ResilPhase: Plug-and-Play Phase Mapping and Noise-Resilient Macro-Trajectory Extrapolation for Diffusion Acceleration

Qicheng Zhao, Yu Li, Qi Sun, and Zheyu Yan*

Zhejiang University, Hangzhou, China
zyan2@zju.edu.cn

Abstract. The adoption of powerful diffusion models is hindered by their significant inference latency. Recent "cache-then-forecast" schemes alleviate this issue by accelerating DiTs using derivative-based polynomials, but they suffer from severe quality degradation at high acceleration ratios. Our analysis reveals its root cause: the discrete extrapolation performed on representations that are misaligned with the continuous diffusion trajectory and are numerically unstable. Thus, accelerated DiTs suffer from accumulated spatial errors, noisy derivative amplification, and high-order instability. We therefore reformulate accelerated inference as stable macro-trajectory extrapolation in ordinary differential equation (ODE) space. Instead of predicting intermediate features, we align forecasting with the model's Global Drift (GD), i.e., the end-to-end state evolution, thereby eliminating feature inconsistency and memory overhead. However, even this smooth macro-trajectory remains vulnerable to the derivative fallacy: its higher-order temporal derivatives are intrinsically noisy. Thus, we introduce a derivative-free barycentric Lagrange extrapolator to effectively bypass derivative instability and approximation error. We further propose a bounded Phase Mapping that regularizes the extrapolation domain, suppressing oscillatory error growth. These elements collectively constitute ResilPhase, a noise-resilient acceleration framework. Experiments on FLUX.1-dev and HunyuanVideo demonstrate state-of-the-art fidelity under aggressive acceleration ratios. Code is publicly available at `https://github.com/zqc214/ResilPhase`.

Keywords: Diffusion Model · Efficient Inference · Generative Model

1 Introduction

Diffusion Transformers (DiTs) [11, 33] have become the gold standard for high-fidelity visual generation [1, 30, 32, 35, 41, 45, 46, 50, 56], because of their scalable architecture. However, their performance is achieved through an iterative denoising process, requiring tens to hundreds of sequential forward steps that can not be parallelized. This has become a significant latency bottleneck that obstructs

* Corresponding author.

real-time and large-scale deployment of DiTs. FLUX.1-dev takes 23.69 seconds to generate an image on an A100 graphics card.

To address this latency bottleneck, recent training-free acceleration efforts have evolved from passive "cache-then-reuse" paradigms [19,24,31,44] to "cache-then-forecast" approaches [3,25,26,55] that employ polynomials to predict feature trajectories. However, this prevailing paradigm suffers from severe quality degradation at high acceleration ratios. We argue this failure stems from a single root cause: discrete extrapolation is performed on intermediate representations misaligned with the continuous diffusion trajectory and numerically unstable. This root cause manifests in three intertwined limitations.

First, from a spatial perspective, most methods rely on a computationally intensive layer-wise paradigm. Fitting polynomials to highly erratic micro-features within Transformer blocks cascades errors across the network depth, incurring massive memory overhead. While directly forecasting absolute outputs (e.g., FreqCa [23]) bypasses this, as highlighted by Δ-DiT [4], it discards highly correlated input priors, restricting prediction fidelity bounds. Conversely, methods reusing residual displacements (e.g., Δ-DiT [4]) remain trapped in localized micro-updates, missing macroscopic continuous dynamics. Thus, existing paradigms lack an end-to-end target that simultaneously preserves input priors and aligns strictly with the ODE vector field.

Second, from a temporal perspective, current forecasting methods (e.g., TaylorSeer [25] and HiCache [7]) rely exclusively on derivative-based approximations. They critically overlook the mathematical nature of the dynamic trajectory: while the macro-trajectory itself is smooth, its higher-order derivatives over time are intrinsically chaotic (as shown in Fig. 2(b)). Estimating these derivatives via finite differences exponentially amplifies the noise.

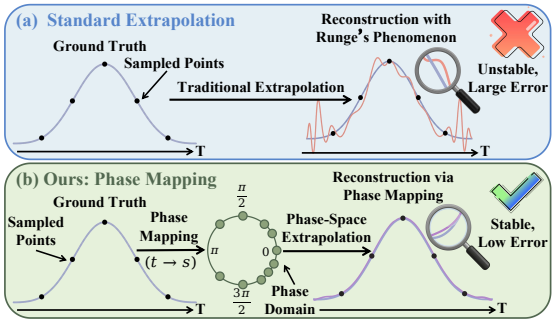
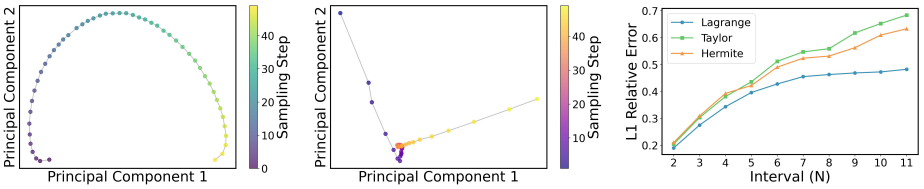


Fig. 1: Suppressing numerical instability via Phase Mapping. (a) Standard trajectory extrapolation on uniform timesteps suffers from Runge's phenomenon, causing chaotic edge oscillations. (b) Our method maps discrete steps to a bounded phase space, minimizing the error bound for a stable fit.

Third, from a numerical approximation standpoint, performing polynomial extrapolation over uniform, discrete time steps inherently triggers Runge's phenomenon. This numerical instability causes the extrapolation error bound to grow uncontrollably at the interval edges, inevitably leading to catastrophic quality degradation at aggressive acceleration ratios.

To overcome these intertwined bottlenecks, we propose **ResilPhase**, a noise-resilient acceleration framework explicitly addressing the spatial, temporal, and numerical limitations of existing paradigms. First, to resolve spatial cascading errors inherent in layer-wise forecasting, ResilPhase shifts the prediction objec-



(a) Global Drift Trajectory(Ours) (b) Finite Difference Trajectory (c) Prediction Error vs. Interval N

Fig. 2: Effectiveness of the derivative-free prediction strategy. Unlike (a) our smooth global macro-trajectory, (b) the finite-difference derivative trajectory used by prior methods is intrinsically chaotic. Applying polynomial extrapolation to such unstable dynamics causes severe prediction deviations. As shown in (c), evaluating pure mathematical predictors reveals that our derivative-free Lagrange extrapolator maintains significantly lower L_1 relative error across varying extrapolation intervals (N) than derivative-based solvers (Taylor, Hermite).

tive to the network's macroscopic dynamic evolution. We propose ODE-Aligned Macro-Trajectory Targeting, formulating the prediction target as the Global Drift (GD): the end-to-end state displacement between the model's final output and initial input. Unlike layer-wise approaches fitting highly oscillatory micro-signals within Transformer blocks, forecasting the GD aligns the extrapolator with the continuous probability flow ODE. This paradigm shift eliminates the memory overhead of caching block features and strictly severs error accumulation across network layers.

While predicting the GD yields a smooth macro-trajectory ideal for forecasting (Fig. 2(a)), it paradoxically amplifies the weakness of derivative-based methods. Like layer-wise micro-features, the GD's higher-order temporal derivatives remain intrinsically chaotic. Applying derivative-based solvers like Taylor or Hermite to such noisy signals causes divergent predictions. Furthermore, finite-difference estimation of these derivatives across discrete steps introduces compounding approximation errors. To avoid this, we propose a derivative-free extrapolator inspired by Lagrange interpolation. Avoiding the standard $O(N^2)$ formulation, we develop a barycentric prediction scheme caching and reusing weights. This reduces complexity to $O(N)$ while eliminating derivative noise. As a purely mathematical predictor, our approach achieves significantly lower and more stable error across extrapolation intervals than prior solvers (Fig. 2(c)).

