# REVE: REGION-AWARE VELOCITY EXTRAPOLATION FOR ACCELERATED IMAGE EDITING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Instruction-based image editing models produce high-quality edits but often incur substantial inference latency due to repeated evaluations of diffusion transformers. Existing training-free acceleration methods reduce this inference cost by reusing cached velocities or rescaling their magnitudes at skipped timesteps, but these approximations often compromise fine detail. We observe that velocity extrapolation improves fine detail in edit regions but introduces artifacts in non-edit regions, while velocity reuse helps preserve non-edit content. Building on this insight, we propose REgion-aware Velocity Extrapolation (REVE), a training-free framework that approximates velocities at skipped timesteps in a region-aware and temporally adaptive manner. REVE distinguishes edit from non-edit regions by measuring agreement between predicted velocities and reference velocities that reconstruct the reference image. At skipped timesteps, REVE reuses velocities in non-edit regions and extrapolates velocities in edit regions, with a time-dependent weight that progressively strengthens extrapolation as denoising advances from structure formation to detail refinement. Experiments with Step1X-Edit, Boogu-Image-0.1-Edit, and LongCat-Image-Edit on GEdit-Bench and GIE-Bench show that REVE achieves $2.54$–$2.73\times$ speedups, matching or exceeding those of the compared training-free acceleration methods. Among these methods, REVE achieves the best semantic consistency, perceptual quality, and detail fidelity, together with the highest pixel-level and perceptual similarity to the unaccelerated results.

## 1 INTRODUCTION

Instruction-based Image Editing (IIE) enables users to modify an image through a natural-language instruction while preserving instruction-irrelevant content (Brooks et al., 2023). Recent IIE models built on flow matching (Lipman et al., 2023; Liu et al., 2023) have substantially improved editing quality and flexibility (Liu et al., 2025d; Chen et al., 2026a; Zhao et al., 2026). However, these models typically require repeated evaluations of large-scale Diffusion Transformers (DiTs) (Peebles & Xie, 2023) throughout an iterative denoising process, resulting in substantial inference latency.

To reduce inference cost, existing training-free acceleration methods address three questions: *when* to skip model evaluations, *where* to allocate computation, and *how* to approximate outputs at skipped timesteps. Temporal methods skip model evaluations at fixed intervals (Selvaraju et al., 2024; Zou et al., 2025) or according to adaptive criteria (Kahatapitiya et al., 2025; Liu et al., 2025a; Ma et al., 2025). Spatial methods concentrate computation on regions that require updates (Liu et al., 2026; Qin et al., 2026). Skipped outputs are approximated by reusing (Liu et al., 2025a; Ma et al., 2025; Zhou et al., 2025; Haghighi & Alahi, 2026) or extrapolating from recent evaluations (Liu et al., 2025c; Zheng et al., 2026; Liu et al., 2025b). RegionE (Chen et al., 2026b) combines spatial and temporal acceleration by restricting computation to edit regions and rescaling cached velocities at skipped timesteps. Nevertheless, these methods either preserve cached directions or apply the same extrapolation across tokens, without accounting for the distinct requirements of detail refinement in edit regions and content preservation in non-edit regions.

In this work, we focus on accelerating IIE models built on flow matching, where each model evaluation predicts a velocity field and each skipped timestep requires approximated velocities to advance the denoising process. Along the denoising trajectory, the predicted velocity changes in both magnitude and direction (Fig. 3 (a)–(b)), which we collectively term *velocity variation*. To examine its



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

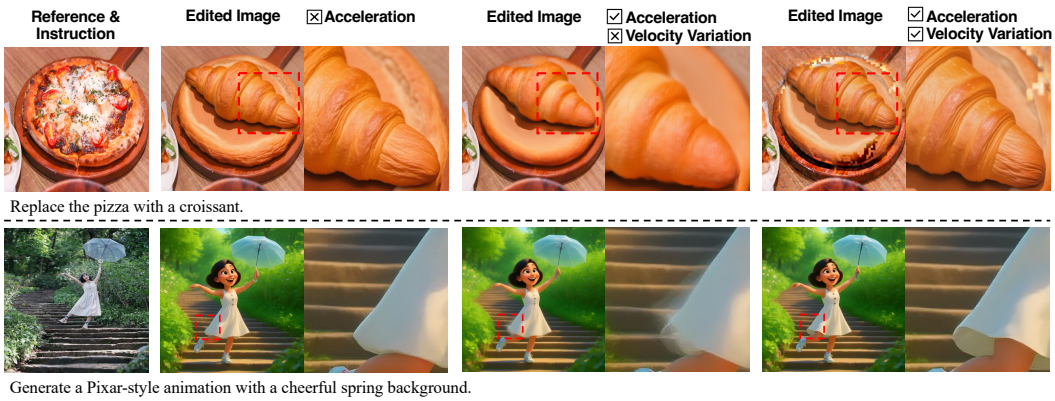


Figure 1: **Effect of velocity variation on accelerated image editing.** Top: An example of a local edit. Ignoring velocity variation preserves the non-edit region but smooths the croissant texture, whereas modeling velocity variation restores fine texture but introduces artifacts in the non-edit region. Bottom: An example of a global edit. Disregarding velocity variation blurs the dress hem, while accounting for velocity variation recovers it.

impact on accelerated image editing, we compare two velocity approximations at skipped timesteps using a local edit and a global edit in Fig. 1: *velocity reuse*, which reuses the most recently evaluated velocity that ignores velocity variation, and *velocity extrapolation*, which extrapolates from recent model evaluations to account for this variation. For the local edit, velocity reuse preserves the non-edit region but produces blurred texture in the edited croissant, whereas velocity extrapolation yields richer croissant texture but introduces artifacts in the non-edit region. For the global edit, where the edit region spans the entire image, velocity reuse likewise blurs fine structures such as the dress hem, whereas velocity extrapolation recovers its sharp contour. Our frequency-domain analysis provides further insight into the detail improvements in the edit region: velocity variation is more pronounced in higher-frequency bands, and the extrapolation increment exhibits a similar spectral profile, suggesting that extrapolation supplies fine-scale updates neglected by velocity reuse (Fig. 3 (c)–(d)).

Motivated by this observation, we propose REgion-aware Velocity Extrapolation, REVE, a training-free framework that improves how velocities are approximated at skipped timesteps by accounting for velocity variation in a region-aware and temporally adaptive manner. REVE first partitions an image into edit and non-edit regions by measuring agreement between the predicted velocity and a reference velocity that reconstructs the reference image. At skipped timesteps, REVE reuses the most recent velocity in the non-edit region to preserve instruction-irrelevant content, while extrapolating from recent model evaluations in the edit region to capture velocity variation. Since denoising typically establishes coarse structures before refining fine details (Qian et al., 2024; Qin et al., 2026), we further introduce a time-dependent weight that attenuates extrapolation during early structure formation and progressively strengthens it during later detail refinement. Moreover, since REVE addresses how to approximate model outputs rather than when to skip model evaluations, it can be readily integrated with adaptive skipping strategies such as TeaCache (Liu et al., 2025a).

We evaluate our framework on GEdit-Bench (Liu et al., 2025d) and GIE-Bench (Qian et al., 2025) using three IIE models: Step1X-Edit (Liu et al., 2025d), Boogu-Image-0.1-Edit (Chen et al., 2026a), and LongCat-Image-Edit (Meituan LongCat Team et al., 2025). Across all three models and both benchmarks, REVE achieves $2.54$–$2.73\times$ speedups while delivering the best semantic consistency, perceptual quality, and detail fidelity among the compared training-free acceleration methods. It also achieves the highest PSNR and SSIM alongside the lowest LPIPS when measured against the unaccelerated outputs.

Our contributions are summarized as follows:

- We identify region-dependent effect of velocity approximations at skipped timesteps: extrapolation improves detail refinement in edit regions but introduces artifacts in non-edit regions, while reuse helps preserve non-edit content. Our frequency-domain analysis further suggests that this improvement stems from higher-frequency updates neglected by reuse.
- We propose REVE, a training-free framework that exploits velocity agreement to guide region-specific approximation at skipped timesteps, combining velocity reuse in non-edit

regions with velocity extrapolation in edit regions. A time-dependent weight progressively strengthens extrapolation, following the progression from structure formation to detail refinement.

- Experiments on three IIE models and two benchmarks demonstrate that REVE achieves the best editing quality among the compared training-free acceleration methods with equal or higher speedups, and remains effective under adaptive skipping.

## 2 Related work

### 2.1 Instruction-based image editing

Instruction-based Image Editing (IIE) aims to modify an image according to a natural-language instruction while preserving instruction-irrelevant content. InstructPix2Pix (Brooks et al., 2023) is a pioneering IIE framework, which trains a diffusion model on synthetically generated instruction-image pairs. Subsequent works improve the quality and coverage of editing data, ranging from the manually annotated MagicBrush (Zhang et al., 2023) to the large-scale and diverse UltraEdit (Zhao et al., 2024) and AnyEdit (Yu et al., 2025). To better interpret user intent, MGIE (Fu et al., 2024) and Step1X-Edit (Liu et al., 2025d) employ multimodal large language models (MLLMs) to guide the editing process. More recently, OmniGen2 (Wu et al., 2026) and Qwen-Image-2.0 (Zhao et al., 2026) unify generation and editing within a single framework. Despite these advances, such editors are predominantly built on diffusion or flow-matching models (Ho et al., 2020; Lipman et al., 2023), whose iterative denoising incurs substantial inference cost.

