PRICE: PROPAGATION-RISK-INFORMED DYNAMIC DIFFUSION CACHING UNDER EXACT COMPUTE BUDGETS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion caching accelerates sampling by reusing features across denoising steps. Given a target compute budget, a caching policy must decide which steps should use full computation and which can reuse the cache. Existing adaptive caching methods often make this decision based on local signals, such as the difference between adjacent steps. However, the goal of caching is to preserve the final output of full computation, and a small local error does not always lead to a small final error. Our single-step caching experiments show that similar local errors can have very different final effects depending on where they occur in the denoising process. Motivated by this observation, we propose PRICE (**P**ropagation-**R**isk-**I**nformed **C**aching under **E**xact Budgets), a training-free method that estimates this final effect from the local discrepancy, solver step size, and downstream propagation. At each step, it compares the estimated risk with offline reference risks for the remaining steps and adjusts the threshold to the remaining reuse quota. On FLUX.1-dev, Qwen-Image, and Wan2.1, **PRICE** achieves $\mathbf{5.24}\times$, $\mathbf{6.87}\times$, and $\mathbf{3.49}\times$ acceleration, respectively, while better preserving full computation outputs than state-of-the-art caching baselines and maintaining competitive perceptual quality.

1 INTRODUCTION



Figure 1: Qwen-Image samples generated by PRICE with $3.78\times$ acceleration under a fixed full-computation budget across diverse prompts.

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

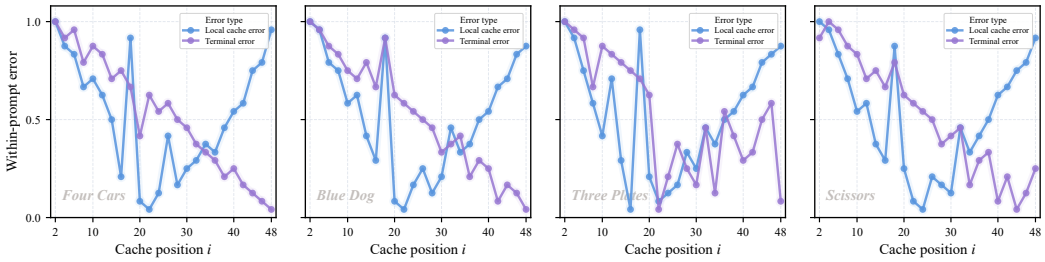

Figure 2: Local and final errors when caching at only one position i in a 50-step FLUX.1-dev trajectory. Each panel uses a different prompt; all other steps use full computation. Positions with small local errors do not necessarily produce small deviations in the final output.

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021b) and flow matching (Lipman et al., 2023; Liu et al., 2023) have driven progress in visual generation. Early image diffusion models commonly used U-Net backbones (Rombach et al., 2022; Ronneberger et al., 2015; Liu et al., 2024), while recent models increasingly adopt diffusion transformers (DiTs) (Peebles & Xie, 2023; Esser et al., 2024). Their ability to scale has supported image and video models demonstrating impressive fidelity and video temporal coherence (Labs, 2024; Labs et al., 2025; Wan et al., 2025; Wu et al., 2025; Zheng et al., 2024). Despite these advances, generating a sample still requires repeated forward passes through a large network. This cost grows with image resolution and video length, making inference speed a major obstacle to practical use.

Feature caching reduces this cost by reusing intermediate features that remain similar across nearby denoising steps. It avoids repeated computation without retraining the model or changing the sampling grid. DeepCache (Ma et al., 2024) demonstrated this approach in U-Net models, and FORA (Selvaraju et al., 2024) applies feature reuse to diffusion transformers. TaylorSeer (Liu et al., 2025b) goes beyond direct reuse by predicting features through Taylor expansions. Other methods adapt when the cache is refreshed: TeaCache (Liu et al., 2025a) estimates output changes from modulated input features, MagCache (Ma et al., 2025) models changes in residual magnitudes, and DiCache (Bu et al., 2026) monitors shallow-layer features online. These methods show that features need not be recomputed at every step, but deciding where reuse is appropriate remains central to preserving generation quality.

Adaptive caching policies typically compare a local discrepancy measure with a prescribed threshold, as in TeaCache (Liu et al., 2025a). However, similar local errors at different denoising steps can have different effects on the final output. An error introduced by reuse is scaled by the solver update and then propagates through subsequent denoising steps. Our single-step caching experiment in Figure 2 illustrates this mismatch: local and final errors can rank reuse positions differently, and small local errors can yield large final deviations. Under a fixed compute budget, a decision based only on local similarity may therefore reuse features at a step where full computation would better preserve the final output. The allocation decision needs an estimate of how the error will propagate through the rest of the trajectory. The decision also depends on how many reuse steps remain to be allocated. An estimated error may be acceptable when many steps must still reuse the cache, but avoidable when fewer reuse steps are needed and better opportunities are expected later. The threshold should therefore change with both the remaining denoising steps and the number of reuse decisions still required to meet the budget.

Therefore, we propose PRICE (**P**ropagation-**R**isk-**I**nformed **C**aching under **E**xact Budgets), a training-free method for this task. We first analyze how cache errors propagate through discrete sampling updates. This motivates a risk score that estimates the final effect of cache reuse. We compute the score using the current state's change since the last full evaluation, the solver step size, and local sensitivity estimated offline. It estimates the error rather than providing a guaranteed bound, and requires no extra full evaluations during sampling. At each step, PRICE compares the score with reference scores collected offline for the chosen budget. It uses only reference scores from the remaining steps and adjusts the threshold to the reuse quota still needed. The current sample determines the risk score, while the reference scores describe the opportunities expected over the rest of the trajectory. After each decision, the remaining quota and eligible positions are updated. Feasibility rules force full computation or reuse when the remaining budget requires it. The policy

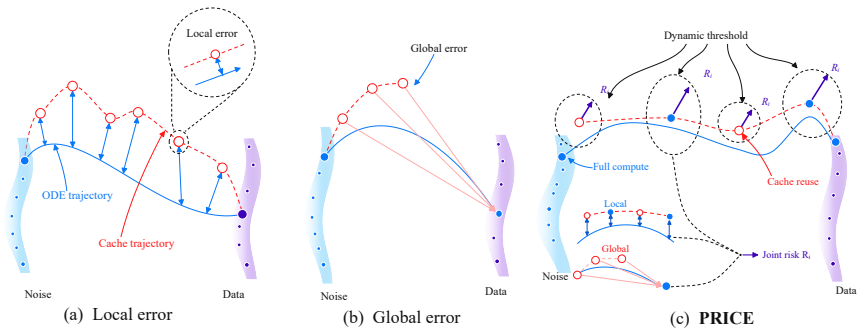

Figure 3: **Overview of PRICE. (a)** Local error measures the immediate effect of cache reuse. **(b)** The solver step size and later denoising steps determine its effect on the final output. **(c)** PRICE combines these factors into a risk score R_i. The threshold depends on reference risks for the remaining steps and the remaining budget, choosing full computation (blue) or cache reuse (red).

thus adapts to each sample while meeting its budget exactly. Figure 1 shows Qwen-Image samples generated with PRICE.

Empirically, PRICE achieves state-of-the-art reconstruction fidelity on FLUX.1-dev, Qwen-Image, and Wan2.1 under matched compute budgets, outperforming the evaluated baselines in LPIPS, SSIM, and PSNR across all reported budgets. On Qwen-Image, it delivers $6.87\times$ acceleration while reducing LPIPS by 27.2% and improving PSNR by 3.61 dB over the strongest compared baseline (Table 2). Our contributions are summarized as follows:

- We derive a propagation analysis on how cache errors affect the final output and derive a risk score combining local discrepancy, solver scaling, and downstream propagation.
- We propose a dynamic caching rule that allocates reuse according to terminal risk and remaining opportunities, with feasibility rules ensuring the requested NFE budget for every trajectory.
- Experiments on FLUX.1-dev (Labs, 2024), Qwen-Image (Wu et al., 2025), and Wan2.1 (Wan et al., 2025) show improved reconstruction fidelity over the compared caching methods under matched compute budgets.

2 Related Work

Diffusion Model Acceleration. Numerical solvers reduce the number of network evaluations through more accurate integration (Zhao et al., 2026b; Wang et al., 2025; Tong et al., 2025). Examples include DDIM (Song et al., 2021a), DPM-Solver (Lu et al., 2022; 2025), iPNDM (Zhang & Chen, 2023), and UniPC (Zhao et al., 2023). AMED-Solver learns trajectory corrections for few-step ODE sampling (Zhou et al., 2024). Other approaches train generators for fewer steps: consistency models (Song et al., 2023; Lu & Song, 2025; Frans et al., 2025) learn consistent outputs along probability-flow trajectories, Distribution Matching Distillation matches a student's output distribution to that of a diffusion teacher (Yin et al., 2024), and MeanFlow trains an average-velocity field for one-step generation without a pretrained teacher (Geng et al., 2025).

Caching in Diffusion Models. SeaCache and SenCache guide feature reuse using spectral evolution and feature sensitivity, respectively (Chung et al., 2026; Haghighi & Alahi, 2026). MeanCache constructs average-velocity approximations (Gao et al., 2026). ERTACache (Peng et al., 2026) decomposes cache-induced error into feature shift and step amplification, and combines offline reuse scheduling with timestep adjustment and closed-form residual error rectification. Trajectory-level methods explicitly organize cache placement: ECAD uses evolutionary search (Aggarwal et al., 2026), LeMiCa constructs a lexicographic minimax caching path (Gao et al., 2025), and BudCache addresses budget-constrained step-level selection (Lei et al., 2026). Unlike a fixed schedule, PRICE uses the current sample and the remaining budget to select reuse steps online. Its score accounts for error propagation, and each sample meets the requested NFE budget without online schedule search.

3 METHOD

3.1 PRELIMINARIES

Diffusion Models. Let $\mathbf{x}_0 \sim p_{\text{data}}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Diffusion models construct the state at noise level t as

$$\mathbf{x}_t = \alpha_t \mathbf{x}_0 + \sigma_t \boldsymbol{\epsilon}, \tag{1}$$

where the signal and noise schedules α_t, σ_t cover continuous and discrete diffusion (Song et al., 2021b; Ho et al., 2020; Kingma et al., 2021). We use optimal-transport flow matching (Lipman et al., 2023; Liu et al., 2023), with $\alpha_t = 1 - t$ and $\sigma_t = t$ for $t \in [0, 1]$, and sample via

$$\mathrm{d}\mathbf{x}_t = v_\theta(\mathbf{x}_t, t, \mathbf{c})\,\mathrm{d}t, \tag{2}$$

where $\mathbf{c}$ denotes the text prompt.

Diffusion Transformers. At step i, write $\mathbf{x}_i = \mathbf{x}_{t_i}$ and $\mathbf{v}_i = v_\theta(\mathbf{x}_i, t_i, \mathbf{c})$. A DiT (Peebles & Xie, 2023) embeds the latent as $\mathbf{y}_i^0 = \mathcal{P}_\theta(\mathbf{x}_i, t_i, \mathbf{c})$, then applies D blocks combining attention, MLPs, residual connections, and conditioning:

$$\mathbf{y}_i^\ell = \big(\mathcal{B}_\ell \circ \mathcal{B}_{\ell-1} \circ \cdots \circ \mathcal{B}_1\big)(\mathbf{y}_i^0, t_i, \mathbf{c}), \qquad \ell = 1, \ldots, D.$$

The composition acts on the features, with t_i and $\mathbf{c}$ held fixed across all blocks. The output head $\mathcal{O}_\theta$, including final normalization and projection, yields the velocity $\mathbf{v}_i = \mathcal{O}_\theta(\mathbf{y}_i^D, t_i, \mathbf{c})$.

Residual Caching in DiTs. Following feature reuse across timesteps (Liu et al., 2025a; Ma et al., 2024), PRICE stores the stack's hidden-state residual at each FULL step and reuses it at CACHE steps. With $q(i) < i$ denoting the most recent FULL step,

$$\mathbf{r}_i = \mathbf{y}_i^D - \mathbf{y}_i^0, \qquad \widetilde{\mathbf{y}}_i^D = \mathbf{y}_i^0 + \mathbf{r}_{q(i)}, \qquad \widetilde{\mathbf{v}}_i = \mathcal{O}_\theta(\widetilde{\mathbf{y}}_i^D, t_i, \mathbf{c}). \tag{3}$$

The input embedding and output head still run; a CACHE step skips only the D blocks.

3.2 PROBLEM SETUP

We consider sampling on a fixed T-step grid $\{t_i\}_{i=0}^T$. At each step, a caching policy π chooses between full computation ($z_i = 0$) and cache reuse ($z_i = 1$). The choice is made before the current full evaluation, using information already available, such as the current state, cached features, and remaining budget. To measure the effect of caching, we compare the cached trajectory $\widetilde{\mathbf{x}}_i$ with the full-computation trajectory $\mathbf{x}_i^\star$, using the same prompt, initial noise, and solver. Their deviation is $\Delta_i = \|\widetilde{\mathbf{x}}_i - \mathbf{x}_i^\star\|_2$, with $\Delta_0 = 0$.

