Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

DTC: Dependency-Aware Cache Execution and Deferred Trajectory Correction

Guandong Li

leeguandon@gmail.com

iFLYTEK https://orcid.org/0000-0002-5731-9931

Mengxia Ye

Aegon THTF https://orcid.org/0009-0001-4660-8770

Research Article

Keywords: Diffusion models, Feature caching, Dependency-aware execution, Trajectory correction, Image editing

Posted Date: November 16th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11725324/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

DTC: Dependency-Aware Cache Execution and Deferred Trajectory Correction

Guandong Li*
iFLYTEK, Hefei, China
leeguandon@gmail.com

Mengxia Ye
Aegon THTF, Nanjing, China

Abstract

Learned feature caching reduces diffusion model inference latency by replacing full denoising forwards with low-cost predictions. We propose a joint approach combining Dependency-Aware Cache Execution (DACE) and Deferred Trajectory Correction (DTC). When cache decomposition, prediction, and output projection are token-independent, DACE computes features only for target tokens and manages history liveness by future read positions, reducing cache-path computation and storage. DTC reuses scheduled full steps to measure endpoint counterfactual defects before old history refresh, approximates the error integral using signed sampler step sizes and cache age, and applies correction after normal state updates. A unified dependency plan makes execution pruning compatible with feedback-required history; both layers preserve model weights and the scheduled full DiT call budget. On Qwen-Image-Edit background-replacement tasks, the full method reduces GPU-resident N7 latency by 12.1% and mean LPIPS distance by 18.9% relative to LinCa; DTC adds 2.1% overhead to DACE while further reducing LPIPS by 18.6%.

Keywords: Diffusion models; feature caching; dependency-aware execution; trajectory correction; image editing.

1. Introduction

Diffusion [4, 26] and flow-matching models [13, 17] generate images through repeated network evaluations along a noise schedule. Tens of denoising steps make these evaluations the main inference cost. Latent diffusion models [24] operate on compressed representations; Diffusion Transformers [22] adopt scalable Transformer denoisers. Classifier-free guidance [3] controls generation through conditional and unconditional branch composition. Fast ODE solvers [18, 31] reduce evaluation counts, but full forwards remain the bottleneck. Feature caching exploits temporal redundancy for further acceleration.

Execution dependency. When cache decomposition, prediction, and output projection are token-independent, final consumption uses only target image token outputs, leaving condition token cache features unused. This enables pruning by output dependency: execute decomposition and prediction only for target tokens. History item validity depends on subsequent read positions and predictor stages; managing history by future dependencies reduces wasteful computation and storage.

Sampling state. Cache prediction errors are immediately written into the latent state by the sampler. Refreshing history at full steps updates subsequent predictions, while the accumulated displacement persists in the sampling state. Prior work leverages prediction calibration, counterfactual errors, or history feature compensation [5, 12, 23, 33]. We focus on state displacement over executed cache intervals: whether existing full endpoints provide low-cost feedback observations to retroactively correct prediction defects.

We propose **Dependency-Aware Cache Execution** (DACE) and **Deferred Trajectory Correction** (DTC). DACE restricts decomposition and prediction to target tokens and manages history lifetimes by their last planned read. DTC forms endpoint counterfactual predictions before old history refresh, compares them with scheduled full outputs after CFG composition, estimates the prior interval's defect integral using cache age and signed step sizes, and applies correction after normal updates. Endpoint reads are incorporated into the dependency plan, making execution pruning compatible with feedback-required history.

On Qwen-Image-Edit [29] background-replacement tasks, GPU-resident N7 configuration reduces latency by 12.1% and mean LPIPS distance by 18.9% relative to LinCa [16]. DACE reduces execution cost by 13.9%; DTC further reduces LPIPS by 18.6% with 2.1% overhead. Endpoint signal, weight, and interval contrasts verify feedback approximations and time–quality tradeoffs.

Our contributions are dependency-aware execution over target outputs and history reads, deferred state correction from scheduled full endpoints, and an error decomposition

evaluated through fixed-trajectory diagnostics and closed-loop ablations.

2. Related Work

Diffusion models and numerical sampling. Denoising diffusion [4, 26] and flow matching [13, 17] establish foundations for high-quality generation. Latent diffusion [24] operates on compressed representations; DiT [22] adopts Transformer denoisers. CFG [3] enhances controllability through conditional/unconditional branch composition. Fast ODE solvers [18, 31] achieve similar quality with fewer steps, but full network evaluation remains the primary overhead.

Cache reuse and granularity. Direct caching reuses historical features. DeepCache [20] reuses temporally stable U-Net features; BlockCache [28] constructs nonuniform schedules by analyzing layer-block variation. PAB [32] broadcasts attention outputs; FasterCache [19] dynamically reuses features in video diffusion. ToCA [34] adaptively selects tokens for caching.

Adaptive scheduling and feature prediction. TeaCache [14] rescales differences between timestep-modulated inputs to guide output caching. TaylorSeer [15] predicts future features with Taylor expansions; MagCache [21] models successive residual-output magnitude ratios to guide skip decisions. DiCache [1] uses shallow probes for deep-layer caching; MeanCache [2] constructs interval-average velocities from cached Jacobian–vector products. LayerCache [10] exploits layer-group velocity heterogeneity; SpectralCache [9] characterizes nonuniformity across time, depth, and features. LinCa [16] learns invertible feature decomposition, assigning different temporal rules to different components. This work focuses on target output dependency and accumulated state displacement, jointly designing cache execution and deferred feedback.

Error feedback and state correction. FoCa [33] views feature prediction as ODE approximation, combining prediction and correction steps. RACER [12] uses forecast disagreement to shrink unreliable predictions and reallocate refresh steps. AlignCache [5] couples CFG-output counterfactual calibration with momentum-based cached drafts. GOC [23] uses history differences and gradient directions to compensate representations. DTC focuses on sampling state: approximating defect integrals over executed cache intervals and applying opposite displacement.

Editing control. Qwen-Image [29] achieves text rendering and precise editing through multimodal attention. EditID [8] decouples extraction, fusion, integration of ID features. Dual-channel guidance [7] operates Key and Value spaces simultaneously. AdaEdit [6] combines progressive injection with channel-selective latent perturbation. PhysEdit [11] adaptively allocates inference depth and spatial scope by editing complexity.

3. Method

DACE prunes cache computation by target output and manages history by future reads; DTC uses scheduled full endpoints to measure output defects, estimates interval integrals, and corrects sampling state. Figure 1 presents the joint mechanism; Algorithm 1 specifies execution order.

3.1. Preliminaries

Let $i = 0, \ldots, T-1$ denote execution order, with decreasing noise schedule σ_i and $\Delta\sigma_i = \sigma_{i+1} - \sigma_i < 0$. We use Euler updates:

$$x_{i+1} = x_i + \Delta\sigma_i v_i. \tag{1}$$

The sampler output v_i includes CFG and norm rescaling. Given conditional and unconditional outputs c_i, u_i and guidance strength $g = 4$,

$$q_i = u_i + g(c_i - u_i), \qquad v_i = q_i \frac{\|c_i\|_2}{\|q_i\|_2}. \tag{2}$$

Let $a < b$ be adjacent full steps; cache steps $a < i < b$ produce $\widehat{v}_i$ based on old history $\mathcal{H}_a$. The local defect at current state is

$$d_i = \widehat{v}_i - v_\theta(x_i, \sigma_i). \tag{3}$$

The norms in Eq. (2) are computed over channels independently for each token. Equation (1) writes $\Delta\sigma_i d_i$ into the sampling state. Refreshing history at b updates future predictions while leaving this accumulated displacement in the state. DTC uses the scheduled full output to estimate and correct that displacement (Figure 1).

3.2. Dependency-Aware Cache Execution

Given a fixed full-step schedule $\mathcal{A} \supseteq \mathcal{W}$ and a staged predictor, DACE determines which target features and history items to compute and retain.

Interface assumptions. Let the cache system contain a predictor $P_s(\mathcal{H}, i)$ and output projection π_{out}. The full DiT outputs joint features $\mathbf{f} \in \mathbb{R}^{(|\mathcal{T}|+|\mathcal{C}|)\times d}$, where $\mathcal{T}$ and $\mathcal{C}$ index target and condition tokens. Output projection $\pi_{\text{out}} : \mathbb{R}^d \to \mathbb{R}^{d_v}$ acts independently per token, forming $\mathbf{z}_i = \pi_{\text{out}}(\mathbf{f})$. Pruning is valid when: (i) decomposition and prediction P_s are token-independent; (ii) π_{out} acts independently per token; (iii) only target output $\mathbf{z}_i[\mathcal{T}]$ is consumed, followed by CFG (Eq. (2)).

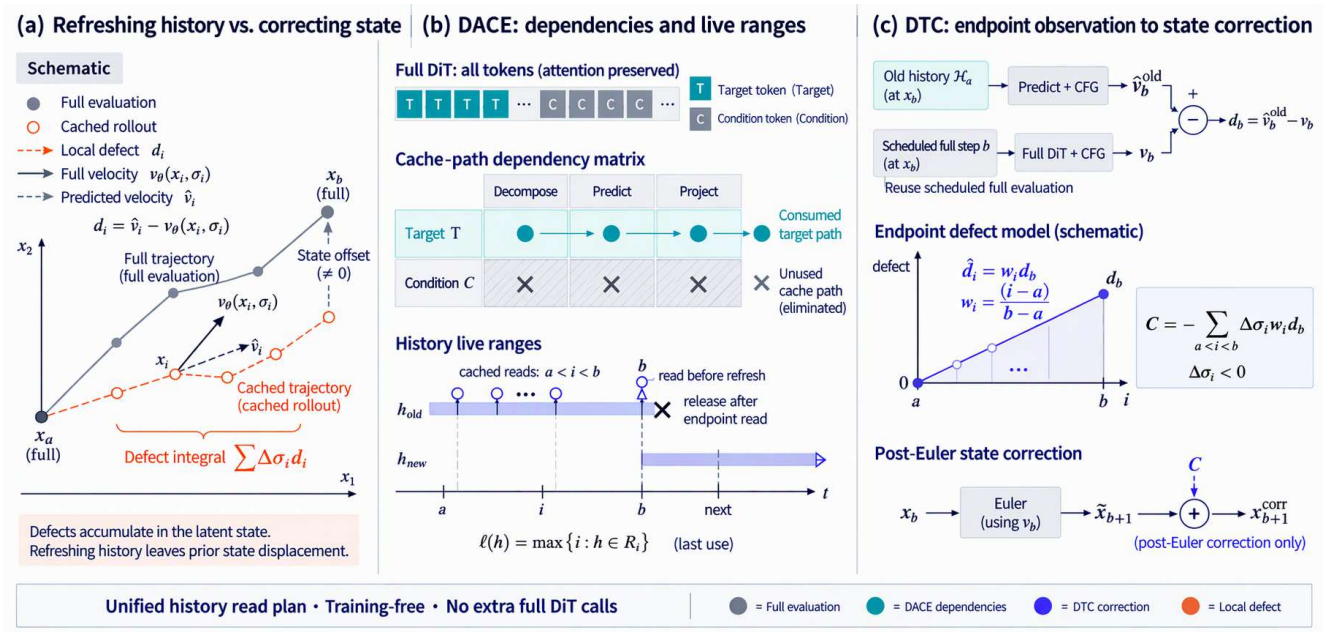


Figure 1. Mechanism overview. Trajectories and defect curves are schematic. (a) Local cache defects are written into the state by the sampler; history refresh updates the predictor while accumulated state displacement persists; the defect integral is one component of trajectory error. (b) DACE prunes cache decomposition, prediction, projection by target output dependency; full DiT retains all tokens; old history remains live until endpoint feedback read before release. (c) At full step b, two output paths separately complete CFG then measure d_b; endpoint model approximates interval defects and performs signed integration; the normal Euler update is followed by correction C. $\mathcal{T}/\mathcal{C}$ denote target/condition tokens. Cross-stage processing is separate; last unclosed tail is uncorrected.

