FINE-GRAINED CACHING FOR DIFFUSION TRANSFORMERS WITH FEW CALIBRATION CONDITIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion transformers require repeated denoiser evaluations, making image and video generation computationally expensive. We propose a training-free framework that uses a few calibration conditions to construct a fixed, fine-grained module-reuse schedule without schedule search. The design is motivated by an empirical effect along deterministic sampling trajectories: with initial noise fixed, condition-dependent deviations in several module outputs evolve similarly across adjacent steps, so temporal differencing attenuates much of their variation. We term this effect Conditional Common-Mode Rejection (CCMR). Motivated by this observation, we rank timestep–layer–module locations using normalized ratios of consecutive feature displacements and construct a Cache Book through two-stage calibration. The second stage updates a scorer-only reference with each provisional bit after recording its indexed score, while retaining full-compute trajectories. At inference, the fixed schedule requires no per-input policy estimation and can be nested within fixed step-wise cache policies. Module caching alone achieves 1.66–$1.72\times$ speedup on the evaluated image models and 1.54–$1.68\times$ on the video models. Combining it with MagCache or SeaCache provides further acceleration at a model-dependent cost in paired fidelity.

1 INTRODUCTION

Diffusion models have become central to image and video synthesis (Ho et al., 2020; Rombach et al., 2022; Ho et al., 2022; Song et al., 2021b; Kong et al., 2024), and diffusion transformers (DiTs) scale with model size and data (Peebles & Xie, 2023; Esser et al., 2024). Sampling remains expensive because it repeatedly evaluates the denoiser, particularly over long video token sequences. Acceleration methods reduce sampling steps through solvers or distillation (Song et al., 2021a; Lu et al., 2022; 2025; Zhao et al., 2023; Salimans & Ho, 2022; Yin et al., 2024b;a; Song et al., 2023), or lower per-step cost through sparse attention and model compression (Xi et al., 2025; Wu et al., 2024). Feature caching instead reuses intermediate computations without changing the pretrained weights or sampling steps (Ma et al., 2024b; Lyu et al., 2025; Liu et al., 2025b). Its central challenge is choosing what to reuse: aggressive reuse saves computation but can accumulate error, while temporal redundancy varies across steps, layers, and modules.

Many high-performing caching methods use a shared decision to reuse intermediate features across denoising steps. TeaCache uses variation in timestep-modulated inputs to decide when to reuse a transformer residual (Liu et al., 2025a). MagCache schedules residual reuse from magnitude variation, while SeaCache evaluates reuse through spectrally filtered feature changes (Ma et al., 2025; Chung et al., 2026). Some methods examine redundancy at the block level while retaining a shared step-wise reuse decision. AdaCache stores block residuals under a schedule synchronized across transformer blocks (Kahatapitiya et al., 2025), while BWCache aggregates block-wise changes into a global indicator that triggers residual reuse across the transformer stack (Cui et al., 2026). These methods can save substantial computation, but their shared decisions provide limited flexibility for modules that evolve differently within the same denoising step.

Several methods provide reuse decisions across layers or modules (Selvaraju et al., 2024; Liu et al., 2024; Zhao et al., 2025). Learning-to-Cache optimizes a timestep-dependent router, and HarmoniCa trains a cache policy using an image-error proxy (Ma et al., 2024a; Huang et al., 2025). Cache Me if You Can derives block schedules from profiled feature changes and trains scale-shift corrections in its quality-enhanced variant (Wimbauer et al., 2024).

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.

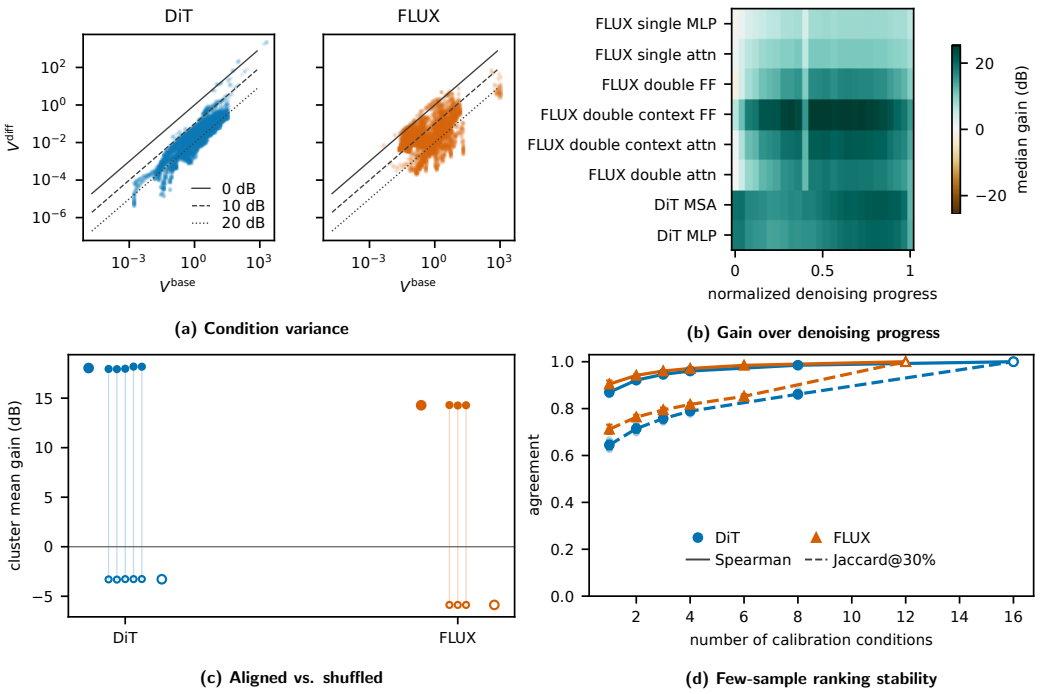


Figure 1: Empirical evidence for Conditional Common-Mode Rejection. **(a)** Condition variance before and after aligned differencing, with 0, 10, and 20 dB references; negative gain indicates greater variance after differencing. **(b)** Module-family median gain over normalized denoising progress. **(c)** Filled/open markers show aligned/condition-shuffled gains. Large markers and error bars denote means and 95% bootstrap intervals; small paired markers are seed means. **(d)** Subset-to-full Spearman correlation and Jaccard overlap of the lowest-scoring 30% of positions. Hollow endpoints are full-condition references (1.0 by definition). DiT-XL/2 uses 16 classes and five seeds; FLUX.1-dev uses 12 prompts and three seeds (sampling and bootstrap details in the supplement).

Among training-free offline approaches, ECAD searches binary timestep–block–component schedules through evolutionary evaluation; for PixArt, it reports 44 GPU-hours for one schedule after engineering changes and approximately 700 NVIDIA A6000 GPU-hours for a full three-point frontier with its research implementation (Aggarwal et al., 2026). SODA models sensitivity across timesteps, layers, and modules to optimize cache intervals and guide token pruning (Shao et al., 2026). Foresight instead profiles each generation to guide layer reuse (Adnan et al., 2025). Given fixed reuse settings, our method constructs a deterministic module schedule from one or two calibration conditions in the main experiments, without generated-output quality feedback, combinatorial schedule search, or per-generation profiling.

We observe that condition-induced deviations in module outputs often remain aligned across adjacent steps of deterministic diffusion-ODE trajectories (Lipman et al., 2023; Liu et al., 2022) when the initial noise is fixed and only the class or text condition varies. Adjacent differencing thus attenuates shared condition variation while retaining the module's temporal evolution, although the outputs remain condition dependent. We separately examine the stability of reuse rankings across calibration subsets in Figure 1; attenuation alone does not establish ranking stability. We use a normalized ratio ρ of consecutive ℓ_1 module displacements to rank timestep–layer–module positions and aggregate priorities from a few conditions into a provisional binary *Cache Book*. Because consecutive reuse can make the stored reference older than the preceding timestep, a second full-compute calibration pass recalibrates the priorities using a scorer-only reference updated by the provisional book. The final book assigns fixed module-wise compute-or-reuse decisions during inference and can operate within a step-wise cache gate, with each schedule acting at a different granularity.

Our contributions are threefold:

- We document reduced sample condition variance after aligned adjacent-step differencing in two evaluated diffusion transformers, termed Conditional Common-Mode Rejection (CCMR), and separately assess the stability of displacement-ratio rankings across calibration subsets.

- We introduce a deterministic, training-free procedure that converts full-compute calibration trajectories into a fixed timestep–layer–module reuse schedule. A second scoring pass updates a scorer-only reference according to the provisional schedule without changing the latent trajectory.
- We evaluate the resulting schedules on image and video diffusion transformers and show that module-level reuse can reduce computation on its own and can provide additional acceleration when nested within fixed step-wise cache policies. The reported operating points exhibit model- and setting-dependent fidelity trade-offs.

2 Related Work

Diffusion models and inference acceleration. Diffusion transformers extend iterative image and video synthesis to scalable Transformer backbones (Ho et al., 2020; Rombach et al., 2022; Peebles & Xie, 2023; Ho et al., 2022). Inference acceleration includes faster solvers (Song et al., 2021a; Lu et al., 2022), distillation (Salimans & Ho, 2022; Yin et al., 2024a;b), sparse attention (Xi et al., 2025), and model compression (Wu et al., 2024). Feature caching instead reuses intermediate computation without changing pretrained weights.

Feature caching for diffusion transformers. Existing caching methods differ in both reuse granularity and schedule construction. TeaCache, MagCache, SeaCache, AdaCache, and BWCache reuse residual features through decisions shared across substantial portions of a denoising step (Liu et al., 2025a; Ma et al., 2025; Chung et al., 2026; Kahatapitiya et al., 2025; Cui et al., 2026). More selective strategies exploit structural or token-level redundancy (Zou et al., 2026; Lou et al., 2024), including the stage-dependent block reuse of Δ-DiT and the token-wise selection of ToCa (Chen et al., 2024; Zou et al., 2024). Fine-grained schedules have also been obtained through learned policies (Ma et al., 2024a; Wimbauer et al., 2024; Huang et al., 2025). ECAD searches binary timestep–block–component cache schedules with an evolutionary quality–cost objective (Aggarwal et al., 2026). SODA estimates offline sensitivity at the timestep, layer, and module levels, then optimizes cache intervals by dynamic programming and adaptively prunes tokens (Shao et al., 2026). Foresight adapts layer reuse during each generation (Adnan et al., 2025). Unlike these schedule-construction procedures, ours assigns complete-module compute-or-reuse decisions through a fixed family-wise ranking calibrated on a few conditions, with neither evolutionary or dynamic-programming search nor per-generation profiling. It can also be nested within an existing step-wise cache.

3 Understanding Condition-Robust Temporal Differences

Because class labels and text prompts can substantially change intermediate activations, few-condition calibration requires stable reuse rankings. We test whether adjacent differencing suppresses condition variation and whether rankings remain stable across calibration subsets.

3.1 Problem Setup and Condition-Centered Differences

Let $Z_{l,m,t}(c;\xi) \in \mathbb{R}^{D_{l,m}}$ denote the flattened output of module m in layer l at inference step t, under condition c and random state ξ. We index t by sampling order rather than by the training diffusion index and isolate condition variation by fixing ξ while varying c. For the finite condition set $\mathcal{C}$ used in each measurement, let $\mathbb{E}_c$ denote its empirical mean at fixed ξ and define

$$\mu_{l,m,t}(\xi) = \mathbb{E}_c[Z_{l,m,t}(c;\xi)], \qquad Q_{l,m,t}(c;\xi) = Z_{l,m,t}(c;\xi) - \mu_{l,m,t}(\xi). \tag{1}$$

The centered variable $Q_{l,m,t}$ captures deviations of module outputs across conditions under the same random state. Centering the adjacent displacement $d_{l,m,t}(c;\xi) = Z_{l,m,t}(c;\xi) - Z_{l,m,t-1}(c;\xi)$ then gives the exact identity

$$d_{l,m,t}(c;\xi) - \mathbb{E}_c[d_{l,m,t}(c;\xi)] = Q_{l,m,t}(c;\xi) - Q_{l,m,t-1}(c;\xi). \tag{2}$$

Equation 2 is algebraic; attenuation of condition variation additionally requires the centered deviations to remain approximately aligned at adjacent steps, $Q_{l,m,t}(c;\xi) \approx Q_{l,m,t-1}(c;\xi)$. In that case, the shared condition-dependent component largely cancels while the condition-averaged temporal displacement remains. This alignment is an empirical property of the trajectory, not an architectural identity or a claim that the original outputs are condition independent. It does not guarantee that a module location can be cached without affecting the generated output.