Even with a robust derivative-free formulation, polynomial extrapolation over uniform discrete time steps remains susceptible to Runge's phenomenon. As shown in Fig. 1(a), this numerical instability causes severe edge oscillations; ResilPhase is the first work to identify this as a critical bottleneck in diffusion acceleration. To address this, we introduce a plug-and-play Phase Mapping mechanism (Fig. 1(b)). As the first theoretically grounded technique to remap the discrete temporal extrapolation domain, we bridge classical numerical analysis and diffusion acceleration by utilizing Chebyshev nodes to project linear time steps (t) into a bounded phase domain (s), achieving optimal stability for class-conditional generation. Furthermore, we propose a data-driven Balanced Mapping tailored to complex text-to-image and video tasks. By performing extrapolation within this bounded continuous space, our mappings effectively transform

a divergent numerical issue into a highly stable prediction, strictly minimizing the mathematical upper bound of the extrapolation error.

To summarize, our primary contributions are as follows:

- We propose ODE-Aligned Macro-Trajectory Targeting. By forecasting the model's Global Drift (GD) rather than layer-wise features, we strictly sever spatial cascading errors, align with the continuous diffusion ODE, and eliminate heavy memory overhead.
- We introduce a noise-resilient, derivative-free Barycentric Lagrange extrapolator. This $O(N)$ framework perfectly synergizes with the GD by completely bypassing the inherent noise and chaotic high-order derivatives of finite-difference approximations.
- We design a plug-and-play Phase Mapping mechanism. By non-linearly projecting discrete time steps into a bounded phase space, it regularizes the extrapolation domain to suppress oscillatory error growth, strictly minimizing the polynomial extrapolation error bound.
- Extensive experiments show ResilPhase achieves $\sim 5\times$ speedups on FLUX.1-dev and HunyuanVideo while maintaining highly competitive fidelity. Furthermore, Phase Mapping acts as a plug-and-play stabilizer, mitigating extrapolation errors in existing derivative-based accelerators.

2 Related Work

2.1 Diffusion Transformers and Traditional Acceleration

Diffusion Transformers (DiTs) achieve high-fidelity generation through a sequential noise-reversal process, which fundamentally imposes a severe inference latency bottleneck. To mitigate this, early acceleration efforts primarily focused on model compression, such as network pruning [6,13,14,16,49,62] and quantization [8,15,17,20,40,59], or step reduction via efficient solvers [28,29,47,52,53,60] and knowledge distillation [5, 21, 38, 42, 58, 61]. However, these approaches typically demand computationally expensive retraining to recover generation quality and often rely on complex algorithmic designs. This reliance fundamentally limits their plug-and-play generality, driving recent research toward more flexible, training-free acceleration paradigms.

2.2 Acceleration Based on Feature Caching

Feature caching, the main training-free diffusion acceleration strategy, evolved from passively reusing temporal features (DeepCache [31], FORA [39], Δ-DiT [4], TeaCache [22], ToCa [63]) to actively predicting them (TaylorSeer [25], FoCa [55], HiCache [7], SpeCa [26], ClusCa [57], FreqCa [23]). While recent methods like DiCache [2] dynamically adjust caching intervals, these predictive approaches share critical limitations. Spatially, layer-wise forecasting incurs memory overhead and amplifies micro-feature errors across network depth. Temporally, these methods

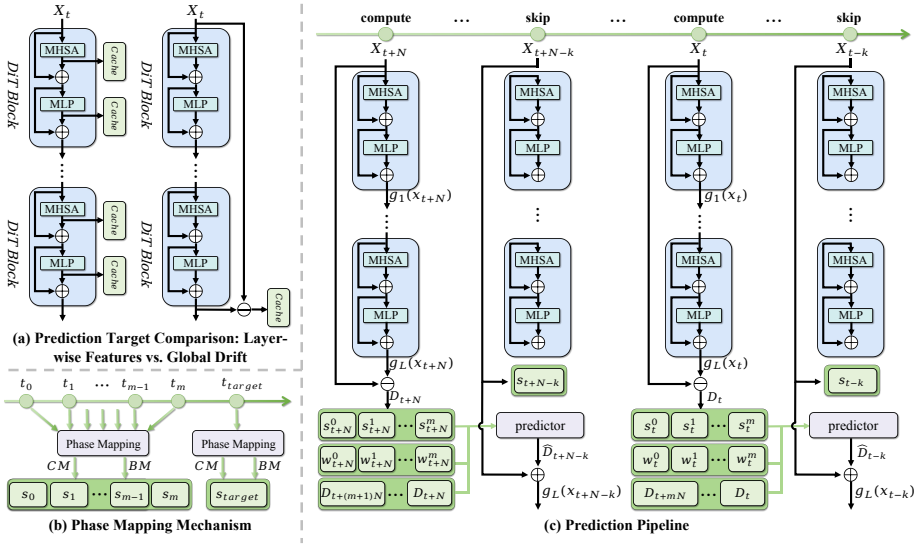


Fig. 3: The ResilPhase acceleration framework. (a) We shift from layer-wise features to the ODE-aligned Global Drift (GD), the end-to-end state displacement. (b) Phase Mapping non-linearly projects discrete time steps into a bounded phase space via Chebyshev (CM) or Balanced (BM) to suppress numerical instability. (c) The pipeline runs full computations every N steps, while our derivative-free Barycentric Lagrange extrapolator of order m estimates the GD for intermediate steps.

implicitly assume smooth high-order derivatives, relying on finite-difference approximations. However, approaches using Taylor series [25, 26], Hermite polynomials [7,23], or ODE-based forecasting [55] encounter a mathematical bottleneck: while the macro-trajectory is smooth, its temporal derivatives remain intrinsically chaotic. Consequently, applying derivative-based forecasting to these noisy signals makes predictions susceptible to numerical instabilities like Runge's phenomenon, restricting maximum acceleration and robustness.

3 Methodology

3.1 Preliminaries and the Derivative-Induced Instability

Diffusion Transformers (DiT) A DiT model functions as a learnable velocity field in a probability flow ODE. It consists of a stack of L Transformer blocks. We denote the transformation of the l-th block as g_l, where $l \in \{1, \dots, L\}$. For the input latent state $\mathbf{x}_t$ at timestep t and condition c, the end-to-end mapping is a composition of these blocks: $G(\mathbf{x}_t, c) = g_L(g_{L-1}(\dots g_1(\mathbf{x}_t, c) \dots))$. Thus, the input $(\mathbf{x}_t, c)$ is transformed into the final output velocity (or noise prediction) through this complete process G.

Predictive Caching for Acceleration. To reduce inference latency, predictive caching uses a sparse schedule. Given an acceleration rate N, full forward passes occur only at anchor timesteps (e.g., $t, t-N, \dots$). For intermediate steps, computations are skipped and features are estimated via a lightweight predictor.

Conventionally, this prediction is layer-wise. Let $F(x_t^l)$ denote the feature output of layer l at timestep t. Existing methods (e.g., TaylorSeer [25], SpeCa [26]) assume smooth feature trajectories and employ polynomial expansions (like Taylor series) for forecasting:

$$F_{\text{pred},m}(x_{t-k}^l) = F(x_t^l) + \sum_{i=1}^{m} \frac{\Delta^i F(x_t^l)}{i!}(-k)^i, \tag{1}$$

where $\Delta^i F$ is the i-th order finite difference derivative approximation.

The Derivative-Induced Instability. Applying Equation (1) to discrete diffusion dynamics is inherently unstable. While feature values are smooth, their finite-difference derivatives ($\Delta^i F$) are dominated by chaos (Fig. 2). Because finite differences act as high-pass filters that exponentially amplify noise, derivative-based solvers (including Hermite splines [7]) inherently destabilize trajectory reconstruction. This necessitates our derivative-free paradigm.