When these hold, cache steps compute only target token features and execute $\pi_{\text{out}}(\cdot)[\mathcal{T}]$, bypassing condition token decomposition, prediction, projection. Full DiT attention processes all tokens; DACE pruning applies to cache paths.

Execution. At full steps $i \in \mathcal{A}$, construct decomposition history only for target tokens and decide which stage items to generate based on future reads. Let R_i denote history items read at step i; item h includes anchor step and stage identifier. For items with read demand,

$$\ell(h) = \max\{i : h \in R_i\}. \tag{4}$$

At full step i, release items satisfying $\ell(h) < i$ and generate only stages satisfying $\ell(h) \geq i$. At cache steps, the predictor outputs target features and applies $\pi_{\text{out}}(\cdot)[\mathcal{T}]$, skipping unused preprocessing.

DTC's endpoint counterfactual prediction requires reading $\mathcal{H}_a$ before refresh at b. This read is counted in R_b, extending old history liveness until feedback completion, ensuring consistency between execution pruning and feedback-required data. This design preserves model weights, prediction rules, and full-step positions. Changing matrix shapes can introduce finite-precision differences, which are measured in the closed-loop ablation. CFG velocity field $P(\mathcal{H}, b)$ in Eq. (5) includes branch prediction P_s, output projection, CFG composition.

3.3. Endpoint Counterfactual Measurement

At b, predict from the old history before overwriting it:

$$\widehat{v}_b^{\text{old}} = P(\mathcal{H}_a, b), \qquad d_b = \widehat{v}_b^{\text{old}} - v_\theta(x_b, \sigma_b). \tag{5}$$

The full output $v_\theta(x_b, \sigma_b)$ is already scheduled; the added cost is an endpoint prediction from the old cache. Both outputs are evaluated at the current cached state x_b. The Base reference follows its own full-computation trajectory.

The counterfactual pair $(\widehat{c}_b, \widehat{u}_b)$ and full pair (c_b, u_b) separately pass through Eq. (2); we then subtract their final outputs. Because norm rescaling is nonlinear, taking branch differences before CFG generally differs from taking final output difference. The defect is thus measured in the sampler's output space.

3.4. Sampler-Weighted Trajectory Correction

The true cache-step defects d_i are unavailable online. Taking a as a zero-defect anchor, we approximate them with an age-weighted endpoint observation:

$$\widehat{d}_i = w_{i;a,b} d_b, \qquad w_{i;a,b} = \frac{i-a}{b-a}. \tag{6}$$

This is a first-order endpoint model of the unknown defect sequence. Its correction is

$$C_{a,b} = -\sum_{i=a+1}^{b-1} \Delta\sigma_i w_{i;a,b} d_b. \tag{7}$$

The signed step sizes preserve sampling direction: for a coordinate with $d_b > 0$, $\Delta\sigma_i < 0$ makes the estimated displacement negative and its correction positive.

Full step b first executes normal update on x_b, then applies correction:

$$x_{b+1}^{\text{corr}} = x_b + \Delta\sigma_b v_\theta(x_b, \sigma_b) + C_{a,b}. \tag{8}$$

The full output is evaluated at x_b, and the correction is applied after the Euler update. Subsequent steps use x_{b+1}^{corr}. The gain is fixed at 1 and model weights remain frozen. The final unclosed cache interval receives no correction.

3.5. Error Propagation Analysis

On a specified state sequence, write $d_i = w_{i;a,b} d_b + r_i$. Under the Euler model in Eq. (1),

$$\sum_{i=a+1}^{b-1} \Delta\sigma_i d_i + C_{a,b} = \sum_{i=a+1}^{b-1} \Delta\sigma_i r_i. \tag{9}$$

DTC cancels the integral component explained by the endpoint model, leaving the integrated residuals. State perturbations also change subsequent velocity evaluations.

Let y_i be the matched Base trajectory and $\delta_i = x_i - y_i$. Starting after step a and its correction, the first cached transition is $x_{a+1} \to x_{a+2}$. Through the correction at b,

$$\begin{aligned}\delta_{b+1}^{\text{corr}} = \delta_{a+1} &+ \sum_{j=a+1}^{b} \Delta\sigma_j \big[v_\theta(x_j, \sigma_j) - v_\theta(y_j, \sigma_j)\big] \\ &+ \sum_{i=a+1}^{b-1} \Delta\sigma_i r_i.\end{aligned} \tag{10}$$

The second term captures the velocity-field response to differing states; the third captures defects unexplained by the endpoint model. Even when $r_i = 0$, state-dependent responses can remain. Fixed-trajectory diagnostics isolate endpoint approximation, whereas closed-loop experiments also include cumulative state perturbations.

3.6. Stage-Aware Joint Execution

For a predictor with stage index $s(i)$, partition a cross-stage interval into $I_s = \{i \in (a, b) : s(i) = s\}$. Each stage uses its own old history to form an endpoint defect $d_b^{(s)}$ after CFG:

$$C_{a,b} = -\sum_s \left(\sum_{i \in I_s} \Delta\sigma_i w_{i;a,b} \right) d_b^{(s)}. \tag{11}$$

Algorithm 1 Joint inference with DACE and DTC

Require: Initial state x_0, noise schedule $\{\sigma_i\}_{i=0}^{T}$
Require: Full step set $\mathcal{A} \supseteq \mathcal{W}$, read dependency plan

$a \leftarrow -1, \quad \mathcal{I} \leftarrow \varnothing, \quad \mathcal{H} \leftarrow \varnothing$
for $i = 0, \dots, T-1$ **do**
  if $i \in \mathcal{A}$ **then**
    for $s \in \text{stages}(\mathcal{I})$ **do**
      $\widehat{v}_i^{(s)} \leftarrow \text{ForecastCFG}(\mathcal{H}, i, s)$
      $S_s \leftarrow \sum_{j \in I_s} \Delta\sigma_j (j - a)/(i - a)$
    end for
    $v_i, \mathcal{H} \leftarrow \text{FullCFGUpdate}(x_i, i)$
    $C \leftarrow -\sum_{s \in \text{stages}(\mathcal{I})} S_s (\widehat{v}_i^{(s)} - v_i)$
    $\widetilde{x}_{i+1} \leftarrow x_i + \Delta\sigma_i v_i$
    $x_{i+1} \leftarrow \widetilde{x}_{i+1} + C$
    $a \leftarrow i, \quad \mathcal{I} \leftarrow \varnothing$
  else
    $\widehat{v}_i \leftarrow \text{CachedCFG}(\mathcal{H}, i, s(i))$
    $x_{i+1} \leftarrow x_i + \Delta\sigma_i \widehat{v}_i$
    $\mathcal{I} \leftarrow \mathcal{I} \cup \{i\}$
  end if
end for
return x_T

Figure 2 shows the N7 interval $(23, 30)$: step 24 belongs to stage 0 and steps 25–29 to stage 1. Both groups retain the shared anchor age $(i - 23)/7$, accumulate separate S_s, and measure their defects at the same endpoint.

Accumulating $S_s = \sum_{i \in I_s} \Delta\sigma_i w_{i;a,b}$ gives $C = -\sum_s S_s d_b^{(s)}$ at the endpoint, avoiding storage of per-step outputs. The implementation retains step indices, sizes, and stage metadata for integral checks; their costs are included in the measurements.

In Algorithm 1, $\mathcal{I}$ stores the current interval's cache steps and $I_s = \{j \in \mathcal{I} : s(j) = s\}$. Empty sums are zero. Endpoint prediction precedes the history refresh in FullCFGUpdate; both prediction paths include output projection and CFG. The update constructs only future-needed target decompositions and releases items with $\ell(h) < i$.

4. Experiments

Experiments compare external methods in closed-loop generation with matched inputs, prompts, seeds, then quantify incremental effects through LinCa, DACE, DACE+DTC ablation. Fixed-trajectory diagnostics analyze endpoint signals and weights; interval contrasts examine time–quality tradeoffs.

