TORE: TOKEN-WISE TAYLOR ORDER SELECTION VIA DECOMPOSED RESIDUAL ERROR ESTIMATION FOR DIFFUSION ACCELERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models excel at image and video generation, but iterative denoising incurs substantial inference latency. Recent token-adaptive acceleration methods select Taylor prediction orders for the full-model residual using shallow-layer proxy scores. However, candidate orders are evaluated on these proxy features but used to predict the full-model residual, creating a proxy–target mismatch: the proxy-preferred order may not minimize residual prediction error. To address this mismatch, we propose **TORE** (**T**aylor **O**rder selection via **R**esidual **E**rror estimation), which selects token-wise Taylor orders based on estimated prediction errors for the full-model residual. Yet direct error evaluation requires future ground-truth residuals, which are unavailable during order selection. TORE therefore estimates prediction error by analyzing how errors arise in Taylor extrapolation. Specifically, it decomposes prediction error into truncation error and finite-difference derivative error, then estimates both contributions from cached residuals to preselect token-wise orders for upcoming skip steps. Extensive experiments show that TORE achieves $4.31\times$ speedup on FLUX.1-dev and $4.18\times$ on Wan2.1 while maintaining high generation quality and fidelity.



Figure 1: **Teaser of TORE.** Representative samples generated by TORE on FLUX.1-dev, achieving a **4.31**$\times$ latency speedup while preserving high generation quality across diverse text prompts.

1 INTRODUCTION

Diffusion models have become a major class of generative models and have been widely applied to image and video generation (Rombach et al., 2022; Ho et al., 2022). With the shift from U-Net to Transformer backbones, Diffusion Transformers (DiTs) achieve better generation quality but incur substantially higher computational costs (Peebles & Xie, 2023). The iterative denoising process of diffusion models, combined with the quadratic complexity of self-attention, makes DiT inference very slow and costly. Therefore, accelerating DiT inference is important for practical deployment.

Training-free acceleration methods have recently emerged as an effective way to reduce the inference cost of diffusion models without additional training (So et al., 2024; Ma et al., 2024b; Zheng et al., 2025). Existing caching approaches mainly exploit temporal redundancy across denoising steps,

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.

either by reusing cached features (Zou et al., 2025; Liu et al., 2025a; Kahatapitiya et al., 2025) or by forecasting future features from historical observations (Cui et al., 2026; Han et al., 2026), with TaylorSeer using Taylor expansion for such prediction (Liu et al., 2025b). Building on TaylorSeer, TAP introduces token-wise adaptive Taylor order selection (Zhu et al., 2026). It evaluates candidate orders on the modulated input to the first Transformer block, but applies the selected order to predict the full-model residual. This creates a proxy–target mismatch between order evaluation and residual prediction. This strategy implicitly assumes that candidate orders have consistent prediction-error rankings on shallow-layer features and on the full-model residual. However, this assumption does not always hold. A natural gap exists between the two quantities. Even strong feature correlation does not guarantee consistent prediction-error rankings. We verify this phenomenon in Sec. 3.2.

To address the proxy–target mismatch, we propose **TORE** (**T**aylor **O**rder Selection via **R**esidual **E**rror Estimation). As illustrated in Fig. 2, TORE bases order selection on the estimated error of the residual prediction target itself, rather than on a separate shallow proxy, thereby narrowing the proxy–target gap and better matching the oracle-optimal order. Yet measuring these errors directly requires ground-truth residuals at skipped steps, which are unavailable without full-model computation. Our key idea is to estimate prediction error by analyzing the approximations in the Taylor predictor, rather than by comparing predictions with ground truth. We estimate the prediction error from cached residuals and use it to select the lowest-error order for each token. The estimate captures two main error sources: *Truncation Error*, caused by omitting the remainder of the Taylor expansion, and *Finite-Difference Derivative Error*, introduced by approximating higher-order derivatives with finite differences. This decomposition also explains why Taylor order selection is important. For an ideal Taylor expansion, increasing the order reduces truncation error. In practice, however, higher-order derivatives are estimated using finite differences, and the resulting approximation error can accumulate as the order increases. Therefore, a higher order reduces truncation error but may increase finite-difference derivative error. The overall prediction error is determined by the balance between these two terms. The optimal order should thus minimize their combined error, and may vary across tokens. Besides, all error estimation and order selection are performed at full-computation steps, with the resulting token-wise orders reused for the subsequent skip steps without further estimation or selection.

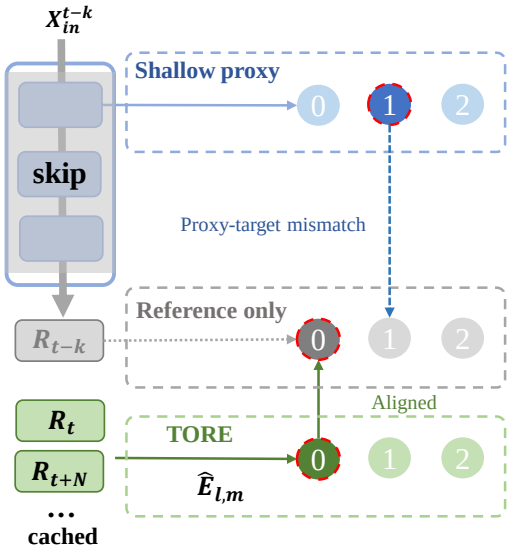

Figure 2: **Resolving the proxy–target mismatch.** Shallow-proxy scoring (blue) ranks Taylor orders by prediction errors on shallow features. TORE (green) instead ranks them by residual prediction errors estimated from cached residuals.

Our main contributions are summarized as follows:

TORE. We propose TORE, a training-free method for token-wise Taylor order selection in diffusion acceleration. TORE replaces shallow-proxy scores with estimates of each candidate order's prediction error on the full-model residual. This aligns order selection with the actual prediction target.

Error Decomposition. We decompose Taylor prediction error into *Truncation Error* and *Finite-Difference Derivative Error*, and estimate both contributions from cached residuals. This provides a computable selection criterion without requiring future true residuals. TORE uses this criterion at each full-computation step to preselect orders for the next $N - 1$ skip steps.

Evaluation. Experiments on text-to-image, text-to-video, and class-to-image generation demonstrate favorable quality–latency trade-offs against four representative prediction-based baselines. On FLUX.1-dev, TORE achieves a $4.31\times$ speedup with an CLIP score of 28.020 while preserving high

visual quality, as illustrated by the representative samples in Fig. 1. Further analyses show that our error estimates effectively guide token-wise Taylor order selection.