3.2 The ResilPhase Framework

To bypass derivative instability, we adopt a derivative-free approach. For $m+1$ historical data points $\{(t_0, F_0), \ldots, (t_m, F_m)\}$, the unique degree-$m$ interpolation polynomial $P(t)$ is:

$$P(t) = \sum_{j=0}^{m} F_j L_j(t). \tag{2}$$

ResilPhase (Fig. 3) makes two choices. First, the target F_j is the Global Drift, replacing erratic layer-wise features with a smooth macro-trajectory. Second, we stabilize the basis L_j via Barycentric Interpolation and Phase Mapping. This enables accurate extrapolation every N steps without gradient noise.

3.3 ODE-Aligned Macro-Trajectory Targeting

Previous caching-based acceleration methods rely on layer-wise prediction. However, from a dynamic systems perspective, forcing polynomials to fit highly nonlinear, erratic micro-features within individual Transformer blocks causes prediction errors to cascade across the network depth.

The Spatial Cascading Error of Layer-wise Forecasting. Let x_t^{l-1} be the latent input to the l-th Transformer block at timestep t. In a DiT block, the exact forward computation updates the hidden state via residual connections across the self-attention ($f_{\text{attn},l}$) and MLP ($f_{\text{mlp},l}$) modules. For mathematical simplicity, we group the sum of these residual updates at layer l into a single function f_l, meaning the exact transformation is $x_t^l = x_t^{l-1} + f_l(x_t^{l-1})$. During a skipped step, layer-wise methods must approximate these internal updates via a polynomial predictor $\mathcal{P}_l$, yielding the estimated feature:

$$\hat{x}_t^l = \hat{x}_t^{l-1} + \mathcal{P}_l(\hat{x}_t^{l-1}). \tag{3}$$

Let $e_l = \|\mathcal{P}_l(\hat{x}_t^{l-1}) - f_l(\hat{x}_t^{l-1})\|$ and $E_l = \|\hat{x}_t^l - x_t^l\|$ denote the local prediction error and the total accumulated state error up to layer l, respectively. Assuming

the transformation f_l is Lipschitz continuous with constant L_f, we bound the accumulated error via the triangle inequality:

$$\begin{aligned} E_l &= ||(\hat{x}_t^{l-1} + \mathcal{P}_l(\hat{x}_t^{l-1})) - (x_t^{l-1} + f_l(x_t^{l-1}))|| \\ &\leq ||\hat{x}_t^{l-1} - x_t^{l-1}|| + ||\mathcal{P}_l(\hat{x}_t^{l-1}) - f_l(\hat{x}_t^{l-1})|| + ||f_l(\hat{x}_t^{l-1}) - f_l(x_t^{l-1})|| \\ &\leq E_{l-1} + e_l + L_f E_{l-1} = (1 + L_f)E_{l-1} + e_l. \end{aligned} \tag{4}$$

Solving this recurrence relation from the first layer $l = 1$ to the final layer L (noting that the input is exact, thus $E_0 = 0$), the total spatial cascading error is bounded by:

$$E_L \leq \sum_{l=1}^{L} (1 + L_f)^{L-l} e_l. \tag{5}$$

This rigorous mathematical bound reveals a fundamental flaw in existing paradigms: spatial prediction errors amplify exponentially across the network depth L. Fitting high-frequency micro-features at each intermediate layer merely exacerbates the local error e_l, driving the total divergence E_L uncontrollably high at aggressive acceleration ratios.

Formulating the Global Drift. To fundamentally sever this spatial error accumulation, we must elevate the prediction objective from intermediate micro-features to the macroscopic evolution of the ODE. We define the Global Drift (GD), denoted as $D(x_t)$, as the end-to-end state displacement between the model's final output $G(x_t)$ and its initial input x_t:

$$D(x_t) = G(x_t) - x_t. \tag{6}$$

Instead of predicting L distinct layer updates, a single macro-predictor $\mathcal{P}_{\text{macro}}$ forecasts this trajectory during skipped steps, yielding $\hat{D}(x_t)$. The final output is reconstructed via addition: $\hat{G}(x_t) = x_t + \hat{D}(x_t)$.

Mathematically, the approximation error of our macro-trajectory framework is strictly determined by a single term:

$$E_{\text{macro}} = ||\hat{G}(x_t) - G(x_t)|| = ||\hat{D}(x_t) - D(x_t)|| = e_{\text{macro}}. \tag{7}$$

By treating the entire DiT stack as a unified probability flow step, we completely bypass the cascading recurrence relation. The error bound is decoupled from the network depth L, successfully reducing the spatial error accumulation from $O((1 + L_f)^L)$ to $O(1)$.

While the GD formulation successfully eliminates spatial error amplification, forecasting this macro-trajectory with derivative-based solvers remains vulnerable to the chaotic noise highlighted in Section 3.1. This necessitates a robust, derivative-free extrapolation framework, which we introduce next.

3.4 Derivative-Free Barycentric Interpolation

To circumvent the catastrophic noise amplification inherent in derivative-based solvers as highlighted in the Derivative Paradox, we strictly ground our framework in a derivative-free approach. We define our target feature value F_j as

the fully computed Global Drift $D(x_t)$ at historical timestep t_j. A straightforward yet effective formulation of the interpolation polynomial $P(t)$ relies on the classical Lagrange basis polynomials L_j, where

$$L_j(t) = \prod_{k=0, k\neq j}^{m} \frac{t - t_k}{t_j - t_k}. \tag{8}$$

However, the standard Lagrange formula is computationally expensive with a computational complexity of $O(m^2)$. To address this issue, we adopt a more stable and efficient variant: the Barycentric Lagrange Interpolation Formula:

$$P(t) = \frac{\sum_{j=0}^{m} \frac{w_j}{t - t_j} F_j}{\sum_{j=0}^{m} \frac{w_j}{t - t_j}}, \tag{9}$$

$$\text{where} \quad w_j = \frac{1}{\prod_{k=0, k\neq j}^{m} (t_j - t_k)}. \tag{10}$$

In this formulation, the barycentric weights w_j depend only on the relative positions of the interpolation nodes $\{t_j\}$, not the target feature values $\{F_j\}$. This pivotal property allows the weights to be precomputed and cached during the full computation steps, making them readily available for all subsequent predictions. Thus, the complexity can be reduced to $O(m)$.

3.5 Phase Mapping for Prediction Stability

While the barycentric extrapolator isolates gradient noise, applying polynomial extrapolation over uniform discrete timesteps triggers severe numerical instability. This classic issue, Runge's phenomenon, causes uncontrollable edge oscillations that degrade fidelity at high acceleration ratios. To suppress this error growth, we introduce a plug-and-play Phase Mapping mechanism (Fig. 3(b)).

The Source of Numerical Instability. To understand this instability mathematically, we examine the error term for Lagrange interpolation. For a function $F(t)$ that is $m + 1$ times differentiable, there exists a real number $\xi \in [t_{\min}, t_{\max}]$ such that the error $E(t) = F(t) - P(t)$ of its degree-m polynomial interpolation $P(t)$ is:

$$E(t) = \frac{F^{(m+1)}(\xi)}{(m+1)!} \prod_{j=0}^{m} (t - t_j), \tag{11}$$

where t_js are time steps used for interpolation. There are two independent components of this function: $D_{m+1} = \frac{F^{(m+1)}(\xi)}{(m+1)!}$, and $e(t) = \prod_{j=0}^{m} (t - t_j)$.

Because the D_{m+1} term is an inherent property of the DiT model and cannot be modified, the only way to reduce the total error $E(t)$ is by minimizing the node-dependent term $e(t)$. We achieve this by remapping the uniformly spaced time steps t_j to a new, non-uniform distribution of nodes s_j (i.e., a mapping of $t \to s$). Based on the properties of DiTs, we propose two such mapping techniques suitable for different tasks: Chebyshev Mapping and Balanced Mapping.