4.1. Setup

We evaluate Qwen-Image-Edit [29] using LinCa's pretrained predictor [16], with both sets of weights fixed. Euler sampling runs for 50 steps at 1024×1024 resolution with CFG strength 4. The predictor decomposes 3072-dimensional features into components of 2048, 512, and 512 dimensions,

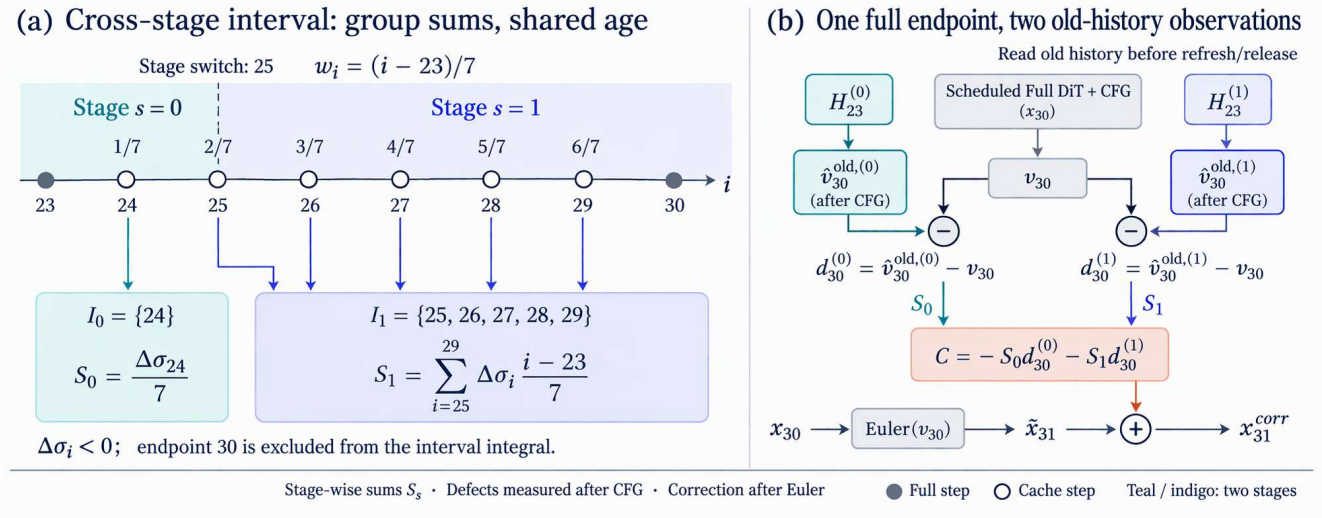


Figure 2. Cross-stage correction. N7 interval with stage switch at step 25: (a) Cache steps grouped by stage; age weights relative to anchors 23, 30; (b) Two old-history paths separately predict endpoint, compare with scheduled full CFG output; signed scalars S_s compose correction. Old history read first, then refreshed; correction after normal Euler update.

Table 1. External methods under matched budget. All use 18 CFG-branch full recomputations; cache methods share N7 positions. Text conditions are pre-encoded; timing covers VAE encoding, sampling, and decoding. TaylorSeer uses first-order prediction in this implementation. Eight-image means; memory: mean peak allocated per image. Bold: best estimates; shading: DACE+DTC.

Method	Full calls	Time/s↓	LPIPS↓	PSNR/dB↑	SSIM↑	Memory/GiB↓
Euler (9 steps)	18	**21.504**	0.3236	17.508	0.6920	**43.477**
FORA	18	28.244	0.2124	21.718	0.6895	56.624
TaylorSeer	18	30.062	0.7323	9.790	0.4656	69.770
LinCa	18	29.984	0.1895	22.965	0.8219	45.729
DACE+DTC	18	26.270	**0.1537**	**23.957**	**0.8418**	43.670

switching stages at step 25. We use an NVIDIA A800-SXM4-80GB with PyTorch 2.8.0 and Diffusers 0.36.0. The initialization set is $\mathcal{W} = \{0, 1, 2\}$; N7 then schedules a full step every 7 steps from step 2.

The Base configuration executes all 50 steps. LinCa applies the published full-token cache. DACE implements target prediction, computation pruning, and history management; DACE+DTC combines this execution layer with trajectory correction. All use identical predictor weights and schedules. N7+DTC, N9+DTC refer to DACE+DTC under respective schedules.

Table 1 compares Euler 9-step sampling, FORA [25], and TaylorSeer [15]. FORA reuses attention and MLP outputs, TaylorSeer performs first-order extrapolation, and both retain modulation, gating, and residual connections. All cache methods use N7 positions $[0, 1, 2, 9, 16, 23, 30, 37, 44]$, totaling 18 CFG-branch full calls; Euler 9 has same budget. For external contrasts, text conditions are encoded during warmup and the encoder moved to CPU, while DiT and VAE remain GPU-resident; timing covers VAE encoding, sampling, and decoding. Module ablation keeps the full pipeline GPU-resident. Both timing protocols are reported separately.

The development set contains three editing cases from two images (city background replacement, hydrant color change, and forest background replacement) for weight screening. Offline diagnostics sample 256 of 4096 target tokens along Base trajectories. The confirmation cohort uses background replacements on img_3 through img_10 with seeds 1003–1010, disjoint from the development inputs. Formulas, weights, and gain were fixed before confirmation. The 40 additional external-comparison runs reuse these eight confirmation inputs. We measure LPIPS-AlexNet v0.1 [30], PSNR, SSIM [27], and latent RMSE against matched Base outputs, with one timing run per image and configuration. Paired bootstrap intervals resample images 10,000 times with seed 20261110.

4.2. External Method Comparison

In Table 1, DACE+DTC achieves LPIPS 0.1537, PSNR 23.957 dB, and SSIM 0.8418: the lowest LPIPS and highest PSNR and SSIM in this comparison. Relative to LinCa, LPIPS decreases by 18.88% and latency falls from 29.984s

Table 2. Module ablation under fixed N7. Speedup is mean Base time divided by the current mean time. Fidelity metrics use the matched Base reference; PSNR is in dB. Memory is the mean peak allocation per image. Bold marks the best estimates among cache configurations.

Method	Time/s↓	Speedup/×	LPIPS↓	PSNR↑	SSIM↑	latent RMSE↓	Memory/GiB↓
Base	117.291	1.00	0.0000	–	1.0000	0.0000	58.968
LinCa	30.270	3.87	0.1895	22.965	0.8219	0.1485	61.219
DACE	**26.061**	**4.50**	0.1888	22.980	0.8220	0.1486	**59.116**
DACE+DTC	26.605	4.41	**0.1537**	**23.957**	**0.8418**	**0.1252**	59.158

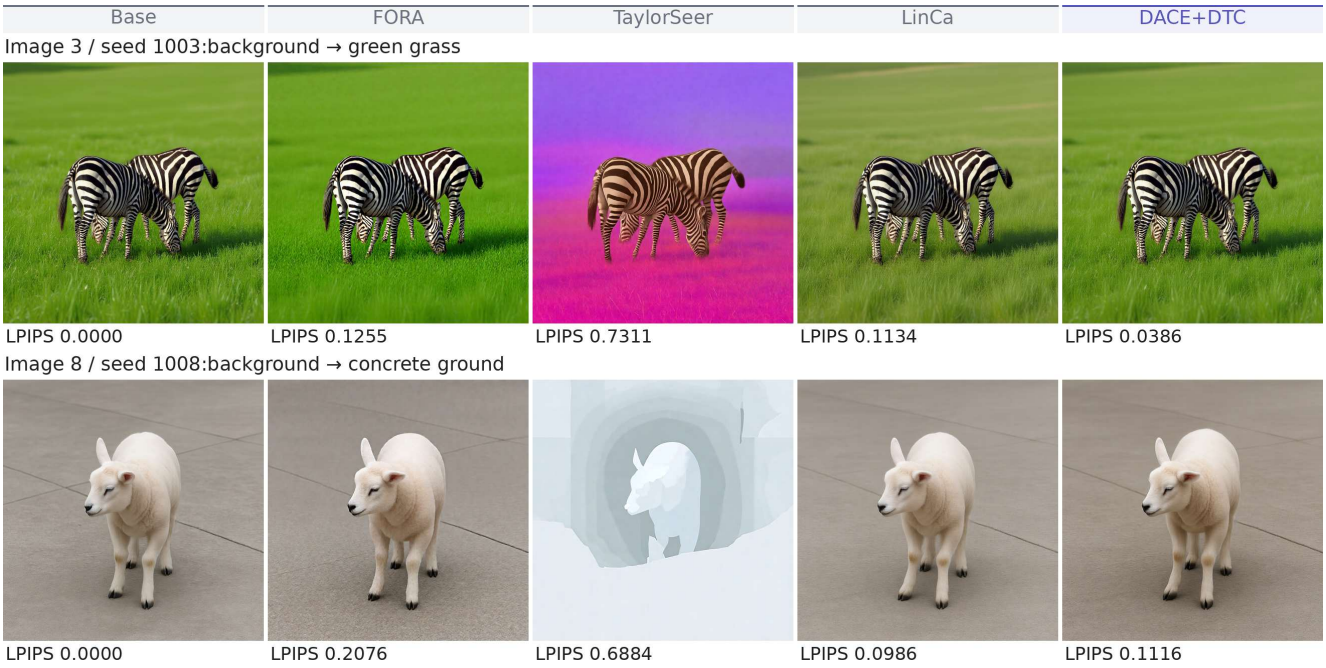


Figure 3. Output comparison. Top: first confirmation; bottom: greatest N7 PSNR degradation. Values: full-image LPIPS; Base is reference. Saved inference results.

to 26.270s (12.39%); relative to FORA, LPIPS and latency decrease by 27.66% and 6.99%. Mean peak memory is 43.670 GiB, below the three cache comparators.

At the same 18-call budget, Euler 9-step sampling has the shortest latency and LPIPS 0.3236. FORA and first-order TaylorSeer retain per-layer modulation, gating, and residual operations, taking 28.244s and 30.062s, respectively. First-order TaylorSeer has larger reference errors under this fixed schedule. These measurements characterize the implemented interfaces and fixed schedules. Table 2 separately reports full-pipeline ablations.

4.3. Main Results and Module Ablation

Table 2 shows that the full N7 method reduces latency from LinCa's 30.270s to 26.605s (12.11%) and LPIPS from 0.1895 to 0.1537 (18.88%). Relative to the 117.291s Base, the overall speedup is 4.41×, combining the reused predictor with DACE and DTC.

DACE reduces latency from 30.270s to 26.061s (13.91%). LPIPS changes from 0.1895 to 0.1888 (paired 95% interval $[-0.0021, +0.0005]$). Peak memory falls from 61.219 to 59.116 GiB through target-only decomposition and history release.

Adding DTC to DACE costs 0.545s (2.09%) while reducing LPIPS to 0.1537 (18.60%). Latent RMSE falls from 0.1486 to 0.1252 and SSIM rises from 0.8220 to 0.8418. Together, the two components reduce both latency and LPIPS relative to LinCa under the same schedule.

The paired LPIPS difference is -0.0358 versus LinCa (95% interval $[-0.0580, -0.0136]$) and -0.0351 versus DACE ($[-0.0574, -0.0129]$). The PSNR difference versus LinCa is +0.992 dB with interval $[-0.777, 2.718]$ dB. Table 5 reports the full paired statistics.