The budget allows exactly $B \in \{3, \ldots, T\}$ full evaluations, leaving $M = T - B$ steps for cache reuse. We always use full computation at the first two steps and the last step, so reuse is allowed only at $i \in \mathcal{E} = \{2, \ldots, T-2\}$. We seek a policy that minimizes the deviation at the final step:

$$\begin{aligned} &\min_\pi \mathbb{E}[\Delta_T], \quad \text{s.t.} \quad z_i \in \{0, 1\}, \qquad \sum_{i=0}^{T-1} (1 - z_i) = B, \\ &\text{where } i = 0, \ldots, T-1, \qquad z_i = 0 \quad \text{for } i \notin \mathcal{E}. \end{aligned} \tag{4}$$

The expectation measures average deviation from the full-computation output. The budget constraint applies separately to each trajectory, so an allocation cannot compensate for exceeding the budget on one sample by using fewer full evaluations on another. We use the risk score below to guide online decisions for this objective.

3.3 PROPAGATION-RISK-INFORMED DYNAMIC CACHING

To address Eq. (4), PRICE combines local discrepancy, solver scaling, and downstream propagation into a risk score. A dynamic caching rule compares this risk with the remaining opportunities and quota (Figures 2 and 4).

Local Discrepancy Estimation. Residual reuse error depends on the displacement from the cached anchor and the denoiser's sensitivity along the trajectory. On a full-computation calibration set $\mathcal{C}$

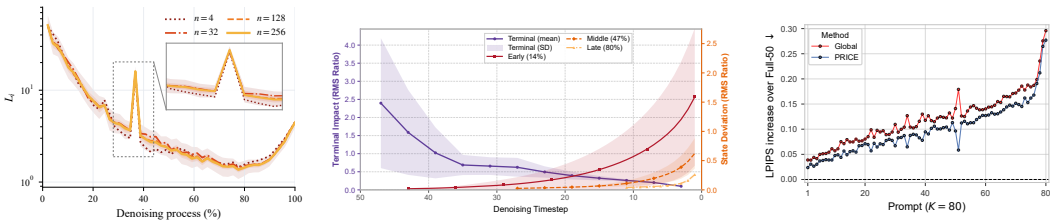

Figure 4: **Left:** Local sensitivity estimated from 4 to 256 calibration trajectories exhibits nearly identical temporal variation. **Middle:** Equal-RMS, cache-aligned interventions produce strongly position-dependent terminal effects and downstream deviations. **Right:** Across 80 randomly selected prompts, the suffix-adaptive threshold of PRICE generally reduces the LPIPS increase over Full-50 relative to a fixed global quantile constructed from the same risk bank.

with $K = |\mathcal{C}|$ trajectories, we record denoiser input–output pairs $(\mathbf{v}_i^{(k)}, \mathbf{x}_i^{(k)})$ for each $k \in \mathcal{C}$. We estimate local sensitivity at each timestep using the median adjacent secant:

$$L_i = \mathrm{Median}_{k\in\mathcal{C}} \left[\frac{\|\mathbf{v}_i^{(k)} - \mathbf{v}_{i-1}^{(k)}\|_2}{\max(\|\mathbf{x}_i^{(k)} - \mathbf{x}_{i-1}^{(k)}\|_2, \varepsilon)} \right]. \tag{5}$$

Here $\varepsilon > 0$ is a small numerical stabilizer. We compute local sensitivity once per model–scheduler pair and freeze it before evaluation. In Figure 4 (left), estimates from $n = 4, 32, 128$, and 256 FLUX trajectories follow the same temporal profile. This stability across calibration sizes suggests that a small offline set suffices to estimate L_i.

Local sensitivity is shared across trajectories, but the displacement from the cached anchor depends on the current sample and its previous cache decisions. Before the expensive forward pass, PRICE measures this displacement as $d_i = \|\mathbf{x}_i - \mathbf{x}_{q(i)}\|_2$. The anchor is the state at the most recent full evaluation, when the cached residual was refreshed. Combining this displacement with the precomputed local sensitivity L_i gives an estimate of the local discrepancy:

$$S_i = L_i d_i \approx \left\|\mathbf{v}_i - \mathbf{v}_{q(i)}\right\|_2. \tag{6}$$

From Cache Error to Terminal Risk. A local error alone does not determine terminal deviation: errors injected at different steps experience different solver scaling and future propagation. To isolate this effect, Figure 4 (middle) applies equal-RMS, cache-aligned interventions at different denoising positions. Their terminal impacts differ substantially, confirming that the same local perturbation can be amplified differently by the remaining trajectory.

For the paired trajectories in Section 3.2 (Euler updates; higher-order error terms in Appendix B.2; for multistep solvers such as the UniPC solver used for Wan2.1, we set h_i to the adjacent σ difference), we have $\mathbf{x}^\star_{i+1} = \mathbf{x}^\star_i + h_i v_\theta(\mathbf{x}^\star_i, t_i, \mathbf{c})$ and $\widetilde{\mathbf{x}}_{i+1} = \widetilde{\mathbf{x}}_i + h_i \widetilde{\mathbf{v}}_i$ (7). Assume the full denoiser is Λ_i-Lipschitz at step i, and let E_i bound the actual local output perturbation introduced by a cache action, $\|\widetilde{\mathbf{v}}_i - v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c})\|_2 \le E_i$. Adding and subtracting $v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c})$ yields the one-step recursion

$$\Delta_{i+1} \le (1 + |h_i|\Lambda_i)\,\Delta_i + z_i |h_i| E_i. \tag{8}$$

Theorem 3.1. *Under the conditions above and $\Delta_0 = 0$, the deviation between the cached and full-computation discrete trajectories satisfies*

$$\Delta_T \le \sum_{i=0}^{T-1} z_i \underbrace{E_i}_{\text{local error}} \underbrace{|h_i|}_{\text{solver scale}} \underbrace{\prod_{j=i+1}^{T-1} (1 + |h_j|\Lambda_j)}_{\text{downstream propagation}}.$$

Thus, the terminal contribution of a cache perturbation factorizes into its local magnitude, the solver coefficient at the injection step, and the sensitivity accumulated over subsequent steps. The proof is given in Appendix B.

The true E_i and future Λ_j are unavailable at decision time. We therefore replace them by the causal proxy S_i and the precomputed local sensitivity L_j, obtaining

$$w_i = \prod_{j=i+1}^{T-1} (1 + |h_j| L_j), \qquad R_i := \underbrace{S_i}_{\text{local}} \underbrace{|h_i|}_{\text{injection}} \underbrace{w_i}_{\text{propagation}}. \tag{9}$$

Here w_i is precomputed backward with $w_{T-1} = 1$, while R_i is evaluated online.

Allocation under an Exact Budget. The acceptable risk depends on the lower-risk opportunities expected later and the number of cache actions still required. For each target full-computation budget B, PRICE runs a deterministic uniform schedule on a small calibration set disjoint from evaluation. We record the risk at each eligible position before taking the prescribed action, then freeze the resulting budget-specific bank:

$$\mathbf{R}^{(B)} = [R_{k,j}^{(B)}]_{k \in \mathcal{C},\, j \in \mathcal{E}} \in \mathbb{R}^{K \times |\mathcal{E}|}. \tag{10}$$

Each budget has its own bank because it changes the cache anchors and the resulting risk distribution.

Before deciding at $i \in \mathcal{E}$, the controller maintains the remaining cache quota $m_i = M - \sum_{j<i,\, j \in \mathcal{E}} z_j$ and the remaining eligible suffix $\mathcal{E}_i = \{j \in \mathcal{E} : j \geq i\}$. Let $n_i = |\mathcal{E}_i|$ and $p_i = m_i / n_i$. PRICE pools calibration risks from this suffix and sets the threshold at the quantile corresponding to the fraction of steps that must still reuse the cache:

$$\widehat{F}_{i:}^{(B)}(a) = \frac{1}{K n_i} \sum_{k=1}^{K} \sum_{j \in \mathcal{E}_i} \mathbb{I}[R_{k,j}^{(B)} \leq a], \qquad \tau_i = \left(\widehat{F}_{i:}^{(B)}\right)^{-1}(p_i). \tag{11}$$

Here, $(\widehat{F}_{i:}^{(B)})^{-1}$ denotes the generalized inverse (quantile function) of the empirical CDF; equivalently, τ_i is the smallest risk level a such that $\widehat{F}_{i:}^{(B)}(a) \geq p_i$. The complete policy is

$$z_i = \begin{cases} 0, & i \notin \mathcal{E} \text{ or } m_i = 0, \\ 1, & i \in \mathcal{E} \text{ and } m_i = n_i, \\ \mathbb{I}[R_i \leq \tau_i], & 0 < m_i < n_i. \end{cases} \tag{12}$$

Figure 4 (right) compares this suffix-adaptive rule with a fixed global threshold $\tau = \widehat{F}^{-1}(m/n)$ constructed from the same risk bank, where m is the total cache quota and n is the number of eligible positions. Across 80 prompts, updating the quantile with the remaining quota and eligible suffix generally produces a smaller LPIPS increase over Full-50. Appendix A gives the complete offline and online procedures.

4 Experiments

4.1 Experimental Setup

Models and Baselines. We evaluate PRICE on two text-to-image models, FLUX.1-dev (Labs, 2024) (50 steps, 1024^2) and Qwen-Image (Wu et al., 2025) (50 steps, 1024^2), and on the Wan2.1-1.3B text-to-video model (Wan et al., 2025) (50-step UniPC (Zhao et al., 2023), 832×480, 65

Table 1: Reference-free image quality on **FLUX.1-dev** and **Qwen-Image**. Speedups are from Table 2. Blue rows denote PRICE (best: bold; second best: underlined).

FLUX.1-dev 1024×1024					
Method	**Speed** ↑	**Cycle Reward** ↑	**HPS v2.1** ↑	**CLIP-B** ↑	**CLIP-L** ↑
Full-50	1.00×	0.8682	0.3027	31.289	27.647
SeaCache ($\delta = 0.49$)	3.23×	**0.8744**	**0.2997**	31.186	<u>27.475</u>
SenCache ($\delta = 609$)	3.26×	0.8725	0.2961	**31.269**	27.426
PRICE ($B = 15$)	3.23×	<u>0.8732</u>	<u>0.2966</u>	<u>31.225</u>	**27.497**
SeaCache ($\delta = 0.29$)	2.25×	0.8576	0.3009	31.248	<u>27.522</u>
SenCache ($\delta = 256$)	2.25×	<u>0.8616</u>	**0.3012**	<u>31.251</u>	27.496
PRICE ($B = 21$)	2.26×	**0.8685**	<u>0.3010</u>	**31.277**	**27.563**

Qwen-Image 1024×1024					
Method	**Speed** ↑	**Cycle Reward** ↑	**HPS v2.1** ↑	**CLIP-B** ↑	**CLIP-L** ↑
Full-50	1.00×	1.4548	0.3064	32.837	29.187
TaylorSeer ($S = 12$)	6.88×	0.2060	0.1997	29.133	25.916
ERTACache ($B = 7$)	6.88×	<u>1.1147</u>	<u>0.2580</u>	<u>32.072</u>	<u>28.396</u>
PRICE ($B = 7$)	6.87×	**1.3723**	**0.2706**	**32.731**	**29.041**
TaylorSeer ($S = 5$)	3.78×	1.3502	<u>0.2814</u>	32.402	28.584
ERTACache ($B = 13$)	3.78×	<u>1.4214</u>	**0.3013**	<u>32.868</u>	<u>28.945</u>
PRICE ($B = 13$)	3.78×	**1.4252**	0.2804	**32.912**	**29.417**

Table 2: Quantitative comparison in text-to-image generation on **FLUX.1-dev** and **Qwen-Image** (best: bold; second best: underlined).

