TAYLORGAP: PROGRESSIVE ANCHOR SCHEDULING AND GEOMETRY-AWARE TAYLOR FORECASTING FOR TRAINING-FREE DIFFUSION ACCELERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Feature caching accelerates diffusion and flow-matching models by forecasting intermediate features at timesteps that skip a backbone evaluation. TaylorSeer estimates feature derivatives from cached full evaluations, but its fixed refresh interval commits to a full-length forecast block while only the minimum history is available; for a second-order predictor, increasing the offset from one to five enlarges the horizon-dependent factor in the Taylor remainder bound by $5^3 = 125$. The subsequent transition from unit-spaced startup anchors to the wider refresh interval also creates a non-uniform anchor grid. On such a grid, adjacent lower-order factors have order-dependent effective centers whose separation generally differs from the latest anchor interval, so normalizing the recursive higher-order differences by that interval leaves those factors persistently mis-scaled. Based on these observations, we propose TaylorGAP, a training-free method that couples progressive anchor scheduling with grid-aware Taylor forecasting. Its progressive schedule uses anchor intervals $\{1, 2, 3, \ldots\}$ to delay long extrapolations while preserving a fixed budget of ten full evaluations in a 50-step trajectory. Its geometry-aware recursion normalizes each order by the corresponding effective-center distance; on the progressive grid, this yields the closed-form coefficient $\lambda_{r,k} = 2k/(2k - r + 1)$. The correction is applied only at offsets one and two for stability. At a matched NFE budget, TaylorGAP improves consistently over equal-NFE TaylorSeer for orders two through four on both FLUX.1-dev and HunyuanVideo. Relative to the corresponding baseline, it reduces LPIPS by 28.69%–32.35% on images and 41.00%–43.00% on videos, raises PSNR by 18.28%–19.21% and 26.17%–28.64%, and raises SSIM by 12.23%–13.77% and 13.54%–15.29%. TaylorGAP thus provides a principled, training-free connection between cache scheduling and feature forecasting.

1 INTRODUCTION

Diffusion models have become a central paradigm for high-fidelity visual generation, spanning image synthesis and video generation (Ho et al., 2020; Rombach et al., 2022), and their transition from convolutional U-Nets to Diffusion Transformers (DiTs) has further improved scalability (Peebles & Xie, 2023). This progress comes at a substantial inference cost: generating a single sample requires repeatedly evaluating a large denoising or flow-matching backbone over many timesteps, even though adjacent evaluations produce highly redundant intermediate representations. Existing acceleration methods attack this cost through fewer sampling steps, model compression, token sparsification, or feature caching; among them, feature caching is particularly appealing because it skips expensive internal computations without retraining and composes with the other techniques. Its central challenge is therefore not whether computation can be skipped, but how the omitted features can be approximated without deviating from the original generation trajectory. Early caching methods exploit temporal redundancy through a 'cache-then-reuse' strategy, directly substituting previously computed activations at subsequent timesteps (Ma et al., 2024; Chen et al., 2024; Selvaraju et al., 2024); direct reuse, however, degrades as the cache interval grows. TaylorSeer moves from reuse to forecasting, estimating temporal derivatives from cached features by finite differences and applying a truncated Taylor expansion at skipped timesteps (Liu et al., 2025). Subsequent work improves

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.

how missing features are estimated through stable numerical integration, input–output relational calibration, stage-aware learned predictors, rational or spectral approximation, and subspace-aware modeling (Zheng et al., 2026; Son et al., 2026; Cai et al., 2026; Cui et al., 2026a; Han et al., 2026; Chen et al., 2026), or when to refresh using trajectory indicators, error proxies, global path planning, online probes, or residual-magnitude budgets (Cui et al., 2026b; Bu et al., 2026; Ma et al., 2026). These two lines of work treat forecasting and scheduling as separate components.

This modular separation overlooks a structural coupling. TaylorSeer recursively constructs each higher-order factor by differencing the preceding one and normalizing by the latest anchor interval q_k, which is consistent only when anchors are uniformly spaced; on a non-uniform grid, histories of different orders correspond to different effective centers, so q_k mis-scales the higher-order terms. The coupling surfaces most sharply in the startup regime. First, a fixed refresh interval commits to the longest extrapolation immediately after the minimum history: with $\mathcal{N} = 6$, TaylorSeer fully evaluates steps $0, 1, 2$ and then predicts steps 3–7 from the anchor at $s_k = 2$. For an order-m predictor whose remainder satisfies $|\mathbf{R}_{m+1}(x)| \leq M_{m+1}|x|^{m+1}/(m+1)!$ with $M_{m+1} = \sup_{\xi\in[s_k,s_k+x]} \|\mathbf{F}^{(m+1)}(\xi)\|$, the offsets $x = 1$ and $x = 5$ at $m = 2$ give horizon-dependent bound factors differing by $(5/1)^3 = 125$ under a common envelope; since every predicted feature enters the next latent update, such early deviations can propagate along the trajectory. Second, the transition from unit-spaced startup anchors to the fixed interval creates exactly the non-uniform grid described above, leaving the higher-order terms geometrically mis-scaled while the history is still limited.

We address these two limitations with TaylorGAP, a training-free extension of TaylorSeer that couples progressive cache placement with grid-consistent forecasting. Starting from $s_0 = 0$, its scheduler increases successive refresh intervals as $1, 2, 3, \ldots$, yielding anchors $0, 1, 3, 6, 10, \ldots$; on a 50-step trajectory this retains ten full evaluations while shifting budget toward the early stage, avoiding an immediate five-step extrapolation. For the resulting grid, where $q_k = k$, we derive the order-dependent effective-center distance $d_{r,k} = k - (r-1)/2$ and the closed-form correction $\lambda_{r,k} = q_k/d_{r,k} = 2k/(2k-r+1)$; because the order-$r$ contribution scales with the r-th power of the forecast distance, and an unrestricted correction would magnify finite-difference noise, the correction is applied only at offsets $x \in \{1, 2\}$, reverting to the original TaylorSeer formula for $x \geq 3$. TaylorGAP requires no auxiliary training and leaves the generative backbone unchanged.

