VELOCACHE: FREQUENCY-DECOUPLED VELOCITY CACHING FOR TRAINING-FREE VIDEO DIFFUSION ACCELERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Training-free step caches accelerate video diffusion by reusing cached computation at selected denoising steps. Prior work mostly refines *when* to reuse; *what* is reused is typically the whole cached signal, which can wash out texture, or per-block feature forecasts whose cost grows with model size. Analyzing the per-step spectrum of the denoiser output, we find a *volatility crossover*: about one fifth into the trajectory, changes in the low spatial band that carries coarse structure decline, while the high band that carries fine texture keeps changing. This motivates two decisions. *What to reuse:* **VeloCache** retains the cached low band and advances the high band by a first-order trend from the two most recent computed outputs. Only true evaluations refresh this history, so skipped steps never feed predictions back into the cache. A local error analysis gives the condition under which the correction improves on replay, and shows how unequal sampling intervals limit it. *When to evaluate:* an early segment set by the pooled crossover is computed in full and the remaining budget is spread uniformly, yielding one shared rule across backbones. VeloCache reads no internal features and stores two outputs per guidance stream. At comparable evaluation budgets, it reaches the lowest LPIPS to the full-step reference on Wan2.1 and CogVideoX (0.052 vs. 0.080 on Wan2.1), competitive LPIPS on Cosmos3-Nano, and the highest VBench quality of the compared caches on all three backbones. Code will be released.

1 INTRODUCTION

Video diffusion models (Blattmann et al., 2023), from Sora (Brooks et al., 2024) to open backbones such as Cosmos (NVIDIA, 2025), Wan (Team Wan et al., 2025) and CogVideoX (Yang et al., 2025), synthesize high-fidelity clips. Our evaluated backbones require dozens of evaluations of a billion-scale denoiser. In our setup, full-step Wan2.1-14B at 720p takes an estimated 31 minutes of denoising time on one H100 (App. D.2). *Step caching* is a training-free remedy: replace selected network evaluations with cached computation, leaving the sampler and weights untouched (Liu et al., 2025a; Ma et al., 2025).

A step cache decides *when* to skip a step and *what* to put in its place. Prior work concentrates on the first: TeaCache (Liu et al., 2025a) estimates output change from timestep-modulated inputs, MagCache (Ma et al., 2025) calibrates a per-step magnitude curve, and EasyCache (Zhou et al., 2025) adapts its reuse criterion at runtime. These methods reuse a cached output or residual as a whole, with method-specific scaling or input corrections. Such reuse costs little, but can leave detail stale and over-smooth texture (Fig. 1). TaylorSeer (Liu et al., 2025c) instead forecasts per-block activations, paying reconstruction costs at skipped steps and storing a feature cache that grows with the model—about 33 GB for the 16B Cosmos backbone by the shapes its blocks emit (§4.2). On image models, FreqCa (Liu et al., 2025b) reuses the low band of a cumulative residual feature and predicts the high band with a second-order interpolator. Can an output-level reuse rule improve on whole-signal replay at similarly low storage and reconstruction cost?

We investigate this question through the spatial spectrum of the denoiser output. Along full-step trajectories, low-band changes decline early while high-band changes remain larger later in sampling (Fig. 2). Their magnitudes cross at a pooled fraction 0.22 ± 0.10 across three backbones. This

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

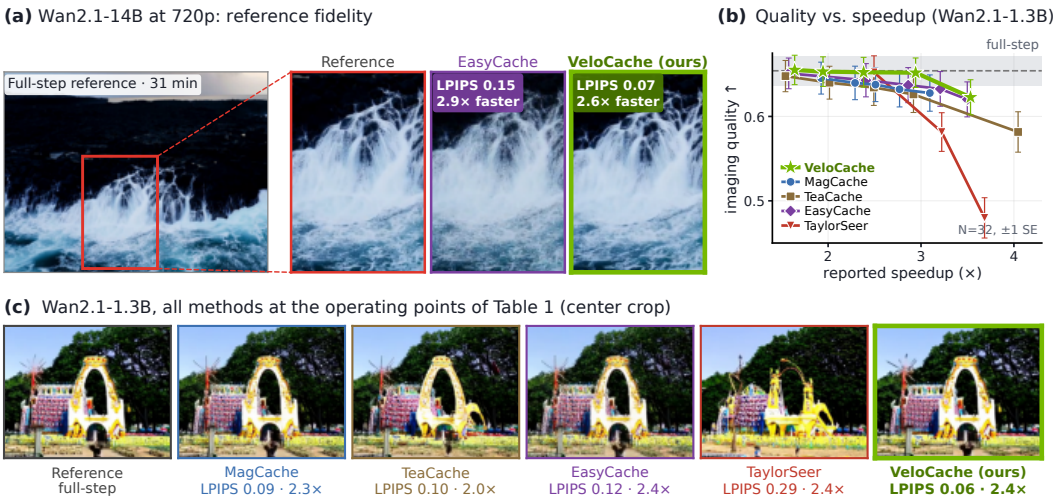

Figure 1: **What to reuse, not only when to skip.** **(a)** Wan2.1-14B at 720p: reference (50 steps, estimated 31 min denoising on one H100), released-setting EasyCache and VeloCache (40 forwards each). Badges show per-clip LPIPS; tile speedups use the earlier per-step convention, superseded by the budget ratios in App. D.2. VeloCache preserves more wave spray. **(b)** Wan2.1-1.3B imaging versus reported acceleration (32 prompts, ± 1 SE). VeloCache stays near the reference mean through $2.9\times$; timing conventions are in App. B. **(c)** One Wan2.1-1.3B clip at Table 1 settings; badges give per-clip LPIPS and aggregate speedup.

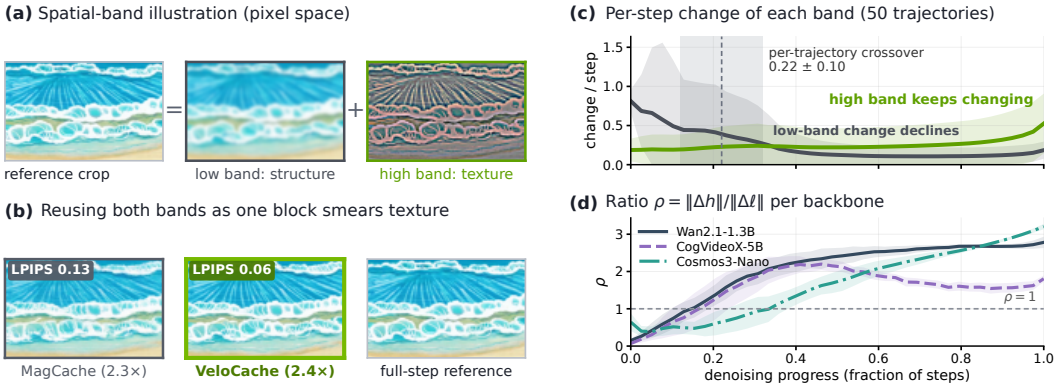

Figure 2: **Frequency-dependent output dynamics.** **(a)** Pixel-space low/high spatial bands (high band rescaled), not latent velocities. **(b)** Wan2.1 crops under MagCache and VeloCache, with per-clip LPIPS and reported acceleration. **(c)** Band changes on 50 full-step trajectories (18 Wan, 16 CogVideoX, 16 Cosmos), normalized by the sum of low/high change norms at the first update (shading: ± 1 standard deviation). **(d)** The ratio $\rho = \|\Delta h\|/\|\Delta \ell\|$ first exceeds 1 at a pooled fraction 0.22 ± 0.10: Cosmos 10.7 ± 4.1 of 35 steps, CogVideoX 9.6 ± 3.0 and Wan 8.5 ± 2.5 of 50. This describes relative band changes; it does not imply zero low-band drift or accurate extrapolation.

volatility crossover reveals a mismatch in whole-signal replay: the same reuse rule is applied to components with different rates of change. Whole-output replay can preserve the slowly changing coarse component while leaving fine detail stale. This motivates retaining the low band while using recent high-band changes to correct the output.

We act on that asymmetry at the denoiser output, the single tensor the sampler consumes, so the correction needs no layer bookkeeping and no reconstruction path.

VeloCache implements this idea directly at the model output. At a skipped step, it retains the cached low band and adds a first-order high-band increment computed from the two latest network outputs. The pooled crossover sets an early protected segment; the remaining computation budget is spread uniformly to refresh the cache. Only true network outputs enter the history. With two cached outputs per guidance stream, storage depends on latent size rather than network depth, and the rule requires no hidden-feature access. Our local error analysis identifies how trend alignment and correction magnitude jointly determine the benefit over replay, and accounts for unequal sampling intervals.

We evaluate one default recipe on Wan2.1, CogVideoX and Cosmos3-Nano. VeloCache attains the highest mean over six selected VBench quality dimensions (Huang et al., 2024) and the highest imaging score among the tested caches on all three backbones, the lowest LPIPS (Zhang et al., 2018) to the full-step reference on Wan2.1 and CogVideoX, and competitive LPIPS on Cosmos3-Nano. Fixed-schedule comparisons show that replacing direct output replay with the high-band correction improves reference fidelity on each backbone, isolating the reuse rule from step selection. Our contributions are threefold:

- **Analysis.** We identify a volatility crossover in the output spectrum across three video backbones. Our local analysis gives the condition for high-band correction to improve on replay and a conditional error bound that accounts for unequal sampling intervals.
- **Method.** We introduce VeloCache: a high-band correction from two cached outputs, coupled with early trajectory protection and uniformly spaced later evaluations. The method uses one gain, filter and schedule rule across backbones, and requires no internal features.
- **Experiments.** Evaluation across three video diffusion backbones shows that VeloCache achieves the highest mean score over six selected VBench quality dimensions among the compared caches on each backbone. It attains the lowest LPIPS to the full-step reference on Wan2.1 and CogVideoX, and competitive LPIPS on Cosmos3-Nano.

2 Related Work

Video diffusion and acceleration. Denoising, latent and flow formulations (Ho et al., 2020; Rombach et al., 2022; Lipman et al., 2023) and advances in transformer denoisers (Peebles & Xie, 2023) underpin recent video generators (Brooks et al., 2024; Blattmann et al., 2023; Yang et al., 2025; Team Wan et al., 2025; Kong et al., 2024; NVIDIA, 2025). Fast samplers reduce network evaluations through numerical integration (Song et al., 2021; Lu et al., 2022; Zhao et al., 2023), while distillation and consistency training learn one- or few-step generation (Salimans & Ho, 2022; Song et al., 2023; Luo et al., 2023; Yin et al., 2024). Coarse-to-fine generation and noise-dependent perceptual learning provide insight into denoising dynamics (Rissanen et al., 2023; Choi et al., 2022). FreeU reweights backbone and skip features (Si et al., 2024), while WaveDiff uses wavelet decomposition to process low- and high-frequency components (Phung et al., 2023). VeloCache uses spatial-frequency dynamics to reduce computation within sampling steps. The pretrained weights and solver remain fixed: every point on the original time grid still receives a sampler update, using either a freshly computed output or a cached approximation.