Method	Extra Search	Mean NFE	Acceleration			Reconstruction Quality		
			FLOPs(T) ↓	Latency(s) ↓	Speed ↑	LPIPS ↓	SSIM ↑	PSNR ↑
FLUX.1-dev 1024×1024								
Original: 50 steps	✗	50	2976.00	17.66	1.00×	–	–	–
Vanilla 9 steps	✗	9	535.68	3.27	5.40×	0.5026	0.6200	14.527
TeaCache ($\delta = 1.20$)	✗	9	536.23	3.36	5.26×	0.4656	0.6541	15.656
TaylorSeer ($S = 10$)	✗	9	536.81	4.88	3.62×	0.3846	0.6767	16.686
SeaCache ($\delta = 0.96$)	✗	9	536.22	3.37	5.24×	0.2998	<u>0.7582</u>	19.736
MagCache ($K = 5, \delta = 2.01$)	✗	9	536.22	3.47	5.09×	0.3428	0.7179	19.455
SenCache ($\delta = 2896$)	✗	9	536.23	3.37	5.24×	<u>0.2939</u>	0.7545	<u>20.916</u>
BudCache ($B = 9$)	✓	9	536.22	3.36	5.26×	0.3312	0.7371	19.086
ERTACache ($B = 9$)	✓	9	536.22	3.35	5.27×	0.4433	0.6564	16.030
PRICE ($B = 9$)	✗	9	536.23	3.37	5.24×	**0.2869**	**0.7619**	**21.429**
Vanilla 15 steps	✗	15	892.80	5.38	3.28×	0.4078	0.6784	15.825
TeaCache ($\delta = 0.60$)	✗	15	893.27	5.43	3.25×	0.3249	0.7341	17.798
TaylorSeer ($S = 5$)	✗	15	893.77	6.64	2.66×	0.1672	0.8460	23.298
SeaCache ($\delta = 0.49$)	✗	15	893.63	5.46	3.23×	<u>0.1614</u>	0.8502	23.440
MagCache ($K = 5, \delta = 0.16$)	✗	15	893.26	5.41	3.26×	0.1828	0.8348	22.491
SenCache ($\delta = 609$)	✗	15	889.10	5.41	3.26×	0.1691	<u>0.8503</u>	<u>25.035</u>
BudCache ($B = 15$)	✓	15	893.26	5.42	3.26×	0.1712	0.8457	24.613
ERTACache ($B = 15$)	✓	15	893.26	5.42	3.26×	0.4057	0.6755	16.102
PRICE ($B = 15$)	✗	15	893.27	5.47	3.23×	**0.1604**	**0.8526**	**25.089**
Vanilla 21 steps	✗	21	1249.92	7.58	2.33×	0.3298	0.7296	17.298
TeaCache ($\delta = 0.38$)	✗	21	1250.31	7.68	2.30×	0.2746	0.7676	18.827
TaylorSeer ($S = 3$)	✗	21	1250.74	8.79	2.01×	<u>0.0794</u>	0.9134	27.042
SeaCache ($\delta = 0.29$)	✗	21	1250.38	7.85	2.25×	0.0802	0.9157	28.003
MagCache ($K = 5, \delta = 0.05$)	✗	21	1250.30	7.83	2.26×	0.1047	0.8964	25.951
SenCache ($\delta = 256$)	✗	21	1250.60	7.86	2.25×	0.0827	<u>0.9168</u>	28.268
BudCache ($B = 21$)	✓	21	1250.30	7.87	2.24×	0.0828	0.9152	<u>28.497</u>
ERTACache ($B = 21$)	✓	21	1250.30	7.84	2.25×	0.3839	0.6941	16.149
PRICE ($B = 21$)	✗	21	1250.31	7.82	2.26×	**0.0788**	**0.9172**	**28.513**
Qwen-Image 1024×1024								
Original: 50 steps	✗	50	6833.21	30.04	1.00×	–	–	–
Vanilla 7 steps	✗	7	956.65	4.31	6.97×	0.5377	<u>0.5735</u>	13.104
TeaCache ($\delta = 1.02$)	✗	7	956.99	4.38	6.86×	0.5515	0.5583	13.060
TaylorSeer ($S = 12$)	✗	7	956.98	4.37	6.88×	0.7514	0.4623	9.869
ERTACache ($B = 7$)	✓	7	956.98	4.37	6.88×	<u>0.5324</u>	0.5642	<u>13.332</u>
PRICE ($B = 7$)	✗	7	956.98	4.37	6.87×	**0.3877**	**0.6491**	**16.941**
Vanilla 13 steps	✗	13	1776.63	7.89	3.81×	0.4125	0.6495	14.438
TeaCache ($\delta = 0.51$)	✗	13	1776.93	7.96	3.78×	<u>0.2867</u>	<u>0.7410</u>	<u>17.391</u>
TaylorSeer ($S = 5$)	✗	13	1776.92	7.94	3.78×	0.4462	0.6474	14.083
ERTACache ($B = 13$)	✓	13	1776.92	7.95	3.78×	0.3068	0.7084	16.499
PRICE ($B = 13$)	✗	13	1776.92	7.95	3.78×	**0.2360**	**0.7659**	**22.910**

frames). To evaluate fidelity at a given computation budget, we compare with Vanilla step reduction and seven representative caching methods: TeaCache (Liu et al., 2025a), TaylorSeer (Liu et al., 2025b), SeaCache (Chung et al., 2026), MagCache (Ma et al., 2025), SenCache (Haghighi & Alahi, 2026), BudCache (Lei et al., 2026), and ERTACache (Peng et al., 2026).

Evaluation protocol and metrics. We evaluate one image per DrawBench prompt (200 prompts) (Saharia et al., 2022) and videos on 100 carefully curated VBench prompts (Huang et al., 2024) (Appendix D.1). Reconstruction fidelity to paired 50-step outputs is measured by LPIPS (Zhang et al., 2018), SSIM (Wang et al., 2004), and PSNR. Reference-free image metrics include CycleReward (Bahng et al., 2025), HPS v2.1 (Wu et al., 2023), and CLIP-B and CLIP-L (Radford et al., 2021). For videos, we report seven reference-free VBench dimensions; standard evaluation of the other nine requires the official prompt suite. Efficiency is measured by the number of expensive transformer evaluations (NFE), FLOPs where applicable, end-to-end latency, and speedup over the 50-step model. For each baseline, we retain its official inference procedure and vary only its budget-controlling hyperparameter to match the target NFE as closely as possible; dynamic methods are compared at their realized mean NFE, whereas PRICE satisfies the requested budget for every sample. Full settings are provided in Appendix C.

4.2 Text-to-Image Generation

Table 2 shows that PRICE achieves state-of-the-art reconstruction fidelity on FLUX.1-dev and Qwen-Image under matched mean NFE, leading LPIPS, SSIM, and PSNR across all reported budgets. On FLUX, at $B = 9$, it delivers $5.24\times$ acceleration and improves over SenCache from 0.2939 to 0.2869 in LPIPS and from 20.916 to 21.429 dB in PSNR at the same reported latency of 3.37 s. These gains persist at $B = 15$ and $B = 21$, where PRICE also leads all three metrics. On Qwen-

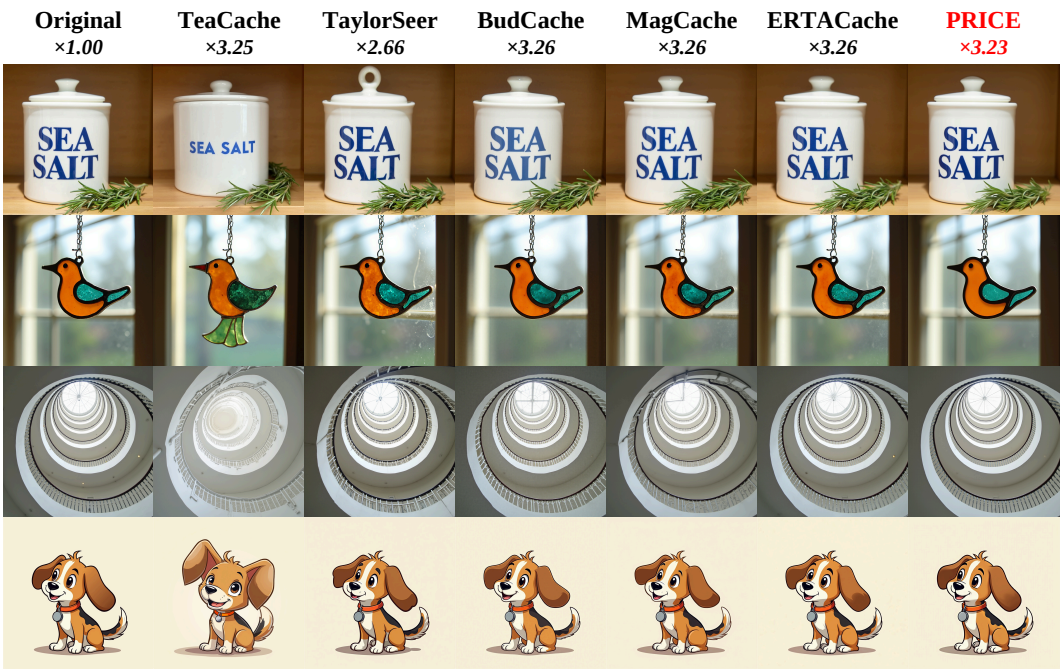

Figure 5: Visual comparison of 3.2× accelerated **FLUX.1-dev**.

Table 3: Ablations on **FLUX.1-dev** at $B = 9$: **(a)** components of $R_i = S_i|h_i|w_i$; **(b)** suffix-adaptive (*w/*) versus fixed global (*w/o*) thresholds at matched mean NFE.

Components	Choices	Variant	LPIPS ↓	SSIM ↑	PSNR ↑
(a) Risk score R_i	**PRICE**	$\boldsymbol{S_i\|h_i\|w_i}$	**0.2869**	**0.7619**	**21.429**
	w/o propagation	$S_i\|h_i\|$	0.3655	0.7248	18.277
	w/o local	$\|h_i\|w_i$	0.5282	0.6700	18.823
	w/o injection	$S_i w_i$	0.3508	0.7265	20.011
(b) Dynamic threshold	*w/*	$\boldsymbol{\tau_i = F_i^{-1}(m_i/n_i)}$	**0.2869**	**0.7619**	**21.429**
	w/o	$\tau = F^{-1}(m/n)$	0.3928	0.7144	19.359

Image, the advantage is particularly pronounced at $B = 7$: at 6.87× acceleration, PRICE reduces LPIPS by 27.2% (0.5324 to 0.3877) and improves PSNR by 3.61 dB over ERTACache. At $B = 13$, it further reaches 0.2360 LPIPS and 22.910 dB PSNR, compared with 0.2867 and 17.391 for TeaCache, the strongest reconstruction baseline at this budget.

Table 1 evaluates reference-free quality. On Qwen-Image at $B = 7$, PRICE leads all four reported metrics among the compared accelerated methods, increasing CycleReward from ERTACache's 1.1147 to 1.3723. It also leads both CLIP variants at $B = 13$. On FLUX at $B = 21$, it achieves the best CycleReward, CLIP-B, and CLIP-L, while HPS v2.1 remains competitive. Figure 5 compares outputs at approximately 3.2× acceleration. Relative to the displayed baselines, PRICE better preserves reference lettering, circular contours, and character appearance, consistent with its reconstruction scores.

4.3 Text-to-Video Generation

Table 4 extends the comparison to Wan2.1. At $B = 13$, PRICE achieves 3.49× acceleration with the best LPIPS (0.2180), SSIM (0.7338), and PSNR (21.119) among the compared methods. At $B = 21$, it again leads all three metrics, improving PSNR by 0.48 dB over ERTACache at a comparable latency of approximately 108 s. Its VBench score at this budget is 85.02%, within 0.28 percentage points of the best accelerated method. At $B = 13$, however, its VBench score is 81.98%, below SeaCache's 84.19%, indicating that stronger reconstruction does not uniformly translate into higher reference-free video quality. Figure 6 shows that PRICE preserves content and fine details closer to full computation across the displayed frames.

Table 4: Results on **Wan2.1**. Blue: PRICE; best: bold; second best: underlined.

Method	Extra Search	Mean NFE	Acceleration		Reconstruction Quality			VBench ↑
			Latency(s)↓	Speed ↑	LPIPS ↓	SSIM ↑	PSNR ↑	
Original: 50 steps	✗	50	244.41	1.00×	–	–	–	85.41%
Vanilla 13 steps	✗	13	68.59	3.56×	0.5271	0.4986	12.504	83.77%
SeaCache ($\delta = 0.53$)	✗	13	69.91	3.49×	0.2268	0.7233	19.078	**84.19%**
MagCache ($K = 5, \delta = 0.19$)	✗	13	69.82	3.50×	0.2769	0.6826	17.531	83.97%
BudCache ($B = 13$)	✓	13	69.79	3.50×	0.2273	0.7178	20.259	81.87%
TeaCache ($\delta = 0.25$)	✗	13	69.87	3.49×	0.2999	0.6658	16.980	83.41%
ERTACache ($B = 13$)	✓	13	69.89	3.49×	0.2215	0.7150	20.741	81.42%
PRICE ($B = 13$)	✗	13	69.86	3.49×	**0.2180**	**0.7338**	**21.119**	81.98%
Vanilla 21 steps	✗	21	106.56	2.29×	0.4604	0.5388	13.325	84.20%
SeaCache ($\delta = 0.25$)	✗	21	107.54	2.27×	0.0912	0.8632	24.419	**85.30%**
MagCache ($K = 5, \delta = 0.13$)	✗	21	107.62	2.27×	0.1079	0.8390	22.973	85.27%
BudCache ($B = 21$)	✓	21	107.60	2.27×	0.0967	0.8579	24.672	85.28%
TeaCache ($\delta = 0.13$)	✗	21	107.56	2.27×	0.1713	0.7677	20.001	84.85%
ERTACache ($B = 21$)	✓	21	107.66	2.27×	0.1039	0.8454	25.079	84.93%
PRICE ($B = 21$)	✗	21	107.54	2.27×	**0.0910**	**0.8653**	**25.559**	85.02%

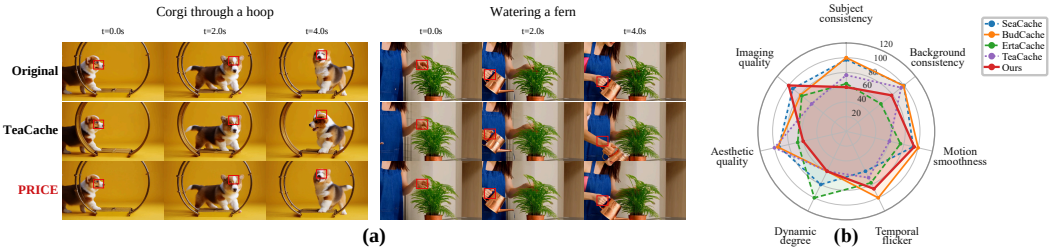

Figure 6: Qualitative and VBench comparison on **Wan2.1** at $B = 21$. The left panel compares representative video frames, and the right panel summarizes normalized VBench dimensions.