2 Related Work

Fast Sampling and Few-Step Generation. A major direction for diffusion acceleration is to reduce the number of denoising steps. Training-free samplers achieve this through improved numerical schemes (Song et al., 2021; Liu et al., 2022; Lu et al., 2022; Zhao et al., 2023; Zhang & Chen, 2023), whereas training-based approaches enable one- or few-step generation through distillation, consistency learning, or distribution matching (Salimans & Ho, 2022; Song et al., 2023; Luo et al., 2023; Yin et al., 2024). Rather than reducing the number of denoising updates, we retain the sampling schedule and replace selected full-network evaluations with residual forecasts whose Taylor orders are chosen per token according to estimated prediction error.

Reuse-Based Caching Acceleration. Another direction exploits temporal redundancy to avoid repeated computation across denoising steps. This motivates reusing encoder or block features in U-Nets (Li et al., 2024; Wimbauer et al., 2024) and cached representations in DiTs (Chen et al., 2024; Selvaraju et al., 2024). To make reuse more selective, layer-selection and attention-specific policies determine which representations to cache and when to refresh them (Ma et al., 2024a; Zhao et al., 2025). These policies refine where and when reuse occurs, but direct reuse remains a zero-order temporal approximation. We instead treat residual reuse as a candidate alongside higher-order forecasts, selecting the order with the lowest estimated residual prediction error for each token.

Forecasting-Based Caching Acceleration. Feature forecasting models temporal evolution rather than directly reusing cached representations. TaylorSeer uses Taylor expansion (Liu et al., 2025b), while alternative formulations include scaled Hermite polynomials in HiCache (Feng et al., 2026), rational approximation in TC-Padé (He et al., 2026), and learned linear predictors (Shen et al., 2026). Beyond predictor design, Confidence-Gated Taylor uses shallow-layer errors for reliability-based fallback (Guan et al., 2025), while TAP uses shallow proxies for token-wise order–horizon selection (Zhu et al., 2026). However, shallow-proxy errors need not preserve the ranking of candidate predictors at the full-model residual target. We instead construct computable token-wise estimates of residual prediction error by accounting for both Taylor truncation and derivative approximation errors. These estimates guide order selection from cached residuals, without requiring future ground-truth residuals or shallow-layer probes at skip steps.

3 Method

3.1 Preliminaries

Diffusion Model. Diffusion models generate samples through an iterative denoising process, where the backbone is repeatedly evaluated over a sequence of timesteps. At timestep t, we denote the backbone input as $\mathbf{x}_{\text{in}}(t)$ and its output as $\mathbf{x}_{\text{out}}(t) = \Phi_\theta(\mathbf{x}_{\text{in}}(t), t)$. We define the *full-model residual* as

$$\mathbf{R}(t) = \mathbf{x}_{\text{out}}(t) - \mathbf{x}_{\text{in}}(t), \tag{1}$$

such that $\mathbf{x}_{\text{out}}(t) = \mathbf{x}_{\text{in}}(t) + \mathbf{R}(t)$. Instead of repeatedly evaluating the entire backbone at every timestep, we exploit the temporal redundancy across neighboring denoising steps and forecast the evolution of $\mathbf{R}_t$ from historical full-computation steps.

Taylor-based forecasting. Taylor-based forecasting methods exploit the local approximation property of the Taylor expansion to extrapolate features at nearby timesteps. For computational efficiency, these methods typically truncate the Taylor series at a finite order and approximate the required high-order derivatives using finite differences. The resulting m-th-order predictor can be written as

$$\widehat{\mathbf{R}}(t-k) = \mathbf{R}(t) + \sum_{i=1}^{m} \frac{(-k)^i}{i!N^i} \Delta^i \mathbf{R}(t), \tag{2}$$

where $\mathbf{R}(t)$ denotes the residual at the reference timestep, m is the Taylor expansion order, k denotes the forecasting distance, and N is the interval used to construct the finite differences. $\Delta^i \mathbf{R}(t)$ denotes the i-th-order finite difference of the residual. However, this formulation uses the same

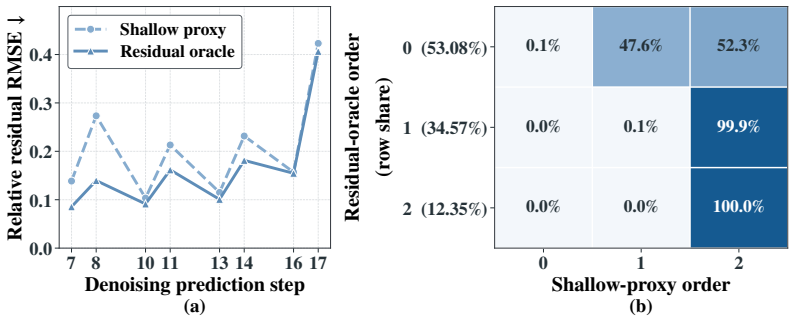

Figure 3: **Limitations of shallow-proxy order selection.** (a) Residual prediction error. (b) Shallow-proxy order distributions conditioned on the residual-oracle order; parentheses indicate each row's sample share.

Taylor order m for all tokens. In practice, tokens can evolve differently across denoising steps, and different prediction orders capture different patterns of temporal change. A single order may therefore not be optimal for every token. Selecting a suitable order for each token is thus important for improving Taylor-based acceleration.

3.2 Motivation Analysis

Shallow-proxy selection assumes that the order preferred on shallow features is also suitable for predicting the full-model residual. However, the two targets need not favor the same Taylor order. We examine this issue using an 18-step sampling schedule, $N = 3$, $M = 2$, and a resolution of 256×256. At each scheduled skip step, we obtain a reference residual through full-network computation and measure the error of the proxy-selected prediction. For comparison, a *residual oracle* uses this reference to select the lowest-MSE candidate order for each token. Figure 3(a) shows that shallow-proxy selection incurs higher normalized RMSE than the oracle at several steps. Since both use the same candidate predictions, this gap indicates that shallow-proxy scores do not always identify the best order for the target residual.