4.4. Qualitative Results

Figure 3 shows the first confirmation image and the case with the largest N7 PSNR decrease. In the grass scene, DACE+DTC produces tone and texture closer to Base, re-

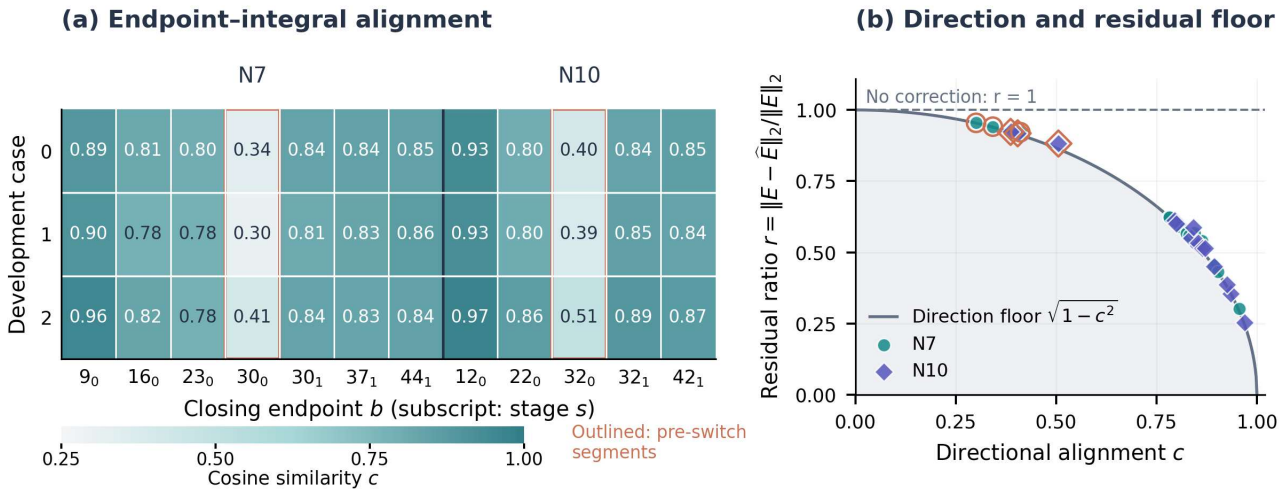


Figure 4. Directional information. Fixed Base trajectory: E actual integral, $\widehat{E}$ endpoint estimate. (a) 36 correlated subintervals from three development cases N7/N10; color/values $c = \cos(E, \widehat{E})$; borders mark pre-switch subsegments. (b) Residual ratio r vs. alignment; outlined markers correspond to outlined columns in (a). Gray: analytical lower bound $\sqrt{1-c^2}$ for scalar magnitude adjustment; gray region geometrically excluded.

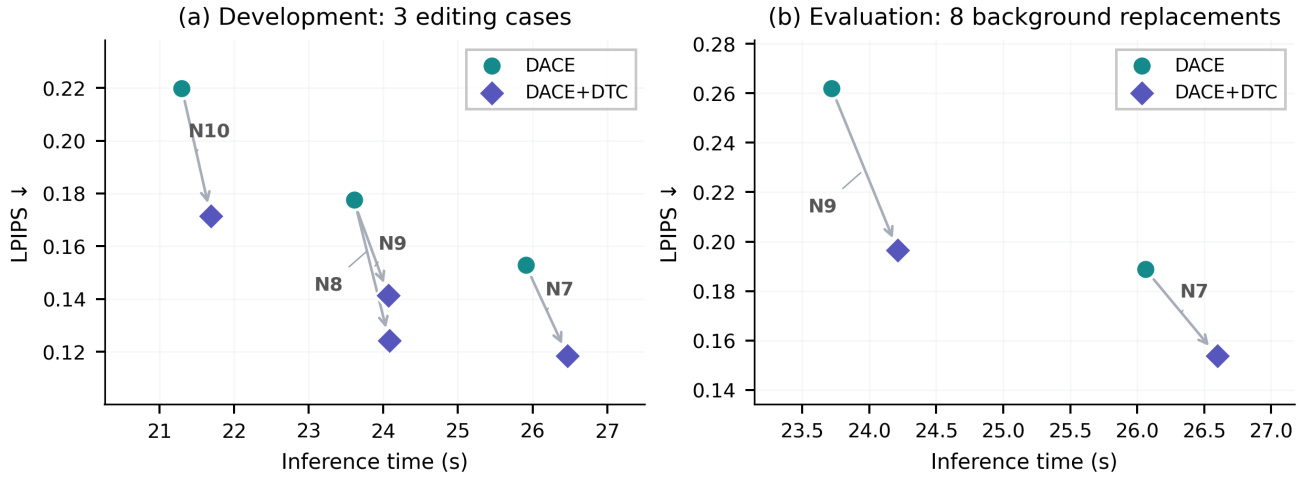


Figure 5. Time–quality tradeoff. Left: N7–N10 on three development cases. Right: N7, N9 on eight images. Each panel: cohort means.

ducing LPIPS from 0.1134 to 0.0386. In the concrete scene, seams differ and LPIPS rises from 0.0986 to 0.1116. FORA and first-order TaylorSeer outputs are included for comparison.

Figure 8 shows that LPIPS improves on six images and worsens on two, while PSNR improves on seven and worsens on one. Image 8 loses 3.937 dB, affecting the mean; image 4 exhibits opposite LPIPS and PSNR trends. The metrics respond differently to local structure, color, and alignment.

4.5. Endpoint Signal

Table 3 compares constant weights $w_i = 1$, sigma age $(\sigma_a - \sigma_i)/(\sigma_a - \sigma_b)$, and step age (Eq. (6)). We measure the L2 norm of cumulative displacement error on fixed Base trajectories and average over the three development cases.

Step-age weighting reduces N7 proxy error from 5.0720 to 3.9989 (21.16%) and N10 error from 6.9938 to 5.4221 (22.47%). Sigma-age weighting gives similar results; constant weighting yields a smaller N10 reduction (Figure 7).

In Figure 4, the actual integral E and endpoint estimate $\widehat{E}$ exhibit positive cosine similarity on all 36 subintervals, with mean 0.774, and correction reduces proxy error in each case.

The normalized residual $r = \|E - \widehat{E}\|_2 / \|E\|_2$ satisfies $r \geq \sqrt{1 - c^2}$ for $c = \cos(E, \widehat{E})$. Direction sets the residual floor attainable by scalar correction; magnitude controls the gap to that floor. Pre-switch segments have lower alignment. These subintervals are correlated within the three development trajectories.

4.6. Latency and Fidelity Tradeoff

Figure 5 compares cache intervals across schedules. On development data, DTC reduces LPIPS at N7 through N10; N9+DTC achieves 24.080s and LPIPS 0.1242, improving both measures relative to N7 DACE. On confirmation data, N9+DTC reduces latency from 26.061s to 24.213s (7.09%) relative to N7 DACE, while LPIPS rises from 0.1888 to 0.1964 and PSNR falls by 1.900 dB. Within the N9 schedule, DTC improves LPIPS from 0.2621 to 0.1964 (Table 6).

DTC preserves the N7 budget of 18 full DiT branch calls while increasing predictor inverse transforms from 82 to 96. Mean peak memory is 59.158 GiB, compared to 59.116 GiB for DACE and 61.219 GiB for LinCa. Appendix D details correction magnitudes, call counts, unclosed tails, and structural changes under longer intervals.

4.7. Discussion

DACE reduces execution overhead using output dependencies and future reads, while DTC corrects state displacement using endpoint feedback. Their shared dependency plan retains the history required by cached predictions and endpoint measurements. The endpoint–residual relationship also suggests jointly adapting feedback strength and cache intervals to the evolving trajectory.

5. Conclusion

We introduced dependency-aware cache execution and deferred trajectory correction, linked by a shared history read plan. DACE prunes token-independent cache computation and history storage; DTC converts scheduled endpoint observations into post-Euler state corrections. Both preserve model weights and the full DiT call budget. Under GPU-resident N7, the joint method reduces latency by 12.1% and mean LPIPS by 18.9% relative to LinCa; DTC further reduces LPIPS by 18.6% with 2.1% overhead on DACE. The analysis and closed-loop evaluations connect these gains to execution dependencies and accumulated state displacement.

References

[1] Jiazi Bu, Pengyang Ling, Yujie Zhou, et al. DiCache: Let Diffusion Model Determine Its Own Cache. arXiv:2508.17356, 2025.

[2] Huanlin Gao, Ping Chen, Fuyuan Shi, et al. MeanCache: From Instantaneous to Average Velocity for Accelerating Flow Matching Inference. arXiv:2601.19961, 2026.

[3] Jonathan Ho, Tim Salimans. Classifier-Free Diffusion Guidance. arXiv:2207.12598, 2022.

[4] Jonathan Ho, Ajay Jain, Pieter Abbeel. Denoising Diffusion Probabilistic Models. NeurIPS, pp. 6840–6851, 2020.

[5] Jiyue Jiang, Zhiyu Huo, Yuhan Chen, et al. Heuristics are Not Enough: Calibrated and Corrected Caching for Video Diffusion Acceleration. arXiv:2610.16545, 2026.

[6] Guandong Li, Zhaobin Chu. AdaEdit: Adaptive Temporal and Channel Modulation for Flow-Based Image Editing. arXiv:2603.21615, 2026.

[7] Guandong Li. Dual-Channel Attention Guidance for Training-Free Image Editing Control in Diffusion Transformers. arXiv:2602.18022, 2026.

[8] Guandong Li, Zhaobin Chu. EditID: Training-Free Editable ID Customization for Text-to-Image Generation. Findings of EMNLP, pp. 6301–6319, 2025.

[9] Guandong Li. Frequency-Aware Error-Bounded Caching for Accelerating Diffusion Transformers. arXiv:2603.05315, 2026.

[10] Guandong Li. LayerCache: Exploiting Layer-wise Velocity Heterogeneity for Efficient Flow Matching Inference. arXiv:2604.16492, 2026.

[11] Guandong Li, Mengxia Ye. PhysEdit: Physically-Consistent Region-Aware Image Editing via Adaptive Spatio-Temporal Reasoning. arXiv:2605.00707, 2026.

[12] Yanchao Li, Jiaqing Xie, Ben Gao, et al. Disagree to Accelerate: Closing the Loop on Diffusion Feature Forecasts. arXiv:2608.01740, 2026.

[13] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le. Flow Matching for Generative Modeling. arXiv:2210.02747, 2022.