### 2.2 Efficient image editing

Most image editing models generate outputs through an iterative sampling process and thus inherit general acceleration strategies, which typically reduce the number of sampling steps or lower the compute cost of each step. Efficient solvers (Song et al., 2021; Lu et al., 2022) and distillation methods (Salimans & Ho, 2022; Song et al., 2023) provide general approaches to reducing the number of function evaluations. Caching methods instead exploit the temporal redundancy of intermediate computations across sampling steps, either by reusing features from previous steps (Ma et al., 2024; Wimbauer et al., 2024) or by adaptively determining when to reuse them (Liu et al., 2025a; Bu et al., 2026). To account for feature evolution between steps, TaylorSeer (Liu et al., 2025c) locally extrapolates features via Taylor series expansion, while Spectrum (Han et al., 2026) globally models their trajectories via polynomial fitting.

Another line of work focuses on mitigating spatial redundancy. For instance, RAS (Liu et al., 2026) accelerates text-to-image generation by selectively updating regions attended to by the model. In image editing, since edits are typically localized to restricted regions, spatial redundancy can be identified more reliably. RegionE (Chen et al., 2026b) is a pioneering work for efficient image editing, which applies one-step prediction to non-edit regions and accelerates edit regions through key-value and velocity caching. SpotEdit (Qin et al., 2026) instead identifies non-edit regions via perceptual similarity to the original image. In contrast, our method partitions regions through the agreement between the predicted velocities and reference velocities. Importantly, we approximate velocities in the edit region by temporally adaptive extrapolation to supply fine details.

## 3 Preliminaries

**Flow matching.** Flow matching (Lipman et al., 2023) constructs a probability density path from a noise distribution $\pi_1$ to a data distribution $\pi_0$, with sample trajectories governed by an Ordinary Differential Equation (ODE) $\frac{d\mathbf{x}(t)}{dt} = \mathbf{v}(\mathbf{x}(t), t)$, where $\mathbf{v}$ represents the time-dependent velocity. Rectified flow (Liu et al., 2023) adopts the straight-line interpolation $\mathbf{x}(t) = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$ between $\mathbf{x}_0 \sim \pi_0$ and $\mathbf{x}_1 \sim \pi_1$, yielding a constant target velocity $\mathbf{v}^*(\mathbf{x}(t), t) = \mathbf{x}_1 - \mathbf{x}_0$. A neural network $\mathbf{v}_\theta(\mathbf{x}(t), t)$ is trained to predict the target velocity by minimizing $\mathbb{E}_{t,\mathbf{x}_0,\mathbf{x}_1} \|\mathbf{v}_\theta(\mathbf{x}(t), t) - \mathbf{v}^*(\mathbf{x}(t), t)\|_2^2$. During sampling, the learned ODE is integrated from $t_T = 1$ to $t_0 = 0$ over discrete timesteps $0 = t_0 < t_1 < \cdots < t_T = 1$ using the first-order Euler method:

$$\mathbf{x}(t_{i-1}) = \mathbf{x}(t_i) - (t_i - t_{i-1})\, \mathbf{v}_\theta(\mathbf{x}(t_i), t_i). \tag{1}$$

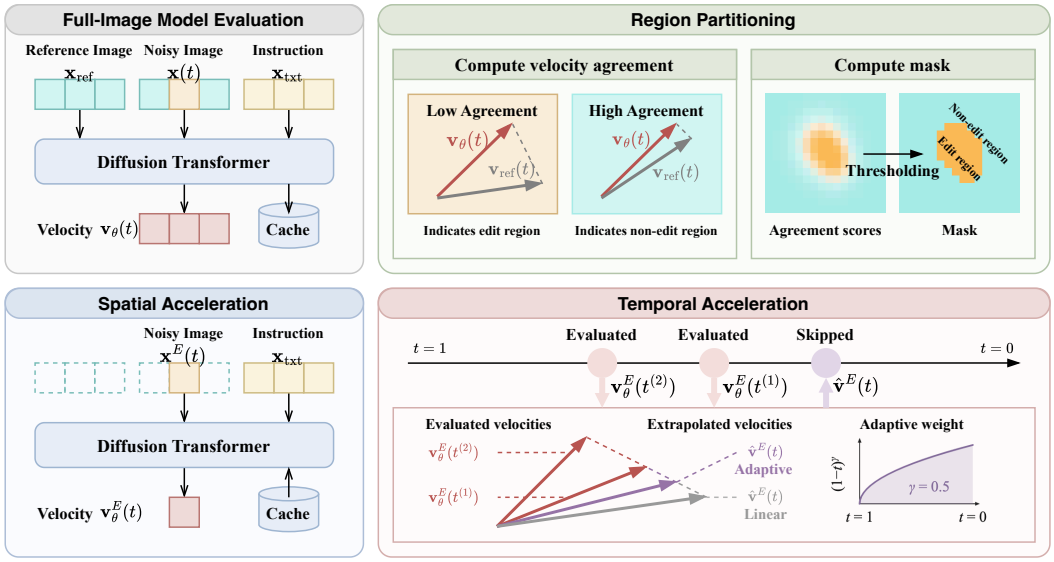


Figure 2: **Overview of our framework.** Top left: Illustration of a full-image model evaluation. Top right: Region partitioning thresholds the agreement scores between predicted and reference velocities to obtain the mask. Bottom left: Spatial acceleration restricts subsequent model evaluations to edit tokens, retaining attention context via cached keys and values (Chen et al., 2026b). Bottom right: Temporal acceleration extrapolates edit-region velocities at skipped timesteps from the two most recent evaluations with an adaptive weight.

**Instruction-based image editing.** Recent IIE models (Liu et al., 2025d; Meituan LongCat Team et al., 2025; Chen et al., 2026a) condition the generative process on a reference image and a natural-language instruction in addition to the initial noise. Given a reference image $\mathbf{x}_{\text{ref}}$, an editing instruction $\mathbf{x}_{\text{txt}}$, and an initial noise $\mathbf{x}_1$, the objective is to generate an edited image $\mathbf{x}_0$ that follows the instruction while preserving instruction-irrelevant content from the reference image. For simplicity, we use the same notation for variables in pixel and latent spaces, and denote discrete timesteps by $t$ instead of $t_i$. At each timestep $t$, the noisy image tokens $\mathbf{x}(t)$ are combined with the reference image tokens $\mathbf{x}_{\text{ref}}$ and instruction tokens $\mathbf{x}_{\text{txt}}$ to form a unified token sequence, which is then processed by a DiT to predict the conditional velocity field $\mathbf{v}_\theta(t) \coloneqq \mathbf{v}_\theta\left(\mathbf{x}(t), t \mid \mathbf{x}_{\text{ref}}, \mathbf{x}_{\text{txt}}\right)$. We refer to such an evaluation over the full token sequence as a *full-image model evaluation*.

## 4 Method

REVE is a training-free framework that accelerates pretrained IIE models by identifying edit regions through velocity agreement and treating edit and non-edit regions differently across both space and time. As illustrated in Fig. 2, it comprises three main components. First, we partition the image into edit and non-edit regions by measuring agreement between the predicted and reference velocities (Sec. 4.1). Second, we restrict subsequent model evaluations to the edit region (Sec. 4.2). Third, at skipped timesteps, we reuse cached velocities in the non-edit region to preserve instruction-irrelevant content and apply temporally adaptive velocity extrapolation in the edit region to support detail refinement (Sec. 4.3). We provide the complete algorithm in Appendix A.

### 4.1 Region Partitioning

Recent IIE models typically process the entire image without distinguishing regions that require modification from those that should remain unchanged, which incurs substantial computational redundancy. To reduce this redundancy and enable region-specific velocity approximation, we partition the image by measuring agreement between two velocity fields. Given the noisy image tokens $\mathbf{x}(t) \in \mathbb{R}^{L \times D}$ at timestep $t$ and the reference image tokens $\mathbf{x}_{\text{ref}} \in \mathbb{R}^{L \times D}$, where $L$ is the number of image tokens and $D$ is the channel dimension, we define the *reference velocity* $\mathbf{v}_{\text{ref}}(t) \in \mathbb{R}^{L \times D}$ that transports $\mathbf{x}(t)$ to $\mathbf{x}_{\text{ref}}$ along a straight trajectory:

$$\mathbf{v}_{\text{ref}}(t) = \frac{\mathbf{x}(t) - \mathbf{x}_{\text{ref}}}{t}. \tag{2}$$

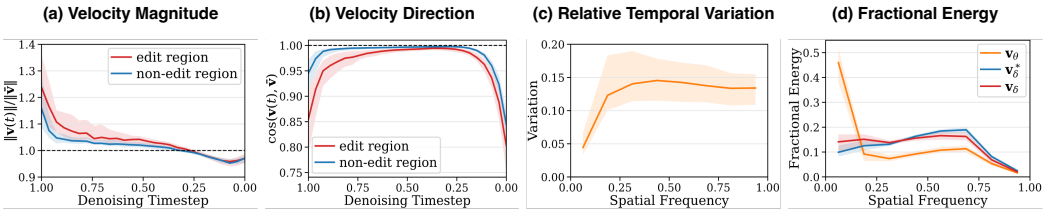


Figure 3: **Velocity variation and frequency-domain analysis.** (a)–(b): Velocity variation in magnitude and direction along the denoising trajectory in edit and non-edit regions. (c)–(d): Frequency-domain analysis of relative temporal variation and fractional energy across spatial frequency bands.

