DARTCACHE:DUAL-STAGE TEMPORAL REUSE PLANNING FOR EFFICIENT INVERSION-BASED EDITING

Anonymous authors
Paper under double-blind review

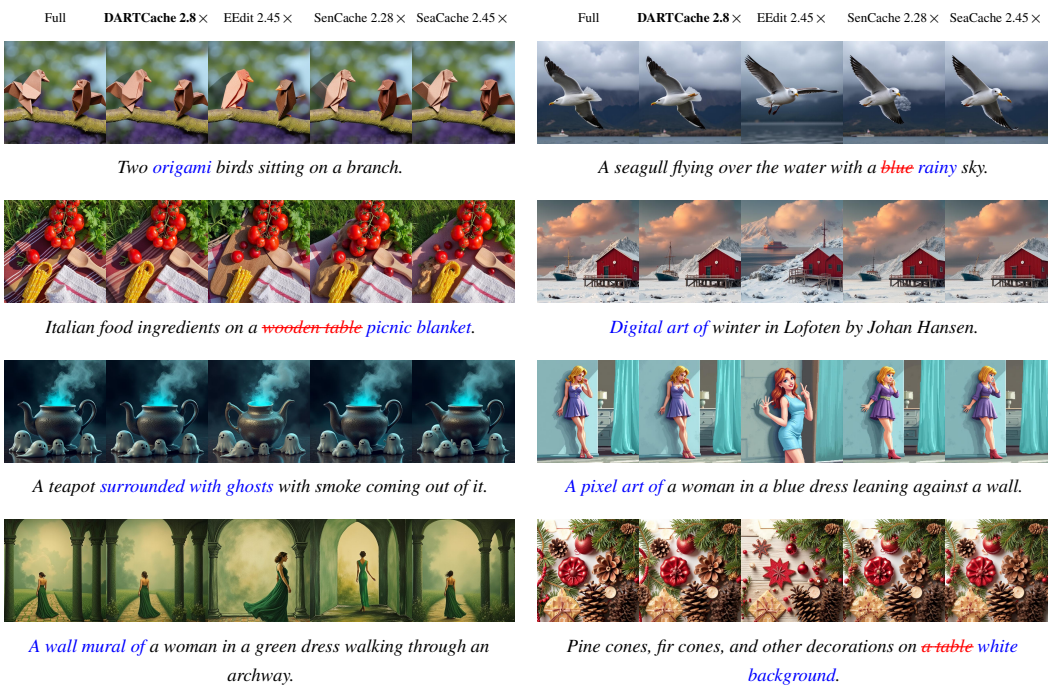


Figure 1: **Qualitative comparison.** DARTCache consistently preserves full-inference editing results across diverse scenarios.

ABSTRACT

Inversion-based image editing requires repeated model evaluations during both inversion and denoising, making inference computationally expensive. Training-free caching can reduce this cost, but applying caching policies independently to the two stages leaves a shared-budget planning problem unresolved: where to refresh predictions within each stage and how to allocate computation between stages. We propose **DARTCache**, a training-free framework that jointly plans these two decisions using a single trajectory-based surrogate objective. We introduce *Temporal Variation Exposure* (TVE), which quantifies cumulative prediction variation between refresh locations. A quadratic TVE objective promotes exposure-balanced refresh placement and guides stage-wise budget allocation. **DARTCache** estimates temporal variation profiles from full-computation trajectories on a small calibration set and constructs fixed refresh schedules without candidate-level replay or online schedule search. On RF-Inversion and OT-RF, **DARTCache** achieves end-to-end speedups of 2.80× and 2.67×, with full-image LPIPS of 0.107 and 0.137 relative to the full-inference outputs, yielding favorable efficiency–fidelity trade-offs against evaluated acceleration baselines. Further experiments on U-Net-based MasaCtrl and EDICT support the applicability of the planning framework beyond FLUX-based editing pipelines.

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.

1 INTRODUCTION

Inversion-based diffusion methods provide an effective approach to real-image editing Mokady et al. (2023); Ju et al. (2024); Rout et al. (2025); Wang et al. (2025), enabling semantic modifications while preserving source-image content. These methods typically involve two iterative stages: an *inversion* stage that maps the source image to a noisy latent representation, followed by a *denoising* stage that produces the edited image under the target condition Rout et al. (2025); Wang et al. (2025). Both stages require repeated evaluations of the underlying network, introducing substantial inference cost. Reducing this cost while preserving the behavior of the original editing pipeline is therefore important for practical deployment.

Training-free caching reduces repeated computation by reusing intermediate features or model predictions across timesteps Ma et al. (2024); Selvaraju et al. (2024); Zou et al. (2025; 2024). Many existing methods design reuse policies for a single sampling trajectory Ma et al. (2024); Selvaraju et al. (2024); Zou et al. (2025); Liu et al. (2025). In inversion-based editing, however, a fixed computation budget must support both inversion and denoising. This introduces two planning decisions: *where* to refresh predictions within each stage and *how many* fresh evaluations to assign to each stage. The decisions are coupled through the shared budget: allocating more evaluations to one stage leaves fewer for the other, while the utility of each allocation depends on its refresh locations. Applying a caching policy independently to each stage does not, by itself, determine an effective joint plan under this constraint.

Figure 2 illustrates two properties that can inform this planning problem. First, adjacent-step prediction variation is non-uniform within each stage, so equally spaced refreshes can span substantially different amounts of trajectory change. Second, inversion and denoising differ in both the temporal distribution and the total magnitude of this variation. These observations motivate using temporal prediction variation to guide both refresh placement and stage-wise budget allocation. The resulting question is whether a common statistic extracted from full-computation trajectories can support both decisions without evaluating the output quality of individual candidate plans.

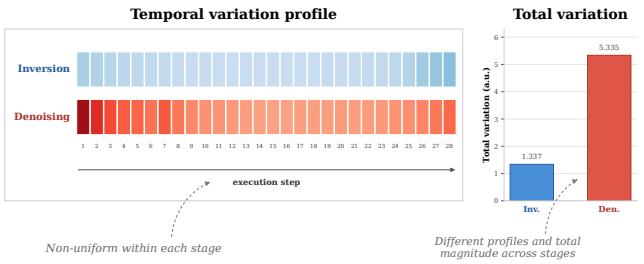


Figure 2: **Dual-stage temporal prediction variation.** Inversion and denoising exhibit non-uniform temporal dynamics with substantially different variation magnitudes and profiles.

To this end, we introduce *Temporal Variation Exposure* (TVE), defined as the cumulative prediction variation between consecutive refresh locations. Along a full-computation trajectory, accumulated adjacent-step variation bounds the prediction difference from an earlier step by the triangle inequality. This motivates TVE as a tractable surrogate that captures the temporal variation accumulated within a reuse interval. We minimize the sum of squared interval exposures under a shared computation budget. For a fixed stage budget, this objective favors refresh locations that balance exposure across intervals. Across stages, the same objective selects the budget split by comparing the exposures of the resulting schedules. Thus, refresh placement and stage allocation follow a single planning criterion.

Based on this formulation, we propose **DARTCache**, a training-free framework for dual-stage temporal reuse planning. DARTCache collects full-computation trajectories on a small calibration set and estimates an average temporal variation profile for each stage. For each feasible stage budget, it constructs refresh locations using cumulative-variation quantiles. It then evaluates the resulting inversion–denoising budget splits with the TVE objective and selects the best plan within this schedule family. The schedules are frozen before evaluation and shared across test examples. Planning requires no candidate-level replay, and inference requires no auxiliary prediction model or online schedule optimization. Once the profiles are collected, schedules for different computation budgets can be constructed without additional model evaluations.

We evaluate DARTCache on RF-Inversion Rout et al. (2025) and OT-RF Lupaşcu & Stupariu (2026), both using FLUX.1-dev. DARTCache achieves end-to-end speedups of 2.80× and 2.67×, with full-

image LPIPS of 0.107 and 0.137 relative to the corresponding full-inference outputs. These results show favorable efficiency–fidelity trade-offs against the evaluated acceleration baselines. Further analyses assess schedule ranking and selection, isolate the contributions of refresh placement and stage allocation, and examine robustness to calibration size and composition. A comparison with an adapted outcome-aware planner shows comparable fidelity at substantially lower offline planning cost. Additional quantitative experiments on U-Net-based MasaCtrl and EDICT support the applicability of the framework beyond FLUX-based editing.

Our contributions are summarized as follows:

- We formulate prediction caching for inversion-based image editing as a *dual-stage temporal reuse planning* problem, jointly selecting within-stage refresh locations and cross-stage computation budgets.
- We introduce TVE as a trajectory-based surrogate and develop a unified objective for refresh scheduling and stage allocation. The resulting planner constructs fixed-budget schedules from full-computation calibration trajectories without candidate-level replay.
- We demonstrate competitive efficiency–fidelity trade-offs across rectified-flow and U-Net-based editing pipelines, supported by analyses of objective validity, component contributions, calibration robustness, and offline planning cost.

2 Related Work

2.1 Diffusion Model Acceleration

Diffusion models require repeated network evaluations along an iterative sampling trajectory, motivating extensive efforts to reduce inference cost. Existing acceleration approaches mainly follow three directions: reducing sampling steps through improved numerical solvers Song et al. (2021); Lu et al. (2022; 2025) or distillation Luo et al. (2023), reducing per-step computation through token reduction Bolya & Hoffman (2023), and reusing intermediate computations across timesteps through training-free caching Ma et al. (2024); Li et al. (2024); So et al. (2024); Selvaraju et al. (2024). Among them, training-free caching is attractive because it requires no additional training and can be directly applied to pretrained diffusion models.

Existing caching methods exploit temporal redundancy along diffusion trajectories through different reuse strategies. Transformer-based approaches, including FORA Selvaraju et al. (2024), ToCa Zou et al. (2025), and DuCa Zou et al. (2024), improve efficiency through feature or token reuse across timesteps. Adaptive methods, such as TeaCache Liu et al. (2025), SenCache Haghighi & Alahi (2026), and SeaCache Chung et al. (2026), determine refresh decisions based on estimated variation, sensitivity, or feature differences. Recent offline planning approaches, such as LeMiCa Gao et al. (2025), formulate cache scheduling as budget-constrained optimization using outcome-aware segment costs obtained through candidate replay. However, existing caching methods mainly focus on reuse decisions within a single trajectory. In inversion-based editing, acceleration involves two iterative stages with different temporal dynamics under a shared computation budget. Therefore, independent stage-wise caching does not explicitly address joint refresh scheduling and budget allocation.

2.2 Image Editing

Diffusion models have enabled a wide range of image editing tasks, including prompt-guided editing, image composition, and spatially controlled manipulation Hertz et al. (2023); Lu et al. (2023); Shi et al. (2024). Training-free editing methods manipulate pretrained diffusion trajectories through latent modification, attention control, feature injection, or conditioning guidance Hertz et al. (2023); Tumanyan et al. (2023); Cao et al. (2023), allowing flexible editing without task-specific retraining.

Among these approaches, inversion-based methods first recover a latent or noise trajectory from the source image and then perform conditional generation. Representative methods include DDIM inversion Song et al. (2021), Null-Text Inversion Mokady et al. (2023), Direct Inversion Ju et al. (2024), and recent rectified-flow-based approaches such as RF-Inversion Rout et al. (2025) and RF-Solver Wang et al. (2025). Meanwhile, inversion-free approaches avoid explicit inversion and perform editing through modified generation trajectories Xu et al. (2024); Kulikov et al. (2025).

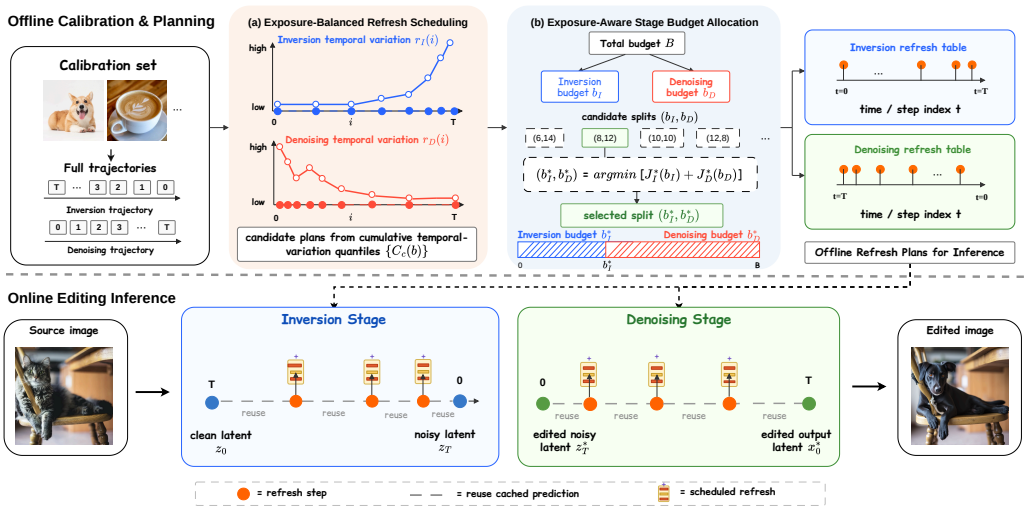


Figure 3: **Overview of DARTCache.** Given a calibration set, DARTCache extracts stage-specific temporal prediction variation from full inversion and denoising trajectories. For each stage-wise budget, it constructs exposure-balanced refresh schedules from cumulative temporal variation and then allocates the global computation budget between inversion and denoising using the same TVE objective. The resulting stage-wise refresh schedules are frozen and directly reused during inference.

