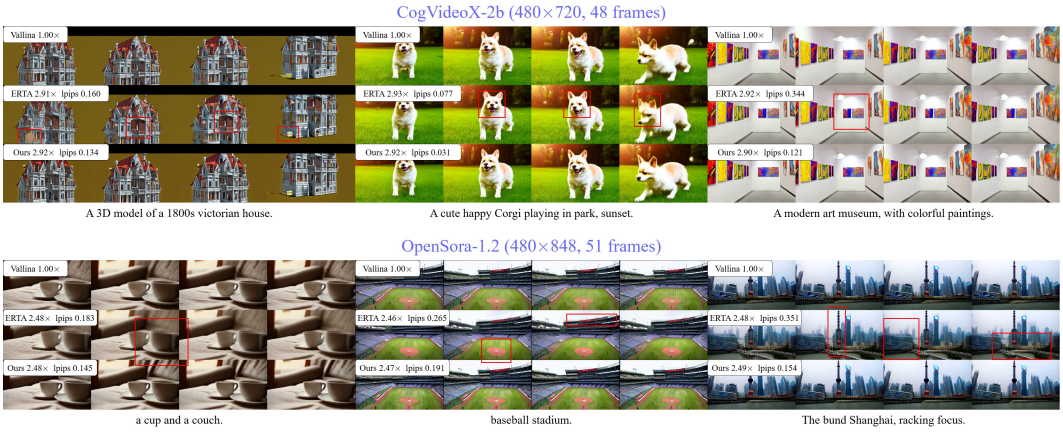

Figure 4: Qualitative comparison on CogVideoX-2b and OpenSora-1.2. Both ERTA-Cache (Peng et al., 2025) and our method are compared against the corresponding ground-truth generations (Vanilla). Our method consistently reconstructs details more faithfully, consistent with its superior LPIPS performance. The corresponding prompt for each example is shown below.

time remains strictly bounded to **below 30 minutes** on a single RTX 4090 GPU. Furthermore, the trained compensator introduce negligible memory footprints, ranging from 4.3 MB to 12.2 MB. These metrics indicate that the proposed framework is highly efficient, enabling rapid deployment without significant computational or storage burdens.

4.3 Ablation Studies

We conduct extensive ablation studies to analyze the key design choices of the proposed framework. Since the effectiveness of timestep-level caching is fundamentally determined by cache scheduling and error compensation, we organize our analysis around these two components.

Scheduler. We investigate the following factors:

- ▷ **Calibration set size (N).** We show that a small calibration set is sufficient to reliably identify the timestep-wise quality landscape, while substantially larger sets introduce considerable preprocessing overhead without consistent quality gains. Details in Appendix E.1.
- ▷ **Greedy search budget (k).** Moderate search budgets achieve the most favorable balance between optimization efficiency and generation quality. Details in Appendix E.2.
- ▷ **Adjacent-step constraint.** The adjacent-step constraint is important for effective timestep allocation, substantially improving the resulting schedule. Details in Appendix E.3.

Compensator. We analyze the following design choices:

- ▷ **Difference prediction vs. direct prediction.** We compare predicting the difference against directly predicting the target vector, analyzed in the following subsection.
- ▷ **Architectural components (module drop-out ablation).** We show that temporal conditioning and global feature aggregation are important for accurate error compensation, while the full architecture provides the most reliable correction. Details in Appendix F.1.
- ▷ **Cross-resolution training mix.** We demonstrate that single-resolution training suffers from severe cross-resolution degradation, whereas mixed-resolution supervision substantially improves generalization and enables robust zero-shot extrapolation to unseen resolutions. Detailed analysis in Appendix F.2.
- ▷ **Historical context order.** We find that anchoring historical context to true cached states consistently improves both training efficiency and compensation fidelity, while a moderate temporal order provides the most effective local trajectory information. See Appendix F.3.

4.3.1 ABLATION ON COMPENSATOR PREDICTION TARGET

Table 2: Quantitative comparison of compensator prediction targets. Evaluated on 200 samples from `MSCOCO-val-2014` (Lin et al., 2014) dataset at 512×512 resolution.

Prediction Target	PSNR ↑	SSIM ↑	LPIPS ↓
Baseline (No Compensator)	21.98	0.7948	0.1874
Direct Prediction	21.21	0.7691	0.2052
Difference Prediction (Ours)	**22.17**	**0.8005**	**0.1761**

We investigate the learning objective of the compensator by comparing *direct prediction* with *difference prediction*. In the direct prediction paradigm, the compensator is tasked with predicting the complete target hidden state $H_{out}^{(t)}$ directly, whereas the difference prediction formulates the objective as estimating the incremental deviation $H_{out}^{(t)} - H_{out}^{(t+i)}$ from the most recently computed state.

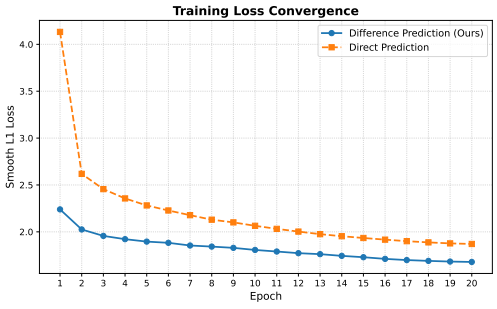

Figure 5: Comparison of training loss curves under the difference prediction and direct prediction settings.

As demonstrated in Table 2, direct prediction severely degrades generation quality, yielding an LPIPS of 0.2052, which is worse than the uncompensated baseline. This performance collapse occurs because reconstructing the entire high-dimensional feature manifold from scratch represents an excessively complex mapping for a lightweight MLP. The network struggles to capture the high-frequency spatial details inherent in $H_{out}^{(t)}$, thereby introducing destructive errors into the trajectory instead of correcting them. In contrast, difference prediction leverages the strong temporal correlation between consecutive timesteps. By casting the task as a difference learning problem, the network only needs to model the local curvature of the trajectory. The variation exhibits a lower magnitude compared to the absolute state, making the optimization significantly easier. This theoretical advantage is empirically validated by the training dynamics (see Figure 5): the difference prediction formulation initializes at a much lower loss (2.23 vs. 4.13) and converges to a superior local minimum.

5 LIMITATIONS

While the proposed framework achieves state-of-the-art performance for timestep-level cache acceleration, several limitations remain and warrant further investigation. The primary intention of this work is not to engineer the step allocation and error rectification modules to their absolute theoretical limits. Rather, our goal is to empirically validate that the efficacy of timestep-level caching is fundamentally governed by these two dimensions. Consequently, the mechanisms we employed—a greedy search algorithm for scheduling and a lightweight MLP-based network for compensation—are intentionally minimalist. More advanced scheduling strategies and compensation architectures may further improve performance, which we leave for future investigation.

6 CONCLUSION

In this paper, we presented a minimalist yet highly effective framework for accelerating diffusion transformers. By re-examining timestep-level caching through Taylor expansion, we revealed that its performance hinges on cache step allocation and cache-induced errors rectification. Motivated by this analysis, we introduced a quality-driven scheduler to directly optimize visual fidelity under given computational budgets, alongside a lightweight neural compensator to dynamically align non-linear denoising trajectories. Extensive experiments across multiple image and video diffusion models demonstrated that our approach significantly outperforms existing baselines, achieving superior quality-acceleration trade-offs with negligible pre-processing overhead.

AI USE STATEMENT

Generative AI was used solely for language editing, including improving grammar, wording, and readability of the manuscript. They were not used for research idea, methodology design, experiment design, implementation, data analysis, and so on. All AI-assisted edits were reviewed and verified by the authors. The authors take full responsibility for the final content of this work.