3.2 Conditional Common-Mode Rejection

To quantify this cancellation for fixed (l, m, t, ξ), we compare the condition variance of the raw output with that of its adjacent displacement, using a symmetric adjacent-step baseline:

$$V^{\mathrm{raw}}_{l,m,t} = \frac{1}{D_{l,m}} \mathbb{E}_c \left[\|Q_{l,m,t}(c;\xi)\|_2^2 \right], \tag{3}$$

$$V^{\mathrm{diff}}_{l,m,t} = \frac{1}{D_{l,m}} \mathbb{E}_c \left[\| Q_{l,m,t}(c;\xi) - Q_{l,m,t-1}(c;\xi) \|_2^2 \right], \tag{4}$$

$$V^{\mathrm{base}}_{l,m,t} = \frac{1}{2} \left(V^{\mathrm{raw}}_{l,m,t} + V^{\mathrm{raw}}_{l,m,t-1} \right). \tag{5}$$

We refer to the observed suppression of condition variance by aligned adjacent differencing as *Conditional Common-Mode Rejection* (CCMR), quantified by the residual ratio and rejection gain

$$R^{\mathrm{CCMR}}_{l,m,t} = \frac{V^{\mathrm{diff}}_{l,m,t}}{V^{\mathrm{base}}_{l,m,t} + \epsilon}, \qquad G^{\mathrm{CCMR}}_{l,m,t} = 10 \log_{10} \frac{V^{\mathrm{base}}_{l,m,t} + \epsilon}{V^{\mathrm{diff}}_{l,m,t} + \epsilon}. \tag{6}$$

Positive gain indicates reduced condition variance, whereas negative gain indicates greater variance after differencing. Gains of 10 and 20 dB correspond to baseline-to-residual variance ratios of approximately $10\times$ and $100\times$. Figure 1 reports the controlled measurements for a class-conditioned DiT and a text-conditioned FLUX model. Defining the family-macro gain as the equally weighted mean of module-family median gains, we obtain 18.23 dB for DiT and 14.66 dB for FLUX. These gains correspond to geometric means of the family-median baseline-to-difference variance ratios of $66.5\times$ for DiT and $29.3\times$ for FLUX. Positive gain occurs in 99.86% of valid DiT module–timestep cells and 98.14% of valid FLUX cells under the measured conditions and seeds. These fractions describe cells rather than probabilities of statistical significance. The degree of suppression is not uniform across modules: FLUX family medians range from 6.79 dB for the single-stream MLP to 22.60 dB for the double-stream context FFN, a variation obscured by a pooled or step-level summary.

To distinguish condition alignment from temporal smoothness, we pair $Z_t(c)$ with $Z_{t-1}(\pi(c))$, where π deranges conditions within each seed. After shuffling, the hierarchical mean gain falls from 18.04 to -3.28 dB for DiT and from 14.28 to -5.87 dB for FLUX, yielding gaps of 21.32 and 20.16 dB. This control shows that the variance reduction depends on preserving condition correspondence across adjacent states rather than on differencing a smooth trajectory alone.

3.3 Few-Sample Ranking Stability

Although CCMR shows reduced condition variance after adjacent differencing, a statistic combining consecutive displacements may yield unstable rankings when small denominators amplify residual variation. We therefore test ranking agreement across calibration subsets.

Panel (d) compares subset-averaged scores with full-condition scores at the same timestep–layer–module positions. Spearman correlation measures overall ranking agreement, while Jaccard overlap compares the lowest-scoring 30%. With one calibration condition, family-macro Spearman correlations are 0.870 for DiT and 0.904 for FLUX, rising to 0.946 and 0.960 with three conditions. The corresponding Jaccard overlaps increase from 0.645 and 0.712 to 0.757 and 0.794. The lower Jaccard values show that agreement on selected reuse positions is weaker than overall rank agreement. These subset measurements show ranking stability in the tested settings, separately from CCMR; neither result assesses cached generation quality.

4 Method

The variance reduction and subset-ranking agreement in Section 3 motivate scoring relative temporal changes rather than condition-specific activations. We build a fixed *Cache Book* in two calibration stages, as summarized with its inference procedure in Figure 2.

4.1 Module-Level Reuse Priority

We retain $Z_{l,m,t}(c;\xi)$ from Section 3.1, with m denoting either an independently cacheable module or an implementation group that must be executed together. Sampling steps are indexed $0, \ldots, T-1$ in generation order. For a deterministic calibration set $\mathcal{D}_{\mathrm{cal}} = \{(c_k, \xi_k)\}_{k=1}^{K}$, define the adjacent ℓ_1 displacement and its ratio at $t = 1, \ldots, T-2$ as

$$\delta^{(k)}_{l,m,t} = \| Z_{l,m,t}(c_k;\xi_k) - Z_{l,m,t-1}(c_k;\xi_k) \|_1, \tag{7}$$

$$\rho^{(0,k)}_{l,m,t} = \frac{\delta^{(k)}_{l,m,t+1}}{\delta^{(k)}_{l,m,t} + \epsilon}. \tag{8}$$

We use ℓ_1 because its linear aggregation of absolute coordinate differences gives large individual differences less relative weight than the ℓ_2 norm. We call ρ the *normalized adjacent ℓ_1 displacement*

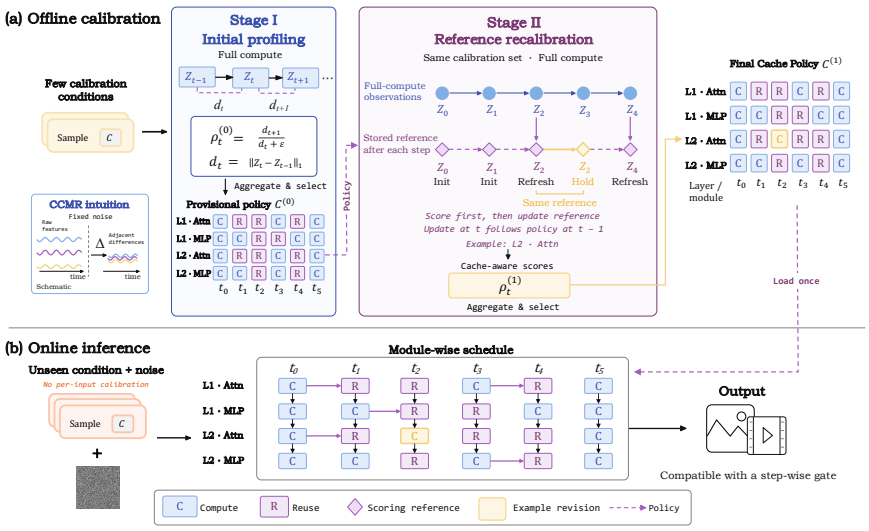


Figure 2: Overview of fine-grained offline caching. **(a) Offline calibration.** Stage I profiles full-compute trajectories and converts normalized adjacent ℓ_1 displacement ratios into a provisional Cache Book $C^{(0)}$. Stage II repeats full computation on the same calibration set; after recording each indexed score, its provisional bit updates or holds the scorer-only reference. These scores produce the final book $C^{(1)}$. Scoring precedes reference updating. **(b) Online inference.** $C^{(1)}$ is loaded once and assigns compute (C) or reuse (R) to every timestep–layer–module position. Purple paths denote references or reuse, and orange marks a decision revised by Stage II. The six-step schedules and CCMR inset are schematic; the module book can be nested inside a step-wise gate.

ratio: values below one indicate approximate contraction of the next displacement and receive higher reuse priority. Relative contraction does not bound cache error or final generation quality: a small ratio can coexist with a large absolute displacement. Although values can vary with class or prompt, Section 3.3 finds comparatively stable induced rankings across the studied calibration subsets.

Here $\delta_{l,m,t} = \|d_{l,m,t}\|_1$ is the scalar magnitude of the vector displacement $d_{l,m,t}$ in Section 3.1; the method figure abbreviates this magnitude as d_t. Book index t denotes the compute-or-reuse action at sampling step t, not the transition to step $t+1$. Its offline score requires full-compute observations at $t-1$, t, and $t+1$: upon receiving Z_{t+1}, the streaming scorer writes ρ_t to position t. This look-ahead is used only during calibration; online inference reads the finished book without computing ρ_t. Positions without two valid adjacent displacements cannot receive a reuse decision.

Scores are averaged over calibration pairs at each location:

$$\bar{\rho}^{(s)}_{l,m,t} = \frac{1}{K}\sum_{k=1}^{K} \rho^{(s,k)}_{l,m,t}, \qquad s \in \{0,1\}, \tag{9}$$

where s indexes the two calibration stages. We average ratios within each location rather than divide averaged displacements, so a small denominator on one trajectory can affect the resulting priority. Let $f(m)$ denote the family of cache unit m, and let $\mathcal{V}^{(s)}_f$ contain the valid aggregated scores at all locations belonging to family f. With a family quantile $q_f \in [0,1]$, we select positions within each family rather than imposing one cutoff across different score distributions:

$$C^{(s)}_{l,m,t} = \mathbf{1}\left[\bar{\rho}^{(s)}_{l,m,t} < Q_{q_{f(m)}}\left(\mathcal{V}^{(s)}_{f(m)}\right)\right]. \tag{10}$$

Here $C = 1$ denotes reuse; the protected set $\mathcal{P}$ includes a configurable prefix, while invalid positions and the final denoising step are also forced to compute. The quantile therefore targets a priority fraction, which may exceed the realized reuse fraction after these restrictions.

4.2 Two-Stage Cache-Book Calibration

Stage I of Algorithm 1 evaluates each cacheable module on full deterministic trajectories, averages adjacent-step ratios over $\mathcal{D}_{\text{cal}}$, and obtains the provisional book $C^{(0)}$ via Equation 10. Because these ratios compare neighboring observations, they may understate displacement from a reference left stale by successive reuse at inference.

Algorithm 1 Two-stage construction of a module-level Cache Book

Require: Denoiser f; calibration set $\mathcal{D}_{\text{cal}}$; family quantiles $\boldsymbol{q}$; protected set $\mathcal{P}$; stabilizer ϵ
Ensure: Final Cache Book $C^{(1)}$

$S^{(0)}_{l,m,t} \leftarrow [\,]$ for each valid book position
for all $(c_k, \xi_k) \in \mathcal{D}_{\text{cal}}$ **do**
  $\{\rho^{(0,k)}_{l,m,t}\} \leftarrow \textsc{AdjacentScores}(f, c_k, \xi_k, \epsilon)$
  append each $\rho^{(0,k)}_{l,m,t}$ to $S^{(0)}_{l,m,t}$
end for
$\bar{S}^{(0)} \leftarrow \textsc{LocationMean}(S^{(0)})$
$C^{(0)} \leftarrow \textsc{FamilyQuantile}(\bar{S}^{(0)}, \boldsymbol{q}, \mathcal{P})$
$S^{(1)}_{l,m,t} \leftarrow [\,]$ for each valid book position
for all $(c_k, \xi_k) \in \mathcal{D}_{\text{cal}}$ **do**
  obtain $Z_{l,m,0}$ and $Z_{l,m,1}$ from the full-compute stream
  $(R_{l,m}, a_{l,m}, h_{l,m}) \leftarrow (Z_{l,m,1}, \|Z_{l,m,1} - Z_{l,m,0}\|_1, 0)$ for all (l, m)
  for $j = 2, \ldots, T-1$ during the full-compute stream **do**
    $t \leftarrow j - 1$ ▷ book position scored by Z_j
    for all cache units (l, m) **do**
      $\widetilde{d} \leftarrow \|Z_{l,m,j} - R_{l,m}\|_1$; append $\widetilde{d}/(a_{l,m} + \epsilon)$ to $S^{(1)}_{l,m,t}$
      if $C^{(0)}_{l,m,t} = 0$ **or** $h_{l,m} = 0$ **then**
        $(R_{l,m}, a_{l,m}) \leftarrow (Z_{l,m,j}, \widetilde{d})$
      end if
      $h_{l,m} \leftarrow C^{(0)}_{l,m,t}$ ▷ score first, then update
    end for
  end for
end for
$\bar{S}^{(1)} \leftarrow \textsc{LocationMean}(S^{(1)})$
return $C^{(1)} \leftarrow \textsc{FamilyQuantile}(\bar{S}^{(1)}, \boldsymbol{q}, \mathcal{P})$

Stage II performs *cache-aware reference recalibration* on the same calibration pairs, computing every module along the unchanged full-compute latent trajectory while updating only the scorer's reference according to $C^{(0)}$. Suppressing (l, m, k) indices, initialize each cache unit with $R_1 = Z_1$, $a_1 = \|Z_1 - Z_0\|_1$, and reuse-run state $h_1 = 0$. For each full-compute observation Z_j, $j \geq 2$, score book position $t = j - 1$ before updating the scorer state:

$$\widetilde{d}_j = \|Z_j - R_{j-1}\|_1, \qquad \rho^{(1)}_{j-1} = \frac{\widetilde{d}_j}{a_{j-1} + \epsilon}. \tag{11}$$

After recording $\rho^{(1)}_t$, the scorer applies $C^{(0)}_t$: it sets $R_j = Z_j$ and $a_j = \widetilde{d}_j$ if $C^{(0)}_t = 0$ or $h_{j-1} = 0$, and otherwise retains R_{j-1} and a_{j-1}; then $h_j = C^{(0)}_t$. Continuing reuse is thus scored against an increasingly stale reference. Although $C^{(0)}_t$ specifies the decision at step t online, applying it after observing Z_{t+1} is only an offline scoring convention, not a replay of cached activations. Aggregation and the same family-wise quantiles yield the final book $C^{(1)}$, without changing the full-compute trajectory or modeling downstream perturbations from cached features.