We evaluate TaylorGAP against the 50-step full-computation baseline and equal-NFE TaylorSeer on FLUX.1-dev (200 DrawBench prompts) and HunyuanVideo (944 VBench prompts, 4,720 videos). On images it improves ImageReward, CLIP score, PSNR, SSIM, and LPIPS over equal-NFE TaylorSeer ($\mathcal{N} = 6$) at orders two through four; at order two it gains 3.06 dB in PSNR and reduces LPIPS by 0.132. On videos it improves the total VBench score by 0.72 points over equal-NFE TaylorSeer ($\mathcal{N} = 5$) at every tested order while remaining within 0.22 points of the baseline, and paired fidelity comparisons show PSNR gains of 4.30–4.70 dB and LPIPS reductions of 0.172–0.180. Our contributions are threefold: (i) a Taylor-remainder analysis quantifying horizon-dependent truncation-error growth, together with a non-uniform finite-difference analysis exposing the effective-center mismatch in TaylorSeer's startup regime (Section 2.1); (ii) the two components derived from these analyses, a progressive anchor schedule and a localized geometry-aware correction with closed-form, order-dependent coefficients (Sections 2.2–2.3); and (iii) matched-NFE validation on both image and video generation (Section 3).

2 METHOD

We develop TaylorGAP in three steps. Section 2.1 analyzes the startup regime of TaylorSeer and isolates two coupled numerical errors: the horizon-dependent growth of the Taylor truncation-error scale, and a geometric scale error that the recursive finite-difference normalization introduces once anchor intervals become non-uniform. The two errors then determine the two components of TaylorGAP: Section 2.2 derives the progressive anchor schedule that keeps early forecast horizons short, and Section 2.3 derives the geometry-aware correction that rescales higher-order factors on the resulting grid, localized to short offsets for stability.

2.1 NUMERICAL LIMITATIONS OF TAYLORSEER

We index the denoising trajectory by a monotonically increasing sampling index s, independently of the direction or spacing of the underlying continuous noise-time parameter. Let $\mathbf{F}(s)$ denote a feature tensor produced by a full evaluation of the generative backbone, and let $s_0 < s_1 < \cdots < s_K$ be the full-computation anchors. We write $q_k = s_k - s_{k-1}$ for the latest anchor interval. In the derivative-scaled form analyzed here, TaylorSeer recursively constructs finite-difference factors from the cached features (Liu et al., 2025):

$$\mathbf{D}_k^{(0)} = \mathbf{F}(s_k), \qquad \mathbf{D}_k^{(r)} = \frac{\mathbf{D}_k^{(r-1)} - \mathbf{D}_{k-1}^{(r-1)}}{q_k}, \quad r \geq 1. \tag{1}$$

When every anchor interval equals N, equation 1 is algebraically equivalent to TaylorSeer's raw r-th finite difference divided by N^r. The distinction becomes consequential only when the intervals contained in the recursive history are unequal. Given an offset $x > 0$ from the latest anchor, its order-m predictor is

$$\widehat{\mathbf{F}}_{\mathrm{TS}}(s_k + x) = \sum_{r=0}^{m} \frac{x^r}{r!} \mathbf{D}_k^{(r)}. \tag{2}$$

The first limitation is the long extrapolation imposed at startup by a fixed refresh interval. TaylorSeer initially performs several consecutive full evaluations to construct its history and then switches immediately to the target interval N. For the $N = 6$ configuration used in our matched-NFE comparison, its full-computation anchors begin with

$$0,\ 1,\ 2,\ 8,\ 14, \ldots. \tag{3}$$

After evaluating only steps $0, 1, 2$, the method predicts steps $3, \ldots, 7$ from the anchor at $s_k = 2$; the forecast offsets are therefore $x = 1, \ldots, 5$. To isolate the effect of the extrapolation horizon, suppose that $\mathbf{F}$ admits an $(m+1)$-times differentiable local extension in the sampling index. Its order-m Taylor expansion about s_k is

$$\mathbf{F}(s_k + x) = \mathbf{F}(s_k) + x\mathbf{F}^{(1)}(s_k) + \frac{x^2}{2!}\mathbf{F}^{(2)}(s_k) + \cdots + \frac{x^m}{m!}\mathbf{F}^{(m)}(s_k) + \mathbf{R}_{m+1}(x), \tag{4}$$

where $\mathbf{R}_{m+1}(x)$ denotes the truncation error after retaining terms through order m and there exists $\xi_x \in (s_k, s_k + x)$ such that

$$\mathbf{R}_{m+1}(x) = \frac{\mathbf{F}^{(m+1)}(\xi_x)}{(m+1)!} x^{m+1}. \tag{5}$$

Under the same smoothness assumption, the norm form of Taylor's theorem gives, for the chosen tensor norm,

$$\|\mathbf{R}_{m+1}(x)\| \leq \frac{M_{m+1}(x)}{(m+1)!} |x|^{m+1}. \tag{6}$$

Here,

$$M_{m+1}(x) = \sup_{\xi \in [s_k, s_k + x]} \left\| \mathbf{F}^{(m+1)}(\xi) \right\|. \tag{7}$$

For second-order prediction, take a common envelope $M_3^\star = \sup_{\xi \in [2,7]} \|\mathbf{F}^{(3)}(\xi)\|$. The two endpoints of the startup prediction block then obey

$$\|\mathbf{R}_3(1)\| \leq \frac{M_3^\star}{6}, \qquad \|\mathbf{R}_3(5)\| \leq \frac{125 M_3^\star}{6}. \tag{8}$$

Thus, the horizon-dependent factors in the two theoretical upper bounds differ by

$$\frac{5^3}{1^3} = 125. \tag{9}$$

This ratio does not claim that the realized feature error at step 7 is exactly, or universally at most, 125 times the error at step 3. It isolates the cubic amplification of the theoretical truncation-error scale contributed by the forecast horizon while holding the derivative envelope fixed. Consequently, committing to the longest reuse interval immediately after the minimum startup history imposes the largest local truncation-error scale in the earliest prediction block. Because every predicted feature

participates in the update of the next latent state, an early deviation is also passed into subsequent denoising steps and can influence later predictions. This sequential dependence motivates controlling early truncation error, without requiring the stronger assumption that the realized error grows monotonically.