ETHICS STATEMENT

This work presents a cache-based inference acceleration method for diffusion models, with the primary goal of reducing computational cost and improving inference efficiency. The proposed acceleration technique does not introduce additional capabilities beyond those of the underlying diffusion models. By reducing redundant model computations, it may also reduce the computational resources and energy consumption required for diffusion inference. We are not aware of any specific ethical concerns arising from the proposed method or its experimental use.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the model configuration, implementation settings, generation protocol and ablation settings in the main paper and appendix. Code will be made available during the review process and will be publicly released upon acceptance.

REFERENCES

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets. *CoRR*, abs/2311.15127, 2023.

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. *CoRR*, abs/2406.01125, 2024.

Jiwoo Chung, Sangeek Hyun, MinKyu Lee, Byeongju Han, Geonho Cha, Dongyoon Wee, Youngjun Hong, and Jae-Pil Heo. Seacache: Spectral-evolution-aware cache for accelerating diffusion models. *CoRR*, abs/2602.18993, 2026. doi: 10.48550/ARXIV.2602.18993.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 8780–8794, 2021.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Yuchao Gu, Guian Fang, Yuxin Jiang, Weijia Mao, Song Han, Han Cai, and Mike Zheng Shou. Anyflow: Any-step video diffusion model with on-policy flow map distillation. *CoRR*, abs/2605.13724, 2026. doi: 10.48550/ARXIV.2605.13724.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Vbench: Comprehensive benchmark suite for video generative

models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pp. 21807–21818. IEEE, 2024. doi: 10.1109/CVPR52733.2024.02060.

Alexia Jolicoeur-Martineau, Ke Li, Rémi Piché-Taillefer, Tal Kachman, and Ioannis Mitliagkas. Gotta go fast when generating data with score-based models. *CoRR*, abs/2105.14080, 2021.

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. *CoRR*, abs/2411.02397, 2024.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

Senmao Li, Taihang Hu, Joost van de Weijer, Fahad Shahbaz Khan, Tao Liu, Linxuan Li, Shiqi Yang, Yaxing Wang, Ming-Ming Cheng, and Jian Yang. Faster diffusion: Rethinking the role of the encoder for diffusion model inference. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

Yanjing Li, Sheng Xu, Xianbin Cao, Xiao Sun, and Baochang Zhang. Q-DM: an efficient low-bit quantized diffusion model. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pp. 740–755. Springer, 2014. doi: 10.1007/978-3-319-10602-1_48.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025, Nashville, TN, USA, June 11-15, 2025*, pp. 7353–7363. Computer Vision Foundation / IEEE, 2025a.

Haozhe Liu, Wentian Zhang, Jinheng Xie, Francesco Faccio, Mengmeng Xu, Tao Xiang, Mike Zheng Shou, Juan-Manuel Perez-Rua, and Jürgen Schmidhuber. Faster diffusion via temporal attention decomposition, 2025b.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with taylorseers. *CoRR*, abs/2503.06923, 2025c.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *CoRR*, abs/2101.02388, 2021.

Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K. Wong. Fastercache: Training-free video diffusion model acceleration with high quality. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024a.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pp. 15762–15772. IEEE, 2024b. doi: 10.1109/CVPR52733.2024.01492.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pp. 4172–4182. IEEE, 2023.

Xurui Peng, Hong Liu, Chenqian Yan, Rui Ma, Fangmin Chen, Xing Wang, Zhihua Wu, Songwei Liu, and Mingbao Lin. Ertacache: Error rectification and timesteps adjustment for efficient diffusion. *CoRR*, abs/2508.21091, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 10674–10685. IEEE, 2022.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. *CoRR*, abs/2311.17042, 2023.

Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. FORA: fast-forward caching in diffusion transformer acceleration. *CoRR*, abs/2407.01425, 2024. doi: 10.48550/ARXIV.2407.01425.

Junhyuk So, Jungwon Lee, Daehyun Ahn, Hyungjun Kim, and Eunhyeok Park. Temporal dynamic quantization for diffusion models. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis R. Bach and David M. Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 2256–2265. JMLR.org, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 11895–11907, 2019.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. Cogvideox: Text-to-video diffusion models with an expert transformer. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation. *Trans. Mach. Learn. Res.*, 2022, 2022.

Zhihang Yuan, Hanling Zhang, Lu Pu, Xuefei Ning, Linfeng Zhang, Tianchen Zhao, Shengen Yan, Guohao Dai, and Yu Wang. Ditfastattn: Attention compression for diffusion transformer models. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024.

Dingkun Zhang, Sijia Li, Chen Chen, Qingsong Xie, and Haonan Lu. Laptop-diff: Layer pruning and normalized distillation for compressing diffusion models. *CoRR*, abs/2404.11098, 2024.

Qinsheng Zhang, Molei Tao, and Yongxin Chen. gddim: Generalized denoising diffusion implicit models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 586–595. Computer Vision Foundation / IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00068.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all. *CoRR*, abs/2412.20404, 2024. doi: 10.48550/ARXIV.2412.20404.

Zhixin Zheng, Xinyu Wang, Chang Zou, Shaobo Wang, and Linfeng Zhang. Compute only 16 tokens in one timestep: Accelerating diffusion transformers with cluster-driven feature caching. *CoRR*, abs/2509.10312, 2025.

Xin Zhou, Dingkang Liang, Kaijin Chen, Tianrui Feng, Xiwu Chen, Hongkai Lin, Yikang Ding, Feiyang Tan, Hengshuang Zhao, and Xiang Bai. Less is enough: Training-free video diffusion acceleration via runtime-adaptive caching. *CoRR*, abs/2507.02860, 2025.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

Appendix

Appendix Overview. This appendix is organized as follows:

- ▷ Section A presents additional related work beyond the main paper.
- ▷ Section B provides the detailed greedy search procedure for step allocation.
- ▷ Section C presents additional experimental results.
- ▷ Section D details the synthetic dataset construction.
- ▷ Section E provides additional ablation studies on the scheduler.
- ▷ Section F provides additional ablation studies on the compensator.
- ▷ Section G and Section H provides additional image and video comparison examples.

A Comprehensive Related Works

Orthogonal acceleration methods. A large body of work has investigated accelerating diffusion inference by improving either the model or the sampling process. Including **pruning** (Fang et al., 2023; Zhang et al., 2024), **quantization** (So et al., 2023; Li et al., 2023), and **distillation** (Luhman & Luhman, 2021; Salimans & Ho, 2022; Sauer et al., 2023; Gu et al., 2026), which aim to reduce model complexity while preserving generation quality. In parallel, a series of advances in **sampling solvers** (Jolicoeur-Martineau et al., 2021; Song et al., 2021; Liu et al., 2022; Zhang et al., 2023) have further improved efficiency by reducing the number of denoising steps.

Cache-based Acceleration. Cache-based acceleration exploits temporal redundancy during the diffusion inference process to reduce computational overhead without altering the backbone model. Existing methods can be systematically categorized into block-/layer-level caching and timestep-level caching based on the granularity of feature reuse.

Block-/layer-level methods focus on reusing intermediate hidden representations within the network architecture. Early approaches, such as DeepCache (Ma et al., 2024b) and FORA (Selvaraju et al., 2024), perform periodic feature reuse at fixed intervals for U-Net and DiT, respectively. To address the resulting errors, subsequent works introduce incremental caching (ΔDiT (Chen et al., 2024)). Other studies exploit specific architectural redundancies, including reusing U-Net encoder features (Li et al., 2024), broadcasting attention outputs (Zhao et al., 2025), and caching window attention (Yuan et al., 2024). Optimization at this level has also been extended to token-cache (Zou et al., 2025; Zheng et al., 2025) and learning-based routing (Ma et al., 2024a). In contrast, **timestep-level** methods directly reusing timestep-wise features, offering an architecture-agnostic solution. Rather than relying on static intervals, recent studies in this category increasingly adopt dynamic scheduling. For instance, TeaCache (Liu et al., 2025a) dynamically determines computation reuse through manually designed thresholds and probe mechanisms, respectively. In the video modality, AdaCache (Kahatapitiya et al., 2024) and EasyCache (Zhou et al., 2025) adapt caching policies according to motion complexity. Beyond direct reuse, several methods incorporate error rectification (ERTACache (Peng et al., 2025)) or feature forecasting techniques (TaylorSeer (Liu et al., 2025c)).