Caching and forecasting. Caching methods choose both the computations to reuse and their refresh schedule. DeepCache and FORA replay internal features (Ma et al., 2024b; Selvaraju et al., 2024); ToCa selects tokens and PAB assigns attention-specific reuse intervals (Zou et al., 2025; Zhao et al., 2025). Block Caching learns feature corrections, while Learning-to-Cache learns layer routing (Wimbauer et al., 2024; Ma et al., 2024a). TeaCache, MagCache, EasyCache and AdaCache adapt reuse using proxies for output or feature changes (Liu et al., 2025a; Ma et al., 2025; Zhou et al., 2025; Kahatapitiya et al., 2025). Prediction changes the reused quantity: TaylorSeer forecasts per-block features (Liu et al., 2025c), FasterCache updates attention features from two cached states and amplifies frequency-split guidance residuals (Lv et al., 2025), and FreqCa reuses the low band of a cumulative residual feature while predicting its high band on image models (Liu et al., 2025b). SeaCache bases reuse decisions on a spectrally aligned representation (Chung et al., 2026); SpectralCache jointly gates reuse with asymmetric thresholds on feature-dimension bands (Li, 2026). Cache-DiT provides infrastructure for caching and other inference optimizations (DefTruth, vipshop.com, etc., 2025). VeloCache corrects the high band from two cached outputs on a shared

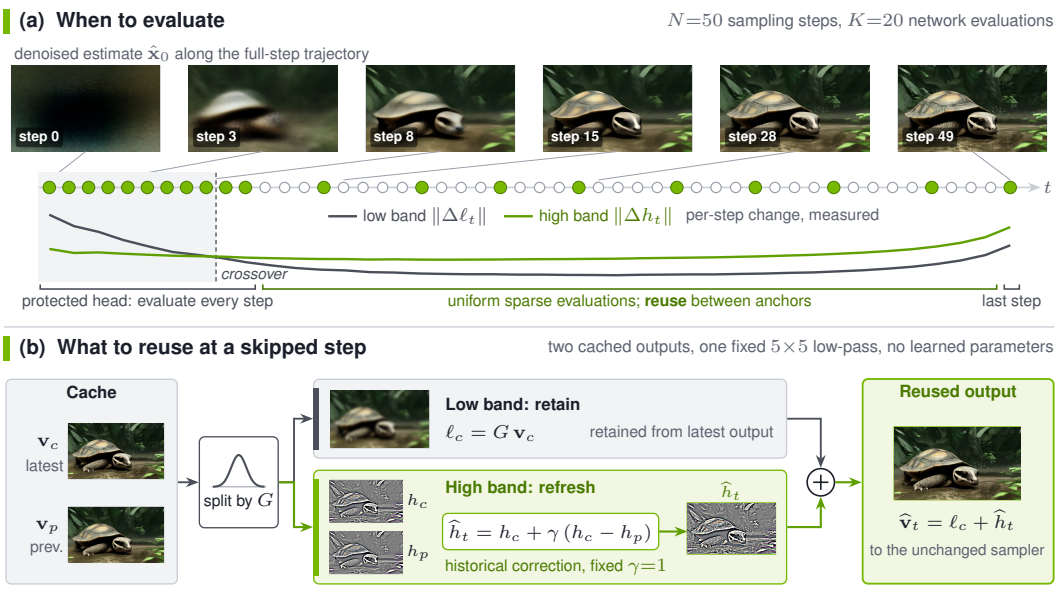

Figure 3: **VeloCache overview.** **(a)** Denoised estimates along a full-step Wan2.1 trajectory and the schedule ($N = 50$, $K = 20$ true evaluations, filled): the first 11 steps are evaluated and the remaining budget is spread uniformly, including the final step. Band-change curves motivate early protection. **(b)** Two cached outputs are split by G; the low component is retained and the high component receives a historical trend correction ($\gamma = 1$, Eq. (1)); tiles are pixel-space illustrations.

schedule. Storage and filtering depend on latent size, independently of depth, without blockwise reconstruction.

3 Method

A step cache must decide *what* to reuse and *when* to evaluate again. VeloCache addresses these choices with a frequency-selective output correction and a schedule that protects early sampling. Figure 3 illustrates the design; Algorithm 1 gives the sampling loop.

3.1 Problem formulation

Let $t \in \{0, \dots, N-1\}$ index the sampling steps and τ_t denote the sampler time, which need not be uniformly spaced. The denoiser maps latent $\mathbf{x}_t$ to $\mathbf{v}_t = f_\theta(\mathbf{x}_t, \tau_t)$; conditioning is suppressed here and restored in Algorithm 1. We use $\mathbf{v}$ for the model output consumed by the sampler: flow velocity for Wan/Cosmos and native v-prediction for CogVideoX. The sampler applies $\mathbf{x}_{t+1} = \Phi_t(\mathbf{x}_t, \mathbf{v}_t; \mathcal{M}_t)$, where $\mathcal{M}_t$ denotes any solver history. This covers the UniPC and DDIM updates used in our experiments. Evaluating f_θ dominates the cost. A step cache chooses evaluation steps $\mathcal{A} \subseteq \{0, \dots, N-1\}$ and supplies an inexpensive $\hat{\mathbf{v}}_t$ elsewhere. At each evaluation, f_θ acts on the current accelerated latent; cached outputs are not taken from a precomputed full-step trajectory. The schedule allocates network computations, while the reuse rule supplies outputs between them. The pretrained weights and sampler updates remain unchanged.

3.2 Frequency dynamics motivate selective reuse

Let G be a spatial low-pass and decompose each output into $\ell_t = G\mathbf{v}_t$ and $h_t = (I - G)\mathbf{v}_t$. We measure $\Delta\ell_t = \ell_t - \ell_{t-1}$ and $\Delta h_t = h_t - h_{t-1}$ along full-step trajectories, using Euclidean norms over all tensor entries. The low-band change declines early, while the high-band change remains larger later in sampling (Fig. 2). For each trajectory, the first crossing of $\rho_t = \|\Delta h_t\|/\|\Delta\ell_t\|$ above 1 defines its crossover. The pooled fraction is 0.22 ± 0.10 across the three backbones.

This observation suggests an asymmetric approximation. Early evaluations capture the rapidly changing coarse component; later, direct replay can leave the high-frequency component stale even when the low-frequency component changes less. We therefore retain the cached low component and use a recent high-band increment to correct the output. The crossing is a relative-change statistic, with variation across prompts and backbones. It motivates the design without guaranteeing stationary low frequencies or predictable high-frequency changes.

3.3 What to reuse: a high-band trend correction

Let $p < c < t$ denote the two most recent evaluated steps and a skipped step. VeloCache uses

$$\hat{\mathbf{v}}_t = \underbrace{\ell_c}_{\text{retain low component}} + \underbrace{h_c + \gamma(h_c - h_p)}_{\text{correct high component}} = \mathbf{v}_c + \gamma(I - G)(\mathbf{v}_c - \mathbf{v}_p). \tag{1}$$

The two terms serve different roles: ℓ_c supplies the cached coarse component, while $h_c - h_p$ provides a correction from the most recent true outputs. Setting $\gamma = 0$ recovers direct output replay; $\gamma = 1$, our shared default, adds one historical high-band increment. Two-state first-order corrections also appear at the feature level (Lv et al., 2025); applying one at the output and only to the high band needs no layer hooks or blockwise reconstruction.

Spatial split and cache updates. We use the separable binomial kernel $[1, 4, 6, 4, 1]/16$ along height and width, with reflect padding, independently for every frame and channel. Thus the split is inexpensive and sums exactly to the cached output. It does not filter along the video-time axis. The filter is not a projector: in general, $G^2 \neq G$, so the two components need not be orthogonal. Retaining ℓ_c means keeping that summand in Eq. (1); it does not assert $G\hat{\mathbf{v}}_t = G\mathbf{v}_c$ after filtering the corrected output again. Only true evaluations update $(\mathbf{v}_p, \mathbf{v}_c)$. A reused output is never fed back into this history, so consecutive skips use the same correction rather than recursively extrapolating predictions. Before two true outputs are available, the correction is zero.

The binomial kernel also makes the correction easy to interpret. Away from the reflected boundary, its one-dimensional frequency response is $\cos^4(\omega/2)$; the two-dimensional response is the product along height and width. Hence $(I - G)$ suppresses slowly varying spatial increments and leaves more of the fine-scale increment. It is a smooth weighting rather than a binary frequency partition. The filter has unit sum, so a spatially constant historical increment contributes no correction.

One skipped interval. In the default 50-step schedule, steps 10 and 14 are true evaluations and steps 15–18 are skipped. Each skipped step receives $\mathbf{v}_{14} + (I - G)(\mathbf{v}_{14} - \mathbf{v}_{10})$. Although this output is held fixed, the latent continues to change through the sampler's step-dependent update. At step 19 the network is evaluated on the current latent, replacing the cached pair by $(\mathbf{v}_{14}, \mathbf{v}_{19})$ for the next interval. This distinction between a fixed output approximation and an evolving sampling state is important: the method removes network calls, not sampler steps, and does not accumulate extra historical increments on successive skips.

When does the correction help? Condition on the cached pair and the exact output $\mathbf{v}_t = f_\theta(\mathbf{x}_t, \tau_t)$ at a fixed target state. Define $q = \mathbf{v}_t - \mathbf{v}_c$, $D = (I - G)(\mathbf{v}_c - \mathbf{v}_p)$, $E_0 = \|q\|^2$, and $E_\gamma = \|\gamma D - q\|^2$. Expanding the square gives

$$E_0 - E_\gamma = 2\gamma\langle q, D\rangle - \gamma^2\|D\|^2. \tag{2}$$

The first term rewards alignment with the target change; the second penalizes correction magnitude. For $D \neq 0$ and $\gamma > 0$, improvement requires $\gamma < 2\langle q, D\rangle/\|D\|^2$: an aligned correction can still be too large. The unconstrained local optimum is $\gamma_t^\star = \langle q, D\rangle/\|D\|^2$; the default $\gamma = 1$ helps precisely when $2\langle q, D\rangle > \|D\|^2$. This identity holds for our actual filter (Prop. 1, App. A). The unavailable target q makes this an analysis criterion, not an online test. If $\langle q, D\rangle \leq 0$, no positive gain improves this local error; if $D = 0$, the rule reduces to direct replay. These cases explain why change magnitude alone cannot certify a useful correction.

Eq. (2) holds for any fixed linear G, including $G = I$, so the identity alone does not argue for a frequency split. What the crossover adds is an empirical reason to expect an aligned and appropriately sized D for the high band and not for the whole signal, which §4.3 tests directly.

History and prediction intervals also matter. For an affine high-band trajectory in an increasing sampling coordinate u, exact extrapolation uses $(u_t - u_c)/(u_c - u_p)$. A fixed gain can therefore incur interval-mismatch error even without curvature. We use $\gamma = 1$ as an empirical default, validated by the gain ablation and fixed-schedule replay comparisons; the same rule can overshoot across longer gaps. Appendix A gives conditional local error bounds, not final-video LPIPS.

3.4 When to evaluate: protect the early trajectory

For N sampling steps, we set $H = \lceil c_0 N \rceil$ with the pooled protection fraction $c_0 = 0.22$. The user-specified budget K counts steps with full network computation, shared across guidance streams. For $H < K \leq N$, the evaluated steps are

$$\mathcal{A} = \{0, \ldots, H-1\} \cup \left\{ H - 1 + \mathrm{round}\big((N-H)\tfrac{j}{K-H}\big) \right\}_{j=1}^{K-H}. \tag{3}$$

The head evaluates the first H steps; the tail assigns the remaining $K - H$ evaluations, including the final step. Placement is uniform in step index, not necessarily in sampler time τ. For $(N, K) = (50, 20)$, this gives $\{0, \ldots, 10, 14, 19, 23, 27, 32, 36, 40, 45, 49\}$. Reducing K lengthens the gaps while preserving the head; $K = N$ recovers full computation. The schedule is fixed before generation, using a shared fraction checked on 12 development prompts (App. C.1), rather than measuring a new crossover for each generated clip. The shared fraction is a calibration choice: Cosmos's mean crossing is 10.7 of 35 steps while its head contains 8. The later crossing on H3 provides a further transfer test (App. D.3).