TaylorSeer's second limitation is a finite-difference geometry mismatch during the mixed-spacing startup regime. Here, *mixed spacing* refers to the transition from history initialization to fixed-interval reuse. TaylorSeer first performs consecutive full evaluations, producing unit-spaced startup anchors, and then switches directly to the target refresh interval N. For example, with $N = 6$ and a second-order history, the anchor triple $1, 2, 8$ gives $q_{k-1} = 2 - 1 = 1$ and $q_k = 8 - 2 = 6$. The same recursive update therefore combines lower-order factors constructed on different interval lengths. This transient mixed-spacing regime lasts until all anchor intervals required by the current difference table have been replaced by the fixed interval N. Once all anchor intervals involved in the current difference table become equal to N, the effective separation between adjacent lower-order factors also equals $q_k = N$, making TaylorSeer's normalization geometrically consistent. During the mixed-spacing transition, however, this effective separation generally differs from q_k, so dividing by q_k mis-scales the resulting higher-order factor. Consider one scalar component of $\mathbf{F}$. The first-order factors on two adjacent intervals are the secant slopes

$$D_k^{(1)} = \frac{F(s_k) - F(s_{k-1})}{q_k}, \qquad D_{k-1}^{(1)} = \frac{F(s_{k-1}) - F(s_{k-2})}{q_{k-1}}. \tag{10}$$

To compare the accuracy of the two second-order factors directly, assume that $F \in C^5$ locally and expand both slopes about s_k:

$$\begin{aligned} D_k^{(1)} &= F^{(1)}(s_k) - \frac{q_k}{2}F^{(2)}(s_k) + \frac{q_k^2}{6}F^{(3)}(s_k) - \frac{q_k^3}{24}F^{(4)}(s_k) + \mathcal{O}(q_k^4), \\ D_{k-1}^{(1)} &= F^{(1)}(s_k) - \frac{2q_k + q_{k-1}}{2}F^{(2)}(s_k) + \frac{3q_k^2 + 3q_kq_{k-1} + q_{k-1}^2}{6}F^{(3)}(s_k) \\ &\quad - \frac{1}{24}\big(4q_k^3 + 6q_k^2q_{k-1} + 4q_kq_{k-1}^2 + q_{k-1}^3\big)F^{(4)}(s_k) + \mathcal{O}\big((q_k + q_{k-1})^4\big). \end{aligned} \tag{11}$$

Subtracting the two equations gives

$$\begin{aligned} D_k^{(1)} - D_{k-1}^{(1)} &= \frac{q_k + q_{k-1}}{2}F^{(2)}(s_k) - \frac{q_k + q_{k-1}}{6}(2q_k + q_{k-1})F^{(3)}(s_k) \\ &\quad + \frac{q_k + q_{k-1}}{24}\big(3q_k^2 + 3q_kq_{k-1} + q_{k-1}^2\big)F^{(4)}(s_k) + \mathcal{O}(q_k^4) + \mathcal{O}\big((q_k + q_{k-1})^4\big). \end{aligned} \tag{12}$$

TaylorSeer divides this difference by q_k:

$$\begin{aligned} D_{k,\mathrm{TS}}^{(2)} &:= \frac{D_k^{(1)} - D_{k-1}^{(1)}}{q_k}, \\ D_{k,\mathrm{TS}}^{(2)} - F^{(2)}(s_k) &= \underbrace{\frac{q_{k-1} - q_k}{2q_k}F^{(2)}(s_k)}_{\text{geometric scale error}} - \frac{q_k + q_{k-1}}{6q_k}(2q_k + q_{k-1})F^{(3)}(s_k) \\ &\quad + \frac{q_k + q_{k-1}}{24q_k}\big(3q_k^2 + 3q_kq_{k-1} + q_{k-1}^2\big)F^{(4)}(s_k) + \mathcal{O}(q_k^3) + \mathcal{O}\left(\frac{(q_k + q_{k-1})^4}{q_k}\right), \end{aligned} \tag{13}$$

where the adjacent-step ratio q_{k-1}/q_k is assumed bounded in the asymptotic expression. The first term is a scale error created by the denominator. If $q_{k-1}/q_k \to r \neq 1$ as both intervals shrink, it converges to $\frac{r-1}{2}F^{(2)}(s_k)$ rather than zero. TaylorSeer's second-order factor is therefore not a consistent approximation of $F^{(2)}(s_k)$ under this non-uniform refinement.

Importantly, equation 12 shows that this non-vanishing term arises from the mismatch between the denominator q_k and the leading-order separation of the two first-order factors. Normalizing their difference by the corresponding effective separation can eliminate the persistent $F^{(2)}(s_k)$ scale term. We defer the derivation of this correction, its extension to higher orders, and its truncation-error analysis to Section 2.3.

In summary, TaylorSeer's startup regime presents two distinct but coupled numerical limitations. First, its fixed refresh interval imposes the longest forecast offsets immediately after only the minimum history has been collected, while the order-m Taylor truncation-error scale grows as $|x|^{m+1}$; for a second-order predictor, increasing the offset from $x = 1$ to $x = 5$ changes this horizon-dependent factor by 125. Second, the transition from unit-spaced initialization anchors to the target refresh interval produces a mixed-spacing difference table in which the recursive denominator no longer matches the geometry of the lower-order factors being differenced, yielding a non-vanishing scale error. The first limitation concerns where the fixed budget of full evaluations should be allocated, whereas the second concerns how the cached Taylor factors should be constructed. Sections 2.2 and 2.3 address these issues through the two TaylorGAP components, progressive anchor scheduling and geometry-aware factor correction, respectively.