Chebyshev Mapping For a given set of $m+1$ recent, fully computed timesteps $\{t_0, \ldots, t_m\}$, we generate the corresponding Chebyshev nodes $\{s_k\}$ as follows:

$$s_k = \cos\left(\frac{(2k+1)\pi}{2(m+1)}\right), \quad \text{for } k = 0, \ldots, m. \tag{12}$$

However, this formula only prescribes the discrete nodes $\{s_k\}$ for known timesteps and is not a continuous function applicable to an arbitrary t_{target}. To extrapolate the phase coordinate s_{target} for a target step $t_{\text{target}} < t_m$, we employ stable linear extrapolation using the two most recent points:

$$s_{\text{target}} = s_m + \frac{s_m - s_{m-1}}{t_m - t_{m-1}}(t_{\text{target}} - t_m). \tag{13}$$

Balanced Mapping While the fixed structure of Chebyshev nodes effectively bounds the error space, its predetermined nature lacks the flexibility to dynamically adjust to varying temporal distributions encountered during highly complex text-conditioned tasks. To provide a more robust and adaptable phase transformation, we propose a novel, data-driven mechanism called Balanced Mapping.

Unlike Chebyshev mapping, this adaptive strategy process begins by analyzing the current set of $m+1$ fully computed timesteps $\{t_j\}_{j=0}^{m}$ to compute their mean μ_t and maximum absolute deviation $d_{\max} = \max_j |t_j - \mu_t|$.

The complete non-linear mapping is then given by:

$$s = \tanh\left(\alpha \cdot \frac{t - \mu_t}{d_{\max}}\right), \tag{14}$$

where $\alpha > 0$ is a configurable hyperparameter. This data-driven transformation dynamically confines mapped coordinates via a hyperbolic tangent function, adapting to the spread of recent timesteps. Although α enables fine-grained control, a default $\alpha = 0.55$ consistently ensures robust performance across diverse settings. Ultimately, Balanced Mapping delivers a resilient, plug-and-play extrapolation domain robust to numerical outliers.

Error Analysis We quantitatively compare the node-dependent error bound, $\max_t |e(t)|$, with and without phase mapping.

Without Phase Mapping. Without phase mapping, the maximum value of the interpolation error polynomial is:

$$\begin{aligned} |e_{\text{ori}}(t_{\text{target}})| &= \prod_{j=0}^{m} |t_{\text{target}} - t_j| \\ &\leq |e_{\text{ori}}(t_m - N + 1)| = \prod_{j=0}^{m} |(m-j+1)N - 1|. \end{aligned} \tag{15}$$

Chebyshev Mapping. With Chebyshev mapping, the corresponding error polynomial is given by:

$$e(s) = \prod_{j=0}^{m}(s - s_j). \tag{16}$$

The point of maximum error still corresponds to the physical time $t_{\text{target}} = t_m - N + 1$, which is mapped to the coordinate s_{target} in the phase space. Although the extrapolated target coordinate s_{target} slightly exceeds the interval $(-1, 1)$, it maintains a mathematically bounded distance to the historical Chebyshev nodes. By algebraically mapping the physical time differences into the phase space, we derive the rigorous upper bound for the error polynomial's magnitude:

$$|e_{\text{cheby}}(s_{\text{target}})| \leq \prod_{j=0}^{m}\left|2(m - j + 1) - \frac{2}{N}\right|. \tag{17}$$

Balanced Mapping. Similar to Chebyshev Mapping, we have:

$$|e_{\text{bal}}(s_{\text{target}})| \leq \prod_{j=0}^{m}|T_{bal}|, \tag{18}$$

where $T_{bal} = \frac{\sinh\left(\frac{2\alpha(N-1)}{mN}\right)}{\cosh(\alpha)\cdot\cosh\left(\alpha\frac{2-mN-2N}{mN}\right)} + 2(m - j)$.

Comparative Analysis. We now compare the three error bounds. Let us define a function $f(N, j, m)$ as the difference between the terms inside the products. The difference between no mapping and Chebyshev Node Mapping is:

$$f(N, j, m) = (m - j)(N - 2) + N + \frac{2}{N} - 3. \tag{19}$$

In our acceleration task, constraints are $N \geq 2$ and $m - j > 0$. As $f(N, j, m)$ increases monotonically with $N \geq 2$, it follows that $f(N, j, m) > 0$. This demonstrates that Chebyshev Mapping significantly reduces the polynomial's error upper bound. A similar analysis shows Balanced Mapping also lowers the error bound versus the no-mapping baseline.

When comparing the two mapping schemes, extensive empirical evaluations reveal a distinct, task-dependent superiority. Specifically, Chebyshev Mapping proves to be highly effective for class-conditional image generation tasks. In contrast, for highly complex, text-conditioned generation tasks such as text-to-image and text-to-video, our novel Balanced Mapping consistently demonstrates a clear advantage in preserving semantic alignment and visual fidelity.

3.6 Methodology Summary

In conclusion, ResilPhase integrates three core components to construct a noise-resilient acceleration framework: (1) targeting the ODE-aligned Global Drift to bypass spatial cascading errors; (2) a derivative-free Barycentric Lagrange Interpolator to eliminate temporal gradient noise; and (3) a bounded Phase Mapping mechanism to suppress numerical extrapolation instability.

The final predicted macro-trajectory displacement is formulated as:

$$\hat{D}(x_{\text{pred}}) = P(s_{\text{pred}}) = \frac{\sum_{j=0}^{m} \frac{w_j}{s_{\text{pred}} - s_j} D_j}{\sum_{j=0}^{m} \frac{w_j}{s_{\text{pred}} - s_j}}. \tag{20}$$

Ultimately, ResilPhase delivers remarkably stable, high-fidelity extrapolations across diverse generative tasks, maintaining exceptional generation quality even under extreme acceleration regimes.

4 Experiments

4.1 Settings

Model Configurations. We evaluate four mainstream models using a 50-step sampling schedule for fair comparison. For text-to-image, we use FLUX.1-dev [18] with the Rectified Flow [27] sampler and SDXL-base-1.0 [34]. For text-to-video, we use HunyuanVideo-Large [43]. For class-conditional generation, we employ DiT-XL/2 [33] with DDIM [41]. Detailed settings and SDXL analyses are in the Supplementary Material.

Evaluation. We evaluate across three tasks. For the text-to-image task, we use 200 DrawBench [37] prompts, assessing quality and alignment with ImageReward [51] and CLIP Score [9]. For the text-to-video task, we use 946 prompts, evaluating on 16 core dimensions. For both tasks, we also report PSNR, SSIM [48], and LPIPS [54] for fidelity against original results. For the class-conditional task, we generate images from 1,000 ImageNet [36] categories and evaluate using FID-50k [10], sFID, and Inception Score (IS). Further details are in the supplementary material.

4.2 Text-to-Image Generation

As shown in Tab. 1, ResilPhase consistently outperforms competitors in both speed and generation quality, with its advantage widening at higher acceleration ratios. Notably, at an aggressive 4.97x speedup on FLUX.1-dev, ResilPhase maintains an exceptional ImageReward of 1.0258, delivering a $> 64\%$ improvement over the leading forecasting baseline TaylorSeer (0.6241), alongside a 52% lower LPIPS score. This

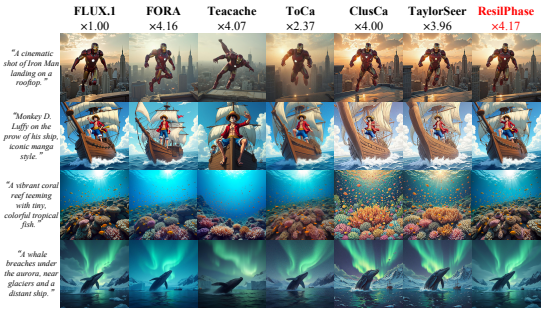


Fig. 4: Qualitative comparison on FLUX.1-dev. While baseline methods suffer from visual artifacts and semantic errors at high speedup ratios, ResilPhase preserves both high fidelity and prompt accuracy, showing its superiority.