We further compare the orders selected by the shallow proxy and the residual oracle in Fig. 3(b). Rows correspond to oracle orders and columns to proxy orders. Each row shows the distribution of proxy selections among tokens with the corresponding oracle order. The percentages in parentheses indicate each row's share of all evaluated tokens. Diagonal entries therefore show the percentage of tokens in each row for which the proxy and oracle select the same order. The shallow proxy strongly favors higher orders. It matches the oracle on 100% of tokens with oracle order 2, but on only 0.1% of tokens in each of the oracle-order-0 and oracle-order-1 groups. These two groups together account for 87.65% of all evaluated tokens. This discrepancy shows that the order preferred by the shallow proxy is not necessarily optimal for predicting the full-model residual and it also motivates selecting orders based on estimated residual prediction error rather than shallow-proxy scores.

3.3 TORE: Taylor Order Selection via Residual Error Estimation

3.3.1 Error-Guided Residual Forecasting

We formulates Taylor order selection as residual-error minimization and construct a computable selection criterion by estimating the approximation errors in Taylor forecasting. As illustrated in Fig. 4, this criterion determines which order to use for each token and prediction horizon, with order evaluation and forecasting sharing the full-model residual as their target.

Let t denote the current full-computation step, N the interval between full steps, and $k \in \{1, \ldots, N-1\}$ a prediction horizon. For token l, let $\widehat{R}_{l,m}(t-k)$ denote the order-m forecast anchored at t, where $m \in \{0, \ldots, M\}$. Our objective is to minimize $\|R_l(t-k) - \widehat{R}_{l,m}(t-k)\|_2$. Since obtaining the target residual requires full-model computation, we instead evaluate candidate orders using the cached residual finite differences $\{\Delta_N^i R(t)\}_{i=0}^{M+1}$, with $\Delta_N^0 R(t) = R(t)$.

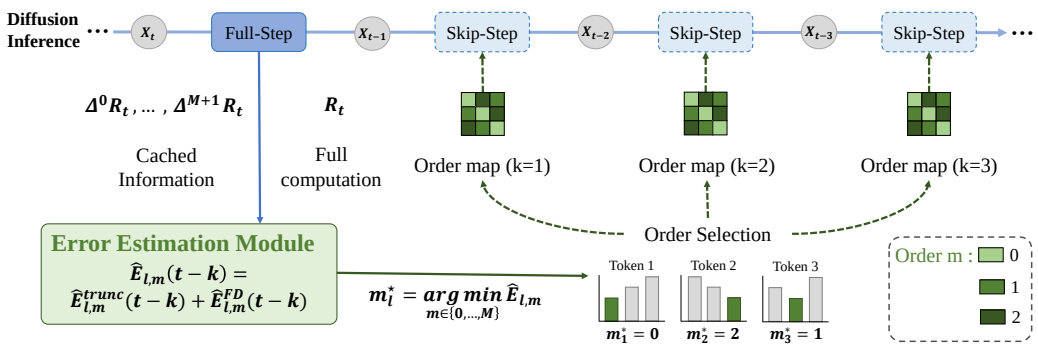

Figure 4: **Overview of TORE.** Full-steps preselect Taylor orders for each token at each of the next $N-1$ skip steps. Skip steps use the stored orders for residual forecasting, without further error estimation or order selection.

The *Error Estimation Module* assigns each candidate the score

$$\widehat{E}_{l,m}(t-k) = \widehat{E}^{\text{trunc}}_{l,m}(t-k) + \widehat{E}^{\text{FD}}_{l,m}(t-k). \tag{3}$$

The two terms estimate the contributions from omitted Taylor terms and finite-difference approximations of the retained derivatives, respectively. Their combination provides a criterion for comparing candidate orders without requiring an exact evaluation of the future residual error. Minimizing this score yields the token- and horizon-specific order map:

$$\mathcal{O}[k,l] = \widehat{m}_l(t-k) = \arg\min_{m\in\{0,\dots,M\}} \widehat{E}_{l,m}(t-k). \tag{4}$$

In Fig. 4, the bars compare candidate-order scores for individual tokens, while each grid records the selected orders for one prediction horizon. Order 0 corresponds to direct residual reuse, whereas higher orders incorporate additional temporal variation. TORE precomputes these maps at the full-computation step t. At each skip step $t-k$, TORE performs token-wise Taylor forecasting using the corresponding precomputed order map, then adds the predicted residual to the network input to obtain the output. No further error estimation or order selection is required. The next subsection derives the two error estimates from cached residual differences under a local smoothness assumption. Appendix A provides the complete inference procedure.

3.3.2 Decomposed Residual Error Estimation

We now derive the two components of equation 3. Assuming that the residual trajectory is locally smooth around t, the order-m Taylor expansion of the target residual is

$$R_l(t-k) = R_l(t) + \sum_{i=1}^{m} \frac{(-k)^i}{i!} R_l^{(i)}(t) + \mathcal{R}_{l,m+1}, \tag{5}$$

where $\mathcal{R}_{l,m+1}$ denotes the Taylor remainder. In practical forecasting, the derivatives are approximated using finite differences computed from cached residuals. Subtracting the predictor in equation 2 from the exact expansion gives

$$e_{l,m} = \mathcal{R}_{l,m+1} + \sum_{i=1}^{m} \frac{(-k)^i}{i!} \left(R_l^{(i)}(t) - \frac{\Delta_N^i R_l(t)}{N^i} \right), \tag{6}$$

equation 6 gives an exact decomposition of the prediction error. The first term is the truncation error introduced by omitting the Taylor remainder, while the second term is the error caused by replacing the exact derivatives with finite differences. Neither component can be evaluated exactly at a forecasted step because the remainder and the exact derivatives are unavailable. We therefore construct local estimates from the finite differences of previously computed residuals. These estimates are used to compare candidate orders rather than to recover the exact error magnitude.

Then, we define

$$d_{l,i}(t) = \left\| \Delta_N^i R_l(t) \right\|_2, \tag{7}$$

Table 1: Text-to-image generation results on DrawBench using FLUX.1-dev. The best and second-best quality results within each comparison group are marked in **bold** and underline, respectively.

