TAILSTEER: TRAJECTORY-TAIL STEERING FOR DRIFT-RESISTANT MULTI-TURN IMAGE EDITING

Anonymous authors
Paper under double-blind review

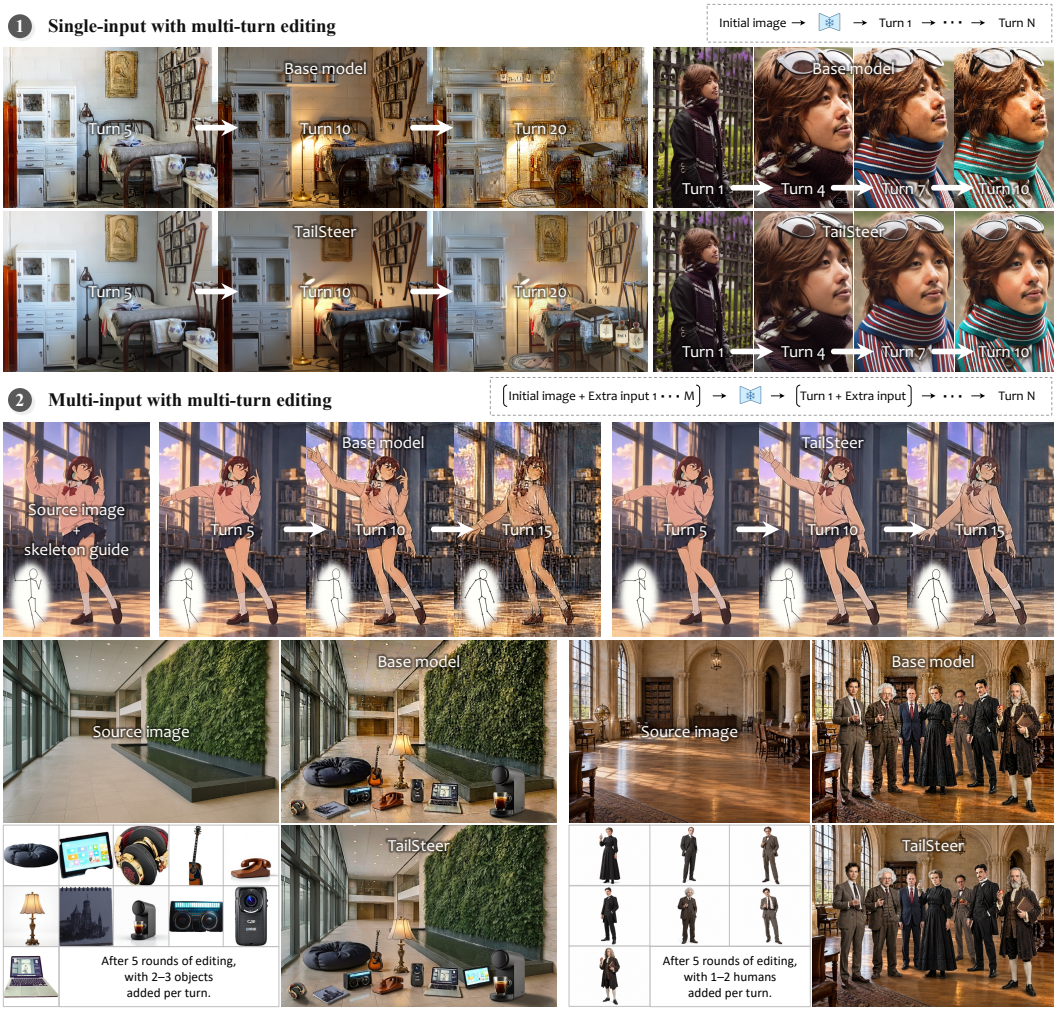


Figure 1: **Single- and multi-input multi-turn editing.** TailSteer substantially reduces cumulative drift over editing turns while retaining the editing capability of the base model (FLUX.2 Klein-9B).

ABSTRACT

When using most generative image editors to edit an image over multiple turns, minor unintended changes may accumulate, resulting in a corrupted result in the end. This long-term editing drift is normally mitigated through heavy large-scale pretraining or additional modules that may introduce inference overhead. In this work, we introduce **TailSteer**, a simple post-training framework that improves the multi-turn robustness of pretrained editors without changing their architecture or introducing additional computational overhead. TailSteer follows a simple principle: drift is often introduced in the last few denoising steps. Therefore, it keeps earlier denoising dynamics mostly unchanged and corrects only the tail of the diffusion trajectory to preserve fine-grained details in the source image.

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.

These corrections are learned from the editor's own denoising rollouts, using unchanged source content as a preservation reference rather than requiring curated ground-truth editing pairs. Experiments on three backbones (FLUX 1, FLUX 2, and SenseNova families) show that TailSteer can consistently reduce cumulative non-target drift: after TailSteer fine-tuning, the models can normally support 5–10 editing turns without significant errors, while the original models often produce obvious artifacts in fewer than five turns.

1 INTRODUCTION

Large-scale generative models have substantially advanced instruction-based image editing, achieving increasingly strong instruction following, visual fidelity, and content preservation in single-turn editing (OpenAI, 2025a;b; Black Forest Labs et al., 2025; Black Forest Labs, 2025a; Google, 2025). At the same time, image generation is increasingly embedded in interactive and conversational creative workflows. Recent systems, such as Nano Banana 2, ChatGPT Images, Reve, or Muse Image, allow users to repeatedly refine an image through natural-language feedback, visual references, and agentic interactions (Google, 2026; OpenAI, 2026b; Reve, 2026; Meta, 2026). This interaction paradigm better matches how visual content is actually created: rather than specifying a complete target at once, users progressively develop an idea through a sequence of edits.