Table 1: Quantitative comparison of text-to-image generation for FLUX.1-dev. Methods are compared at similar latency acceleration ratios across three tiers of speedups to evaluate generation quality.

Method FLUX.1-dev	Acceleration Latency(s) ↓	Acceleration Speed ↑	ImageReward ↑ DrawBench	CLIP↑ Score	PSNR↑	SSIM↑	LPIPS↓
[dev]: 50 steps	23.69	1.00×	1.0804	32.711	-	-	-
[dev]: 11 steps	5.21	4.55×	0.9541	32.485	28.397	0.5939	0.5001
FORA ($\mathcal{N} = 6$) [39]	5.20	4.56×	0.8468	32.178	28.230	0.5700	0.5405
TeaCache ($l = 1.4$) [22]	4.92	4.82×	0.7850	32.588	27.954	0.3837	0.8349
ToCa ($\mathcal{N} = 12, R = 90\%$) [63]	9.19	2.58×	0.7284	31.475	28.444	0.5380	0.5768
ClusCa ($\mathcal{N} = 12, O = 2$) [57]	4.88	4.85×	0.4958	30.353	28.079	0.3724	0.7340
SpeCa ($\tau_0 = 12, \beta = 0.3$) [26]	4.96	4.78×	0.9798	32.571	28.366	0.5567	0.5324
PFDiff ($\mathcal{K} = 3, H = 3$) [44]	5.82	4.07×	1.0386	32.816	28.671	0.6162	0.4670
HiCache ($\mathcal{N} = 11, O = 2$) [7]	5.08	4.66×	0.8040	31.604	28.268	0.5261	0.5955
FreqCa ($\mathcal{N} = 6, O = 2$) [23]	4.85	4.88×	1.0130	32.114	28.120	0.4023	0.6877
TaylorSeer ($\mathcal{N} = 11, O = 2$) [25]	5.10	4.65×	0.6241	31.895	27.940	0.3014	0.8012
ResilPhase ($\mathcal{N} = 6, O = 1$)	**4.77**	**4.97×**	**1.0258**	**32.847**	**29.536**	**0.6655**	**0.3834**
[dev]: 13 steps	6.11	3.88×	0.9821	32.586	28.498	0.6136	0.4682
FORA ($\mathcal{N} = 5$) [39]	5.69	4.16×	0.9235	32.370	28.288	0.5694	0.5156
TeaCache ($l = 1.0$) [22]	5.82	4.07×	0.8518	32.750	27.960	0.3832	0.8253
ToCa ($\mathcal{N} = 10, R = 90\%$) [63]	10.00	2.37×	0.8867	32.038	28.601	0.5778	0.5176
ClusCa ($\mathcal{N} = 8, O = 2$) [57]	5.92	4.00×	0.9913	32.410	28.430	0.5528	0.5310
SpeCa ($\tau_0 = 8, \beta = 0.3$) [26]	5.84	4.06×	1.0459	32.764	28.715	0.6176	0.4460
PFDiff ($\mathcal{K} = 3, H = 2$) [44]	5.83	4.06×	1.0442	32.731	28.923	0.6481	0.4223
HiCache ($\mathcal{N} = 7, O = 2$) [7]	5.96	3.97×	1.0079	32.622	28.705	0.6147	0.4542
FreqCa ($\mathcal{N} = 5, O = 2$) [23]	5.76	4.11×	1.0496	32.716	28.132	0.4101	0.6835
TaylorSeer ($\mathcal{N} = 7, O = 2$) [25]	5.98	3.96×	0.9406	32.657	28.049	0.3931	0.7119
ResilPhase ($\mathcal{N} = 5, O = 1$)	**5.68**	**4.17×**	**1.0591**	**32.901**	**29.556**	**0.7024**	**0.3342**
[dev]: 15 steps	7.00	3.38×	1.0044	32.618	28.612	0.6345	0.4375
Δ-DiT ($\mathcal{N} = 10$)	7.60	3.12×	0.1164	30.369	28.157	0.5587	0.6408
FORA ($\mathcal{N} = 4$) [39]	7.50	3.16×	0.9791	32.684	28.342	0.6078	0.4725
TeaCache ($l = 0.7$) [22]	7.44	3.18×	0.9536	32.794	27.961	0.3813	0.8176
ToCa ($\mathcal{N} = 8, R = 90\%$) [63]	10.78	2.20×	0.9537	32.490	28.786	0.5954	0.4764
ClusCa ($\mathcal{N} = 6, O = 2$) [57]	6.77	3.50×	1.0429	32.795	28.813	0.6205	0.4384
SpeCa ($\tau_0 = 4, \beta = 0.3$) [26]	6.78	3.49×	1.0598	**33.108**	29.120	0.6662	0.3786
PFDiff ($\mathcal{K} = 2, H = 1$) [44]	7.66	3.09×	1.0566	32.989	29.557	0.7030	0.3517
HiCache ($\mathcal{N} = 5, O = 2$) [7]	7.34	3.23×	1.0438	32.898	29.240	0.6830	0.3568
FreqCa ($\mathcal{N} = 4, O = 2$) [23]	6.74	3.51×	1.0457	33.041	28.146	0.4086	0.6850
TaylorSeer ($\mathcal{N} = 5, O = 2$) [25]	7.41	3.20×	1.0566	32.811	29.132	0.6701	0.3757
ResilPhase ($\mathcal{N} = 4, O = 1$)	**6.62**	**3.58×**	**1.0647**	32.874	**30.020**	**0.7216**	**0.2989**

– **Note:** ResilPhase results were obtained utilizing Balanced Mapping.

quantitative robustness translates directly to superior visual fidelity. As visually confirmed in Fig. 4, while baseline methods suffer from severe distortions and artifacts at extreme speedups, ResilPhase successfully preserves detailed structures and color accuracy, establishing a new state-of-the-art balance between inference acceleration and exceptional image quality.

4.3 Text-to-Video Generation

As shown in Tab. 2, ResilPhase achieves the optimal speed-quality balance across all acceleration levels. At an aggressive $\sim$5$\times$ speedup, it outperforms the SOTA TeaCache with a higher VBench [12] score (79.78) and a 10% lower LPIPS, an advantage that widens further at the $\sim$4.3$\times$ tier. These quantitative gains translate directly to superior visual fidelity (Fig. 5). While competing methods suffer from severe artifacts like object blur, incorrect scales, and distorted motions at high speeds, ResilPhase consistently delivers smooth, temporally coherent videos with crisp, semantically accurate details.

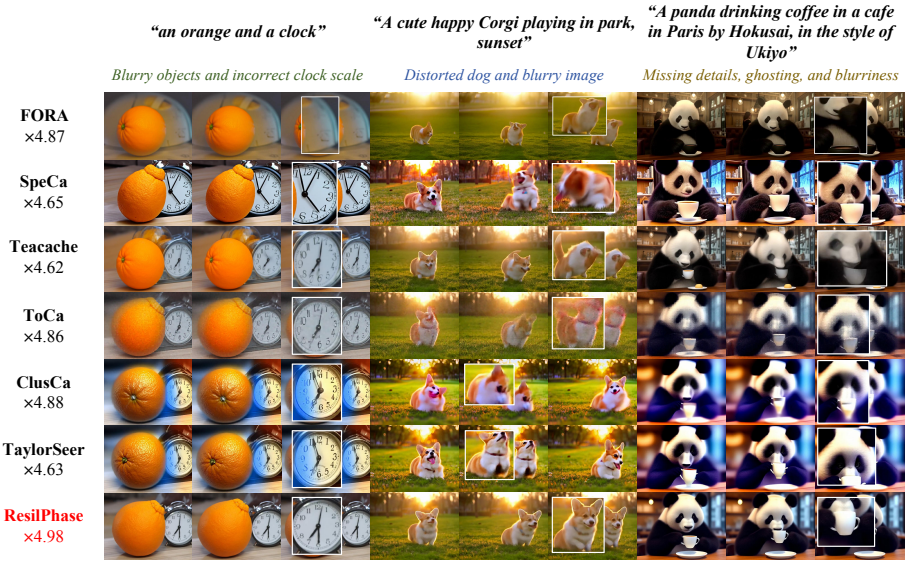


Fig. 5: Qualitative comparison of text-to-video generation methods. While competing methods suffer from object distortion, missing details, and motion inconsistency, ResilPhase maintains superior temporal coherence and visual quality.