4.3 Offline Schedule Execution

At sampling step t, the fixed Cache Book entry $C^{(1)}_{l,m,t}$ directs cache unit (l, m) to compute and overwrite the cache (0) or reuse its last computed output (1). The first and final steps and protected positions always compute; other decisions follow network order without a router, per-input profiling, threshold search, or online score update. With classifier-free guidance (Ho & Salimans, 2022), both branches share the Boolean book but cache activations separately, as distinct batch elements or in branch-specific slots. A step-wise gate can enclose the module schedule: whole-step reuse bypasses it, while computed steps follow $C^{(1)}$; both policies retain their original construction.

We evaluate each book on conditions and random states held out from calibration, with fixed model weights, module partition, sampler, step count, resolution, guidance, and numerical precision. Changing these settings requires recalibration or a separate transfer evaluation.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

We evaluate DiT-XL/2 and FLUX.1-dev for images, and Wan2.1-T2V-1.3B and HunyuanVideo-1.5 for videos (Peebles & Xie, 2023; Labs, 2024; Wan et al., 2025; Wu et al., 2025). Tables 1 and 2 list resolutions, sampling steps, frame counts, and reuse settings. Within each model, methods share the checkpoint, conditions, seeds, initial noise, and sampler; calibration conditions are held out from evaluation. ToCa uses model adaptations (Zou et al., 2024), while MagCache and SeaCache retain fixed step-wise policies (Ma et al., 2025; Chung et al., 2026). For module caching alone, the Cache Book uses one calibration condition on DiT, Wan2.1, and HunyuanVideo, and two on FLUX; both stages use the same conditions. In Ours rows, P protects initial steps, K counts calibration conditions, and $\boldsymbol{q}$ lists module-family reuse quantiles in the order given in the supplementary material. MagCache's K instead sets its maximum reuse horizon; τ is the step-wise threshold. For ToCa, $\mathcal{N}$ is the freshness threshold and R is the nominal token-reuse eligibility fraction.

Table 1: Image-generation results on DiT-XL/2 and FLUX.1-dev.

(a) DiT-XL/2, 512^2, DDIM-50

Method (configuration)	Time (s)↓	Speed↑	TFLOPs↓	FID↓	IS↑	PSNR↑	SSIM↑	LPIPS↓
Full	1.325	1.00×	104.90	18.70	449.12	–	–	–
ToCa ($\mathcal{N}=5$, $R=60\%$; 512 adaptation)	1.006	1.32×	51.43	14.46	**382.53**	15.50	0.522	0.510
ToCa ($\mathcal{N}=8$, $R=60\%$; 512 adaptation)	0.977	1.36×	**47.08**	15.80	240.05	14.41	0.418	0.616
Ours ($P=2$, $q_{\mathrm{MSA}}=55\%$, $q_{\mathrm{MLP}}=55\%$, $K=1$)	**0.771**	**1.72**×	51.61	**14.16**	351.24	**19.10**	**0.658**	**0.323**

(b) FLUX.1-dev, 1024^2, 28 steps

Method (configuration)	Time (s)↓	Speed↑	PFLOPs↓	FID↓	IS↑	CLIP↑	IR↑	PSNR↑	SSIM↑	LPIPS↓
Full	8.084	1.00×	2.083	20.02	29.43	0.703	0.934	–	–	–
ToCa ($\mathcal{N}=6$, $R=75\%$; FLUX adaptation)	4.954	1.63×	**0.792**	**21.62**	27.79	**0.700**	0.822	**19.85**	**0.715**	**0.357**
Ours ($P=2$, $\boldsymbol{q}=$ [.70, .70, .30, .23, .50, .02], $K=2$)	**4.881**	**1.66**×	1.582	22.52	**28.03**	0.694	**0.869**	16.29	0.672	0.414
MagCache ($\tau=.24$)	3.953	2.04×	0.818	**20.91**	**29.76**	**0.702**	**0.877**	**24.89**	**0.834**	**0.180**
MagCache + Ours ($\tau=.24$, $\boldsymbol{q}=$ [.3, .3, 0, 0, .3, 0])	**3.707**	**2.18**×	**0.759**	21.12	28.98	0.699	0.826	18.31	0.697	0.366
SeaCache ($\tau=.30$)	5.438	1.49×	1.202	20.28	**29.49**	**0.704**	0.913	**29.18**	**0.918**	**0.075**
SeaCache + Ours ($\tau=.30$, $\boldsymbol{q}=$ [.3, .3, .2, .2, .3, .2])	**4.624**	**1.75**×	**1.041**	**19.93**	28.90	**0.704**	**0.917**	20.99	0.786	0.239

Image FID and Inception Score (IS) (Heusel et al., 2018; Salimans et al., 2016) are computed from 5,000 generated samples using ImageNet (Deng et al., 2009) for DiT and a category-balanced MJHQ-30K subset for FLUX (Li et al., 2024); FLUX also reports CLIPScore and ImageReward (Radford et al., 2021; Hessel et al., 2022; Xu et al., 2023). PSNR, SSIM, and LPIPS (Wang et al., 2004; Zhang et al., 2018) compare paired outputs with Full. Videos use 16 prompts and two seeds per model; the six reported VBench custom-input dimensions are percentages (Huang et al., 2024). We measure batch-one sampling time after a timing warm-up run on an H800 and estimate executed denoiser matrix-product FLOPs. Speedups are computed against the displayed original-model Full time, irrespective of the caching method. Dataset splits, calibration settings, and metric accounting are detailed in the supplementary material.

5.2 IMAGE GENERATION

On DiT (Table 1), Ours achieves 1.72× speedup and a FID-5K of 14.16, compared with 1.32–1.36× and 14.46–15.80 for ToCa, with higher paired PSNR and SSIM and lower LPIPS. ToCa at



Figure 3: Matched FLUX.1-dev outputs at 1024^2, 28 steps, seed 42, for P1–P5 (full prompts in the supplementary material). MagCache uses $\tau = .24$; speedups are from Table 1.

$\mathcal{N} = 5$ nevertheless retains higher IS. On FLUX, Ours reaches $1.66\times$ against $1.63\times$ for ToCa and has higher IS and ImageReward, whereas ToCa has lower FID and better paired fidelity.

On FLUX, our module Cache Book raises MagCache speedup from $2.04\times$ to $2.18\times$ and SeaCache speedup from $1.49\times$ to $1.75\times$ without changing either step-wise policy. In this FLUX evaluation, SeaCache + Ours has lower FID (19.93 versus 20.28) and higher ImageReward (0.917 versus 0.913); MagCache + Ours has lower ImageReward, and both increase paired LPIPS. Figure 3 shows acceleration: at $2.18\times$, Ours + MagCache retains the principal subjects and broadly similar layouts from module caching alone at $1.66\times$, despite changes in composition and detail.

5.3 Video Generation and Cross-Granularity Composition

Table 2 shows module caching alone reaches $1.68\times$ on Wan2.1 and $1.54\times$ on HunyuanVideo-1.5. These points trade off fidelity for speed: on Wan2.1, dynamic degree falls from 31.2 to 15.6, and paired PSNR, SSIM, and LPIPS are less favorable than for the step-wise baselines on both models.



Figure 4: Matched HunyuanVideo-1.5 generations at 720p, 121 frames, 50 steps, and seed 4100. Frames are sampled at 0.5, 2.5, and 4.5 s; SeaCache uses $\tau = .20$. Row speedups are from Table 2.

Table 2: Video-generation results on Wan2.1 and HunyuanVideo-1.5. VBench columns report subject and background consistency (Subj, Back), motion smoothness and dynamic degree (Mot, Dyn), and aesthetic and imaging quality (Aes, Img) as separate percentages.

Method (configuration)	Efficiency			VBench (%)						Paired fidelity		
	Time (s)↓	Speed↑	PFLOPs↓	Subj↑	Back↑	Mot↑	Dyn↑	Aes↑	Img↑	PSNR↑	SSIM↑	LPIPS↓
Wan2.1-T2V-1.3B (832 × 480, 81 frames, 50 steps)												
Full	114.91	1.00×	28.320	97.3	97.7	98.8	31.2	64.0	71.3	–	–	–
Ours ($P = 2$, $K = 1$, $\boldsymbol{q} = [.50, .35, .20]$)	**68.39**	**1.68**×	**17.030**	96.1	96.3	98.8	15.6	57.9	66.5	15.66	0.539	0.416
MagCache ($\tau = .12$)	83.79	1.37×	12.463	**97.3**	**97.8**	**98.8**	28.1	**64.2**	**71.6**	**25.64**	**0.875**	**0.073**
MagCache + Ours ($\tau = .12$, $\boldsymbol{q} = [.3, .3, 0]$)	**57.30**	**2.01**×	**11.484**	96.9	97.7	**98.8**	**31.2**	63.8	71.0	19.90	0.720	0.183
SeaCache ($\tau = .20$)	84.81	1.35×	13.596	**97.3**	**97.8**	**98.8**	28.1	63.8	**71.5**	**30.02**	**0.931**	**0.039**
SeaCache + Ours ($\tau = .20$, $\boldsymbol{q} = [.3, .3, 0]$)	**59.41**	**1.93**×	**12.438**	97.0	97.7	**98.8**	**31.2**	**64.5**	71.4	19.90	0.723	0.180
HunyuanVideo-1.5 (720p, 121 frames, 50 steps)												
Full	1880.89	1.00×	632.470	92.7	93.8	98.8	84.4	57.6	67.3	–	–	–
Ours ($P = 2$, $K = 1$, $\boldsymbol{q} = [.90, .45, .04, .12, 0, 0]$)	**1219.03**	**1.54**×	**387.610**	92.4	93.8	99.0	75.0	52.4	62.1	12.94	0.432	0.571
MagCache ($\tau = .03$, $K = 2$, $r = .25$)	1124.05	1.67×	366.834	**92.8**	**93.9**	**98.8**	**84.4**	**57.3**	**66.7**	**22.04**	**0.746**	**0.181**
MagCache + Ours ($\tau = .03$, $K = 2$, $r = .25$; $P = 5$, $\boldsymbol{q} = [.90, .80, 0, 0, 0, 0]$)	**793.88**	**2.37**×	**237.404**	92.3	93.5	98.7	**84.4**	55.4	65.6	15.41	0.506	0.412
SeaCache ($\tau = .20$)	904.50	2.08×	290.542	**92.8**	93.6	**98.8**	**84.4**	**57.0**	**66.7**	**20.49**	**0.697**	**0.223**
SeaCache + Ours ($\tau = .20$, $\boldsymbol{q} = [.6, .5, 0, 0]$)	**797.78**	**2.36**×	**241.924**	92.7	**94.0**	98.7	**84.4**	56.4	65.2	15.66	0.527	0.386

On Hunyuan, SeaCache + Ours raises speedup from $2.08\times$ to $2.36\times$ with similar VBench scores. Caching substantially reduces fidelity to matched Full samples, but may still produce plausible images and videos that retain semantic content and overall visual quality (Figures 3 and 4).

5.4 Ablation Studies

Reuse-priority score. We compare our normalized adjacent-displacement score with adjacent-feature mean squared difference and cosine distance. For each score, we rebuild both Cache Books with fixed calibration conditions, module-family quantiles, protected prefix, and inference settings. These fixed quantiles yield slightly different realized reuse and FLOPs. Table 3 reports a small

paired FLUX and Wan2.1 evaluation. Compared with adjacent MSE and cosine distance, our score yields higher ImageReward and lower paired LPIPS on FLUX, and higher paired PSNR and lower LPIPS on Wan2.1, while reusing slightly fewer modules.

Table 3: Reuse-priority diagnostic with fixed family quantiles; realized reuse and FLOPs are not matched. Evaluation uses 32 paired FLUX and eight paired Wan2.1 prompts. PSNR/LPIPS compare same-input Full outputs; Temp.Δ is the temporal LPIPS difference from Full (Appendix C). Timing and FLOPs accounting are detailed in Appendix G.