Method	Efficiency				Generation Quality				
	Latency (s) ↓	Latency Speedup ↑	FLOPs(T) ↓	FLOPs Speedup ↑	Image Reward ↑	CLIP Score ↑	PSNR ↑	SSIM ↑	LPIPS ↓
Full	16.27	1.00×	3719.23	1.00×	1.0414	28.010	–	–	–
60% steps	9.77	1.66×	2232.00	1.66×	1.0069	27.440	19.060	0.7401	0.2477
50% steps	8.11	2.00×	1860.00	1.99×	1.0126	27.607	17.408	0.6947	0.3102
40% steps	6.48	2.51×	1488.00	2.49×	0.9418	27.505	16.363	0.6604	0.3599
TaylorSeer (N=2,O=1)	8.84	1.84×	1935.21	1.92×	**1.0494**	27.873	22.142	0.8043	0.1575
HiCache (N=2,O=2)	10.79	1.50×	1935.21	1.92×	0.9913	27.668	23.953	0.8109	0.1394
TC-Padé (N=3,θ=1.0)	7.16	2.27×	1562.54	2.38×	0.9968	27.608	23.035	**0.8384**	0.1546
TAP (N=2)	8.47	1.92×	1937.09	1.92×	1.0224	27.826	21.368	0.7928	0.1817
TORE (N=3,M=4)	6.46	2.51×	1487.79	2.49×	1.0462	**28.043**	**24.450**	0.8291	**0.1188**
TaylorSeer (N=3,O=2)	6.75	2.41×	1414.01	2.63×	1.0266	27.919	19.297	0.7345	0.2391
HiCache (N=3,O=2)	6.58	2.47×	1340.18	2.77×	1.0275	27.945	19.554	0.7384	0.2319
TC-Padé (N=6,θ=1.0)	5.41	3.00×	1116.63	3.33×	0.9808	27.509	22.034	0.8043	0.2067
TAP (N=3)	5.74	2.83×	1339.27	2.77×	0.9781	27.559	18.683	0.7281	0.2588
TORE (N=5,M=6)	5.17	3.14×	1190.53	3.12×	**1.0364**	**28.011**	**24.387**	**0.8210**	**0.1302**
TaylorSeer (N=5,O=2)	4.83	3.36×	968.50	3.84×	**1.0528**	27.819	16.657	0.6546	0.3600
HiCache (N=4,O=2)	5.25	3.09×	1042.68	3.56×	1.0252	27.879	18.023	0.6893	0.2942
TC-Padé (N=8,θ=1.4)	4.13	3.93×	893.47	4.16×	0.9469	27.430	20.644	**0.7679**	0.2728
TAP (N=5)	4.11	3.95×	967.28	3.84×	0.9431	27.585	15.583	0.6410	0.3998
TORE (N=8,M=5)	3.77	4.31×	892.94	4.16×	1.0308	**28.020**	**21.699**	0.7582	**0.2126**

where $\Delta_N^i R_l(t)$ denotes the i-th-order finite difference of the residual $R_l(t)$ with interval N, and $\|\cdot\|_2$ denotes the L_2 norm.

Proposition 1 (Truncation Error Estimate). Under the local smoothness assumption, we approximate the Taylor remainder using the first omitted term and estimate its derivative from the corresponding finite difference. This gives the local truncation error estimate

$$\widehat{E}_{l,m}^{\mathrm{trunc}}(t-k) = \frac{k^{m+1}}{(m+1)!N^{m+1}} d_{l,m+1}(t). \tag{8}$$

Proposition 2 (Finite Difference Approximation Error Estimate). The local expansions of the i-th and $(i+1)$-st finite differences give

$$R_l^{(i)}(t) - \frac{\Delta_N^i R_l(t)}{N^i} = -\frac{i}{2N^i}\Delta_N^{i+1} R_l(t) + \rho_{l,i}(t), \tag{9}$$

where $\rho_{l,i}(t)$ collects the remaining terms in the local expansion. Thus, $\Delta_N^{i+1} R_l(t)$ is not treated as an exact substitute for the unknown derivative. Instead, it provides the first observable correction to the i-th finite difference approximation. After applying the corresponding Taylor coefficients, we sum the magnitudes of the retained correction terms to obtain

$$\widehat{E}_{l,m}^{\mathrm{FD}}(t-k) = \frac{1}{2}\sum_{i=1}^{m} \frac{k^i}{(i-1)!N^i} d_{l,i+1}(t). \tag{10}$$

Taking the norm of each contribution before summation produces a nonnegative score and avoids cancellation between derivative approximation terms. The detailed derivation of both estimators is provided in the Appendix B.

4 Experiments

4.1 Experiment Settings

We conduct comprehensive experiments on three representative generative models across different generation tasks. For text-to-image generation, we evaluate FLUX.1-dev (Black Forest Labs, 2024)

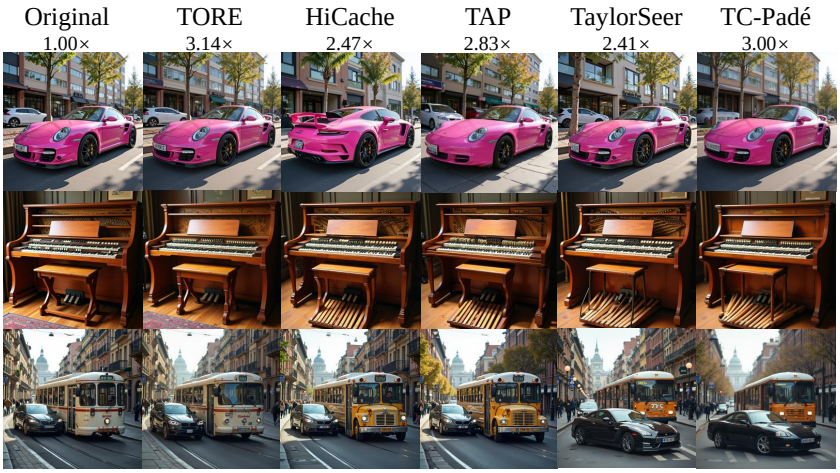

Figure 5: Qualitative comparison of different acceleration methods on the text-to-image generation task across diverse prompts. All methods use the same prompts and generation settings.

Table 2: Comparison on text-to-video generation for Wan2.1 on VBench. The best results among accelerated methods within each comparison group are shown in **bold**.