Table 2: Quantitative comparison for HunyuanVideo text-to-video generation, grouped by two tiers of similar latency acceleration ratio.

Method	Speed ↑	PSNR ↑	SSIM ↑	LPIPS ↓	VBench ↑
50-steps	1.00×	-	-	-	80.87
FORA ($\mathcal{N} = 6$)	4.87×	15.871	0.6073	0.4584	78.70
TeaCache ($l = 0.4$)	4.62×	17.923	0.6547	0.3760	79.77
ToCa ($\mathcal{N} = 10, R = 90\%$)	4.86×	17.581	0.5857	0.4501	76.37
ClusCa ($\mathcal{N} = 9, O = 1$)	4.88×	14.690	0.5353	0.5139	76.98
SpeCa ($\tau_0 = 1.5, \beta = 0.2$)	4.65×	16.461	0.5883	0.4219	79.59
TaylorSeer ($\mathcal{N} = 7, O = 1$)	4.63×	15.520	0.5641	0.4581	79.07
ResilPhase ($\mathcal{N} = 6, O = 1$)	**4.98×**	**18.920**	**0.6709**	**0.3341**	**79.78**
FORA ($\mathcal{N} = 4$)	3.57×	16.582	0.6244	0.4010	80.10
TeaCache ($l = 0.3$)	4.01×	19.050	0.6846	0.3207	80.38
ToCa ($\mathcal{N} = 7, R = 90\%$)	4.09×	18.299	0.6246	0.3688	79.00
ClusCa ($\mathcal{N} = 6, O = 1$)	4.05×	16.116	0.5881	0.4230	79.74
SpeCa ($\tau_0 = 1.2, \beta = 0.1$)	4.20×	16.488	0.5888	0.4198	79.77
TaylorSeer ($\mathcal{N} = 5, O = 1$)	3.70×	17.117	0.6316	0.3690	80.27
ResilPhase ($\mathcal{N} = 5, O = 1$)	**4.28×**	**19.481**	**0.6992**	**0.2954**	**80.42**

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

Method	Speed ↑	FID ↓	sFID ↓	IS ↑
DDIM-50 steps	1.00×	2.367	4.387	236.06
DDIM-12 steps	4.20×	8.465	8.664	180.41
FORA ($\mathcal{N} = 8$)	3.64×	16.694	23.396	128.08
ToCa ($\mathcal{N} = 12, R = 93\%$)	3.53×	34.110	30.341	85.51
ClusCa ($\mathcal{N} = 12, K = 4, O = 2$)	3.84×	15.206	9.405	124.03
SpeCa ($\tau_0 = 1.5, \beta = 0.5$)	4.36×	6.866	8.250	171.68
TaylorSeer ($\mathcal{N} = 13, O = 2$)	2.72×	15.415	16.005	120.76
ResilPhase ($\mathcal{N} = 5, O = 3$)	**4.41×**	**2.832**	**5.026**	**219.19**
DDIM-20 steps	2.50×	3.858	5.201	216.76
FORA ($\mathcal{N} = 4$)	2.60×	4.770	7.591	210.98
ToCa ($\mathcal{N} = 5, R = 93\%$)	2.67×	6.321	7.027	196.06
ClusCa ($\mathcal{N} = 6, K = 8, O = 4$)	2.61×	3.251	5.207	217.60
SpeCa ($\tau_0 = 0.1, \beta = 0.5$)	2.65×	2.633	4.848	231.32
TaylorSeer ($\mathcal{N} = 8, O = 3$)	2.14×	4.807	7.088	197.22
ResilPhase ($\mathcal{N} = 3, O = 4$)	**2.78×**	**2.347**	**4.672**	**233.57**

4.4 Class-Conditional Image Generation

On DiT-XL/2, ResilPhase uniquely improves quality while accelerating. At a 2.78× speedup, it achieves an FID of 2.347, outperforming both the full 50-step DDIM baseline (2.367) and the SOTA competitor ClusCa (3.251). Its robustness shines at a 4.41× speedup, maintaining a low FID of 2.832 while most caching baselines collapse (FID > 15). Even against the strongest competitor in this tier (SpeCa, FID 6.866), ResilPhase delivers a massive 58% FID reduction. Consistently leading in sFID and IS, ResilPhase proves highly effective at preserving original quality under aggressive acceleration.

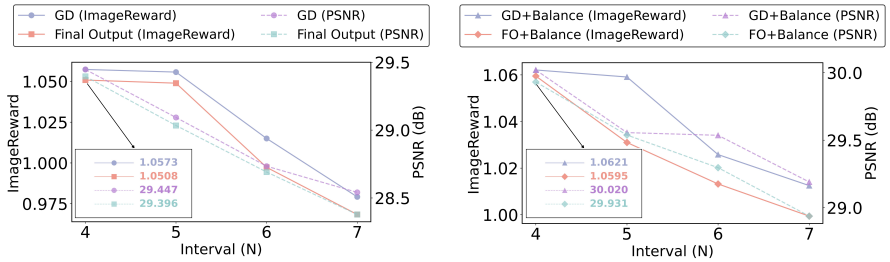
Table 4: Ablation study of ResilPhase components on the FLUX-1.dev.

Configuration	Predictive Objective		Phase Mapping		Speed ↑	ImageReward ↑ DrawBench	CLIP↑ Score	PSNR↑	SSIM↑	LPIPS↓
	Global Drift	Layer-wise Output	Chebyshev	Balance						
		✔			3.08×	1.0546	32.937	29.435	0.6948	0.3323
	✔				3.57×	1.0573	32.929	29.447	0.7003	0.3284
Lagrange		✔	✔		3.08×	1.0559	**32.975**	29.436	0.6947	0.3324
($\mathcal{N}=4, O=1$)		✔		✔	3.08×	1.0580	32.874	29.929	0.7139	0.2995
	✔		✔		3.58×	1.0595	32.914	29.448	0.7004	0.3281
	✔			✔	**3.58×**	**1.0647**	32.918	**30.020**	**0.7216**	**0.2989**
		✔			3.64×	1.0404	32.682	29.013	0.6659	0.3853
	✔				4.17×	1.0557	32.758	29.093	0.6786	0.3808
Lagrange		✔	✔		3.64×	1.0409	32.655	29.014	0.6659	0.3853
($\mathcal{N}=5, O=1$)		✔		✔	3.64×	1.0528	**32.998**	29.472	0.6955	0.3343
	✔		✔		4.17×	1.0517	32.797	29.034	0.6715	0.3811
	✔			✔	**4.17×**	**1.0591**	32.901	**29.556**	**0.7024**	**0.3342**
		✔			4.06×	1.0098	32.803	28.706	0.6184	0.4479
	✔				4.96×	1.0150	32.658	28.734	0.6292	0.4388
Lagrange		✔	✔		4.06×	1.0126	32.688	28.704	0.6183	0.4478
($\mathcal{N}=6, O=1$)		✔		✔	4.05×	**1.0347**	32.707	29.192	0.6526	**0.3831**
	✔		✔		4.97×	1.0186	32.708	28.734	0.6291	0.4388
	✔			✔	**4.97×**	1.0258	**32.847**	**29.536**	**0.6655**	0.3834

– **Note:** Lagrange refers to the baseline configuration utilizing only Barycentric Lagrange extrapolation, without any phase mapping.