[14] Feng Liu, Shiwei Zhang, Xiaofeng Wang, et al. Timestep Embedding Tells: It's Time to Cache for Video Diffusion Model. CVPR, pp. 7353–7363, 2025.

[15] Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, Linfeng Zhang. From Reusing to Forecasting: Accelerating Diffusion Models with TaylorSeers. ICCV, pp. 15853–15863, 2025.

[16] Jinshan Liu, Haoran Qin, Xiaobing Tu, et al. LinCa: Accelerating Diffusion Models via Learnable Decomposed Feature Caching. arXiv:2608.17973, 2026.

[17] Xingchao Liu, Chengyue Gong, Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. arXiv:2209.03003, 2022.

[18] Cheng Lu, Yuhao Zhou, Fan Bao, et al. DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps. arXiv:2206.00927, 2022.

[19] Zhengyao Lv, Chenyang Si, Junhao Song, et al. FasterCache: Training-Free Video Diffusion Model Acceleration with High Quality. arXiv:2410.19355, 2024.

[20] Xinyin Ma, Gongfan Fang, Xinchao Wang. DeepCache: Accelerating Diffusion Models for Free. CVPR, pp. 15762–15772, 2024.

[21] Zehong Ma, Longhui Wei, Feng Wang, Shiliang Zhang, Qi Tian. MagCache: Fast Video Generation with Magnitude-Aware Cache. arXiv:2506.09045, 2025.

[22] William Peebles, Saining Xie. Scalable Diffusion Models with Transformers. ICCV, pp. 4195–4205, 2023.

[23] Junxiang Qiu, Lin Liu, Shuo Wang, et al. Accelerating Diffusion Transformer via Gradient-Optimized Cache. arXiv:2503.05156, 2025.

[24] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. CVPR, pp. 10684–10695, 2022.

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

[26] Jiaming Song, Chenlin Meng, Stefano Ermon. Denoising Diffusion Implicit Models. arXiv:2010.02502, 2020.

[27] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, Eero P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. IEEE Transactions on Image Processing, 13(4):600–612, 2004.

[28] Felix Wimbauer, Bichen Wu, Edgar Schoenfeld, et al. Cache Me if You Can: Accelerating Diffusion Models through Block Caching. CVPR, pp. 6211–6220, 2024.

[29] Chenfei Wu, Jiahao Li, Jingren Zhou, et al. Qwen-Image Technical Report. arXiv:2508.02324, 2025.

[30] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. CVPR, pp. 586–595, 2018.

[31] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, Jiwen Lu. UniPC: A Unified Predictor-Corrector Framework for Fast Sampling of Diffusion Models. arXiv:2302.04867, 2023.

[32] Xuanlei Zhao, Xiaolong Jin, Kai Wang, Yang You. Real-Time Video Generation with Pyramid Attention Broadcast. arXiv:2408.12588, 2024.

[33] Shikang Zheng, Liang Feng, Xinyu Wang, et al. Forecast then Calibrate: Feature Caching as ODE for Efficient Diffusion Transformers. arXiv:2508.16211, 2025.

[34] Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, Linfeng Zhang. Accelerating Diffusion Transformers with Token-wise Feature Caching. arXiv:2410.05317, 2024.

A. Conditional Analysis and Derivations

A.1. Defect Integral on a Fixed State Sequence

For $a < i < b$, write local defect as $d_i = w_{i;a,b} d_b + r_i$; direct expansion yields

$$\sum_i \Delta\sigma_i d_i + C_{a,b} = \sum_i \Delta\sigma_i (w_{i;a,b} d_b + r_i) - \sum_i \Delta\sigma_i w_{i;a,b} d_b = \sum_i \Delta\sigma_i r_i,$$

where all sums range $a+1$ to $b-1$. By triangle inequality, remaining integral norm does not exceed $\sum_i |\Delta\sigma_i| \|r_i\|$. If $r_i = 0$, the linear defect part on this fixed state sequence cancels exactly. This identity describes cancellation on the specified state sequence; Eq. (10) additionally includes the velocity-field response to differing states.

A.2. Endpoint Direction and Scalar Correction Geometry

For any stage subinterval I_s, let $E = \sum_{i \in I_s} \Delta\sigma_i d_i$, $\widehat{E} = S_s d_b^{(s)}$. For nonzero $E, \widehat{E}$ define $c = \langle E, \widehat{E} \rangle / (\|E\|_2 \|\widehat{E}\|_2)$, $\eta = \|\widehat{E}\|_2 / \|E\|_2$. Directly expanding squared norm yields

$$r^2 = \frac{\|E - \widehat{E}\|_2^2}{\|E\|_2^2} = 1 + \eta^2 - 2\eta c = (\eta - c)^2 + 1 - c^2. \tag{12}$$

When $c \geq 0$, adjusting nonnegative magnitude along the same estimated direction, minimum residual ratio is $\sqrt{1 - c^2}$, attained at $\eta = c$. This lower bound is determined by direction; integral components orthogonal to the estimated direction remain in the residual. Figure 4 uses actual correction magnitudes from frozen logs; gray line shows above analytical lower bound. Stage grouping follows established step-25 switch position.

A.3. Difference from Base Trajectory

Matched full trajectory satisfies $y_{j+1} = y_j + \Delta\sigma_j v_\theta(y_j, \sigma_j)$. Cache steps use $\widehat{v}_j = v_\theta(x_j, \sigma_j) + d_j$; full steps have $d_j = 0$. Starting from state after completing step a and its correction, with x_{a+1}, y_{a+1} as interval start, stepwise subtracting, accumulating, and adding $C_{a,b}$ after b's normal update yields Eq. (10). This interval start includes step a and its prior-interval correction.

If local Lipschitz condition holds in the comparison range

$$\|v_\theta(x_j, \sigma_j) - v_\theta(y_j, \sigma_j)\| \leq L_j \|\delta_j\|,$$

then

$$\|\delta_{b+1}^{\text{corr}}\| \leq \|\delta_{a+1}\| + \sum_{j=a+1}^{b} |\Delta\sigma_j| L_j \|\delta_j\| + \sum_{i=a+1}^{b-1} |\Delta\sigma_i| \|r_i\|. \tag{13}$$

This conditional bound separates endpoint-model residual and cross-state response, characterizing both error types' effect on trajectory difference. Multi-stage case decomposes and sums separately by I_s; conclusion form unchanged.

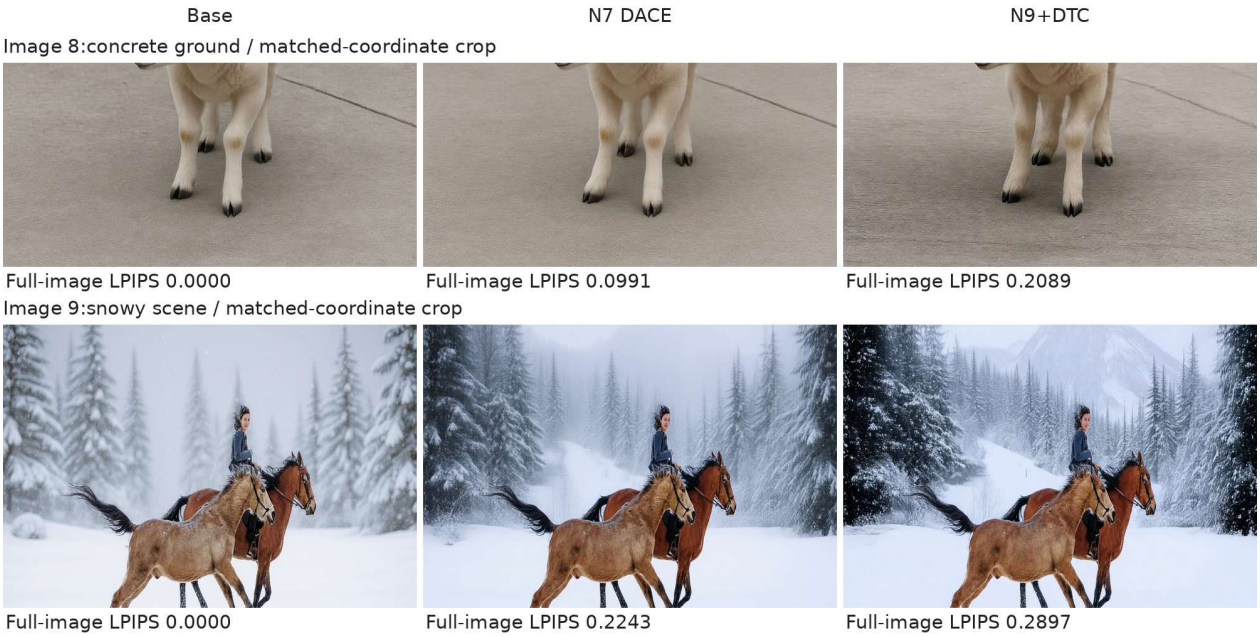


Figure 6. Local structure comparison under longer cache intervals. Three columns: Base, N7 DACE, N9+DTC. Top row: image 8 concrete ground, crop $(0, 520, 1024, 1024)$; bottom row: image 9 snowy background, crop $(0, 0, 1024, 700)$. Coordinates on original 1024-pixel image; marked LPIPS are full-image scores; same row uses same crop window.

B. Development Experiments and Full Paired Statistics

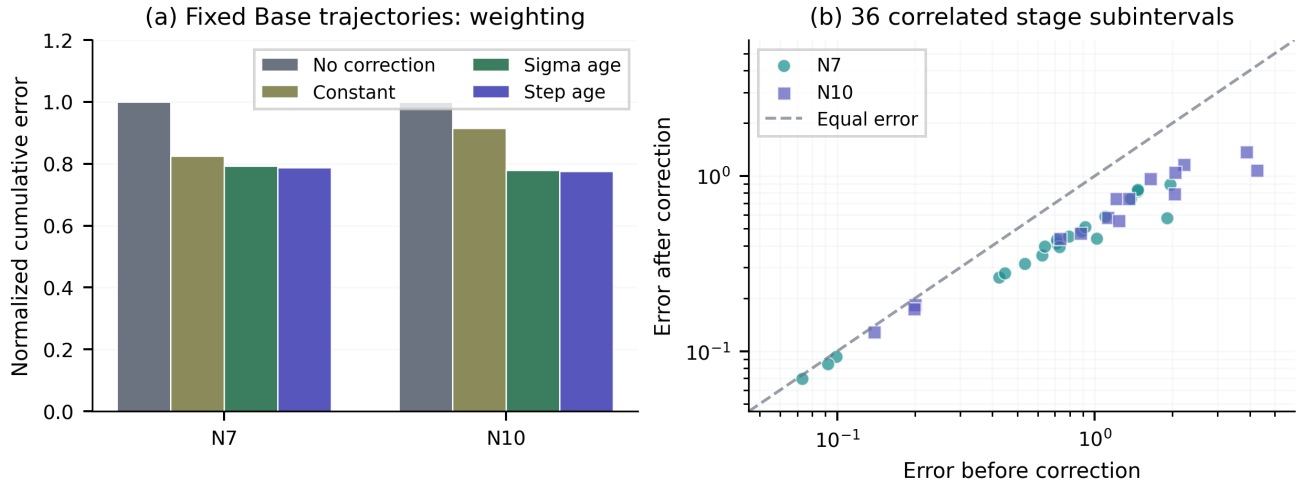


Figure 7. Offline mechanism diagnostics. Left: cumulative proxy errors for each weight relative to no-correction under same schedule. Right: pre- vs. post-correction errors for step age on 36 closable stage subintervals; dashed lines are isolines; points below indicate improvement. Subintervals from three development cases' fixed trajectories, mutually correlated.