However, applying a one-shot editor to iterative editing introduces a critical challenge: small errors in each edit can accumulate over a long editing sequence. In a multi-turn workflow, each edited output becomes the input to the next turn, causing unintended changes to be repeatedly inherited by subsequent generations. As illustrated in Fig. 1, repeated editing can wash out fine textures, alter non-target object or facial details, distort background structures, and eventually produce visible artifacts. Thus, strong single-turn editing performance does not necessarily translate into robust long-horizon editing.

Existing approaches to improving multi-turn image editing can be broadly categorized into three families. i) *Adding auxiliary modules or constraints* to a pretrained editor, by introducing spatial guidance (e.g., masks), specially trained LoRA adapters, or frequency corrections (Couairon et al., 2023; dx8152, 2026; Liao et al., 2026; Wang et al., 2026). These mechanisms improve preservation by sacrificing or constraining part of the large-scale generative prior learned by the pretrained model, and are therefore often limited to a relatively narrow range of localized edit types. ii) *Refining the representation space* to better support image editing. As reconstruction fidelity and generative capability are widely recognized as a latent-generative dilemma (Yao et al., 2025; Dieleman, 2025), Black Forest Labs specifically treats the VAE representation to obtain a more suitable editing space (Black Forest Labs, 2025b), while other approaches bypass the VAE and perform pixel-diffusion (Diao et al., 2026b). Although these designs effectively alleviate errors within a single editing turn, they still suffer from noticeable degradation across repeated edits. iii) *Maintaining explicit memory across edits* within creative workflows, by caching latent states or previous visual context (Zhu et al., 2025; Kim et al., 2025), or by leveraging reasoning models for agentic planning and iterative refinement in commercial systems (Meta, 2026; OpenAI, 2026b). While such mechanisms help preserve historical information across turns, they may introduce significant computational overhead and also their performance remains bounded by the robustness of each editing step produced by the underlying base editor.

Although these approaches can improve multi-turn consistency, they often require auxiliary modules, alternative representations, or dedicated models. We instead directly improve the pretrained editor, so that multi-turn robustness is built into the generator. This will not introduce additional inference components and also remain complementary to existing workflow-level solutions.

To improve the existing editor, we first analyze why drifting happens in image editing, by inspecting the denoising trajectory. Prior studies show that diffusion sampling follows a coarse-to-fine progression: low-frequency structure is established early, whereas high-frequency details are mainly recovered toward the trajectory tail (Qian et al., 2024). This is particularly relevant to multi-turn editing, where repeated edits progressively degrade high-frequency information (Liao et al., 2026). Meanwhile, at inference time, each denoising state is produced from the model's preceding predictions, so prediction errors can propagate along the sampling trajectory (Ning et al., 2023; 2024; Li et al., 2024). Fig. 2 reveals these two effects in the tail of the trajectory: low-frequency similarity

is already high before the final steps, while higher-frequency similarity rises sharply near the tail; at the same time, preservation error in unchanged regions continues to decrease but remains nonzero at the endpoint. This analysis suggests that, to reduce multi-turn drift, we should modify the trajectory tail, which has a greater impact on the fine-grained details of the output.

Based on this insight, we introduce **TailSteer**, a post-training method that improves multi-turn robustness by steering the tail of a pretrained editor's denoising trajectory. TailSteer is motivated by a trajectory-level train–test gap: during inference, the editor repeatedly operates on states produced by its own predictions, allowing small errors to accumulate along the trajectory and affect the fine-grained details finalized near its tail. Rather than attaching specialized mechanisms on top of a pretrained editor or training dedicated models for multi-turn consistency from scratch, *TailSteer fine-tunes the existing editor on states reached by its own rollouts*. This helps to further reduce train–test gap. With this design, our training objective progressively steers the later trajectory toward a preservation-aware endpoint while keeping the earlier dynamics mostly unchanged.

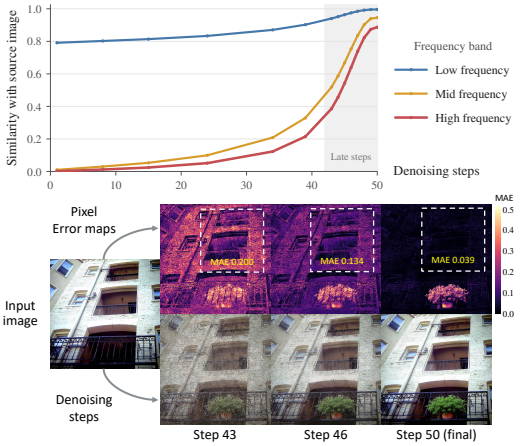


Figure 2: **Trajectory tail remains correctable.** Low-frequency content is recovered early, whereas higher-frequency details are primarily recovered near the tail (top), while preservation error in unchanged regions continues to decrease but remains non-zero at the endpoint (bottom).

TailSteer's tail-focused design brings several practical advantages. First, by preserving the pretrained model's earlier denoising dynamics and concentrating corrective steering near the trajectory tail, TailSteer preserves the original model's editing capability acquired through large-scale training while improving robustness to repeated editing. Our experiments show that TailSteer substantially improves multi-turn robustness while closely matching the original editor's single-turn performance. Second, TailSteer generalizes naturally across different editing models by operating directly on their multi-step denoising trajectories, rather than depending on a particular latent representation, attention mechanism, or model architecture. We validate this trajectory-centric design on FLUX Black Forest Labs (2025a) and SenseNova Diao et al. (2026a), and further show that the same post-training strategy improves multi-turn robustness when these models are conditioned on multiple input images. Finally, TailSteer requires no specially curated pairs of source images and ground-truth edited targets. Instead, it constructs supervision from source images and the pretrained editor's own rollouts, allowing the training data to be scaled and diversified without additional target annotation. Empirically, we find that increasing both the amount and diversity of rollout data further improves robustness.