4.5 Ablation Study

Ablation Study of ResilPhase Components. Tab. 4 shows forecasting Global Drift outperforms layer-wise prediction by eliminating cascading errors. Phase Mapping further boosts quality, with Balanced Mapping proving more suitable for text-to-image tasks than Chebyshev Mapping. We also compared Global Drift against predicting the Final Output, an alternative avoiding intermediate caching. Fig. 6 confirms Global Drift consistently maintains superiority over the Final Output across extrapolation intervals.



(a) GD vs. FO (w/o Phase Mapping) **(b) GD vs. FO (w/ Phase Mapping)**

Fig. 6: Comparing the performance of Global Drift (GD) and Final Output (FO) predictions across extrapolation intervals (N) with and without Balance Phase Mapping.

Generalizability of the Phase Mapping. Integrating Phase Mapping into TaylorSeer and HiCache consistently boosts ImageReward (Fig. 7) and other perceptual metrics. Furthermore, our appendix provides theoretical proofs that this mechanism strictly reduces their extrapolation error bounds, which strongly confirms its plug-and-play generalizability for existing polynomial accelerators.

5 Conclusion

We introduce ResilPhase, a noise-resilient, training-free acceleration framework for Diffusion Transformers. It overcomes spatial, temporal, and numerical bottlenecks in existing paradigms by synergizing an ODE-aligned Global Drift target to eliminate cascading errors, a derivative-free Barycentric Lagrange extrapolator

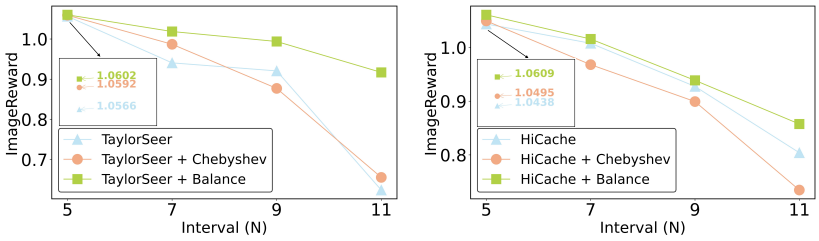


Fig. 7: Comparing the performance of TaylorSeer and HiCache with different phase mapping strategies on FLUX.1-dev across acceleration intervals.

to bypass gradient noise, and a bounded Phase Mapping mechanism to suppress extrapolation instability. Experiments confirm it achieves state-of-the-art speed and quality at aggressive acceleration ratios, establishing a robust real-time inference paradigm and plug-and-play stabilizer for existing accelerators.

Acknowledgements

This study is partially supported by the National Natural Science Foundation of China (Grant No.62504204).

References

1. Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al.: Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127 (2023)
2. Bu, J., Ling, P., Zhou, Y., Wang, Y., Zang, Y., Lin, D., Wang, J.: Dicache: Let diffusion model determine its own cache. arXiv preprint arXiv:2508.17356 (2025)
3. Cai, P., Liu, J., Xu, H., Wang, X., Zou, C., Zhang, L.: Lesa: Learnable stage-aware predictors for diffusion model acceleration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 43300–43309 (2026)
4. Chen, P., Shen, M., Ye, P., Cao, J., Tu, C., Bouganis, C.S., Zhao, Y., Chen, T.: δ-dit: A training-free acceleration method tailored for diffusion transformers. arXiv preprint arXiv:2406.01125 (2024)
5. Chi, H., Qiu, D., Su, H., Liu, H., Li, Z., Zhang, H., Lv, C.: Driver-wm: A driver-centric traffic-conditioned latent world model for in-cabin dynamics rollout (2026), `https://arxiv.org/abs/2605.05092`
6. Fang, G., Ma, X., Wang, X.: Structural pruning for diffusion models (2023), `https://arxiv.org/abs/2305.10924`
7. Feng, L., Zheng, S., Liu, J., Lin, Y., Zhou, Q., Cai, P., Wang, X., Chen, J., Zou, C., Ma, Y., et al.: Hicache: Training-free acceleration of diffusion models via hermite polynomial-based feature caching. arXiv preprint arXiv:2508.16984 (2025)
8. Guo, Y., Yan, Z., Yu, X., Kong, Q., Xie, J., Luo, K., Zeng, D., Wu, Y., Jia, Z., Shi, Y.: Hardware design and the fairness of a neural network. Nature Electronics **7**(8), 714–723 (2024)

9. Hessel, J., Holtzman, A., Forbes, M., Le Bras, R., Choi, Y.: Clipscore: A reference-free evaluation metric for image captioning. In: Proceedings of the 2021 conference on empirical methods in natural language processing. pp. 7514–7528 (2021)
10. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)
11. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)
12. Huang, Z., He, Y., Yu, J., Zhang, F., Si, C., Jiang, Y., Zhang, Y., Wu, T., Jin, Q., Chanpaisit, N., et al.: Vbench: Comprehensive benchmark suite for video generative models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21807–21818 (2024)
13. Jia, J., Liu, Y., Sun, Y., Chen, H., Qin, F., Wang, C., Peng, Y.: Geodesic prototype matching via diffusion maps for interpretable fine-grained recognition. In: ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1786–1790. IEEE (2026)
14. Jia, J., Wang, J., Liu, Y., Jing, H., Wu, Y., Wu, X., Zheng, Y.: This looks distinctly like that: Grounding interpretable recognition in stiefel geometry against neural collapse. arXiv preprint arXiv:2603.08374 (2026)
15. Jia, J., Wu, Y., Chen, H., Jing, H., Wang, H., Bu, J., Wu, L.: Unsupervised causal prototypical networks for de-biased interpretable dermoscopy diagnosis. arXiv preprint arXiv:2602.23752 (2026)
16. Jia, J., Zheng, H., Wu, Y., Chen, H., Wang, H., Bu, J., Wu, L.: SCOUT: Selective coupling via optimal unbalanced transport for interpretable text classification. In: Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 6413–6431. Association for Computational Linguistics (Jul 2026), https://aclanthology.org/2026.acl-long.290/
17. Kim, S., Lee, H., Cho, W., Park, M., Ro, W.W.: Ditto: Accelerating diffusion model via temporal value similarity. In: 2025 IEEE International Symposium on High Performance Computer Architecture (HPCA). pp. 338–352. IEEE (2025)
18. Labs, B.F., Batifol, S., Blattmann, A., Boesel, F., Consul, S., Diagne, C., Dockhorn, T., English, J., English, Z., Esser, P., et al.: Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. arXiv preprint arXiv:2506.15742 (2025)
19. Li, S., Hu, T., van de Weijer, J., Khan, F.S., Liu, T., Li, L., Yang, S., Wang, Y., Cheng, M.M., Yang, J.: Faster diffusion: Rethinking the role of the encoder for diffusion model inference. Advances in Neural Information Processing Systems **37**, 85203–85240 (2024)
20. Li, X., Liu, Y., Lian, L., Yang, H., Dong, Z., Kang, D., Zhang, S., Keutzer, K.: Q-diffusion: Quantizing diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 17535–17545 (2023)
21. Li, Y., Wang, H., Jin, Q., Hu, J., Chemerys, P., Fu, Y., Wang, Y., Tulyakov, S., Ren, J.: Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. Advances in Neural Information Processing Systems **36**, 20662–20678 (2023)
22. Liu, F., Zhang, S., Wang, X., Wei, Y., Qiu, H., Zhao, Y., Zhang, Y., Ye, Q., Wan, F.: Timestep embedding tells: It's time to cache for video diffusion model. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 7353–7363 (2025)