Our proposed method aligns with the timestep-level paradigm. Through two concise optimization strategies, our approach achieves competitive acceleration performance with minimal implementation complexity, demonstrating measurable empirical advantages over existing timestep-level methods as well as multiple block-/layer-level techniques.

B Detailed algorithmic procedure

As described in Section 3.1, our scheduler determines the cache schedule by directly optimizing final perceptual quality, rather than relying on heuristic proxies such as adjacent feature differences. We formulate step allocation as a discrete optimization problem and solve it using a quality-driven greedy search. The complete procedure is summarized in Algorithm 1.

Algorithm 1 Greedy Search Algorithm for Step Allocation

Input: Target cache size N_{target}, initial available steps S_{avail}, calibration set, top-K.
Output: Optimal cache schedule S_{cache}, with $|S_{cache}| = N_{target}$.

Initialize $S_{cache} \leftarrow \emptyset$
Generate ground-truth un-cached tensors for all calibration set
while $|S_{cache}| < N_{target}$ and $S_{avail} \neq \emptyset$ **do**
  $K_{current} \leftarrow \min(K, N_{target} - |S_{cache}|)$; $Scores \leftarrow \emptyset$
  for each candidate $c \in S_{avail}$ **do**
    Evaluate diffusion with schedule $S_{cache} \cup \{c\}$
    Compute LPIPS against ground-truth
    $Scores \leftarrow Scores \cup \{(c, \text{LPIPS})\}$
  end for
  Sort $Scores$ in ascending order of LPIPS
  $Selected \leftarrow \emptyset$
  for each (c, LPIPS) in sorted $Scores$ **do**
    if $|Selected| = K_{current}$ **then**
      break
    end if
    if $\forall s \in Selected, |c - s| > 1$ **then**
      $Selected \leftarrow Selected \cup \{c\}$
    end if
  end for
  $S_{cache} \leftarrow S_{cache} \cup Selected$
  $S_{avail} \leftarrow S_{avail} \setminus Selected$
end while
return Sort(S_{cache})

C Additional Experimental Results

Table S1: Comparison of output and residual caching under the same ERTACache schedule. The two variants yield nearly identical results, with only negligible differences across all metrics. This empirically supports the motivation in Section 1: timestep-level caching is fundamentally an output reuse mechanism and can therefore be interpreted as enlarging the numerical integration step size.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
Output Cache	20.0588	0.7510	0.2418
Residual Cache	20.0485	0.7510	0.2420

Table S2: Pre-processing cost of the proposed method. "Search" denotes the greedy search stage for select the great scheduler and "Train" denotes compensator training stage. All experiments were conducted on a single NVIDIA GeForce RTX 4090 48 GB GPU.

Model	Search		Train		Total (s)
	Setting	Time (s)	Size (MB)	Time (s)	
FLUX.1-DEV	G3-top3-10++	764.27	4.3	940.63	1704.90
COGVIDEOX-2B	G1-top5-17++	647.86	12.2	1082.69	1730.55
OPENSORA-1.2	G1-top3-10++	463.58	7.8	787.15	1250.73

D Synthetic Prompt Set Construction

To mitigate the risk that the greedy-search procedure and the lightweight MLP compensator overfit to a narrow prompt distribution, we construct an auxiliary synthetic prompt set using ChatGPT-5.5. The purpose of this set is not to introduce additional supervision, but to assess whether the learned

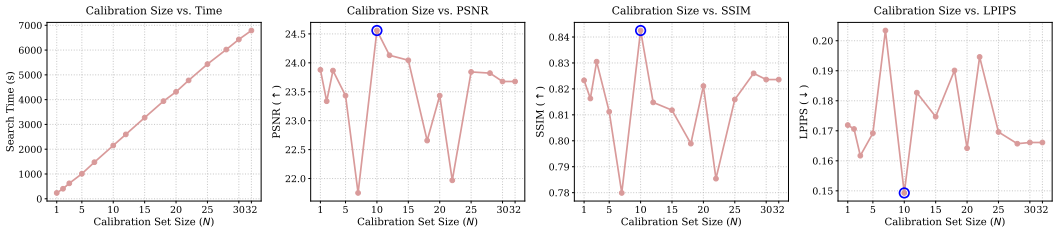

Figure S1: Impact of calibration set size (N) on cache scheduler optimization. We vary the number of calibration samples from $N = 1$ to 32 and report both the search cost and the resulting quality.

mechanism generalizes beyond a particular prompt distribution, rather than merely overfitting to the statistical characteristics of a single dataset. In particular, we intentionally encourage semantic diversity, broad scene coverage, and physically grounded descriptions, so that the resulting prompts better resemble the heterogeneous caption styles found in datasets such as MSCOCO (Lin et al., 2014), PartiPrompts (Yu et al., 2022), and VBench (Huang et al., 2024).

We generate two synthetic prompt subsets. For the FLUX.1-DEV experiments, we sample 2,000 prompts to cover a wider range of everyday scenes, human activities, object interactions, and multi-agent compositions. For COGVIDEOX-2B and OPENSORA-1.2, we generate 100 prompts each, focusing on concise yet visually grounded descriptions that are suitable for video generation and evaluation. Across all subsets, the prompts are designed to avoid duplication, avoid brand names and copyrighted entities, and remain simple enough to be parsed by the generation models.

Generation instruction. The following instruction is used to generate the synthetic prompts:

> Generate **{N}** image/video caption prompts in English. The prompts should be diverse, realistic, and visually grounded, following the general style of MSCOCO, PartiPrompts, and VBench. Cover a wide variety of everyday scenes, human actions, object interactions, indoor and outdoor environments, animals, vehicles, and small group activities. Favor concrete, physically plausible situations with clear observable content. Avoid abstract concepts, metaphorical descriptions, brand names, copyrighted characters, and overly rare or fantastical objects. Keep each prompt concise, natural, and grammatical, typically in a single sentence. Ensure that the prompts are semantically distinct and that the overall set spans a broad range of physical events, motion patterns, and scene compositions. Do not add numbering, bullet points, explanations, or any extra text; output only the prompts, one per line.

Examples of the image-generation prompts include:

> Mechanical wings attached to a robot.
> A white horse running across a grassy hill.
> A man is tightening screws with a screwdriver.
> A striped carpet continues halfway up the wall.
> ...

Examples of the video-generation prompts include:

> A red balloon and a child holding the string.
> A teddy bear and a gift box on a bed.
> A skateboard and a helmet on the sidewalk.
> A suitcase and a passport on a table.
> ...

This synthetic prompt set serves as a stress test for prompt-level generalization. If the compensator performs consistently on this auxiliary distribution, it suggests that the learned correction is not merely memorizing a single caption corpus, but capturing more transferable structure.

E Additional Scheduler Ablation Studies

E.1 Calibration set size (N)

We investigate the impact of the calibration set size N, which defines the number of samples used during the greedy search for step allocation. The objective is to determine the optimal data scale required for the scheduler to reliably estimate the truncation error distribution across the denoising trajectory. We systematically scale N from 1 to 32 and evaluate both the pre-processing temporal cost and the final generation fidelity. Corresponding results are shown in Figure S1.

Empirical results reveal a decoupled relationship between the calibration set scale and the resulting schedule quality. As N scales up, the required search time increases strictly linearly, escalating from approximately 235s at $N = 1$ to 6,788s at $N = 32$. However, this substantial expansion in computational overhead does not yield a monotonic improvement in generation quality. The final metrics fluctuate within a narrow band (e.g., LPIPS between [0.1493, 0.2034]), with larger sample sizes frequently underperforming smaller ones.