(a) FLUX.1-dev, 1024^2, 28 steps

Priority score	Reuse (%)	Time (s)↓	PFLOPs↓	CLIP↑	IR↑	LPIPS↓
Full	0	7.018	2.083	0.728	1.136	–
Ours	26.21	4.905	1.582	0.712	1.182	0.412
Adjacent MSE	28.36	4.774	1.547	0.720	0.567	0.536
Adjacent cosine distance	28.46	4.742	1.543	0.703	0.839	0.499

(b) Wan2.1-T2V-1.3B, 832×480, 81 frames, 50 steps

Priority score	Reuse (%)	Time (s)↓	PFLOPs↓	PSNR↑	LPIPS↓	Temp.Δ ↓
Full	0	118.646	28.323	–	–	–
Ours	31.67	68.796	17.032	16.242	0.471	0.023
Adjacent MSE	33.49	69.006	16.516	14.540	0.690	0.046
Adjacent cosine distance	33.53	68.297	16.490	14.876	0.652	0.024

Two-stage calibration. With other settings fixed, we compare the provisional Stage-I and final books on identical inputs. Despite similar reuse counts, Stage II changes 670 of 4,256 FLUX decisions, 1,015 of 2,800 DiT decisions, and 1,033 of 4,500 Wan2.1 decisions. The supplementary material reports paired, prompt- or class-conditioned image quality. Figure 5 uses the FLUX prompt "Artophagous." and ImageNet class 279 (Arctic fox); Figure 6 uses "A happy fuzzy panda playing guitar nearby a campfire, snow mountain in the background."

Figures 5 and 6 illustrate the visual effect of the second calibration stage. In these examples, Stage I exaggerates the FLUX canine face, softens the DiT fox's details, and shifts the Wan2.1 panda relative to the campfire. The final book reduces the facial distortion and restores a Wan2.1 composition closer to Full, while retaining a recognizable DiT subject.

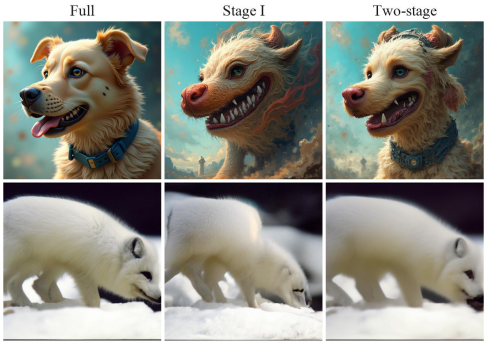


Figure 5: Matched-noise FLUX.1-dev (top) and DiT-XL/2 (bottom) examples chosen after inspection; not estimates of average performance.

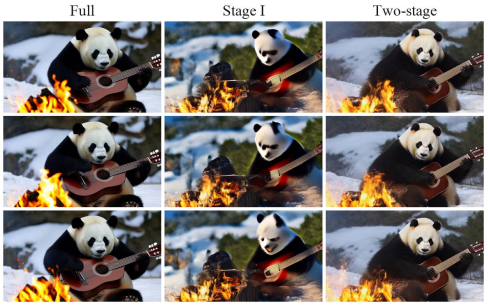


Figure 6: A Wan2.1 example chosen after inspection, with matched noise. Columns: Full, Stage I, and two-stage calibration; rows: frames 16, 40, and 64 of 81. This case illustrates differences, not average performance.

6 Limitations and Conclusion

We introduced a training-free, fine-grained cache that ranks normalized temporal-displacement ratios to build fixed module schedules from few calibration conditions. Each Cache Book depends on model weights, module partition, sampler, step count, resolution, guidance, and numerical precision; changes require recalibration or separate transfer validation. Family-wise reuse quantiles are operating-point hyperparameters, not automatically selected quality constraints. CCMR and ranking stability on finite condition sets motivate reuse but bound neither cache error nor generation quality. Stage II updates scorer-only references using the provisional book along full-compute trajectories, without modeling cached-activation error propagation; current evidence does not establish aggregate quality gains over Stage I. Appendix H details evaluation and comparison limitations. Within the tested configurations, our method accelerates image and video sampling and composes with fixed step-wise caching, but greater reuse can alter outputs despite recognizable content, with fidelity depending on model, cache configuration, and operating point.

AI USE STATEMENT

We used generative AI tools to assist with research implementation, literature review, and manuscript preparation. Specifically, AI tools assisted in writing code for visualization experiments, including those reported in Figure 1, and for benchmark evaluation pipelines, as well as in organizing experimental data and results. For the literature review, we used AI tools to search for relevant work on caching in diffusion models, summarize prior studies, and assist with comparative analysis of related methods. During manuscript preparation, AI tools helped refine wording to improve clarity and readability and adjust LaTeX formatting and typesetting.

The authors reviewed the AI-assisted work, tested the relevant code, and checked the organized data and cited sources. We take full responsibility for the final content of this work, including the code, data processing, visualizations, interpretations, citations, and scientific claims produced with AI assistance.

ETHICS STATEMENT

This work studies inference acceleration for pretrained image and video diffusion models. Reducing generation cost may benefit legitimate applications but may also facilitate the production of misleading or harmful synthetic media. Our caching method does not address biases or safety limitations of the underlying models, and the reported quality metrics do not establish safety. Use of the method should respect the licenses and use restrictions of the underlying models and datasets and retain appropriate safeguards for downstream applications.

REPRODUCIBILITY STATEMENT

Section 4 and Algorithm 1 describe Cache-Book calibration and execution. Section 5.1 and Appendix A document experimental configurations and evaluation protocols; Appendices C and G clarify fidelity diagnostics, timing, and FLOPs accounting. Appendices D and E provide the prompts used in the main qualitative comparisons. The accompanying anonymous code includes calibration and sampling implementations, configuration presets, and model-specific setup instructions. Pretrained weights must be obtained separately under their providers' terms.

REFERENCES

Muhammad Adnan, Nithesh Kurella, Akhil Arunkumar, and Prashant Nair. Foresight: Adaptive layer reuse for accelerated and high-quality text-to-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. 169078–169109. Curran Associates, Inc., 2025. doi: 10.52202/085713-5631. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/f76b6afa2b400989a3f3c0d50f0f9736-Paper-Conference.pdf.

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.

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

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.

Hanshuai Cui, Zhiqing Tang, Zhifei Xu, Zhi Yao, Wenyi Zeng, and Weijia Jia. Bwcache: Accelerating video diffusion transformers through block-wise caching. In *International Conference on Learning Representations*, volume 2026, pp. 105765–105791, 2026.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=FPnUhsQJ5B.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning, 2022. URL https://arxiv.org/abs/2104.08718.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018. URL https://arxiv.org/abs/1706.08500.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

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

Yushi Huang, Zining Wang, Ruihao Gong, Jing Liu, Xinjie Zhang, Jinyang Guo, Xianglong Liu, and Jun Zhang. Harmonica: Harmonizing training and inference for better feature caching in diffusion transformer acceleration. In *Forty-second International Conference on Machine Learning*, 2025.

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.

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 *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15240–15252. IEEE, 2025.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

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

Daiqing Li, Aleks Kamko, Ehsan Akhgari, Ali Sabet, Linmiao Xu, and Suhail Doshi. Playground v2. 5: Three insights towards enhancing aesthetic quality in text-to-image generation. *arXiv preprint arXiv:2402.17245*, 2024.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 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 *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7353–7363. IEEE, 2025a.

Haozhe Liu, Wentian Zhang, Jinheng Xie, Francesco Faccio, Mengmeng Xu, Tao Xiang, Mike Zheng Shou, Juan-Manuel Perez-Rua, and Jürgen Schmidhuber. Faster diffusion via temporal attention decomposition. *arXiv preprint arXiv:2404.02747*, 2024.

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, 2022. URL https://arxiv.org/abs/2209.03003.

Jinming Lou, Wenyang Luo, Yufan Liu, Bing Li, Xinmiao Ding, Weiming Hu, Yuming Li, and Chenguang Ma. Token caching for diffusion transformer acceleration. *arXiv preprint arXiv:2409.18523*, 2024. URL https://arxiv.org/abs/2409.18523.

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, June 2025. ISSN 2731-5398. doi: 10.1007/s11633-025-1562-4. URL http://dx.doi.org/10.1007/s11633-025-1562-4.

Zhengyao Lyu, 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*, volume 2025, pp. 33132–33156, 2025.

Xinyin Ma, Gongfan Fang, Michael Bi Mi, and Xinchao Wang. Learning-to-cache: Accelerating diffusion transformer via layer caching. *Advances in Neural Information Processing Systems*, 37: 133282–133304, 2024a.

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, 2024b.

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.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

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.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/8a3363abe792db2d8761d6403605aeb7-Paper.pdf.

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.

Tong Shao, Yusen Fu, Guoying Sun, Jingde Kong, Zhuotao Tian, and Jingyong Su. SODA: Sensitivity-Oriented Dynamic Acceleration for Diffusion Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 33012–33021, June 2026.

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

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, 2023. URL https://arxiv.org/abs/2303.01469.

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.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, Xiaoliang Dai, Ji Hou, Zijian He, Artsiom Sanakoyeu, Peizhao Zhang, Sam Tsai, Jonas Kohler, et al. Cache me if you can: Accelerating diffusion models through block caching. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6211–6220. IEEE, 2024.

Bing Wu, Chang Zou, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Jack Peng, Jianbing Wu, Jiangfeng Xiong, Jie Jiang, et al. Hunyuanvideo 1.5 technical report. *arXiv preprint arXiv:2511.18870*, 2025.

Junyi Wu, Haoxuan Wang, Yuzhang Shang, Mubarak Shah, and Yan Yan. Ptq4dit: Post-training quantization for diffusion transformers. *Advances in neural information processing systems*, 37: 62732–62755, 2024.

Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, et al. Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. *arXiv preprint arXiv:2502.01776*, 2025.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: learning and evaluating human preferences for text-to-image generation. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 15903–15935, 2023.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo 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, 2024b.

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, 2018. doi: 10.1109/CVPR.2018.00068.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *NeurIPS*, 2023.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. In *International Conference on Learning Representations*, volume 2025, pp. 3296–3319, 2025.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. *arXiv preprint arXiv:2410.05317*, 2024.

Chang Zou, Shikang Zheng, Evelyn Zhang, Runlin Guo, Haohang Xu, Zhengyi Shi, Conghui He, Xuming Hu, and Linfeng Zhang. Rethinking token-wise feature caching: Accelerating diffusion transformers with dual feature caching. *IEEE Transactions on Image Processing*, 2026.

SUPPLEMENTARY MATERIAL

A EXPERIMENTAL PROTOCOL AND CONFIGURATION

DiT-XL/2 uses DDIM-50 at 512^2 and FLUX.1-dev uses 28 steps at 1024^2; Wan2.1-T2V-1.3B uses 50 steps and 81 frames at 832×480, and HunyuanVideo-1.5 uses 50 steps and 121 frames at 720p. Cache Books are calibrated on conditions separate from evaluation inputs, and video metrics use native-length outputs.

The two calibration stages reuse the same K_{cal} condition–random-state pairs, corresponding to $\mathcal{D}_{\text{cal}}$ in the main paper. Stage I records adjacent full-compute module displacements; Stage II again executes every module, updating its scorer-only reference with each provisional bit after recording the corresponding score. Thus calibration requires $2K_{\text{cal}}$ full-compute denoising trajectories, in addition to scoring and book construction. This is a trajectory count, not a measured wall-clock cost. The final book is fixed before evaluation: it is neither retrained nor selected using the reported 5K image or 32-video results. For fixed reuse settings, book construction does not search candidate schedules; choosing those settings is a separate step.

A.1 DIFFERENT SAMPLING STEP COUNTS

We recalibrate the Cache Book at each reduced step count while retaining the main-experiment module thresholds and a two-step protected prefix. Table S1 uses 100 ImageNet classes with one seed for DiT-XL/2 at 25 DDIM steps and 64 prompts with one seed for FLUX.1-dev at 20 steps. Full and accelerated outputs share the same initial noise. These paired sets evaluate the method after recalibration at shorter sampling schedules; they are not substitutes for the 5K distributional evaluations in the main paper.

Table S1: Image generation at fewer sampling steps. Time is measured on one H800 at batch size one; FLOPs are estimated denoiser matrix products. PSNR, SSIM, and LPIPS compare matched Full outputs. FLOPs× is the analytic Full-to-method ratio, not a latency speedup.

(a) DiT-XL/2, 512^2, DDIM-25

Method	Time (s)↓	Speed↑	FLOPs (T)↓	FLOPs×	PSNR↑	SSIM↑	LPIPS↓
Full	0.662	1.00×	52.452	1.00×	–	–	–
ToCa ($\mathcal{N} = 3$, $R = 93\%$)	0.521	1.271×	22.474	2.334×	18.204	0.622	0.390
Ours	0.399	1.661×	27.942	1.877×	14.529	0.385	0.615

(b) FLUX.1-dev, 1024^2, 20 steps

Method	Time (s)↓	Speed↑	FLOPs (P)↓	FLOPs×	CLIP↑	IR↑	PSNR↑	SSIM↑	LPIPS↓
Full	5.043	1.00×	1.488	1.00×	0.720	1.030	–	–	–
ToCa ($\mathcal{N} = 5$, $R = 90\%$)	3.819	1.320×	0.754	1.972×	0.725	0.812	21.848	0.727	0.340
Ours	3.588	1.405×	1.145	1.299×	0.719	0.842	19.438	0.725	0.373

Here Ours uses $K_{\text{cal}} = 1$ and $\boldsymbol{q} = (.55, .55)$ on DiT, and $K_{\text{cal}} = 2$ and $\boldsymbol{q} = (.70, .70, .30, .23, .50, .02)$ on FLUX.1-dev. On these small paired sets, the faster module-cache rows do not uniformly preserve paired fidelity better than ToCa.