23. Liu, J., Cai, P., Zhou, Q., Lin, Y., Kong, D., Huang, B., Pan, Y., Xu, H., Zou, C., Tang, J., et al.: Freqca: Accelerating diffusion models via frequency-aware caching. arXiv preprint arXiv:2510.08669 (2025)
24. Liu, J., Wang, X., Lin, Y., Wang, Z., Wang, P., Cai, P., Zhou, Q., Yan, Z., Yan, Z., Shi, Z., et al.: A survey on cache methods in diffusion models: Toward efficient multi-modal generation. arXiv preprint arXiv:2510.19755 (2025)
25. Liu, J., Zou, C., Lyu, Y., Chen, J., Zhang, L.: From reusing to forecasting: Accelerating diffusion models with taylorseers. arXiv preprint arXiv:2503.06923 (2025)
26. Liu, J., Zou, C., Lyu, Y., Ren, F., Wang, S., Li, K., Zhang, L.: Speca: Accelerating diffusion transformers with speculative feature caching. In: Proceedings of the 33rd ACM International Conference on Multimedia. pp. 10024–10033 (2025)
27. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. arXiv preprint arXiv:2209.03003 (2022)
28. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. Advances in neural information processing systems **35**, 5775–5787 (2022)
29. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. Machine Intelligence Research pp. 1–22 (2025)
30. Ma, X., Wang, Y., Jia, G., Chen, X., Liu, Z., Li, Y.F., Chen, C., Qiao, Y.: Latte: Latent diffusion transformer for video generation. arXiv e-prints pp. arXiv–2401 (2024)
31. Ma, X., Fang, G., Wang, X.: Deepcache: Accelerating diffusion models for free. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 15762–15772 (2024)
32. Meng, Y., Jin, X., Lei, L., Guo, C.L., Li, C.: Ultraled: Learning to see everything in ultra-high dynamic range scenes. In: NeurIPS. pp. 41466–41495 (2025)
33. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4195–4205 (2023)
34. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952 (2023)
35. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
36. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)
37. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. Advances in neural information processing systems **35**, 36479–36494 (2022)
38. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. arXiv preprint arXiv:2202.00512 (2022)
39. Selvaraju, P., Ding, T., Chen, T., Zharkov, I., Liang, L.: Fora: Fast-forward caching in diffusion transformer acceleration. arXiv preprint arXiv:2407.01425 (2024)
40. Shang, Y., Yuan, Z., Xie, B., Wu, B., Yan, Y.: Post-training quantization on diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1972–1981 (2023)
41. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502 (2020)

42. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency models (2023)
43. Sun, X., Chen, Y., Huang, Y., Xie, R., Zhu, J., Zhang, K., Li, S., Yang, Z., Han, J., Shu, X., et al.: Hunyuan-large: An open-source moe model with 52 billion activated parameters by tencent. arXiv preprint arXiv:2411.02265 (2024)
44. Wang, G., Cai, Y., Li, L., Peng, W., Su, S.: Pfdiff: Training-free acceleration of diffusion models combining past and future scores. arXiv preprint arXiv:2408.08822 (2024)
45. Wang, J., Lin, C., Nie, L., Liao, K., Shao, S., Zhao, Y.: Digging into contrastive learning for robust depth estimation with diffusion models. In: Proceedings of the 32nd ACM International Conference on Multimedia. p. 4129–4137. MM '24, ACM (Oct 2024). https://doi.org/10.1145/3664647.3681168, http://dx.doi.org/10.1145/3664647.3681168
46. Wang, J., Lin, C., Sun, L., Liu, R., Nie, L., Li, M., Liao, K., Chu, X., Zhao, Y.: From editor to dense geometry estimator. arXiv preprint arXiv:2509.04338 (2025)
47. Wang, L., Jin, Z., Hao, Y., Chen, Y., Liu, K., Ao, Y., Zhao, J.: Think while watching: Online streaming segment-level memory for multi-turn video reasoning in multimodal large language models. CoRR **abs/2603.11896** (2026). https://doi.org/10.48550/ARXIV.2603.11896, https://doi.org/10.48550/arXiv.2603.11896
48. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)
49. Wu, Y., Yan, Z., Yin, X., He, L., Zhuo, C.: Anas: Software–hardware co-design of approximate neural network accelerators via neural architecture search. Integration **104**, 102469 (2025)
50. Xia, Z., Li, Y., Tang, S., Fan, Z., Fang, X., Tao, H., Cai, X., Ke, G., Zhang, L., Hong, Y., et al.: Uniem-3m: A universal electron micrograph dataset for microstructural segmentation and generation. arXiv preprint arXiv:2508.16239 (2025)
51. Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., Dong, Y.: Imagereward: Learning and evaluating human preferences for text-to-image generation. Advances in Neural Information Processing Systems **36**, 15903–15935 (2023)
52. Yin, J., Chen, T., Chen, Y., Pei, G., Shu, X., Yao, Y., Shen, F.: Pca-seg: Revisiting cost aggregation for open-vocabulary semantic and part segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 27633–27643 (June 2026)
53. Yin, J., Jiang, X., Chen, T., Pei, G., Yao, Y., Shen, F., Shen, H.T.: Depmatch: Boosting semi-supervised semantic segmentation by exploring depth difference knowledge. IEEE Transactions on Image Processing **35**, 3256–3270 (2026)
54. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018)
55. Zheng, S., Feng, L., Wang, X., Zhou, Q., Cai, P., Zou, C., Liu, J., Lin, Y., Chen, J., Ma, Y., et al.: Forecast then calibrate: Feature caching as ode for efficient diffusion transformers. arXiv preprint arXiv:2508.16211 (2025)
56. Zheng, Z., Peng, X., Yang, T., Shen, C., Li, S., Liu, H., Zhou, Y., Li, T., You, Y.: Open-sora: Democratizing efficient video production for all. arXiv preprint arXiv:2412.20404 (2024)
57. Zheng, Z., Wang, X., Zou, C., Wang, S., Zhang, L.: Compute only 16 tokens in one timestep: Accelerating diffusion transformers with cluster-driven feature caching. In: Proceedings of the 33rd ACM International Conference on Multimedia. pp. 10181–10189 (2025)

58. Zhou, X., Zhang, Y., Sun, S., Wen, C., Yan, Z.: Deploying edge llms for wafer defect detection in chip manufacturing. In: 2025 International Symposium of Electronics Design Automation (ISEDA). pp. 7–11. IEEE (2025)
59. Zhu, C., Lin, Y., Chen, S., Wang, Y., Lin, J.: Medeyes: Learning dynamic visual focus for medical progressive diagnosis. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 40, pp. 13916–13924 (2026)
60. Zhu, C., Lin, Y., Shao, J., Lin, J., Wang, Y.: Pathology-aware prototype evolution via llm-driven semantic disambiguation for multicenter diabetic retinopathy diagnosis. In: Proceedings of the 33rd ACM International Conference on Multimedia. pp. 9196–9205 (2025)
61. Zhu, C., Wang, Y., Lin, J., Wang, F., Wang, H., Zhao, L., Li, S., Li, K.: Anatomy-anchored self-supervision: Distilling vision foundation models for invariant ultrasound representation. MICCAI 2026 (Early Accept) (2026)
62. Zhu, H., Tang, D., Liu, J., Lu, M., Zheng, J., Peng, J., Li, D., Wang, Y., Jiang, F., Tian, L., et al.: Dip-go: A diffusion pruner via few-step gradient optimization. Advances in Neural Information Processing Systems **37**, 92581–92604 (2024)
63. Zou, C., Liu, X., Liu, T., Huang, S., Zhang, L.: Accelerating diffusion transformers with token-wise feature caching. arXiv preprint arXiv:2410.05317 (2024)