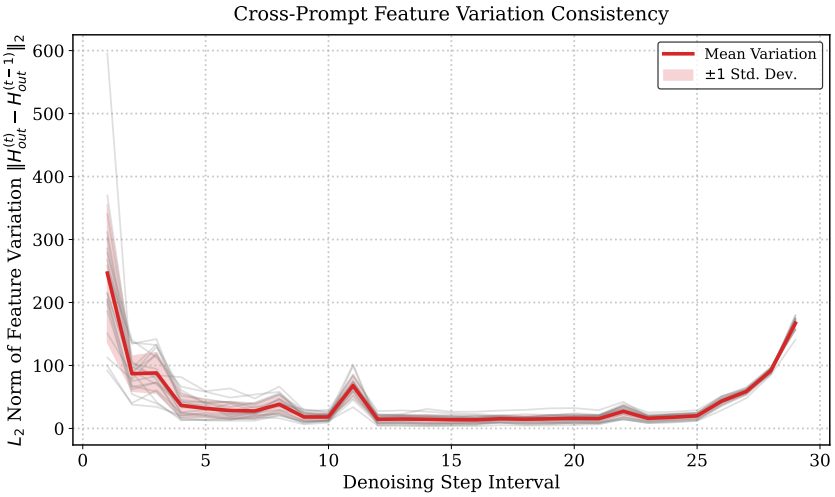

Figure S2: Temporal feature variation across diverse prompts. We track the L_2 feature variation magnitude throughout the denoising trajectory for 20 diverse prompts.

This phenomenon can be primarily attributed to a property of diffusion models: The temporal dynamics of diffusion models are inherently content-agnostic. As established by (Karras et al., 2022), the curvature of the Probability Flow ODE trajectories is fundamentally determined by the predefined noise schedule functions. Consequently, the magnitude of feature variations is predominantly governed by these global temporal parameters rather than the specific semantic layout of individual prompts. While recent studies like DeepCache (Ma et al., 2024b) and TeaCache (Liu et al., 2025a) have observed similar universal patterns, we conduct a targeted empirical study to directly substantiate this property. As illustrated in Figure S2, we track the L_2 norm of the feature variation, $\|H_{out}^{(t)} - H_{out}^{(t-1)}\|_2$, throughout the entire denoising trajectory for 20 highly diverse prompts (spanning minimalist still-life, complex cyberpunk scenes, abstract shapes, and high-frequency textures). Despite the extreme semantic and structural disparities, all trajectories exhibit a remarkably consistent temporal profile, bounded by a remarkably narrow variance (shaded region). This compelling visual evidence confirms that the error evolution is strictly a function of the integration timestep t.

Crucially, this content-agnostic behavior at the feature level directly bridges to the efficacy of our quality-driven scheduler. Because the local truncation errors introduced at any specific timestep are highly uniform across prompts, the compound perceptual degradation resulting from a specific schedule also exhibits a universal trend. Consequently, when our greedy algorithm evaluates candidate steps, the relative impact—and therefore the ranking—of these candidates remains highly consistent regardless of the calibration samples used. A minimal sample set is therefore sufficient to capture this quality-degradation landscape. Scaling up the calibration set (e.g., $N \geq 10$) merely av-

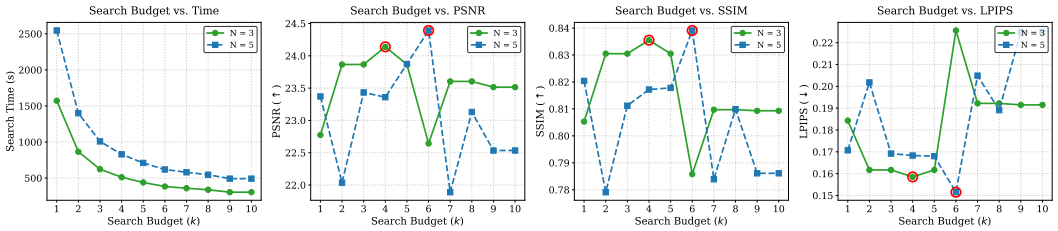

Figure S3: Quantitative ablation study of the greedy search budget (k) across four evaluation metrics: search time, PSNR, SSIM, and LPIPS, under calibration set sizes of $N = 3$ and $N = 5$. Red circles highlight the optimal performance points. The results demonstrate that moderate values of k yield the best generation fidelity while significantly reducing computational overhead.

erages the LPIPS scores across a broader distribution while substantially increasing the optimization time, yielding little additional benefit.

Driven by these observations, we conclude that large-scale calibration is fundamentally unnecessary for timestep-level caching. Setting $N = 3$ emerges as the optimal configuration, effectively capturing the essential temporal dynamics and quality constraints while maintaining the pre-processing overhead strictly within practical bounds.

E.2 Greedy search budget (k)

We investigate the impact of the greedy search budget, denoted as k, which dictates the number of candidate timesteps selected and appended to the cache schedule during a single iteration. A comprehensive evaluation is conducted across varying values of $k \in [1, 10]$ under two different calibration set sizes ($N = 3$ and $N = 5$). The analysis focuses on the trade-off between the pre-processing computational cost (search time) and the final generation fidelity (measured by LPIPS). The corresponding results are shown in Figure S3.

Empirical results reveal two primary observations regarding this hyperparameter. First, the search time exhibits a monotonically decreasing trend with diminishing marginal returns as k increases. By selecting more candidate steps per iteration, the algorithm reduces the total number of required optimization rounds (bounded by $\lceil N_{target}/k \rceil$). However, the reduction in execution time is strictly sub-linear rather than an ideal $O(1/k)$ scaling. To formally quantify this non-proportional decay, we model the search time $T(k)$ using an inverse proportional function with a constant baseline:

$$T(k) \approx \frac{\alpha}{k} + \beta$$

where α represents the scalable computational overhead strictly tied to the number of search iterations, and β denotes the irreducible constant overhead (e.g., fixed model weight loading, base forward passes, and feature comparison throughput). By fitting this model to our empirical measurements, we derive the specific equations for different calibration set sizes: $T_3(k) \approx \frac{1410}{k} + 163$ for $N = 3$, and $T_5(k) \approx \frac{2283}{k} + 264$ for $N = 5$. This mathematical formulation proves that as $k \to \infty$, the temporal cost asymptotically approaches the theoretical lower bound β ($\lim_{k\to\infty} T(k) = \beta$), structurally explaining the flattening of the latency curve at larger budgets. This behavior is mathematically consistent with the fact that the candidate pool dynamically shrinks across iterations, and there exists an irreducible computational lower bound for the base evaluations. Nonetheless, initial expansions of the search budget yield substantial efficiency gains: transitioning from $k = 1$ to $k = 4$ under the $N = 3$ setting reduces the search time by 67.4% (from 1572.90s to 511.71s), whereas further increasing k beyond 4 yields progressively marginal time savings.

Second, the generation quality does not exhibit a monotonic correlation with k. When k is too small (e.g., $k = 1$), the search procedure reduces to purely step-wise greedy selection, which lacks sufficient look-ahead capability and is therefore more likely to become trapped in local optima, yielding suboptimal global schedules. Conversely, when k is excessively large (e.g., $k \geq 7$), the scheduler aggressively selects a massive block of steps within a single iteration. This rapid aggregation prevents the algorithm from accurately evaluating the compound errors introduced by multiple selected steps, leading to noticeable performance degradation.

Optimal performance is consistently achieved at moderate values of k. Specifically, under the setting where inference and evaluation are conducted on 50 MSCOCO (Lin et al., 2014) samples, the best performance for $N = 3$ is achieved at $k = 4$, yielding the lowest LPIPS score of 0.1585; for $N = 5$, the lowest LPIPS (0.1515) is reached at $k = 6$. These moderate budgets strike an optimal balance, providing sufficient granularity to evaluate trajectory deviations while significantly accelerating the schedule optimization process.