2.2 Progressive Anchor Scheduling

The truncation analysis in Section 2.1 indicates that the longest forecast horizons should not be assigned immediately after the minimum history becomes available. Therefore, the fixed refresh interval is replaced with a deterministic progressive schedule. Let T denote the number of sampling steps, indexed by $0, \ldots, T-1$, and let s_k be the k-th full-computation anchor. Starting from $s_0 = 0$, we set

$$q_k := s_k - s_{k-1} = k, \qquad s_k = s_{k-1} + k = \frac{k(k+1)}{2}, \qquad k \geq 1. \tag{14}$$

Thus, the successive refresh intervals are $1, 2, 3, \ldots$, with no preset upper cap. The finite sampling trajectory itself terminates the sequence. For the $T = 50$ trajectory used in our experiments, equation 14 gives

$$\mathcal{S}_{\text{prog}} = \{0, 1, 3, 6, 10, 15, 21, 28, 36, 45\}, \tag{15}$$

and hence ten full evaluations. This matches the ten full evaluations of the fixed-$N = 6$ TaylorSeer configuration considered in Section 2.1, but redistributes them along the trajectory.

More specifically, let $x = t - s_k$ denote the offset of a prediction step t from the latest anchor s_k. Since the next anchor is $s_{k+1} = s_k + k + 1$, the intervening prediction steps have offsets $x = 1, \ldots, k$, unless the sampling trajectory terminates earlier. Longer extrapolations are introduced only after more full evaluations have been accumulated. In contrast to a data-dependent scheduler, the complete anchor grid is known before sampling and requires neither auxiliary probes nor online error estimation. At every s_k, the backbone is evaluated normally and the feature histories are updated. At all intervening positions, the required block features are forecast from the cache. The schedule changes only the placement of these full evaluations and leaves the underlying sampling trajectory and generative backbone unchanged.

2.3 Localized Order-Aware Geometric Correction

The progressive schedule deliberately creates a non-uniform anchor grid, so the Taylor factors must respect the geometry identified in Section 2.1. We first derive a geometry-consistent recursive construction and then localize its use to short forecast horizons.

Geometry-consistent recursive factors. We continue with one scalar component F; all operations are applied elementwise to the cached feature tensors. Let $c_{0,k} = s_k$. For $r \geq 1$, define the leading-order effective center of the r-th factor recursively by

$$c_{r,k} := \frac{c_{r-1,k} + c_{r-1,k-1}}{2}. \tag{16}$$

When constructing an order-r factor, the two order-$(r-1)$ quantities being differenced are therefore separated by

$$d_{r,k} := c_{r-1,k} - c_{r-1,k-1}. \tag{17}$$

For $r = 2$, equation 17 reduces to

$$d_{2,k} = \frac{q_k + q_{k-1}}{2}. \tag{18}$$

This is exactly the leading coefficient exposed by equation 12. Accordingly, the corrected second-order factor is

$$\widetilde{D}_k^{(2)} := \frac{D_k^{(1)} - D_{k-1}^{(1)}}{d_{2,k}} = \underbrace{\frac{q_k}{d_{2,k}}}_{\lambda_{2,k}} D_{k,\mathrm{TS}}^{(2)} = \frac{2q_k}{q_k + q_{k-1}} D_{k,\mathrm{TS}}^{(2)}. \tag{19}$$

The same construction extends recursively. We retain TaylorSeer's first-order factor, $\widetilde{D}_k^{(1)} = D_k^{(1)}$, and for every available order $r \geq 2$ define

$$\lambda_{r,k} := \frac{q_k}{d_{r,k}}, \widetilde{D}_k^{(r)} := \frac{\widetilde{D}_k^{(r-1)} - \widetilde{D}_{k-1}^{(r-1)}}{d_{r,k}} = \lambda_{r,k} \frac{\widetilde{D}_k^{(r-1)} - \widetilde{D}_{k-1}^{(r-1)}}{q_k}. \tag{20}$$

The recursion is essential: for $r \geq 3$, the order-r factor must be built from two already corrected order-$(r-1)$ factors. Independently multiplying each raw TaylorSeer factor by $\lambda_{r,k}$ would retain lower-order scale errors inside the higher-order difference table.

On the progressive grid in equation 14, repeated application of Eqs. equation 16–equation 17 gives

$$d_{r,k} = k - \frac{r-1}{2}, \qquad \lambda_{r,k} = \frac{k}{k-(r-1)/2} = \frac{2k}{2k-r+1}. \tag{21}$$

For example, the first available coefficients are $\lambda_{2,2} = 4/3$, $\lambda_{3,3} = 3/2$, and $\lambda_{4,4} = 8/5$. For any fixed order r, $\lambda_{r,k} > 1$ in the progressive regime and approaches one as k increases. The correction is therefore strongest when adjacent intervals differ most in relative terms and decays automatically as the grid becomes locally closer to uniform.

Second-order error analysis. Substituting equation 19 into equation 12 yields

$$\begin{aligned} \widetilde{D}_k^{(2)} - F^{(2)}(s_k) = &-\frac{2q_k + q_{k-1}}{3} F^{(3)}(s_k) + \frac{3q_k^2 + 3q_k q_{k-1} + q_{k-1}^2}{12} F^{(4)}(s_k) \\ &+ \mathcal{O}\left(\frac{q_k^4}{q_k + q_{k-1}}\right) + \mathcal{O}\big((q_k + q_{k-1})^3\big). \end{aligned} \tag{22}$$

Unlike equation 13, this expansion contains no term proportional to $F^{(2)}(s_k)$. Under a bounded adjacent-step ratio, the leading remaining term is $\mathcal{O}(q_k + q_{k-1})$; hence the corrected factor is first-order consistent as the local grid is refined. The correction removes the persistent geometric scale error, but it does not remove the ordinary $F^{(3)}$, $F^{(4)}$, and higher-order truncation terms. The removal of the curvature term is in fact exact for quadratic trajectories: on such trajectories the corrected second-order factor reproduces $F^{(2)}$ exactly, and the corrected predictor retains only the first-order secant error that it shares with TaylorSeer. A worked example with the resulting closed-form prediction errors is given in Appendix A.