4.4 Ablation Study

Risk-score components. Table 3(a) isolates the factors in $R_i = S_i|h_i|w_i$ at $B = 9$. Removing local discrepancy S_i causes the largest LPIPS degradation, from 0.2869 to 0.5282, showing the importance of adapting decisions to the current sample. Removing downstream propagation w_i or injection scaling $|h_i|$ increases LPIPS to 0.3655 or 0.3508, respectively. The complete score leads all three metrics, supporting the joint use of sample-dependent discrepancy and trajectory-dependent scaling.

Dynamic threshold allocation. Table 3(b) replaces the suffix-adaptive threshold with a fixed global threshold tuned to the same mean NFE. At $B = 9$, adapting the threshold reduces LPIPS from 0.3928 to 0.2869 (27.0%), while increasing SSIM from 0.7144 to 0.7619 and PSNR from 19.359 to 21.429 dB. This improvement at matched computation supports conditioning allocation on the remaining trajectory: the suffix quantile updates the required reuse fraction and the reference opportunities still available. Gains persist at $B = 15$ and $B = 21$, with LPIPS reductions of 11.5% and 11.8%, respectively (Appendix D.2).

5 Conclusion

We presented PRICE, a training-free method for diffusion caching under exact compute budgets. Starting from the observation that similar local errors can have different final effects, we derived a propagation-based risk score combining local discrepancy, solver scaling, and downstream sensitivity. A suffix-quantile controller uses this score to allocate reuse according to the remaining opportunities and quota, while feasibility rules ensure the requested number of full evaluations. Experiments on FLUX.1-dev, Qwen-Image, and Wan2.1 demonstrate state-of-the-art reconstruction fidelity at matched budgets, with speedups of 5.24×, 6.87×, and 3.49×, respectively. The ablations support combining terminal-risk estimation with adaptive allocation. Future work could extend this perspective to layer-wise caching and adaptive timestep selection, and investigate whether calibration statistics can be shared across budgets or sampling grids.

AI USE STATEMENT

We used generative AI tools to improve the clarity and readability of the manuscript and to assist with searching for and identifying relevant literature. The authors reviewed the AI-assisted edits and checked the cited references against their original sources. The authors take full responsibility for the final content of this paper, including its claims, results, and references.

REPRODUCIBILITY STATEMENT

Section 3 and Appendix A (Algorithms 1 and 2) specify the offline calibration and online inference procedures of PRICE. The assumptions and complete proof of Theorem 3.1 are given in Appendix B.1, and further theoretical results are given in Appendix B.2. Section 4 and Appendix C describe the evaluation setup: the metrics (Appendix C.1), the model, budget, and calibration configurations (Appendix C.2), the official codebases and frozen hyperparameters of all baselines (Appendix C.3), and the hardware and time cost (Appendix C.4). The prompt-selection protocol for the video benchmark is described in Appendix D.1.

REFERENCES

Anirud Aggarwal, Abhinav Shrivastava, and Matthew Gwilliam. Evolutionary caching to accelerate your off-the-shelf diffusion model. In *International Conference on Learning Representations*, volume 2026, pp. 142488–142526, 2026.

Hyojin Bahng, Caroline Chan, Fredo Durand, and Phillip Isola. Cycle consistency as reward: Learning image-text alignment without human preferences. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22934–22946. IEEE, 2025.

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.

Jiwoo Chung, Sangeek Hyun, MinKyu Lee, Byeongju Han, Geonho Cha, Dongyoon Wee, Youngjun Hong, and Jae-Pil Heo. Seacache: Spectral-evolution-aware cache for accelerating diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14283–14294, 2026.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, volume 2025, pp. 34668–34684, 2025.

Huanlin Gao, Ping Chen, Fuyuan Shi, Chao Tan, Zhaoxiang Liu, Fang Zhao, Kai Wang, and Shiguo Lian. Lemica: Lexicographic minimax path caching for efficient diffusion-based video generation. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 132388–132415. Curran Associates, Inc., 2025. doi: 10.52202/085713-4408. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/bfa0d4d182de7e7939f67940fe524cce-Paper-Conference.pdf.

Huanlin Gao, Ping Chen, Fuyuan Shi, Ruijia Wu, Li YanTao, Qiang Hui, Youyuren, Ting Lu, Chao Tan, Shaoan Zhao, Zhaoxiang Liu, Fang Zhao, Kai Wang, and Shiguo Lian. Meancache: From instantaneous to average velocity for accelerating flow matching inference. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=GMCyL7Xs9R.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=uWj4s7rMnR.

Yasaman Haghighi and Alexandre Alahi. Sencache: Accelerating diffusion model inference via sensitivity-aware caching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14295–14304, June 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21807–21818. IEEE, 2024.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

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

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. Flux.1 kontext: Flow matching for in-context image generation and editing in latent space, 2025. URL https://arxiv.org/abs/2506.15742.

Mingkun Lei, Tong Zhao, Liangyu Yuan, and Chi Zhang. Budget-constrained step-level diffusion caching. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=kN0NMoVWXG.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7353–7363. IEEE, 2025a.

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, 2025b.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *International Conference on Learning Representations*, 2024.

Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *International Conference on Learning Representations*, volume 2025, pp. 50611–50649, 2025.

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. *Advances in neural information processing systems*, 35:5775–5787, 2022.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22 (4):730–751, 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. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=KZn7TDOL4J.

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.

Xurui Peng, Chenqian Yan, Hong Liu, Rui Ma, Fangmin Chen, XING WANG, Zhihua Wu, Songwei Liu, and Mingbao Lin. ERTACache: Error rectification and timesteps adjustment for efficient diffusion. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=InvyBiYcK5.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

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.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 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.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

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.

Vinh Tong, Trung-Dung Hoang, Anji Liu, Guy Van den Broeck, and Mathias Niepert. Learning to discretize denoising diffusion odes. In *International Conference on Learning Representations*, volume 2025, pp. 47244–47282, 2025.

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.

Ruoyu Wang, Beier Zhu, Junzhi Li, Liangyu Yuan, and Chi Zhang. Adaptive stochastic coefficients for accelerating diffusion sampling. In *Proc. NeurIPS*, 2025.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023.

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

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Loek7hfb46P.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 586–595. IEEE, 2018.

Tong Zhao, Mingkun Lei, Yucheng Han, and Chi Zhang. Bag: Budget-aware gating for diffusion caching. *arXiv preprint arXiv:2608.09231*, 2026a.

Tong Zhao, Mingkun Lei, Liangyu Yuan, Yanming Yang, Chenxi Song, Yang Wang, Beier Zhu, and Chi Zhang. Dyweight: Dynamic gradient weighting for few-step diffusion sampling. *arXiv preprint arXiv:2603.11607*, 2026b.

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

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

Zhenyu Zhou, Defang Chen, Can Wang, and Chun Chen. Fast ode-based sampling for diffusion models in around 5 steps. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7777–7786. IEEE, 2024.

Appendix

Table of Contents

A Complete Algorithms **15**

A.1 Offline Calibration 15

A.2 Online PRICE Inference 16

B Theoretical Analysis **17**

B.1 Proof of Theorem 3.1 17

B.2 Additional Theoretical Results 18

C Experimental Settings **20**

C.1 Metrics 20

C.2 Model Configuration 21

C.3 Baseline Settings 21

C.4 Hardware and Time Cost 21

D Additional Experimental Results **23**

D.1 More Qualitative Results on Wan2.1 23

D.2 Additional Ablation Results 23

D.3 More Experiments for Figure 2 25

E Additional Qualitative Results **26**

E.1 FLUX.1-dev 26

E.2 Wan2.1 29

E.3 Qwen-Image 31

A Complete Algorithms

A.1 Offline Calibration

Algorithm 1 prepares the references used for online allocation: a model–scheduler propagation table for estimating terminal risk, and a budget-specific risk bank for comparing remaining reuse opportunities. Each eligible score is recorded before the prescribed action, matching the information order at inference. The resulting artifacts are frozen, while the inference allocation remains sample-dependent. We write $\mathcal{B}_{1:D}(\cdot, t_i, \mathbf{c})$ for sequentially applying all D blocks from Section 3.1. Both calibration and inference retain the input embedding and output head at every step.

Algorithm 1 PRICE offline calibration

Require: Pretrained denoiser, solver grid $\{t_i\}_{i=0}^{T}$, calibration set $\mathcal{C}$, budgets $\mathcal{B}$, forced-full set $\mathcal{F}$
Ensure: Propagation table $\{|h_i|, L_i, w_i\}$ and banks $\{\mathbf{R}^{(B)}\}_{B\in\mathcal{B}}$

$\mathcal{E} \leftarrow \{0, \dots, T-1\} \setminus \mathcal{F}$
for each calibration prompt $k \in \mathcal{C}$ **do**
    Run an uncached trajectory and store $\{\mathbf{x}_i^{(k)}, \mathbf{v}_i^{(k)}\}_{i=0}^{T-1}$
    Store the adjacent secant at each timestep
end for
Compute local sensitivity L_i across $\mathcal{C}$ using Eq. (5)
Set $w_{T-1} \leftarrow 1$
for $i = T-2, \dots, 0$ **do**
    $w_i \leftarrow (1 + |h_{i+1}|L_{i+1})w_{i+1}$
end for
for each target full-computation budget $B \in \mathcal{B}$ **do**
    $M \leftarrow T - B$; construct a uniform schedule with exactly M cache actions in $\mathcal{E}$
    for each calibration prompt $k \in \mathcal{C}$ **do**
        Initialize the hidden-state residual $\mathbf{r}_q$ and full anchor $(q, \mathbf{x}_q)$ as empty
        for $i = 0, \dots, T-1$ **do**
            if a full anchor is available **then**
                Compute $d_i \leftarrow \|\mathbf{x}_i - \mathbf{x}_q\|_2$ and S_i using Eq. (6)
                Compute $R_i \leftarrow S_i|h_i|w_i$ using Eq. (9)
                if $i \in \mathcal{E}$ **then**
                    Store $R_{k,i}^{(B)} \leftarrow R_i$
                end if
            end if
            $\mathbf{y}_i^0 \leftarrow \mathcal{P}_\theta(\mathbf{x}_i, t_i, \mathbf{c})$
            if the uniform schedule prescribes FULL **then**
                $\mathbf{y}_i^D \leftarrow \mathcal{B}_{1:D}(\mathbf{y}_i^0, t_i, \mathbf{c})$; $\widetilde{\mathbf{y}}_i^D \leftarrow \mathbf{y}_i^D$
                $(q, \mathbf{x}_q, \mathbf{r}_q) \leftarrow (i, \mathbf{x}_i, \mathbf{y}_i^D - \mathbf{y}_i^0)$
            else
                $\widetilde{\mathbf{y}}_i^D \leftarrow \mathbf{y}_i^0 + \mathbf{r}_q$ ▷ Skip the DiT stack
            end if
            $\widetilde{\mathbf{v}}_i \leftarrow \mathcal{O}_\theta(\widetilde{\mathbf{y}}_i^D, t_i, \mathbf{c})$
            Advance the chosen solver using $\widetilde{\mathbf{v}}_i$
        end for
    end for
end for
return $\{|h_i|, L_i, w_i\}$ and $\{\mathbf{R}^{(B)}\}_{B\in\mathcal{B}}$

A.2 ONLINE PRICE INFERENCE

Algorithm 2 converts the current terminal-risk estimate and remaining quota into a full-computation or reuse action. Both scoring and selection precede the expensive DiT stack, so a reuse decision never requires the unavailable full output. Updating the anchors and remaining quota after each action closes the allocation loop. The hidden-state operations are shown for one conditioning branch.