Our work focuses on accelerating inversion-based editing. EEdit Yan et al. (2025) is closely related, as it exploits redundancy in the editing pipeline through editing-specific caching strategies, token index preprocessing, and inversion step skipping. However, it does not jointly optimize refresh placement and stage-wise computation allocation under a shared budget. In contrast, we formulate inversion-based acceleration as a *dual-stage temporal reuse planning* problem, jointly determining refresh locations and computation budgets across inversion and denoising.

3 Method

Unlike standard diffusion generation, inversion-based editing contains two consecutive iterative stages with distinct temporal dynamics. **DARTCache** jointly plans prediction reuse over inversion and denoising under a fixed number of fresh model evaluations, deciding both where to refresh predictions within each stage and how to distribute the refresh budget between the two stages.

DARTCache builds on the observation that reuse difficulty is closely related to the amount of trajectory change covered by stale predictions, rather than timestep distance alone. Based on this observation, **DARTCache** introduces Temporal Variation Exposure (TVE), which quantifies the cumulative prediction variation exposed by each reuse interval. By minimizing the concentration of TVE, **DARTCache** jointly determines where predictions should be refreshed within each stage and how the refresh budget should be allocated between inversion and denoising. The overall framework is illustrated in Figure 3.

3.1 Problem Formulation

Dual-stage inversion-based editing. We consider inversion-based editing as two consecutive iterative stages, $s \in \{I, D\}$, corresponding to inversion and denoising. Timesteps in stage s are indexed by execution order $i = 0, \ldots, N_s - 1$, and p_i^s denotes the model prediction used by the underlying inversion or denoising procedure, e.g., a velocity or noise prediction. The formulation is applicable to different prediction semantics as long as the temporal variation magnitude is measured in the corresponding prediction space.

Dual-stage prediction reuse. For stage s, we define a refresh schedule

$$C_s = \{c_0^s, \ldots, c_{b_s-1}^s\}, \qquad b_I + b_D = B, \tag{1}$$

where C_s contains the timesteps at which fresh model predictions are evaluated, b_s is the number of fresh evaluations assigned to stage s, and B is the global computation budget. At timesteps not included in C_s, the latest available prediction is reused.

Under the shared budget B, temporal reuse therefore involves two coupled decisions: *where* to place refreshes within inversion and denoising, and *how* to divide the available fresh evaluations between the two stages. We next derive a unified objective for both decisions.

3.2 Temporal Variation Exposure Objective

Temporal reuse becomes less reliable when a stale prediction spans a region where model predictions change substantially. Rather than evaluating the reuse error of every candidate schedule, DART-Cache derives a schedule-independent temporal variation profile directly from full-computation trajectories.

Temporal prediction variation. For calibration example j and stage s, we measure the local prediction variation between consecutive steps as

$$q_{j,s}(i) = \left\| p^s_{j,i} - p^s_{j,i-1} \right\|_1, \qquad i \geq 1, \tag{2}$$

Suppose a prediction computed at step a is reused at a later step $t > a$. The triangle inequality gives

$$\left\| p^s_{j,t} - p^s_{j,a} \right\|_1 \leq \sum_{u=a+1}^{t} q_{j,s}(u). \tag{3}$$

Thus, the accumulated temporal variation along a reuse interval provides a trajectory-level surrogate for stale prediction exposure. Given a candidate refresh schedule, TVE summarizes the temporal variation covered by its induced reuse intervals and provides a trajectory-based criterion for offline reuse planning.

Since **DARTCache** uses a shared offline schedule, we aggregate this local variation over the calibration set:

$$r_s(i) = \frac{1}{|\mathcal{D}_{\text{calib}}|} \sum_{j \in \mathcal{D}_{\text{calib}}} q_{j,s}(i) \tag{4}$$

where $r_s(i)$ captures the stage-specific temporal prediction variation used for offline planning.

Temporal Variation Exposure. Given a refresh schedule C_s, two consecutive refresh locations define a reuse interval. We define the *Temporal Variation Exposure* (TVE) of the interval as

$$E^s_\ell(C_s) = \sum_{i=c^s_\ell+1}^{c^s_{\ell+1}} r_s(i). \tag{5}$$

TVE measures the cumulative prediction variation covered between two consecutive refresh locations. A large E^s_ℓ therefore indicates that the same stale prediction is reused across a strongly varying region of the prediction trajectory.

Global TVE objective. Under a fixed computation budget, we seek refresh plans that avoid concentrating excessive temporal prediction variation into a small number of reuse intervals. We therefore define the global planning objective as

$$\mathcal{J}(C_I, C_D) = \sum_{s \in \{I,D\}} \sum_{\ell} \left(E^s_\ell(C_s) \right)^2. \tag{6}$$

The quadratic objective penalizes concentrated TVE and favors plans in which prediction variation is more evenly distributed over reuse intervals. More importantly, Eq. 6 provides a single criterion for both planning decisions: for a fixed stage-wise budget it selects where refreshes should be placed, while under a fixed global budget it selects how refreshes should be distributed between inversion and denoising. We show in the Appendix C.2 that equalizing TVE minimizes maximum interval exposure under the TVE formulation, while the quadratic objective provides a discrete concentration penalty motivated by this equal-exposure principle.

3.3 EXPOSURE-BALANCED REFRESH SCHEDULING

We first optimize the refresh locations for a given stage-wise budget b_s. Under a fixed b_s, all reuse intervals together span the same stage trajectory, and hence their total accumulated temporal variation is fixed. Therefore, minimizing the stage-wise contribution of Eq. 6 is equivalent to minimizing the dispersion of TVE across reuse intervals. The continuous optimum is attained when all intervals have equal TVE; a detailed derivation is provided in the Appendix C.3. On the discrete trajectory, we realize this equal-exposure principle through cumulative-variation quantiles.

Specifically, we define the normalized cumulative variation

$$F_s(i) = \frac{\sum_{u=1}^{i} r_s(u)}{\sum_{u=1}^{N_s-1} r_s(u)}. \tag{7}$$

DARTCache places refreshes at uniformly spaced quantiles of F_s:

$$C_s(b_s) = \left\{0,\ F_s^{-1}\left(\frac{1}{b_s-1}\right), \ldots,\ F_s^{-1}\left(\frac{b_s-2}{b_s-1}\right), N_s-1\right\}. \tag{8}$$

This construction approximately equalizes TVE among reuse intervals on a discrete trajectory. Consequently, refreshes become denser where model predictions change rapidly and sparser where they evolve slowly. When temporal prediction variation is uniform, cumulative variation is proportional to timestep distance, and Eq. 8 naturally reduces to uniform refresh scheduling.

3.4 EXPOSURE-AWARE STAGE BUDGET ALLOCATION

The previous section constructs an exposure-balanced refresh schedule $C_s(b_s)$ for any given stage-wise budget. We now optimize the second variable of the global planning problem: how the total budget B should be divided between inversion and denoising.

The same TVE objective also guides cross-stage budget allocation. Let $V_s = \sum_{i=1}^{N_s-1} r_s(i)$ denote the total temporal variation of stage s. Under exposure-balanced scheduling,

$$\sum_{\ell} (E_\ell^s)^2 \geq \frac{V_s^2}{b_s - 1},$$

indicating that stages with larger temporal variation are favored to receive more refreshes under the TVE objective; in the continuous relaxation, $b_s - 1 \propto V_s$. A full derivation is provided in the Appendix C.4.

On the discrete trajectory, however, exact equal exposure may not be attainable. We therefore evaluate the actual TVE objective induced by $C_I(b_I)$ and $C_D(b_D)$ for every feasible budget split and select

$$(b_I^\star, b_D^\star) = \arg\min_{b_I + b_D = B} \mathcal{J}\left(C_I(b_I), C_D(b_D)\right). \tag{9}$$

Because the feasible number of stage splits is small, **DARTCache** evaluates all candidate splits offline. Eq. 9 therefore gives the exact optimum over the family of exposure-balanced schedules constructed in Sec. 3.3. Importantly, the allocation retains the native magnitude of temporal prediction variation in the two stages, allowing stages with different amounts of temporal variation to receive different computation budgets.

3.5 DARTCACHE INFERENCE

DARTCache separates offline reuse planning from online execution. During offline planning, full-computation trajectories from a calibration set are used to estimate the stage-specific temporal variation $r_I(i)$ and $r_D(i)$. The exposure-balanced schedules are then constructed for feasible stage-wise budgets, and the global TVE objective selects the final budget split, yielding frozen refresh schedules $C_I^\star$ and $C_D^\star$.

During inference, these schedules are directly shared across evaluation examples. A fresh model evaluation is performed only at the selected refresh locations; otherwise, the latest available prediction is reused. Thus, **DARTCache** requires no per-instance scheduling or online optimization, introducing only the overhead of cache management during inference.

Table 1: **Main quantitative comparison on PIE-Bench.** We compare **DARTCache** with acceleration baselines under RF-Inversion and OT-RF using FLUX.1-dev. Fidelity is measured against full inference over the editing mask, background, and full image.

Method	Fidelity to Full Inference						Efficiency			
	$\text{PSNR}_{\text{M}}\uparrow$	$\text{LPIPS}_{\text{M}}\downarrow$	$\text{PSNR}_{\text{BG}}\uparrow$	$\text{LPIPS}_{\text{BG}}\downarrow$	$\text{PSNR}_{\text{All}}\uparrow$	$\text{LPIPS}_{\text{All}}\downarrow$	Inv. (s) ↓	Den. (s) ↓	Latency (s) ↓	Speedup ↑
RF-Inversion *(FLUX.1-dev)*										
Full Inference	∞	0.0000	∞	0.0000	∞	0.0000	2.28	2.33	4.71	1.00×
50%-step	19.22	0.1901	29.80	0.0402	22.31	0.2191	1.15	1.15	2.39	1.97×
FORA	18.83	0.1938	29.16	0.0452	21.71	0.2251	0.89	0.97	1.96	2.40×
ToCa	17.26	0.2472	25.86	0.0941	19.95	0.3143	1.50	1.54	3.14	1.50×
DuCa	18.94	0.1924	29.38	0.0444	21.83	0.2232	1.28	1.38	2.76	1.71×
TeaCache	20.81	0.1503	29.45	0.0451	23.43	0.1829	0.90	0.97	1.97	2.40×
SenCache	23.16	0.1102	30.29	0.0440	25.37	0.1423	1.01	1.05	2.06	2.28×
SeaCache	21.76	0.1372	27.39	0.0749	23.58	0.1929	0.92	0.92	1.93	2.45×
EEdit	19.02	0.1909	29.49	0.0438	21.93	0.2215	0.73	1.10	1.92	2.45×
DARTCache	**24.06**	**0.0777**	**31.94**	**0.0387**	**26.90**	**0.1071**	0.48	1.12	**1.68**	2.80×
OT-RF *(FLUX.1-dev)*										
Full Inference	∞	0.0000	∞	0.0000	∞	0.0000	2.03	2.02	4.04	1.00×
50%-step	19.91	0.1151	22.27	0.1468	21.08	0.2195	1.04	1.03	2.07	1.95×
FORA	17.70	0.1777	19.79	0.2210	18.84	0.3358	0.77	0.76	1.53	2.62×
ToCa	18.61	0.1483	20.73	0.1848	19.74	0.2794	1.32	1.26	2.58	1.56×
DuCa	22.05	0.0817	24.44	0.1041	23.14	0.1568	1.48	1.41	2.89	1.39×
TeaCache	21.44	0.0957	23.78	0.1231	22.50	0.1848	0.74	0.81	1.56	2.58×
SenCache	20.19	0.1230	22.46	0.1533	21.23	0.2331	0.71	0.83	1.55	2.60×
SeaCache	21.95	0.0906	24.36	0.1127	23.04	0.1562	0.67	0.87	1.54	2.62×
EEdit	14.78	0.2345	16.87	0.2990	15.90	0.3434	0.76	0.98	1.74	2.32×
DARTCache	**23.70**	**0.0706**	**25.98**	**0.0906**	**24.67**	**0.1374**	0.54	0.97	**1.51**	2.67×

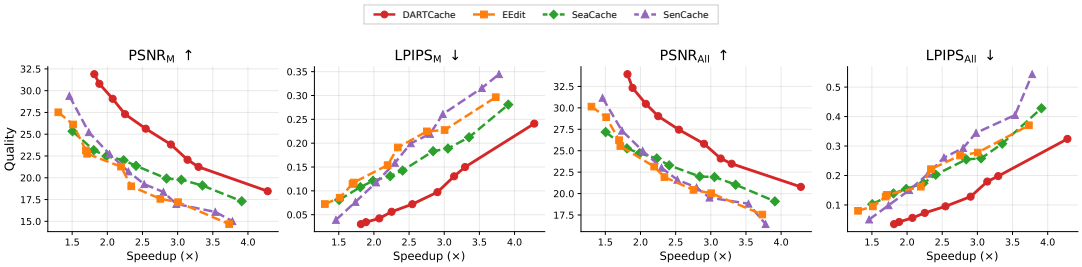


Figure 4: **Speed–quality trade-off.** DARTCache consistently achieves better fidelity at comparable speedups.