Localized prediction. Although geometric normalization removes the leading scale mismatch, it can increase the magnitude of higher-order factors because $\lambda_{r,k} > 1$. Moreover, the contribution of an order-r factor to a forecast is weighted by $x^r/r!$. For a fixed factor perturbation, its influence therefore grows polynomially with the forecast distance. We consequently use the corrected factors only at the first two positions after an anchor:

$$\widehat{\mathbf{F}}_{\mathrm{corr}}(s_k + x) = \mathbf{D}_k^{(0)} + x\mathbf{D}_k^{(1)} + \sum_{r=2}^{m} \frac{x^r}{r!} \widetilde{\mathbf{D}}_k^{(r)}. \tag{23}$$

The complete predictor is the hard switch

$$\widehat{\mathbf{F}}(s_k + x) = \begin{cases} \widehat{\mathbf{F}}_{\mathrm{corr}}(s_k + x), & x \in \{1, 2\} \text{ and corrected factors are available,} \\ \widehat{\mathbf{F}}_{\mathrm{TS}}(s_k + x), & \text{otherwise.} \end{cases} \tag{24}$$

Thus, positions $x \geq 3$ use the complete original TaylorSeer formula rather than a mixture of raw and corrected high-order terms. Since both the progressive schedule and the geometric correction coefficients are determined analytically from the anchor indices, TaylorGAP requires no auxiliary training or data-dependent probing and can be integrated into TaylorSeer without modifying the underlying generative model or sampling trajectory.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETTINGS

Model Configurations. The experiments are conducted on two visual generative models: the text-to-image generation model FLUX.1-dev (Black Forest Labs, 2024) and the text-to-video generation model HunyuanVideo (Sun et al., 2024). **FLUX.1-dev** predominantly employs the Rectified Flow sampling method with 50 steps as the standard configuration. All experimental evaluations of FLUX.1-dev were conducted on a single NVIDIA B200 GPU. **HunyuanVideo** is evaluated with its standard 50-step flow-matching inference protocol while preserving the remaining default sampling parameters for rigorous experimental consistency. All HunyuanVideo experiments were conducted on a single NVIDIA B200 GPU.

Evaluation and Metrics. For the text-to-image generation task, we perform inference on 200 DrawBench prompts to generate images with a resolution of 1024×1024. We then evaluate the generated samples using ImageReward and CLIP Score as key metrics to assess image quality and text alignment. For the text-to-video generation task, we leverage the VBench evaluation framework, utilizing its 944 unique benchmark prompts. For each prompt, we generate five samples with different random seeds at a resolution of 480×640 with 65 frames, totaling 4,720 videos. We then systematically assess the generated results based on the 16 core evaluation dimensions defined by the VBench framework to provide a comprehensive evaluation of the model's performance. We assess the fidelity of our accelerated outputs against the original full-computation results using PSNR, SSIM, and LPIPS.

3.2 TEXT-TO-IMAGE GENERATION

We compare TaylorGAP with the original TaylorSeer on FLUX.1-dev, as shown in Table 1. TaylorGAP aims to correct TaylorSeer's feature forecasting so that the accelerated outputs remain closer to the full 50-step baseline. Table 1 additionally reports FORA (Selvaraju et al., 2024) with $\mathcal{N} = 3$, as a traditional approach for comparison. At the same computational cost as TaylorSeer with $\mathcal{N} = 6$, TaylorGAP improves all five quality metrics for $\mathcal{O} \in \{2, 3, 4\}$. To be specific, PSNR increases by 2.9015–3.0586 dB (18.28%–19.21%), SSIM increases by 0.0749–0.0846 (12.23%–13.77%), and LPIPS decreases by 0.1176–0.1317 (28.69%–32.35%). This advantage does not rely on a favorable budget: against the less aggressive TaylorSeer with $\mathcal{N} = 4$, which consumes about 40% more computation (834.095 T FLOPs versus 596.107 T), TaylorGAP still wins eight of the nine included metrics at every order (win rate $8/9$), trailing only on ImageReward by at most 0.0066. The consistent improvements in PSNR, SSIM, and LPIPS show that our modified predictor produces outputs substantially closer to the full-computation baseline. Among the evaluated orders, $\mathcal{O} = 2$ achieves the best overall trade-off, providing a $3.548\times$ latency speedup and a $4.992\times$ FLOPs speedup with the highest PSNR, SSIM, and lowest LPIPS. To complement the quantitative evaluation, we further present generation results for several representative DrawBench prompts, enabling a direct visual comparison between TaylorGAP and TaylorSeer, shown in Figure 1.

3.3 TEXT-TO-VIDEO GENERATION

We further compare TaylorGAP with the original TaylorSeer on HunyuanVideo, as shown in Table 2. At the same computational cost as TaylorSeer with $\mathcal{N} = 5$ (identical 2807.06 T FLOPs and $5.00\times$ FLOPs-based speedup), it improves all four quality metrics for $\mathcal{O} \in \{2, 3, 4\}$. Specifically, the VBench score increases by 0.72 percentage points ($+0.90\%$) at every order, bringing it to within 0.22 points of the full 50-step baseline (81.17), while PSNR increases by 4.30–4.70 dB (26.17%–28.64%), SSIM by 0.0854–0.0964 (13.54%–15.29%), and LPIPS decreases by 0.1718–0.1804 (41.00%–43.00%). The large gains in PSNR, SSIM, and LPIPS show that our predictor reproduces the full-computation video far more faithfully, and the higher VBench score confirms that this fidelity does not come at the expense of perceptual video quality. TaylorGAP also dominates the less aggressive TaylorSeer with $\mathcal{N} = 4$, which uses about 30% more computation (3649.17 T FLOPs): it wins all eight included acceleration and quality metrics ($8/8$) at every order. Among the evaluated orders, $\mathcal{O} = 2$ again achieves the best trade-off, providing a $3.13\times$ latency speedup and a $5.00\times$ FLOPs speedup together with the highest PSNR, SSIM, and the lowest LPIPS.