We then compare the predicted and reference velocities to derive the *agreement scores*:

$$A(t) = 1 - \frac{\|\mathbf{v}_\theta(t) - \mathbf{v}_{\text{ref}}(t)\|_2}{\|\mathbf{v}_\theta(t)\|_2 + \|\mathbf{v}_{\text{ref}}(t)\|_2 + \epsilon}, \tag{3}$$

where the Euclidean norm is computed over channels and $\epsilon > 0$ is a small constant for numerical stability. The score is bounded in $(0, 1]$ by the triangle inequality. A high score indicates that the predicted velocity closely matches the reference velocity, which signals a *non-edit token*, whereas a low score indicates a deviation from the reference image, which signals an *edit token*. The mask for region partitioning is therefore obtained by thresholding the agreement scores with $\tau \in (0, 1)$, i.e., $M(t) = \mathbf{1}[A(t) < \tau] \in \{0, 1\}^L$, where a value of 1 denotes an edit token and a value of 0 denotes a non-edit token.

### 4.2 Spatial Acceleration

Given the mask $M(t)$, $\mathbf{x}(t)$ can be divided into edit tokens $\mathbf{x}^E(t)$ and non-edit tokens $\mathbf{x}^N(t)$. Following RegionE (Chen et al., 2026b), we drop $\mathbf{x}_{\text{ref}}$ and $\mathbf{x}^N(t)$ from the unified sequence, keeping only $\mathbf{x}^E(t)$ and $\mathbf{x}_{\text{txt}}$, while reusing the keys and values of the dropped tokens cached from the most recent full-image evaluation to retain global context during attention computation. The DiT therefore predicts only the edit-region velocities $\mathbf{v}_\theta^E(t)$ to update the edit tokens, which effectively reduces the number of tokens processed per evaluation. The non-edit tokens are updated with velocities reused from the most recent full-image evaluation.

### 4.3 Temporal Acceleration

Temporal acceleration is achieved by skipping model evaluations, which requires approximating velocities of edit tokens at skipped timesteps. We first analyze the evolution of predicted velocities along the denoising trajectory, and use the findings to guide our velocity approximation.

**Velocity variation.** We characterize the predicted velocity in terms of its magnitude and direction. Let $\bar{\mathbf{v}} = \frac{1}{T}\sum_t \mathbf{v}_\theta(t)$ denote the velocity averaged over timesteps. At each timestep $t$, we measure the normalized magnitude $\|\mathbf{v}_\theta(t)\|_2 / \|\bar{\mathbf{v}}\|_2$ and the directional alignment $\cos(\mathbf{v}_\theta(t), \bar{\mathbf{v}})$, separately within the edit and non-edit regions. As shown in Fig. 3 (a)–(b), the magnitude drops sharply at the beginning and then decreases steadily, while the direction deviates substantially from its average at both ends of the trajectory and remains relatively stable in the middle, with larger deviations in the edit region than in the non-edit region. These observations indicate that, despite the straight-line training objective of rectified flow (Liu et al., 2023), the predicted velocity varies over time in both magnitude and direction (Chen et al., 2026b), particularly in the edit region.

**Velocity reuse and velocity extrapolation.** Hereafter, velocities are restricted to the edit region, and we omit the superscript $E$ for brevity. A straightforward approach to approximating the velocity at a skipped timestep is to reuse the most recently evaluated velocity (Liu et al., 2025a; Ma et al., 2025; Haghighi & Alahi, 2026), which we term *velocity reuse*. However, as the velocity varies temporally along the denoising trajectory, velocity reuse may degrade visual quality, producing attenuated details or blurred textures in edited images. To address this issue, we introduce *velocity extrapolation* to approximate the velocity at a skipped timestep from its evaluation history. Given a skipped timestep $t$, let $t^{(1)}$ and $t^{(2)}$ denote the timesteps of the two most recent evaluations, with

$t^{(2)} > t^{(1)} > t$. We approximate the velocity at $t$ as:

$$\hat{\mathbf{v}}(t) = \mathbf{v}_\theta(t^{(1)}) + \alpha(t)\mathbf{v}_\delta(t), \tag{4}$$

$$\mathbf{v}_\delta(t) = \frac{t - t^{(1)}}{t^{(1)} - t^{(2)}} \left( \mathbf{v}_\theta(t^{(1)}) - \mathbf{v}_\theta(t^{(2)}) \right), \tag{5}$$

where $\mathbf{v}_\delta(t)$ is the *extrapolation increment* and $\alpha(t)$ controls its contribution. Setting $\alpha(t) = 0$ recovers velocity reuse, while $\alpha(t) = 1$ yields linear extrapolation. We restrict velocity extrapolation to the edit region, as applying it to the non-edit region can introduce artifacts in content that should remain unchanged (Fig. 1; Tab. 2; Fig. 5).

**Frequency-domain analysis.** To understand why velocity reuse compromises fine details and what extrapolation contributes beyond reuse, we conduct a frequency-domain analysis. Let $\mathbf{v}^*_\delta(t) = \mathbf{v}_\theta(t) - \mathbf{v}_\theta(t^{(1)})$ denote the *temporal variation*, i.e., the velocity variation missed by reuse. We define *relative temporal variation* in spatial frequency band $f$ as $R_f(t) = \|[\mathbf{v}^*_\delta(t)]_f\|_2 / \|[\mathbf{v}_\theta(t)]_f\|_2$, where $[\cdot]_f$ denotes a band-pass filter, and *fractional energy* $E_f(\cdot)$ as the fraction of a quantity's spectral energy in band $f$ (see Appendix B for detailed definitions). As depicted in Fig. 3 (c), $R_f$ is smallest in the lowest-frequency band and larger in higher-frequency bands, indicating that fine-scale velocity components exhibit greater relative variation along the denoising trajectory. Fig. 3 (d) further compares the fractional energies of the predicted velocity $\mathbf{v}_\theta$, the extrapolation increment $\mathbf{v}_\delta$, and the temporal variation $\mathbf{v}^*_\delta$. The predicted velocity places its largest energy fraction in the lowest-frequency band, whereas the temporal variation allocates a greater fraction of its energy to higher-frequency bands. It is worth noting that the spectral profile of the extrapolation increment resembles that of the temporal variation, suggesting that *velocity extrapolation supplies fine-scale updates absent from velocity reuse*.

**Temporally adaptive velocity extrapolation.** The denoising process typically follows a coarse-to-fine progression, with low-frequency structures established in early stages and high-frequency details refined in later stages (Qian et al., 2024; Qin et al., 2026). To align extrapolation with this progression, we define $\alpha(t)$ in Eq. 4 as a time-dependent weight:

$$\alpha(t) = (1 - t)^\gamma, \tag{6}$$

where $\gamma > 0$ controls the progression of extrapolation strength. As denoising proceeds from $t = 1$ to $t = 0$, $\alpha(t)$ increases monotonically from 0 to 1, gradually shifting the approximation from velocity reuse toward linear extrapolation.

# 5 Experiments

## 5.1 Experimental settings

**IIE models and acceleration methods.** We evaluate REVE on three open-source IIE models built on flow-matching DiTs: Step1X-Edit (Liu et al., 2025d), Boogu-Image-0.1-Edit (Chen et al., 2026a), and LongCat-Image-Edit (Meituan LongCat Team et al., 2025). We compare against five representative training-free acceleration methods: TaylorSeer (Liu et al., 2025c), TeaCache (Liu et al., 2025a), RAS (Liu et al., 2026), SpotEdit (Qin et al., 2026), and RegionE (Chen et al., 2026b). Of these, SpotEdit and RegionE specifically target IIE, with RegionE jointly exploiting spatial and temporal redundancies.

**Datasets and evaluation metrics.** We evaluate on GEdit-Bench (Liu et al., 2025d), which contains 606 real-world image-instruction pairs across 11 editing tasks, and GIE-Bench (Qian et al., 2025), which contains over 1000 pairs spanning natural and synthetic images across 9 tasks. Following Step1X-Edit (Liu et al., 2025d), we assess Semantic Consistency (SC) for instruction following and Perceptual Quality (PQ) for visual naturalness and absence of artifacts. Both metrics are scored from 0 to 10 using Qwen3-VL-8B-Instruct (Bai et al., 2025). We additionally evaluate Detail Fidelity (DF) to assess detail preservation. DF is computed as the absolute log-ratio of the mean luminance gradient magnitudes of the accelerated and unaccelerated results within the edit region, with lower scores indicating better fidelity (see Appendix C). We also report PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) against the unaccelerated results to measure pixel-level, structural, and perceptual fidelity.