Algorithm 2 PRICE exact-budget inference

Require: Pretrained denoiser, T, full budget B, forced-full set $\mathcal{F}$, propagation table $\{|h_i|, L_i, w_i\}$, frozen bank $\mathbf{R}^{(B)}$

Ensure: Generated sample using exactly B full evaluations

$\mathcal{E} \leftarrow \{0, \dots, T-1\} \setminus \mathcal{F}$; $M \leftarrow T - B$
$m \leftarrow M$; $n \leftarrow |\mathcal{E}|$; initialize $\mathbf{r}_q$ and $(q, \mathbf{x}_q)$ as empty
for $i = 0, \dots, T-1$ **do**
  Form $\mathbf{x}_i$ and compute $\mathbf{y}_i^0 \leftarrow \mathcal{P}_\theta(\mathbf{x}_i, t_i, \mathbf{c})$
  if a full anchor q exists **then**
    $d_i \leftarrow \|\mathbf{x}_i - \mathbf{x}_q\|_2$
    $S_i \leftarrow L_i d_i$; $R_i \leftarrow S_i |h_i| w_i$
  end if
  if $i \in \mathcal{F}$ **then**
    $z_i \leftarrow 0$
  else if $m = 0$ **then**
    $z_i \leftarrow 0$
  else if $m = n$ **then**
    $z_i \leftarrow 1$
  else
    $p_i \leftarrow m/n$
    $\mathcal{Y}_i^{(B)} \leftarrow \{R_{k,j}^{(B)} : k \in \mathcal{C},\ j \in \mathcal{E},\ j \geq i\}$
    $\tau_i \leftarrow \mathrm{Quantile}(\mathcal{Y}_i^{(B)}, p_i)$
    $z_i \leftarrow \mathbb{I}[R_i \leq \tau_i]$
  end if
  if $z_i = 0$ **then**
    $\mathbf{y}_i^D \leftarrow \mathcal{B}_{1:D}(\mathbf{y}_i^0, t_i, \mathbf{c})$; $\widetilde{\mathbf{y}}_i^D \leftarrow \mathbf{y}_i^D$
    $(q, \mathbf{x}_q, \mathbf{r}_q) \leftarrow (i, \mathbf{x}_i, \mathbf{y}_i^D - \mathbf{y}_i^0)$
  else
    $\widetilde{\mathbf{y}}_i^D \leftarrow \mathbf{y}_i^0 + \mathbf{r}_q$ ▷ Skip the DiT stack
  end if
  $\widetilde{\mathbf{v}}_i \leftarrow \mathcal{O}_\theta(\widetilde{\mathbf{y}}_i^D, t_i, \mathbf{c})$ ▷ Evaluate the output head
  Advance the chosen solver using $\widetilde{\mathbf{v}}_i$
  if $i \in \mathcal{E}$ **then**
    $m \leftarrow m - z_i$; $n \leftarrow n - 1$
  end if
end for
assert $\sum_i z_i = M$ and $\sum_i (1 - z_i) = B$

B Theoretical Analysis

B.1 Proof of Theorem 3.1

Assumptions. A1 (Local Lipschitz continuity of the full denoising field). For each timestep t_i and condition $\mathbf{c}$,

$$\|v_\theta(\mathbf{x}, t_i, \mathbf{c}) - v_\theta(\mathbf{y}, t_i, \mathbf{c})\|_2 \le \Lambda_i \|\mathbf{x} - \mathbf{y}\|_2, \qquad \Lambda_i = \sup_{\mathbf{x}} \|J_{v_\theta}(\mathbf{x}, t_i, \mathbf{c})\|_{\mathrm{op}}. \tag{13}$$

A2 (Bounded cache perturbation). At the cached state, the output perturbation induced by the selected action satisfies

$$\|\widetilde{\mathbf{v}}_i - v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c})\|_2 \le z_i E_i. \tag{14}$$

When $z_i = 0$, the full output is evaluated and the perturbation is zero; when $z_i = 1$, E_i bounds the velocity perturbation after the output head induced by hidden-state residual reuse.

Paired trajectories and deviation. Let $\mathbf{x}_i^\star$ denote the full-computation discrete trajectory and $\widetilde{\mathbf{x}}_i$ the cached trajectory. They use the same timestep grid and solver update,

$$\widetilde{\mathbf{x}}_{i+1} = \widetilde{\mathbf{x}}_i + h_i \widetilde{\mathbf{v}}_i, \qquad \mathbf{x}_{i+1}^\star = \mathbf{x}_i^\star + h_i v_\theta(\mathbf{x}_i^\star, t_i, \mathbf{c}). \tag{15}$$

Define the per-step deviation

$$\Delta_i := \|\widetilde{\mathbf{x}}_i - \mathbf{x}_i^\star\|_2, \qquad \Delta_0 = 0. \tag{16}$$

Lemma 1 (One-step cache-error recursion)**.** *Under A1 and A2,*

$$\Delta_{i+1} \le (1 + |h_i|\Lambda_i)\, \Delta_i + z_i |h_i| E_i. \tag{17}$$

Proof. Subtracting the two update rules gives

$$\widetilde{\mathbf{x}}_{i+1} - \mathbf{x}_{i+1}^\star = (\widetilde{\mathbf{x}}_i - \mathbf{x}_i^\star) + h_i \left[\widetilde{\mathbf{v}}_i - v_\theta(\mathbf{x}_i^\star, t_i, \mathbf{c})\right]. \tag{18}$$

Insert and subtract $v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c})$ inside the bracket:

$$\begin{aligned} \widetilde{\mathbf{x}}_{i+1} - \mathbf{x}_{i+1}^\star = \widetilde{\mathbf{x}}_i - \mathbf{x}_i^\star \\ + h_i \left[\widetilde{\mathbf{v}}_i - v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c})\right] \\ + h_i \left[v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c}) - v_\theta(\mathbf{x}_i^\star, t_i, \mathbf{c})\right]. \end{aligned} \tag{19}$$

Taking norms and applying the triangle inequality,

$$\begin{aligned} \Delta_{i+1} \le \Delta_i + |h_i| \, \|\widetilde{\mathbf{v}}_i - v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c})\|_2 \\ + |h_i| \, \|v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c}) - v_\theta(\mathbf{x}_i^\star, t_i, \mathbf{c})\|_2. \end{aligned} \tag{20}$$

By A2, the second term is at most $z_i |h_i| E_i$; by A1, the third term is at most $|h_i| \Lambda_i \Delta_i$. Therefore,

$$\begin{aligned} \Delta_{i+1} &\le \Delta_i + |h_i|\Lambda_i \Delta_i + z_i |h_i| E_i \\ &= (1 + |h_i|\Lambda_i)\, \Delta_i + z_i |h_i| E_i, \end{aligned} \tag{21}$$

which proves Lemma 1. □

Upper Bound. We now prove Theorem 3.1 by induction. Specifically, for every $m \in \{0, 1, \ldots, T\}$, we show

$$\Delta_m \le \sum_{i=0}^{m-1} z_i |h_i| E_i \prod_{j=i+1}^{m-1} (1 + |h_j|\Lambda_j). \tag{22}$$

The statement of Theorem 3.1 is the case $m = T$.

Base case ($m = 0$). The right-hand side is an empty sum and therefore equals zero, matching the initial condition $\Delta_0 = 0$.

Inductive step. Assume Eq. (22) holds for some $m \in \{0, \ldots, T-1\}$. Applying Lemma 1 at index $i = m$ gives

$$\Delta_{m+1} \le (1 + |h_m|\Lambda_m)\, \Delta_m + z_m |h_m| E_m. \tag{23}$$

Substituting the inductive hypothesis for Δ_m yields

$$\Delta_{m+1} \leq (1 + |h_m|\Lambda_m) \sum_{i=0}^{m-1} z_i |h_i| E_i \prod_{j=i+1}^{m-1} (1 + |h_j|\Lambda_j) + z_m |h_m| E_m. \tag{24}$$

Absorbing the factor $(1 + |h_m|\Lambda_m)$ into each product extends its upper index from $m-1$ to m:

$$\Delta_{m+1} \leq \sum_{i=0}^{m-1} z_i |h_i| E_i \prod_{j=i+1}^{m} (1 + |h_j|\Lambda_j) + z_m |h_m| E_m. \tag{25}$$

The trailing term $z_m|h_m|E_m$ corresponds to $i = m$, for which the product from $j = m+1$ to m is empty and equals one. Hence, the two terms merge into

$$\Delta_{m+1} \leq \sum_{i=0}^{m} z_i |h_i| E_i \prod_{j=i+1}^{m} (1 + |h_j|\Lambda_j), \tag{26}$$

which is exactly Eq. (22) with m replaced by $m+1$. The induction is complete. Setting $m = T$ yields

$$\Delta_T \leq \sum_{i=0}^{T-1} z_i |h_i| E_i \prod_{j=i+1}^{T-1} (1 + |h_j|\Lambda_j), \tag{27}$$

which proves Theorem 3.1.

B.2 Additional Theoretical Results

We extend Theorem 3.1 to bound error relative to a continuous reference trajectory. In addition to propagating cache perturbations across steps, the analysis accounts for velocity-field approximation and numerical discretization errors. Removing these two error sources recovers the recursion in Theorem 3.1.

Let $\mathbf{x}(t)$ solve the reference ODE

$$\frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} = v^\star(\mathbf{x}(t), t, \mathbf{c}), \qquad h_i = t_{i+1} - t_i, \tag{28}$$

and let $\widetilde{\mathbf{x}}_i$ follow the cached update in Eq. (15). In addition to A1 and A2, we make the following assumptions.

Additional assumptions. **A3 (Bounded velocity-field approximation error).** At each timestep t_i, the learned denoising field satisfies

$$\|v_\theta(\mathbf{x}, t_i, \mathbf{c}) - v^\star(\mathbf{x}, t_i, \mathbf{c})\|_2 \leq \eta_i \tag{29}$$

for all states encountered by the reference and cached trajectories. **A4 (Bounded higher-order total derivatives along the reference trajectory).** For an arbitrary expansion order $n \geq 2$ and the interval I_i with endpoints t_i and t_{i+1}, suppose

$$\sup_{t \in I_i} \left\| \frac{\mathrm{d}^k}{\mathrm{d}t^k} v^\star(\mathbf{x}(t), t, \mathbf{c}) \right\|_2 \leq M_i^{(k)}, \qquad k = 1, \ldots, n-1. \tag{30}$$

Define the deviation from the continuous reference trajectory as

$$\overline{\Delta}_i := \|\widetilde{\mathbf{x}}_i - \mathbf{x}(t_i)\|_2, \qquad \overline{\Delta}_0 = 0. \tag{31}$$

For compactness, define the order-n local discretization bound

$$\mathcal{D}_i^{(n)} := \sum_{k=2}^{n} \frac{|h_i|^k}{k!} M_i^{(k-1)}. \tag{32}$$

Theorem B.1 (Generalized terminal-error bound)**.** *Under A1–A4, for any Taylor expansion order $n \geq 2$, the terminal deviation of a cached trajectory with an arbitrary set of cache actions satisfies*

$$\overline{\Delta}_T \leq \sum_{i=0}^{T-1} \left(z_i |h_i| E_i + |h_i| \eta_i + \mathcal{D}_i^{(n)} \right) \prod_{j=i+1}^{T-1} (1 + |h_j| \Lambda_j) \,. \tag{33}$$

Thus, cache approximation, velocity-field approximation, and all retained orders of the local discretization error are propagated by the same downstream sensitivity factors. Setting $\eta_i = \mathcal{D}_i^{(n)} = 0$ reduces the recursion and its terminal bound to the cache-only result in Theorem 3.1.

Proof. Expanding the reference solution over I_i to arbitrary order $n \geq 2$ gives

$$\begin{aligned} \mathbf{x}(t_{i+1}) &= \mathbf{x}(t_i) + h_i v^\star(\mathbf{x}(t_i), t_i, \mathbf{c}) + \boldsymbol{\rho}_i, \\ \boldsymbol{\rho}_i &= \sum_{k=2}^{n-1} \frac{h_i^k}{k!} \frac{\mathrm{d}^{k-1}}{\mathrm{d}t^{k-1}} v^\star(\mathbf{x}(t), t, \mathbf{c}) \Big|_{t=t_i} + \boldsymbol{\rho}_i^{(n)}, \end{aligned} \tag{34}$$

where $\boldsymbol{\rho}_i$ denotes the local discretization remainder, distinct from the cached hidden-state residual $\mathbf{r}_i$, and the sum is empty for $n = 2$. Taylor's theorem and A4 imply

$$\left\| \boldsymbol{\rho}_i^{(n)} \right\|_2 \leq \frac{|h_i|^n}{n!} M_i^{(n-1)}, \qquad \|\boldsymbol{\rho}_i\|_2 \leq \mathcal{D}_i^{(n)}. \tag{35}$$

Subtracting this expression from the cached update yields

$$\begin{aligned} \widetilde{\mathbf{x}}_{i+1} - \mathbf{x}(t_{i+1}) = \widetilde{\mathbf{x}}_i - \mathbf{x}(t_i) \\ &+ h_i \left[\widetilde{\mathbf{v}}_i - v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c}) \right] \\ &+ h_i \left[v_\theta(\widetilde{\mathbf{x}}_i, t_i, \mathbf{c}) - v_\theta(\mathbf{x}(t_i), t_i, \mathbf{c}) \right] \\ &+ h_i \left[v_\theta(\mathbf{x}(t_i), t_i, \mathbf{c}) - v^\star(\mathbf{x}(t_i), t_i, \mathbf{c}) \right] - \boldsymbol{\rho}_i. \end{aligned} \tag{36}$$

Applying the triangle inequality and A1–A4 gives the one-step recursion

$$\overline{\Delta}_{i+1} \leq (1 + |h_i| \Lambda_i) \, \overline{\Delta}_i + z_i |h_i| E_i + |h_i| \eta_i + \mathcal{D}_i^{(n)}. \tag{37}$$

Unrolling Eq. (37) from $i = 0$ to $T - 1$ with $\overline{\Delta}_0 = 0$, using the same induction argument as in Section B.1, proves Eq. (33). □

To make the accumulated discretization terms explicit, suppose $\Lambda_i \leq \Lambda$, $\eta_i \leq \eta$, and $M_i^{(k)} \leq M^{(k)}$. Let $H := \sum_{i=0}^{T-1} |h_i|$ and $h_{\max} := \max_i |h_i|$. Since $1 + a \leq \exp(a)$, Theorem B.1 further implies

$$\overline{\Delta}_T \leq e^{\Lambda H} \left(\sum_{i=0}^{T-1} z_i |h_i| E_i + \eta H + H \sum_{k=2}^{n} \frac{M^{(k-1)}}{k!} h_{\max}^{k-1} \right). \tag{38}$$

For $n = 2$, Eq. (32) reduces to $\mathcal{D}_i^{(2)} = |h_i|^2 M_i^{(1)}/2$, recovering the usual second-order Taylor remainder. Larger n gives a finer decomposition of $\boldsymbol{\rho}_i$ and retains all corresponding terms in the multi-step terminal bound. Since the numerical update remains Euler, the leading discretization term is still $O(h_{\max})$ globally; increasing the expansion order does not by itself change the solver order.