E.3 Adjacent-step constraint

We investigate the necessity of the adjacent-step constraint during the greedy search phase. This constraint strictly prevents the selection of neighboring timesteps within the same iteration. Initial intuition suggested that the absence of this constraint might lead the scheduler to collapse into skipping excessively long consecutive segments. However, empirical observations reveal that while the global structural distribution of the schedules remains comparable, the removal of this constraint results in significant quantitative degradation.

Table S3: Ablation on the adjacent-step constraint across different search budgets (K). All settings use a 20/30 cache budget, evaluated at 512×512 resolution on 50 samples.

Budget (K)	Constraint	PSNR ↑	SSIM ↑	LPIPS ↓
Top-3	w/	23.87	0.8305	0.1617
	w/o	22.73	0.8018	0.1871
Top-4	w/	24.14	0.8355	0.1585
	w/o	22.73	0.8018	0.1871
Top-5	w/	23.87	0.8305	0.1617
	w/o	23.87	0.8222	0.1679

As detailed in Table S3, across varying candidate search budgets ($K \in \{3, 4, 5\}$), configurations without the constraint consistently exhibit degradation. A plausible explanation is that the greedy scheduler evaluates each candidate independently based on its immediate quality impact. Consequently, multiple adjacent timesteps within locally smooth regions may all appear favorable and be selected in the same iteration, even though their combined truncation error can be substantially larger than their individual effects suggest. By preventing adjacent selections within each iteration, the proposed constraint encourages the scheduler to explore more temporally diverse cache locations, reducing redundant selections and yielding more effective cache schedules.

F Additional Compensator Ablation Studies

F.1 Architectural components ablation

To evaluate the contribution of individual architectural choices within the neural compensator, we conduct a module drop-out ablation study. We systematically remove four distinct components: the spatial mixer (*w/o Spatial*), the global mean aggregation (*w/o Global Mean*), the Adaptive Layer Normalization (*w/o AdaLN*), and the SwiGLU activation (*w/o SwiGLU*). All variants are trained under identical configurations and evaluated on 200 samples at 512×512 resolution. The training loss trajectories are visualized in Figure S4, and the quantitative results are summarized in Table S4.

The empirical results reveal that the temporal conditioning mechanism is the most critical component. Removing the AdaLN module (*w/o AdaLN*) causes the most significant performance degradation, increasing the final LPIPS by 0.0056 and reducing PSNR by 0.31. This aligns with the theoretical properties of diffusion models: the curvature and magnitude of the denoising trajectory vary heavily across timesteps. Without dynamic feature modulation conditioned on the temporal embeddings, the network fails to effectively scale and shift its estimations, leading to a noticeable upward shift in the loss curve and diminished perceptual quality. Similarly, the removal of the global mean aggregation (*w/o Global Mean*) yields a measurable degradation in final image fidelity (LPIPS increases by 0.0034). This indicates that establishing a global statistical context assists the compensator in estimating broad structural deviations alongside local feature shifts. Furthermore, replacing

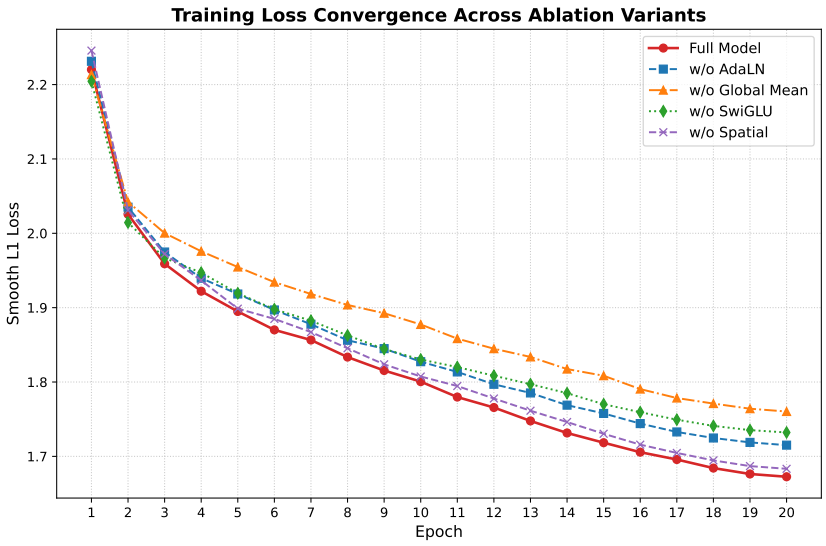

Figure S4: Training loss convergence across different structural variants of the compensator.

the SwiGLU activation with a standard FFN (*w/o SwiGLU*) slightly degrades performance, confirming that gated non-linearities improve the network's capacity to model complex residual mappings.

Interestingly, removing the spatial mixer (*w/o Spatial*) maintains comparable performance to the full model, with only a marginal decrease in PSNR (-0.06) and a slight increase in LPIPS ($+0.0020$). This observation implies that the cache-induced error is predominantly a channel-wise and temporal phenomenon rather than a highly localized spatial deviation. Nevertheless, we adopt the full configuration in all experiments to achieve the best overall performance.

Table S4: Quantitative ablation of compensator architectural components. Values in parentheses denote the absolute performance difference relative to the full model.

Variant	PSNR ↑	SSIM ↑	LPIPS ↓
Full Model	**22.34**	**0.8072**	**0.1712**
w/o Spatial	22.28 (-0.06)	0.8038 ($+0.0044$)	0.1732 ($+0.0020$)
w/o Global Mean	22.23 (-0.11)	0.8026 (-0.0046)	0.1746 ($+0.0034$)
w/o AdaLN	22.03 (-0.31)	0.8002 (-0.0070)	0.1768 ($+0.0056$)
w/o SwiGLU	22.18 (-0.16)	0.8024 (-0.0048)	0.1737 ($+0.0025$)

F.2 Cross-resolution training mix

We investigate the influence of the calibration data's spatial resolution on the generalization capability of the neural compensator. The objective is to determine whether a compensator trained on a specific resolution can reliably correct cache-induced truncation errors at unseen scales. To evaluate this, we train two distinct compensator variants exclusively on single-resolution subsets: one on 1,000 synthetic prompts at 256×256 resolution, and another at 512×512 resolution. We then cross-evaluate both models alongside the baseline ERTACache (Peng et al., 2025) on 1,000 samples from the MSCOCO (Lin et al., 2014) validation set at both resolutions.

The empirical results, detailed in Table S5 and visualized in Figure S5, reveal a severe overfitting phenomenon. While the single-resolution compensators demonstrate strong in-domain performance (e.g., the 256-trained model achieves an LPIPS of 0.1854 on the 256-test set, outperforming ERTA's 0.2131), they suffer from catastrophic degradation when applied to out-of-domain resolutions. Specifically, evaluating the 256-trained compensator on the 512-test set yields an LPIPS of 0.2692, which is noticeably worse than the uncompensated baseline ERTA (0.2418). A symmetric degradation is observed when the 512-trained compensator is evaluated on the 256-test set.

Table S5: Cross-resolution evaluation of the neural compensator. "Train-256" and "Train-512" denote models trained exclusively on 256×256 and 512×512 datasets, respectively. Gray shading highlights in-domain evaluation, where the compensator performs optimally.