Algorithm 1 VeloCache: fixed-budget sampling with output reuse

Require: initial latent $\mathbf{x}_0$, N steps with times $\{\tau_t\}$, budget $H < K \leq N$, filter G, gain $\gamma = 1$

Build $\mathcal{A}$ by Eq. (3); initialize sampler history $\mathcal{M}_0$
For each guidance stream g, initialize $(\mathbf{v}_c^g, \mathbf{v}_p^g) \leftarrow (\varnothing, \varnothing)$
for $t = 0, \ldots, N-1$ **do**
  for all guidance streams g **do**
    if $t \in \mathcal{A}$ **or** $\mathbf{v}_c^g = \varnothing$ **then**
      $\mathbf{v}_t^g \leftarrow f_\theta^g(\mathbf{x}_t, \tau_t)$; $(\mathbf{v}_p^g, \mathbf{v}_c^g) \leftarrow (\mathbf{v}_c^g, \mathbf{v}_t^g)$
    else
      $D^g \leftarrow 0$ if $\mathbf{v}_p^g = \varnothing$, otherwise $(I - G)(\mathbf{v}_c^g - \mathbf{v}_p^g)$
      $\mathbf{v}_t^g \leftarrow \mathbf{v}_c^g + \gamma D^g$ ▷ do not update the caches
    end if
  end for
  Combine $\{\mathbf{v}_t^g\}$ by the original guidance rule to obtain $\mathbf{v}_t$
  $(\mathbf{x}_{t+1}, \mathcal{M}_{t+1}) \leftarrow \mathrm{Step}_t(\mathbf{x}_t, \mathbf{v}_t; \mathcal{M}_t)$
end for
return final latent $\mathbf{x}_N$

Integration and cost. In Algorithm 1, f_θ^g denotes the denoiser conditioned on guidance stream g; streams keep separate caches and may be batched as in CogVideoX. Step_t is the original sampler update Φ_t together with its history update. The sampler consumes the combined output at every step, including skips, so its time grid and history updates are preserved. For one sample with C channels, F frames and spatial size $H_x \times W_x$, filtering and pointwise work scale as $O(CFH_xW_x)$; persistent cache state is two such outputs per stream. With B bytes per element and S_g streams, persistent state is $2S_gBCFH_xW_x$ bytes, excluding temporary filtering workspace. It depends on latent size rather than network depth or hidden width. Applying the rule to other fixed evaluation patterns isolates its benefit from schedule selection (§4.3).

4 Experiments

4.1 Protocol

We use one recipe ($\gamma = 1$, identical band split and schedule rule) on three backbones with 100 shared VBench prompts: Cosmos3-Nano (NVIDIA, 2026a; Hugging Face, 2026; NVIDIA, 2026b)

Table 1: **Main comparison across three backbones** (100 shared prompts). VBench is no-reference; LPIPS uses full-step references. NFE is the rounded mean of full denoiser evaluations; cached paths remain timed. Wan/Cosmos use two guidance passes, CogVideoX one batched pass. Timing and adaptations: App. B; †output-level TaylorSeer on Cosmos. **Bold**/underline: best/second-best cache per metric and backbone, including displayed ties; Reference and NFE are not ranked.

Backbone	Method	NFE	Speedup↑	Imag.↑	Aesth.↑	Subj.↑	Bkgd.↑	Motion↑	Temp.↑	VB-Avg↑	LPIPS↓
Wan2.1	Reference	100	1.00×	0.682	0.569	0.956	0.957	0.988	0.980	0.8552	—
	MagCache	42	2.29×	0.674	0.569	0.957	**0.958**	**0.988**	**0.981**	0.8545	0.080
	EasyCache	40	**2.41**×	0.675	0.566	0.956	0.955	**0.988**	0.980	0.8533	0.084
	TeaCache	48	2.02×	0.673	0.566	0.957	0.957	**0.988**	**0.981**	0.8537	0.121
	TaylorSeer	26	2.40×	0.657	0.534	0.948	0.950	0.985	0.977	0.8420	0.349
	VeloCache	40	2.38×	**0.682**	**0.572**	**0.958**	**0.958**	**0.988**	0.980	**0.8563**	**0.052**
CogVideoX	Reference	50	1.00×	0.605	0.543	0.948	0.956	0.980	0.964	0.8328	—
	MagCache	20	2.45×	0.594	0.537	0.943	0.956	0.978	0.961	0.8281	0.110
	EasyCache	21	2.36×	0.585	0.530	0.941	0.955	0.978	0.962	0.8251	0.140
	TeaCache	23	2.17×	0.592	0.537	0.944	0.956	**0.979**	**0.963**	0.8285	0.106
	TaylorSeer	16	**2.46**×	0.583	0.520	0.934	0.951	**0.979**	0.962	0.8215	0.281
	VeloCache	20	2.44×	**0.598**	**0.546**	**0.948**	**0.957**	**0.979**	**0.963**	**0.8319**	**0.093**
Cosmos3	Reference	70	1.00×	0.727	0.564	0.958	0.965	0.994	0.985	0.8655	—
	MagCache	24	2.91×	0.602	0.502	0.955	0.963	**0.994**	**0.988**	0.8341	0.373
	EasyCache	27	2.62×	0.687	**0.548**	**0.958**	**0.965**	**0.994**	0.987	0.8565	**0.328**
	TeaCache	32	2.19×	0.599	0.490	0.949	0.962	**0.994**	**0.988**	0.8303	0.407
	TaylorSeer†	18	**3.88**×	0.700	0.526	0.946	0.957	0.993	0.984	0.8511	0.479
	VeloCache	24	2.91×	**0.707**	0.544	0.955	0.962	**0.994**	0.984	**0.8577**	0.333

(16B, 720 × 1280, 121 frames, 35 steps), Wan2.1-1.3B (480 × 832, 50 steps) and CogVideoX-5B (480 × 720, 50 steps). We report the mean of six selected VBench (Huang et al., 2024) dimensions, its MUSIQ-based (Ke et al., 2021) imaging term, and LPIPS (Zhang et al., 2018) to the same-seed full-step output. Perceptual quality and reference distortion measure different properties (Blau & Michaeli, 2018); our results therefore retain both. Methods share sampler and seeds. Wan scalar baselines port the released logic; CogVideoX and Cosmos use the adaptations in App. B.1. TaylorSeer is feature-level on Wan/CogVideoX and an output-level adaptation on Cosmos (†). Thresholds target nearby reported costs. H100 timing includes reconstruction work; measurement boundaries and limitations are in App. B. We report paired per-clip bootstrap 95% CIs; uncertainty and resampling protocols are detailed in App. B, together with all settings.

4.2 Main results

In Table 1, VeloCache has the highest mean VB-Avg and imaging among the tested caches on each backbone. LPIPS improvements over scalar baselines are 0.028–0.069 on Wan2.1 and 0.013–0.047 on CogVideoX at nearby reported costs; all paired 95% CIs exclude zero. On the 16B Cosmos backbone it leads imaging (0.707 vs. 0.700 for the best baseline) and has lower LPIPS than MagCache at a nearby reported cost (0.333 at 2.91× vs. 0.373 for MagCache at 2.91×); only EasyCache at 2.62×, about 11% longer runtime, is nominally lower, not significantly (paired $+0.006$ $[-0.003, +0.015]$, 49 of 100 clips). Against the best baseline chosen per clip, VeloCache has the lowest LPIPS on 95, 79 and 47 of 100 clips (App. F). Text–video differences to the reference are not statistically resolved on Wan2.1 and CogVideoX; the mean is 0.005 lower on Cosmos (App. D.1).

Reading the margins. VeloCache has the highest VB-Avg and imaging score on each backbone, but the VB-Avg margins over the strongest baseline (0.0012–0.0034) are comparable to the per-method standard errors (0.0023–0.0039, App. B) and four dimensions are near saturation (> 0.94). We read that ranking as evidence the correction does not cost perceptual quality rather than as a separation between caches. LPIPS separates methods more clearly (SE approximately 0.003–0.012 across the tested caches). VeloCache approaches MagCache's slower released default (0.052 at 2.4× versus 0.055 at 1.9×; paired -0.003 $[-0.005, -0.002]$), under the stated timing conventions.

Table 2: **Operating points near** 3× (100 prompts; settings in App. B.1). Cosmos TeaCache reaches 2.52× with imaging 0.36; Table 1 uses a lower threshold. **Bold**/underline: best/second-best cache.

	Wan2.1				CogVideoX				Cosmos3-Nano			
Method	Speed↑	Imag.↑	VB-Avg↑	LPIPS↓	Speed↑	Imag.↑	VB-Avg↑	LPIPS↓	Speed↑	Imag.↑	VB-Avg↑	LPIPS↓
Reference	1.00×	0.682	0.8552	—	1.00×	0.605	0.8328	—	1.00×	0.727	0.8655	—
MagCache	**3.08**×	0.667	0.8521	0.134	2.98×	0.582	0.8219	0.161	**3.18**×	0.586	0.8311	0.399
EasyCache	2.86×	0.670	0.8510	0.132	2.65×	0.567	0.8174	0.191	3.05×	0.646	0.8458	0.386
TeaCache	2.92×	0.656	0.8498	0.194	**3.02**×	0.568	0.8164	0.190	no setting at this speed			
VeloCache	2.90×	**0.681**	**0.8569**	**0.078**	3.00×	**0.584**	**0.8279**	**0.141**	**3.18**×	**0.677**	**0.8467**	**0.368**

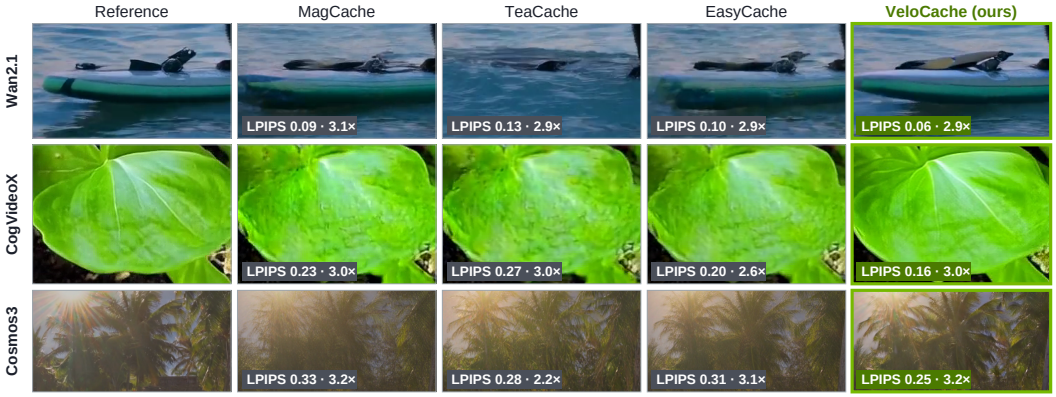

Figure 4: **Qualitative comparison near** 3× (Table 2; Cosmos TeaCache uses its Table 1 configuration). Matched-frame crops highlight the surfboard rim, leaf surface and palm fronds. Badges report whole-clip LPIPS and configuration-level speedup. Scenes and crops were selected for visual comparison (App. E); they illustrate differences, not prevalence. See App. F for failures.

At higher budget pressure. Table 2 increases budget pressure using each method's threshold or evaluation budget. On Wan2.1, VeloCache retains imaging 0.681 versus the full-step 0.682, at LPIPS 0.078; scalar caches give 0.132–0.194 (paired differences −0.054 to −0.116, all CIs excluding zero). On CogVideoX it improves over MagCache by 0.020 $[-0.025, -0.016]$ and TeaCache by 0.049 $[-0.056, -0.043]$. On Cosmos, it now separates from both MagCache (-0.031 $[-0.039, -0.023]$) and EasyCache (-0.018 $[-0.027, -0.009]$), with imaging gains of 0.03–0.09. Figure 4 illustrates the corresponding detail and structure changes. The wider sweep in Fig. 1b (32 prompts) also shows the limit: at $K = 13$, approximately 3.5× acceleration, VeloCache's LPIPS rises to 0.175 as the skipped intervals lengthen.