Table 3. Offline weight contrasts. Values are cumulative displacement proxy error L2 on fixed Base trajectory, sampling 256 target tokens; lower is better; development diagnostics only.

Schedule	No correction	Constant	Sigma age	Step age	Reduction
N7	5.0720	4.1823	4.0213	3.9989	21.16%
N10	6.9938	6.3957	5.4448	5.4221	22.47%

Table 4. Full interval contrasts on three development cases; execution time and reference-fidelity metrics reported separately for each fixed schedule.

Development setting	Time/s	LPIPS	PSNR/dB	SSIM
N7, DACE	25.909	0.1528	22.745	0.7888
N7, DACE+DTC	26.471	0.1184	23.829	0.8271
N8, DACE	23.600	0.1775	21.938	0.7657
N8, DACE+DTC	24.070	0.1414	23.011	0.8031
N9, DACE	23.609	0.1777	21.557	0.7833
N9, DACE+DTC	24.080	0.1242	23.376	0.8166
N10, DACE	21.301	0.2199	20.335	0.7390
N10, DACE+DTC	21.689	0.1714	21.917	0.7632

Evaluation sequence. The predeclared comparison tested whether N9+DTC reduced both LPIPS and runtime relative to N7 DACE, using paired 95% intervals with upper bounds below zero. The LPIPS difference was $+0.00764$ with interval $[-0.02460, 0.04531]$: four images improved and four regressed. The subsequent LinCa N7 comparison reused this cohort for post-confirmation analysis of DACE and DTC contributions. N9, weights, and gain were selected on development data and fixed for confirmation.

Table 5. Paired LPIPS differences under fixed N7. Method before slash is evaluated; after slash is reference. LinCa is cited cache method; DACE+DTC is our full method. Negative values indicate improvement.

Comparison	LPIPS difference	95% interval	Wins/losses
DACE+DTC / LinCa	-0.0358	[-0.0580,-0.0136]	6/2
DACE / LinCa	-0.0007	[-0.0021,+0.0005]	3/5
DACE+DTC / DACE	-0.0351	[-0.0574,-0.0129]	6/2

Table 6. Confirmation results for N7 and N9 on eight images. N7+DTC timing comes from the confirmation run; the main-text table uses the later LinCa-comparison rerun. The final latents are elementwise identical across these runs.

Setting	Time/s↓	LPIPS↓	PSNR↑
N7, DACE	26.061	0.1888	22.980
N7, DACE+DTC	26.599	0.1537	23.957
N9, DACE	23.718	0.2621	19.081
N9, DACE+DTC	24.213	0.1964	21.080

Table 7. Per-image N7 results on all eight confirmation inputs. ΔPSNR is relative to LinCa, in dB.

Image	Seed	LinCa LPIPS	DACE+DTC LPIPS	ΔPSNR
3	1003	0.1134	0.0386	+5.768
4	1004	0.1951	0.1968	+0.378
5	1005	0.1926	0.1380	+1.812
6	1006	0.3528	0.2677	+0.822
7	1007	0.1155	0.0785	+0.675
8	1008	0.0986	0.1116	-3.937
9	1009	0.2240	0.2100	+1.607
10	1010	0.2235	0.1882	+0.812

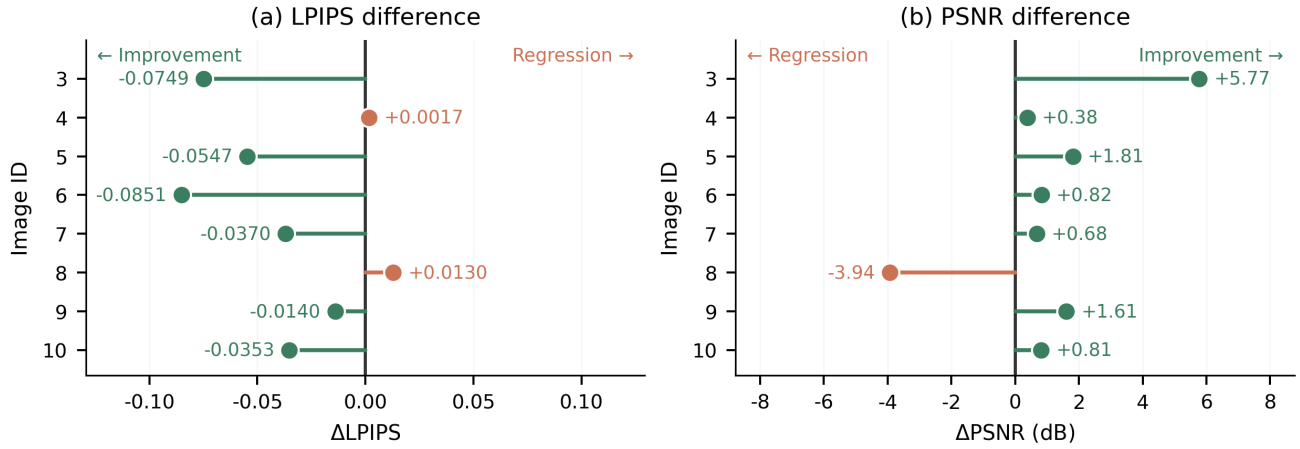


Figure 8. DACE+DTC differences relative to LinCa across all eight images. Left: negative LPIPS is improvement; right: positive PSNR is improvement. Image 4 has a small LPIPS increase alongside a PSNR increase.

Paired bootstrap resamples images; timepoints, tokens, and stage subintervals are nested within images. With one timing observation per image and configuration, the timing intervals quantify across-image variation under the single-run protocol. We report paired means and 95% intervals for LPIPS and PSNR.

C. Reproducibility Details

Paired inputs. The confirmation cohort uses images 3 through 10 with seeds 1003 through 1010. Target backgrounds are sequentially green grass, beach, forest, garden, forest, concrete ground, snowy scene, and mountains. Each comparison holds input scaling, prompt, negative prompt, seed, resolution, and CFG constant. The protocol records input SHA256 hashes. The LinCa comparison reuses the matched Base outputs, and N7+DTC produces the same final latents as in confirmation.

Weights and code. We use upstream commit `8d0df3e9683576121d1455e2d46b39ae9d048de4` and the pretrained predictor with SHA256 `baa13421e57da3176b5994a00deb95154ab397aaf516086c4623419eac42e970`. Generator and predictor weights remain fixed throughout our experiments. DACE+DTC uses step-age weighting and gain 1; sigma-age and constant weightings serve as offline development controls.

Implementation checks. The zero-gain control executes the endpoint measurement path and produces final latents and pixels elementwise identical to same-schedule DACE. N7 DACE reruns on the three development cases reproduce the original latents; N7+DTC reruns on the eight confirmation images reproduce the confirmation latents. Implementation checks cover correction sign, linear defect cancellation, stage switching, and unclosed-tail handling.

Figure construction. Qualitative panels use paired inference outputs with proportional scaling. Captions specify crop coordinates, and methods in the same row use identical windows. Correction magnitudes come from generation logs;

offline diagnostics follow fixed Base trajectories. The external comparison uses the same N7 DACE+DTC configuration as confirmation. The accompanying records contain per-image metrics, timing, memory, call counts, and source-image hashes.

C.1. Relation to Prior Work

Table 8. Mechanism comparison. Each row lists the processing target and its relationship to DTC.

Method	Original primary target	Relationship to this work
LinCa [16]	Learned-decomposition cache feature prediction	Predictor backend selected for experiments
FoCa [33]	Feature ODE prediction and calibration	Related prior work on prediction–correction
RACER [12]	Prediction divergence, reliability, refresh	Feedback control of cache prediction
AlignCache [5]	CFG counterfactual error, decision calibration, draft	Shares output-space counterfactual measurement
GOC [23]	History feature difference, cache representation	Shares general error-compensation goal
MeanCache [2]	JVP interval average velocity, stability schedule	Control local cumulative error via average velocity
LayerCache [10]	Layer-group velocity heterogeneity, budget allocation	Hierarchical compute reuse and schedule design
SpectralCache [9]	Time, depth, feature nonuniformity	Cache granularity and cumulative error budget
DTC (ours)	Defect integral over executed intervals, latent displacement	Deferred correction to sampling state after scheduled full steps

D. Correction Magnitude and Resource Accounting

We measure correction magnitude relative to the normal sampler displacement:

$$\rho_b = \frac{\|C_{a,b}\|_2}{\max(\|\widetilde{x}_{b+1} - x_b\|_2, 10^{-12})}. \tag{14}$$

Here $\widetilde{x}_{b+1}$ is the sampler output immediately before DTC, and its difference from x_b is the normal step displacement. Figure 9 shows that corrections at the first closed endpoint can exceed the normal displacement of that full step. N9 also exhibits larger late-trajectory correction ratios than N7.