For video, Table S2 uses eight Wan2.1 prompts and four HunyuanVideo-1.5 prompts, each with one seed, at 30 sampling steps. The native resolutions and lengths remain 832×480 at 81 frames and 720p at 121 frames, respectively. We recalibrate a model-specific book at 30 steps using $P = 2$ and $K_{\text{cal}} = 1$. The module thresholds remain $\boldsymbol{q} = (.50, .35, .20)$ for Wan and $\boldsymbol{q} = (.90, .45, .04, .12, 0, 0)$ for HunyuanVideo.

Table S2: Video generation at 30 sampling steps. Time is per native-length video on one H800 (batch size one); FLOPs are estimated denoiser matrix products. VBench columns are individual *custom-input* dimensions on these small sets, reported as percentages, not the official Total Score. Paired fidelity compares each accelerated video with its matched Full video.

Model / method	Time (s)↓	Speed↑	FLOPs (P)↓	Subj↑	Back↑	Mot↑	Dyn↑	Aes↑	Img↑	PSNR↑	SSIM↑	LPIPS↓
Wan2.1 / Full	68.960	1.00×	16.994	97.6	97.8	99.2	25.0	61.9	70.0	–	–	–
Wan2.1 / Ours	49.912	1.381×	10.626	92.8	94.0	99.0	12.5	53.9	58.4	15.888	0.549	0.507
HunyuanVideo / Full	1145.015	1.00×	379.482	92.6	92.7	99.1	100.0	54.4	70.9	–	–	–
HunyuanVideo / Ours	772.067	1.483×	241.085	93.1	93.0	99.3	75.0	43.2	53.8	14.817	0.398	0.555

The reduced-step video runs show measured speedups, but several custom-input VBench dimensions decline, particularly aesthetics and imaging. Wan Full media passed a framewise zero-cache consistency check. The earlier HunyuanVideo Full media did not match the zero-cache output of the evaluation implementation, so all four Full references were regenerated using that implementation and the matched input protocol. The reported paired metrics use these regenerated references, not the earlier media; this replacement does not establish pixelwise equivalence to the earlier Full implementation. The Wan 30-step FLOPs count uses the actual 60 classifier-free-guidance denoiser calls rather than the 100 calls of the 50-step protocol. These small sets are diagnostic and do not establish official VBench Total Score differences.

A.2 Module Cache Books

Table S3 specifies the pure module-cache configurations used in the main experiments. Here P is the number of initial denoising steps forced to compute in full, and K_{cal} is the number of calibration condition–random-state pairs. These quantities should not be confused with the MagCache maximum reuse horizon K_{mag}. The entries of $\boldsymbol{q}$ are module-family selection thresholds, *not* measured reuse percentages; actual reuse and estimated FLOPs follow the execution record. Each displayed configuration uses a Cache Book calibrated for its own model, resolution, step count, and video length.

Table S3: Pure module-cache configurations used in the main experiments.

Model	Steps	P	K_{cal}	$\boldsymbol{q}$
DiT-XL/2	50	2	1	$(.55, .55)$
FLUX.1-dev	28	2	2	$(.70, .70, .30, .23, .50, .02)$
Wan2.1	50	2	1	$(.50, .35, .20)$
HunyuanVideo-1.5	50	2	1	$(.90, .45, .04, .12, 0, 0)$

For DiT, the vector order is self-attention and MLP. For FLUX, it is double-stream image attention, double-stream context attention, double-stream image feed-forward, double-stream context feed-forward, single-stream attention, and single-stream MLP. For Wan2.1, it is self-attention, cross-attention, and feed-forward. For HunyuanVideo-1.5, it is double-stream image attention, text attention, image MLP, text MLP, single-stream attention, and single-stream MLP. The last two Hunyuan entries remain zero in this implementation.

The protected prefix applies to the *module* policy. In a hybrid run, the step-wise gate first decides whether to reuse the entire denoiser step; only steps that pass through that gate consult the module Cache Book. A whole-step reuse therefore does not count as a module hit, and the module policy cannot override the step-wise decision. The pure-module settings in Table S3 and the hybrid settings below use separately calibrated books.

A.3 Token-cache Comparison Settings

The image-table ToCa rows use a token-wise partial-refresh policy rather than our module Cache Book. In the reported notation, $\mathcal{N}$ is the freshness threshold and R is the nominal fraction of tokens eligible for reuse (one minus the fresh-token ratio). The DiT rows use a 512^2 adaptation; the FLUX row uses the indicated FLUX adaptation. These are specific tested operating points, not a common threshold shared by the two models.

The DiT adaptation runs the public ToCa DiT model and its `ToCa-ddim50` ratio scheduler at 512^2 under the shared DiT-XL/2, DDIM-50, class-conditioning, CFG, and VAE protocol; its token-selection and refresh rules are unchanged. The FLUX adaptation uses ToCa's released FLUX model and cache logic with FLUX.1-dev weights, while prompt encoding, initial latents, timestep schedule, guidance, and VAE decoding follow the shared diffusers protocol. Neither adaptation uses our module Cache Book.

Table S4: ToCa operating points reported in the image table.

Model	$\mathcal{N}$	R	Reported speedup
DiT-XL/2 (512^2)	5	60%	1.32×
DiT-XL/2 (512^2)	8	60%	1.36×
FLUX.1-dev (1024^2)	6	75%	1.63×

A.4 Fixed Step-wise Policies for Compatibility

For each compatibility pair, B enables only the indicated step-wise policy and $B+M$ adds a module Cache Book without modifying that policy. The MagCache parameters used for the main-paper compatibility experiment are listed in Table S5; r denotes its initial retention ratio. The SeaCache entry gives its fixed threshold. The FLUX MagCache operating point $(\tau, K_{\mathrm{mag}}, r) = (.24, 4, .20)$ differs from the public MagCache4FLUX example $(.24, 5, .10)$.[1]

Table S5: Fixed step-wise policies used for the main-paper compatibility pairs.

Model	MagCache $(\tau, K_{\mathrm{mag}}, r)$	SeaCache τ
FLUX.1-dev	$(.24, 4, .20)$	.30
Wan2.1	$(.12, 4, .20)$	.20
HunyuanVideo-1.5	$(.03, 2, .25)$	.20

The HunyuanVideo-1.5 SeaCache policy is an adaptation of the original HunyuanVideo strategy; we do not describe its numerical setting as an official HunyuanVideo-1.5 recommendation. Module thresholds for each $B + M$ pair are shown beside the method in the main tables.

Table S6 collects the module part of every main-table compatibility row. The vector order follows the model-specific definition above. The Hunyuan SeaCache book only enables its four double-stream families; the corresponding six-position notation has two trailing zeros. The FLUX and Wan hybrid books and the Hunyuan SeaCache hybrid book protect the first $P = \max(1, \lfloor 0.10T \rfloor)$ denoising steps, where T is the prescribed step count: $P = 2$ for FLUX at 28 steps and $P = 5$ for both video models at 50 steps. The Hunyuan MagCache hybrid also uses $P = 5$, whereas all pure-module books in Table S3 use $P = 2$. The prefix is indexed by denoising steps, not by the number of times the step-wise gate permits module execution. Protected and invalid book positions are forced to compute at the module level; the step-wise gate still determines whether the module policy is consulted.

Table S6: Module thresholds in the six $B + M$ compatibility settings. The step-wise parameters remain those of Table S5.

Model	Step-wise policy	Hybrid $\boldsymbol{q}$
FLUX.1-dev	MagCache	$(.30, .30, 0, 0, .30, 0)$
FLUX.1-dev	SeaCache	$(.30, .30, .20, .20, .30, .20)$
Wan2.1	MagCache	$(.30, .30, 0)$
Wan2.1	SeaCache	$(.30, .30, 0)$
HunyuanVideo-1.5	MagCache	$(.90, .80, 0, 0, 0, 0)$
HunyuanVideo-1.5	SeaCache	$(.60, .50, 0, 0)$

[1] Official example: `https://github.com/Zehong-Ma/MagCache/blob/df81cb181776c2c61477c08e1d21f87fda1cd938/MagCache4FLUX/magcache_flux.py`.

A.5 Evaluation Sets and Metric Interpretation

The image distributional results use 5,000 generated samples: 1,000 ImageNet classes for DiT and a fixed, category-balanced 5,000-prompt subset of MJHQ-30K for FLUX. The resulting FID and Inception Score are 5K estimates, not standard ImageNet FID-50K or full MJHQ-30K scores. Image fidelity compares 100 classes with two seeds for DiT and 64 prompts with two seeds for FLUX against matched Full outputs. The video paired set contains 16 prompts with two seeds per model. Paired metrics are aligned by prompt, seed, and frame index. VBench is evaluated on those custom inputs; the six reported dimensions are shown as percentages, but are not an official aggregate VBench Total Score.

Sampling time is measured at batch size one after model warm-up on an H800. Estimated denoiser matrix-product FLOPs use the actual compute/reuse trace, with one multiply–accumulate counted as two FLOPs and jointly executed attention counted once. The estimate omits non-matrix operators, text encoding, decoding, and cache-transfer overhead, so it is not a substitute for measured latency.

Video frame-level fidelity is associated with each prompt–seed video; temporal differences are averaged within a video before aggregation over seeds and prompts. The six reported VBench dimensions are subject, background, motion, dynamic degree, aesthetic quality, and imaging quality.

B Compatibility with Few-Step Distillation

Our cross-step module reuse requires more than one denoiser evaluation: a one-step generator offers no preceding step from which our Cache Book can reuse a module output. We therefore test a distinct, eight-step distilled FLUX Transformer, SenseFlow-FLUX, rather than treating the 28-step FLUX.1-dev results as evidence for few-step distillation.[2] The comparison holds the distilled weights, 1024^2 resolution, deterministic Euler sampler, eight steps, prompts, seeds, and initial noise fixed. *Distilled Full* disables all caching; *Distilled Full + Ours* adds only a module Cache Book independently calibrated on this eight-step trajectory. The two stages use $K_{\mathrm{cal}} = 2$ calibration prompts, and the first two steps are forced to compute in full. The FLUX module-family order defined above gives $\boldsymbol{q} = (.50, .50, 0, 0, .50, 0)$. Eight separate prompts were used to choose this point; the 64 evaluation prompts were not used for calibration or selection. A zero-cache run was pixel-identical to the unmodified distilled pipeline under matched inputs.

Table S7: Eight-step SenseFlow-FLUX compatibility on 64 held-out prompts (one fixed seed per prompt, batch size one). Sampling speedup is relative to the *same distilled model* without caching, not to 28-step FLUX.1-dev. FLOPs estimate denoiser matrix products only.

Method	Time (s)↓	Speedup↑	FLOPs (P)↓	Attn. reuse↑
Distilled Full	2.097	1.000×	0.595	0%
Distilled Full + Ours	1.835	1.143×	0.539	22.81%

Table S8: Quality on the same 64 paired prompts. LPIPS compares each cached output with its matched distilled Full output. Full self-comparisons are omitted.

Method	CLIPScore↑	ImageReward↑	LPIPS↓
Distilled Full	0.67519	1.18344	—
Distilled Full + Ours	0.67549	1.18203	0.1475

The execution trace records 24/152 jointly reusable double-stream attention positions and 80/304 single-stream attention positions; neither feed-forward family is reused. The combined 22.81% attention reuse in Table S7 counts jointly executed attention only once. The FLOPs estimate uses one multiply–accumulate as two FLOPs and omits non-matrix operators and cache overhead. Across prompts, the 95% bootstrap interval for the measured speedup is $[1.142, 1.144]\times$. For the cached run minus Full, the paired CLIPScore difference is $+0.00029$ $[-0.00944, +0.01037]$ and the ImageReward difference is -0.00141 $[-0.04498, +0.05020]$. These intervals do not establish quality equivalence. In particular, a prompt requiring storefront lettering showed more distorted text under caching, despite similar aggregate scores. Two-stage calibration took 57 seconds; dividing by the

[2] Public model and sampling instructions: `https://github.com/XingtongGe/SenseFlow`.

observed mean per-image saving of approximately 0.262 seconds gives a rough 217-image amortization point, excluding candidate search and model loading. This experiment supports additional caching on this particular eight-step distilled model; it does not test one-step DMD, DMD2-SDXL, or a four-step SenseFlow setting.