4 Experiments

4.1 Experimental Setup

Models and baselines. We evaluate **DARTCache** on RF-Inversion Rout et al. (2025) and OT-RF Lupaşcu & Stupariu (2026), both using FLUX.1-dev. We follow the official host configurations unless otherwise specified. Baselines include FORA, ToCa, DuCa, TeaCache, SenCache, and SeaCache Selvaraju et al. (2024); Zou et al. (2025; 2024); Liu et al. (2025); Haghighi & Alahi (2026); Chung et al. (2026), and the editing-specific EEdit Yan et al. (2025). For absolute editing-quality comparison in Table 2, we additionally include representative editing methods, including Prompt-to-Prompt Hertz et al. (2023), MasaCtrl Cao et al. (2023), PnP Tumanyan et al. (2023), InfEdit Xu et al. (2024), RF-Solver-Edit Wang et al. (2025), FlowEdit Kulikov et al. (2025), UniEdit-Flow Jiao et al. (2026), and StableFlow Avrahami et al. (2025), following their released or reported configurations. Adaptations and configurations of the acceleration baselines are detailed in Appendix B.1.

Main evaluation settings. Full inference follows the official 28-step configuration per stage. RF-Inversion executes 27 inversion updates (indices 0–26) and 28 denoising updates (0–27), totaling 55 Transformer evaluations. Its guidance scale is 7.0; OT-RF follows its official settings. We calibrate DARTCache on 70 PIE-Bench examples and evaluate all main results in Tables 1 and 2 on the disjoint remaining 630 examples, using a fixed split seed of 0. Profiles and schedules are frozen before evaluation. The default budget $B = 20$ counts fresh Transformer evaluations across both stages, excluding text encoding, VAE operations, and offline calibration.

Data and metrics. PIE-Bench Ju et al. (2024) contains 700 examples across 10 editing types, with source/target prompts and editing masks. All main results are evaluated on the fixed 630-example evaluation split disjoint from the 70-example calibration set. Table 1 measures fidelity to

Table 3: **Component ablation of DARTCache.** Exposure-balanced scheduling improves fidelity, while stage budget allocation provides further gains.

Components		Budget	Fidelity to Full Inference			
Scheduling	Allocation	Inv./Den.	PSNR$_M$ ↑	LPIPS$_M$ ↓	PSNR$_{All}$ ↑	LPIPS$_{All}$ ↓
×	×	14/14	22.770	0.0949	26.303	0.1157
✓	×	14/14	25.427	0.0593	28.003	0.0877
×	✓	8/20	27.022	0.0593	28.990	0.0738
✓	✓	8/20	**27.846**	**0.0403**	**30.287**	**0.0603**

Table 4: **Comparison of offline objectives.** Quadratic TVE achieves the strongest ranking agreement and lowest selection regret.

Objective	ρ ↑	Top-1 regret↓	Top-5 regret↓
Endpoint	0.6433	0.0501	0.0371
Integrated	0.7695	0.0194	0.0134
Lexi-Minimax	0.9316	0.0294	0.0178
Quadratic TVE	**0.9637**	**0.0116**	**0.0102**

the corresponding full-inference outputs using full-image, edited-region, and background PSNR and LPIPS Zhang et al. (2018). Table 2 evaluates absolute editing quality on the same 630-example split. Its PSNR and AlexNet LPIPS compare outputs with full RGB source images. CLIP-I and CLIP-T use normalized-feature cosine similarity from `CLIP ViT-L/14` against source images and target prompts, respectively. We also report latency and speedup relative to full inference.

Implementation details. Experiments use PyTorch and NVIDIA H200 GPUs. Caches are reset between inversion and denoising. OT-RF reuses Transformer predictions while executing OT control and latent updates at every step. For Table 1 and 2, eight single-GPU workers process separate examples at 512×512, in BF16, with batch size 1. Each case is timed once with CUDA synchronization and no additional warm-up. Timing includes text encoding and VAE operations, but excludes model loading and image saving.

4.2 Main Results

Quantitative comparison. Table 1 reports fidelity and efficiency on RF-Inversion and OT-RF. At the reported operating points, **DARTCache** achieves the lowest full-image LPIPS among the evaluated acceleration methods, with values of 0.107 and 0.137 at $2.80\times$ and $2.67\times$ speedups, respectively. It also preserves edited and background content effectively, supporting its applicability to both evaluated editors. Appendix D.3 compares fidelity and offline cost with an adapted outcome-aware planner.

Editing quality and efficiency. Table 2 complements the full-inference fidelity evaluation in Table 1 with source-referenced editing and preservation metrics on the same held-out 630-example evaluation split. On RF-Inversion, **DARTCache** reduces latency from 4.71 s to 1.68 s while maintaining CLIP scores and source-preservation metrics close to the full host. It also achieves higher CLIP-T and PSNR, lower LPIPS, and lower latency than the reported EEdit-SLoC+ISS configuration. On OT-RF, latency decreases from 4.04 s to 1.51 s. CLIP-T, CLIP-I, and PSNR increase, whereas source-referenced LPIPS rises from 0.287 to 0.402. These results indicate a trade-off among quality metrics rather than uniform improvement.

Table 2: **Editing quality and efficiency on PIE-Bench.** PSNR and LPIPS are measured against source images.

Method	CLIP-T ↑	CLIP-I ↑	LPIPS ↓	PSNR ↑	Latency ↓
Prompt-to-Prompt	25.75	86.32	0.240	20.94	45.70
MasaCtrl	24.00	87.70	0.180	19.12	6.48
MasaCtrl+DI	24.39	90.88	0.153	19.50	9.36
PnP	25.00	91.13	0.116	22.30	6.32
PnP+DI	25.48	87.92	0.182	19.12	4.77
InfEdit	25.67	91.44	0.173	21.86	0.63
RF-Inversion	25.81	85.26	0.365	18.01	4.71
RF-Solver-Edit	25.80	86.08	0.300	21.05	9.78
FlowEdit	26.16	87.97	0.280	18.53	3.91
OT-RF	24.96	85.08	0.287	16.84	4.04
UniEdit-Flow	25.71	89.53	0.154	25.49	2.16
StableFlow	24.76	92.70	0.180	19.93	10.85
EEdit-SLoC	25.83	85.59	0.367	17.82	3.13
EEdit-SLoC+ISS	25.80	85.57	0.367	17.82	1.92
DARTCache-RF	25.86	85.55	0.359	18.21	1.68
DARTCache-OT-RF	25.37	85.40	0.402	17.28	1.51

Trade-off curves. Figure 4 compares efficiency–fidelity trade-offs across multiple operating points. DARTCache achieves higher PSNR and lower LPIPS than the plotted baselines at comparable speedups for both edited-region and full-image evaluation. The advantage therefore extends beyond the default budget.

4.3 Ablation Studies

Unless otherwise specified, ablations use RF-Inversion with FLUX.1-dev. We examine component contributions, objective validity, stage allocation, and calibration sensitivity.

Contribution of each component. Table 3 compares uniform and exposure-balanced scheduling under equal and TVE-selected allocations at $B = 28$. With the $14/14$ split, exposure-balanced scheduling reduces full-image LPIPS from 0.1157 to 0.0877. The $8/20$ allocation improves both

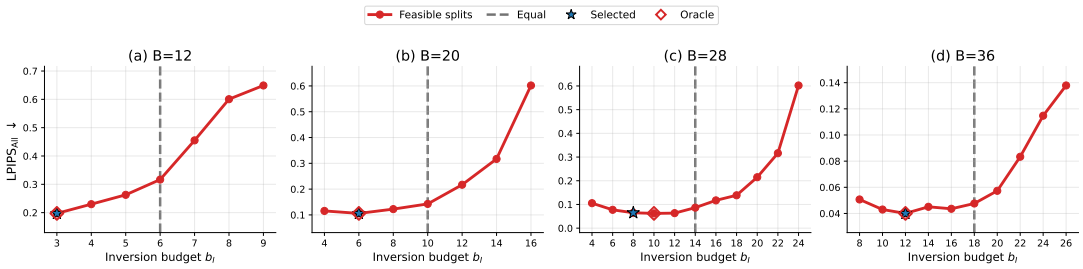


Figure 6: **Validation of stage budget allocation.** The objective-selected split matches the empirical LPIPS oracle at $B = 12$, 20, and 36, and remains within one search interval at $B = 28$, consistently outperforming equal allocation.

scheduling variants, and combining it with exposure-balanced scheduling achieves 0.0603. Both placement and allocation thus contribute to the improvement.

Validation of the TVE objective. We sample 200 refresh schedules at a fixed $10/10$ stage split and evaluate each on 50 prompts. The objective $\mathcal{J}$ is computed solely from frozen calibration profiles. Figure 5 shows a Spearman correlation of 0.9637 between TVE and mean full-image LPIPS relative to full inference. The TVE-selected schedule also has low regret relative to the best candidate in the sampled pool, supporting its use for schedule ranking and selection.

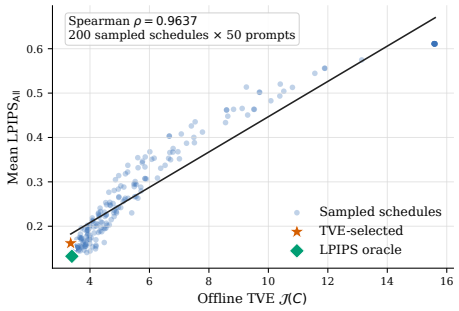


Figure 5: **Validation of the optimization objective.** The offline objective $\mathcal{J}(C)$ strongly aligns with actual schedule degradation over sampled refresh schedules.

Table 4 compares quadratic TVE with Endpoint, Integrated, and Lexi-Minimax on the same candidates. Quadratic TVE achieves the highest rank correlation and lowest Top-1 and Top-5 regret at this setting. Top-k regret compares the best empirical result among the objective's top-k candidates with the sampled-pool oracle. Additional budgets and host comparisons appear in Appendix D.2.

Validation of stage budget allocation. We evaluate allocation across $B \in \{12, 20, 28, 36\}$ using exposure-balanced schedules for each split. The TVE-selected split outperforms equal allocation at all four budgets, matches the empirical LPIPS oracle among evaluated splits at $B = 12, 20, 36$, and differs by one evaluated split interval at $B = 28$. This supports its effectiveness as an allocation criterion in the tested settings.

Calibration profile sensitivity. We construct profiles from 35, 70, 175, 350, and 700 PIE-Bench examples, corresponding to 5%, 10%, 25%, 50%, and 100% coverage. All resulting schedules are evaluated on the full 700 examples, including their calibration subsets. Performance varies little across these settings, with full-image LPIPS ranging from 0.1051 to 0.1064. This indicates limited sensitivity to calibration coverage. This experiment assesses profile-based planning sensitivity, not held-out generalization.

Table 5: **Calibration profile sensitivity.**

Calib.	PSNR$_M$ ↑	LPIPS$_M$ ↓	PSNR$_{All}$ ↑	LPIPS$_{All}$ ↓
5%	24.0072	0.0767	26.8243	0.1056
10%	24.1234	0.0734	26.9247	0.1064
25%	24.1863	0.0764	26.9561	0.1061
50%	24.1862	**0.0760**	26.9685	0.1053
100%	**24.2489**	0.0761	**26.9931**	**0.1051**

5 Conclusion

We formulate training-free caching for inversion-based image editing as a dual-stage temporal reuse planning problem. **DARTCache** uses Temporal Variation Exposure (TVE) to align refresh decisions with prediction variation along the trajectory, and jointly optimizes stage-wise refresh scheduling and inversion–denoising budget allocation under a shared computation budget. Experiments on RF-Inversion and OT-RF show that this dual-stage planning consistently improves the efficiency-fidelity trade-off across the evaluated inversion-based editing pipelines.

AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on research methodology and experimental design, assist with implementation and result interpretation, and support manuscript editing and literature search, and assist with checking mathematical derivations. We did not use generative AI to generate experimental results or replace human verification of scientific claims. All AI-assisted code, analyses, and manuscript content were reviewed and verified by the authors. We take full responsibility for the final content of this work, including all reported results and scientific claims.