Table 1: **Comparison of editing quality and inference efficiency on GEdit-Bench.** "Type" denotes the acceleration strategy: F feature forecasting, T temporal redundancy reduction, and S spatial redundancy reduction. "Lat." denotes latency in seconds, and "Spd." denotes speedup over the unaccelerated model. Best and second-best results among acceleration methods are shown in **bold** and underlined, respectively.

| Method | Type | Editing Quality | | | | | | Editing Efficiency | |
|---|---|---|---|---|---|---|---|---|---|
| | | SC ↑ | PQ ↑ | DF ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Lat. ↓ | Spd. ↑ |
| Step1X-Edit | | 8.287 | 8.107 | 0.000 | – | – | – | 79.11 | 1.00× |
| + TaylorSeer | F | 8.162 | 7.916 | 0.141 | 25.44 | 0.879 | 0.130 | **29.47** | **2.68**× |
| + TeaCache | T | 8.215 | 7.983 | 0.119 | 26.65 | 0.879 | 0.131 | 31.14 | 2.54× |
| + RAS | S | 8.190 | 7.743 | 0.101 | 27.18 | 0.872 | 0.135 | 40.51 | 1.95× |
| + SpotEdit | S | 8.064 | 7.914 | 0.112 | 27.01 | 0.929 | 0.078 | 43.46 | 1.82× |
| + RegionE | S & T | 8.187 | 8.031 | 0.068 | 30.84 | 0.940 | 0.060 | 32.26 | 2.45× |
| + REVE (Ours) | S & T | **8.238** | **8.050** | **0.046** | **31.79** | **0.947** | **0.049** | **29.47** | **2.68**× |
| Boogu-Image-0.1-Edit | | 8.467 | 7.894 | 0.000 | – | – | – | 97.40 | 1.00× |
| + TaylorSeer | F | 7.967 | 7.467 | 0.244 | 24.06 | 0.854 | 0.151 | 41.24 | 2.36× |
| + TeaCache | T | 8.399 | 7.827 | 0.092 | 30.40 | 0.938 | 0.058 | 40.86 | 2.38× |
| + RAS | S | 8.264 | 7.574 | 0.100 | 27.88 | 0.901 | 0.093 | 45.85 | 2.12× |
| + SpotEdit | S | 8.310 | 7.728 | 0.087 | 26.96 | 0.889 | 0.089 | 51.92 | 1.88× |
| + RegionE | S & T | 8.386 | 7.837 | 0.050 | 32.82 | 0.953 | 0.041 | 37.87 | 2.57× |
| + REVE (Ours) | S & T | **8.408** | **7.855** | **0.032** | **33.72** | **0.959** | **0.033** | **36.76** | **2.65**× |
| LongCat-Image-Edit | | 8.528 | 8.233 | 0.000 | – | – | – | 36.17 | 1.00× |
| + TaylorSeer | F | 8.269 | 7.988 | 0.104 | 25.02 | 0.873 | 0.113 | 14.78 | 2.45× |
| + TeaCache | T | 8.404 | 8.079 | 0.112 | 24.17 | 0.857 | 0.128 | 14.33 | 2.52× |
| + RAS | S | 8.307 | 7.922 | 0.089 | 25.26 | 0.847 | 0.137 | 19.68 | 1.84× |
| + SpotEdit | S | 8.429 | 7.995 | 0.075 | 26.01 | 0.887 | 0.096 | 23.48 | 1.54× |
| + RegionE | S & T | 8.477 | 8.135 | 0.054 | 30.01 | 0.928 | 0.062 | 15.44 | 2.34× |
| + REVE (Ours) | S & T | **8.517** | **8.186** | **0.031** | **30.77** | **0.936** | **0.049** | **14.25** | **2.54**× |

**Implementation details.** Following RegionE, we adopt 28 denoising steps and retain its full-image evaluation schedule: six initial steps for stabilization, one intermediate step to update the key-value cache, and two final steps for smoothing. Since our method focuses on how to approximate velocities at skipped timesteps rather than when to skip, we employ a fixed schedule for edit-region evaluations, at steps $\{9, 12\}$ before and $\{20, 24\}$ after the cache update. We set $\tau = 0.82$ and $\gamma = 0.5$ across all models and benchmarks. We run Step1X-Edit (12B DiT) on RTX 4090 GPUs (48 GB), and Boogu-Image-0.1-Edit (10B DiT) and LongCat-Image-Edit (6B DiT) on A100 GPUs (40 GB). Latency is measured on a single GPU with a batch size of 1.

### 5.2 Comparison with state-of-the-art methods

**Quantitative comparison.** Tab. 1 presents quantitative comparison results on GEdit-Bench, with GIE-Bench results provided in Tab. 4 of Appendix D. Across all three models and both benchmarks, REVE achieves the best scores on all six editing quality metrics among the compared acceleration methods with equal or higher speedups ($2.54$–$2.73\times$). Compared with unaccelerated models, the SC and PQ scores of REVE deviate by at most $0.059$ and $0.057$, respectively, indicating that instruction following and perceptual quality are largely preserved. Compared with RegionE, which also exploits spatial and temporal redundancies, REVE consistently improves editing quality at lower latency. For instance, on Step1X-Edit with GEdit-Bench, REVE reduces DF from $0.068$ to $0.046$ and LPIPS from $0.060$ to $0.049$, while increasing PSNR from $30.84$ to $31.79$ and speedup from $2.45\times$ to $2.68\times$. In this setting, TaylorSeer achieves the same speedup but yields the highest DF ($0.141$) and lowest PSNR ($25.44$) among the compared methods.

**Qualitative comparison.** Fig. 4 shows qualitative comparisons on Step1X-Edit across object addition, global stylization, and text replacement. TaylorSeer and TeaCache introduce an additional camel head in object addition and noticeably shift the zebra's colors from the unaccelerated baseline in global stylization. For text replacement, RAS blurs the letter strokes, while RegionE produces irregular letter boundaries. SpotEdit produces visually clean results but alters the outlined lettering style. In comparison, REVE preserves structural coherence, color fidelity, and sharp letter boundaries at a $2.68\times$ speedup.




Figure 4: **Qualitative comparison of acceleration methods.** Red boxes indicate regions enlarged for detail comparison. Our method preserves structural coherence, color consistency, and fine details while achieving a 2.68× speedup over the unaccelerated baseline.

Table 2: **Ablation of spatial and temporal acceleration.** "S.A." and "T.A." denote spatial acceleration and temporal acceleration. "T.A." is specified by approximation strategies in "non-edit/edit" regions, with "R", "E", and "A" representing reuse, linear extrapolation, and adaptive extrapolation. "Pert." denotes perturbation of the edit-region extrapolation increment, where "D" represents randomizing token directions and "M" represents shuffling magnitudes across tokens. Best and second-best results among rows 3–6 are in **bold** and underlined.

| No. | S.A. | T.A. | Pert. | Editing Quality | | | | | | Editing Efficiency | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | SC ↑ | PQ ↑ | DF ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Lat. ↓ | Spd. ↑ |
| 1 | ✗ | ✗ | – | 8.287 | 8.107 | 0.000 | – | – | – | 79.11 | 1.00× |
| 2 | ✓ | ✗ | – | 8.243 | 8.059 | 0.037 | 32.07 | 0.953 | 0.043 | 44.95 | 1.76× |
| 3 | ✓ | R / R | – | 8.218 | 8.036 | 0.073 | 31.71 | 0.941 | 0.057 | **29.47** | **2.68**× |
| 4 | ✓ | E / E | – | 7.668 | 6.794 | 0.089 | 26.09 | 0.858 | 0.197 | **29.47** | **2.68**× |
| 5 | ✓ | R / E | – | 8.188 | 8.023 | **0.042** | 30.90 | 0.942 | 0.054 | **29.47** | **2.68**× |
| 6 | ✓ | R / A | – | **8.238** | **8.050** | 0.046 | **31.79** | **0.947** | **0.049** | **29.47** | **2.68**× |
| 7 | ✓ | R / A | D | 8.224 | 8.013 | 0.073 | 31.70 | 0.941 | 0.057 | 29.47 | 2.68× |
| 8 | ✓ | R / A | M | 8.224 | 8.045 | 0.050 | 31.82 | 0.947 | 0.050 | 29.47 | 2.68× |

### 5.3 Ablation Studies

We ablate the main components of our framework, the extrapolation increment, and the integration with adaptive timestep skipping, using Step1X-Edit on GEdit-Bench unless otherwise specified. Additional ablations on the threshold $\tau$, the extrapolation strength $\gamma$, and the extrapolation order are provided in Appendices E, F, and G, respectively.

**Ablation of main components.** Tab. 2 evaluates the spatial and temporal acceleration components, with qualitative comparisons of velocity approximation strategies in Fig. 5. Spatial acceleration alone achieves a 1.76× speedup while largely preserving editing quality (row 2). Adding temporal

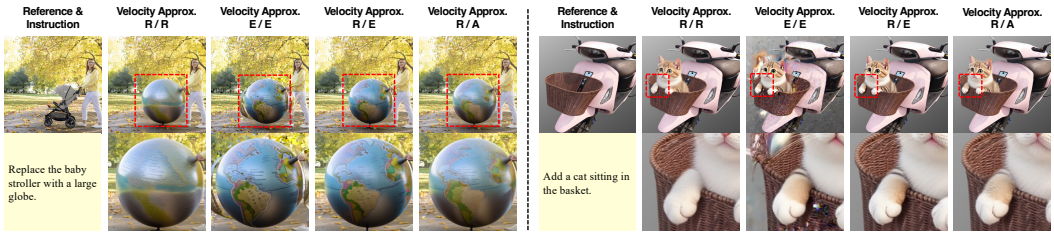


Figure 5: **Qualitative comparison of velocity approximation strategies.** "R/R" blurs details, "E/E" restores details in edit regions but introduces artifacts in non-edit regions, "R/E" and "R/A" retain details while keeping non-edit regions free of artifacts.