Against feature forecasting. At comparable reported costs, TaylorSeer's LPIPS is 0.349 versus 0.052 on Wan and 0.281 versus 0.093 on CogVideoX, with lower VB-Avg. We use its more favourable warm-up; the shorter default-like warm-up degrades CogVideoX to 0.554 (App. B.1). Reconstructing block features costs approximately 20% of a full evaluation in our Wan implementation. A faithful feature-level cache on the 16B Cosmos backbone would need about 33 GB by the shapes its blocks emit, which is why our Cosmos row forecasts the output instead (App. B.1). VeloCache uses only two outputs per stream. Wan2.1-14B at 720p and MiniMax-H3 test its scaling and transfer (Apps. D.2, D.3).

4.3 Ablations

Isolating the correction at fixed schedules. Spreading the same budget uniformly collapses fidelity (Table 4), and common baselines also protect an early segment, so early protection is a shared design principle. On our own anchors, LPIPS falls from 0.078 to 0.052 and imaging rises by 0.010 (Table 4, $\gamma = 0$). The same substitution on fixed MagCache-derived evaluation patterns improves all three reported metrics on every backbone (paired LPIPS -0.019, -0.003 and -0.039, all intervals

Table 3: **Removing the band split at fixed anchors** (LPIPS to the full-step output; paired bootstrap 95% CIs of the row minus VeloCache, so positive favours VeloCache). All rules share VeloCache's anchors, its two cached outputs and $\gamma = 1$ unless stated: *whole output* drops the split, *interval-scaled* sets $\gamma = \alpha_t$, *direct replay* sets $\gamma = 0$. Runs stopped at their wall-clock limits and cover a prefix of the prompt list; the Cosmos3-Nano subset mean sits far below its full-set 0.333 (Table 1), so that column covers the easier part of the benchmark. **Bold**/underline: best/second-best LPIPS.

	Wan2.1 (n=77)		CogVideoX (n=59)		Cosmos3-Nano (n=60)	
Reuse rule	LPIPS↓	Δ [CI]	LPIPS↓	Δ [CI]	LPIPS↓	Δ [CI]
whole output	0.0559	+.0056 [+.0042, +.0073]	0.0978	+.0064 [+.0034, +.0095]	0.2106	+.0260 [+.0198, +.0329]
direct replay	0.0752	+.0249 [+.0203, +.0302]	0.0969	+.0055 [+.0028, +.0083]	<u>0.1833</u>	−.0012 [−.0057, +.0033]
interval-scaled	<u>0.0532</u>	+.0030 [+.0020, +.0040]	**0.0878**	−.0036 [−.0067, −.0006]	**0.1822**	−.0023 [−.0049, +.0002]
VeloCache	**0.0503**	—	<u>0.0914</u>	—	0.1845	—

Table 4: **Design ablations** (Wan2.1, 100 prompts, 2.4× budget). Each column changes one default: $\gamma = 1$, first order, retain low/update high, crossover schedule, 5×5 filter. *Higher order*: three-output difference; *cosine*: earlier edge-anchored schedule. **Bold**/underline: best/second best; no variant beats the default on both metrics.

		Refresh gain γ			Trend	Bands	Schedule		Low-pass width		
	Default	0	0.5	1.5	Higher order	swapped	uniform	cosine	3×3	9×9	13×13
imaging↑	0.682	0.672	0.678	**0.685**	0.681	0.672	0.651	0.675	0.680	<u>0.683</u>	<u>0.683</u>
LPIPS↓	**0.052**	0.078	0.056	0.070	0.084	0.067	0.548	0.161	**0.052**	<u>0.053</u>	<u>0.053</u>

excluding zero; App. C), so the gain does not depend on our schedule. The closest published rule, FreqCa's second-order forecast of a frequency-split block-stack output (Liu et al., 2025b), ported to Wan2.1 and run on the same anchors reaches LPIPS 0.114 against our 0.052 (lower on 100 of 100 clips), and its fixed-interval configuration gives 0.078 at a reported 2.0× versus our 0.078 near 2.9× (Table 7). Representation, filter and predictor all differ, so cache location is not isolated (App. B.1).

Does the band split matter, or only the history? The comparison that isolates the frequency split holds the anchors, the gain and the two cached outputs fixed and removes only the split, predicting $\hat{\mathbf{v}}_t = \mathbf{v}_c + \gamma(\mathbf{v}_c - \mathbf{v}_p)$ on the whole output (Table 3). The band split is better on all three backbones, by 0.006 to 0.026 LPIPS with every paired interval excluding zero: a same-history whole-output trend does not recover the correction: the gain depends on which component is updated.

How the two ingredients divide that gain varies by backbone, and the interval-scaled gain $\alpha_t = b/a$ does not dominate the fixed one; both are examined in App. C.

Design ablations. Table 4 changes one choice at a time on Wan2.1. The high-band correction ($\gamma = 0 \to 1$) is the largest single effect; $\gamma = 1.5$ sharpens further but overshoots, the tested higher-order variant does not normalize unequal intervals, updating the low band instead is worse on both metrics, and uniform or cosine placement is far worse than the protected head. Filter width matters least. Two further schedule probes support the protected head (App. C.1).

5 Conclusion and Limitations

In this paper, we proposed VeloCache, a training-free caching method that retains the coarse component of the denoiser output and corrects its high band with a recent increment. Coupled with early trajectory protection, this gives consistent reference-fidelity gains over direct replay on fixed schedules, without storing internal features, and a matched whole-output control confirms that the band split carries the improvement. The same gain, filter and schedule rule transfer across three video backbones, with only two cached outputs per guidance stream and storage determined by latent size rather than network depth. These results identify frequency-selective output correction as a simple, training-free complement to step selection. The fixed gain can still overshoot across long gaps; broader motion evaluation and unified end-to-end timing remain future work.

Reproducibility Statement

The sampling algorithm and the reuse and scheduling rules are provided in Algorithm 1 and Eqs. (1) and (3). Appendix A presents the theoretical analysis, while Appendix B documents the experimental settings, baseline implementations, evaluation metrics, and timing procedures. We will release the code, prompt list, per-clip results, and figure-generation scripts.

AI Use Statement

We used generative AI tools to refine the language of this paper and to assist with identifying and summarizing related literature. Writing assistance focused on improving the phrasing, grammar, and clarity of text drafted by the authors. All AI-assisted content was reviewed and edited by the authors to ensure that it accurately reflects the work presented in this paper. The authors take full responsibility for the final manuscript, including its claims, results, and references.

References

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets. *arXiv preprint arXiv:2311.15127*, 2023. URL https://arxiv.org/abs/2311.15127.

Yochai Blau and Tomer Michaeli. The Perception-Distortion Tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6228–6237, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Blau_The_Perception-Distortion_Tradeoff_CVPR_2018_paper.html.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video Generation Models as World Simulators. OpenAI technical report, 2024. URL https://openai.com/index/video-generation-models-as-world-simulators/.

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception Prioritized Training of Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11472–11481, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Choi_Perception_Prioritized_Training_of_Diffusion_Models_CVPR_2022_paper.html.

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 (CVPR)*, pp. 14283–14294, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Chung_SeaCache_Spectral-Evolution-Aware_Cache_for_Accelerating_Diffusion_Models_CVPR_2026_paper.html.

DefTruth, vipshop.com, etc. Cache-DiT: A PyTorch-native Inference Engine with Cache, Parallelism, Quantization and CPU Offload for DiTs., 2025. URL https://github.com/vipshop/cache-dit.git. Open-source software; accessed September 26, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive Benchmark Suite for Video Generative Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Huang_VBench_Comprehensive_Benchmark_Suite_for_Video_Generative_Models_CVPR_2024_paper.html.

Hugging Face. Cosmos 3: Diffusers documentation. Software documentation, 2026. URL https://github.com/huggingface/diffusers/blob/main/docs/source/en/api/pipelines/cosmos3.md. Documentation snapshot accessed September 25, 2026.

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 (ICCV)*, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Kahatapitiya_Adaptive_Caching_for_Faster_Video_Generation_with_Diffusion_Transformers_ICCV_2025_paper.html.

Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. MUSIQ: Multi-Scale Image Quality Transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5148–5157, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Ke_MUSIQ_Multi-Scale_Image_Quality_Transformer_ICCV_2021_paper.html.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. HunyuanVideo: A Systematic Framework for Large Video Generative Models. *arXiv preprint arXiv:2412.03603*, 2024. URL https://arxiv.org/abs/2412.03603.

Guandong Li. Frequency-Aware Error-Bounded Caching for Accelerating Diffusion Transformers. *arXiv preprint arXiv:2603.05315*, 2026. URL https://arxiv.org/abs/2603.05315.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://arxiv.org/abs/2210.02747.

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 (CVPR)*, 2025a. URL https://openaccess.thecvf.com/content/CVPR2025/html/Liu_Timestep_Embedding_Tells_Its_Time_to_Cache_for_Video_Diffusion_CVPR_2025_paper.html.

Jiacheng Liu, Peiliang Cai, Qinming Zhou, Yuqi Lin, Deyang Kong, Benhao Huang, Yupei Pan, Haowen Xu, Chang Zou, Junshu Tang, Shikang Zheng, and Linfeng Zhang. FreqCa: Accelerating Diffusion Models via Frequency-Aware Caching. *arXiv preprint arXiv:2510.08669*, 2025b. URL https://arxiv.org/abs/2510.08669.

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 (ICCV)*, 2025c. URL https://openaccess.thecvf.com/content/ICCV2025/html/Liu_From_Reusing_to_Forecasting_Accelerating_Diffusion_Models_with_TaylorSeers_ICCV_2025_paper.html.

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 (NeurIPS)*, 2022. URL https://arxiv.org/abs/2206.00927.

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. URL https://arxiv.org/abs/2310.04378.

Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K. Wong. FasterCache: Training-Free Video Diffusion Model Acceleration with High Quality. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=W49UjcpGxx.

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 (NeurIPS)*, 2024a. URL https://openreview.net/forum?id=ZupoMzMNrO.

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 (CVPR)*, 2024b. URL https://openaccess.thecvf.com/content/CVPR2024/html/Ma_DeepCache_Accelerating_Diffusion_Models_for_Free_CVPR_2024_paper.html.

Zehong Ma, Longhui Wei, Feng Wang, Shiliang Zhang, and Qi Tian. MagCache: Fast Video Generation with Magnitude-Aware Cache. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. URL https://openreview.net/forum?id=KZn7TDOL4J.

MiniMax. MiniMax H3: Official model documentation. Model and software documentation, 2026. URL https://github.com/MiniMax-AI/MiniMax-H3. Documentation snapshot accessed September 25, 2026.

NVIDIA. Cosmos World Foundation Model Platform for Physical AI. *arXiv preprint arXiv:2501.03575*, 2025. URL https://arxiv.org/abs/2501.03575.

NVIDIA. Cosmos 3: Omnimodal World Models for Physical AI. *arXiv preprint arXiv:2606.02800*, 2026a. URL https://arxiv.org/abs/2606.02800.

NVIDIA. Cosmos3-Nano: Model card. Model card, 2026b. URL https://huggingface.co/nvidia/Cosmos3-Nano. Reports 16B trainable parameters; accessed September 26, 2026.

William Peebles and Saining Xie. Scalable Diffusion Models with Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.html.

Hao Phung, Quan Dao, and Anh Tran. Wavelet Diffusion Models Are Fast and Scalable Image Generators. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Phung_Wavelet_Diffusion_Models_Are_Fast_and_Scalable_Image_Generators_CVPR_2023_paper.html.