In summary, our contributions are three-fold: (1) We identify the denoising trajectory tail as a key intervention point for multi-turn editing, where fine-grained details are finalized while residual prediction errors induced by the trajectory-level train–test gap remain correctable. (2) We introduce **TailSteer**, a tail-focused post-training framework that learns from the pretrained editor's own rollouts and steers the late trajectory toward a preservation-aware endpoint while preserving its earlier dynamics. (3) Extensive experiments show that TailSteer substantially slows the accumulation of non-target drift over repeated edits, improving masked PSNR by up to 4.95 dB after 10 turns while largely retaining the pretrained model's single-turn editing capability. GEdit and ImgEdit overall scores remain within 0.07 and 0.10 of the original models, respectively.

2 Related Work

Strong single-turn performance does not necessarily translate into reliable multi-turn interaction. Similar degradation has been observed in LLMs (Laban et al., 2026). This issue is increasingly relevant to agentic image-generation workflows, where generated outputs are repeatedly reused and refined. Existing work addresses this challenge from several complementary directions.

Representation fidelity. Reconstruction fidelity and generative learnability can impose competing demands on latent representations (Yao et al., 2025). FLUX.2 revisits autoencoder design to improve reconstruction while retaining a learnable latent space (Black Forest Labs, 2025b), whereas SenseNova-U1 bypasses the VAE bottleneck through pixel-space flow matching (Diao et al., 2026b). These developments improve the representation used for editing, addressing a different aspect of preservation from refining the sampling dynamics of an existing editor.

Inference-time preservation. Blended Diffusion and DiffEdit use spatial masks to localize edits (Avrahami et al., 2022; Couairon et al., 2023). For repeated editing, FreqEdit injects high-frequency information from reference velocity fields (Liao et al., 2026), while VAE-LFA aligns low-frequency latent statistics across editing rounds (Wang et al., 2026). Zhou et al. (2025) combine flow-based inversion with linear-quadratic control and adaptive attention to mitigate error accumulation. These methods enforce preservation through interventions in the inference procedure.

Feature reuse and editing memory. KV-Edit preserves background content by caching and reusing key–value features (Zhu et al., 2025). Layer-wise Memory explicitly retains latent states, masks, and prompt embeddings from previous edits, using latent blending and attention control to maintain consistency across successive modifications (Kim et al., 2025). The former reuses source-image features, while the latter maintains an explicit editing history.

Learning multi-turn robustness. Recent work also trains generators for iterative editing. VINCIE learns in-context editing from video-derived sequences (Qu et al., 2026). MT-EditFlow optimizes flow-based editors through reinforcement learning with rewards aggregated across editing turns (Huang et al., 2026). TailSteer instead improves multi-turn robustness by post-training the pretrained editor along its own denoising trajectories, with preservation refinement concentrated near the trajectory tail. It requires neither model changes nor additional inference-time components.

3 METHOD

TailSteer follows a simple principle: preserve the pretrained dynamics along the early denoising trajectory while steering only its tail toward better preservation. We present the framework through three progressively connected parts. First, we construct preservation-aware supervision directly from the frozen editor's own rollouts, forming a "data-free" post-training paradigm without manually curated ground-truth editing pairs. Second, we study a one-step terminal intervention with supervision applied only to the final sampling step. This verifies that endpoint correction alone can already improve multi-turn stability. Finally, we extend this single-step intervention to **TailSteer**, distributing preservation-oriented steering over multiple steps in the trajectory tail while retaining the pretrained dynamics of the earlier steps.

3.1 PRESERVATION-AWARE TRAJECTORY ENDPOINT

To steer the trajectory tail, the first question is what the trajectory should be steered toward. This target should remain aligned with the model's own editing trajectory while correcting only the unintended deviations accumulated during sampling. However, conventional manually curated source–target editing pairs are poorly suited to this purpose. Given the same source image and instruction, a generative editor may produce different yet valid outputs under different sampling seeds. For example, an instruction to add an apple may yield apples with different sizes, colors, or surface patterns. Our goal is not to force the model toward one particular edit realization, but to correct the rollout-induced errors while preserving the edit already chosen by the model.

Remarkably, the preservation target is already available in the source image: the content that is not requested to change. For example, when replacing an object in a local region, the surrounding subject and background should remain unchanged. Although modern editors exhibit strong instruction following to localize the requested edit, their denoising rollouts still introduce small deviations into such non-target content. These model-induced deviations therefore provide exactly the preservation errors that TailSteer aims to correct.

To obtain reliable supervision, we construct two complementary trajectory types in which the resulting edit is expected to preserve part of the source image, *identity editing* and *local editing*,

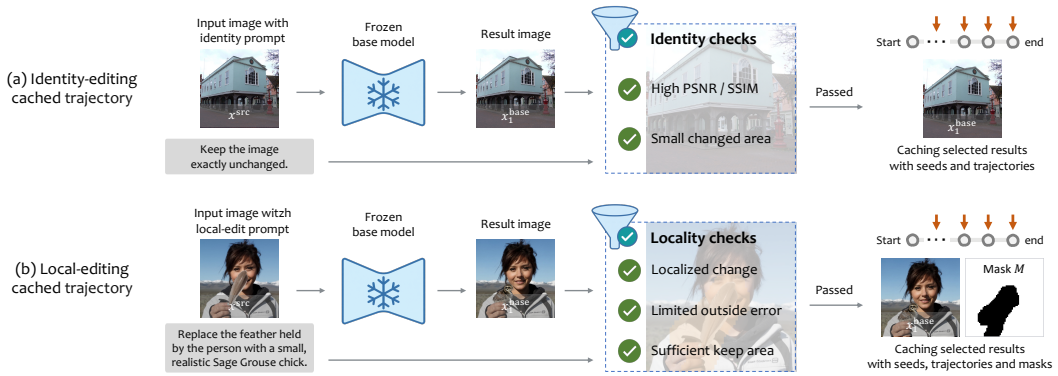


Figure 3: **Data curation of preservation-aware trajectory endpoints.** We construct supervision from rollouts of the frozen base editor using two complementary edit types: (a) identity editing provides full-image preservation trajectories, while (b) local editing provides partial-preservation trajectories with spatially localized changes. Only samples that pass the corresponding sanity checks are retained and cached as training trajectories for subsequent TailSteer optimization.

as illustrated in Fig. 3. Identity editing requests no visual change and therefore provides full-image preservation supervision, whereas local editing confines the requested modification to a localized region, leaving the remaining content available for partial preservation supervision. We generate rollouts with the frozen base editor and filter them using the corresponding identity or locality checks, retaining only trajectories with reliable preservation regions. For accepted edits, these regions are further converted into preservation masks, as described below.