Table 1: **Quantitative comparison of text-to-image generation** with FLUX.1-dev on DrawBench.

Method	Acceleration				ImageReward↑	CLIP↑	PSNR↑	SSIM↑	LPIPS↓	Win rate (Ours vs. row)
	Latency (s)↓	Speed↑	FLOPs (T)↓	Speed↑						
Baseline: 50 steps	7.583	1.000×	2975.992	1.000×	0.9897	27.4994	—	—	—	—
FORA ($\mathcal{N}=3$)	2.819	2.691×	1012.587	2.939×	0.9336	27.6976	15.6613	0.6328	0.3946	9/9
TaylorSeer ($\mathcal{N}=4,\mathcal{O}=2$)	2.735	2.773×	834.095	3.568×	0.9727	27.5259	17.7385	0.6853	0.3041	8/9
TaylorSeer ($\mathcal{N}=5,\mathcal{O}=2$)	2.439	3.109×	715.101	4.162×	0.9867	27.5694	16.7859	0.6583	0.3507	8/9
TaylorSeer ($\mathcal{N}=6,\mathcal{O}=2$)	2.164	3.504×	596.107	4.992×	0.9603	27.4936	15.9227	0.6144	0.4071	5/5
TaylorGAP ($\mathcal{O}=2$)	**2.137**	**3.548**×	**596.107**	**4.992**×	**0.9661**	**27.8601**	**18.9813**	**0.6990**	**0.2754**	—
TaylorSeer ($\mathcal{N}=4,\mathcal{O}=3$)	2.838	2.672×	834.095	3.568×	0.9746	27.5021	17.7044	0.6842	0.3057	8/9
TaylorSeer ($\mathcal{N}=5,\mathcal{O}=3$)	2.552	2.972×	715.101	4.162×	0.9934	27.5997	16.7387	0.6568	0.3536	8/9
TaylorSeer ($\mathcal{N}=6,\mathcal{O}=3$)	2.270	3.340×	596.107	4.992×	0.9667	27.4974	15.8643	0.6126	0.4097	5/5
TaylorGAP ($\mathcal{O}=3$)	**2.263**	**3.351**×	**596.107**	**4.992**×	**0.9725**	**27.8734**	**18.7956**	**0.6897**	**0.2879**	—
TaylorSeer ($\mathcal{N}=4,\mathcal{O}=4$)	2.940	2.580×	834.095	3.568×	0.9755	27.5354	17.7077	0.6841	0.3058	8/9
TaylorSeer ($\mathcal{N}=5,\mathcal{O}=4$)	2.642	2.870×	715.101	4.162×	0.9914	27.6347	16.7438	0.6568	0.3535	8/9
TaylorSeer ($\mathcal{N}=6,\mathcal{O}=4$)	2.347	3.231×	596.107	4.992×	0.9632	27.5230	15.8717	0.6125	0.4099	5/5
TaylorGAP ($\mathcal{O}=4$)	**2.383**	**3.182**×	**596.107**	**4.992**×	**0.9727**	**27.8655**	**18.7732**	**0.6874**	**0.2923**	—

• **Win rate:** Winning metrics / total included metrics for Ours versus each row within the same order block, following the arrow directions. Acceleration metrics (latency, speedup, FLOPs, and FLOPs speedup) with relative differences of at most 3% ($|x_{\text{ours}} - x_{\text{row}}|/|x_{\text{row}}| \leq 0.03$) are excluded from both counts. All five quality metrics are included.

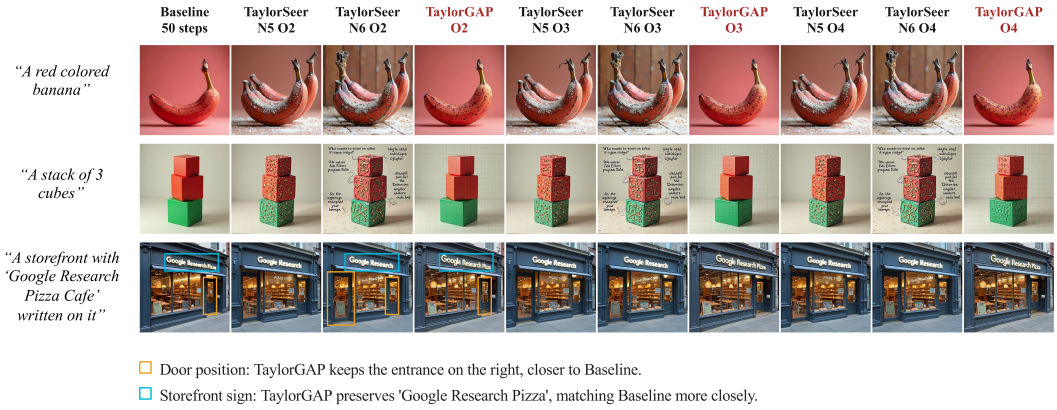

Figure 1: **Qualitative comparison of output fidelity** between TaylorSeer and TaylorGAP relative to the full 50-step FLUX.1-dev baseline.

3.4 Ablation Studies

The preceding comparisons fix the trajectory at 50 steps. To verify that the gains do not depend on this choice, we repeat the equal-NFE comparison on FLUX.1-dev at 40 and 60 denoising steps (Tables 3). The progressive schedule adapts automatically, requiring nine full evaluations at $T = 40$ and eleven at $T = 60$; we match these budgets with TaylorSeer at $\mathcal{N} = 6$ and $\mathcal{N} = 7$, respectively. In both settings, TaylorGAP improves all five metrics over the equal-NFE TaylorSeer at every order $\mathcal{O} \in \{2, 3, 4\}$. At 40 steps the gains are $+2.25$–2.50 dB PSNR ($+13.9\%$–15.4%), $+0.047$–0.061 SSIM, and -0.109–-0.128 LPIPS (-26.3%–-31.0%); at 60 steps they grow to $+3.97$–4.13 dB PSNR ($+26.3\%$–27.2%), $+0.099$–0.108 SSIM, and -0.178–-0.190 LPIPS (-39.7%–-42.9%), with higher ImageReward and CLIP score in every row. The advantage therefore persists across trajectory lengths and widens when the matched-NFE budget forces a coarser refresh interval ($\mathcal{N} = 7$ at 60 steps versus $\mathcal{N} = 6$ at 40 steps), exactly the regime in which the horizon-dependent truncation scale of Section 2.1 grows.