REFERENCES

Omri Avrahami, Or Patashnik, Ohad Fried, Egor Nemchinov, Kfir Aberman, Dani Lischinski, and Daniel Cohen-Or. Stable flow: Vital layers for training-free image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7877–7888, June 2025.

Daniel Bolya and Judy Hoffman. Token Merging for Fast Stable Diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 4599–4603, 2023.

Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. MasaCtrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22560–22570, 2023.

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.

Huanlin Gao, Ping Chen, Fuyuan Shi, Chao Tan, Zhaoxiang Liu, Fang Zhao, Kai Wang, and Shiguo Lian. Lemica: Lexicographic minimax path caching for efficient diffusion-based video generation. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

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

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-Prompt Image Editing with Cross-Attention Control. In *International Conference on Learning Representations*, 2023.

Guanlong Jiao, Biqing Huang, Kuan-Chieh Jackson Wang, and Renjie Liao. Uniedit-flow: Unleashing inversion and editing in the era of flow models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ArU2CeB7Tm.

Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Direct Inversion: Boosting diffusion-based editing with 3 lines of code. In *International Conference on Learning Representations*, 2024.

Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. FlowEdit: Inversion-free text-based editing using pre-trained flow models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19721–19730, 2025.

Senmao Li, Taihang Hu, Joost van de Weijer, Fahad Shahbaz Khan, Tao Liu, Linxuan Li, Shiqi Yang, Yaxing Wang, Ming-Ming Cheng, and Jian Yang. Faster Diffusion: Rethinking the role of the encoder for diffusion model inference. In *Advances in Neural Information Processing Systems*, volume 37, pp. 85203–85240, 2024.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep Embedding Tells: It's Time to Cache for Video Diffusion Model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7353–7363, 2025.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems*, volume 35, pp. 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: 730–751, 2025. doi: 10.1007/s11633-025-1562-4.

Shilin Lu, Yanzhu Liu, and Adams Wai-Kin Kong. TF-ICON: Diffusion-based training-free cross-domain image composition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2294–2305, 2023.

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

Marian Lupaşcu and Mihai Sorin Stupariu. Optimal transport for rectified flow image editing: Unifying inversion-based and direct methods. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 6764–6774, March 2026.

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

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. NULL-Text Inversion for Editing Real Images Using Guided Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6038–6047, 2023.

Litu Rout, Yujia Chen, Nataniel Ruiz, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Semantic Image Inversion and Editing using Rectified Stochastic Differential Equations. In *International Conference on Learning Representations*, 2025.

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.

Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Hanshu Yan, Wenqing Zhang, Vincent Y. F. Tan, and Song Bai. DragDiffusion: Harnessing diffusion models for interactive point-based image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8839–8849, 2024.

Junhyuk So, Jungwon Lee, and Eunhyeok Park. FRDiff: Feature reuse for universal training-free acceleration of diffusion models. In *European Conference on Computer Vision*, pp. 328–344, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*, 2021.

Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1921–1930, 2023.

Bram Wallace, Akash Gokul, and Nikhil Naik. Edict: Exact diffusion inversion via coupled transformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22532–22541, 2023. doi: 10.1109/CVPR52729.2023.02158.

Jiangshan Wang, Junfu Pu, Zhongang Qi, Jiayi Guo, Yue Ma, Nisha Huang, Yuxin Chen, Xiu Li, and Ying Shan. Taming Rectified Flow for Inversion and Editing. In *International Conference on Machine Learning*, volume 267, pp. 64044–64058, 2025.

Sihan Xu, Yidong Huang, Jiayi Pan, Ziqiao Ma, and Joyce Chai. Inversion-Free Image Editing with Language-Guided Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9452–9461, 2024.

Zexuan Yan, Yue Ma, Chang Zou, Wenteng Chen, Qifeng Chen, and Linfeng Zhang. EEdit: Rethinking the spatial and temporal redundancy for efficient image editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 17474–17484, 2025.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

Chang Zou, Evelyn Zhang, Runlin Guo, Haohang Xu, Conghui He, Xuming Hu, and Linfeng Zhang. Accelerating Diffusion Transformers with Dual Feature Caching. *arXiv preprint arXiv:2412.18911*, 2024.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating Diffusion Transformers with Token-Wise Feature Caching. In *International Conference on Learning Representations*, 2025.

A LIMITATIONS

DARTCache still has several limitations. First, TVE is designed as a tractable surrogate for reuse difficulty based on accumulated prediction variation. While it is strongly aligned with empirical fidelity degradation, it does not explicitly model the nonlinear propagation and interaction of reuse errors across subsequent inversion and denoising steps.

Second, **DARTCache** adopts dataset-level temporal variation profiles estimated from an offline calibration set and freezes the resulting schedules during evaluation. This design keeps test-time overhead negligible, but may be less responsive to atypical samples or distribution shifts whose temporal dynamics differ substantially from the calibration data.

Third, TVE relies on the temporal prediction variation measured under a fixed model architecture and prediction parameterization. Therefore, the resulting schedules are intended for the same host pipeline rather than arbitrary cross-model transfer.

Finally, refresh locations are constructed through exposure-balanced cumulative-variation quantiles. Therefore, the resulting stage allocation is optimized over this structured schedule family rather than the complete space of discrete refresh schedules. Exploring more general discrete scheduling while retaining the simplicity and robustness of **DARTCache** is left for future work.

B IMPLEMENTATION DETAILS

B.1 BASELINE CONFIGURATIONS

Common adaptation and tuning protocol. We use the official implementations and released configurations when available, and explicitly adapt methods that do not provide native support for the evaluated inversion–denoising pipelines. Unless otherwise stated, caching is applied independently to inversion and denoising, with separate cache states reset at the stage boundary. The host-specific editing procedure, guidance, controller operations, and latent updates are otherwise left unchanged.

For adaptive baselines requiring hyperparameter selection, parameters are selected exclusively on the same 70-case calibration split used by DARTCache (split seed 0), without accessing the 630 evaluation cases. Adaptive gating does not necessarily yield exactly $B = 20$ fresh model evaluations on every example; such methods are therefore evaluated at comparable rather than strictly identical computation levels.

FORA Selvaraju et al. (2024) is a training-free caching method for diffusion transformers that exploits temporal redundancy across adjacent denoising steps. It caches and reuses intermediate outputs from the attention and MLP modules, allowing selected transformer computations to be skipped at subsequent timesteps without model retraining. In our experiments, we follow the official implementation and apply FORA independently to the inversion and denoising stages of both RF-Inversion and OT-RF, with the cache state reset at the stage boundary. We set the cache interval to 3.

ToCa Zou et al. (2025) is a training-free acceleration method for diffusion transformers that exploits token-wise temporal redundancy by selectively recomputing important tokens while reusing cached features for less sensitive ones. We follow the official FLUX implementation with attention-based caching, a fresh threshold of 4, a fresh ratio of 0.1, the ToCa ratio scheduler, global forced refresh, and a soft fresh weight of 0.25. We apply ToCa independently to inversion and denoising in both RF-Inversion and OT-RF, resetting the cache state at the stage boundary.

DuCa Zou et al. (2024) is a training-free diffusion transformer acceleration method that employs dual feature caching to exploit temporal redundancy while retaining important feature updates. Following the official configuration, we use attention-based caching with a fresh threshold of 3, a fresh ratio of 0.05, the ToCa ratio scheduler, global forced refresh, and a soft fresh weight of 0.25. We apply DuCa independently to inversion and denoising in both RF-Inversion and OT-RF, resetting the cache state at the stage boundary.

TeaCache Liu et al. (2025) is a training-free caching method that estimates model-output changes from timestep embeddings and uses an accumulated relative-error criterion to determine whether a Transformer evaluation can be reused. We adopt the official TeaCache4FLUX gating rule and its released `rel_l1_thresh` $= 0.6$ setting. Because the official FLUX implementation is designed

for the denoising trajectory rather than inversion-based editing, we extend the same gating rule independently to inversion and denoising. The two stages maintain separate cache states, which are reset at the stage boundary, while the host-specific guidance, control, and latent updates remain unchanged.

SenCache Haghighi & Alahi (2026). We apply its sensitivity-aware gating independently to inversion and denoising, with separate cache states reset at the stage boundary. At step k, the reuse score is $E_k = \widehat{J}_{z,a}\|z_k - z_a\|_2 + \widehat{J}_{t,a}|t_k - t_a|$, where a is the most recent full-computation position and $\widehat{J}_{z,a}, \widehat{J}_{t,a}$ are stage- and position-specific finite-difference sensitivity estimates. We use absolute tensor-scale errors without feature-norm normalization and set `max_skips`=3. For RF-Inversion, we use $(\tau_{\text{start}}, \tau_{\text{main}}) = (130, 220)$; for OT-RF, we use $(157.5, 262.5)$. The start threshold is used at inversion indices 0–4 and denoising indices 0–5, after which the main threshold is applied. Reuse is allowed when $E_k < \tau_k$ subject to the original SenCache constraints.

SeaCache Chung et al. (2026). We adopt the official SeaCache FLUX gating mechanism, which uses the SEA-filtered modulated input and accumulated relative-L_1 change to determine whether the Transformer blocks should be recomputed or the cached residual reused. We apply the same gating rule independently to inversion and denoising, with separate cache states reset at the stage boundary. Host-specific thresholds are selected from a sweep on the 70-case calibration split to target an end-to-end speedup comparable to DARTCache, without using evaluation-set fidelity. This yields $\delta = 0.40$ for RF-Inversion and $\delta = 0.50$ for OT-RF; the selected configurations are then frozen for evaluation.

EEdit Yan et al. (2025) combines Spatial Locality Caching (SLoC), Token Index Preprocessing (TIP), and Inversion Step Skipping (ISS). For RF-Inversion, we use the official prompt-guided configuration: `cascade_num`=3, `fresh_ratio`=0.1, `fresh_threshold`=3, `soft_fresh_weight`=0.25, `tailing_step`=1, `inv_skip`=3, $\eta = \gamma = 0.7$, `stop_timestep`=6, `mask_timestep`=18, and `cache_type`=`ours_predefine`. For OT-RF, we adapt ISS to inversion with interval 3. At skipped inversion steps, the previous velocity is reused while OT control and latent updates continue. During denoising, we retain the same SLoC+TIP configuration; all 28 steps invoke the Transformer, with 10 full evaluations and 18 feature/token-level partial refreshes. The original OT control windows are preserved.

B.2 Calibration Details

Calibration protocol. For the main fidelity evaluation, PIE-Bench is split into 70 calibration and 630 evaluation cases using seed 0. RF-Inversion and OT-RF are calibrated independently on the same split, with schedules frozen before evaluation. All results reported in Tables 1 and 2 are computed on the disjoint 630-case evaluation split. The calibration-profile sensitivity experiment in Table 5 is a separate analysis: it constructs profiles using 35, 70, 175, 350, and 700 examples and evaluates the resulting schedules on all 700 PIE-Bench examples. Because calibration and evaluation may overlap in that analysis, Table 5 is intended to measure profile sensitivity rather than held-out generalization.

Calibration of stage-wise variation profiles. For each calibration sample, we run full, non-cached inference and record the effective model prediction that drives the update at every inversion and denoising step. For sample j and stage s, we compute the adjacent-step variation as $q_{j,s}(i) = \left\|p^s_{j,i} - p^s_{j,i-1}\right\|_1$. We then average these traces timestep-wise across the calibration set to obtain the dataset-level stage profile $r_s(i)$. Calibration is performed separately for each host pipeline so that the profiles reflect its native inversion and denoising dynamics.

Calibration of stage allocation and refresh schedules. Given a global fresh-evaluation budget B, we construct candidate inversion and denoising refresh schedules from equal-mass quantiles of their cumulative variation profiles. For every feasible allocation satisfying $b_I + b_D = B$, we construct the two stage schedules and evaluate the TVE objective defined in Sec. 3. The allocation with the minimum TVE is selected, and the resulting schedules are fixed before evaluation. At the default $B = 20$, RF-Inversion selects $(b_I, b_D) = (6, 14)$ on PIE-Bench. OT-RF is calibrated independently under the same procedure.

Offline calibration cost and inference-time overhead. Calibration is performed entirely offline and requires only full, non-cached trajectories from the 70 calibration samples. Profile aggregation,

quantile schedule construction, and stage-allocation search take only milliseconds. Once the profiles have been collected, changing the global budget B does not require rerunning the diffusion model; new schedules are obtained directly from the frozen profiles. At inference time, **DARTCache** simply loads the precomputed schedules and therefore introduces no additional model forward pass, learned predictor, or online optimization.

Calibration reuse. The calibrated stage profiles and refresh schedules are shared by all samples in the corresponding evaluation split. Thus, calibration is performed once per host pipeline rather than per image, and the same profiles can be reused to construct schedules for different computation budgets.