For local editing, we further derive a binary preservation mask from the difference between the source image and the editor's generated result. The mask is not specified in advance as we do not require the generated edit to match a ground-truth target; instead, we only require the observed change in the masked regions to be sufficiently localized. Let x^{src} denote the source state, x_1^{base} the endpoint of the frozen base editor's rollout, and M the resulting preservation mask, where $M = 1$ denotes regions that should remain unchanged. We construct preservation-aware target endpoint as

$$x_1^{\mathrm{tgt}} = M \odot x^{\mathrm{src}} + (1 - M) \odot x_1^{\mathrm{base}}. \tag{1}$$

For identity editing, we simply set $M = 1$. Thus, x_1^{tgt} restores the source content in preservation regions while retaining the edit realization produced by the base editor elsewhere.

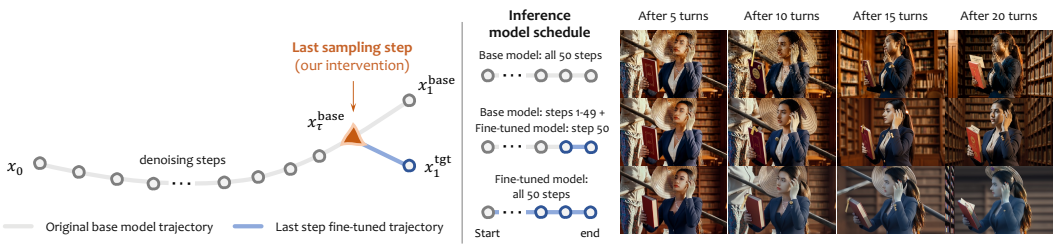
For each accepted rollout, we cache the source state, selected intermediate states, the terminal state, the preservation mask, and the sampling seed for subsequent post-training. Importantly, the two restricted prompt families are used only to obtain reliable preservation supervision; TailSteer itself learns a trajectory-level correction rather than an edit-specific behavior. We later show that this supervision generalizes beyond these training edits, improving multi-turn robustness for subject identity and multi-image-conditioned editing without sacrificing broader editability.

3.2 One-step Terminal Intervention

We first test whether the curated preservation-aware endpoint can improve multi-turn stability through the smallest possible intervention: correcting only the final sampling step. To test this native solution, we initialize from the pretrained base editor and fine-tune it exclusively on the final sampling step: starting from the cached pre-terminal state, the model is supervised toward the preservation-aware endpoint, while the preceding timesteps receive no training objective (Fig. 4 (a)).

Let $\tau < 1$ denote the pre-terminal sampling time, with time increasing from noise at 0 to output at 1. Given the cached state x_τ^{base}, the velocity toward the preservation-aware endpoint x_1^{tgt} is

$$v_\tau = \frac{x_1^{\mathrm{tgt}} - x_\tau^{\mathrm{base}}}{1 - \tau}. \tag{2}$$



(a) Terminal–step fine–tuning (b) Multi-turn editing under different model schedules

Figure 4: **One-step terminal intervention: benefits and limitations.** (a) We fine-tune FLUX.2-klein-9B with supervision only at the final sampling step, steering from the cached pre-terminal state x_τ^{base} toward the preservation-aware target x_1^{tgt}. (b) The base editor accumulates visible identity drift over repeated edits (top row). Applying the fine-tuned editor only at the final step markedly improves long-horizon preservation (middle row), whereas using it throughout sampling causes desaturation and detail loss (bottom row). Therefore, it is necessary to preserve the earlier pretrained dynamics.

We then use v_τ as the supervision target only at the terminal update, encouraging the velocity predicted from x_τ^{base} to endpoint x_1^{tgt}:

$$\mathcal{L}_{\text{Terminal}} = \mathbb{E}\left[\left\| v_\theta(x_\tau^{\text{base}}, \tau, c) - v_\tau \right\|_2^2\right], \tag{3}$$

where c denotes the editing condition. This confines the intervention to the final sampling update, directly examining how much multi-turn stability can be improved by modifying only the last step.

We evaluate this terminal intervention on FLUX.2-klein-9B with 50 sampling steps per edit (Fig. 4(b)). When the base editor is used for all 50 steps, the base editor exhibits visible facial degradation after five turns, which worsens with continued editing. In contrast, using the base editor for the first 49 steps and the fine-tuned editor only for the final update markedly reduces visible drift and better preserves subject identity, with visually stable results even after 20 turns. This demonstrates that the preservation-aware endpoint provides effective supervision for long-horizon multi-turn stability, even when correction is confined to a single sampling update.

However, applying the fine-tuned editor to all 50 steps reveals a different trade-off. Although the outputs exhibit fewer conspicuous artifacts than those of the base editor, they become desaturated and overly smooth, losing facial and scene details. Since the denoiser parameters are shared across timesteps, terminal-only supervision can also disturb the pretrained dynamics at earlier steps; preserving the original model prior therefore requires explicit model switching. Moreover, concentrating all correction in the final update leaves no opportunity for gradual adjustment. These limitations motivate our full TailSteer framework, which explicitly trains the editor to preserve the earlier pretrained dynamics while distributing the correction over a short trajectory tail, allowing a single refined editor to perform the entire sampling process with improved multi-turn stability.