We also test video generation with the official HunyuanVideo-1.5 480p I2V step-distilled Transformer at its 12-step setting.[3] This is a separate image-to-video comparison, not the 720p, 50-step text-to-video model in the main paper. The square source images are assigned a 640×640 output bucket; each output has 121 frames. Super-resolution and prompt rewriting are disabled. Nineteen source images are taken in a fixed SHA-256 order from the existing FLUX Full-image set: one for calibration, two for selecting a module threshold, and 16 held out for evaluation. All use the same fixed motion prompt. Within each pair, the source image, motion prompt, seed, initial noise, weights, and sampler are identical. The first two denoising steps are protected. Both calibration stages use the same calibration image; the selected Cache Book has $(q_{\text{img attn}}, q_{\text{txt attn}}) = (.90, .40)$ and zero MLP quantiles. The zero-cache implementation produced a byte-identical video to the uncached path. A more aggressive text-attention quantile of $.60$ was rejected on the two selection videos because of visible brightness and contrast drift; the 16 test videos were not used to choose $.40$.

Table S9: Compatibility on 16 held-out source images with the same 12-step distilled I2V model. Time is the mean of two post-warm-up runs on one H800 with batch size one; speedup uses the distilled Full time. FLOPs estimate denoiser matrix products, counting jointly executed attention once.

Method	Time (s)↓	Speedup↑	FLOPs (P)↓	Joint attn. reuse↑
Distilled Full	66.58	1.000×	17.491	0/648
Distilled Full + Ours	52.17	1.276×	13.655	163/648

Table S10: Quality on the same 16 held-out videos. Top: six raw VBench `custom_input` dimensions (not the official Total Score). Bottom: frame-aligned fidelity to the matched distilled Full video, aggregated per video and then over 16 videos; Full self-comparisons are omitted.

Method	Subject↑	Background↑	Motion↑	Dynamic↑	Aesthetic↑	Imaging↑
Distilled Full	0.9725	0.9742	0.9953	0.0000	0.6811	0.6993
Distilled Full + Ours	0.9721	0.9735	0.9952	0.0000	0.6811	0.6982

Method	PSNR↑	SSIM↑	LPIPS↓	Temporal LPIPS Δ ↓
Distilled Full	—	—	—	—
Distilled Full + Ours	30.74	0.935	0.0258	0.00078

The execution trace shows that both sides of the jointly executed attention reuse the same 163 of 648 block–step positions; the requested image-side quantile alone does not represent saved compute. Across the 16 paired generations, Full takes 69.61 seconds and Full + Ours takes 53.10 seconds on average, consistent in direction with the separate post-warm-up timing in Table S9. The six custom VBench dimensions remain close on this small set, but dynamic degree is zero for both methods and therefore offers no evidence of motion fidelity. Visual inspection also reveals a limitation: on one held-out input containing prominent lettering, the cached run exhibits stronger late-frame text distortion than its matched Full video. Thus these results demonstrate nonzero module reuse and a measured speed benefit on this particular 12-step distilled I2V model, not quality equivalence or compatibility with one-step DMD variants.

[3] Checkpoint and sampling code: `https://github.com/Tencent-Hunyuan/HunyuanVideo-1.5`.



Figure S1: Matched distilled Full and Full + Ours outputs for one held-out image-to-video input (frames 0, 30, 60, 90, and 120). Both rows use the same source image, motion prompt, seed, and initial noise. This example illustrates one matched pair; the text-distortion failure case described above is retained in the evaluation set.

C Additional Video Fidelity Diagnostic

The main tables include frame-aligned PSNR, SSIM, and LPIPS. Here we report the temporal LPIPS difference that was measured on the same native-length paired videos. For a candidate video Y and its matched Full result Y^0, let d_{LPIPS} be the perceptual distance between two frames. We compute

$$D_{\mathrm{temp}}(Y, Y^0) = \frac{1}{F-1} \sum_{t=1}^{F-1} \left| d_{\mathrm{LPIPS}}(Y_t, Y_{t+1}) - d_{\mathrm{LPIPS}}(Y_t^0, Y_{t+1}^0) \right|. \tag{S1}$$

The reported value first averages transitions within each video, then seeds within a prompt, and finally prompts. Lower values mean closer adjacent-frame LPIPS changes to the matched Full video; this diagnostic does not by itself establish perceptual motion quality or freedom from flicker.

Table S11: Temporal LPIPS difference from matched Full videos (lower is better).

Method	Wan2.1	HunyuanVideo-1.5
Ours	0.0210	0.0240
MagCache (B)	0.0032	0.0049
MagCache + Ours ($B + M$)	0.0054	0.0133
SeaCache (B)	0.0029	0.0070
SeaCache + Ours ($B + M$)	0.0055	0.0127

D Full Prompts for the FLUX Visual Comparison

The five prompts in the main-paper FLUX figure compare Ours and Ours + MagCache ($\tau = .24$). They are prompts P1–P5 of the fixed seed-42 comparison set, shown in full below. Both methods use FLUX.1-dev at 1024^2, 28 steps, and guidance 3.5; only the cache policy changes. The headings above the images abbreviate these prompts to keep them legible.

1. The grand, opulent lobby of an Art Deco skyscraper. Polished brass, geometric patterns on the marble floor, and a massive, intricate chandelier casting warm light. Cinematic, symmetrical, elegant, 1920s style.
2. A detailed portrait of a vibrant, colorful toucan perched on a mossy branch. The background is a lush, out-of-focus jungle with soft morning light filtering through the leaves. Photorealistic, natural style, high detail, shallow depth of field.
3. A steampunk robot reading a book in a Victorian library.
4. Macro portrait of a silver tabby cat with vivid green eyes, crisp whiskers, rich fur texture, shallow depth of field, cyberpunk megacity at night, rain-slick streets reflecting vivid neon signs and holograms. Flying cars, towering glass and steel facades, cinematic wide angle, high contrast.

5. a photo of a white sandwich

E Full Prompts for the HunyuanVideo Visual Comparison

The two columns of the main-paper HunyuanVideo-1.5 figure use the following prompts, each at 720p, 121 frames, 50 denoising steps, and seed 4100:

1. A cute happy Corgi playing in park, sunset by Hokusai, in the style of Ukiyo
2. An astronaut flying in space, zoom in

Each row uses the same prompt and initial noise across methods. The montage shows zero-based frame indices 12, 60, and 108 (0.5, 2.5, and 4.5 seconds at 24 frames per second). MagCache is a separate step-wise comparison row; the controlled compatibility pair is SeaCache versus SeaCache + Ours at the same $\tau = .20$. The row labels report full evaluation-set speedups from Table 2 of the main paper, not timings of the two illustrative videos. Selected frames cannot establish temporal consistency on their own; the native-length paired diagnostics are reported in Table S11.

F Stage-Wise Score Drift and Cache-Book Revision

We visualize the module-level two-stage calibration using the same operating points as the main experiments: DiT-XL/2 at 512^2 with 50 DDIM steps, $P = 2$, and $(q_{\mathrm{MSA}}, q_{\mathrm{MLP}}) = (.55, .55)$; and FLUX.1-dev at 1024^2 with 28 steps, $P = 2$, and the six family quantiles in Table S3. The DiT calibration uses ImageNet class 207 and seed 2027; FLUX uses "A red ceramic teapot beside two blue cups on a wooden table" and "An elderly violinist performing beneath warm stage lights," both with seed 2027. Both stages execute every module on full-compute trajectories. Stage II changes only the scoring reference according to the provisional book $C^{(0)}$; these plots therefore show *score-reference drift*, not drift of a generated latent trajectory.

Figures S2 and S3 compare the averaged normalized displacement ratios $\bar{\rho}^{(0)}$ and $\bar{\rho}^{(1)}$. The horizontal axis follows denoising order. White hollow circles mark reuse decisions in the corresponding provisional or final Cache Book. Gray columns are protected or otherwise invalid positions, not measured $\rho = 1$ values. Each family uses one shared color scale across its two stages, clipped to the pooled 2nd–98th percentiles for legibility. FLUX shows one preselected double-stream and one single-stream attention family; the decision counts below include all six families.

To isolate consecutive reuse, Figure S4 groups valid module–step positions by the number of consecutive $C^{(0)}$ reuse decisions *before* that score is computed. It plots the median $\log_2 \bar{\rho}^{(1)} - \log_2 \bar{\rho}^{(0)}$; shaded bands are interquartile ranges over positions, not confidence intervals. After at least two preceding reuse decisions, the median shift is $1.45/1.54$ for DiT attention/MLP and $1.42/0.63$ for the two displayed FLUX attention families. This is a descriptive calibration-set effect, not a bound on cache error or a claim that every position's score grows.

Stage II changes $1{,}015/2{,}800$ DiT and $670/4{,}256$ FLUX module–step decisions. Because the family-wise quantiles remain fixed, these changes largely redistribute reuse locations rather than reducing the total reuse count: DiT has $1{,}424$ versus $1{,}423$ reused positions and FLUX has $1{,}352$ requested reuse bits in both books. These FLUX counts treat the image- and context-attention bits separately. At execution, the two bits must both request reuse for their joint attention unit to be reused. Counting that unit once gives $3{,}724 = 28(3 \cdot 19 + 2 \cdot 38)$ executable unit–step positions and 999 versus 976 reused positions, or 26.83% versus 26.21%. Thus identical requested-bit totals need not yield identical effective reuse. Neither count is FLOPs-weighted. No cross-step cache is enabled in these diagnostics. The score and book differences explain why recalibration changes the schedule, but do not by themselves establish a uniform generation-quality improvement.

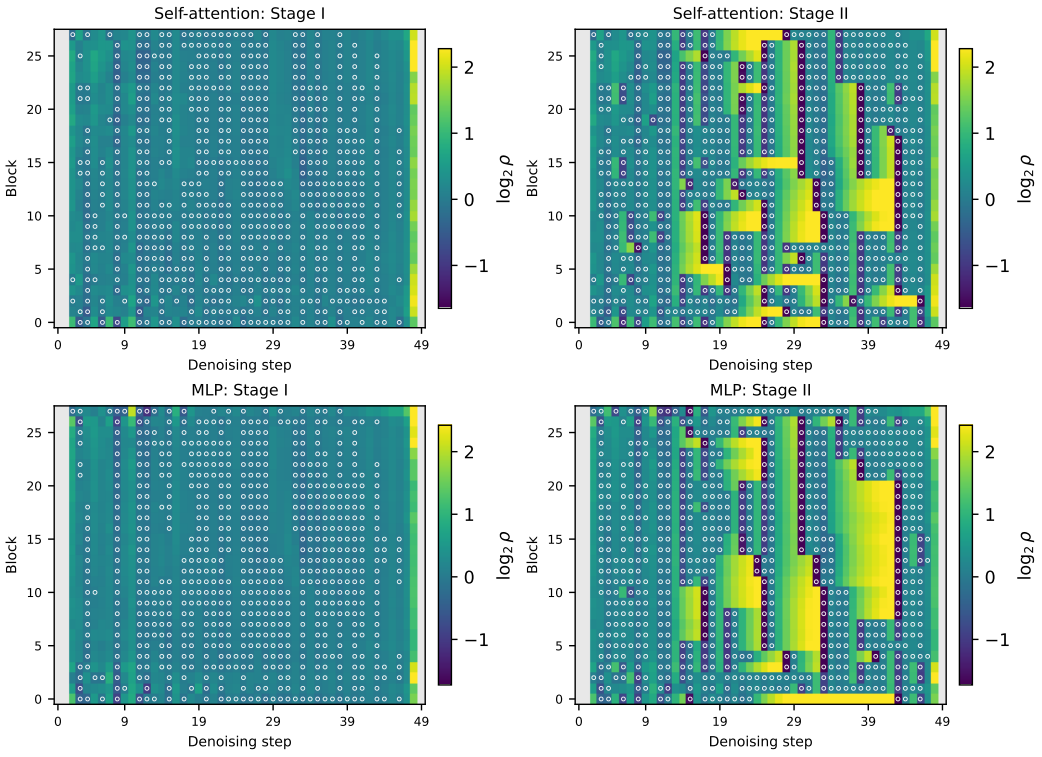


Figure S2: DiT-XL/2 module scores and decisions before (left) and after (right) cache-aware reference recalibration. Rows show self-attention and MLP. Both passes perform full computation; white circles denote positions selected for reuse by the corresponding Cache Book.