C Theoretical Analysis of the TVE Objective

This section analyzes TVE as a reference-trajectory surrogate. For fixed calibration profiles and trajectory endpoints, the sum of interval exposures is invariant to refresh placement. In an unconstrained exposure relaxation, both minimax and quadratic objectives favor equal exposure. This yields a common interpretation of within-stage scheduling and cross-stage allocation. The analysis does not establish equivalence between these objectives on the discrete schedule space or bound downstream editing error.

C.1 A Minimax Principle for Temporal Variation Exposure

Exposure as a surrogate for reuse discrepancy. Recall that for calibration example j and stage s, the local temporal prediction variation is defined as

$$q_{j,s}(i) = \left\| p^s_{j,i} - p^s_{j,i-1} \right\|_1 . \tag{10}$$

Consider consecutive refresh locations $a < b$. The cached prediction is reused only at steps $a < t < b$; step b is refreshed. For any $a < t \le b$, prediction drift along the full-computation reference trajectory satisfies

$$\begin{aligned} \left\| p^s_{j,t} - p^s_{j,a} \right\|_1 &= \left\| \sum_{u=a+1}^{t} \left(p^s_{j,u} - p^s_{j,u-1} \right) \right\|_1 \\ &\le \sum_{u=a+1}^{t} q_{j,s}(u) \\ &\le \sum_{u=a+1}^{b} q_{j,s}(u). \end{aligned} \tag{11}$$

We define the corresponding per-example interval exposure as

$$\bar{E}^s_j(a,b) = \sum_{u=a+1}^{b} q_{j,s}(u). \tag{12}$$

Eq. 11 shows that the accumulated temporal variation over a reuse interval upper-bounds the prediction drift spanned by that interval along the reference full-computation trajectory. Consequently, for a complete pair of stage schedules (C_I, C_D),

$$\max_{s,\ell,t} \left\| p^s_{j,t} - p^s_{j,c^s_\ell} \right\|_1 \le \max_{s,\ell} \bar{E}^s_{j,\ell}(C_s), \tag{13}$$

where t ranges over the steps covered by the corresponding reuse interval.

This relation motivates controlling the largest exposure: if one reuse interval spans substantially more temporal variation than the others, the stale prediction associated with that interval must cover a more strongly varying portion of the prediction trajectory.

Why exposure concentration matters. Let

$$n_s = b_s - 1 \tag{14}$$

denote the number of reuse intervals in stage s. Since $b_I + b_D = B$, the total number of reuse intervals across inversion and denoising is

$$K = n_I + n_D = B - 2. \tag{15}$$

We define the total temporal variation of stage s as

$$V_s = \sum_{i=1}^{N_s-1} r_s(i), \tag{16}$$

and the total variation over both stages as

$$V = V_I + V_D. \tag{17}$$

Because the reuse intervals form a partition of the two stage trajectories, their exposures always satisfy

$$\sum_{s\in\{I,D\}} \sum_{\ell=1}^{n_s} E_\ell^s = V. \tag{18}$$

Therefore, under a fixed editing trajectory and a fixed global budget, the total exposure mass V is unavoidable and cannot be reduced by refresh scheduling. In particular, a linear objective such as $\sum_{s,\ell} E_\ell^s$ is constant for all feasible reuse plans. What refresh planning can control is instead how this fixed temporal variation is distributed across the K reuse intervals. The optimization problem is therefore fundamentally one of controlling exposure concentration.

A minimax interpretation of temporal variation exposure. A conservative way to control exposure concentration is to minimize the largest interval exposure:

$$\min_{C_I,C_D} \max_{s,\ell} E_\ell^s(C_s). \tag{19}$$

This objective directly targets the interval in which one stale prediction must span the largest amount of temporal prediction variation, and serves as the robust formulation underlying the TVE objective.

Continuous minimax optimum. Since the K reuse intervals have a fixed total exposure V, the maximum exposure cannot be smaller than their average:

$$\max_{s,\ell} E_\ell^s \geq \frac{1}{K} \sum_{s,\ell} E_\ell^s = \frac{V}{K}, \tag{20}$$

where equality holds if and only if

$$E_\ell^s = \frac{V}{K}, \qquad \forall s, \ell. \tag{21}$$

Therefore, the continuous minimax optimum corresponds to global equal exposure across all reuse intervals. This condition provides the common principle underlying both within-stage refresh scheduling and cross-stage budget allocation, which we derive in the following sections.

C.2 From Minimax Exposure to Quadratic TVE

The minimax formulation identifies global equal exposure as the ideal solution. However, actual inversion and denoising trajectories consist of a finite set of timesteps. Refresh locations must lie on this discrete grid, and exact global equal exposure is therefore generally unattainable.

Let

$$\mu = \frac{V}{K} \tag{22}$$

denote the continuous minimax exposure, and collect all interval exposures into a vector

$$\mathbf{E} = (E_1, \ldots, E_K). \tag{23}$$

The ideal minimax solution is

$$\mathbf{E}^\star = \mu \mathbf{1}. \tag{24}$$

A natural discrete counterpart is to select the feasible exposure configuration closest to this continuous equal-exposure solution. Using the squared Euclidean distance,

$$\begin{aligned} \|\mathbf{E} - \mathbf{E}^\star\|_2^2 &= \sum_{k=1}^{K} (E_k - \mu)^2 \\ &= \sum_{k=1}^{K} E_k^2 - 2\mu \sum_{k=1}^{K} E_k + K\mu^2. \end{aligned} \tag{25}$$

Since

$$\sum_{k=1}^{K} E_k = V, \qquad \mu = \frac{V}{K}, \tag{26}$$

we obtain

$$\|\mathbf{E} - \mathbf{E}^\star\|_2^2 = \sum_{k=1}^{K} E_k^2 - \frac{V^2}{K}. \tag{27}$$

The second term is constant for a fixed global budget. Therefore,

$$\arg\min_{\mathbf{E}} \|\mathbf{E} - \mathbf{E}^\star\|_2^2 = \arg\min_{\mathbf{E}} \sum_{k=1}^{K} E_k^2. \tag{28}$$

This gives the quadratic TVE objective used by **DARTCache**:

$$\mathcal{J}(C_I, C_D) = \sum_{s\in\{I,D\}} \sum_{\ell} \left(E_\ell^s(C_s)\right)^2. \tag{29}$$

Quadratic TVE as exposure imbalance. Combining Eq. 27 and Eq.29, we can rewrite the objective as

$$\mathcal{J}(C_I, C_D) = \frac{V^2}{K} + \left\|\mathbf{E} - \frac{V}{K}\mathbf{1}\right\|_2^2 = \frac{V^2}{K} + K\ \mathrm{Var}(\mathbf{E}). \tag{30}$$

Here the first term, V^2/K, is fixed for a given global budget, while the second term measures the dispersion of interval exposures around the equal-exposure solution. Therefore, minimizing quadratic TVE is equivalent to minimizing the variance of interval exposure. The quadratic objective can thus be interpreted as a concentration penalty that preserves the same continuous equal-exposure optimum as minimax TVE while providing a finer ranking over discrete schedules.

Relation to worst-case exposure. The quadratic formulation also continues to control the worst-exposed interval. Since

$$\max_k (E_k - \mu) \le \|\mathbf{E} - \mu\mathbf{1}\|_2, \tag{31}$$

Eq. 30 gives

$$\mathcal{M}(C_I, C_D) - \frac{V}{K} \le \sqrt{\mathcal{J}(C_I, C_D) - \frac{V^2}{K}}. \tag{32}$$

Therefore, reducing the excess quadratic TVE also limits how far the worst-exposed reuse interval can deviate from the continuous minimax lower bound.

Why not optimize the discrete maximum directly? The minimax objective remains valid on the discrete trajectory, but it provides a relatively coarse ordering among candidate schedules because it depends only on the largest interval exposure. For example, $\mathbf{E}_A = (4, 4, 4, 8)$ and $\mathbf{E}_B = (2, 5, 5, 8)$ have the same maximum exposure of 8, although $\mathbf{E}_A$ is closer to the global equal-exposure solution.

Quadratic TVE resolves this ambiguity by penalizing the deviation of all interval exposures from equal exposure. Importantly, this does not change the continuous optimum: both minimax exposure and quadratic TVE are minimized by the same equal-exposure configuration. The quadratic form therefore provides a finer criterion for ranking feasible schedules when exact equal exposure cannot be realized on the discrete timestep grid.

More generally, any strictly convex symmetric penalty under a fixed exposure sum favors equal exposure. We adopt the quadratic form because it admits the exact decomposition in Eq. 30, provides a simple scalar objective over discrete schedules, and preserves the same continuous optimum as the minimax exposure formulation.

C.3 Exposure-Balanced Refresh Scheduling

We first consider refresh placement within a single stage s under a fixed stage-wise budget b_s. Let $n_s = b_s - 1$ denote the number of reuse intervals. Since these intervals partition the entire stage trajectory, their exposures satisfy

$$\sum_{\ell=1}^{n_s} E_\ell^s = V_s, \tag{33}$$

where $V_s = \sum_{i=1}^{N_s-1} r_s(i)$ is the total temporal variation of the stage.

For a fixed n_s, the stage-wise quadratic TVE satisfies

$$\sum_{\ell=1}^{n_s}(E_\ell^s)^2 \geq \frac{1}{n_s}\left(\sum_{\ell=1}^{n_s} E_\ell^s\right)^2 = \frac{V_s^2}{n_s}, \tag{34}$$

where the inequality follows from Cauchy–Schwarz. Equality holds if and only if all reuse intervals have identical exposure, giving

$$E_\ell^s = \frac{V_s}{n_s}, \qquad \ell = 1, \ldots, n_s. \tag{35}$$

Therefore, under a fixed stage budget, minimizing quadratic TVE reduces to partitioning the trajectory into intervals with equal accumulated temporal variation.

To realize this condition, define the normalized cumulative temporal variation

$$F_s(i) = \frac{\sum_{u=1}^{i} r_s(u)}{\sum_{u=1}^{N_s-1} r_s(u)} = \frac{1}{V_s}\sum_{u=1}^{i} r_s(u). \tag{36}$$

If each of the n_s intervals carries exposure V_s/n_s, then the ℓ-th internal refresh boundary should accumulate exactly an ℓ/n_s fraction of the total stage variation:

$$F_s(c_\ell^s) = \frac{\ell}{n_s}, \qquad \ell = 1, \ldots, n_s - 1. \tag{37}$$

This directly yields

$$c_\ell^s = F_s^{-1}\left(\frac{\ell}{n_s}\right), \qquad \ell = 1, \ldots, n_s - 1. \tag{38}$$

Hence, the cumulative-variation quantile schedule used by **DARTCache** follows directly from the equal-exposure optimum. Because refresh locations are restricted to discrete timesteps, exact equal exposure may not always be attainable; the quantile construction therefore serves as the discrete realization of the continuous equal-exposure solution.

C.4 Exposure-Aware Stage Budget Allocation

The same equal-exposure principle also determines how the global refresh budget should be divided between inversion and denoising.

For stage s with n_s reuse intervals, Eq. 30 immediately gives

$$\mathcal{J}_s \geq \frac{V_s^2}{n_s}, \tag{39}$$

with equality when exposure is perfectly balanced within that stage. Consequently, under ideal within-stage equalization, the global objective reduces to

$$\mathcal{J} = \frac{V_I^2}{n_I} + \frac{V_D^2}{n_D}, \qquad n_I + n_D = K. \tag{40}$$

Applying the Cauchy–Schwarz inequality,

$$\begin{aligned}(V_I + V_D)^2 &= \left(\frac{V_I}{\sqrt{n_I}}\sqrt{n_I} + \frac{V_D}{\sqrt{n_D}}\sqrt{n_D}\right)^2 \\ &\leq (n_I + n_D)\left(\frac{V_I^2}{n_I} + \frac{V_D^2}{n_D}\right).\end{aligned} \tag{41}$$

Since $n_I + n_D = K$,

$$\frac{V_I^2}{n_I} + \frac{V_D^2}{n_D} \geq \frac{(V_I + V_D)^2}{K} = \frac{V^2}{K}. \tag{42}$$

Equality holds when

$$\frac{V_I}{n_I} = \frac{V_D}{n_D}. \tag{43}$$

Therefore,

$$\frac{n_I}{n_D} = \frac{V_I}{V_D}, \tag{44}$$

and equivalently

$$n_s^\star = K\frac{V_s}{V_I + V_D}. \tag{45}$$

Recalling that $n_s = b_s - 1$, we obtain

$$b_s^\star - 1 \propto V_s. \tag{46}$$

This result has a direct interpretation: a stage with larger total temporal variation should receive more refresh intervals, so that the average temporal variation carried by each reuse interval remains balanced across inversion and denoising.

A unified lower bound. Combining the within-stage and cross-stage inequalities gives

$$\boxed{\mathcal{J} \geq \frac{V_I^2}{n_I} + \frac{V_D^2}{n_D} \geq \frac{(V_I + V_D)^2}{K}.} \tag{47}$$

The two equality conditions correspond exactly to the two planning decisions of **DARTCache**.