Method	Efficiency				VBench ↑		
	Latency (s) ↓	Latency Speedup ↑	FLOPs(T) ↓	FLOPs Speedup ↑	Overall Score	Quality Score	Semantic Score
Full	131.51	1.00×	28303.52	1.00×	80.72	84.41	65.98
TaylorSeer (N=4,O=1)	42.22	3.11×	7927.24	3.57×	77.24	82.18	57.48
HiCache (N=4,O=1)	42.22	3.11×	7927.24	3.57×	79.03	83.19	62.41
TC-Padé (N=6,θ=1.0)	41.09	3.20×	8492.35	3.33×	79.83	83.66	64.53
TAP (N=4)	37.68	3.49×	7928.15	3.57×	79.86	83.47	65.43
TORE (N=6,M=5)	36.96	3.56×	7925.99	3.57×	**80.47**	**84.13**	**65.82**
TaylorSeer (N=5,O=1)	37.82	3.47×	6795.20	4.16×	74.46	80.09	51.96
HiCache (N=5,O=1)	37.57	3.50×	6795.20	4.16×	77.86	82.51	59.27
TC-Padé (N=8,θ=1.4)	32.55	4.04×	6794.23	4.16×	75.29	79.69	57.71
TAP (N=5)	32.31	4.07×	6795.56	4.16×	79.71	83.14	**65.99**
TORE (N=7,M=4)	31.46	4.18×	6793.88	4.16×	**80.22**	**83.87**	65.64

using the 200 prompts from DrawBench. For text-to-video generation, we evaluate Wan2.1 (Wan et al., 2025) following the standard VBench evaluation protocol. For class-to-image generation, we evaluate DiT-XL/2 (Peebles & Xie, 2023) on ImageNet. These settings follow widely adopted evaluation protocols, enabling fair comparisons with existing acceleration methods.

4.2 Experiment Results

Text-to-Image Generation. We evaluate TORE on FLUX.1-dev against TaylorSeer, TAP, HiCache, and TC-Padé at multiple forecasting intervals. As shown in Table 1, TORE achieves favorable quality–latency trade-offs. In the second comparison group, TORE achieves a $3.14\times$ speedup and the best results on all five quality metrics, including an ImageReward score of 1.0364. Compared with TC-Padé at $3.00\times$ speedup, TORE improves PSNR by 2.35 dB and reduces LPIPS by 37.0%. At the more aggressive setting, TORE achieves a $4.31\times$ speedup. Compared with TC-Padé in this group, it reduces latency by 8.7% and LPIPS by 22.1% at a comparable FLOP count. Figure 5 compares outputs generated with the same prompts. In these examples, TORE better preserves the scene layout and visual details of the full-inference baseline despite its higher speedup. **Text-to-Video Generation.** We evaluate TORE on Wan2.1 using VBench. As shown in Table 2, TORE achieves the lowest latency and the highest overall and quality scores among accelerated methods in both comparison groups. At a $3.56\times$ speedup, TORE obtains an overall score of 80.47, only 0.25

Table 3: Comparison for class-to-image generation on ImageNet with DiT-XL/2. The best results among accelerated methods within each comparison group are shown in **bold**.

Method	Efficiency				Generation Quality		
	Latency (s) ↓	Latency Speedup ↑	FLOPs(T) ↓	FLOPs Speedup ↑	FID ↓	sFID ↓	Inception Score ↑
Full	0.42	1.00×	23.73	1.00×	2.41	4.39	236.31
60% steps	0.26	1.61×	14.23	1.66×	2.81	4.49	228.78
50% steps	0.22	1.90×	11.86	2.00×	3.15	4.77	227.45
TaylorSeer (N=3,O=2)	0.35	1.18×	8.59	2.76×	2.56	4.66	233.72
HiCache (N=2,O=1)	0.33	1.25×	12.42	1.91×	2.64	4.55	234.06
TC-Padé (N=2,θ=0.8)	0.29	1.42×	13.33	1.78×	3.53	5.27	221.81
TAP (N=2)	0.29	1.45×	13.33	1.78×	2.48	4.41	**234.26**
TORE (N=2,M=3)	0.26	1.61×	12.82	1.85×	**2.47**	**4.39**	232.68
TaylorSeer (N=5,O=2)	0.31	1.34×	5.27	4.50×	2.91	5.25	227.11
HiCache (N=4,O=2)	0.28	1.49×	6.70	3.54×	3.63	6.30	222.44
TC-Padé (N=3,θ=1.0)	0.25	1.66×	9.97	2.38×	5.40	6.77	199.91
TAP (N=4)	0.24	1.75×	8.09	2.93×	2.88	4.58	224.56
TORE (N=3,M=2)	0.22	1.90×	9.02	2.63×	**2.67**	**4.40**	**229.29**

points below the full-inference baseline. At a $4.18\times$ speedup, it achieves an overall score of 80.22, with a drop of 0.50 points from the baseline. Compared with TAP in this group, TORE improves the overall and quality scores by 0.51 and 0.73 points, respectively, while maintaining slightly lower latency. Figure 6 compares generated videos at similar speedups. In these examples, TORE better preserves the semantic content and visual appearance of the full-inference baseline.

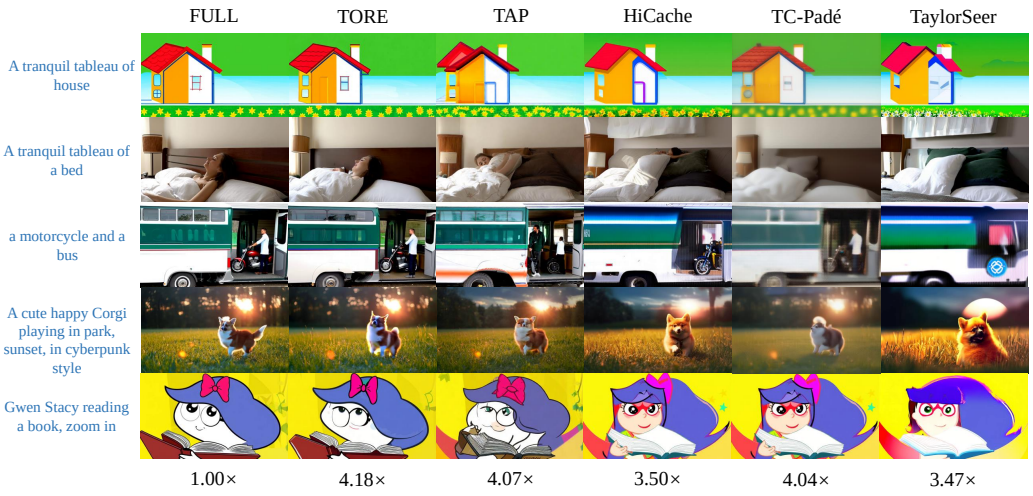

Figure 6: **Qualitative comparison on text-to-video generation.** Representative frames generated by different acceleration methods under comparable speedup ratios. TORE achieves a $4.18\times$ latency speedup while preserving high visual consistency with the full-inference baseline across diverse prompts.