C Experimental Settings

C.1 Metrics

We evaluate the allocation objective through fidelity to the full-computation output and separately assess reference-free generation quality. For text-to-image generation, we use CLIP Score, CycleReward, and HPS v2.1 to measure prompt alignment and perceptual preference. For both image and video generation, the corresponding 50-step full-computation output is treated as the paired reference for LPIPS, SSIM, and PSNR. Video generation quality is additionally evaluated using seven VBench dimensions. Unless otherwise stated, all metrics are averaged over the full evaluation prompt set.

CLIP Score. CLIP Score (Radford et al., 2021) measures text–image alignment through the cosine similarity between pretrained CLIP embeddings of the input prompt and generated image. We report results using both the base and large visual encoders, denoted as CLIP-B and CLIP-L, respectively. Higher scores indicate stronger semantic correspondence with the prompt.

CycleReward. CycleReward (Bahng et al., 2025) is a learned image–text alignment metric trained on preferences derived from cycle consistency rather than direct human annotations. It rewards generated images whose visual content supports an accurate reconstruction of the conditioning text, making it particularly sensitive to detailed prompt compliance. Higher scores indicate better image–text alignment.

HPS v2.1. Human Preference Score v2.1 (HPS v2.1) (Wu et al., 2023) evaluates generated images with a CLIP-based preference model trained to predict human choices. It captures perceptual quality and semantic agreement with the input prompt; higher values indicate stronger predicted human preference.

LPIPS. Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018) compares a generated output I with its full-computation reference $I^\star$ using deep feature representations:

$$\mathrm{LPIPS}(I, I^\star) = \sum_\ell \alpha_\ell \, \mathrm{Dist}(F_\ell(I), F_\ell(I^\star)), \tag{39}$$

where F_ℓ denotes the feature map at layer ℓ and α_ℓ is its learned weight. Lower LPIPS indicates greater perceptual similarity.

SSIM. The Structural Similarity Index Measure (SSIM) (Wang et al., 2004) evaluates luminance, contrast, and structural agreement between an output x and its reference y:

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{40}$$

where μ_x and μ_y are the means, σ_x^2 and σ_y^2 are the variances, σ_{xy} is the covariance, and C_1, C_2 are stability constants. Higher SSIM indicates stronger structural consistency.

PSNR. Peak Signal-to-Noise Ratio (PSNR) measures pixel-level reconstruction fidelity:

$$\mathrm{PSNR}(x, y) = 10 \log_{10}\left(\frac{R^2}{\mathrm{MSE}(x, y)}\right), \tag{41}$$

where R is the maximum representable pixel value and $\mathrm{MSE}(x, y)$ is the mean squared error. Higher PSNR indicates smaller pixel-level distortion. For videos, LPIPS, SSIM, and PSNR are first computed frame by frame and then averaged across frames and samples.

VBench. VBench (Huang et al., 2024) provides a multidimensional evaluation of video generation quality. We use its official implementation and report the seven reference-free dimensions under our custom-input evaluation protocol: Subject Consistency, Background Consistency, Temporal Flickering, Motion Smoothness, Dynamic Degree, Aesthetic Quality, and Imaging Quality. The VBench score reported in our experiments aggregates only these seven dimensions, rather than the complete

Table 5: PRICE configuration across different models. N_L and N_{bank} denote the numbers of calibration prompts used to estimate local sensitivity L_i and construct each risk bank, respectively.

Model	**Budget** B	$\mathbf{N_L}$	$\mathbf{N_{bank}}$
FLUX.1-dev	$\{9, 15, 21\}$	32	32
Qwen-Image	$\{7, 13\}$	32	32
Wan2.1	$\{13, 21\}$	32	32

16-dimensional suite. The remaining nine dimensions require the official prompt suite under the standard evaluation protocol and are not evaluated here. Prompt selection details are provided in Appendix D.1. Higher values indicate better video quality on the evaluated dimensions.

C.2 Model Configuration

The full-computation budget B specifies how many denoising steps execute the transformer stack; the remaining steps reuse a cached hidden-state residual. PRICE allocates full evaluations online using the calibrated terminal-risk score. For every model, we use the same 32-prompt calibration set to estimate local sensitivity L_i from uncached trajectories and construct each budget-specific risk bank $\mathbf{R}^{(B)}$. Calibration and evaluation prompts do not overlap. We freeze all calibration outputs before evaluation and use no quality metrics during calibration. Table 5 lists the configurations.

C.3 Baseline Settings

All caching baselines are implemented using their official codebases. For threshold-based methods, we tune the control threshold separately for each model and target budget so that the realized mean NFE matches B as closely as possible; individual samples may still have different NFEs because these baselines make input-dependent decisions. In contrast, PRICE satisfies B exactly for every sample. For TaylorSeer, S denotes the caching interval and O denotes `max_order`. The final frozen settings used in our experiments are described below.

FLUX.1-dev. At $B = 9$, TeaCache uses $\delta = 1.202$, TaylorSeer uses $S = 10$ and $O = 2$, SeaCache uses $\delta = 0.956$, MagCache uses $K = 5$ with $\delta = 2.0109$, and SenCache uses $\delta = 2896.31$. At $B = 15$, TeaCache uses $\delta = 0.600$, TaylorSeer uses $S = 5$ and $O = 2$, SeaCache uses $\delta = 0.490$, MagCache uses $K = 5$ with $\delta = 0.16471$, and SenCache uses $\delta = 608.874$. At $B = 21$, TeaCache uses $\delta = 0.380$, TaylorSeer uses $S = 3$ and $O = 2$, SeaCache uses $\delta = 0.290$, MagCache uses $K = 5$ with $\delta = 0.05439$, and SenCache uses $\delta = 256$.

Qwen-Image. At $B = 7$, TeaCache uses $\delta = 1.024$, and TaylorSeer uses $S = 12$ and $O = 2$. At $B = 13$, TeaCache uses $\delta = 0.508$, and TaylorSeer uses $S = 5$ and $O = 2$.

Wan2.1. At $B = 13$, TeaCache, SeaCache, and MagCache use $\delta = 0.250$, $\delta = 0.53125$, and $(K, \delta) = (5, 0.1875)$, respectively. At $B = 21$, their corresponding settings are $\delta = 0.125$, $\delta = 0.250$, and $(K, \delta) = (5, 0.125)$.

The official repositories used for all caching baselines are:

- **TeaCache**: `https://github.com/ali-vilab/TeaCache`
- **TaylorSeer**: `https://github.com/Shenyi-Z/TaylorSeer`
- **SeaCache**: `https://github.com/jiwoogit/SeaCache`
- **MagCache**: `https://github.com/Zehong-Ma/MagCache`
- **SenCache**: `https://github.com/vita-epfl/SenCache`
- **BudCache**: `https://github.com/Westlake-AGI-Lab/BudCache`
- **ERTACache**: `https://github.com/bytedance/ERTACache`

C.4 Hardware and Time Cost

The text-to-image experiments are executed on NVIDIA A100-SXM4 GPUs with 80 GB memory, while the Wan2.1 experiments use eight Huawei Ascend 910B3 NPUs with 64 GB HBM each. Be-

Table 6: Hardware configuration and estimated time cost of PRICE. Image-model costs are normalized to two A100 GPUs; video-model costs use eight Ascend 910B3 NPUs. The reported hours cover calibration and sample generation but exclude model loading and metric post-processing.

Model	**Hardware Config.**	**Budget** B	**Time Cost (h)**
FLUX.1-dev	2× A100 80GB	9	≈ 0.18
		15	≈ 0.21
		21	≈ 0.29
Qwen-Image	2× A100 80GB	7	≈ 0.21
		13	≈ 0.32
Wan2.1	8× Ascend 910B3	13	≈ 0.46
		21	≈ 0.63

cause the available number of A100 GPUs varied across individual jobs, we normalize the FLUX.1-dev and Qwen-Image costs to a two-A100 configuration for consistent reporting. Wan2.1 is reported using the actual eight-NPU configuration.

Table 6 summarizes the resulting generation cost. The estimates use the measured per-sample end-to-end latency reported in the main paper, the evaluation-set sizes (200 DrawBench prompts for image generation and 100 VBench prompts for video generation), and ideal data parallelism across the listed devices. They additionally include the budget-specific 32-prompt risk-bank construction and amortize the shared 32-prompt estimation of local sensitivity L_i across the evaluated budgets. Model loading and metric post-processing are excluded because they are independent of the cache budget.

D Additional Experimental Results

D.1 More Qualitative Results on Wan2.1

Prompt selection. Following the selection protocol of BAG (Zhao et al., 2026a), we construct the 100-prompt evaluation set by sampling uniformly across the 19 categories in the VBench prompt pool. We also follow its prompt-isolation rules when separating evaluation from calibration: a calibration prompt is excluded if it is identical to an evaluation prompt, forms a substring pair with one, or shares the same base sentence while differing only in style or camera suffixes. These checks keep the calibration and evaluation sets disjoint.

We evaluate all seven reported VBench dimensions on Wan2.1 using the same two budgets and protocol as in the main paper. Tables 7 and 8 report the results at $B = 13$ and $B = 21$, respectively.

Table 7: Complete VBench results on Wan2.1 at $B = 13$. All values are reported as percentages.

Method	Subject	Background	Motion	Flickering	Dynamic	Aesthetic	Imaging	Average
Vanilla 13 steps	90.67	94.02	96.89	95.13	94.00	55.50	60.19	83.77
SeaCache ($\delta = 0.53$)	92.33	94.91	98.07	96.76	87.00	54.20	66.07	84.19
MagCache ($K = 5, \delta = 0.19$)	92.23	94.80	98.02	96.82	88.00	53.95	63.95	83.97
BudCache ($B = 13$)	90.63	93.42	97.96	96.73	76.00	52.10	66.23	81.87
TeaCache ($\delta = 0.25$)	92.36	95.00	97.98	96.72	83.00	54.17	64.68	83.41
ERTACache ($B = 13$)	89.70	93.16	97.88	96.70	76.00	50.76	65.76	81.42
PRICE ($B = 13$)	90.12	93.65	97.88	96.67	80.00	50.07	65.47	81.98

Table 8: Complete VBench results on Wan2.1 at $B = 21$. All values are reported as percentages.

Method	Subject	Background	Motion	Flickering	Dynamic	Aesthetic	Imaging	Average
Vanilla 21 steps	91.26	94.29	97.45	95.67	91.00	56.74	63.00	84.20
MagCache ($K = 5, \delta = 0.13$)	93.00	95.15	97.98	96.50	91.00	56.30	66.98	85.27
BudCache ($B = 21$)	93.07	95.12	98.05	96.59	91.00	56.17	66.96	85.28
TeaCache ($\delta = 0.13$)	92.80	95.09	97.94	96.51	89.00	56.13	66.51	84.85
ERTACache ($B = 21$)	92.66	94.85	98.04	96.60	90.00	55.43	66.95	84.93
SeaCache ($\delta = 0.25$)	93.03	95.13	98.01	96.51	91.00	56.28	67.14	85.30
PRICE ($B = 21$)	92.47	94.74	98.02	96.55	91.00	55.11	67.25	85.02

At $B = 13$, PRICE obtains an average VBench score of 81.98%, which is 2.21 percentage points below the strongest caching baseline, SeaCache at 84.19%. Its dimension-wise scores remain comparable to the matched-budget baselines while it achieves the best reconstruction fidelity in Table 4. At $B = 21$, PRICE reaches 85.02%, remaining within 0.28 percentage points of the best result, SeaCache at 85.30%, while attaining the highest Imaging Quality score of 67.25%. These per-dimension results indicate competitive video quality at both budgets, alongside the reconstruction fidelity that PRICE targets.

D.2 Additional Ablation Results

The full risk-score ablation on FLUX.1-dev covers $B \in \{9, 15, 21\}$. The complete score gives the best reconstruction fidelity at all three budgets (Table 9). Local discrepancy S_i is the strongest overall component, and omitting it substantially degrades fidelity. As in the main $B = 9$ ablation, solver scaling and downstream propagation help translate this local signal into an estimate of final-output error.

Table 10 compares a Global threshold, $\tau = F^{-1}(m/n)$, with the proposed Non-global suffix threshold, $\tau_i = F_i^{-1}(m_i/n_i)$, at the same mean NFE. The Non-global rule improves LPIPS, SSIM, and PSNR at every budget; its LPIPS decreases from 0.3928 to 0.2869 at $B = 9$, from 0.1812 to 0.1604 at $B = 15$, and from 0.0893 to 0.0788 at $B = 21$. At matched mean NFE, these results support adjusting the risk threshold to the remaining cache quota and eligible suffix.

Table 9: Additional ablation of the risk-index formulation on **FLUX.1-dev** at $B \in \{9, 15, 21\}$. All variants satisfy the exact requested budget. Gray rows denote the complete formulation (best: bold; second best: underlined).