Table 3: **Integration with adaptive timestep skipping.** Rows marked with † use TeaCache's adaptive skipping strategy in place of REVE's fixed schedule.

| Benchmark | Method | Editing Quality | | | | | | Editing Efficiency | |
|---|---|---|---|---|---|---|---|---|---|
| | | SC ↑ | PQ ↑ | DF ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Lat. ↓ | Spd. ↑ |
| GEdit-Bench | TeaCache (Liu et al., 2025a) | 8.215 | 7.983 | 0.119 | 26.65 | 0.879 | 0.131 | **31.14** | **2.54**× |
| | TeaCache w/ REVE† | **8.221** | **8.051** | **0.039** | **31.98** | **0.950** | **0.047** | 31.50 | 2.51× |
| GIE-Bench | TeaCache (Liu et al., 2025a) | 8.113 | 8.510 | 0.105 | 29.16 | 0.921 | 0.076 | 31.03 | 2.55× |
| | TeaCache w/ REVE† | **8.173** | **8.567** | **0.043** | **32.69** | **0.952** | **0.034** | **29.99** | **2.63**× |

acceleration increases the speedup to 2.68× across all approximation strategies (rows 3–6). However, reusing velocities in both regions worsens DF from 0.037 to 0.073 and LPIPS from 0.043 to 0.057 (row 3). The corresponding "R/R" results show blurred map detail on the globe and faded fur color on the cat's foreleg. Extrapolating velocities in both regions restores these details, but the "E/E" results exhibit pronounced artifacts in the surrounding scene, particularly around the globe and on the basket, with PQ dropping to 6.794 and LPIPS rising to 0.197 (row 4). Restricting extrapolation to the edit region retains the recovered details while suppressing artifacts in the non-edit region and achieves the lowest DF of 0.042 among these strategies (row 5). Adding temporal adaptivity ("R/A") further improves SC from 8.188 to 8.238 and PQ from 8.023 to 8.050, and achieves the best PSNR, SSIM, and LPIPS among these strategies, with a competitive DF of 0.046 at the same speedup (row 6).

**Ablation of the extrapolation increment.** We examine the contributions of increment direction and magnitude through two perturbations in the edit region: randomizing each token's increment direction while preserving its magnitude, and shuffling magnitudes across tokens while preserving their directions. As shown in Tab. 2, direction randomization worsens DF from 0.046 to 0.073 and LPIPS from 0.049 to 0.057 (rows 6 and 7), degrading both metrics to the levels of velocity reuse (row 3). In contrast, magnitude shuffling retains most of the improvement over reuse, with DF and LPIPS both at 0.050 (row 8). These results suggest that extrapolation-derived directions drive much of the detail improvement, while token-specific magnitudes further enhance it.

**Effectiveness beyond fixed skipping.** To assess whether the benefits of our framework extend beyond a fixed schedule, we replace its skipping schedule with TeaCache's adaptive skipping strategy. As shown in Tab. 3, the integrated variant improves all six quality metrics over TeaCache on both benchmarks at comparable inference cost. For instance, DF decreases from 0.119 to 0.039 on GEdit-Bench and from 0.105 to 0.043 on GIE-Bench. These results suggest that our velocity approximation method remains effective under adaptive skipping strategies.

## 6 Conclusion

We presented REVE, a training-free framework for accelerating IIE models. Our analysis shows that velocity extrapolation recovers fine details in edit regions but introduces artifacts in non-edit regions, while velocity reuse preserves non-edit content. Guided by this finding, REVE identifies edit regions via velocity agreement, reuses velocities in non-edit regions, and applies progressively stronger extrapolation in edit regions as denoising advances. Across three IIE models and two benchmarks, REVE achieves comparable or faster speedups while attaining the best editing quality among the compared methods. Limitations and future work are discussed in Appendix I.

## REFERENCES


Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Tim Brooks, Aleksander Holynski, and Alexei A Efros. InstructPix2Pix: Learning to follow image editing instructions. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18392–18402. IEEE, 2023.

Jiazi Bu, Pengyang Ling, Yujie Zhou, Yibin Wang, Yuhang Zang, Dahua Lin, and Jiaqi Wang. DiCache: Let diffusion model determine its own cache. In *International Conference on Learning Representations*, 2026.

Guoxuan Chen, Chufeng Xiao, Haoran Yang, Siyue Xie, Binxiao Huang, Ming Zhang, Cheuk Him Chau, Xinyu Fu, Yingzhao Lian, Tom S. Y. Li, et al. Boogu-Image-0.1: Boosting open agentic multimodal generation via understanding under a minimal budget. *arXiv preprint arXiv:2607.13125*, 2026a.

Pengtao Chen, Xianfang Zeng, Maosen Zhao, Mingzhu Shen, Peng Ye, Bangyin Xiang, Zhibo Wang, Wei Cheng, Gang Yu, and Tao Chen. RegionE: Adaptive region-aware generation for efficient image editing. In *International Conference on Learning Representations*, 2026b.

Tsu-Jui Fu, Wenze Hu, Xianzhi Du, William Yang Wang, Yinfei Yang, and Zhe Gan. Guiding instruction-based image editing via multimodal large language models. In *International Conference on Learning Representations*, 2024.

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*, 2026.

Jiaqi Han, Juntong Shi, Puheng Li, Haotian Ye, Qiushan Guo, and Stefano Ermon. Adaptive spectral feature forecasting for diffusion sampling acceleration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

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

Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15240–15252. IEEE, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

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, 2025a.

Jiacheng Liu, Peiliang Cai, Qinming Zhou, Yuqi Lin, Deyang Kong, Benhao Huang, Yupei Pan, Haowen Xu, Chang Zou, Junshu Tang, Shikang Zheng, and Linfeng Zhang. FreqCa: Accelerating diffusion models via frequency-aware caching. *arXiv preprint arXiv:2510.08669*, 2025b.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with TaylorSeers. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15853–15863. IEEE, 2025c.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pp. 38–55, 2024.

Shiyu Liu, Yucheng Han, Peng Xing, Fukun Yin, Rui Wang, Wei Cheng, Jiaqi Liao, Yingming Wang, Honghao Fu, Chunrui Han, et al. Step1X-Edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*, 2025d.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Ziming Liu, Yifan Yang, Chengruidong Zhang, Yiqi Zhang, Lili Qiu, Yang You, and Yuqing Yang. Region-adaptive sampling for diffusion transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2346–2356, 2026.

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.

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

Zehong Ma, Longhui Wei, Feng Wang, Shiliang Zhang, and Qi Tian. MagCache: Fast video generation with magnitude-aware cache. In *Advances in Neural Information Processing Systems*, volume 38, pp. 34348–34380, 2025.

Meituan LongCat Team, Hanghang Ma, Haoxian Tan, Jiale Huang, Junqiang Wu, Jun-Yan He, Lishuai Gao, Songlin Xiao, Xiaoming Wei, Xiaoqi Ma, Xunliang Cai, Yayong Guan, and Jie Hu. LongCat-Image technical report. *arXiv preprint arXiv:2512.07584*, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4172–4182. IEEE, 2023.

Yurui Qian, Qi Cai, Yingwei Pan, Yehao Li, Ting Yao, Qibin Sun, and Tao Mei. Boosting diffusion models with moving average sampling in frequency domain. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8911–8920. IEEE, 2024.

Yusu Qian, Jiasen Lu, Tsu-Jui Fu, Xinze Wang, Chen Chen, Yinfei Yang, Wenze Hu, and Zhe Gan. GIE-Bench: Towards grounded evaluation for text-guided image editing. *arXiv preprint arXiv:2505.11493*, 2025.

Zhibin Qin, Zhenxiong Tan, Zeqing Wang, Songhua Liu, and Xinchao Wang. SpotEdit: Selective region editing in diffusion transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18683–18692, 2026.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. SAM 2: Segment anything in images and videos. In *International Conference on Learning Representations*, 2025.

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

Pratheba Selvaraju, Tianyu Ding, Tianyi Chen, Ilya Zharkov, and Luming Liang. FORA: Fast-forward caching in diffusion transformer acceleration. *arXiv preprint arXiv:2407.01425*, 2024.

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

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

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

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

Chenyuan Wu, Jiahao Wang, Pengfei Zheng, Ruiran Yan, Shitao Xiao, Xin Luo, Yueze Wang, Wanli Li, Xiyan Jiang, Yexin Liu, et al. OmniGen2: Towards instruction-aligned multimodal generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21964–21975, 2026.

Qifan Yu, Wei Chow, Zhongqi Yue, Kaihang Pan, Yang Wu, Xiaoyang Wan, Juncheng Li, Siliang Tang, Hanwang Zhang, and Yueting Zhuang. AnyEdit: Mastering unified high-quality image editing for any idea. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26125–26135. IEEE, 2025.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. MagicBrush: A manually annotated dataset for instruction-guided image editing. In *Advances in Neural Information Processing Systems*, 2023.

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.

Bing Zhao, Chenfei Wu, Deqing Li, Hao Meng, Jiahao Li, Jie Zhang, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kuan Cao, et al. Qwen-Image-2.0 technical report. *arXiv preprint arXiv:2605.10730*, 2026.