3.3 Tail Steering

The terminal intervention above leaves earlier predictions unconstrained and concentrates all correction in a single update. We therefore introduce **TailSteer**, an on-policy post-training framework that preserves the pretrained dynamics over the earlier trajectory while progressively steering its tail toward the preservation-aware endpoint. Starting from the frozen base editor, we initialize a trainable copy, denoted by v_θ, as the *refined editor*. Importantly, all supervision is constructed on states reached by the refined editor's own denoising rollouts, as illustrated in Fig. 5.

Steering onset. We extend the terminal correction to a trajectory tail by introducing a steering onset $k_s < \tau$. This partitions the trajectory into a *prior region*, $0 \le k \le k_s$, and a *transition region*, $k_s < k < 1$. Before k_s, the refined editor is trained to reproduce the base dynamics; after k_s, the supervision gradually shifts toward the preservation endpoint.

Cached trajectory anchors. To obtain rollout states from the refined editor without repeatedly replaying the full trajectory, we use cached base-model states as rollout anchors. In the prior region, states are cached at intervals of at most three sampling updates, including the onset state $x_{k_s}^{\text{base}}$. For

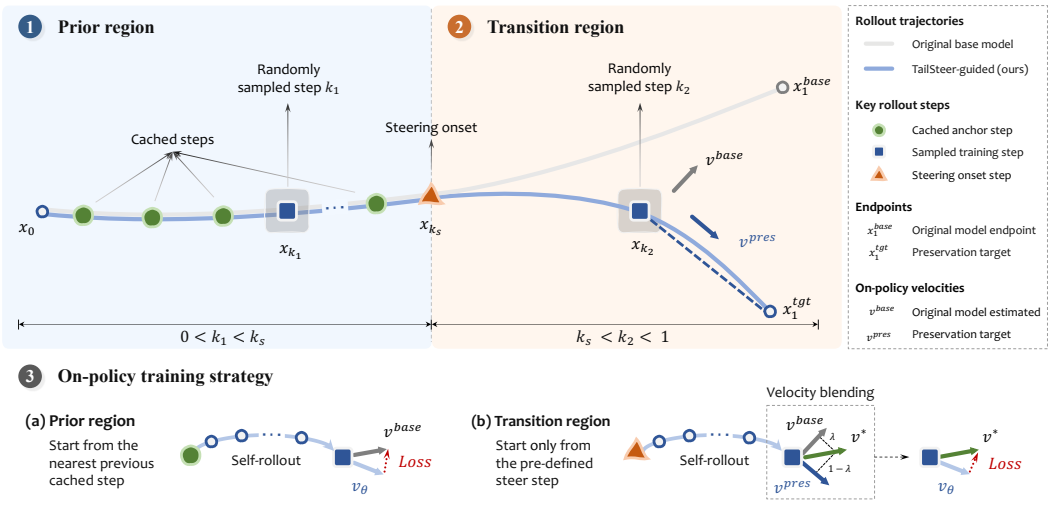


Figure 5: **On-policy Training of TailSteer.** At randomly sampled training states x_{k_1} and x_{k_2}, the refined editor receives region-dependent supervision. In the prior region, its prediction is matched to the frozen base editor's velocity v^{base}; in the transition region, the target v^* progressively blends v^{base} with the preservation velocity v^{pres} toward x_1^{tgt}. To obtain these states on-policy, the refined editor rolls out from the nearest preceding cached anchor in prior region, and from the fixed steering-onset state in transition region. The velocity target is then recomputed at the state actually reached by the refined editor, with gradients applied only to its prediction at the sampled step.

a sampled prior-region step, the refined editor rolls out from the nearest strictly preceding cached state. In the transition region, all rollouts instead start from $x_{k_s}^{\text{base}}$, with no intermediate tail states used as restart anchors. This keeps early rollouts short while allowing the refined editor to evolve along its own trajectory throughout the tail. We denote the state reached at the sampled step by x_k.

Velocity blending. At the state x_k reached by the refined editor, we evaluate the frozen base editor to obtain $v_k^{\text{base}} = v^{\text{base}}(x_k, k, c)$. We generalize the endpoint-directed velocity in Eq. 2 to $v_k^{\text{pres}} = (x_1^{\text{tgt}} - x_k)/(1-k)$, where $1-k$ is the remaining sampling time. The supervision target combines these velocities according to the trajectory region:

$$v_k^* = \begin{cases} v_k^{\text{base}}, & 0 < k \le k_s, \\ \lambda(k) v_k^{\text{base}} + [1-\lambda(k)] v_k^{\text{pres}}, & k_s < k \le \tau, \end{cases} \tag{4}$$

The weight $\lambda(k)$ decreases from 1 at k_s to 0 at τ, gradually shifting supervision from the base dynamics to endpoint correction. At the final update, the target has the same endpoint-directed form as in Sec. 3.2, now evaluated at the state reached by the refined editor. The weight schedule is detailed in the appendix.

On-policy optimization. Crucially, both v_k^{base} and v_k^{pres} are recomputed at the state x_k actually reached by the refined editor. The supervision therefore adapts to deviations along the current rollout instead of relying on targets precomputed on the frozen base trajectory. We optimize

$$\mathcal{L}_{\text{TailSteer}} = \mathbb{E}[\text{MSE}(v_\theta(x_k, k, c), \text{sg}[v_k^*])], \tag{5}$$

where the expectation is over cached examples and sampled training steps, and sg denotes stop-gradient. The rollout itself is performed without gradient tracking, so gradients are backpropagated only through the refined editor's prediction at the sampled step, while all trainable parameters of the refined editor are updated by this loss.

At inference, the refined editor performs the entire sampling trajectory on its own. All auxiliary components—including the frozen base editor, preservation targets, masks, and velocity blending—are used only during post-training, requiring no model switching, external adapters, or additional consistency mechanisms at inference. TailSteer therefore improves multi-turn robustness as an intrinsic capability of the refined editor.