Severi Rissanen, Markus Heinonen, and Arno Solin. Generative Modelling with Inverse Heat Dissipation. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://arxiv.org/abs/2206.13397.

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 (CVPR)*, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Rombach_High-Resolution_Image_Synthesis_With_Latent_Diffusion_Models_CVPR_2022_paper.html.

Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://arxiv.org/abs/2202.00512.

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. URL https://arxiv.org/abs/2407.01425.

Chenyang Si, Ziqi Huang, Yuming Jiang, and Ziwei Liu. FreeU: Free Lunch in Diffusion U-Net. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Si_FreeU_Free_Lunch_in_Diffusion_U-Net_CVPR_2024_paper.html.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://arxiv.org/abs/2010.02502.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. In *International Conference on Machine Learning (ICML)*, 2023. URL https://proceedings.mlr.press/v202/song23a.html.

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. URL https://arxiv.org/abs/2503.20314.

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 (CVPR)*, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Wimbauer_Cache_Me_if_You_Can_Accelerating_Diffusion_Models_through_Block_CVPR_2024_paper.html.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. CogVideoX: Text-to-Video Diffusion Models with An Expert Transformer. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=LQzN6TRFg9.

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 (CVPR)*, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Yin_One-step_Diffusion_with_Distribution_Matching_Distillation_CVPR_2024_paper.html.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

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 (NeurIPS)*, 2023. URL https://arxiv.org/abs/2302.04867.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-Time Video Generation with Pyramid Attention Broadcast. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=hDBrQ4DApF.

Xin Zhou, Dingkang Liang, Kaijin Chen, Tianrui Feng, Xiwu Chen, Hongkai Lin, Yikang Ding, Feiyang Tan, Hengshuang Zhao, and Xiang Bai. Less is Enough: Training-Free Video Diffusion Acceleration via Runtime-Adaptive Caching. *arXiv preprint arXiv:2507.02860*, 2025. URL https://arxiv.org/abs/2507.02860.

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 (ICLR)*, 2025. URL https://openreview.net/forum?id=yYZbZGo4ei.

A LOCAL REUSE ERROR AND THE ROLE OF TIME INTERVALS

Condition on cached outputs at the two most recent evaluated steps $p < c$ and a target at skipped step $t > c$. With exact model output $\mathbf{v}_t$ at the target state and fixed linear spatial filter G, define

$$D = (I - G)(\mathbf{v}_c - \mathbf{v}_p), \qquad q = \mathbf{v}_t - \mathbf{v}_c, \qquad \hat{\mathbf{v}}_t = \mathbf{v}_c + \gamma D.$$

Choose the target on a recorded full-step trajectory or, for an oracle diagnostic, the accelerated trajectory; these diagnostics differ once their latent states diverge.

Proposition 1 (Conditional benefit of the cached correction)**.** *For any fixed linear filter G, including the binomial filter used in our implementation, define $E_\gamma = \|\hat{\mathbf{v}}_t - \mathbf{v}_t\|^2$ and $E_0 = \|\mathbf{v}_c - \mathbf{v}_t\|^2$. Then*

$$E_\gamma = \|q\|^2 - 2\gamma\langle q, D\rangle + \gamma^2\|D\|^2, \qquad E_0 - E_\gamma = 2\gamma\langle q, D\rangle - \gamma^2\|D\|^2. \tag{4}$$

If $D \neq 0$, the unconstrained minimizing gain and its residual are

$$\gamma_t^\star = \frac{\langle q, D\rangle}{\|D\|^2}, \qquad E_{\gamma_t^\star} = \|q\|^2 - \frac{\langle q, D\rangle^2}{\|D\|^2}. \tag{5}$$

For a nonnegative gain the optimum is $\max(0, \gamma_t^\star)$. At the default $\gamma = 1$, the correction strictly improves on replay precisely when $2\langle q, D\rangle > \|D\|^2$. For $D = 0$, all gains recover replay.

Proof. Since $\hat{\mathbf{v}}_t - \mathbf{v}_t = \gamma D - q$, expanding its squared norm gives Eq. (4). For $D \neq 0$ this is a strictly convex quadratic in γ. Setting its derivative to zero gives Eq. (5); substituting $\gamma = 1$ and comparing with E_0 gives the stated improvement condition. □

What the frequency split explains. For this paragraph, hypothetically set $G = P$ for an orthogonal projector P and $D = (I - P)(\mathbf{v}_c - \mathbf{v}_p)$. With $\ell_j = P\mathbf{v}_j$, $h_j = (I - P)\mathbf{v}_j$, $L = \ell_c - \ell_t$, and $q_h = h_t - h_c$, orthogonality gives

$$E_\gamma = \|L\|^2 + \|\gamma D - q_h\|^2, \qquad D = h_c - h_p.$$

If each skipped low-band increment is at most ε, then $\|L\| \leq \sum_{u=c+1}^{t} \|\ell_u - \ell_{u-1}\| \leq (t - c)\varepsilon$. This requires an absolute drift bound: the crossover inequality $\|\Delta h\| > \|\Delta \ell\|$ supplies neither that bound nor a forecast-alignment condition. The low-band term cancels when comparing to direct output replay on the same target.

Returning to the actual smooth filter G and $D = (I - G)(\mathbf{v}_c - \mathbf{v}_p)$, $L = G(\mathbf{v}_c - \mathbf{v}_t)$ and $R = (I - G)(\mathbf{v}_c - \mathbf{v}_t) + \gamma D$ instead give $E_\gamma = \|L\|^2 + \|R\|^2 + 2\langle L, R\rangle$. The cross term need not vanish for this non-orthogonal split; Eq. (4) accounts for it.

Unequal time intervals. Let u be an increasing coordinate along sampling, such as step index or an oriented sampler time, and define the high-band output $h(u_j) = (I - G)\mathbf{v}_j$ on the same chosen trajectory. Set $a = u_c - u_p > 0$, $b = u_t - u_c > 0$, $\alpha_t = b/a$, and $D = h(u_c) - h(u_p)$. If $h(u)$ is affine, the exact prediction is $h(u_c) + \alpha_t D$, whereas the fixed-gain rule has error $(\gamma - \alpha_t)D$. Thus $\gamma = 1$ is exact for an affine high band only when the prediction interval equals the history interval (or $D = 0$). More generally, if $\|h''(u)\| \leq M$ between u_p and u_t, Taylor expansion at u_c gives

$$\|h(u_t) - h(u_c) - \alpha_t D\| \leq \tfrac{M}{2} b(a + b),$$

and therefore

$$\|h(u_c) + \gamma D - h(u_t)\| \leq |\gamma - \alpha_t|\,\|D\| + \tfrac{M}{2} b(a + b). \tag{6}$$

Indeed, the forward remainder is bounded by $Mb^2/2$ and the backward remainder by $Ma^2/2$; scaling the latter by b/a and adding proves the bound. The first term in Eq. (6) cannot be absorbed into curvature error. For $h_j = j\,d$ and $(p, c, t) = (10, 14, 15)$, the default predicts $18d$ instead of $15d$.

Scope of the default. The implementation fixes $\gamma = 1$ and both cached outputs throughout each skipped interval, giving a constant trend correction rather than distance-scaled linear extrapolation. The target-dependent $\gamma_t^\star$ requires evaluating the skipped output and serves only for analysis. Gain and schedule ablations support this default empirically, without establishing optimality across backbones, budgets, target states, or perceptual metrics. These conditional identities do not bound final video LPIPS; relating local output errors to accumulated trajectory error additionally requires stability assumptions on the model and actual sampler.

B EXPERIMENTAL DETAILS

Backbones and sampling. Table 5 lists the sampling configurations. These runs use bfloat16 (the Wan VAE in float32), one H100 (80 GB), PyTorch 2.7.1 and Diffusers 0.38.0. H3 uses the separate setup in App. D.3. Prompt i uses seed $1234{+}i$ for the full-step reference and every cache, so LPIPS compares outputs generated from identical initial noise. Wan2.1 and Cosmos3-Nano use one fixed negative prompt each and evaluate classifier-free guidance as two forward passes per step, with one cache per guidance stream; CogVideoX uses no negative prompt and batches the two streams into one pass. Clips are written at 16 (Wan2.1), 8 (CogVideoX) and 24 (Cosmos3-Nano) frames per second.

Table 5: **Sampling configuration.** N: sampling steps; Forwards: network forwards of the full-step run; K: steps at which VeloCache evaluates the network (the remaining $N{-}K$ steps use the reuse rule). The 100 prompts are a fixed subset of the VBench prompt suite shared by all backbones and methods; the 14B study uses the first 32.

Backbone	Resolution	Frames	N	Sampler	Guidance	Forwards	K	Prompts
Wan2.1-1.3B	480×832	81	50	UniPC, flow shift 3	5.0	100	20	100
CogVideoX-5B	480×720	49	50	DDIM (v-prediction)	6.0	50	20	100
Cosmos3-Nano (16B)	720×1280	121	35	UniPC, flow shift 10	6.0	70	12	100
Wan2.1-14B	720×1280	81	50	UniPC, flow shift 5	5.0	100	20 / 16	32

Anchor sets. With $c_0 = 0.22$, Eq. (3) evaluates the first $H{=}11$ steps of a 50-step trajectory and the first 8 of a 35-step one. The anchor sets are $\{0, \ldots, 10, 14, 19, 23, 27, 32, 36, 40, 45, 49\}$ for $(N, K) = (50, 20)$ (Tab. 1 on Wan2.1 and CogVideoX, and Wan2.1-14B), $\{0, \ldots, 10, 18, 26, 33, 41, 49\}$ for $(50, 16)$ (Tab. 2 on Wan2.1 and CogVideoX, and the fast 14B setting), $\{0, \ldots, 7, 14, 21, 27, 34\}$ for $(35, 12)$ (Tab. 1 on Cosmos3-Nano) and $\{0, \ldots, 7, 16, 25, 34\}$ for $(35, 11)$ (Tab. 2). The final step is always evaluated.

These sets instantiate Algorithm 1 at each budget; guidance streams share evaluation steps but keep separate output caches.

Reuse rule. We use the spatial filter and correction in Sec. 3.3, storing only $\mathbf{v}_c$ and $\mathbf{v}_p$.

Metrics. LPIPS uses the AlexNet backbone on 8 uniformly spaced frames per clip against the backbone's own full-step output for the same prompt and seed. No-reference quality is the mean of six VBench dimensions (subject consistency, background consistency, motion smoothness, temporal flickering, aesthetic quality, imaging quality, the last rescaled to $[0, 1]$); text alignment is VBench overall consistency. Confidence intervals resample paired prompts with replacement. Main LPIPS comparisons use 20,000 resamples with seed 0 reset per comparison. The text-alignment script uses 5,000 resamples; the Wan14B script uses 10,000, each with an initially seeded random generator and the comparison order in its script. These procedures and per-clip values will accompany the release. For VeloCache, VB-Avg SEs range from 0.0023 to 0.0039 and imaging SEs from 0.0053 to 0.0144 across the three main backbones. Reported speedup is the reference runtime divided by the method runtime under the recorded timing conventions detailed below.

Timing boundaries. The tables retain the recorded runtime estimates of the original experiments. In the Wan and CogVideoX pilot/full scripts, method time brackets the pipeline call (text encoding, denoising and VAE decoding, excluding file export). Wan summaries reuse a fixed reference runtime; CogVideoX explicitly uses stored denoising times from earlier reference runs as its numerator. Thus the reported ratios must not be interpreted as uniformly measured end-to-end speedups. The Wan14B study uses callback intervals: it omits the first step and multiplies the mean remaining step duration by the step count, an estimate rather than a complete loop measurement; its ratios in Table 12 are reported as N/K, not the $(N{-}1)/(K{-}1)$ form used in earlier drafts, which overstated acceleration by about 3% because step 0 is always evaluated. Cosmos summaries use stored denoising-time fields for both reference and method. H3 timing brackets its denoising loop. NFE and paired quality are unaffected by any of this.