Class-to-Image Generation. We evaluate TORE on ImageNet using DiT-XL/2. As shown in Table 3, TORE achieves the lowest FID and sFID and the highest latency speedup among forecasting methods in both comparison groups. At a $1.56\times$ speedup, TORE achieves an FID of 2.47, only 0.06 above the full-inference baseline, and matches its sFID of 4.39. At a $1.90\times$ speedup, TORE obtains an FID of 2.67 and an sFID of 4.40. Compared with using 50% of the original sampling steps, TORE reduces FID by 15.2% and sFID by 7.8% at the same reported latency of 0.22 seconds.

4.3 Ablation and Analysis

Effect of Adaptive Order Selection. With a full-step interval of $N = 5$ and maximum order $M = 2$, we independently forecast the next four steps from the same full-computation anchor to exclude recursive error propagation. Full-model residuals at each target step serve as ground truth. We compare our adaptive strategy with `fixed-0`, `fixed-1`, and `fixed-2`, which apply a uniform Taylor order to all tokens. Our strategy outperforms all fixed-order baselines across forecasting distances, reducing the error at $k = 4$ by 7% relative to the best fixed order (Fig. 7).

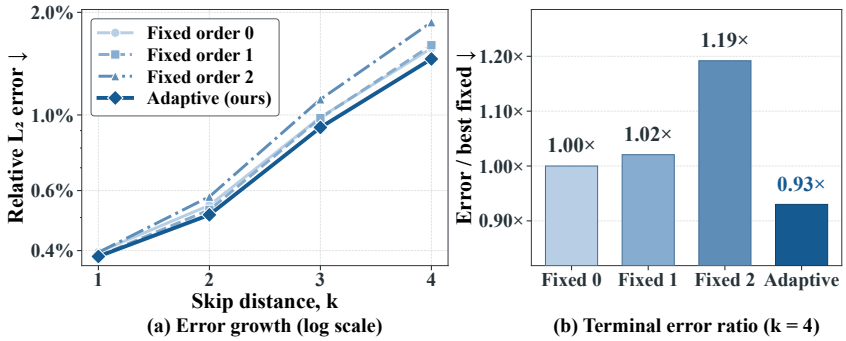

Figure 7: **Effectiveness of error-guided token-wise order selection. Left:** Relative L_2 residual prediction error versus skip distance k. **Right:** Normalized error at $k = 4$, with the best fixed-order baseline (`fixed-0`) set to 1.

Component Ablation. We conduct a 2×2 ablation to isolate the effects of full-model residual prediction and error-guided order selection. As shown in Table 4, full-model residual prediction consistently reduces latency with nearly unchanged FLOPs, while error-guided order selection substantially improves reconstruction fidelity in terms of PSNR, SSIM, and LPIPS. Combining both components achieves a favorable efficiency-quality trade-off, reaching a $3.32\times$ latency speedup with the best reconstruction fidelity.

Table 4: Ablation study of full-model residual prediction and token-adaptive order selection. ✓ and × denote whether each component is enabled. Best results are shown in **bold**.

Components		Efficiency				Generation Quality				
Residual Pred	Token Adaptive	Latency (s) ↓	Speed ↑	FLOPs(T) ↓	Speed ↑	Image Reward ↑	CLIP Score ↑	PSNR ↑	SSIM ↑	LPIPS ↓
×	×	5.81	2.80×	1042.88	3.56×	**1.0477**	27.810	17.646	0.6821	0.3087
✓	×	**4.56**	**3.57**×	**1041.72**	**3.57**×	1.0469	27.862	17.529	0.6838	0.3103
×	✓	5.83	2.79×	1117.22	3.32×	1.0405	27.951	20.018	0.7417	0.2282
✓	✓	4.90	3.32×	1116.12	3.33×	1.0413	**27.971**	**20.115**	**0.7435**	**0.2279**

5 Conclusion

We presented **TORE**, a training-free diffusion acceleration framework that selects token- and horizon-specific Taylor orders by estimating residual prediction errors from cached full-model residuals. The estimates account for truncation and finite-difference derivative errors without requiring future ground truth. Precomputed order maps enable skipped steps to forecast residuals while bypassing the entire denoiser. Experiments on text-to-image, text-to-video, and class-to-image generation demonstrate favorable quality–efficiency trade-offs and validate our error estimates and adaptive order selection.

AI USE STATEMENT

Generative AI tools were used solely for language editing and polishing, including improving grammar, wording, clarity, and readability of the manuscript. All AI-assisted edits were reviewed and revised by the authors. The authors take full responsibility for the final content of the paper.

REFERENCES

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

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.

Hanshuai Cui, Zhiqing Tang, Qianli Ma, Zhi Yao, and Weijia Jia. Predict to skip: Linear multistep feature forecasting for efficient diffusion transformers. *arXiv preprint arXiv:2602.18093*, 2026.

Liang Feng, Shikang Zheng, Jiacheng Liu, Yuqi Lin, Qinming Zhou, Peiliang Cai, Xinyu Wang, Junjie Chen, Chang Zou, Yue Ma, et al. Hicache: A plug-in scaled-hermite upgrade for taylor-style cache-then-forecast diffusion acceleration. In *International Conference on Learning Representations*, volume 2026, pp. 112361–112386, 2026.

Xiaoliu Guan, Lielin Jiang, Hanqi Chen, Xu Zhang, Jiaxing Yan, Guanzhong Wang, Yi Liu, Zetao Zhang, and Yu Wu. Forecasting when to forecast: Accelerating diffusion models with confidence-gated taylor. *Knowledge-Based Systems*, 330:114635, 2025. doi: 10.1016/j.knosys.2025.114635.

Jiaqi Han, Juntong Shi, Puheng Li, Haotian Ye, Qiushan Guo, and Stefano Ermon. Adaptive spectral feature forecasting for diffusion sampling acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Shaoxuan He, Benlei Cui, Bukun Huang, Zhizeng Ye, Yunyun Sun, Longtao Huang, Hui Xue, Yang Yang, Haiwen Hong, Jingqun Tang, and Zhou Zhao. TC-Padé: Trajectory-consistent pade approximation for diffusion acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 35768–35778, 2026.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 8633–8646, 2022.

Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S. Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Senmao Li, Taihang Hu, Joost van de Weijer, Fahad Shahbaz Khan, Tao Liu, Linxuan Li, Shiqi Yang, Yaxing Wang, Ming-Ming Cheng, and Jian Yang. Faster diffusion: Rethinking the role of the encoder for diffusion model inference. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with TaylorSeers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15853–15863, 2025b.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *International Conference on Learning Representations*, 2022.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems*, volume 35, pp. 5775–5787, 2022.

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

Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. In *Advances in Neural Information Processing Systems*, volume 37, 2024a.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15762–15772, 2024b.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023. doi: 10.1109/ICCV51070.2023.00387.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 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.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 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.

Zhirong Shen, Rui Huang, Jiacheng Liu, Chang Zou, Peiliang Cai, Shikang Zheng, Zhengyi Shi, Liang Feng, and Linfeng Zhang. Beyond fixed formulas: Data-driven linear predictor for efficient diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Junhyuk So, Jungwon Lee, and Eunhyeok Park. FRDiff: Feature reuse for universal training-free acceleration of diffusion models. In *European Conference on Computer Vision*, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 2023.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, Christian Rupprecht, Daniel Cremers, Peter Vajda, and Jialiang Wang. Cache me if you can: Accelerating diffusion models through block caching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6211–6220, 2024.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6613–6623, 2024.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *International Conference on Learning Representations*, 2023.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 49842–49869, 2023.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. In *International Conference on Learning Representations*, 2025.

Zhixin Zheng, Xinyu Wang, Chang Zou, Shaobo Wang, and Linfeng Zhang. Compute only 16 tokens in one timestep: Accelerating diffusion transformers with cluster-driven feature caching. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 10181–10189. Association for Computing Machinery, 2025.

Haowei Zhu, Tingxuan Huang, Xing Wang, Tianyu Zhao, Jiexi Wang, Weifeng Chen, Xurui Peng, Fangmin Chen, Junhai Yong, and Bin Wang. TAP: A token-adaptive predictor framework for training-free diffusion acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 36082–36091, 2026.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *International Conference on Learning Representations*, 2025.

A Algorithm

Algorithm 1 summarizes the inference procedure of TORE. At each full-computation step, TORE updates the residual cache and preselects a Taylor order for each token at each of the next $N-1$ skip steps. Skip steps then retrieve the selected orders and reconstruct the network output from the input and the predicted residual.

Algorithm 1: Token-wise Error-Guided Residual Forecasting

Require: sampling steps T, full-step interval N, maximum order M

Initialize residual cache $\mathcal{C}$ and order map $\mathcal{O}$
for $\tau = T, T-1, \ldots, 1$ **do**
  if τ *is a full-computation step* **then**
    $t \leftarrow \tau$
    $x_{\text{out}}(t) \leftarrow \Phi_\theta(x_{\text{in}}(t), t)$
    $R(t) \leftarrow x_{\text{out}}(t) - x_{\text{in}}(t)$
    Update $\mathcal{C}$ with $\{R_l(t)\}_l$ and the corresponding finite differences
    $\mathcal{O} \leftarrow \varnothing$
    for $k = 1$ **to** $N-1$ **do**
      foreach *token* l **do**
        Estimate $\{\widehat{E}_{l,m}(t-k)\}_{m=0}^{M}$ from $\mathcal{C}$ using Eqs. (8)–(10)
        $\widehat{m}_l(t-k) \leftarrow \arg\min_{m \in \{0,\ldots,M\}} \widehat{E}_{l,m}(t-k)$
        $\mathcal{O}[k,l] \leftarrow \widehat{m}_l(t-k)$
  else
    $k \leftarrow t - \tau$
    foreach *token* l **do**
      $\widehat{m}_l(t-k) \leftarrow \mathcal{O}[k,l]$
      $\widehat{R}_{l,\widehat{m}_l(t-k)}(t-k) \leftarrow \text{TaylorPredict}\big(\mathcal{C}_l, \widehat{m}_l(t-k), k\big)$
    $\widehat{x}_{\text{out}}(t-k) \leftarrow x_{\text{in}}(t-k) + \widehat{R}(t-k)$

B Derivation of the Token-wise Error Estimator

Let $R_l(t)$ denote the full-model residual of the l-th token at the current full-computation step t. Consecutive full-computation steps are separated by an interval N. Since denoising proceeds toward smaller timesteps, $t+N, t+2N, \ldots$ correspond to previously evaluated full steps whose residuals are available in the cache. We define the forward finite difference over the full-step interval N as

$$\Delta_N R_l(t) = R_l(t+N) - R_l(t), \tag{11}$$

and recursively define the higher-order differences as

$$\Delta_N^i R_l(t) = \Delta_N^{i-1} R_l(t+N) - \Delta_N^{i-1} R_l(t), \qquad i \geq 2. \tag{12}$$

For a skip step $t-k$, where $k \in \{1, \ldots, N-1\}$ denotes the prediction horizon from the current full step, the m-th order Taylor prediction is

$$\widehat{R}_{l,m}(t-k) = R_l(t) + \sum_{i=1}^{m} \frac{(-k)^i}{i!N^i} \Delta_N^i R_l(t). \tag{13}$$

Error decomposition. Expanding the target residual $R_l(t-k)$ around the current full step t gives

$$R_l(t-k) = R_l(t) + \sum_{i=1}^{m} \frac{(-k)^i}{i!} R_l^{(i)}(t) + \mathcal{T}_{l,m+1}(t,k), \tag{14}$$

where $\mathcal{T}_{l,m+1}(t,k)$ denotes the Taylor truncation remainder after order m. Defining the prediction error as

$$e_{l,m}(t-k) = R_l(t-k) - \widehat{R}_{l,m}(t-k), \tag{15}$$

we obtain

$$e_{l,m}(t-k) = \mathcal{T}_{l,m+1}(t,k) + \sum_{i=1}^{m} \frac{(-k)^i}{i!} \left(R_l^{(i)}(t) - \frac{\Delta_N^i R_l(t)}{N^i} \right). \tag{16}$$

The prediction error therefore arises from two sources: the Taylor truncation error and the error introduced by approximating the derivatives with finite differences.