Figure 6: **Qualitative comparison after 10 editing turns.** The text above each example highlights non-target content that should remain unchanged. More turn-by-turn results are in the supplements.



Figure 7: **Ablation study.** Results after 10 editing turns. TailSteer better preserves the base model's editing prior while suppressing cumulative error. More turn-by-turn results are in the supplements.

4 Experiments

4.1 Experimental Details

We evaluate TailSteer on three open-weight editors: FLUX.1 Kontext [dev] (30 sampling steps), FLUX.2 [klein] Base 9B (50 steps), and SenseNova-U1.5-8B-MoT (50 steps). The FLUX models are VAE-based latent-space flow editors, while SenseNova uses a VAE-free unified architecture for visual understanding and generation. For each backbone, we cache 3,000 local-edit and 1,000 identity-edit trajectories using source images from Open Images (Kuznetsova et al., 2020). We fine-tune the full model, including the multimodal understanding branch of SenseNova, for 2,000 updates at 1024^2 resolution using AdamW with a learning rate of 10^{-6}, BF16 precision, and an effective batch size of 16 on eight NVIDIA H100 GPUs. At inference, the refined model is used for the entire sampling trajectory with the original sampling schedule and inference settings.

4.2 Qualitative Evaluation

Visual Comparison. Using the three pretrained editors as backbones, we compare TailSteer with training-free MTC (Zhou et al., 2025), the video-trained multi-turn editor VINCIE (Qu et al., 2026), and backbone-specific enhancements: FreqEdit (Liao et al., 2026) on Kontext and the community-released Consistency LoRA V2 (dx8152, 2026) on Klein. We also evaluate GPT-Image-2.5 Sunburst (OpenAI, 2026a) as a strong proprietary generator through direct API calls, without an external agentic workflow. As shown in Fig. 6, TailSteer better preserves non-target content after 10 editing turns across all three backbones, while visible drift remains even with GPT-Image-2.5. FreqEdit further illustrates that lower drift alone does not guarantee retention of the pretrained editing behavior: although some non-target regions change less, the edited birdhouse and basket deviate noticeably from pretrained Kontext, consistent with its lower single-turn semantic consistency and GEdit overall in Table 1. TailSteer instead reduces cumulative drift while remaining closer to the pretrained editor's behavior. Full turn-by-turn results are provided in the supplementary material.

Ablation study. Fig. 7 isolates the effects of tail-focused refinement, on-policy training, and multi-step steering. With the same training data, standard SFT is trained on interpolated noisy states rather than model-rollout states, resulting in a train–test mismatch that only partially reduces drift while noticeably altering the pretrained editing behavior. Removing on-policy training leads to a similar

Table 1: **Single-turn capability retention across three backbones.** We evaluate GEdit-Bench (English), RISEBench, and ImgEdit-Bench (Basic) here. Higher is better for all metrics.

			GEdit EN ↑			RISE Accuracy (%) ↑					ImgEdit ↑
Backbone	Baseline	k_s	G_{SC}	G_{PQ}	G_{O}	Temporal	Causal	Spatial	Logical	Overall	Overall
FLUX.1 Kontext dev	Base model	–	7.17	7.20	6.52	7.06	8.89	12.00	1.18	7.50	3.75
	w/ FreqEdit	–	6.24 (−0.94)	7.30 (+0.10)	5.72 (−0.80)	5.88 (−1.18)	5.56 (−3.33)	11.00 (−1.00)	1.18 (+0.00)	6.11 (−1.39)	3.62 (−0.14)
	w/ TailSteer	24	7.16 (−0.01)	7.35 (+0.15)	6.59 (+0.07)	4.71 (−2.35)	7.78 (−1.11)	11.00 (−1.00)	2.35 (+1.18)	6.67 (−0.83)	3.74 (−0.01)
		26	7.11 (−0.07)	7.30 (+0.10)	6.52 (+0.00)	4.71 (−2.35)	8.89 (+0.00)	12.00 (+0.00)	2.35 (+1.18)	7.22 (−0.28)	3.74 (−0.01)
		28	7.13 (−0.04)	7.24 (+0.04)	6.47 (−0.06)	4.71 (−2.35)	5.56 (−3.33)	9.00 (−3.00)	1.18 (+0.00)	5.28 (−2.22)	3.69 (−0.06)
FLUX.2 Klein-9B	Base model	–	8.12	7.69	7.47	8.24	14.44	28.00	3.53	14.17	4.20
	w/ ConsistLoRA	–	6.71 (−1.41)	8.31 (+0.61)	6.58 (−0.89)	7.06 (−1.18)	8.89 (−5.56)	16.00 (−12.00)	4.71 (+1.18)	9.44 (−4.72)	3.86 (−0.34)
	w/ TailSteer	38	8.11 (−0.01)	7.75 (+0.05)	7.50 (+0.03)	7.06 (−1.18)	14.44 (+0.00)	28.00 (+0.00)	2.35 (−1.18)	13.61 (−0.56)	4.17 (−0.03)
		44	8.20 (+0.08)	7.72 (+0.03)	7.53 (+0.06)	9.41 (+1.18)	17.78 (+3.33)	24.00 (−4.00)	3.53 (+0.00)	14.17 (+0.00)	4.19 (−0.01)
		47	8.11 (−0.01)	7.65 (−0.04)	7.42 (−0.05)	8.24 (+0.00)	14.44 (+0.00)	24.00 (−4.00)	1.18 (−2.35)	12.50 (−1.67)	4.15 (−0.05)
SenseNova U1.5-8B MoT	Base model	–	9.23	7.75	8.33	24.71	36.67	46.00	15.29	31.39	4.43
	w/ TailSteer	42	9.28 (+0.05)	7.61 (−0.14)	8.28 (−0.06)	30.59 (+5.88)	34.44 (−2.22)	37.00 (−9.00)	14.12 (−1.18)	29.44 (−1.94)	4.42 (−0.01)
		45	9.25 (+0.02)	7.70 (−0.06)	8.30 (−0.03)	24.71 (+0.00)	27.78 (−8.89)	37.00 (−9.00)	15.29 (+0.00)	26.67 (−4.72)	4.33 (−0.10)
		47	9.21 (−0.01)	7.67 (−0.08)	8.28 (−0.05)	24.71 (+0.00)	34.44 (−2.22)	30.00 (−16.00)	15.29 (+0.00)	26.39 (−5.00)	4.35 (−0.08)