A common-boundary check. To test whether the main-table ratios survive one convention, we re-measured 12 prompts per backbone with a single definition on both sides: one CUDA-synchronized pipeline call covering text encoding, denoising and VAE decoding, excluding load, warm-up, file export and metric computation, with the reference generated in the same batch and a discarded warm-up call before every measurement. Wan2.1 gives 101.9 s for the reference against 43.0 s for VeloCache (2.37×, against 2.38× in Table 1) and CogVideoX 109.8 s against 45.5 s (2.41×, against 2.44×). Both agree with the reported values to about 1%, so the quoted accelerations are not an artefact of the mixed conventions above, though the check covers 12 of the 100 prompts and does not revisit the baseline rows. On Cosmos3-Nano, where the main table times the denoising loop, the same check gives 163.1 s against 56.1 s (2.91×, matching the reported value); its full-pipeline ratio is lower at 2.79× because VAE decoding is unaffected by caching and takes 16.9 s of the accelerated run, about a quarter of it, on a 720×1280, 121-frame output. The same runs bound the cost of the band split itself: replacing it with direct replay at identical anchors changes the pipeline time by 0.4% on Wan2.1 and 0.03% on CogVideoX, so the two low-passes and the pointwise update are not measurable at this resolution.

Crossover measurement. We record the velocity at every step of full-step runs (18 Wan2.1, 16 CogVideoX and 16 Cosmos3-Nano trajectories), split it with the same G, and compute $\|\Delta\ell_t\|$ and $\|\Delta h_t\|$ between consecutive steps; curves are normalized by the sum of the low- and high-band change norms at the first update. A trajectory's crossover is its first step satisfying $\|\Delta h_t\| > \|\Delta\ell_t\|$.

B.1 Baseline implementation details

One pipeline. All methods in Tab. 1 share the Diffusers pipeline, sampler and seeds (timing boundaries above). Wan2.1 scalar caches are line-by-line ports of their released Wan2.1 scripts: MagCache and TeaCache add cached block-stack hidden-state changes to the current embedded input; EasyCache caches the difference between network output and input and adds it to the current input. MagCache uses the released 1.3B magnitude curve and TeaCache the released polynomial. Released retention protects the first 20% of steps for MagCache, the first ten for EasyCache, and the first and last for TeaCache in its default mode. Our EasyCache port gives 0.084 LPIPS at approximately 2.4× reported acceleration on our prompts; comparison across different prompt sets is not a paired reproduction test. On CogVideoX and Cosmos3-Nano, output-level adaptations protect the first 20% of steps and implement MagCache's magnitude curve calibrated on one non-benchmark prompt, EasyCache's transformation-rate criterion, and TeaCache's accumulated relative-L_1 latent-input change. Thresholds match VeloCache's budget (Table 6); all methods evaluate the final step.

Table 6: **Settings of every method in Tab. 1.** τ: EasyCache's accumulated predicted-error threshold; δ: TeaCache's accumulated relative-L_1 threshold.

Method	Wan2.1-1.3B	CogVideoX-5B	Cosmos3-Nano
MagCache	released code path: released magnitude curve, first 20% of steps retained, error threshold 0.12, ≤ 4 consecutive skips (released default ≤ 2 gives 1.9×)	calibrated curve, first 10 steps retained, threshold 0.06, ≤ 4 consecutive skips (matched to 20 evaluations)	calibrated curve, first 7 steps retained, threshold 0.1, ≤ 8 consecutive skips (matched to 12 evaluations)
EasyCache	released code path: threshold 0.05 (default), first 10 steps retained, last step evaluated	τ=0.35, first 10 steps retained	τ=1.5, first 7 steps retained
TeaCache	released code path: released polynomial, threshold 0.1 (released 2× setting)	δ=0.15, first 10 steps retained	δ=0.4, skips allowed on steps 7–28 (larger thresholds are much worse)
TaylorSeer	feature level, order 1, refresh every 5th step, 4 warm-up steps	as on Wan2.1, with 8 warm-up steps	velocity level, order 1, refresh every 5th step
VeloCache	Eq. (3), K=20, γ=1	Eq. (3), K=20, γ=1	Eq. (3), K=12, γ=1

The $\sim 3\times$ operating points. Table 2 retains the protected warm-up with these thresholds and budgets: on Wan2.1 MagCache 0.6 with ≤ 8 consecutive skips, EasyCache 0.08, TeaCache 0.2

and VeloCache K=16; on CogVideoX MagCache calibrated to 16 evaluations, EasyCache τ=0.55, TeaCache δ=0.8 and VeloCache K=16; on Cosmos3-Nano MagCache calibrated to 10 evaluations, EasyCache τ=3.0, TeaCache δ=1.5 and VeloCache K=11. Cosmos3-Nano TeaCache cannot reach 3×: its skip window 7–28 protects 13 of 35 steps. Near this limit, $\delta = 1.5$ reaches 2.52× but collapses (imaging 0.361 against reference 0.727, LPIPS 0.517); Tab. 2 therefore leaves the entry empty, while Tab. 1 reports its working setting. CogVideoX TeaCache requires the largest threshold increase: δ=0.25 to 0.6 give 2.43× to 2.88× at LPIPS 0.126 to 0.168, all slower than VeloCache's 3.00×; δ=0.8 first reaches 3×.

TeaCache on CogVideoX. Our port of the released CogVideoX-5B path applies its polynomial to the first block's timestep-modulated input and reuses hidden- and encoder-state residuals. It underperforms the harness at both tested thresholds: the released 0.1 evaluates 33 of 50 steps (1.52×), LPIPS 0.128; 0.2 reaches 2.33×, LPIPS 0.203 versus the harness's 0.118 at matched speed. Released coefficients were fitted to a different sampler configuration. We report the stronger harness variant, which accumulates relative-L_1 changes in the latent input with the same protected warm-up.

FreqCa port. FreqCa (Liu et al., 2025b) (arXiv:2510.08669v1) evaluates image generation and editing. Our Wan2.1 port retains its cumulative block-stack feature, reuses its low band and predicts its high band with a second-order Hermite-basis fit. The source describes a K-sample fit and three high-band states in its memory analysis, versus two activated steps in its Fig. 3 caption; our three computed samples yield an exact quadratic at distinct times. Port choices (not all specified in FreqCa's App. B.3) include the isotropic spatial FFT mask, cutoff, sampler-timestep abscissa, warm-up and recomputation of the current Wan output head. Recomputing the head from a fully evaluated feature reproduces the model output to numerical precision in our verification. With no source cutoff, we swept $\{0.13, 0.25, 0.5, 0.75\}$ of the Nyquist frequency on development prompts and report the best (0.5); varying prediction order (0–2) and abscissa (step index or timestep) changes the result by less than 0.01 there. Table 7 reports two schedules: our anchors fix evaluation positions but change representation, filter and predictor jointly; FreqCa's fixed interval uses the same 20% protected warm-up as Wan MagCache and EasyCache. Without warm-up, skipping begins immediately and LPIPS exceeds 0.4 at every tested cutoff and order, consistent with Table 4's uniform-schedule failure.

Table 7: **FreqCa's rule ported to Wan2.1** (100 prompts, LPIPS to the full-step output). Our port of a rule published for image models; the filter, cutoff, time variable, warm-up and output-head handling are our choices (above), so this compares implementations rather than isolating cache location. **Bold**/underline: best/second-best speedup and LPIPS.

Configuration	Evals	Speedup↑	LPIPS↓
FreqCa rule, our anchors (Eq. (3), K=20)	20	<u>2.37×</u>	0.114
FreqCa rule, native interval S=3, 20% warm-up	24	2.00×	<u>0.078</u>
FreqCa rule, native interval S=4, 20% warm-up	21	2.29×	0.097
VeloCache (K=20)	20	**2.38×**	**0.052**

TaylorSeer. On Wan and CogVideoX, our feature-level port follows TaylorSeer's official formulation, caching each block's self-/cross-attention and feed-forward outputs for first-order Taylor forecasting. An all-full-step run matches the reference to numerical precision; this checks the full evaluation path, not the accuracy of skipped-step predictions. Quality improves with the compute budgets we tested. Refreshes occur every fifth step after protected warm-ups chosen for similar reported cost: four steps on Wan (2.40× vs. 2.38×), eight on CogVideoX (2.46× vs. 2.44×), under the timing conventions above. Four warm-up steps (near the upstream default of three) cause CogVideoX divergence: 100 prompts yield LPIPS 0.554, five clips above 0.8 collapsing to colour fields, and VB-Avg 0.784. Stable zeroth-order feature reuse on the same schedule implicates first-order extrapolation across steep early feature changes, amplified by guidance; second-order forecasting exhausts an 80 GB GPU. We report the more favourable eight-step warm-up. The authors' model-agnostic Diffusers hook attaches to CogVideoX but never enters its prediction branch in the version we tested (runtime and outputs equal the full-step run), so we report our block-level port. Cosmos's mixture-of-transformers changes per-block feature shapes across denoising steps, making

faithful feature-level caching impractical; Tab. 1's † entry therefore adapts forecasting to the velocity output.

C Additional ablations

C.1 Choosing the schedule

Eq. (3) uses the pooled protection fraction $c_0 = 0.22$. We fixed its form on a development set of 12 Wan2.1 prompts that are not part of the benchmark (seeds $4000{+}i$), holding the budget and reuse rule fixed (Table 8). Protecting the head is what matters: the edge-anchored cosine schedule used by our earlier drafts, which evaluates only 7 of the first 12 steps, is worse by 0.09 LPIPS at $K = 20$ and worse than the 10-step tapered-head configuration by 0.12 at $K = 16$; an 8-step head (below the measured crossover) is worse by 0.03; once the head is covered, the placement of the remaining evaluations matters little (uniform, a sine-spaced tail that densifies toward the end, or a 10-step head with a tapered tail are within 0.01 of each other), and the uniform tail is the simplest. On the 10-step-head tapered schedule, scalar reuse is worse by 0.02 on the development set. The benchmark repeats this comparison at the default anchors (§4.3).

Table 8: **Schedule selection on the development set** (12 prompts outside the benchmark, Wan2.1, LPIPS to the full-step output, two-band reuse unless stated). Early protection helps the tested schedules; this small comparison does not establish an optimal head or tail placement. **Bold**/underline: lowest/second-lowest LPIPS within each budget.

Schedule (N=50)	Anchors at $K = 20$	K=20	K=16
Crossover rule (Eq. (3), default)	11-step head, uniform tail	**0.053**	—
11-step head, sine-spaced tail	$\{0..10, 17, 23, 30, 35, 40, 44, 47, 48, 49\}$	0.062	—
10-step head, tapered tail	$\{0..9, 13, 18, 23, 28, 33, 38, 42, 45, 47, 49\}$	<u>0.058</u>	**0.075**
8-step head, sine-spaced tail	$\{0..7, 12, 18, 23, 28, 33, 37, 40, 43, 46, 47, 48, 49\}$	0.084	<u>0.095</u>
Edge-anchored cosine (earlier drafts)	$\{0, 1, 2, 3, 5, 8, 11, 15, 18, 22, 27, 31, 34, 38, 41, 44, 46, 47, 48, 49\}$	0.143	0.196
10-step head, tapered tail, scalar reuse	as above	0.077	—

Early versus middle evaluations. We test the six Cosmos clips with the largest LPIPS deficit under the earlier cosine schedule (K=11, anchors $\{0, 1, 3, 7, 12, 17, 22, 27, 31, 33, 34\}$) and add four evaluations either at skipped early steps ($\{0, \ldots, 7, 12, 17, 22, 27, 31, 33, 34\}$), or in the middle of the trajectory (steps 10, 15, 20 and 25). Early evaluations cut mean LPIPS from 0.509 to 0.300; middle evaluations leave it at 0.507 (Table 9). Early repair beats the best baseline on five clips and is within 0.003 on the sixth. Baseline minima use earlier development configurations of MagCache, EasyCache and TeaCache, rather than the main-table settings. These selected cases support early protection, but do not establish that every pre-crossover evaluation is necessary.