Finite-difference error estimation. For the i-th order forward finite difference, a local expansion around t gives

$$\frac{\Delta_N^i R_l(t)}{N^i} = R_l^{(i)}(t) + \frac{iN}{2} R_l^{(i+1)}(t) + \mathcal{O}(N^2). \tag{17}$$

The adjacent finite difference similarly satisfies

$$\frac{\Delta_N^{i+1} R_l(t)}{N^{i+1}} = R_l^{(i+1)}(t) + \mathcal{O}(N). \tag{18}$$

Substituting the latter into equation 17 eliminates the unknown $(i+1)$-th derivative and yields

$$R_l^{(i)}(t) - \frac{\Delta_N^i R_l(t)}{N^i} = -\frac{i}{2N^i} \Delta_N^{i+1} R_l(t) + \mathcal{O}(N^2). \tag{19}$$

Thus, although the original finite-difference approximation error contains a first-order term in N, the residual error of this approximation after substituting the adjacent finite difference starts from second order.

For convenience, we define the magnitude of the i-th order cached finite difference as

$$d_{l,i}(t) = \left\| \Delta_N^i R_l(t) \right\|_2. \tag{20}$$

Using equation 19, the finite-difference derivative error associated with an m-th order predictor is estimated by

$$\widehat{E}_{l,m}^{\mathrm{FD}}(t-k) = \frac{1}{2} \sum_{i=1}^{m} \frac{k^i}{(i-1)!N^i} d_{l,i+1}(t). \tag{21}$$

Truncation error estimation. The Taylor truncation error is approximated by the first neglected term, whose $(m+1)$-th derivative is in turn approximated using the corresponding cached finite difference:

$$\widehat{E}_{l,m}^{\mathrm{trunc}}(t-k) = \frac{k^{m+1}}{(m+1)!N^{m+1}} d_{l,m+1}(t). \tag{22}$$

Combining the two error sources gives the computable residual prediction error surrogate

$$\begin{aligned} \widehat{E}_{l,m}(t-k) &= \widehat{E}_{l,m}^{\mathrm{trunc}}(t-k) + \widehat{E}_{l,m}^{\mathrm{FD}}(t-k) \\ &= \frac{k^{m+1}}{(m+1)!N^{m+1}} d_{l,m+1}(t) + \frac{1}{2} \sum_{i=1}^{m} \frac{k^i}{(i-1)!N^i} d_{l,i+1}(t). \end{aligned} \tag{23}$$

Token-wise adaptive order selection. Since different tokens exhibit different residual dynamics, TORE selects the prediction order independently for each token and each prediction horizon:

$$\widehat{m}_l(t-k) = \underset{m\in\{0,\dots,M\}}{\arg\min}\ \widehat{E}_{l,m}(t-k), \qquad k \in \{1,\dots,N-1\}. \tag{24}$$

For $m = 0$, the finite-difference derivative term vanishes and the estimator reduces to

$$\widehat{E}_{l,0}(t-k) = \frac{k}{N} d_{l,1}(t), \tag{25}$$

corresponding to direct reuse of the residual at the current full step, i.e., $\widehat{R}_{l,0}(t-k) = R_l(t)$.

For further interpretation, we define the estimated error reduction when increasing the prediction order from $m-1$ to m as

$$\widehat{G}_{l,m}(t-k) = \widehat{E}_{l,m-1}(t-k) - \widehat{E}_{l,m}(t-k), \qquad m \geq 1. \tag{26}$$

Substituting equation 23 gives

$$\widehat{G}_{l,m}(t-k) = \frac{k^m}{m!N^m}\left[d_{l,m}(t) - \left(\frac{k}{(m+1)N} + \frac{m}{2}\right) d_{l,m+1}(t)\right]. \tag{27}$$

Therefore, increasing the prediction order from $m-1$ to m is estimated to be beneficial when $\widehat{G}_{l,m}(t-k) > 0$, i.e., when the variation captured at the current order is sufficiently large relative to the remaining higher-order variation.

C Additional Experimental Results

Fig. 8 further examines the behavior of TORE as the acceleration level increases. We compare generations obtained at $2.51\times$, $3.14\times$, and $4.31\times$ latency speedups using the same set of prompts. As the forecasting becomes more aggressive, the generated samples largely preserve their semantic content and global visual structure, indicating that TORE remains stable across a broad range of acceleration levels.

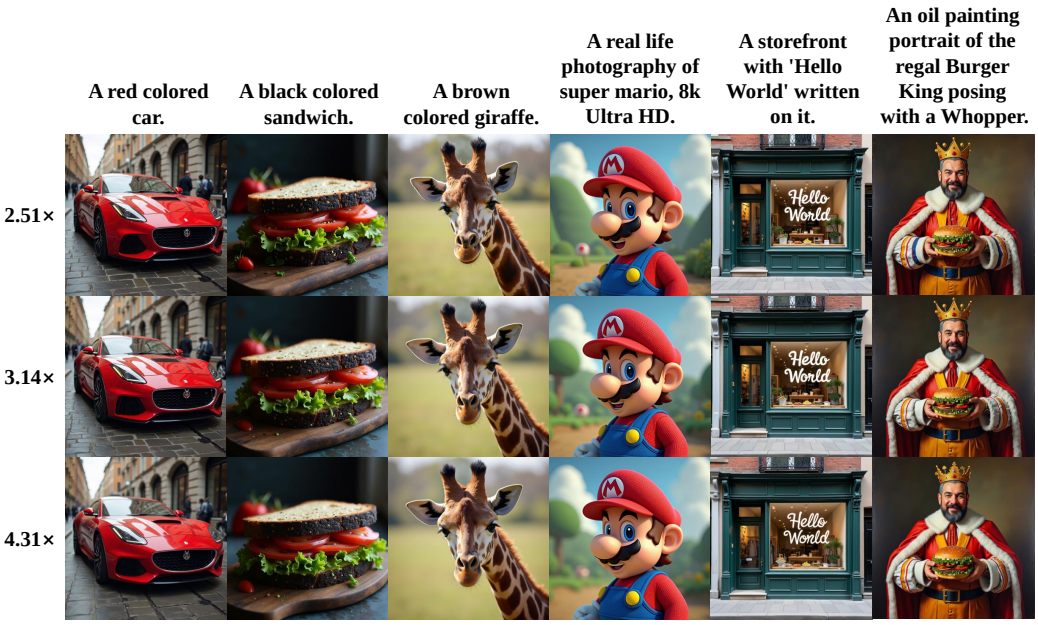

Figure 8: Qualitative results of TORE under increasing acceleration. We show text-to-image generation results at $2.51\times$, $3.14\times$, and $4.31\times$ latency speedups. TORE preserves semantic content and overall visual quality as the acceleration becomes more aggressive, demonstrating its robustness across different acceleration levels.