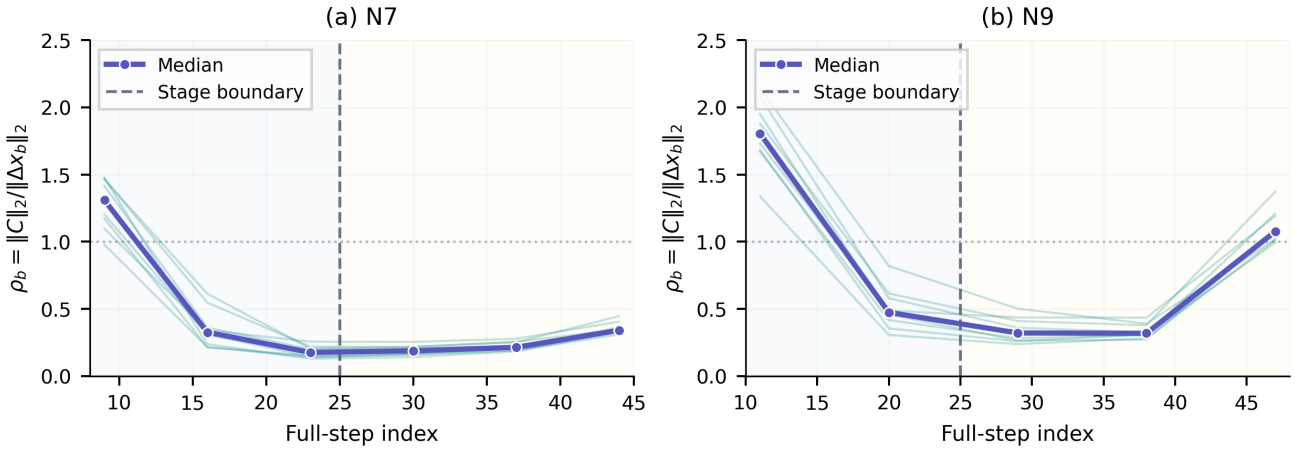


Figure 9. Correction magnitude on eight-image evaluation cohort. Faint lines: individual images; solid lines: cross-image median; dashed lines: stage boundary 25. Values computed from actual generation logs; N7 and N9 use same vertical axis.

Table 9 separately counts full DiT and predictor inverse transforms, summing both CFG branches. N7 maintains 18 full forward passes and increases inverse transforms from 82 to 96; N9 maintains 16 full forward passes and increases inverse transforms from 84 to 96. With zero gain, endpoint measurement produces final latents and pixels elementwise identical to DACE.

Table 9. Call budgets and unclosed tails. Forward and inverse-transform counts sum both CFG branches. N8/N10 are development configurations; N7/N9 also have confirmation data.

Schedule	Full steps	DiT calls	DACE inverses	DTC inverses	Tail steps
N7	9	18	82	96	5
N8	8	16	84	96	7
N9	8	16	84	96	2
N10	7	14	86	96	7

Let F denote the count of full timepoints and M the count of closable stage subintervals requiring measurement. The baseline performs $2(T - F)$ inverse transforms; DTC adds $2M$. The three-step initialization $[0, 1, 2]$ contains no cache steps internally, leaving $F - 3$ closable intervals afterward. All four schedules include exactly one cross-stage interval that splits into two subsegments, yielding $M = (F - 3) + 1 = F - 2$. The measured intervals contribute M endpoint observations, yielding $2(T - 2) = 96$ inverse transforms. Full forwards and endpoint predictions contribute separately to inference latency.

Under fixed N7, mean peak allocated memory increases from 59.116 GiB for DACE to 59.158 GiB for the full method (approximately 0.0425 GiB). DACE's target-token decomposition and history dependency management keep the full method below LinCa's 61.219 GiB. N7 leaves the last five cache steps uncorrected and N9 leaves two. The schedule affects end-to-end latency beyond the simple proportion of full calls.

Appendix Figure 6 examines images 8 and 9, which exhibit the greatest N9+DTC LPIPS degradation relative to N7 DACE. Image 8 has LPIPS difference $+0.10981$ and PSNR drop 6.386 dB; image 9 has $+0.06539$ and 3.103 dB. Concrete-ground

seams and snowy-scene tree layouts change noticeably under the longer cache intervals. Equation (10) separates the remaining difference into endpoint approximation residual and cross-state velocity-field response, isolating two error sources under longer intervals.

E. Complete N7 Confirmation Comparisons

The figures show all inputs and corresponding outputs by image number. Columns are input, Base, LinCa, DACE, and DACE+DTC; full-image LPIPS is marked below each cell. Base is the reference generation from the fixed model.

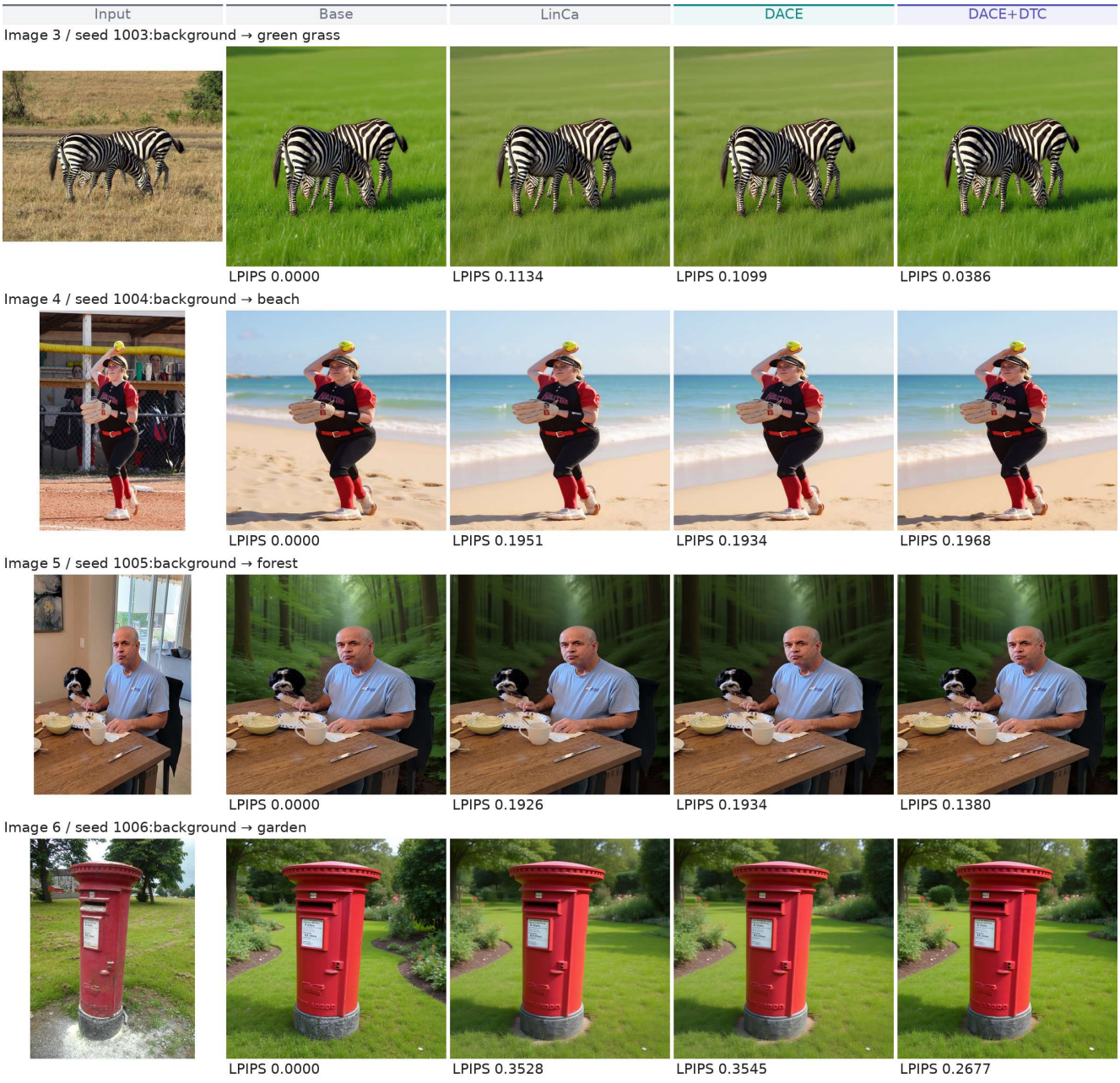


Figure 10. All N7 contrasts part one: images 3–6, seeds 1003–1006.

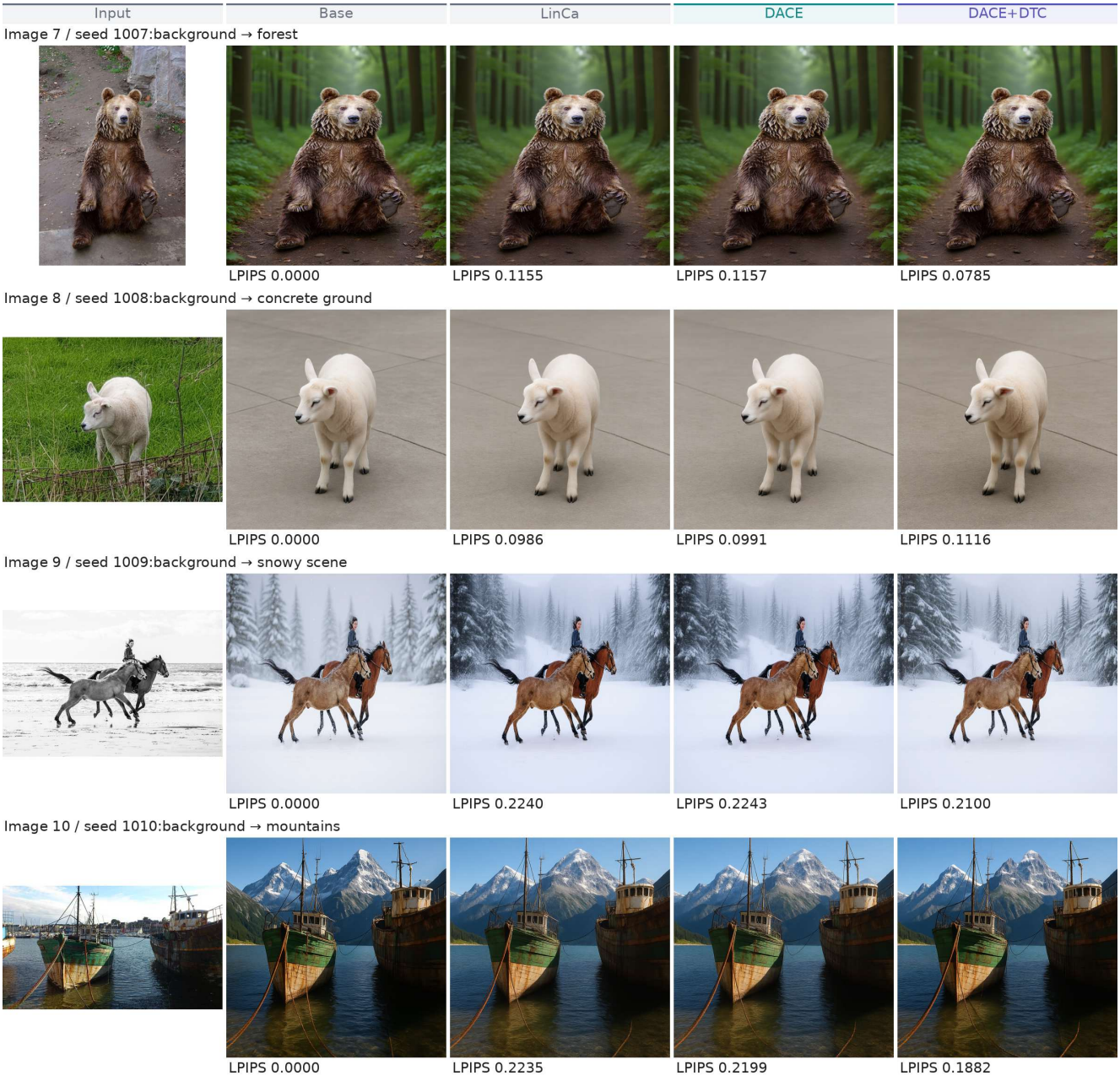


Figure 11. All N7 contrasts part two: images 7–10, seeds 1007–1010.