The first inequality becomes tight when

$$E_1^s = \cdots = E_{n_s}^s = \frac{V_s}{n_s}, \tag{48}$$

which corresponds to exposure-balanced refresh scheduling *within* each stage.

The second inequality becomes tight when

$$\frac{V_I}{n_I} = \frac{V_D}{n_D}, \tag{49}$$

which corresponds to exposure-balanced computation allocation *across* the two stages.

Hence, refresh scheduling and stage budget allocation are not two independent heuristics. They are the two equality conditions required to attain the same global equal-exposure lower bound.

Discrete stage allocation. The continuous solution in Eq. 45 need not correspond to valid integer budgets, and exact equal exposure may not be attainable on the discrete timestep grid. **DARTCache** therefore does not directly round the continuous solution. Instead, for every feasible integer split (b_I, b_D) satisfying

$$b_I + b_D = B, \tag{50}$$

it first constructs the corresponding exposure-balanced quantile schedules $C_I(b_I)$ and $C_D(b_D)$ and then evaluates their actual discrete quadratic TVE:

$$(b_I^\star, b_D^\star) = \arg\min_{b_I + b_D = B} \mathcal{J}\left(C_I(b_I), C_D(b_D)\right). \tag{51}$$

The selected stage split is therefore the exact optimum over all feasible integer allocations within the structured family of exposure-balanced quantile schedules considered by **DARTCache**.

D Additional Experimental Analysis

D.1 Cross-Stage Scale Ablation

DARTCache preserves the native magnitude of temporal variation when allocating the global refresh budget across inversion and denoising. To examine whether this cross-stage scale is informative, we compare Equal Allocation, Stage-Normalized TVE, and the default Native-scale TVE under $B \in \{12, 20, 28, 36\}$ on both RF-Inversion and OT-RF.

Table 6: **Cross-stage scale ablation on RF-Inversion and OT-RF.** Native-scale TVE achieves the lowest $\text{LPIPS}_{\text{All}}$ across all tested budgets.

(a) RF-Inversion

B	Variant	b_I/b_D	$\text{PSNR}_{\text{All}}\uparrow$	$\text{LPIPS}_{\text{All}}\downarrow$	$\text{PSNR}_M\uparrow$	$\text{LPIPS}_M\downarrow$
12	Equal Allocation	6/6	20.6133	0.3294	17.8809	0.2058
12	Stage-Normalized TVE	6/6	20.6133	0.3294	17.8809	0.2058
12	**Native-scale TVE**	**3/9**	**22.6713**	**0.2422**	**20.0566**	**0.1549**
20	Equal Allocation	10/10	25.5217	0.1416	22.6298	0.0957
20	Stage-Normalized TVE	10/10	25.5217	0.1416	22.6298	0.0957
20	**Native-scale TVE**	**6/14**	**26.8982**	**0.1071**	**24.0576**	**0.0777**
28	Equal Allocation	14/14	28.0032	0.0877	25.4267	0.0593
28	Stage-Normalized TVE	14/14	28.0032	0.0877	25.4267	0.0593
28	**Native-scale TVE**	**8/20**	**30.2869**	**0.0603**	**27.8462**	**0.0403**
36	Equal Allocation	18/18	30.5434	0.0553	28.6747	**0.0351**
36	Stage-Normalized TVE	19/17	30.4463	0.0548	27.9501	0.0374
36	**Native-scale TVE**	**12/24**	**32.5377**	**0.0468**	**29.8159**	0.0393

(b) OT-RF

B	Variant	b_I/b_D	$\text{PSNR}_{\text{All}}\uparrow$	$\text{LPIPS}_{\text{All}}\downarrow$	$\text{PSNR}_M\uparrow$	$\text{LPIPS}_M\downarrow$
12	Equal Allocation	6/6	17.0580	0.3805	16.2478	0.2038
12	Stage-Normalized TVE	6/6	17.0580	0.3805	16.2478	0.2038
12	**Native-scale TVE**	**4/8**	**22.5477**	**0.2110**	**21.4540**	**0.1098**
20	Equal Allocation	10/10	23.4609	0.1484	22.4227	0.0768
20	Stage-Normalized TVE	10/10	23.4609	0.1484	22.4227	0.0768
20	**Native-scale TVE**	**9/11**	**24.6725**	**0.1374**	**23.7038**	**0.0706**
28	Equal Allocation	14/14	24.5543	0.1172	23.6552	0.0602
28	Stage-Normalized TVE	13/15	24.8935	0.1127	23.9976	0.0581
28	**Native-scale TVE**	**12/16**	**25.9119**	**0.1089**	**25.0925**	**0.0558**
36	Equal Allocation	18/18	25.6789	0.0976	24.8644	0.0504
36	Stage-Normalized TVE	17/19	25.7507	0.0958	24.9700	0.0493
36	**Native-scale TVE**	**16/20**	**26.8415**	**0.0925**	**26.1110**	**0.0475**

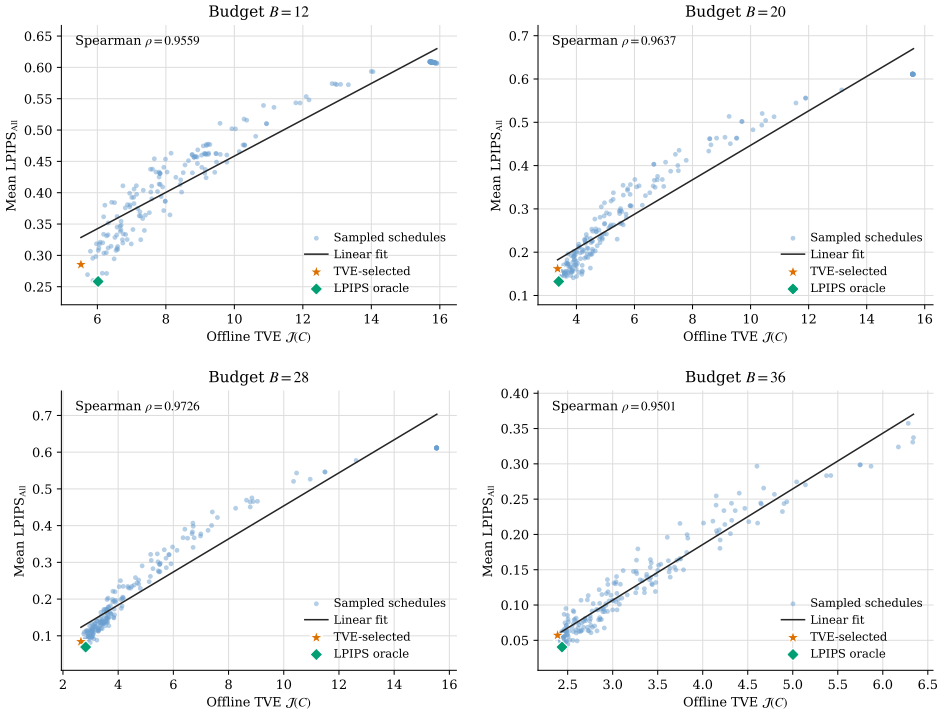


Figure 7: **TVE objective validation across budgets.** Each point represents a sampled refresh schedule, with empirical LPIPS degradation averaged over 50 prompts. TVE maintains a strong monotonic correlation with fidelity degradation across different computation budgets.

Stage-Normalized TVE removes the relative scale between the two stages by minimizing

$$\frac{\mathcal{J}_I}{V_I^2} + \frac{\mathcal{J}_D}{V_D^2},$$

whereas Native-scale TVE directly minimizes $\mathcal{J}_I + \mathcal{J}_D$. For any fixed stage budget, all variants use exactly the same quantile-based refresh schedule; hence, the comparison changes only the cross-stage allocation criterion. Schedules are selected using the 70-case calibration split and frozen before evaluation on the remaining 630 cases. For RF-Inversion, the calibrated denoising trajectory has approximately $3.99\times$ larger total variation than inversion ($V_D = 5.335$ versus $V_I = 1.337$); OT-RF is calibrated independently using its own native stage-wise variation magnitudes.

Table 6 shows that preserving the native cross-stage scale consistently favors more computation for denoising. For RF-Inversion, Native-scale TVE selects inversion–denoising splits of $3/9$, $6/14$, $8/20$, and $12/24$ as the global budget increases. For OT-RF, the corresponding splits are $4/8$, $9/11$, $12/16$, and $16/20$. In contrast, Equal Allocation fixes balanced splits, while Stage-Normalized TVE remains substantially closer to equal allocation.

This allocation difference translates into improved fidelity on both hosts. On RF-Inversion, Native-scale TVE achieves the lowest $\text{LPIPS}_{\text{All}}$ at all four budgets, reducing it by approximately 14.6%–31.2% relative to Stage-Normalized TVE. It also achieves the highest PSNR_{All} and PSNR_M throughout. At $B = 36$, it exhibits a small masked-LPIPS trade-off (0.0393 versus 0.0374 for Stage-Normalized TVE), while retaining the best whole-image fidelity.

The same trend holds on OT-RF. Native-scale TVE achieves the lowest $\text{LPIPS}_{\text{All}}$ and LPIPS_M and the highest PSNR_{All} and PSNR_M at every tested budget. The improvement is especially pronounced at $B = 12$, where preserving the native scale changes the allocation from $6/6$ to $4/8$ and reduces $\text{LPIPS}_{\text{All}}$ from 0.3805 to 0.2110.

Overall, the consistent gains across two independently calibrated hosts indicate that the relative magnitude of temporal variation across stages contains useful information for inversion–denoising budget allocation, rather than being an RF-Inversion-specific artifact.

D.2 Objective and Solver Validation

Alternative objective comparison. To validate the design of the proposed TVE objective, we compare it with three alternative offline objectives for evaluating candidate refresh schedules: Endpoint, Integrated, and Lexi-Minimax. All objectives are computed from the same frozen calibration trajectories and are used to rank the same set of candidate schedules. For validating the refresh scheduling objective, we fix B=20 and the inversion-denoising split to 10/10. Only refresh locations within each stage are randomly sampled.

For a candidate schedule C with reuse intervals $\{(a_l, b_l)\}_{l=1}^{L}$, let E_l denote the TVE exposure of the l-th interval. We compare the following objectives:

The *Endpoint* objective focuses on the accumulated prediction discrepancy at the interval boundary:

$$J_{\text{end}}(C) = \sum_{l=1}^{L} \|p_{b_l} - p_{a_l}\|_1 .$$

The *Integrated* objective considers the temporal evolution within each reuse interval by aggregating intermediate discrepancies:

$$J_{\text{int}}(C) = \sum_{l=1}^{L} \sum_{t=a_l+1}^{b_l} \|p_t - p_{a_l}\|_1 .$$

The *Lexi-Minimax* criterion ranks schedules by their descending-sorted exposure vectors, preferring the lexicographically smaller vector:

$$\mathbf{J}_{\text{lexi}}(C) = \text{sort}_{\downarrow}(E_1, \ldots, E_L).$$

In contrast, the proposed *Quadratic TVE* objective minimizes the sum of squared interval exposures:

$$J_{\text{TVE}}(C) = \sum_{l=1}^{L} E_l^2,$$

which encourages globally balanced exposure across all reuse intervals.

We evaluate these objectives by comparing their rankings against empirical fidelity degradation measured by LPIPS. Detailed results are reported in Table 7.

As shown in Table 7, Quadratic TVE consistently achieves the highest Spearman correlation across both RF-Inversion and OT-RF over all tested computation budgets. For RF-Inversion, the correlation remains above 0.95 for all budgets, while for OT-RF it ranges from 0.9262 to 0.9793. In contrast, endpoint-based objectives show substantially weaker ranking ability, especially under aggressive compression budgets, while Integrated and Lexi-Minimax provide stronger but less consistent correlations.

Top-k regret is the minimum empirical LPIPS among the objective-ranked top-k candidates minus the minimum LPIPS over the full candidate pool. Quadratic TVE attains the lowest reported Top-1 regret, including ties, across the tested settings. Top-k regret measures shortlist quality using

Table 7: **Comparison of alternative offline objectives across hosts and computation budgets.** Each entry reports Spearman correlation ρ with empirical fidelity degradation or Top-1 LPIPS regret Δ_{Top1}. All objectives share the same candidate schedule pool and calibration profiles for each host.