4 Conclusion

We have argued that cache scheduling and Taylor forecasting are not independent design choices in feature-cached diffusion inference: the anchor grid simultaneously fixes the extrapolation horizon of every prediction and the geometric scale of the recursively estimated higher-order factors. TaylorGAP makes this coupling explicit. Its progressive schedule keeps early horizons short under a fixed

Table 2: **Quantitative comparison of text-to-video generation** with HunyuanVideo on VBench.

Method	Acceleration				VBench↑ Score (%)	PSNR↑	SSIM↑	LPIPS↓	Win rate (Ours vs. row)
	Latency (s)↓	Speed↑	FLOPs (T)↓	Speed↑					
Baseline: 50 steps	75.08	1.00×	14035.28	1.00×	81.17	—	—	—	—
TaylorSeer ($\mathcal{N}=4, \mathcal{O}=2$)	27.60	2.72×	3649.17	3.85×	80.53	17.182	0.6630	0.3784	8/8
TaylorSeer ($\mathcal{N}=5, \mathcal{O}=2$)	24.34	3.08×	2807.06	5.00×	80.23	16.427	0.6304	0.4195	4/4
TaylorGAP ($\mathcal{O}=2$)	**23.99**	**3.13**×	**2807.06**	**5.00**×	**80.95**	**21.131**	**0.7268**	**0.2391**	—
TaylorSeer ($\mathcal{N}=4, \mathcal{O}=3$)	28.10	2.67×	3649.17	3.85×	80.57	17.189	0.6630	0.3782	8/8
TaylorSeer ($\mathcal{N}=5, \mathcal{O}=3$)	24.81	3.03×	2807.06	5.00×	80.29	16.436	0.6307	0.4190	4/4
TaylorGAP ($\mathcal{O}=3$)	**24.50**	**3.06**×	**2807.06**	**5.00**×	**81.01**	**20.738**	**0.7161**	**0.2472**	—
TaylorSeer ($\mathcal{N}=4, \mathcal{O}=4$)	28.45	2.64×	3649.17	3.85×	80.58	17.188	0.6631	0.3782	8/8
TaylorSeer ($\mathcal{N}=5, \mathcal{O}=4$)	25.15	2.99×	2807.06	5.00×	80.30	16.435	0.6307	0.4190	4/4
TaylorGAP ($\mathcal{O}=4$)	**25.09**	**2.99**×	**2807.06**	**5.00**×	**81.02**	**20.901**	**0.7202**	**0.2438**	—

• **Win rate:** Similar to Table 1.

Table 3: **Equal-NFE ablation of TaylorGAP on FLUX.1-dev** at 40 denoising steps (left; 9 NFE, TaylorSeer with $N=6$) and at 60 denoising steps (right; 11 NFE, TaylorSeer with $N=7$).

Method	ImageReward ↑	CLIP ↑	PSNR ↑	SSIM ↑	LPIPS ↓	Method	ImageReward ↑	CLIP ↑	PSNR ↑	SSIM ↑	LPIPS ↓
Baseline: 40 steps	1.0340	27.5227	—	—	—	**Baseline: 60 steps**	0.9913	27.6136	—	—	—
TaylorSeer ($N=6, \mathcal{O}=2$)	0.9585	27.6525	16.2679	0.6293	0.4123	TaylorSeer ($N=7, \mathcal{O}=2$)	0.9566	27.3444	15.1508	0.6075	0.4432
TaylorGAP ($\mathcal{O}=2$)	**1.0074**	**27.8054**	**18.7704**	**0.6904**	**0.2846**	**TaylorGAP** ($\mathcal{O}=2$)	**0.9672**	**27.7540**	**19.2764**	**0.7155**	**0.2533**
TaylorSeer ($N=6, \mathcal{O}=3$)	0.9536	27.6252	16.2066	0.6282	0.4153	TaylorSeer ($N=7, \mathcal{O}=3$)	0.9562	27.3027	15.0733	0.6057	0.4472
TaylorGAP ($\mathcal{O}=3$)	**1.0213**	**27.8938**	**18.4525**	**0.6766**	**0.3034**	**TaylorGAP** ($\mathcal{O}=3$)	**0.9851**	**27.7252**	**19.0610**	**0.7064**	**0.2648**
TaylorSeer ($N=6, \mathcal{O}=4$)	0.9563	27.6357	16.2155	0.6279	0.4161	TaylorSeer ($N=7, \mathcal{O}=4$)	0.9550	27.3303	15.0837	0.6056	0.4474
TaylorGAP ($\mathcal{O}=4$)	**1.0244**	**27.8429**	**18.4633**	**0.6751**	**0.3068**	**TaylorGAP** ($\mathcal{O}=4$)	**0.9795**	**27.7354**	**19.0566**	**0.7043**	**0.2698**

NFE budget, and its geometry-aware recursion restores the correct finite-difference scale through the closed-form coefficients $\lambda_{r,k} = 2k/(2k-r+1)$, applied only at offsets one and two so that long-horizon predictions remain stable. At matched NFE, the resulting predictor improves every reported quality metric over equal-NFE TaylorSeer at every tested order on both FLUX.1-dev and HunyuanVideo: on images it raises PSNR by 18.28%–19.21% and SSIM by 12.23%–13.77% while lowering LPIPS by 28.69%–32.35%, and on videos the corresponding gains are 26.17%–28.64%, 13.54%–15.29%, and 41.00%–43.00%. The equal-NFE comparison at 40 and 60 denoising steps shows that this margin persists, and widens, when the budget forces a coarser refresh interval. One limitation remains: although the accelerated outputs move markedly closer to the full-computation baseline in fidelity terms, the gain in ImageReward, which scores human preference rather than pixel-level agreement, is comparatively modest. Closing this gap between fidelity and preference quality is therefore an open problem, and our future work will study more effective numerical forecasting schemes, such as higher-order or stability-preserving prediction formats, that translate the reduced trajectory deviation into higher ImageReward scores and better overall perceptual quality.