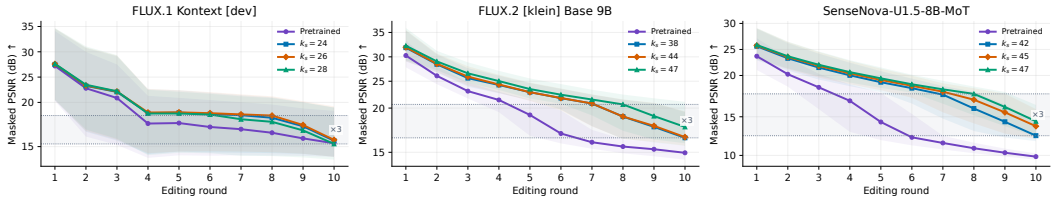


Figure 8: **Cumulative non-target preservation over editing turns.** Masked PSNR to the source is averaged over eight 10-turn sequences with a fixed non-target mask.

deviation, whereas last-step-only refinement better preserves the pretrained behavior but provides weaker drift correction. TailSteer resolves this trade-off through on-policy multi-step tail refinement, suppressing cumulative drift while minimally disturbing the pretrained editing prior.

4.3 Quantitative Evaluation

Single-turn Capability Retention. To test whether refinement for multi-turn consistency compromises single-turn editing, we evaluate GEdit-Bench (Liu et al., 2025), RISEBench (Zhao et al., 2025), and ImgEdit-Bench (Ye et al., 2025). Table 1 compares each refined backbone with its pretrained counterpart, including FreqEdit (Liao et al., 2026) for Kontext and publicly released Consistency-LoRA-v2 (dx8152, 2026) for Klein-9B. Across the reported steering onsets, TailSteer's GEdit and ImgEdit overall scores remain within 0.07 and 0.10 of the pretrained models, respectively. By comparison, FreqEdit and Consistency-LoRA-v2 reduce GEdit overall by 0.80 and 0.89; their gains in perceptual quality are accompanied by semantic-consistency losses of 0.94 and 1.41, respectively. The minimal degradation across these benchmarks supports the central design premise of TailSteer: by restricting intervention to the trajectory tail, TailSteer gains multi-turn robustness with minimal disruption to the base model's generative and editing priors.

Multi-turn Non-target Preservation. We track masked PSNR to the source over eight 10-turn local-edit sequences with a fixed non-target mask. As shown in Fig. 8, TailSteer slows non-target degradation over successive edits, with increasingly large gains over the base models on FLUX.2 [klein] Base 9B and SenseNova-U1.5-8B-MoT, reaching up to $+4.95$ dB at turn 10. The improvement is more modest on FLUX.1 Kontext [dev], which may partly reflect its guidance-distilled nature. Overall, the growing late-turn margin demonstrates reduced accumulation of non-target drift.

5 Conclusion

We presented **TailSteer**, a post-training framework that addresses cumulative drift within the pretrained editor itself. It learns from the editor's own denoising rollouts, concentrating preservation corrections near the trajectory tail while keeping earlier dynamics close to the pretrained model. Across three backbones, it improves multi-turn robustness without architectural changes or additional inference computation. Stronger preservation need not restrict what a model can edit: it can be learned by refining how the model completes each edit.

AI USE STATEMENT

AI-assisted tools were used only to polish the language and improve the clarity of the manuscript. All experiments were conducted by the authors, who also developed and verified the technical content and conclusions.

ETHICS STATEMENT

All datasets used in this work are publicly available. Our study does not involve private or sensitive data, and we do not identify additional ethical concerns specific to this work.

REPRODUCIBILITY STATEMENT

We provide the implementation details and experimental settings required for reproduction in the paper and appendix. All code will be released upon acceptance.

REFERENCES

Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18208–18218, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Avrahami_Blended_Diffusion_for_Text-Driven_Editing_of_Natural_Images_CVPR_2022_paper.html.

Black Forest Labs. FLUX.2: Frontier visual intelligence. Technical blog post, November 2025a. URL https://bfl.ai/blog/flux-2.

Black Forest Labs. FLUX.2: Analyzing and enhancing the latent space of FLUX—representation comparison. Technical report, November 2025b. URL https://bfl.ai/research/representation-comparison.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. FLUX.1 Kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025. URL https://arxiv.org/abs/2506.15742.

Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. In *International Conference on Learning Representations*, 2023.

Haiwen Diao, Jiahao Wang, Chenjing Ding, Hanming Deng, Jiangnan Chen, Ruixi Zhang, Ruohui Wang, Wenwen Tong, Xiangyu Fan, Yubo Wang, et al. Sensenova-u1. 5: Towards native unified visual intelligence. *arXiv preprint arXiv:2609.11929*, 2026a.

Haiwen Diao, Penghao Wu, Hanming Deng, Jiahao Wang, Shihao Bai, Silei Wu, Weichen Fan, Wenjie Ye, Wenwen Tong, Xiangyu Fan, et al. Sensenova-u1: Unifying multimodal understanding and generation with neo-unify architecture, 2026b. URL https://arxiv.org/abs/2605.12500.

Sander Dieleman. Generative modelling in latent space. Technical blog post, April 2025. URL https://sander.ai/2025/04/15/latents.html.

dx8152. Flux2-Klein-9B-Consistency. Hugging Face model repository, 2026. URL https://huggingface.co/dx8152/Flux2-Klein-9B-Consistency. Version 2, released April 17, 2026. Accessed September 21, 2026.