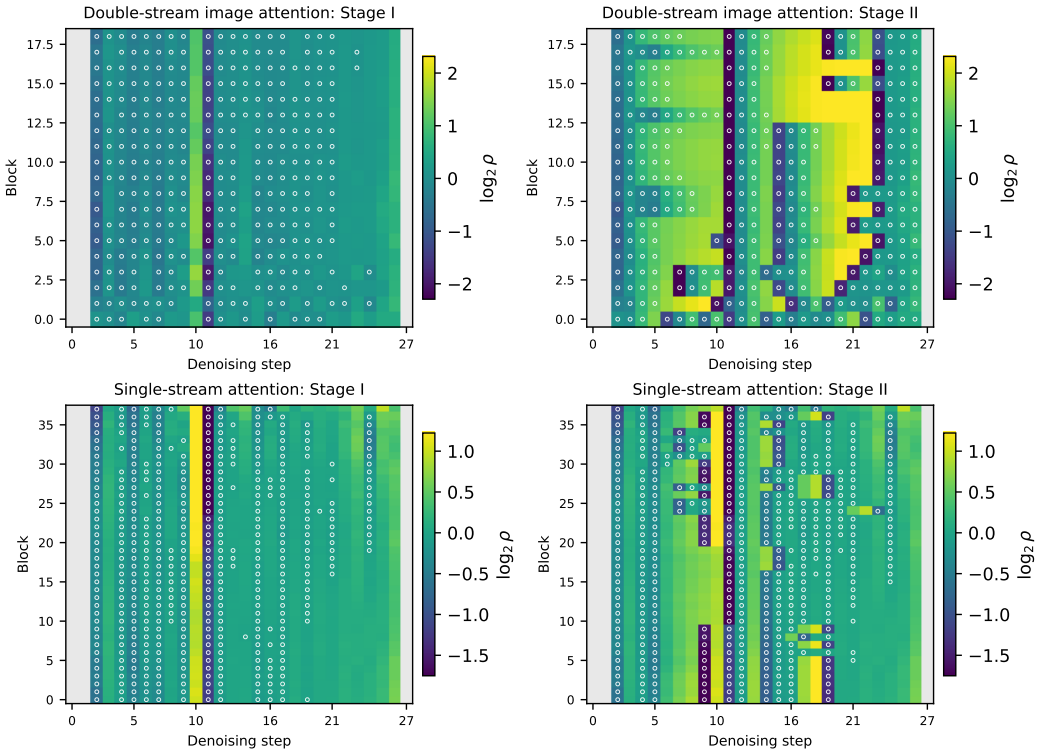


Figure S3: FLUX.1-dev Stage-I and Stage-II module scores, with the same conventions as Figure S2. The displayed families are double-stream image attention and single-stream attention; the decision counts in the text also include the remaining four families.

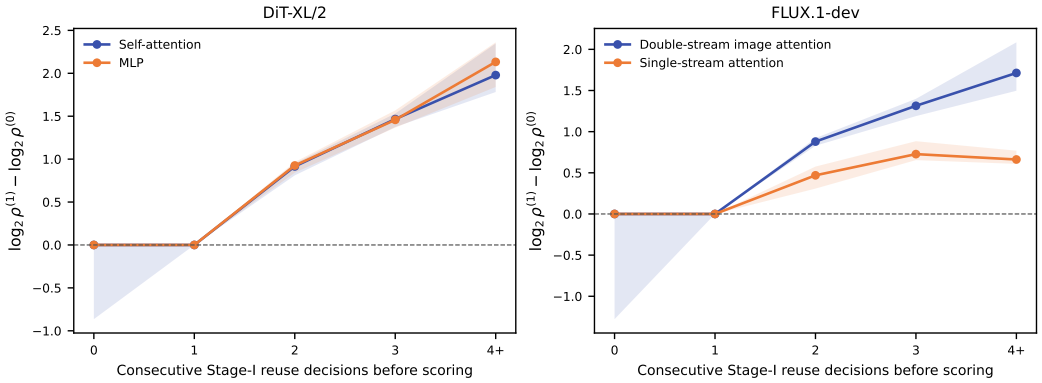


Figure S4: Change in the normalized displacement ratio as the provisional reuse run grows. A positive value means that Stage II assigns a larger ratio to a position than adjacent-step Stage I. Curves show medians and shading shows interquartile ranges across valid module–step positions.

G Scope of the Mechanism and Ablation Evidence

The CCMR analysis in the main paper fixes the initial random state when comparing conditions. DiT uses 16 classes, five seeds, and 50 sampling steps; FLUX uses 12 prompts, 12 prespecified balanced-cycle condition pairs, three seeds, and 28 steps. DiT uses ImageNet class indices $\{0, 7, 37, 142, 207, 279, 340, 387, 417, 569, 656, 751, 817, 933, 980, 992\}$ and seeds 0–4; FLUX uses seeds 0–2. The CCMR collectors use float32 statistics, $\epsilon = 10^{-12}$, and the recorded validity and degeneracy masks: the first step has no adjacent difference, and collector baseline variance below 10^{-8} is marked degenerate and excluded from the gain summaries. DiT uses float32 inference with classifier-free guidance 4.0; FLUX uses bfloat16 inference with guidance 3.5 and maximum sequence length 512.

DiT computes centered variances over its complete 16-class batch. FLUX estimates them from the 12 condition pairs: for raw outputs and adjacent displacements, we average $\|x_i - x_j\|_2^2/(2D)$ over pairs and multiply by $11/12$ before forming the baseline and residual ratio. This is a sampled-pair estimate, not an exact evaluation of all 66 pairs. The pairs form a closed cycle through prompt IDs $02, 04, 07, 10, 05, 11, 03, 00, 09, 01, 08, 06$ (listed below), with the last ID connected to the first; every prompt occurs twice. The cycle was fixed before the formal analysis using selection seed 2027.

Figure 1's confidence intervals use 2,000 seed-then-cluster bootstrap resamples: seeds are independent top-level replicates, while condition pairs, shuffles, and calibration subsets are nested within a seed; layer–step cells are not independent top-level replicates. The condition-shuffled control uses 100 distinct derangements per DiT seed and swaps the two earlier-step conditions within each FLUX pair. For each seed and subset size, ranking stability uses all distinct condition subsets if there are at most 100, otherwise 100 distinct uniformly sampled subsets, each sampled without replacement internally. Subset selection uses seed $2027 + 10{,}000s + k$ for generation seed s and subset size k. Spearman correlation assesses the entire priority ranking, whereas Jaccard overlap assesses only the lowest-scoring 30% of module–timestep positions. Neither measure is an image- or video-quality metric.

Reuse-priority diagnostic. Table 3 uses 32 paired FLUX prompts and eight paired Wan2.1 prompts for quality and FLOPs evaluation. Sampling times are means of two measurements after one warm-up; FLOPs estimate executed denoiser matrix products. Calibration conditions, module-family quantiles, protected prefix, and inference settings are held fixed, but realized reuse and FLOPs differ between scores. This diagnostic therefore does not isolate score quality at matched compute or establish score optimality. Temp.Δ follows the definition in Appendix C.

Two-stage calibration. Table S12 compares the provisional Stage-I book $C^{(0)}$ with the final book $C^{(1)}$ under fixed inference settings and matched initial noise. Each image set contains 32 fixed conditions with one seed; the FLUX prompts and DiT classes are those screened for the displayed examples, so this is an exploratory paired comparison rather than a held-out estimate of population quality. Stage II changes 670 of 4,256 FLUX and 1,015 of 2,800 DiT module–time decisions, while the measured effective, unweighted execution-unit reuse fractions remain close (26.83/26.21% and 50.86/50.82% for $C^{(0)}/C^{(1)}$, respectively). We assess prompt-conditioned quality with CLIPScore and ImageReward on FLUX and target-class CLIPScore on DiT. The latter uses the text template "a photo of a [class name]"; scores are compared only within a model, not across FLUX and DiT.

Table S12: Prompt- or class-conditioned quality for Stage-I versus two-stage calibration, using 32 paired conditions per model. Δ is Stage II minus Stage I; 95% intervals use 10,000 prompt/class bootstrap resamples.

Model	Metric	Stage I	Stage II	Δ [95% CI]
FLUX.1-dev	CLIPScore	0.7092	0.7122	$+0.0031$ [-0.0093, $+0.0135$]
FLUX.1-dev	ImageReward	1.0815	1.1824	$+0.1009$ [-0.0310, $+0.2422$]
DiT-XL/2	Class CLIPScore	0.6307	0.6332	$+0.0026$ [-0.0097, $+0.0153$]

CLIPScore is 2.5 times the nonnegative CLIP ViT-L/14 image–text cosine. All three confidence intervals include zero, so this small set does not establish a consistent semantic-quality gain from

Stage II. FID is not reported: 32 images per book are inadequate for a useful distributional estimate, and a matched large-sample Stage-I generation set has not been produced.

The Wan2.1 books change 1,033 of 4,500 decisions, but the two paired videos used for the main-paper illustration do not support an aggregate quality estimate. The selected image and video examples likewise do not establish a uniform Stage-II benefit. More generally, CCMR supports a few-condition ranking rationale, not a guarantee that each selected reuse position preserves generation quality.

CCMR prompt set. The prompt IDs used in the FLUX cycle and subset analysis refer to the following fixed conditions, distinct from the generation examples in Appendix I:

p00 A cinematic photograph of a red fox sitting beside a moss-covered tree in a sunlit forest, detailed fur and natural colors.

p01 A studio portrait of an elderly astronaut, weathered face, soft rim lighting, highly detailed photography.

p02 A stylish woman walking down a Tokyo street at night, warm neon signs reflecting on wet pavement.

p03 A transparent glass teapot on a dark wooden table, premium product photography, soft window light.

p04 A gourmet cheeseburger with crisp lettuce and melted cheese, restaurant food photography.

p05 A snowy mountain village at dusk, glowing windows, smoke rising from chimneys, cinematic wide view.

p06 A surreal underwater city with glowing jellyfish, crystal towers and schools of colorful fish.

p07 A watercolor painting of a lighthouse on a rocky coast during a storm, expressive brush strokes.

p08 A clean white poster with the large black word INVAR centered on it, minimal graphic design.

p09 A red cube placed to the left of a blue sphere on a plain gray surface, orthographic studio view.

p10 A vintage steam train crossing a desert valley at sunrise, long shadows and cinematic atmosphere.

p11 A small friendly robot watering yellow flowers inside a glass greenhouse filled with morning light.

H LIMITATIONS AND SCOPE

Our conclusions are limited to the evaluated models and inference configurations. A Cache Book is tied to model weights, module partition, sampler, step count, resolution, guidance setting, and numerical precision; transfer across these choices has not been established and requires separate validation or recalibration. The family-wise reuse quantiles are operating-point hyperparameters rather than automatically selected quality constraints. CCMR and subset-ranking stability are empirical observations on finite condition sets, not bounds on cache error or final generation quality. Greater reuse can alter outputs even when scene content remains recognizable, and the speed–fidelity trade-off depends on the model, cache configuration, and operating point.

Stage II updates the scorer's reference history while retaining a full-compute trajectory, so it does not model errors propagated by cached activations. Although it changes many selected locations, the current evaluation does not establish an aggregate quality improvement over the Stage-I book. The exploratory image comparisons use 32 conditions per model screened for the displayed examples, and all reported quality-difference confidence intervals include zero; the two paired video examples do not support an aggregate quality estimate (Appendix G). More broadly, the main video evaluation uses a small custom-input set of 16 prompts and two seeds per model, image FID values are 5K-sample estimates, and the reuse-priority score ablation fixes quantiles rather than matching realized FLOPs. Finally, comparisons with ECAD and SODA concern methodology, not controlled head-to-head performance evaluations.

I Additional Qualitative Examples

The DiT panel shows six complete class pairs selected from twelve candidates. The remaining panels show complete method groups under fixed prompts, seeds, and initial noise. Video examples were screened for visible subject motion and coherent results across all six methods; each row samples four positions from one native-length video. These selected cases illustrate appearance and motion, not average visual quality or temporal-consistency statistics. Speedup labels are taken from the main evaluation tables rather than timings of the displayed examples.

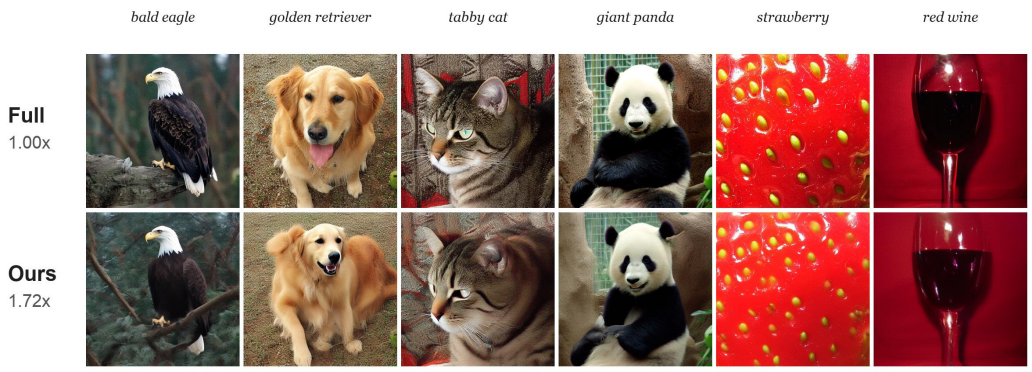


Figure S5: Selected DiT-XL/2 generations at 512^2 and DDIM-50, seed 5200. DiT is class-conditioned; column headings are ImageNet class names rather than text prompts. Each column retains its complete Full/Ours pairing.