REFERENCES

Black Forest Labs. FLUX. `https://github.com/black-forest-labs/flux`, 2024.

Jiazi Bu, Pengyang Ling, Yujie Zhou, Yibin Wang, Yuhang Zang, Dahua Lin, and Jiaqi Wang. DiCache: Let Diffusion Model Determine Its Own Cache. In *International Conference on Learning Representations*, volume 2026, pp. 73778–73803, 2026.

Peiliang Cai, Jiacheng Liu, Haowen Xu, Xinyu Wang, Chang Zou, and Linfeng Zhang. LESA: Learnable Stage-Aware Predictors for Diffusion Model Acceleration. *arXiv preprint arXiv:2602.20497*, 2026.

Guantao Chen, Shikang Zheng, Yuqi Lin, and Linfeng Zhang. Forecast the Principal, Stabilize the Residual: Subspace-Aware Feature Caching for Efficient Diffusion Transformers. *arXiv preprint arXiv:2601.07396*, 2026.

Pengtao Chen, Mingzhu Shen, Peng Ye, Jianjian Cao, Chongjun Tu, Christos-Savvas Bouganis, Yiren Zhao, and Tao Chen. Δ-DiT: A Training-Free Acceleration Method Tailored for Diffusion Transformers. *arXiv preprint arXiv:2406.01125*, 2024.

Benlei Cui, Shaoxuan He, Bukun Huang, Zhizeng Ye, Yunyun Sun, Longtao Huang, Hui Xue, Yang Yang, Jingqun Tang, Zhou Zhao, et al. TC-Padé: Trajectory-Consistent Padé Approximation for Diffusion Acceleration. *arXiv preprint arXiv:2603.02943*, 2026a.

Bowen Cui, Yuanbin Wang, Huajiang Xu, Biaolong Chen, Aixi Zhang, Hao Jiang, Zhengzheng Jin, Xu Liu, and Pipei Huang. Denoising as Path Planning: Training-Free Acceleration of Diffusion Models with DPCache. *arXiv preprint arXiv:2602.22654*, 2026b.

Jiaqi Han, Juntong Shi, Puheng Li, Haotian Ye, Qiushan Guo, and Stefano Ermon. Adaptive Spectral Feature Forecasting for Diffusion Sampling Acceleration. *arXiv preprint arXiv:2603.01623*, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From Reusing to Forecasting: Accelerating Diffusion Models With TaylorSeers. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15853–15863. IEEE, 2025.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating Diffusion Models for Free. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15762–15772. IEEE, 2024.

Zehong Ma, Longhui Wei, Feng Wang, Shiliang Zhang, and Qi Tian. MagCache: Fast Video Generation with Magnitude-Aware Cache. *Advances in Neural Information Processing Systems*, 38: 34348–34380, 2026.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. FORA: Fast-Forward Caching in Diffusion Transformer Acceleration. *arXiv preprint arXiv:2407.01425*, 2024.

Byunggwan Son, Jeimin Jeon, Jeongwoo Choi, and Bumsub Ham. Relational Feature Caching for Accelerating Diffusion Transformers. In *International Conference on Learning Representations*, volume 2026, pp. 37871–37891, 2026.

Xingwu Sun, Yanfeng Chen, Yiqing Huang, Ruobing Xie, Jiaqi Zhu, Kai Zhang, Shuaipeng Li, Zhen Yang, Jonny Han, Xiaobo Shu, et al. Hunyuan-Large: An Open-Source MoE Model with 52 Billion Activated Parameters by Tencent. *arXiv preprint arXiv:2411.02265*, 2024.

Shikang Zheng, Liang Feng, Xinyu Wang, Qinming Zhou, Peiliang Cai, Chang Zou, Jiacheng Liu, Yuqi Lin, Junjie Chen, Yue Ma, et al. Forecast then Calibrate: Feature Caching as ODE for Efficient Diffusion Transformers. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 13449–13457, 2026.

A Worked Example: Exactness on a Quadratic Trajectory

Consider the scalar trajectory $F(s) = s^2/2$ and the first progressive mixed-spacing triple $(s_{k-2}, s_{k-1}, s_k) = (0, 1, 3)$, for which $(q_{k-1}, q_k) = (1, 2)$. Then

$$\begin{aligned} D^{(1)}_{k-1} &= \frac{1}{2}, & D^{(1)}_k &= 2, \\ D^{(2)}_{k,\mathrm{TS}} &= \frac{2-\frac{1}{2}}{2} = \frac{3}{4}, & \widetilde{D}^{(2)}_k &= \frac{2-\frac{1}{2}}{\frac{2+1}{2}} = 1 = F^{(2)}(3). \end{aligned} \tag{25}$$

Using the same zeroth- and first-order factors in both predictors gives, for $x > 0$,

$$\begin{aligned} F(3+x) &= \frac{9}{2} + 3x + \frac{x^2}{2}, \\ \widehat{F}_{\mathrm{TS}}(3+x) &= \frac{9}{2} + 2x + \frac{3x^2}{8}, \\ \widehat{F}_{\mathrm{corr}}(3+x) &= \frac{9}{2} + 2x + \frac{x^2}{2}. \end{aligned} \tag{26}$$

Their absolute errors are $x + x^2/8$ and x, respectively. Thus, the geometric correction removes the additional quadratic error caused by the mis-scaled curvature factor, while leaving the common first-order secant error unchanged.