Method	Test on 256×256			Test on 512×512		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
ERTACache	19.50	0.7092	0.2131	20.06	0.7510	0.2418
Train-256	**20.25**	**0.7407**	**0.1854**	18.63	0.7141	0.2692
Train-512	19.39	0.7084	0.2181	**20.40**	**0.7670**	**0.2177**

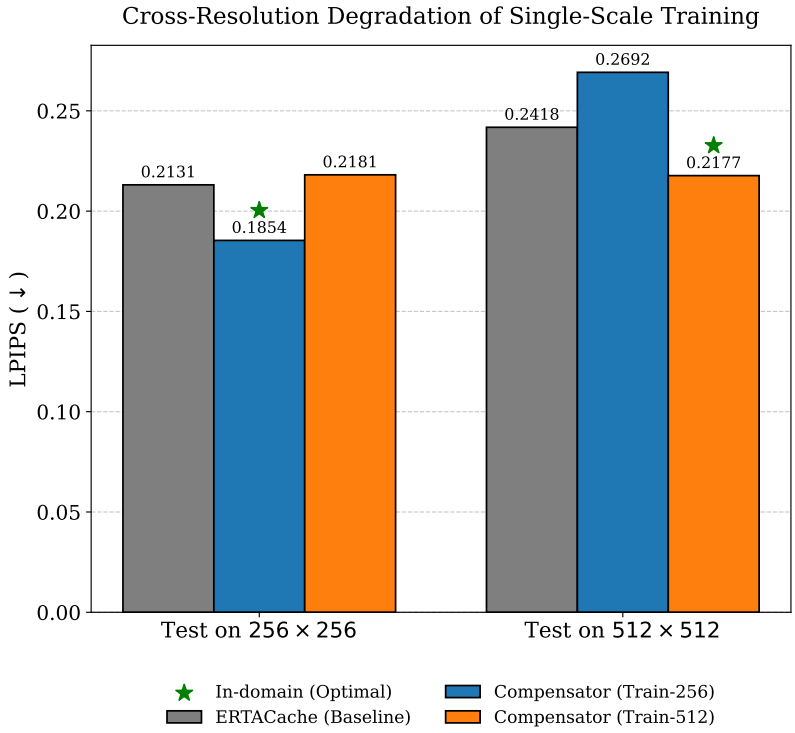

Figure S5: Cross-resolution performance trends corresponding to Table S5. The visualization highlights the strong in-domain advantage of single-resolution training.

This severe cross-resolution degradation can be attributed to the inherent scale variance in DiTs. As spatial resolution increases, the token sequence length expands quadratically, altering the feature statistics and the geometry of the denoising trajectory. A compensator optimized on a singular sequence length inevitably overfits to the specific high-frequency spatial distributions of that resolution. When subjected to OOD features, the predicted residual corrections become misaligned in magnitude and direction, actively perturbing the trajectory away from the ground-truth.

To address this issue, our final framework adopts a mixed-resolution training strategy (30% samples at 256 and 70% at 512). Training across multiple resolutions acts as an effective structural regularizer, encouraging the compensator to learn scale-invariant feature corrections rather than resolution-specific representations. Remarkably, this mixed-training paradigm not only resolves the degradation between 256 and 512 but also unlocks robust zero-shot extrapolation to unseen resolutions. As demonstrated in our main quantitative results (Table 1a), the compensator trained purely on the 256/512 mixture maintains exceptional error-rectification efficacy at 1024×1024. For instance, under a $\sim 2.77\times$ speedup regime, our baseline scheduler yields an LPIPS of 0.2207 at 1024×1024; integrating the mixed-trained compensator further suppresses this error to 0.1914, establishing a definitive advantage over ERTACache (0.2725). This confirms that mixed-resolution supervision is essential for establishing a generalized, scale-agnostic correction mechanism.

F.3 HISTORICAL CONTEXT ORDER

To dynamically align the non-linear denoising trajectory, the lightweight neural compensator relies on historical feature states to estimate the local curvature. We conduct an ablation study to investigate two fundamental dimensions of this historical context: the anchoring strategy of the history (i.e., whether the historical states precede the *current* step or the most recent *cached* step) and the temporal order magnitude ($O \in \{1, 2, 3, 4\}$). The results are presented in Figure S6.

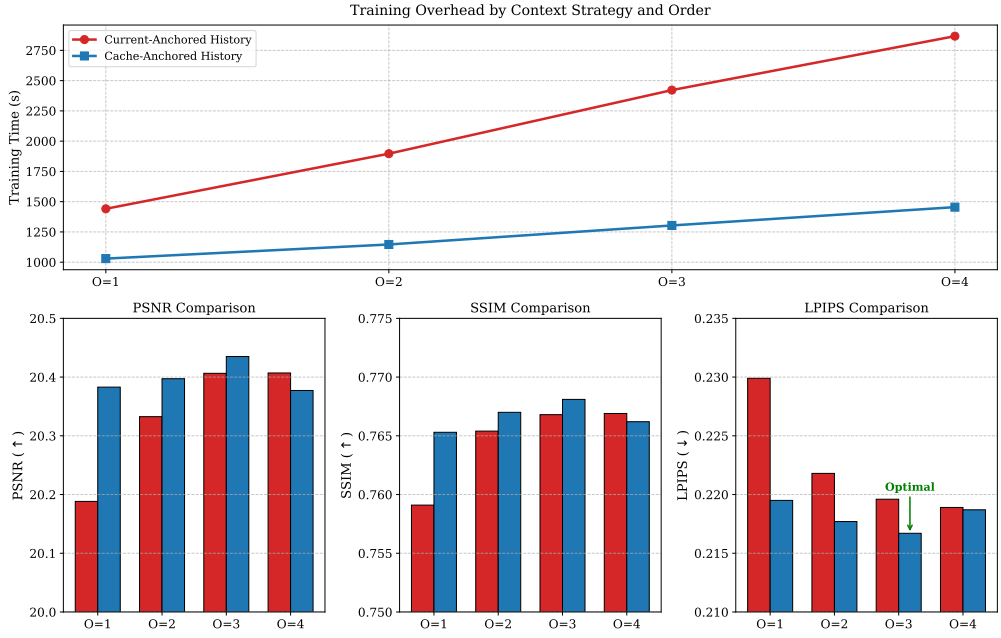

Figure S6: Ablation on historical context. The figure compares two anchoring strategies (Current-Anchored vs. Cache-Anchored) across different temporal orders $O \in \{1, 2, 3, 4\}$.

Context Anchoring Strategy. We evaluate two history-fetching mechanisms. The first strategy, *Current-Anchored History*, continuously updates the historical buffer with the intermediate features produced at every timestep, including those approximated by the compensator during cache steps. The second strategy, *Cache-Anchored History*, exclusively records the true features generated by the full model and freezes this context buffer throughout subsequent cache steps.

Empirical evidence decisively favors the Cache-Anchored strategy. As shown in the top panel of Figure S6, the Cache-Anchored mechanism incurs substantially lower training overhead across all orders. For instance, at $O = 4$, it reduces training time by nearly 50% (from 2867s to 1454s). This efficiency stems from the elimination of redundant buffer updates during continuous cache intervals. More importantly, the Cache-Anchored strategy consistently yields superior perceptual fidelity (lower LPIPS). By anchoring the compensator's input to the pristine states of the latest computed step, it effectively isolates the prediction from the compounding truncation errors that inevitably pollute the approximated intermediate states in the Current-Anchored buffer.

Impact of Temporal Order (O). Within the Cache-Anchored framework, we further observe the influence of the order size O. Increasing O initially improves generation quality by providing the compensator with higher-order historical information along the trajectory. As O increases from 1 to 3, the LPIPS drops monotonically from 0.2195 to 0.2167. However, this improvement plateaus and slightly degrades when further extending to $O = 4$ (LPIPS rises to 0.2187), suggesting that incorporating more distant historical states provides limited additional benefit for local error rectification. Consequently, we adopt $O = 3$ as the default configuration, achieving the best trade-off between compensation effectiveness and computational efficiency.