Figure S6: FLUX.1-dev at 1024^2, 28 steps, seed 5200: a sunlit botanical greenhouse. The complete prompt appears beneath the six paired outputs. SeaCache and SeaCache + Ours share $\tau = .30$; MagCache and MagCache + Ours share $\tau = .24$.

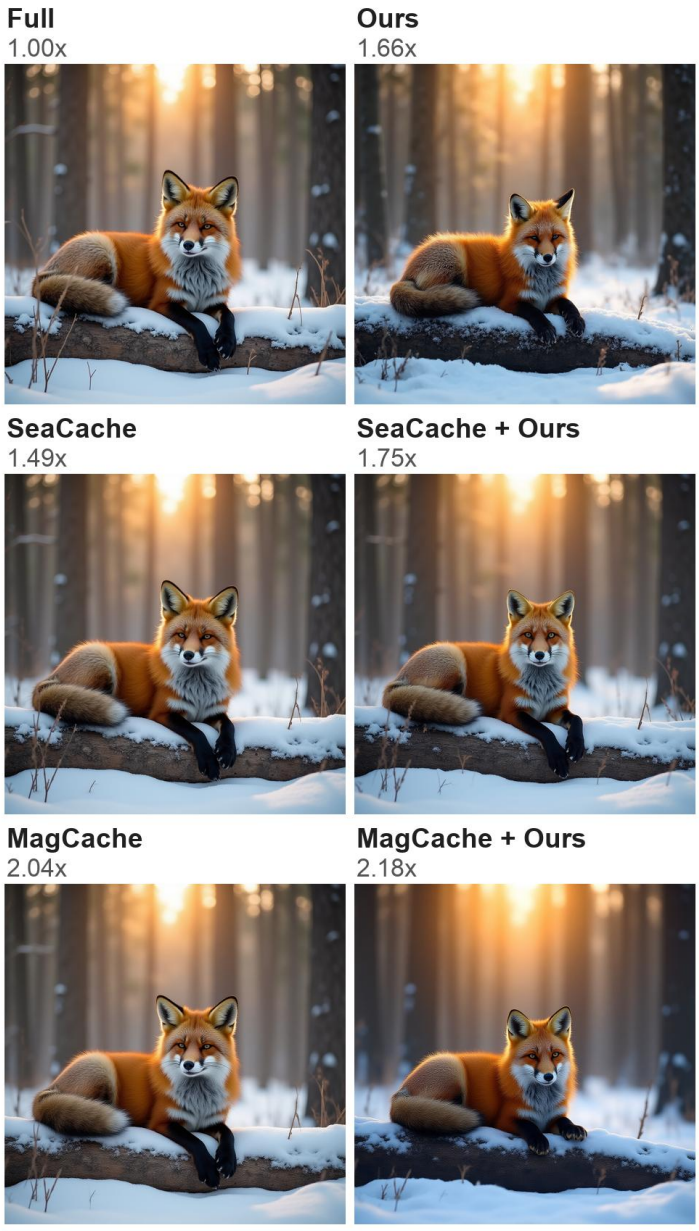


Figure S7: FLUX.1-dev under the same configurations and seed: a red fox in bright winter sunlight. This second prompt contrasts wildlife with the architectural greenhouse scene in Fig. S6.

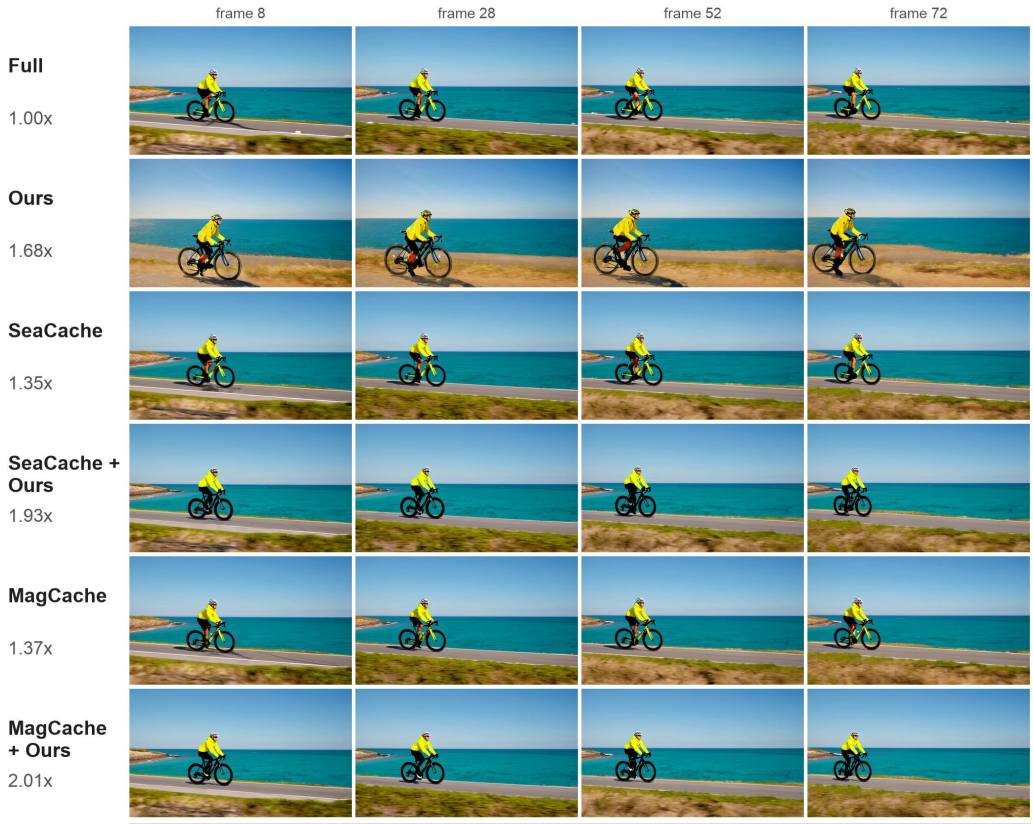


Figure S8: Wan2.1-T2V-1.3B: a cyclist on a sunlit coastal road at 832×480, 81 frames, 50 steps, seed 5200. The complete prompt appears beneath the grid; columns show frames 8, 28, 52, and 72. SeaCache and MagCache paired rows use $\tau = .20$ and $\tau = .12$, respectively.

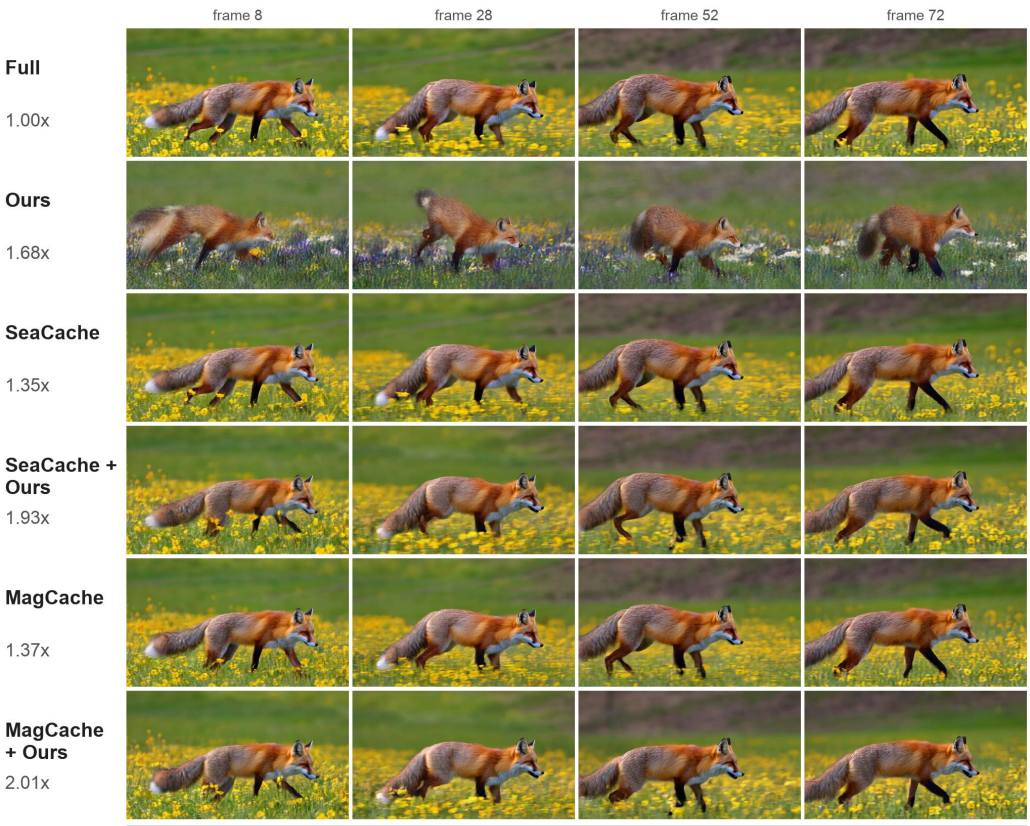
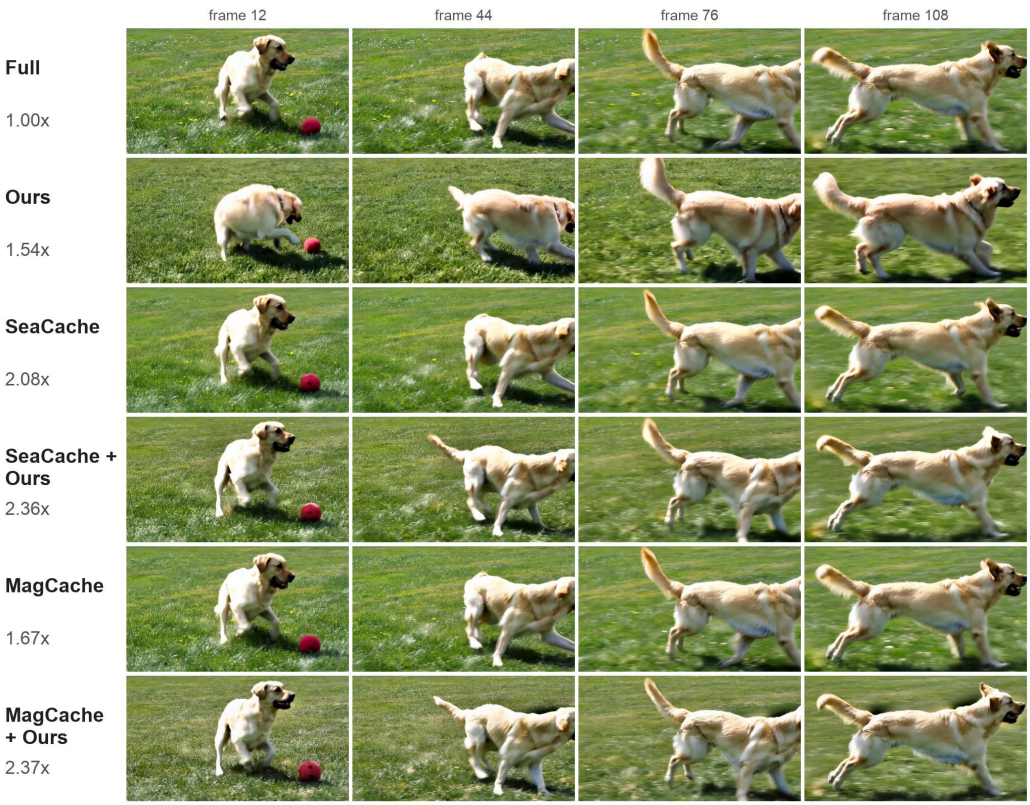


Figure S9: Wan2.1-T2V-1.3B: a fox running through a bright wildflower meadow, under the same six method configurations. All four frames in each row come from one 81-frame video generated with the printed prompt and seed 5200.



"A golden retriever runs after a red ball across a sunlit grassy field, smooth tracking camera, bright clear afternoon."

Figure S10: HunyuanVideo-1.5 at 720p, 121 frames, 50 steps, seed 5200: a golden retriever running after a ball in daylight. Columns show frames 12, 44, 76, and 108. SeaCache paired rows use $\tau = .20$; MagCache paired rows use $\tau = .03$, $K = 2$, and retention $r = .25$.



Figure S11: HunyuanVideo-1.5 under the same six configurations: a red train crossing a sunlit stone bridge. Four temporally aligned frames per method show a distinct moving subject from Fig. S10.