Budget	Risk index R_i	LPIPS ↓	SSIM ↑	PSNR ↑
9	w_i	0.5406	0.6384	18.913
9	S_i	<u>0.3321</u>	<u>0.7322</u>	19.262
9	$w_i S_i$	0.3508	0.7265	<u>20.011</u>
9	$w_i \lvert h_i \rvert$	0.5282	0.6700	18.823
9	$\lvert h_i \rvert S_i$	0.3655	0.7248	18.277
9	$\boldsymbol{w_i \lvert h_i \rvert S_i}$	**0.2869**	**0.7619**	**21.429**
15	w_i	0.4653	0.6893	20.952
15	S_i	<u>0.1712</u>	<u>0.8425</u>	23.691
15	$w_i S_i$	0.2627	0.7882	<u>23.889</u>
15	$w_i \lvert h_i \rvert$	0.5148	0.6808	19.057
15	$\lvert h_i \rvert S_i$	0.2249	0.8055	21.092
15	$\boldsymbol{w_i \lvert h_i \rvert S_i}$	**0.1604**	**0.8526**	**25.089**
21	w_i	0.2876	0.7764	23.405
21	S_i	<u>0.0861</u>	<u>0.8820</u>	<u>26.446</u>
21	$w_i S_i$	0.1843	0.8353	26.340
21	$w_i \lvert h_i \rvert$	0.4166	0.7304	20.667
21	$\lvert h_i \rvert S_i$	0.1547	0.8544	23.698
21	$\boldsymbol{w_i \lvert h_i \rvert S_i}$	**0.0788**	**0.9172**	**28.513**

Table 10: Additional ablation of threshold allocation on **FLUX.1-dev** at $B \in \{9, 15, 21\}$. Global uses one quantile over the complete risk bank, whereas Non-global recomputes a suffix-specific quantile from the remaining quota and eligible positions. Gray rows denote the proposed Non-global rule.

Budget	Choices	Variant	Mean NFE	LPIPS ↓	SSIM ↑	PSNR ↑
9	Global	$\tau = F^{-1}(m/n)$	9.00	0.3928	0.7144	19.359
9	**Non-global**	$\boldsymbol{\tau_i = F_i^{-1}(m_i/n_i)}$	9.00	**0.2869**	**0.7619**	**21.429**
15	Global	$\tau = F^{-1}(m/n)$	15.00	0.1812	0.8394	24.748
15	**Non-global**	$\boldsymbol{\tau_i = F_i^{-1}(m_i/n_i)}$	15.00	**0.1604**	**0.8526**	**25.089**
21	Global	$\tau = F^{-1}(m/n)$	21.00	0.0893	0.9077	28.075
21	**Non-global**	$\boldsymbol{\tau_i = F_i^{-1}(m_i/n_i)}$	21.00	**0.0788**	**0.9172**	**28.513**

D.3 More Experiments for Figure 2

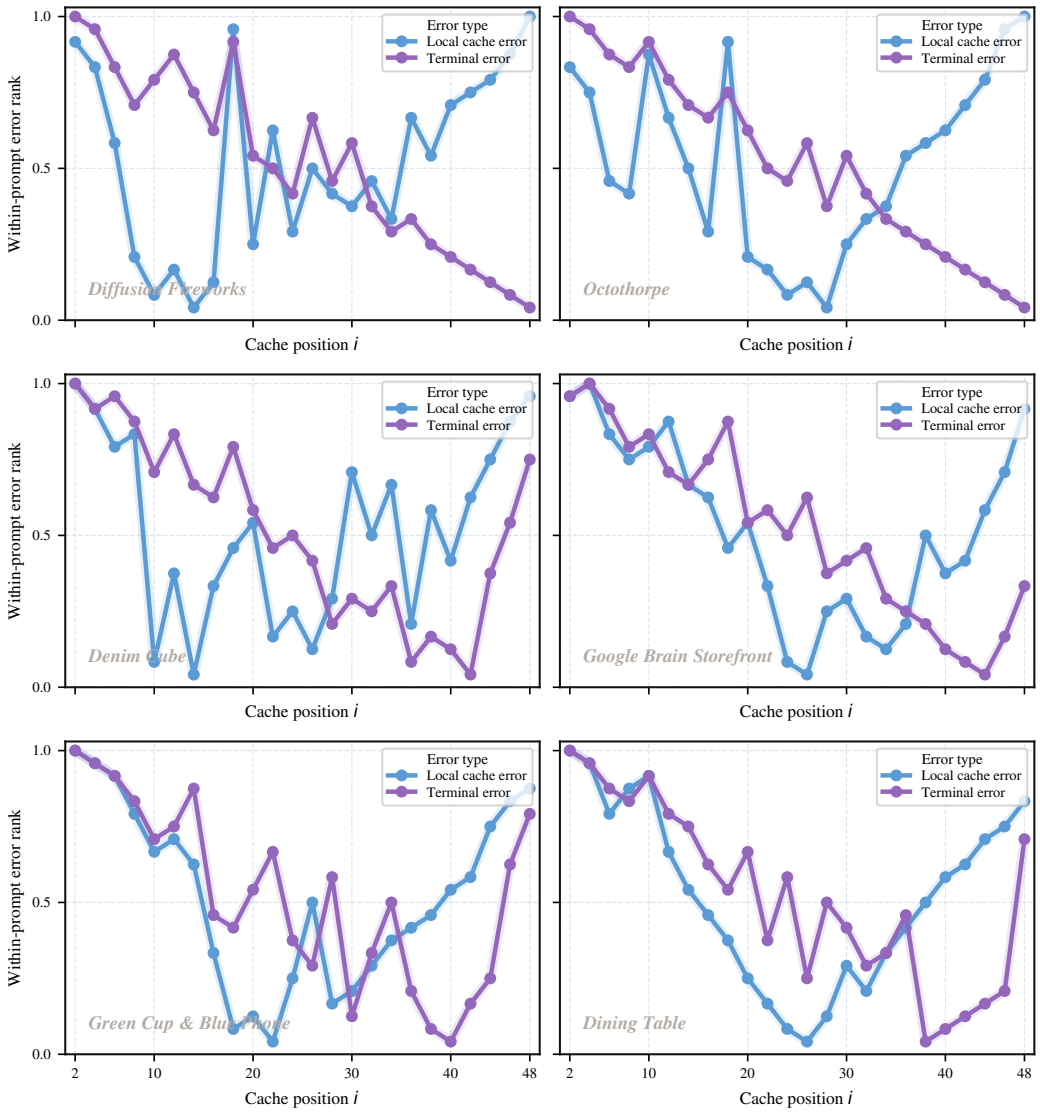

Figure 7: More experiments for Figure 2 on FLUX.1-dev (top), Qwen-Image (middle), and Wan2.1 (bottom).

E ADDITIONAL QUALITATIVE RESULTS

E.1 FLUX.1-DEV

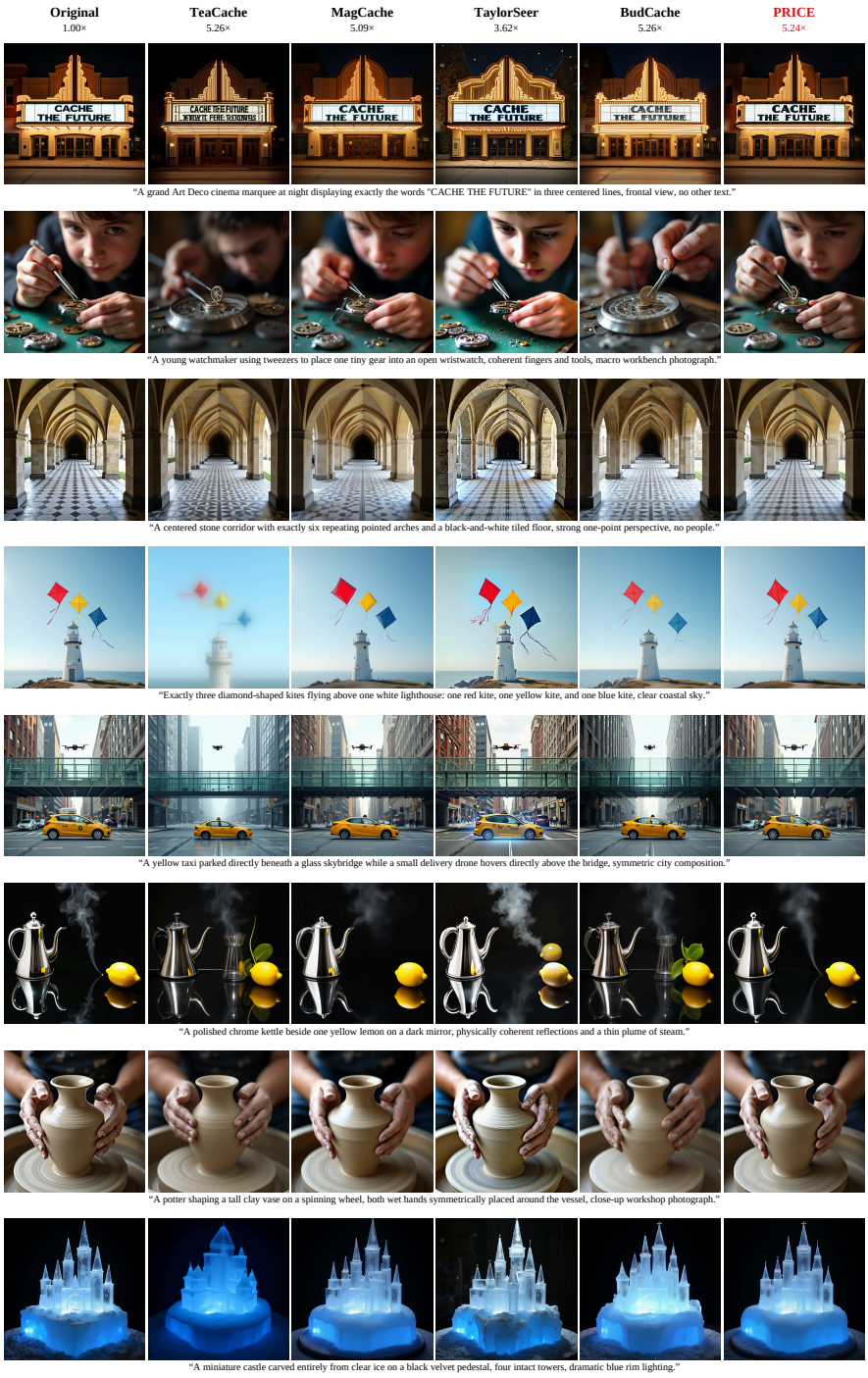

Figure 8: Additional qualitative comparisons on FLUX.1-dev at $B = 9$ (NFE $= 9$). All methods use the same prompt and initial noise; Original denotes the corresponding 50-step full-computation reference.

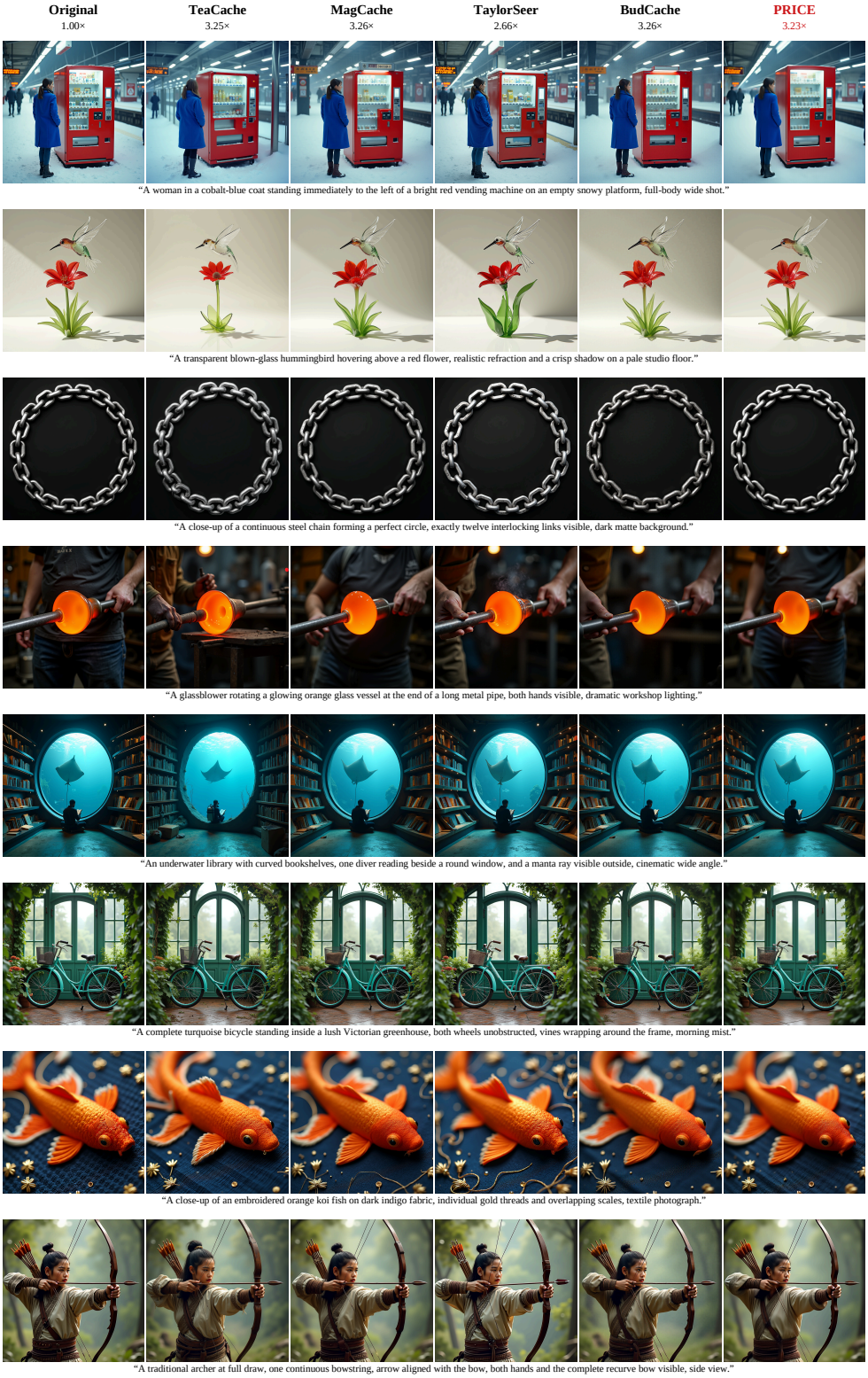

Figure 9: Additional qualitative comparisons on FLUX.1-dev at $B = 15$ (NFE $= 15$). All methods use the same prompt and initial noise; Original denotes the corresponding 50-step full-computation reference.