F. Complete N9 Confirmation Comparisons

The following figures use the same cohort, prompts, and seeds. Columns show Base, N7 DACE, N7+DTC, N9 DACE, and N9+DTC, presenting both within-schedule and cross-schedule contrasts.

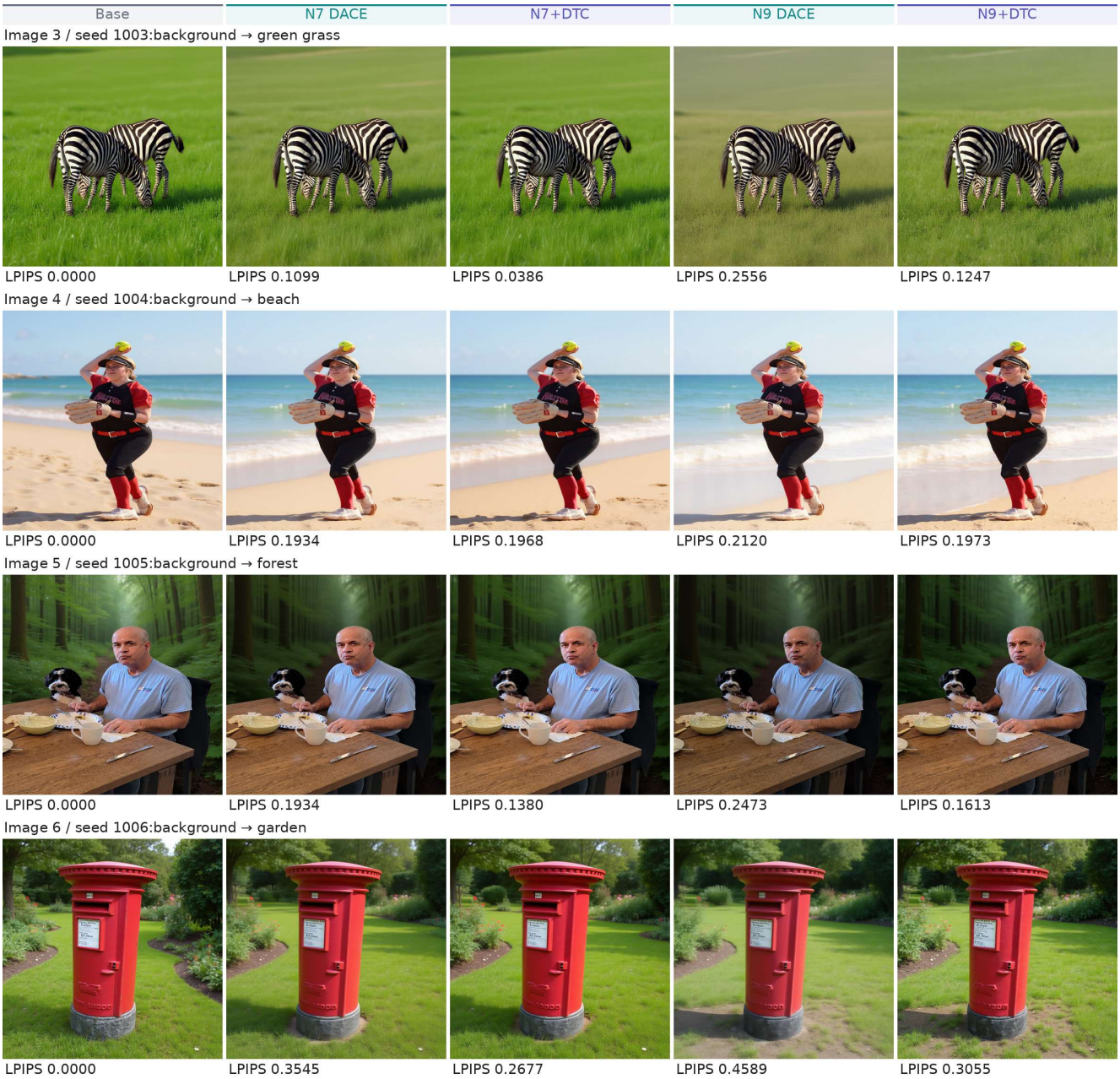


Figure 12. All N9 contrasts part one: images 3–6. Each cell's LPIPS compares to Base with same input and seed.

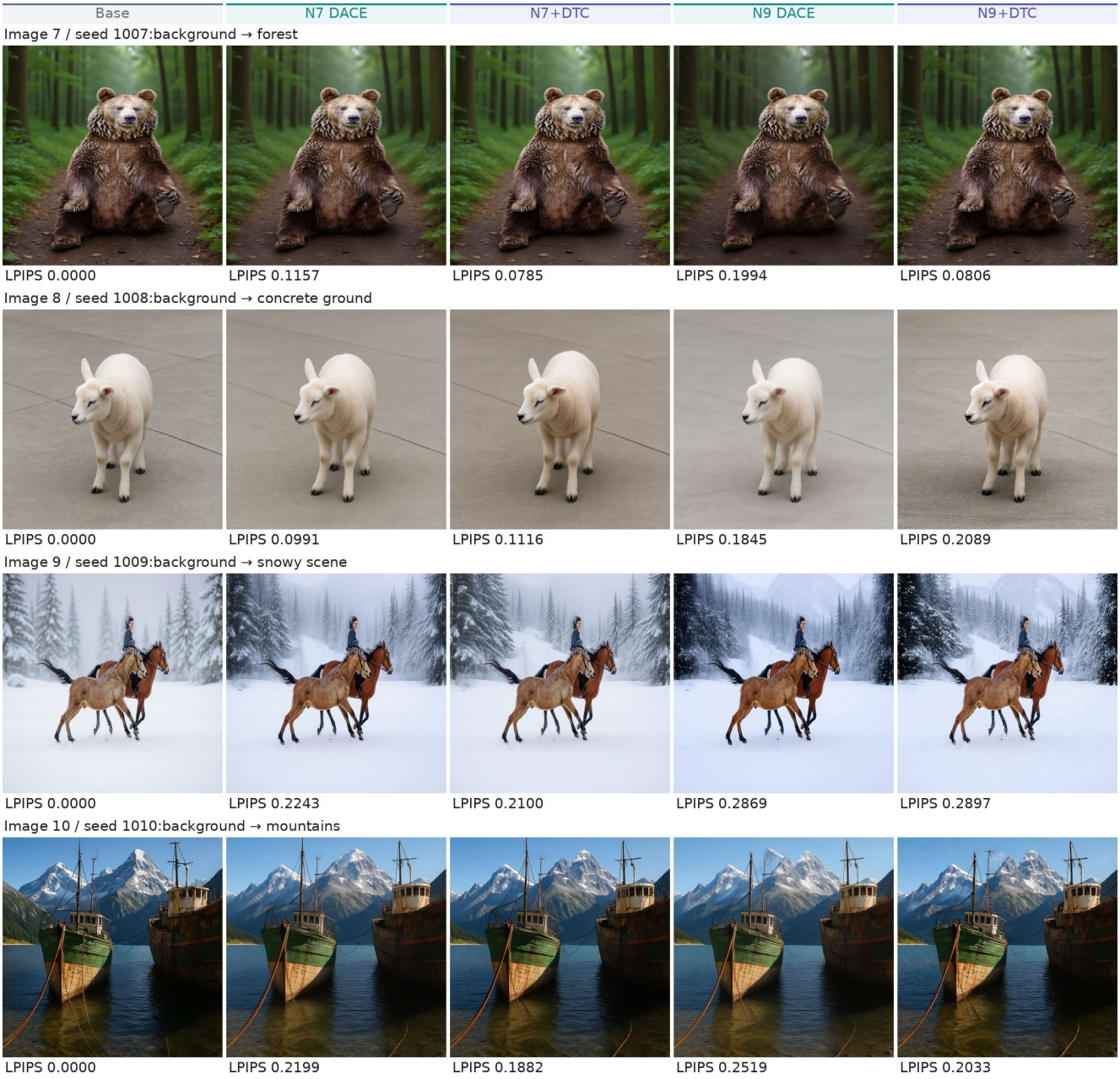


Figure 13. All N9 contrasts part two: images 7–10. Images 8, 9 are cases with greatest N9+DTC LPIPS degradation relative to N7 DACE; see Figure 6 for same-coordinate crops.

Statements and Declarations

Corresponding author. *Guandong Li, iFLYTEK, Hefei, Anhui, China. Email: leeguandon@gmail.com. Mengxia Ye is affiliated with Aegon THTF, Nanjing, Jiangsu, China.

Funding. This research received no specific funding.

Competing interests. The authors declare no competing interests relevant to this work.

Author contributions. Guandong Li: Methodology, experimental design and execution, and writing the original draft. Mengxia Ye: Result verification and manuscript review and editing.

Data availability. Per-image numerical results, sampling configurations, paired comparisons, and qualitative outputs are provided in the article and its supplementary material. The accompanying numerical records (Online Resource 1) contain the reported module-ablation and external-comparison results. The pretrained generator and predictor are identified and attributed in the article.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- DTCsupplementarydata.zip