G Additional Image Visualization Examples

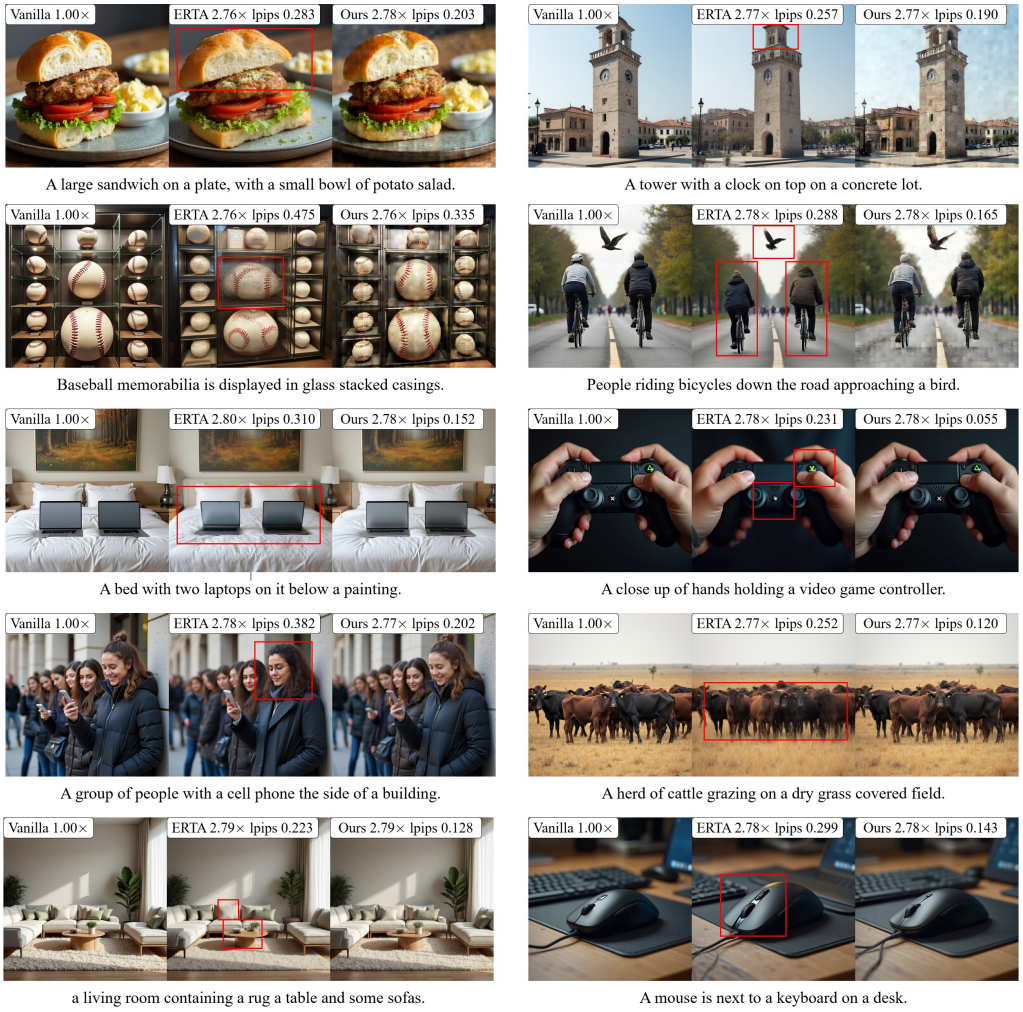

Figure S7: Additional image generation examples on FLUX.1-DEV. The top two rows correspond to 512×512 resolution, while the bottom three rows correspond to 1024×1024. Red boxes highlight representative artifacts and structural inconsistencies introduced by ERTACache (Peng et al., 2025). Our method preserves higher visual fidelity (measured by LPIPS scores).

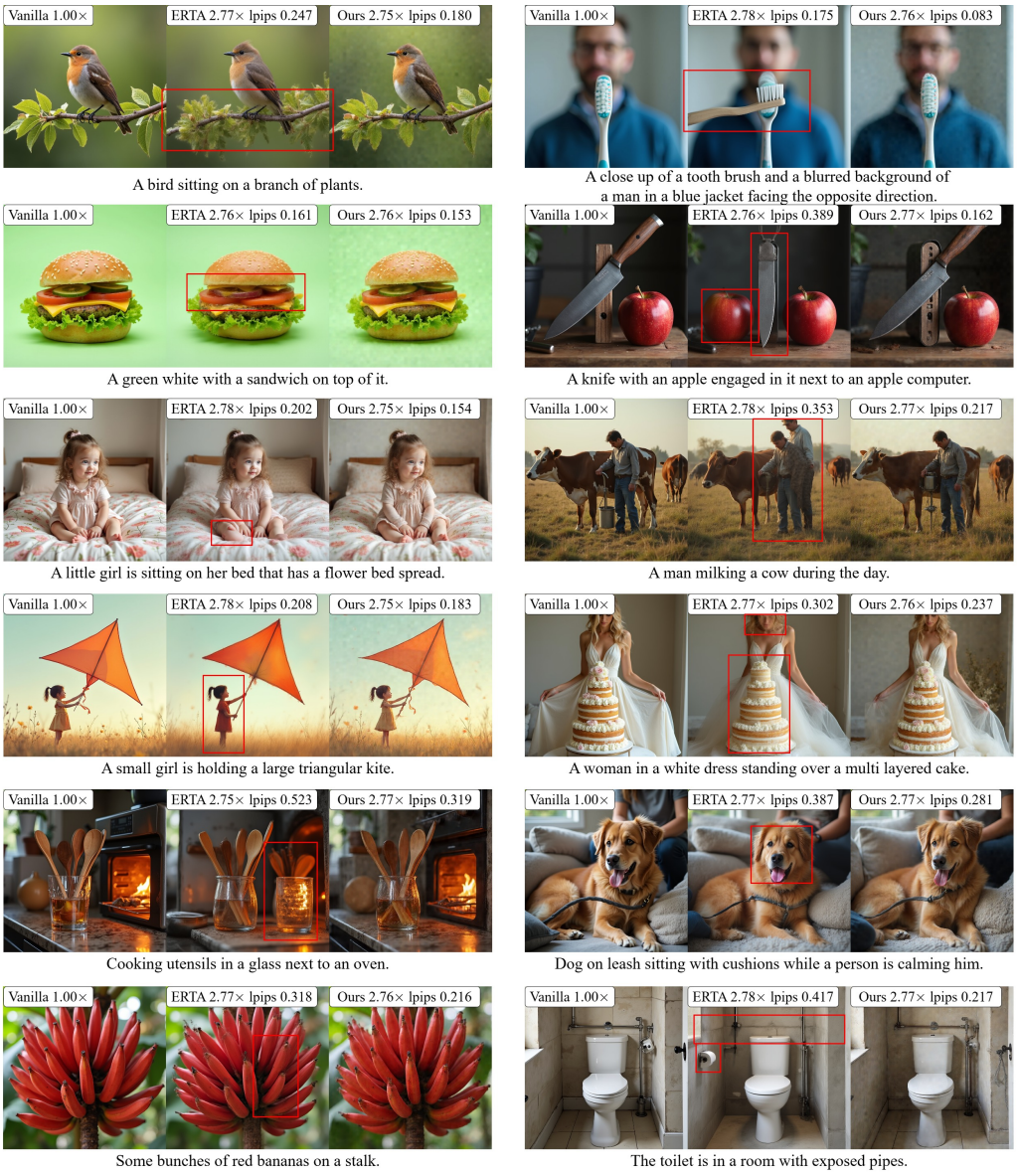

Figure S8: Additional image generation examples on FLUX.1-DEV at 768 × 768 resolution. Red boxes highlight representative artifacts and structural inconsistencies introduced by ERTACache. Corresponding LPIPS scores are provided for each sample for quantitative comparison.



Figure S9: Additional image generation examples on FLUX.1-DEV. The top two rows correspond to 768×768 resolution, while the bottom three rows correspond to 1024×1024. Red boxes highlight representative artifacts and structural inconsistencies introduced by ERTACache (Peng et al., 2025).