Haozhe Zhao, Xiaojian Ma, Liang Chen, Shuzheng Si, Rujie Wu, Kaikai An, Peiyu Yu, Minjia Zhang, Qing Li, and Baobao Chang. UltraEdit: Instruction-based fine-grained image editing at scale. In *Advances in Neural Information Processing Systems*, volume 37, pp. 3058–3093, 2024.

Shikang Zheng, Guantao Chen, Qinming Zhou, Yuqi Lin, Lixuan He, Chang Zou, Peiliang Cai, Jiacheng Liu, and Linfeng Zhang. Let features decide their own solvers: Hybrid feature caching for diffusion transformers. In *International Conference on Learning Representations*, 2026.

Xin Zhou, Dingkang Liang, Kaijin Chen, Tianrui Feng, Xiwu Chen, Hongkai Lin, Yikang Ding, Feiyang Tan, Hengshuang Zhao, and Xiang Bai. Less is enough: Training-free video diffusion acceleration via runtime-adaptive caching. *arXiv preprint arXiv:2507.02860*, 2025.

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.

## APPENDIX CONTENTS

**A Algorithm** **14**

**B Frequency-domain analysis** **14**

**C Detail fidelity** **15**

**D GIE-Bench comparison results** **15**

**E Ablation of the threshold** **16**

**F Ablation of the extrapolation strength** **16**

**G Ablation of the extrapolation order** **17**

**H Mask visualization** **18**

**I Limitations and future work** **19**

**J AI use statement** **19**

**Algorithm 1** Inference with REVE

**Require:** Reference image $\mathbf{x}_{\text{ref}}$, instruction $\mathbf{x}_{\text{txt}}$, initial noise $\mathbf{x}_1$, timesteps $0 = t_0 < t_1 < \cdots < t_T = 1$, schedule $(\mathcal{T}_{\text{full}}, \mathcal{T}_{\text{edit}}, \mathcal{T}_{\text{part}})$, threshold $\tau$, extrapolation strength $\gamma$
**Ensure:** Edited image $\mathbf{x}_0$

$\mathbf{x}(t_T) \leftarrow \mathbf{x}_1$
**for** $i = T, T-1, \ldots, 1$ **do**
  $t \leftarrow t_i$
  **if** $t \in \mathcal{T}_{\text{full}}$ **then** ▷ Full-image evaluation
    Predict $\mathbf{v}_\theta(t)$ from $\mathbf{x}(t)$, $\mathbf{x}_{\text{ref}}$, and $\mathbf{x}_{\text{txt}}$
    Cache the keys and values of $\mathbf{x}_{\text{ref}}$ and $\mathbf{x}(t)$
    **if** $t \in \mathcal{T}_{\text{part}}$ **then** ▷ Region partitioning, Sec. 4.1
      $\mathbf{v}_{\text{ref}}(t) \leftarrow (\mathbf{x}(t) - \mathbf{x}_{\text{ref}})/t$
      Compute $A(t)$ by Eq. 3; $M(t) \leftarrow \mathbf{1}[A(t) < \tau]$; $t_{\text{part}} \leftarrow t$
    $\tilde{\mathbf{v}}(t) \leftarrow \mathbf{v}_\theta(t)$; $t_{\text{full}} \leftarrow t$
  **else if** $t \in \mathcal{T}_{\text{edit}}$ **then** ▷ Edit-region evaluation, Sec. 4.2
    Split $\mathbf{x}(t)$ into $\mathbf{x}^E(t)$ and $\mathbf{x}^N(t)$ by $M(t_{\text{part}})$
    Predict $\mathbf{v}_\theta^E(t)$ from $\mathbf{x}^E(t)$ and $\mathbf{x}_{\text{txt}}$, attending to the cached keys and values of $\mathbf{x}_{\text{ref}}$ and $\mathbf{x}^N(t_{\text{full}})$
    $\tilde{\mathbf{v}}^E(t) \leftarrow \mathbf{v}_\theta^E(t)$; $\tilde{\mathbf{v}}^N(t) \leftarrow \mathbf{v}_\theta^N(t_{\text{full}})$
  **else** ▷ Skipped timestep, Sec. 4.3
    $\alpha(t) \leftarrow (1-t)^\gamma$ ▷ Eq. 6
    $\mathbf{v}_\delta^E(t) \leftarrow \frac{t - t^{(1)}}{t^{(1)} - t^{(2)}} \left(\tilde{\mathbf{v}}^E(t^{(1)}) - \tilde{\mathbf{v}}^E(t^{(2)})\right)$ ▷ Eq. 5
    $\tilde{\mathbf{v}}^E(t) \leftarrow \tilde{\mathbf{v}}^E(t^{(1)}) + \alpha(t)\, \mathbf{v}_\delta^E(t)$ ▷ Extrapolation, Eq. 4
    $\tilde{\mathbf{v}}^N(t) \leftarrow \mathbf{v}_\theta^N(t_{\text{full}})$ ▷ Reuse
  **if** $t \notin \mathcal{S}$ **then**
    $t^{(2)} \leftarrow t^{(1)}$; $t^{(1)} \leftarrow t$ ▷ Update evaluation history
  $\mathbf{x}(t_{i-1}) \leftarrow \mathbf{x}(t) - (t - t_{i-1})\, \tilde{\mathbf{v}}(t)$ ▷ Eq. 1
**return** $\mathbf{x}(t_0)$

# A Algorithm

Algorithm 1 summarizes the inference procedure of REVE. Timesteps are divided into full-image evaluations $\mathcal{T}_{\text{full}}$, edit-region evaluations $\mathcal{T}_{\text{edit}}$, and skipped timesteps $\mathcal{S}$, and the mask is computed at partition timesteps $\mathcal{T}_{\text{part}} \subseteq \mathcal{T}_{\text{full}}$. We denote by $t_{\text{full}}$ and $t_{\text{part}}$ the most recent full-image evaluation and partition timesteps, by $\tilde{\mathbf{v}}(t)$ the velocity used in the Euler update, and by superscripts $E$ and $N$ the edit and non-edit tokens under $M(t_{\text{part}})$. In the default schedule (Sec. 5), $\mathcal{T}_{\text{full}}$ comprises steps 1–6, 16, 27, and 28, $\mathcal{T}_{\text{edit}}$ comprises steps 9, 12, 20, and 24, and $\mathcal{T}_{\text{part}}$ comprises steps 6 and 16.

# B Frequency-domain analysis

We provide details for the frequency-domain analysis in Sec. 4.3 and Fig. 3 (c)–(d). Let $\mathcal{S}$ denote the set of skipped timesteps. At each $t \in \mathcal{S}$, we analyze three quantities: the predicted velocity $\mathbf{v}_\theta(t)$, the extrapolation increment $\mathbf{v}_\delta(t)$ in Eq. 5, and the temporal variation $\mathbf{v}_\delta^*(t) = \mathbf{v}_\theta(t) - \mathbf{v}_\theta(t^{(1)})$, where $t^{(1)}$ is the most recent evaluation timestep before $t$. For each quantity, we reshape its $L$ tokens into their $H \times W$ spatial layout ($L = HW$) and apply a two-dimensional Fourier transform to each channel over the entire image. We then normalize radial frequencies by their maximum value on the discrete Fourier grid and divide $[0, 1]$ into eight equal-width, non-overlapping bands, indexed by $f \in \{1, \ldots, 8\}$ in ascending order of frequency. The band-pass operator is defined as:

$$[\cdot]_f = \mathcal{F}^{-1}(B_f \odot \mathcal{F}(\cdot)), \tag{7}$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the two-dimensional Fourier transform and its inverse, $B_f$ is the binary indicator of band $f$, and $\odot$ denotes elementwise multiplication. We apply the band-pass filter before restricting each quantity to the edit region. All norms below are computed over the filtered quantities within the edit region, across all of its tokens and channels.

The *relative temporal variation* measures, in each band, the magnitude of the component missed by velocity reuse relative to that of the predicted velocity:

$$R_f(t) = \frac{\|[\mathbf{v}_\delta^*(t)]_f\|_2}{\|[\mathbf{v}_\theta(t)]_f\|_2}, \quad R_f = \frac{1}{|\mathcal{S}|} \sum_{t \in \mathcal{S}} R_f(t). \tag{8}$$

A larger $R_f$ indicates that velocity reuse neglects a larger relative change in band $f$.

To compare the distribution of energy across bands, we define the *fractional energy* of a quantity at a timestep as:

$$E_f(\cdot) = \frac{\|[\cdot]_f\|_2^2}{\sum_{f'=1}^{8} \|[\cdot]_{f'}\|_2^2}. \tag{9}$$

We compute $E_f$ separately for $\mathbf{v}_\theta(t)$, $\mathbf{v}_\delta(t)$, and $\mathbf{v}_\delta^*(t)$, and average it over $t \in \mathcal{S}$. This normalization allows their frequency profiles to be compared independently of their overall magnitudes.

## C Detail fidelity

Detail Fidelity (DF) measures how acceleration affects fine-detail strength in the edited image. It compares the mean luminance gradient magnitudes of accelerated and unaccelerated outputs from the same model within the edit region. The metric penalizes both attenuation (e.g., blurred textures) and amplification (e.g., spurious artifacts) of gradient strength relative to the unaccelerated output.