Google. Image editing in Gemini gets a major upgrade. Google product announcement, August 2025. URL https://blog.google/products-and-platforms/products/gemini/updated-image-editing-model/.

Google. Nano Banana 2: Combining pro capabilities with lightning-fast speed. Google product announcement, February 2026. URL https://blog.google/innovation-and-ai/technology/ai/nano-banana-2/.

Jiahui Huang, Yasi Zhang, Tianyu Chen, Shu Wang, Jianwen Xie, Oscar Leong, Mingyuan Zhou, Nanzhu Wang, and Ying Nian Wu. MT-EditFlow: Reinforcement learning for multi-turn image editing with flow matching. *arXiv preprint arXiv:2606.01985*, 2026. URL https://arxiv.org/abs/2606.01985.

Daneul Kim, Jaeah Lee, and Jaesik Park. Improving editability in image generation with layer-wise memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7889–7898, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Kim_Improving_Editability_in_Image_Generation_with_Layer-wise_Memory_CVPR_2025_paper.html.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The Open Images dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*, 128:1956–1981, 2020. doi: 10.1007/s11263-020-01316-z. URL https://doi.org/10.1007/s11263-020-01316-z.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *International Conference on Learning Representations*, 2026.

Mingxiao Li, Tingyu Qu, Ruicong Yao, Wei Sun, and Marie-Francine Moens. Alleviating exposure bias in diffusion models through sampling with shifted time steps. In *International Conference on Learning Representations*, 2024.

Yucheng Liao, Jiajun Liang, Kaiqian Cui, Baoquan Zhao, Haoran Xie, Wei Liu, Qing Li, and Xudong Mao. FreqEdit: Preserving high-frequency features for robust multi-turn image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 43525–43535, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Liao_FreqEdit_Preserving_High-Frequency_Features_for_Robust_Multi-Turn_Image_Editing_CVPR_2026_paper.html.

Shiyu Liu, Yucheng Han, Peng Xing, Fukun Yin, Rui Wang, Wei Cheng, Jiaqi Liao, Yingming Wang, Honghao Fu, Chunrui Han, Guopeng Li, Yuang Peng, Quan Sun, Jingwei Wu, Yan Cai, Zheng Ge, Ranchen Ming, Lei Xia, Xianfang Zeng, Yibo Zhu, Binxing Jiao, Xiangyu Zhang, Gang Yu, and Daxin Jiang. Step1x-edit: A practical framework for general image editing. *arXiv preprint arXiv:2504.17761*, 2025.

Meta. Introducing Muse Image and Muse Video. Meta AI research announcement, July 2026. URL https://ai.meta.com/blog/introducing-muse-image-muse-video-msl/.

Mang Ning, Enver Sangineto, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Input perturbation reduces exposure bias in diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 26245–26265. PMLR, 2023.

Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. Elucidating the exposure bias in diffusion models. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2308.15321.

OpenAI. Introducing our latest image generation model in the API. OpenAI product release, April 2025a. URL https://openai.com/index/image-generation-api/.

OpenAI. The new ChatGPT Images is here. OpenAI product release, December 2025b. URL https://openai.com/index/new-chatgpt-images-is-here/.

OpenAI. Introducing chatgpt images 2.5. https://openai.com/index/introducing-chatgpt-images-2-5/, 2026a. Accessed: 2026-09-26.

OpenAI. Introducing ChatGPT Images 2.5. OpenAI product release, September 2026b. URL https://openai.com/index/introducing-chatgpt-images-2-5/.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8911–8920, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Qian_Boosting_Diffusion_Models_with_Moving_Average_Sampling_in_Frequency_Domain_CVPR_2024_paper.html.

Leigang Qu, Feng Cheng, Ziyan Yang, Qi Zhao, Shanchuan Lin, Yichun Shi, Yicong Li, Wenjie Wang, Tat-Seng Chua, and Lu Jiang. VINCIE: Unlocking in-context image editing from video. In *International Conference on Learning Representations*, 2026.

Reve. Launching Reve 2.1. Reve product announcement, July 2026. URL https://blog.reve.com/posts/launching-reve-2.1/.

Xiaoce Wang, Sifan Zhou, Kaifei Wang, Leli Xu, Xuerui Qiu, Tao He, and Ming Li. Why do DiT editors drift? plug-and-play low frequency alignment in VAE latent space. *arXiv preprint arXiv:2605.08250*, 2026. URL https://arxiv.org/abs/2605.08250.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15703–15712, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Yao_Reconstruction_vs._Generation_Taming_Optimization_Dilemma_in_Latent_Diffusion_Models_CVPR_2025_paper.html.

Yang Ye, Xianyi He, Zongjian Li, Bin Lin, Shenghai Yuan, Zhiyuan Yan, Bohan Hou, and Li Yuan. Imgedit: A unified image editing dataset and benchmark. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Xiangyu Zhao, Peiyuan Zhang, Kexian Tang, Xiaorong Zhu, Hao Li, Wenhao Chai, Zicheng Zhang, Renqiu Xia, Guangtao Zhai, Junchi Yan, Hua Yang, Xue Yang, and Haodong Duan. Envisioning beyond the pixels: Benchmarking reasoning-informed visual editing. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Zijun Zhou, Yingying Deng, Xiangyu He, Weiming Dong, and Fan Tang. Multi-turn consistent image editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15792–15801, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Zhou_Multi-turn_Consistent_Image_Editing_ICCV_2025_paper.html.

Tianrui Zhu, Shiyi Zhang, Jiawei Shao, and Yansong Tang. KV-Edit: Training-free image editing for precise background preservation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16607–16617, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Zhu_KV-Edit_Training-Free_Image_Editing_for_Precise_Background_Preservation_ICCV_2025_paper.html.