RF-Inversion					OT-RF				
Objective	$B=12$	$B=20$	$B=28$	$B=36$	Objective	$B=12$	$B=20$	$B=28$	$B=36$
(a) Spearman correlation $\rho\uparrow$									
Endpoint	0.1336	0.6433	0.7945	0.8399	Endpoint	-0.0417	0.3234	0.5785	0.6621
Integrated	0.8738	0.7695	0.8381	0.8978	Integrated	0.7128	0.6122	0.7471	0.8149
Lexi-Minimax	0.9326	0.9316	0.9383	0.9247	Lexi-Minimax	0.8425	0.8148	0.8592	0.8994
Quadratic TVE	**0.9559**	**0.9637**	**0.9726**	**0.9501**	**Quadratic TVE**	**0.9262**	**0.9400**	**0.9454**	**0.9793**
(b) Top-1 LPIPS regret $\Delta_{\mathrm{Top1}}\downarrow$									
Endpoint	0.1882	0.0501	0.0147	0.0227	Endpoint	0.3058	0.1258	0.0745	0.0400
Integrated	0.0397	0.0194	0.0147	0.0183	Integrated	0.1057	0.0812	0.0186	0.0586
Lexi-Minimax	**0.0271**	0.0294	0.0147	**0.0166**	Lexi-Minimax	**0.0547**	0.0789	**0.0113**	**0.0443**
Quadratic TVE	**0.0271**	**0.0116**	**0.0147**	**0.0166**	**Quadratic TVE**	**0.0547**	**0.0758**	**0.0113**	**0.0443**

Table 8: **Controlled profile-signal ablation at $B=20$.** All variants use fixed stage allocations and identical scheduling constraints.

Profile	$\mathrm{PSNR}_{\mathrm{All}}\uparrow$	$\mathrm{LPIPS}_{\mathrm{All}}\downarrow$	$\mathrm{PSNR}_{\mathrm{M}}\uparrow$	$\mathrm{LPIPS}_{\mathrm{M}}\downarrow$
RF-Inversion, $b_I/b_D=6/14$				
Uniform	23.8509	0.1673	19.8996	0.1427
Time	22.2416	0.1959	18.0098	0.1692
Asymmetric prior	22.0614	0.2474	19.7034	0.1589
Shuffle	23.4400 ± 0.4187	0.1796 ± 0.0088	19.8011 ± 0.5087	0.1461 ± 0.0057
True variation	**26.8982**	**0.1071**	**24.0576**	**0.0777**
OT-RF, $b_I/b_D=9/11$				
Uniform	17.5130	0.4021	16.5013	0.2099
Time	17.7913	0.3866	16.7516	0.2022
Asymmetric prior	21.0006	0.2594	19.9221	0.1356
Shuffle	18.5645 ± 2.0345	0.3547 ± 0.1120	17.5372 ± 1.9895	0.1853 ± 0.0585
True variation	**24.6725**	**0.1374**	**23.7038**	**0.0706**

observed outcomes; it does not imply that the objective alone identifies the best candidate within that shortlist.

Overall, the results across two inversion trajectories and multiple budgets demonstrate that temporal variation exposure provides a reliable surrogate objective for refresh schedule evaluation.

Controlled profile-signal ablation. To isolate the information provided by the scheduling signal, we compare alternative placement profiles at $B=20$, while fixing the inversion–denoising split to $(6,14)$ for RF-Inversion and $(9,11)$ for OT-RF. Within each host, all variants use the same cache target, inference procedure, and cumulative-quantile schedule constructor. The stage allocation is not re-optimized for individual profiles.

We compare five scheduling signals:

$$w_s^{\mathrm{Uniform}}(i)=1, \qquad w_s^{\mathrm{Time}}(i)=|\tau_s(i)-\tau_s(i-1)|,$$
$$w_I^{\mathrm{Asym}}(i)=2u_I(i), \qquad w_D^{\mathrm{Asym}}(i)=2(1-u_D(i)),$$
$$w_s^{\mathrm{Shuffle},m}(i)=r_s(\pi_s^{(m)}(i)), \qquad w_s^{\mathrm{True}}(i)=r_s(i),$$

where $u_s(i)=(i-\frac{1}{2})/(N_s-1)$, $\tau_s(i)$ is the normalized native scheduler timestep at execution position i. Uniform, Time, and Asymmetric use no prediction-derived variation. For Shuffle, we independently permute the frozen calibrated variation within each stage using five fixed seeds, preserving its value distribution and total variation while destroying temporal alignment. All profiles are converted to refresh schedules using the same constructor and constraints.

Table 9: **Single-stage applicability of TVE scheduling on FLUX.1-dev.** Comparison with representative caching baselines under matched computation budgets.

Method	PSNR ↑	LPIPS ↓	Latency (s) ↓	Speedup ↑
Full	∞	0	6.598	1.00×
Uniform	17.574	0.33787	2.885	2.29×
TeaCache	19.441	0.28009	2.890	2.28×
SeaCache	25.490	0.14132	2.919	2.26×
SenCache	26.286	0.13811	2.860	2.31×
TVE	**26.846**	**0.12558**	2.885	2.29×

Table 10: **Quantile solver validation.** Comparison of quantile scheduling and discrete dynamic programming optimization under different computation budgets.

B	TVE Gap	Uniform	Quantile	Exact DP
12	2.29%	0.51598	**0.34434**	0.34659
20	1.25%	0.28370	**0.14544**	0.15082
28	1.83%	0.16704	0.08796	**0.08633**
36	2.56%	0.09603	**0.05577**	0.06388

As shown in Table 8, True variation achieves the lowest whole-image and edited-region LPIPS on both hosts. Compared with the strongest tested deterministic alternative, $\text{LPIPS}_{\text{All}}$ decreases from 0.1673 to 0.1071 on RF-Inversion and from 0.2594 to 0.1374 on OT-RF. True variation also outperforms all five shuffled profiles. These results indicate that the gain is not explained by generic non-uniform scheduling alone: the temporal alignment of prediction variation provides useful information for refresh placement when allocation and cache execution are held fixed.

TVE fidelity correlation across budgets. We first examine whether the relationship between TVE and empirical fidelity degradation is specific to the default operating point. For each global budget $B \in \{12, 20, 28, 36\}$, we randomly sample 200 valid joint refresh schedules and evaluate each schedule on the same 50 held-out prompts, resulting in 40,000 schedule–prompt measurements in total. For every schedule, TVE is computed solely from the frozen calibration profiles, while empirical degradation is measured by $\text{LPIPS}_{\text{All}}$ against Full inference.

As shown in Fig. 7, TVE maintains a strong monotonic relationship with empirical fidelity degradation across all tested budgets. The Spearman correlations are 0.9559, 0.9637, 0.9726, and 0.9501 for $B = 12, 20, 28, 36$, respectively, all exceeding 0.95. This indicates that the ranking ability of TVE remains stable across substantially different compression levels. Although TVE is not optimized to exactly predict downstream perceptual error, its consistent ranking behavior across budgets supports its use as a reliable surrogate objective for refresh schedule selection.

Single-stage applicability of TVE scheduling. Although **DARTCache** is designed for dual-stage inversion-based editing, the proposed TVE principle only relies on temporal prediction variation and does not require stage coupling. To evaluate whether TVE can serve as a general refresh scheduling criterion, we apply TVE as a standalone scheduling strategy to single-stage FLUX.1-dev denoising and compare it with representative cache baselines, including Uniform, TeaCache, SeaCache, and SenCache. All methods are evaluated under matched computation budgets using the same FLUX model and PartiPrompts benchmark.

As shown in Table 9, TVE substantially improves fidelity over conventional cache scheduling strategies under comparable computation budgets. At 12 refresh calls out of 28 denoising steps, TVE achieves higher PSNR and lower LPIPS than Uniform, TeaCache, SeaCache, and SenCache, while maintaining similar inference acceleration. These results indicate that temporal variation exposure provides a more effective criterion for determining refresh locations than uniform or threshold-based reuse strategies.

Quantile scheduling versus exact TVE optimization. We next test whether the cumulative-variation quantile construction used by DARTCache is a sufficiently accurate approximation to exact discrete optimization of the quadratic TVE objective. For each budget $B \in \{12, 20, 28, 36\}$, we fix the stage budgets to $b_I = b_D = B/2$ and compare three within-stage scheduling strategies: uniform placement, the proposed quantile construction, and exact dynamic programming (DP). All three solvers are evaluated on the same 50 prompts. The DP solver exactly minimizes the discrete TVE objective under these constraints and therefore serves as a surrogate-objective oracle.

As summarized in Table 10, Quantile closely approximates the exact discrete optimum: its TVE objective is only 1.25%–2.56% higher than DP across the four budgets. This small surrogate gap is also reflected in empirical fidelity. Quantile achieves lower $\text{LPIPS}_{\text{All}}$ than DP at $B = 12, 20, 36$, while DP is only 0.0016 better at $B = 28$. Both TVE-based solvers substantially outperform uniform

scheduling at every budget. Since DP optimizes TVE rather than LPIPS directly, exact optimality in the surrogate objective does not imply exact optimality in downstream image fidelity.

Quantile construction requires only 0.031–0.048 ms of offline planning, compared with 0.524–1.155 ms for DP. Both costs are negligible relative to image generation; therefore, the main advantage of Quantile is not runtime, but its simple and interpretable construction while retaining nearly the same surrogate and empirical performance as exact TVE optimization. Together, these results validate both components of the planning procedure: TVE provides a stable surrogate for ranking refresh schedules, while the quantile construction closely approximates exact optimization of that surrogate without requiring empirical fidelity feedback.

D.3 Comparison with LeMiCa-style Global Planning

Recent offline cache planning methods, such as LeMiCa Gao et al. (2025), formulate cache scheduling as a global optimization problem over cache segments. Specifically, LeMiCa constructs a graph where each edge corresponds to a candidate cache segment and is assigned an outcome-aware error estimated from cache-induced output deviation. A lexicographic minimax objective is then used to select a cache path under a given computation budget. Therefore, LeMiCa provides a more direct measurement of cache quality by evaluating the impact of individual reuse decisions on the final output, but requires additional candidate-level evaluations during offline planning.

However, directly applying LeMiCa to inversion-based image editing is non-trivial. Unlike conventional single-trajectory diffusion generation, inversion-based editing consists of two coupled iterative processes: inversion and denoising. The total computation budget is shared between these two stages, requiring a joint decision on both where predictions should be refreshed within each stage and how the budget should be allocated across stages. Therefore, we extend the original LeMiCa formulation to the dual-stage setting by optimizing two stage-specific paths under a shared global budget.

Specifically, we construct two stage-specific cache graphs corresponding to the inversion and denoising trajectories, respectively. Each edge (i, j) represents a reuse segment that performs a fresh model evaluation at step i and reuses the cached prediction until step j. The edge cost is defined as the terminal deviation introduced by this reuse segment compared with the corresponding full-computation trajectory.

For each candidate segment (i, j), we refresh at step i, reuse the cached prediction at intermediate steps, and resume full computation from step j until the end of the corresponding stage. The edge cost is computed as the mean absolute deviation between the resulting stage-terminal packed latent and the full-computation stage-terminal packed latent, averaged over the 70 calibration cases.

For inversion-stage segments, the trajectory terminates at the end of inversion before denoising. For denoising-stage segments, the trajectory starts from the full-inversion latent and terminates before VAE decoding. Therefore, both stage costs are measured in latent space rather than final image space. The raw edge costs from the two stages are merged in the lexicographic optimization without cross-stage normalization. Although both costs are measured in latent space, their relative magnitudes may affect the resulting budget allocation. Therefore, this comparison evaluates the specific LeMiCa-style adaptation implemented here, rather than a scale-invariant outcome-aware planner.

Given a total fresh-evaluation budget B, we enumerate feasible stage-wise allocations (B_I, B_D) satisfying $B_I + B_D = B$. For each allocation, we independently solve the lexicographic minimax path optimization on the two stage-specific graphs and combine the resulting segment errors across the two stages. The two stage-wise deviations are combined following the same scale used in our implementation. The final schedule is selected by minimizing the global objective over all candidate budget splits. This adaptation preserves the outcome-aware global planning principle of LeMiCa while extending it to the coupled inversion–denoising setting.

To reduce the search space and maintain consistency with the segment-based planning formulation, we restrict the maximum reuse interval length to $L_{\max} = 8$ when constructing the LeMiCa-style graphs. In contrast, DARTCache does not require explicit segment enumeration or a predefined maximum interval, since it derives schedules directly from trajectory-level temporal variation statistics.

Table 11: **Selected reuse schedules under $B = 20$.** Both planners independently select the same 6/14 inversion–denoising budget split, while producing similar refresh patterns.

Planner	Budget split	Inversion refresh steps	Denoising refresh steps
LeMiCa-style	6 / 14	$\{0, 8, 16, 24, 25, 26\}$	$\{0, 1, 2, 3, 4, 5, 6, 7, 9, 12, 17, 20, 24, 27\}$
DARTCache	6 / 14	$\{0, 5, 15, 24, 25, 26\}$	$\{0, 1, 2, 3, 4, 6, 7, 8, 11, 14, 18, 22, 25, 27\}$

Table 12: **Comparison with LeMiCa-style global planning.** Both methods use the same computation budget ($B = 20$). Online quality is evaluated against full RF-Inversion outputs, and offline calibration cost is measured on the 70-case calibration set.