Table 9: **Where the missing evaluations matter** (Cosmos3-Nano, LPIPS to the full-step output on the six selected clips; earlier cosine schedule and baseline configurations). Four extra evaluations reduce error when allocated early, but not in the middle. **Bold**/underline: lowest/second-lowest LPIPS per column.

Schedule (two-band reuse)	p72	p45	p92	p40	p57	p52	Mean
Earlier cosine schedule, $K = 11$	0.603	0.495	0.506	0.489	0.448	0.513	0.509
$+4$ evaluations mid-trajectory, K=15	0.624	0.490	0.453	0.504	0.456	0.515	0.507
$+4$ evaluations before the crossover, K=15	**0.382**	**0.237**	**0.270**	<u>0.359</u>	**0.274**	**0.280**	**0.300**
Best earlier baseline on the clip	<u>0.402</u>	<u>0.324</u>	<u>0.364</u>	**0.356**	<u>0.317</u>	<u>0.383</u>	<u>0.358</u>

C.2 Higher-order correction

The variant in Table 4 uses the three most recent true high-band outputs: $\hat{\mathbf{v}} = \ell_c + h_c + \gamma(2h_c - 3h_p + h_{pp})$. This is a higher-order difference over cached evaluation indices, with no normalization by history interval or prediction distance. Its result should not be generalized to all second-order

predictors on nonuniform sampling grids. The FreqCa port instead uses the sampler timestamps in a polynomial fit (App. B.1).

C.3 Separating the band split from history

Table 3 isolates the band split at fixed anchors (§4.3); read against its direct-replay row, it also shows that the two ingredients contribute unequally and differently per backbone. On Wan2.1 the whole-output trend supplies most of the gain over replay (0.019 of 0.025) and the split adds the rest. On CogVideoX the trend alone is no better than replay and the split supplies all of it. On Cosmos3-Nano the split only repairs a trend that is harmful on its own and neither improves on replay, which matches that backbone's later measured crossover (§3.4) and the overshoot cases in App. F. Scaling the gain by the interval ratio $\alpha_t = b/a$, the exact choice for an affine high band (App. A), beats $\gamma = 1$ on CogVideoX, loses on Wan2.1 and is not separated on Cosmos3-Nano, so the fixed gain is a shared default rather than a per-backbone optimum. All controls share the default anchors and output history.

C.4 Reuse on baseline evaluation patterns

Table 10 repeats the substitution of §4.3 on evaluation patterns produced by MagCache's released rule rather than by Eq. (3), at equal NFE. Both rows cache complete model outputs, so the *scalar* row is direct output replay on those anchors and not the native MagCache residual/scaling rule; its numbers therefore need not match the MagCache rows of Table 1. Two-band reuse improves all three reported metrics on every backbone.

Table 10: **Reuse rule swapped on MagCache-derived skip patterns** (iso-NFE, 100 prompts). Two-band reuse improves all three reported metrics (paired LPIPS -0.019 $[-0.021, -0.016]$, -0.003 $[-0.005, -0.001]$, -0.039 $[-0.046, -0.031]$). **Bold**/underline: best/second best within each backbone.

Backbone	Skip pattern (released rule, released warm-up)	Reuse	VB-Avg↑	imaging↑	LPIPS↓
Wan2.1	MagCache, released curve, threshold 0.12, ≤ 4 skips, 20% retained	scalar	0.8545	0.672	0.080
		two-band	**0.8563**	**0.681**	**0.061**
CogVideoX	MagCache, calibrated curve, threshold 0.06, ≤ 4 skips, 20% retained	scalar	0.8280	0.591	0.110
		two-band	**0.8313**	**0.596**	**0.106**
Cosmos3	MagCache, calibrated curve, threshold 0.1, ≤ 8 skips, 20% retained	scalar	0.8352	0.607	0.372
		two-band	**0.8503**	**0.684**	**0.333**

D Additional results

D.1 Text alignment

Neither the six VBench dimensions of Table 1 nor LPIPS measures whether an accelerated clip still follows its prompt. Table 11 adds VBench's *overall consistency* score (ViCLIP video–text similarity (Huang et al., 2024)) for every method on the same $N{=}100$ prompts, with a paired bootstrap 95% CI of the per-clip difference to each backbone's full-step reference. On Wan2.1 and CogVideoX VeloCache and the scalar-cache implementations have confidence intervals including zero relative to the reference, and only TaylorSeer loses alignment on Wan2.1. On Cosmos every cache is below the reference; VeloCache loses 0.005, less than MagCache, TeaCache and TaylorSeer; only EasyCache's interval includes zero. The text-alignment comparison does not show uniform dominance over scalar reuse on the same anchors.

D.2 Scaling to Wan2.1-14B at 720p

The main comparison uses the 1.3B Wan backbone at $480{\times}832$. To check that the recipe survives a $10\times$ larger denoiser and a $2.3\times$ larger latent, we run it unchanged on Wan2.1-14B at $720{\times}1280$, 81 frames, 50 steps (UniPC, shift 5), on the first 32 of the shared prompts, against EasyCache with the logic and default threshold of its released Wan2.1 script (Table 12). At the same evaluation budget (40 forwards, $2.50\times$ NFE ratio) VeloCache has lower LPIPS (0.091 vs. 0.110, paired -0.019

Table 11: **Text alignment** (VBench overall consistency, N=100; higher is better). Δ is the mean per-clip score of the row minus the backbone's full-step reference score with a paired bootstrap 95% CI; baselines at the settings of Table 1. **Bold**/underline: highest/second-highest cache score per backbone, including ties at displayed precision; Reference and Δ are not ranked.

	Wan2.1		CogVideoX		Cosmos3	
Method	score↑	Δ vs. ref [CI]	score↑	Δ vs. ref [CI]	score↑	Δ vs. ref [CI]
Reference	0.224	—	0.227	—	0.218	—
MagCache	0.223	-0.001 $[-.003, +.000]$	<u>0.226</u>	-0.001 $[-.003, +.002]$	0.206	-0.011 $[-.018, -.005]$
EasyCache	0.223	-0.001 $[-.003, +.001]$	**0.227**	-0.000 $[-.002, +.002]$	**0.215**	-0.003 $[-.006, +.001]$
TeaCache	<u>0.224</u>	-0.001 $[-.003, +.001]$	**0.227**	-0.000 $[-.002, +.002]$	0.204	-0.013 $[-.021, -.005]$
TaylorSeer	0.209	-0.016 $[-.022, -.009]$	0.223	-0.004 $[-.009, +.000]$	0.210	-0.008 $[-.016, -.001]$
VeloCache	**0.225**	$+0.000$ $[-.001, +.001]$	0.225	-0.002 $[-.004, +.000]$	<u>0.213</u>	-0.005 $[-.010, -.001]$
scalar reuse, same anchors	<u>0.224</u>	-0.001 $[-.002, +.001]$	**0.227**	-0.000 $[-.002, +.002]$	0.211	-0.006 $[-.011, -.002]$

$[-0.030, -0.010]$) and has VB-Avg and imaging means close to the full-step values; at K=16 its NFE ratio is 3.13× against EasyCache's 2.50× for +0.013 LPIPS, with VB-Avg 0.868 versus the reference 0.869. The schedule, band split and γ=1 transfer unchanged from the 1.3B setting, with two outputs cached per guidance stream. The reference denoising time is an estimated 1875 s on one H100; callback timing and the reported N/K ratios are defined in App. B.

Table 12: **Wan2.1-14B,** 720×1280**,** 81 **frames** (32 prompts; LPIPS against the 14B full-step reference). NFE ratio is reference NFE divided by method NFE (App. B). **Bold**/underline: best/second-best cache per metric; Reference and NFE are not ranked.

Method	NFE	NFE ratio↑	Imag.↑	Aesth.↑	VB-Avg↑	LPIPS↓
Reference (50 steps)	100	1.00×	0.696	0.617	0.869	—
EasyCache (released, thresh. 0.05)	40	<u>2.50</u>×	0.690	0.616	0.867	<u>0.110</u>
VeloCache (K=20)	40	<u>2.50</u>×	**0.696**	**0.622**	**0.869**	**0.091**
VeloCache (K=16)	32	**3.13**×	<u>0.691</u>	<u>0.617</u>	<u>0.868</u>	0.123

D.3 TRANSFER TO MINIMAX-H3

MiniMax-H3 (MiniMax, 2026) is a 33B omni transformer that denoises the video and audio rows of one packed sequence in a single forward pass per step; the released checkpoint is guidance-distilled, so there is no classifier-free guidance. Released after the main experiments, it provides a separate transfer test. We use its Diffusers text-to-video-and-audio workflow on two H100s (text encoder on one, the rest on the other) at 960×544, 124 frames at 24 fps, a 50-point sigma grid (49 evaluations), and the same 100 prompts and seeds as the main tables. The video rows are unpacked to their (C, F, H, W) latent grid with the model's own patch layout, the two-band rule is applied there and the result re-packed; the audio rows are one-dimensional and are advanced by the same first-order trend without a band split. The three scalar caches are our implementations of the published rules at the output level with a ten-step (20%) protected warm-up; their thresholds were set on two prompts to give ∼20 evaluations (EasyCache τ=0.5, TeaCache δ=0.2, MagCache calibrated to 20 on one prompt outside the benchmark). Table 13 compares nearby reported speeds; LPIPS evaluates the video track only.

Later crossover and two evaluated schedules. Twelve full-step H3 development trajectories, generated for this backbone and unrelated to the Wan2.1 development set of App. C.1, give a first crossing at 25 ± 6 of 49 steps, $c = 0.52 \pm 0.12$, compared with the pooled 0.22 ± 0.10 on the three main backbones. This calibration statistic would give $H = 26$ in Eq. (3); it is not a necessary budget or a per-trajectory guarantee. The unchanged recipe uses $c_0 = 0.22$, $K = 20$, and $\{0, \ldots, 10, 14, 18, 23, 27, 31, 35, 40, 44, 48\}$. Its sparse segment starts before the mean crossing. The $K = 35$ experiment used the earlier four-trajectory estimate $c_0 = 0.61$, retaining $H = 30$ and anchors $\{0, \ldots, 29, 33, 37, 40, 44, 48\}$. It was retained after the estimate changed to 0.52; these results do not evaluate $H = 26$ or establish an optimal budget.

Table 13: **MiniMax-H3** (960×544, 124 frames; LPIPS on the video track against the full-step output; denoising-loop speedup). Paired bootstrap 95% CIs use $\Delta = \mathrm{LPIPS}_{\mathrm{VeloCache}} - \mathrm{LPIPS}_{\mathrm{row}}$; negative favours VeloCache. Upper block: 100 prompts, pooled-crossover anchors for VeloCache and its scalar control. Lower block: 32 prompts, 30-step head. **Bold**/underline: best/second-best cache per metric and block, including displayed ties; Reference, Evals and Δ are not ranked.