Let $I_{\text{acc}}, I_{\text{unacc}} \in \mathbb{R}^{H\times W\times 3}$ denote the accelerated and unaccelerated RGB outputs, respectively. Let $\Omega_{\text{edit}} \subseteq \{1,\dots,H\}\times\{1,\dots,W\}$ denote the edit region defined by a pseudo ground-truth mask. The mask is obtained by localizing the target with GroundingDINO (Liu et al., 2024) and segmenting it with SAM2 (Ravi et al., 2025) in both the reference image and the unaccelerated edited image. For each output $I \in \{I_{\text{acc}}, I_{\text{unacc}}\}$, the luminance at pixel $p$ is:

$$Y_I(p) = 0.299 I_R(p) + 0.587 I_G(p) + 0.114 I_B(p), \tag{10}$$

where $I_R$, $I_G$, and $I_B$ denote RGB channel intensities on the $[0, 255]$ scale. The mean luminance gradient magnitude within the edit region is:

$$G(I;\Omega_{\text{edit}}) = \frac{1}{|\Omega_{\text{edit}}|} \sum_{p\in\Omega_{\text{edit}}} \sqrt{\big(D_x Y_I(p)\big)^2 + \big(D_y Y_I(p)\big)^2}, \tag{11}$$

where $D_x$ and $D_y$ denote horizontal and vertical finite differences with unit pixel spacing. We define DF as the absolute log-ratio of the two mean gradient magnitudes:

$$\text{DF}(I_{\text{acc}}, I_{\text{unacc}}; \Omega_{\text{edit}}) = \left| \ln \frac{\max\{G(I_{\text{acc}};\Omega_{\text{edit}}), \epsilon\}}{\max\{G(I_{\text{unacc}};\Omega_{\text{edit}}), \epsilon\}} \right|, \tag{12}$$

where $\epsilon > 0$ is a small constant for numerical stability. The absolute log-ratio assigns the same penalty to a ratio and its reciprocal, so that relative increases and decreases in gradient strength are penalized symmetrically. A lower DF indicates that the accelerated output more closely matches the unaccelerated output in mean gradient magnitude within the edit region; DF is zero when the two magnitudes are equal.

## D GIE-Bench comparison results

Tab. 4 presents quantitative comparison results on GIE-Bench (Qian et al., 2025), obtained under the same settings as in Sec. 5. Consistent with the results on GEdit-Bench, REVE achieves the best scores on all six editing quality metrics and the highest speedup ($2.55$–$2.73\times$) among the compared acceleration methods for each of the three models. The SC and PQ scores of REVE remain within $0.052$ and $0.032$ of the unaccelerated scores, respectively. On Boogu-Image-0.1-Edit, REVE achieves a slightly higher PQ than the unaccelerated model ($8.476$ versus $8.470$). Compared with RegionE (Chen et al., 2026b), REVE improves all six editing quality metrics at lower latency across the three models, reducing DF by 26–30%. For instance, on LongCat-Image-Edit, DF decreases from $0.045$ to $0.033$ and LPIPS from $0.052$ to $0.042$, while PSNR increases from $30.36$ to $31.34$ and speedup from $2.41\times$ to $2.55\times$. On Step1X-Edit, TaylorSeer offers a similar speedup ($2.68\times$ versus $2.73\times$), but REVE achieves substantially lower DF ($0.045$ versus $0.116$) and higher PSNR ($32.56$ versus $27.37$).

Table 4: **Comparison of editing quality and inference efficiency on GIE-Bench.** Notation follows Tab. 1. PSNR, SSIM, and LPIPS are computed against the unaccelerated results of the same model. Best and second-best results among acceleration methods are shown in **bold** and underlined, respectively.

| Method | Type | Editing Quality | | | | | | Editing Efficiency | |
|---|---|---|---|---|---|---|---|---|---|
| | | SC ↑ | PQ ↑ | DF ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Lat. ↓ | Spd. ↑ |
| Step1X-Edit | | 8.205 | 8.581 | 0.000 | – | – | – | 79.03 | 1.00× |
| + TaylorSeer | F | 8.100 | 8.467 | 0.116 | 27.37 | 0.907 | 0.087 | 29.50 | 2.68× |
| + TeaCache | T | 8.113 | 8.510 | 0.105 | 29.16 | 0.921 | 0.076 | 31.03 | 2.55× |
| + RAS | S | 8.051 | 8.399 | 0.084 | 28.65 | 0.895 | 0.090 | 40.50 | 1.95× |
| + SpotEdit | S | 7.959 | 8.466 | 0.159 | 27.63 | 0.920 | 0.070 | 36.28 | 2.18× |
| + RegionE | S & T | 8.162 | 8.520 | 0.061 | 32.02 | 0.947 | 0.040 | 29.93 | 2.64× |
| + REVE (Ours) | S & T | **8.188** | **8.554** | **0.045** | **32.56** | **0.951** | **0.035** | **28.93** | **2.73×** |
| Boogu-Image-0.1-Edit | | 8.508 | 8.470 | 0.000 | – | – | – | 97.46 | 1.00× |
| + TaylorSeer | F | 8.136 | 8.146 | 0.274 | 24.01 | 0.847 | 0.147 | 41.95 | 2.32× |
| + TeaCache | T | 8.448 | 8.447 | 0.082 | 30.49 | 0.940 | 0.045 | 40.66 | 2.40× |
| + RAS | S | 8.278 | 8.205 | 0.095 | 27.40 | 0.881 | 0.091 | 45.88 | 2.12× |
| + SpotEdit | S | 8.345 | 8.393 | 0.077 | 27.66 | 0.879 | 0.080 | 52.74 | 1.85× |
| + RegionE | S & T | 8.448 | 8.452 | 0.047 | 32.31 | 0.946 | 0.035 | 37.78 | 2.58× |
| + REVE (Ours) | S & T | **8.474** | **8.476** | **0.033** | **33.22** | **0.951** | **0.030** | **36.63** | **2.66×** |
| LongCat-Image-Edit | | 8.717 | 8.684 | 0.000 | – | – | – | 36.18 | 1.00× |
| + TaylorSeer | F | 8.498 | 8.555 | 0.089 | 26.16 | 0.888 | 0.090 | 14.77 | 2.45× |
| + TeaCache | T | 8.631 | 8.628 | 0.090 | 25.89 | 0.886 | 0.093 | 14.32 | 2.53× |
| + RAS | S | 8.490 | 8.352 | 0.083 | 25.60 | 0.836 | 0.130 | 19.68 | 1.84× |
| + SpotEdit | S | 8.603 | 8.569 | 0.064 | 26.38 | 0.881 | 0.086 | 23.37 | 1.55× |
| + RegionE | S & T | 8.653 | 8.625 | 0.045 | 30.36 | 0.923 | 0.052 | 15.00 | 2.41× |
| + REVE (Ours) | S & T | **8.665** | **8.652** | **0.033** | **31.34** | **0.934** | **0.042** | **14.19** | **2.55×** |

Table 5: **Ablation of the threshold $\tau$.** Results are obtained using only spatial acceleration on a 330-sample subset of GEdit-Bench. The default setting is highlighted in green. Best and second-best results are shown in **bold** and underlined, respectively.

| $\tau$ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Spd. ↑ |
|---|---|---|---|---|
| 0.70 | 30.75 | 0.935 | 0.066 | **1.95×** |
| 0.74 | 31.21 | 0.943 | 0.057 | 1.91× |
| 0.78 | 31.55 | 0.947 | 0.051 | 1.86× |
| 0.82 | 31.80 | 0.950 | 0.047 | 1.82× |
| 0.86 | 32.04 | 0.952 | 0.045 | 1.75× |
| 0.90 | **32.48** | **0.955** | **0.042** | 1.61× |

## E Ablation of the threshold

The threshold $\tau$ controls the extent of the edit region: tokens whose agreement scores (Eq. 3) fall below $\tau$ are identified as edit tokens, so increasing $\tau$ expands the edit region processed by the model. We select the default $\tau = 0.82$ by comparing the predicted masks with pseudo ground-truth masks (Appendix C) on the subject-replace task of GEdit-Bench. To assess sensitivity to $\tau$, we evaluate $\tau \in \{0.70, 0.74, 0.78, 0.82, 0.86, 0.90\}$ on a 330-sample subset of GEdit-Bench comprising 30 samples from each of its 11 tasks, applying only spatial acceleration to isolate the effect of region partitioning. Tab. 5 reveals a consistent trade-off between fidelity and efficiency. As $\tau$ increases from 0.70 to 0.90, fidelity to the unaccelerated outputs improves monotonically, with PSNR rising from 30.75 to 32.48, SSIM from 0.935 to 0.955, and LPIPS falling from 0.066 to 0.042. However, processing more tokens per model evaluation reduces the speedup from 1.95× to 1.61×. Near the default, for $\tau \in \{0.78, 0.82, 0.86\}$, PSNR varies from 31.55 to 32.04 and LPIPS from 0.045 to 0.051, which indicates modest sensitivity. These results support $\tau = 0.82$ as a balanced choice, which we use for all models and benchmarks in Tabs. 1 and 4 without retuning.