	Online Quality						Offline Calibration Cost	
Method	$\text{PSNR}_{\text{M}} \uparrow$	$\text{LPIPS}_{\text{M}} \downarrow$	$\text{PSNR}_{\text{BG}} \uparrow$	$\text{LPIPS}_{\text{BG}} \downarrow$	$\text{PSNR}_{\text{All}} \uparrow$	$\text{LPIPS}_{\text{All}} \downarrow$	Calls $\downarrow$	Time $\downarrow$
LeMiCa-style	**24.20**	**0.0762**	31.41	0.0449	26.91	0.1103	261,590	2974.5s
DARTCache	24.06	0.0777	**31.94**	**0.0387**	26.90	**0.1071**	**3,850**	**67.1s**

DARTCache follows a fundamentally different planning paradigm. Instead of estimating the quality of each candidate reuse segment through additional cache replay, DARTCache extracts schedule-independent temporal variation profiles from full-computation trajectories and evaluates candidate schedules using Temporal Variation Exposure (TVE). The resulting refresh schedules and stage-wise budget allocation are obtained through exposure-balanced scheduling and budget enumeration without requiring candidate-level segment evaluation.

To further examine whether the two planners identify substantially different reuse strategies, we report the selected schedules under the same online computation budget in Table 11. Both planners independently optimize the stage-wise budget allocation and select the same inversion–denoising split of 6/14. Although the exact refresh locations differ due to different planning objectives, the resulting schedules exhibit similar temporal coverage patterns, indicating that, in this evaluated setting, trajectory-level statistics identify reuse structures similar to those obtained by outcome-aware planning.

Table 12 compares DARTCache with the adapted LeMiCa-style planner in terms of reconstruction quality and offline planning cost. Under the same online computation budget, the two approaches achieve comparable fidelity across editing regions, background regions, and the full image. The LeMiCa-style planner obtains slightly better fidelity on the editing region, while DARTCache achieves comparable overall reconstruction quality with substantially lower offline planning overhead.

The difference mainly comes from the offline planning procedure. The LeMiCa-style planner requires evaluating individual cache segments through additional replay trajectories to estimate stage-terminal deviations. For the 70-case calibration set, it evaluates 20,090 candidate cache segments, which introduces 257,740 additional Transformer evaluations beyond the 3,850 full-trajectory evaluations, resulting in 261,590 total Transformer evaluations. In contrast, DARTCache only requires the original full trajectories and temporal profile construction, resulting in 3,850 Transformer evaluations in total.

Both offline costs are measured using the same 70-sample calibration set. For DARTCache, the cost includes full-trajectory extraction, temporal profile construction, and schedule selection. For the LeMiCa-style planner, the cost additionally includes candidate segment replay and terminal deviation estimation.

Under this measured implementation protocol, the complete calibration process of the LeMiCa-style planner takes approximately 44.3$\times$ longer than DARTCache. These results demonstrate that DARTCache achieves competitive scheduling quality while substantially reducing offline planning overhead by avoiding repeated candidate-level replay evaluations.

Table 13: **Bootstrap calibration robustness.**

Calibration	b_I/b_D	$\text{PSNR}_{\text{All}} \uparrow$	$\text{LPIPS}_{\text{All}} \downarrow$
Default	6/14	26.8982	0.1071
Bootstrap 1	5/15	26.6293	0.1125
Bootstrap 2	6/14	26.6723	0.1094
Bootstrap 3	6/14	26.8267	0.1079
Bootstrap 4	6/14	26.9460	0.1067
Bootstrap 5	6/14	26.6863	0.1096
Mean	–	26.7521	0.1092
Std.	–	0.1313	0.0022

Table 14: **Leave-one-category-out transfer.** Δ denotes LOCO minus default calibration.

Category	Eval.	Δ $\text{LPIPS}_{\text{All}} \downarrow$	Δ $\text{PSNR}_{\text{All}} \uparrow$
Random	123	0.0000	0.0000
Change Object	73	-0.0013	-0.1521
Add Object	73	0.0000	0.0000
Delete Object	72	-0.0004	$+0.0290$
Attribute Content	36	0.0000	0.0000
Attribute Pose	35	$+0.0025$	$+0.2195$
Attribute Color	33	$+0.0055$	-0.4153
Attribute Material	36	$+0.0036$	-0.1280
Change Background	73	0.0000	0.0000
Change Style	76	0.0000	0.0000
Mean	–	$+0.0010$	-0.0447

D.4 Calibration Robustness and Transfer

We evaluate the sensitivity of DARTCache to both calibration-set composition and editing-category coverage. First, starting from the fixed 70-case calibration pool, we construct five bootstrap sets by sampling 70 cases with replacement. Each bootstrap profile independently determines the stage allocation and refresh schedules at $B = 20$, while evaluation is always performed on the same 630 cases. Second, we conduct leave-one-category-out (LOCO) calibration over the ten PIE-Bench editing categories: for each category, all corresponding examples are removed from calibration, the TVE profiles and schedule are recomputed, and the resulting schedule is evaluated only on that held-out category.

As shown in Table 13, DARTCache is stable under bootstrap resampling. Four of the five runs recover the default $6/14$ allocation, with only one selecting the neighboring $5/15$ split, while the standard deviation of $\text{LPIPS}_{\text{All}}$ remains only 0.0022. Table 14 further shows that the learned temporal profiles transfer well across editing categories. All ten LOCO calibrations retain the same $6/14$ stage allocation, and five recover exactly the default refresh schedule. The mean change in $\text{LPIPS}_{\text{All}}$ is only $+0.0010$, with a median of zero; the largest degradation is $+0.0055$ for Attribute Color. These results support stability to bootstrap resampling of the calibration pool and omission of individual editing categories within PIE-Bench. They do not establish robustness to cross-dataset distribution shift.

D.5 Extension to U-Net Editing Pipelines

Extension to U-Net editing pipelines. To examine whether DARTCache is specific to the FLUX-based Transformer editing pipeline, we further evaluate the proposed scheduling strategy on two U-Net-based inversion editing pipelines, MasaCtrlCao et al. (2023) and EDICTWallace et al. (2023), using Stable Diffusion v1.5. These experiments evaluate whether the temporal variation-aware scheduling principle can generalize to different U-Net-based editing pipelines while preserving their full-inference behavior. All accelerated outputs are evaluated against the corresponding Full inference outputs of each host pipeline. For both U-Net pipelines, we apply the same TVE-based planning procedure used for FLUX-based models, including refresh scheduling and stage budget allocation. The selected stage budgets are obtained by minimizing Eq. 9 over all feasible stage splits. The calibration procedure and schedule construction are identical to those used for the FLUX-based pipelines. DARTCache only controls U-Net prediction refresh and reuse, while attention control and other editor-specific operations follow the original pipeline implementations.

Table 15: **Extension to U-Net editing pipelines.** Fidelity is measured against the corresponding Full inference outputs.

Pipeline	Method	Budget	PSNR↑	LPIPS↓
MasaCtrl	Uniform+Equal	30+30	11.08	0.4866
	TVE+Equal	30+30	11.99	0.4639
	Uniform+Alloc.	40+20	19.87	0.1346
	TVE+Alloc.	**40+20**	**28.84**	**0.0365**
EDICT	Uniform+Equal	16+16	23.09	0.2326
	TVE+Equal	16+16	24.43	0.1694
	Uniform+Alloc.	15+17	23.29	0.2273
	TVE+Alloc.	**15+17**	**25.12**	**0.1517**

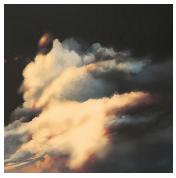
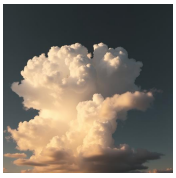
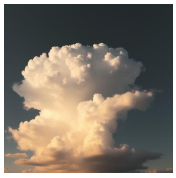
Table 15 summarizes the results under different refresh scheduling strategies. For both pipelines, TVE-based refresh placement consistently improves fidelity compared with uniform scheduling under comparable computation budgets. For MasaCtrl, replacing uniform scheduling with TVE scheduling under the allocated setting improves PSNR from 19.8741 dB to 28.8439 dB and reduces

LPIPS from 0.1346 to 0.0365. For EDICT, TVE scheduling similarly improves PSNR from 23.2943 dB to 25.1173 dB and reduces LPIPS from 0.2273 to 0.1517.

We further observe that stage-aware allocation provides additional fidelity improvements compared with equal allocation. For MasaCtrl, the allocated configuration achieves substantially higher fidelity than the equal-budget setting. For EDICT, allocation improves PSNR from 24.4333 dB to 25.1173 dB and reduces LPIPS from 0.1694 to 0.1517. These results indicate that the temporal variation-based planning principle is not limited to the original FLUX Transformer setting and can also guide reuse scheduling for the evaluated U-Net-based inversion editing pipelines.

E Additional Qualitative Results

The following examples are drawn from the held-out PIE-Bench evaluation split. We first compare DARTCache with the full RF-Inversion host and representative acceleration baselines to assess fidelity preservation under caching. We further provide cross-method comparisons with representative standalone editing methods, including EEdit, Stable Flow, FlowEdit, and UniEdit-Flow. DARTCache uses RF-Inversion as its host pipeline in these visualizations.

Source	Full	**DARTCache** 2.8×	Source	Full	**DARTCache** 2.8×
					
a ~~round~~ square cake with orange frosting on a wooden plate			a ~~cat~~ silver cat sculpture sitting next to a mirror		
					
a cup of coffee with drawing of ~~tulip~~ lion putted on the wooden table			a ~~colorful~~ red bird standing on a branch		
					
white ~~tiger~~ cat on brown ground			an orange van with ~~surfboards~~ flowers on top		
					
a ~~cat~~ fox is shown in a low polygonal style			a cat sitting with beads collar putting hands down		
					
a ~~large red~~ small blue cargo ship in the ocean at sunset			~~painting~~ photo of clouds in the sky		
					
~~rusty~~ shiny car by jill kaufman			a bird standing on ~~clods~~ eggs		
					
a ~~yellow~~ red apple sitting on top of a wooden table			a mid ~~autumn~~ spring road lined with trees and leaves		
a ~~rabbit~~ cat is sitting in a pile of colorful eggs			a fairy with green wings holding a ~~glowing jar~~ crystal ball		

Full	**DARTCache 2.8×**	EEdit 2.45×	SenCache 2.28×	SeaCache 2.45×



A slanted ~~mountain bicycle~~ rusty mountain bicycle on the road in front of a building.



~~Dogs~~ Rabbits running in the grass.



~~Watercolor~~ Oil painting of leaves on a branch.



A ~~gold~~ stone Buddha statue in the temple.



A cartoon man sitting at a desk with a ~~laptop~~ notebook.



~~A beautiful garden~~ A beautiful garden with balloons.



~~*Two cups of coffee with a leaf design on the top*~~ *Two cups of coffee, coffee beans.*



A ~~*woman*~~ *golden woman sculpture in a kimono standing in a river.*



A chair sits in front of a wall with a ~~*floral*~~ *spot pattern.*



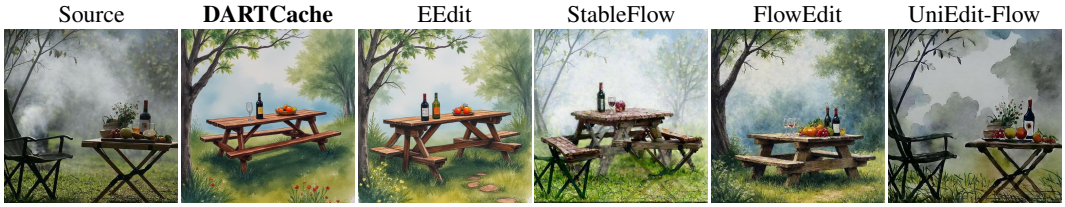
A blue and green illustration of a house in a ~~*hill*~~ *sea.*



A woman in a scarf and coat walking down a ~~*cobblestone street*~~ *thick forest.*



Italian food ingredients on a ~~*wooden table*~~ *picnic blanket.*



Watercolor of a picnic table with a bottle of wine and fruit on it.



A pink zebra and a cat painted on a wall.



A teapot surrounded by ghosts with smoke coming out of it.



A dog wearing a space suit with flowers in its mouth.



Bamboo bonsai plant and ~~notebook~~ calendar on a white table.



Two origami birds sitting on a branch.



A cartoon of a ~~boy~~ girl walking their dog in autumn.



A ~~tiger~~ dog swimming in a pond of green algae.

Source	**DARTCache**	EEdit	StableFlow	FlowEdit	UniEdit-Flow



Blue light, a black-and-white ~~cat~~ dog playing with a flower.



A toy figure of Naruto standing on a ~~table~~ cloud.



A black bird sits on a branch ~~in the snow~~ in the rain.



A ~~cat~~ tiger wearing a pink hat.



Easter eggs, a candle, and a basket against a ~~dark~~ bright background.



A little girl wearing sunglasses and a gray ~~shirt~~ dress leaning against a wall.



A cute little ~~bunny~~ pig with big eyes.



A cute cartoon cat with a chef hat ~~and a pie~~.