H ADDITIONAL VIDEO VISUALIZATION EXAMPLES

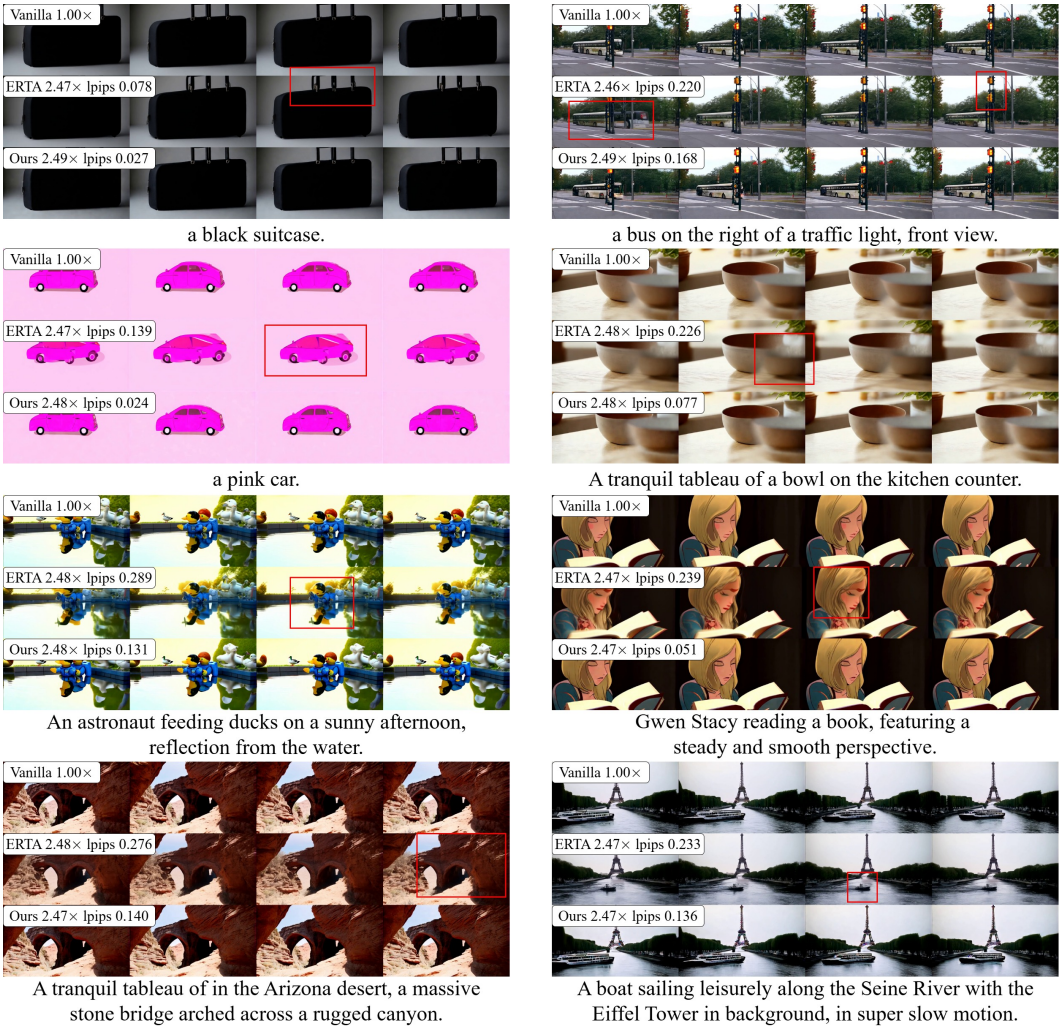

Figure S10: Additional video generation examples on OPENSORA-1.2. Red boxes highlight representative regions where ERTACache (Peng et al., 2025) introduces noticeable visual artifacts or structural inconsistencies. By comparison, the proposed method more faithfully preserves the appearance and structural integrity of the original outputs, resulting in higher visual fidelity.

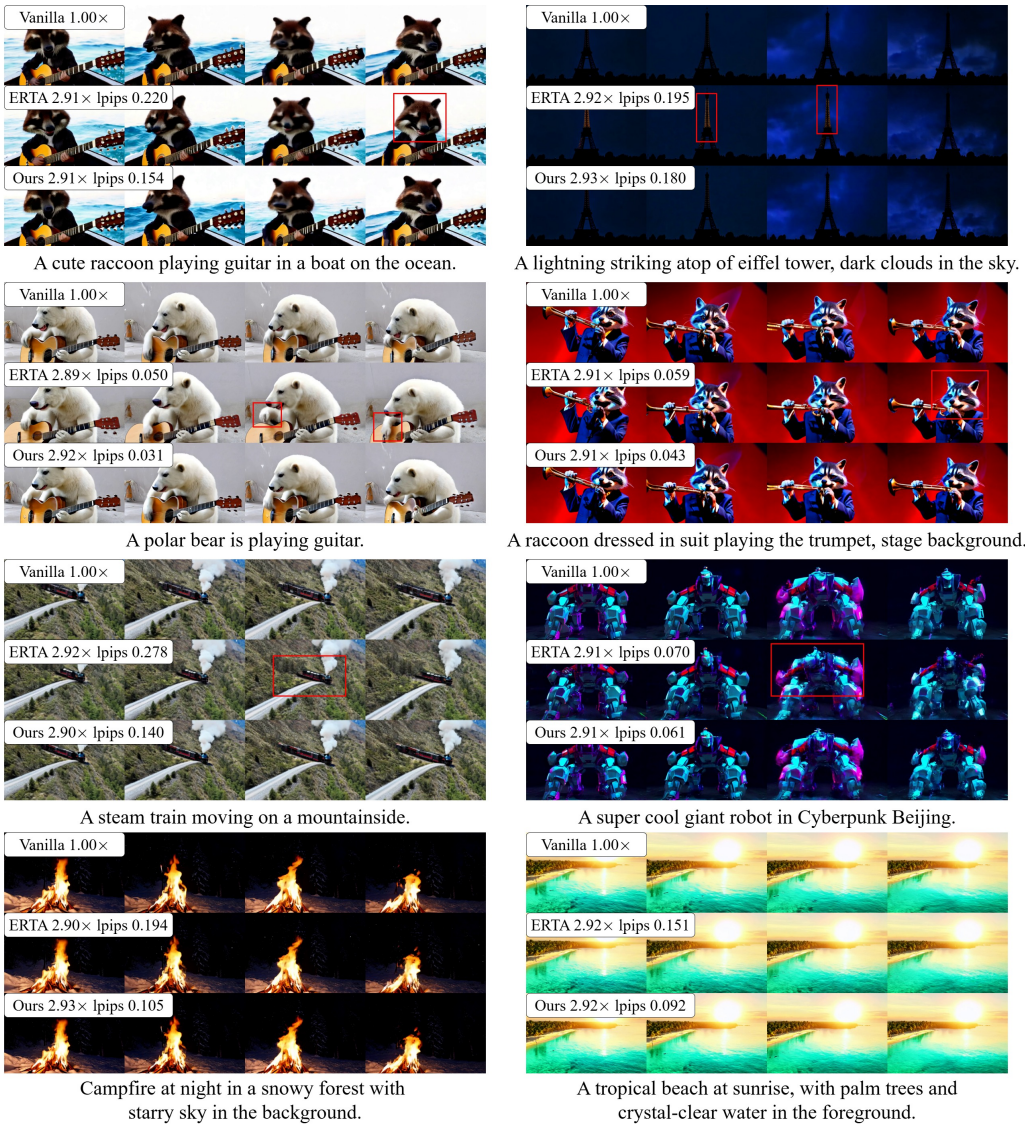

Figure S11: Additional video generation examples on CogVideoX-2b. Red boxes highlight representative regions where ERTACache introduces visible artifacts that can be readily perceived by human observers. In contrast, our method better preserves the visual fidelity of the original outputs. For the bottom two examples, the visual differences are subtle and difficult to distinguish from static frames alone; nevertheless, our method consistently achieves substantially lower LPIPS scores than ERTACache, indicating improved perceptual similarity to the reference generation.