	Method	Evals	Speedup↑	Imag.↑	VB-Avg↑	LPIPS↓	paired Δ $\mathrm{LPIPS}_{\mathrm{VeloCache}} - \mathrm{LPIPS}_{\mathrm{row}}$
	Reference	49	1.00×	0.6636	0.8522	—	—
pooled c=0.22 K=20	MagCache	20	2.27×	0.6617	0.8525	0.1185	-0.004 $[-0.009, +0.001]$
	EasyCache	19.6	2.27×	0.6600	**0.8528**	0.1302	-0.015 $[-0.023, -0.009]$
	TeaCache	19	**2.28**×	0.6597	0.8510	0.1832	-0.068 $[-0.076, -0.061]$
	scalar reuse, same anchors	20	**2.28**×	0.6566	0.8519	0.1260	-0.011 $[-0.014, -0.008]$
	VeloCache	20	2.27×	**0.6631**	0.8522	**0.1148**	—
earlier c_0=0.61 H=30, K=35	scalar reuse, same anchors	35	1.31×	—	—	0.052	-0.016 $[-0.020, -0.013]$
	VeloCache	35	**1.32**×	—	—	**0.036**	—

Results and scope. With 20 of 49 evaluations, VeloCache runs at $2.27\times$ against a $2.45\times$ evaluation ratio, and the band split on a $24\times37\times34\times60$ latent costs nothing measurable. The combined video/audio reuse strategy improves on its scalar control at both budgets: on the same anchors it is lower in LPIPS than scalar reuse on 81 of 100 clips at K=20 (-0.011 $[-0.014, -0.008]$) and on 32 of 32 at K=35 (-0.016 $[-0.020, -0.013]$, a 31% reduction), and its imaging mean is close to the full-step mean (0.6631 vs. 0.6636, with the baselines at 0.6566–0.6617). No-reference quality otherwise does not separate the caches at this budget: VB-Avg spans 0.0018 across all six rows, within its standard error. The difference to MagCache is -0.004 $[-0.009, +0.001]$, lower on 56 of 100 clips, so the two are not statistically separated. The $K = 35$ study changes both budget and sample size; it cannot isolate the crossover's effect. The scalar controls freeze video and audio while the default also corrects audio, so neither comparison isolates the video-band split. A direct diagnostic would measure target increments and their alignment with the cached correction on the actual skipped intervals (Prop. 1).

On development prompts, swapping which band is updated gives LPIPS 0.139 versus 0.115 for the default; updating both bands on pre-crossover steps gives 0.125; concentrating the $K = 20$ budget at the start gives approximately 0.30; freezing audio gives 0.117. These limited probes did not improve the default or rule out other schedules and gains. Audio quality remains unevaluated.

E ADDITIONAL QUALITATIVE RESULTS

Figure 5 adds Wan2.1-14B examples with released-setting EasyCache. On 12 further prompts outside both the benchmark and the development set of App. C.1, mean LPIPS is 0.102 for VeloCache and 0.132 for EasyCache, both at 40 forwards (2.50× NFE ratio). Benchmark means appear in Table 12.

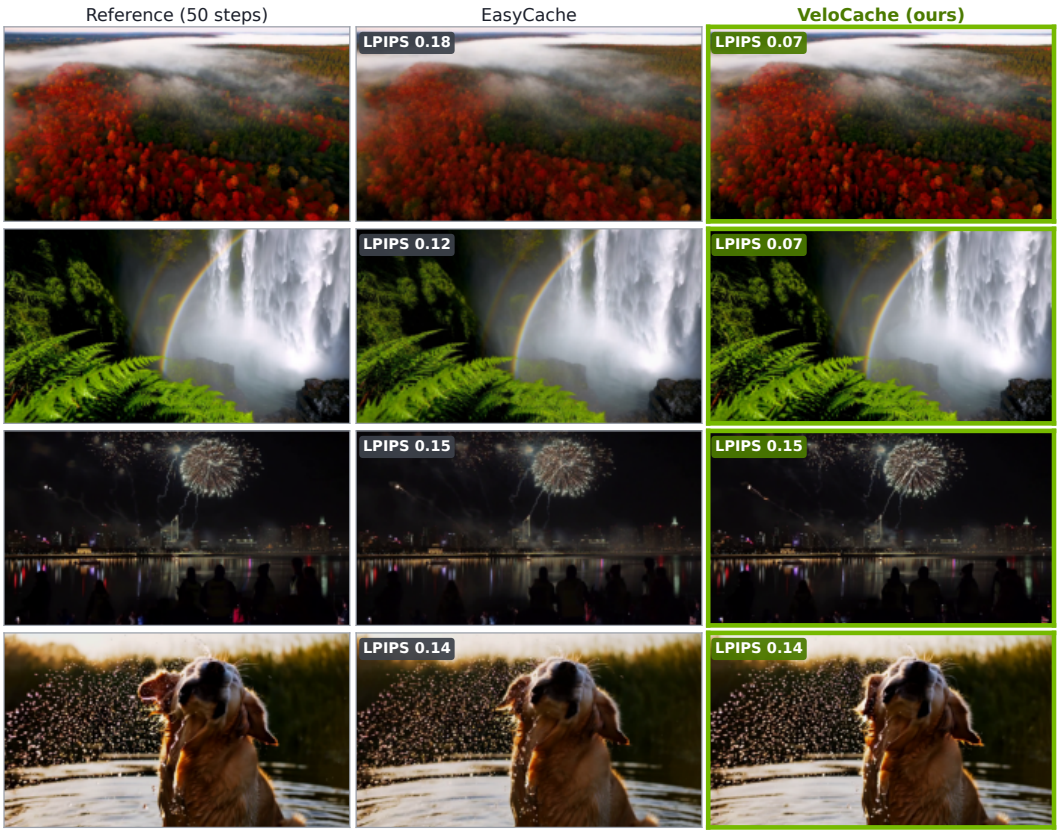

Figure 5: **Wan2.1-14B at** 720×1280 (mid-frames; per-clip LPIPS to the full-step output; EasyCache at its released setting). Both caches use 40 forwards, a 2.50× NFE ratio (Table 12); badges report LPIPS only.

Selection and interpretation. Figure 4 shows Wan2.1 p85, CogVideoX p46 and Cosmos3-Nano p25 at zero-based frames 40, 24 and 60, using identical frame indices and crop coordinates across methods. Candidates had VeloCache whole-clip LPIPS at most 0.20 on Wan/CogVideoX or 0.45 on Cosmos. Mid-frames were split into a 6×8 grid and scored by the largest patchwise margin between the best baseline's mean absolute reference error and VeloCache's. Final crops were chosen after inspecting multiple video times; they are neither random nor necessarily top-ranked. LPIPS badges measure whole clips, not crops. The supplementary video adds six selected benchmark cases at Table 2 settings; App. F instead selects the largest LPIPS deficits. These examples illustrate differences, not prevalence or lossless reconstruction.

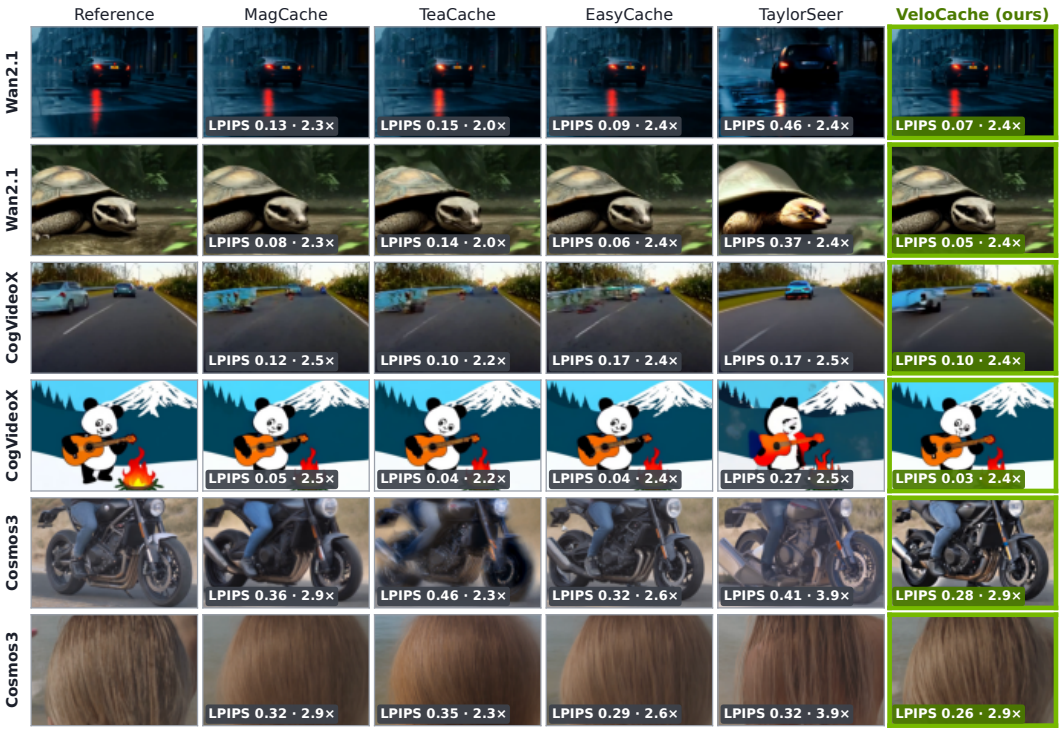

Figure 6: **Additional comparisons**: two scenes per backbone (mid-frame crops; whole-clip LPIPS and configuration-level speedup). Settings follow Table 6, except Cosmos TeaCache ($\delta = 0.6$ instead of 0.4). VeloCache has lower LPIPS in five rows; the first CogVideoX row is nearly tied.

F Failure cases

VeloCache has the lowest LPIPS among five caches on $95/100$ Wan2.1 clips (maximum deficit 0.03), $79/100$ CogVideoX clips (0.02), and $47/100$ Cosmos clips. On Cosmos it beats MagCache, TeaCache, TaylorSeer and the slower EasyCache on 83, 100, 99 and 49 clips, respectively. Figure 7 selects the largest deficits: CogVideoX frames look similar; Cosmos shows overshoot at a static stop-sign edge and drift in the corgi background and fireworks/tree structure, despite competitive no-reference quality (Table 1).

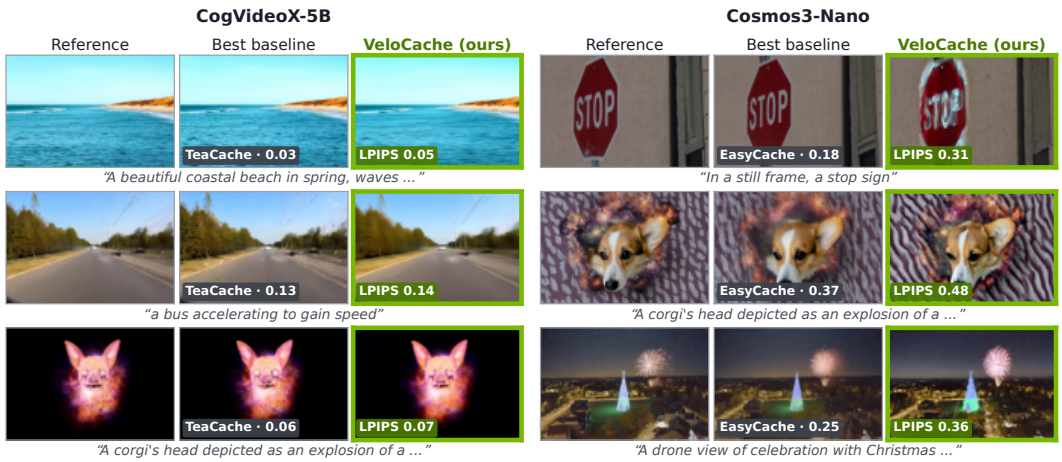

Figure 7: **Failure cases**: three clips per backbone with the largest LPIPS deficit to the best baseline. Mid-frames are shown; badges report whole-clip LPIPS to the full-step output.