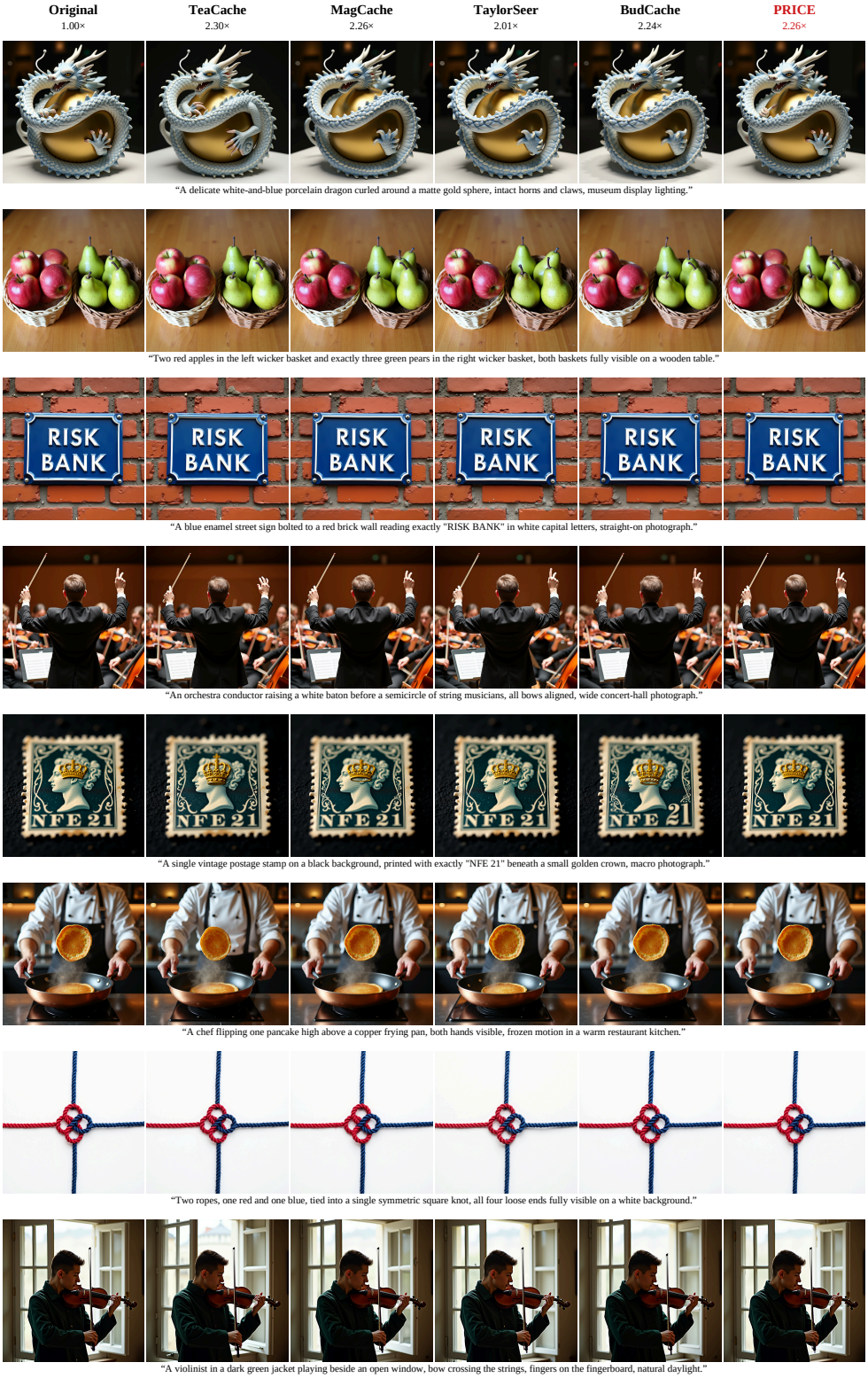

Figure 10: Additional qualitative comparisons on FLUX.1-dev at $B = 21$ (NFE $= 21$). All methods use the same prompt and initial noise; Original denotes the corresponding 50-step full-computation reference.

E.2 WAN2.1



Figure 11: Additional qualitative comparisons on Wan2.1 at $B = 13$ (NFE $= 13$), showing six uniformly sampled frames. All methods use the same prompt and initial noise; Original denotes the corresponding 50-step full-computation reference.

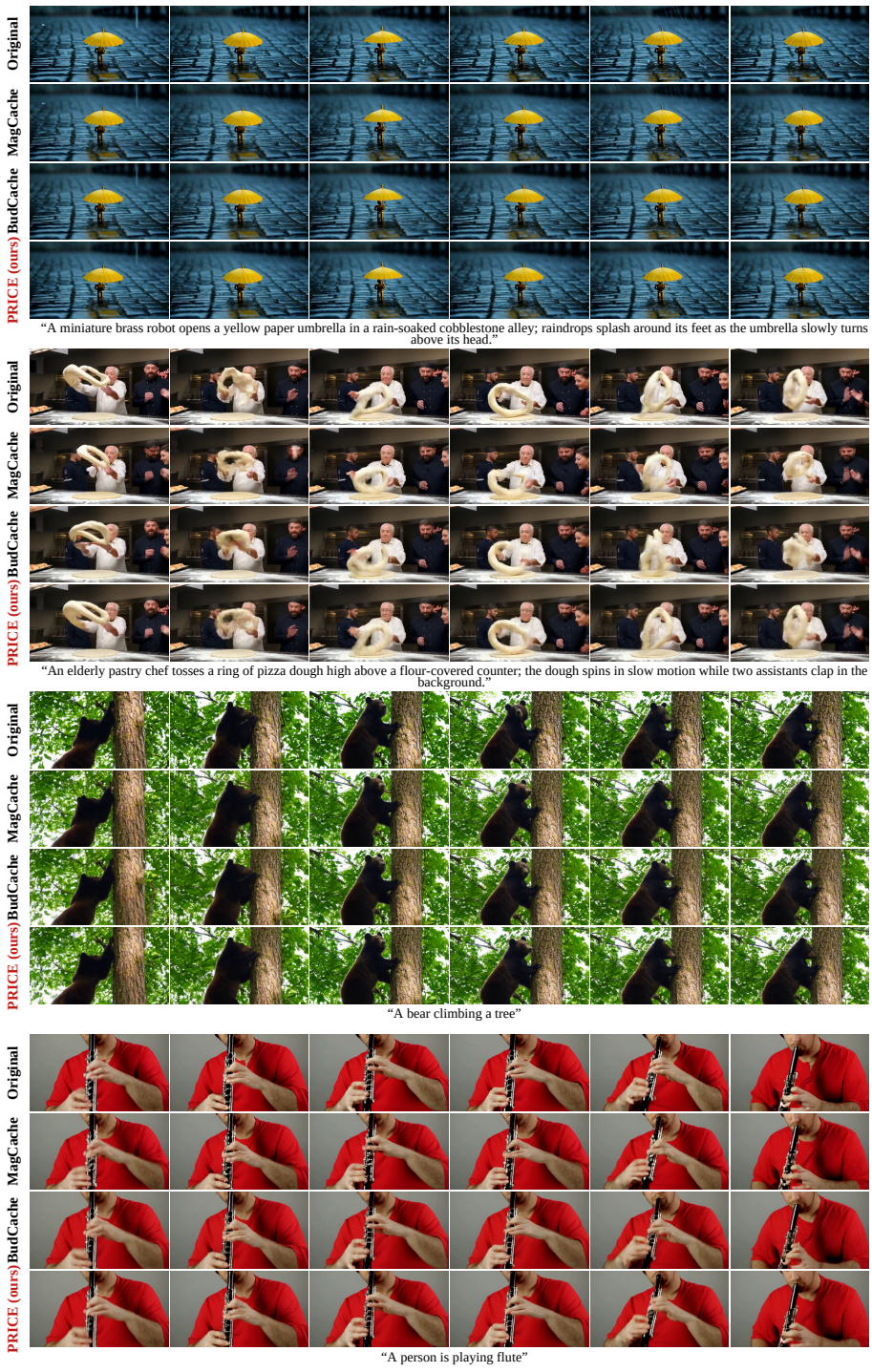

Figure 12: Additional qualitative comparisons on Wan2.1 at $B = 21$ (NFE $= 21$), showing six uniformly sampled frames. All methods use the same prompt and initial noise; Original denotes the corresponding 50-step full-computation reference.

E.3 QWEN-IMAGE

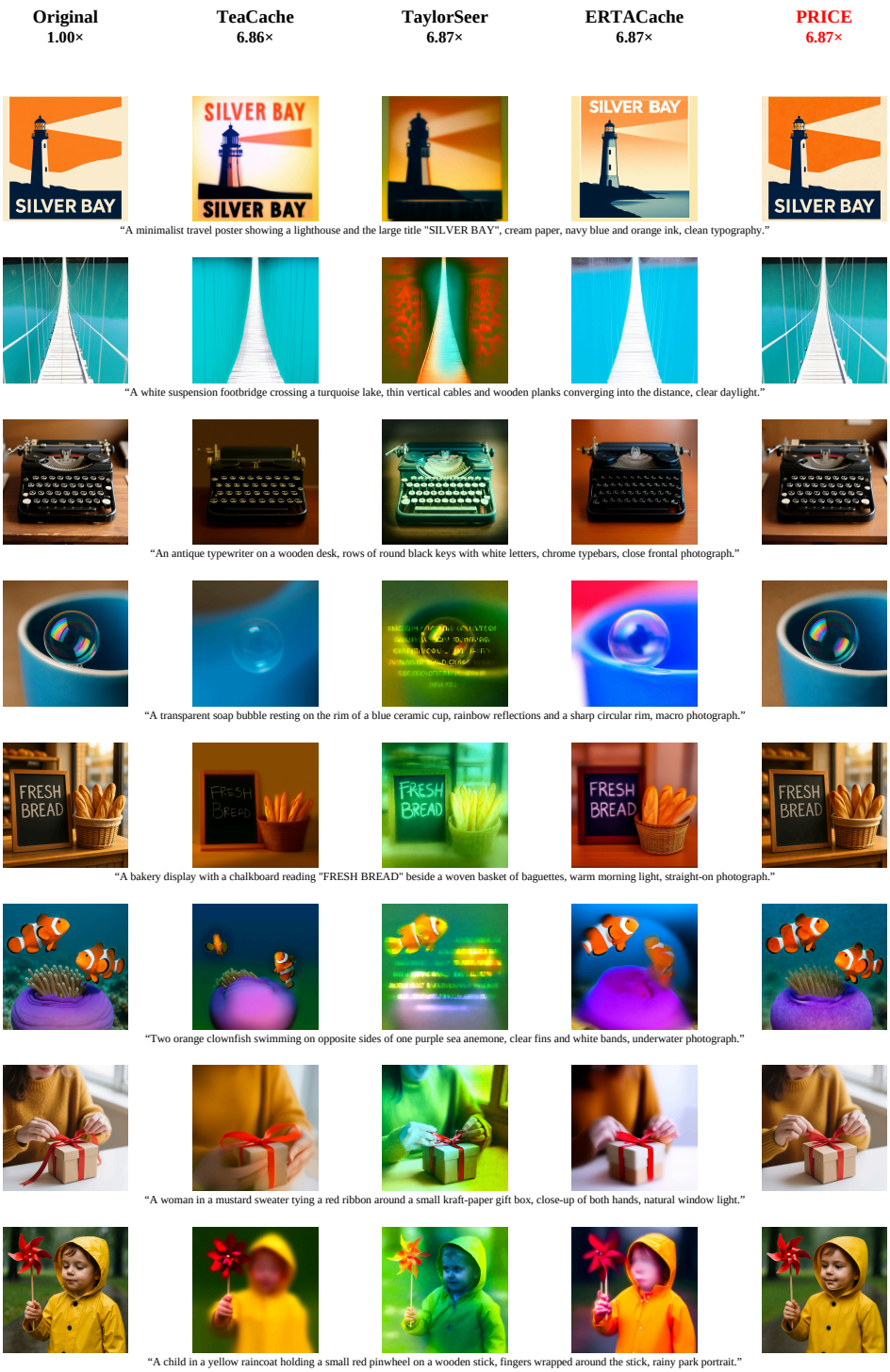

Figure 13: Additional qualitative comparisons on Qwen-Image at $B = 7$ (mean NFE $= 7$). All methods use the same prompt and initial noise; Original denotes the corresponding 50-step full-computation reference.



Figure 14: Additional qualitative comparisons on Qwen-Image at $B = 13$ (mean NFE $= 13$). All methods use the same prompt and initial noise; Original denotes the corresponding 50-step full-computation reference.