## F Ablation of the extrapolation strength

We ablate the extrapolation strength $\gamma$ in Eq. 6 on the same 330-sample subset of GIE-Bench as Appendix E. Since $\gamma$ only changes the weight of the extrapolation increment, all settings share the

Table 6: **Ablation of the extrapolation strength $\gamma$.** Results are obtained on a 330-sample subset of GEdit-Bench. The default setting is highlighted in green. Best and second-best results are shown in **bold** and underlined, respectively.

| $\gamma$ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Spd. ↑ |
|---|---|---|---|---|
| 0.1 | 31.09 | 0.942 | 0.055 | 2.68× |
| 0.5 | 31.66 | **0.945** | **0.052** | 2.68× |
| 1.0 | **31.71** | 0.944 | 0.054 | 2.68× |
| 1.5 | 31.68 | 0.943 | 0.055 | 2.68× |
| 2.0 | 31.66 | 0.942 | 0.057 | 2.68× |

Table 7: **Ablation of extrapolation order.** Order 0 corresponds to velocity reuse, and order 1 is our default. Best and second-best results among accelerated variants are shown in **bold** and underlined, respectively.

| Method | Editing Quality | | | | | | Editing Efficiency | |
|---|---|---|---|---|---|---|---|---|
| | SC ↑ | PQ ↑ | DF ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Lat. ↓ | Spd. ↑ |
| Step1X-Edit | 8.287 | 8.107 | 0.000 | – | – | – | 79.11 | 1.00× |
| Order 0 | 8.218 | 8.036 | 0.073 | 31.71 | 0.941 | 0.057 | 29.47 | 2.68× |
| Order 1 | **8.238** | **8.050** | **0.046** | **31.79** | **0.947** | **0.049** | 29.47 | 2.68× |
| Order 2 | 8.190 | 8.020 | 0.048 | 31.26 | 0.944 | 0.054 | 29.47 | 2.68× |
| Order 3 | 7.748 | 7.361 | 0.090 | 27.67 | 0.905 | 0.099 | 29.47 | 2.68× |

same $2.68\times$ speedup. As $\alpha(t) = (1-t)^\gamma$, a smaller $\gamma$ assigns larger weights to extrapolation in early denoising steps, whereas a larger $\gamma$ attenuates extrapolation over a longer portion of the denoising trajectory. As shown in Tab. 6, $\gamma = 0.1$ yields the lowest PSNR of $31.09$, suggesting that strong extrapolation during early denoising interferes with structure formation. Increasing $\gamma$ to $0.5$ raises PSNR to $31.66$ and yields the best SSIM ($0.945$) and LPIPS ($0.052$). Further increasing $\gamma$ keeps PSNR within $31.66$–$31.71$ but gradually worsens SSIM to $0.942$ and LPIPS to $0.057$ at $\gamma = 2.0$, suggesting that excessive attenuation limits the contribution of extrapolation to detail refinement. We therefore set $\gamma = 0.5$ by default.

## G Ablation of the extrapolation order

REVE constructs its default first-order extrapolation increment from the two most recent evaluated velocities (Eq. 5). To examine the effect of extrapolation order, we interpolate the velocities from the $k+1$ most recent evaluations with a polynomial of degree at most $k$. Let $t^{(j)}$ denote the timestep of the $j$-th most recent evaluation. At a skipped timestep $t$, the extrapolated velocity is given by the Lagrange form:

$$\hat{\mathbf{v}}^{(k)}(t) = \sum_{j=1}^{k+1} w_j(t)\, \mathbf{v}_\theta(t^{(j)}), \quad w_j(t) = \prod_{\substack{m=1 \\ m\neq j}}^{k+1} \frac{t - t^{(m)}}{t^{(j)} - t^{(m)}}. \tag{13}$$

We replace the increment in Eq. 4 with $\mathbf{v}_\delta^{(k)}(t) = \hat{\mathbf{v}}^{(k)}(t) - \mathbf{v}_\theta(t^{(1)})$, retaining the adaptive weight $\alpha(t)$ and applying extrapolation only within the edit region. Here, $k = 0$ yields $\mathbf{v}_\delta^{(0)}(t) = \mathbf{0}$ and thus velocity reuse, while $k = 1$ recovers our default increment. We compare $k \in \{0, 1, 2, 3\}$ using Step1X-Edit on GEdit-Bench under the experimental settings in Sec. 5.

Tab. 7 and Fig. 6 show that first-order extrapolation improves editing quality over velocity reuse, whereas increasing the order further degrades quality. Velocity reuse produces overly smooth details on the robot and the horse, with DF and LPIPS of $0.073$ and $0.057$, respectively. First-order extrapolation recovers finer details, reducing DF to $0.046$ and LPIPS to $0.049$ while improving SC, PQ, PSNR, and SSIM. Second-order extrapolation sharpens some features on the robot's head but introduces unnatural magenta patches on the horse. Its DF ($0.048$) remains close to that of first order, yet SC, PQ, and PSNR fall to $8.190$, $8.020$, and $31.26$, respectively, all below those of velocity reuse. Third-order extrapolation produces severe color and structural distortions, accompanied by further declines in SC and PQ to $7.748$ and $7.361$, an increase in DF to $0.090$, and a decrease in PSNR to $27.67$. With all tested orders achieving the same reported $2.68\times$ speedup, first-order extrapolation provides the best scores on all six quality metrics, supporting our default choice.

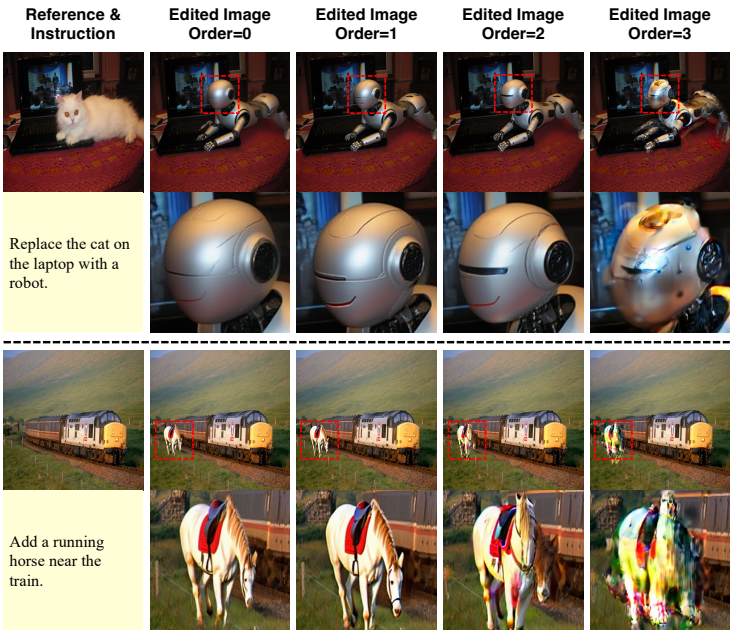


Figure 6: **Effect of extrapolation order on editing results.** First-order extrapolation recovers finer details than velocity reuse (order 0). Second-order extrapolation sharpens some features but introduces color artifacts, while third-order extrapolation produces severe color and structural distortions.




Figure 7: **Region partitioning masks at different thresholds $\tau$.** Rows 1–2 show local edits, and rows 3–4 show global edits. Highlighted and darkened tokens denote edit and non-edit tokens, respectively.

## H MASK VISUALIZATION

Fig. 7 complements the quantitative ablation in Appendix E by visualizing the effect of $\tau$ on mask coverage for local and global edits. For local edits (rows 1–2), low thresholds ($\tau \leq 0.74$) identify the core target regions but leave gaps, including parts of the glowing crystal in row 1 and the torso in row 2. Increasing $\tau$ to 0.78–0.82 produces more complete coverage: the masks in row 1 encompass the

crystal and the extent of the replacement box, while the masks in row 2 cover both the original and raised arm positions. Higher thresholds also include instruction-irrelevant content, with scattered background tokens appearing at $\tau = 0.86$ in row 2 and coverage extending beyond the target regions in both examples at $\tau = 0.90$, which reduces the savings from spatial acceleration. For global edits (rows 3–4), the masks cover nearly the entire image from $\tau = 0.78$ onward in row 3 and at all tested thresholds in row 4. Lower thresholds leave scattered gaps in row 3, excluding parts of the scene that require modification. Together with the quantitative results in Tab. 5, these examples support the default $\tau = 0.82$, which covers the target content in both local and global edits while excluding instruction-irrelevant regions from computation.

## I LIMITATIONS AND FUTURE WORK

REVE focuses on how to approximate velocities at skipped timesteps, while its default evaluation schedule remains fixed: it retains the full-image evaluations of RegionE (Chen et al., 2026b) and evaluates the edit region at predetermined steps (Sec. 5). In addition, the extrapolation weight $\alpha(t)$ depends only on the timestep, with $\gamma$ shared across models, and thus does not adapt to the velocity variation of individual samples or tokens. Moreover, the savings from spatial acceleration depend on the extent of the edit region: For global edits, the edit region covers nearly the entire image, leaving few noisy image tokens that can be excluded from model evaluations.

Future work could use velocity variation to jointly adapt the evaluation schedule and extrapolation strength. Large extrapolation increments could indicate when a model evaluation is needed, while sample- or token-specific velocity variation could guide the weight $\alpha(t)$. Extending REVE to video editing is another promising direction: edits localized in space and time could enable selective computation across both image regions and video frames.

## J AI USE STATEMENT

In this work, we used generative AI tools to assist with experimental implementation. We did not use generative AI tools to generate synthetic datasets or to prove mathematical claims, and the remaining tasks with required disclosure are not applicable to this work. Additionally, we used generative AI tools to identify relevant literature, draft manuscript outlines, and polish sentences. We have reviewed all AI-assisted work by consulting the original references for all cited literature and by reviewing and testing all AI-assisted code. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI tools.