SpotEdit: Selective Region Editing in Diffusion Transformers

Supplementary Material

A. Pseudocode of SpotEdit

Algorithm S1: SpotEdit: Selective Region Editing with Diffusion Transformers

Input : Diffusion Transformer Φ, Editing Instruction P, Condition Image Latent Y, Initial Noise X_T, Total Steps T, Initial Stage Steps K_{init}, Threshold τ ,Time Schedule $\{t_i\}_{i=0}^{T}$ where $t_T = 1, t_0 = 0$

Output: Edited Image Img

Initialize Condition Cache (K_Y, V_Y);
Initialize Feature Cache $(K_{cache}, V_{cache}) \leftarrow \emptyset$;
for $i \leftarrow T, T-1, \ldots, T-K_{init}+1$ **do**
  $v_{t_i}, (K_{curr}, V_{curr}) \leftarrow \Phi(X_{t_i}, t_i, P, Y)$;
  $X_{t_{i-1}} \leftarrow X_{t_i} - (t_i - (t_{i-1})) \cdot v_{t_i}$;
  $\hat{X}_0^{t_i} \leftarrow X_{t_i} - t_i \cdot v_{t_i}$;
  $(K_{cache}, V_{cache}) \leftarrow (K_{curr}, V_{curr})$;
end
for $i \leftarrow T - K_{init}, \ldots, 1$ **do**
  $[\mathcal{A}_{t_i}, \mathcal{R}_{t_i}] \leftarrow$ **SpotSelector**$(\hat{X}_0^{t_i}, Y, \tau)$;
  for *each transformer block* b **do**
    $K^{(b)}_{\mathcal{R}_{t_i}} \leftarrow \alpha(t_i) K^{(b)}_{\mathcal{R}_{t_{i+1}}} + (1-\alpha(t_i)) K^{(b)}_Y$;
    $V^{(b)}_{\mathcal{R}_{t_i}} \leftarrow \alpha(t_i) V^{(b)}_{\mathcal{R}_{t_{i+1}}} + (1-\alpha(t_i)) V^{(b)}_Y$;
  end
  Construct Queries Q_{active} for tokens $\in \mathcal{A}_{t_i}$
  $Q_{\text{active}} \leftarrow [Q_P, Q_{\mathcal{A}_{t_i}}]$
  Construct Keys and Values
  $K_{\text{full}} \leftarrow [K_P, K_{\mathcal{A}_{t_i}}, K_{\mathcal{R}_{t_i}}, K_Y]$
  $V_{\text{full}} \leftarrow [V_P, V_{\mathcal{A}_{t_i}}, V_{\mathcal{R}_{t_i}}, V_Y]$;
  $v_{t_i}[\mathcal{A}_{t_i}] \leftarrow \text{Attention}(Q_{\text{active}}, K_{\text{full}}, V_{\text{full}})$;
  $X_{t_{i-1}}[\mathcal{A}_{t_i}] \leftarrow X_t[\mathcal{A}_{t_i}] - (t_i - t_{i-1}) \cdot v_{t_i}[\mathcal{A}_{t_i}]$;
  $\hat{X}_0^{t_i} \leftarrow X_{t_i} - t_i \cdot v_{t_i}$
end
Identify final non-edited regions

$$\mathcal{R}_{final}$$

$X_0^{final}[\mathcal{A}_{final}] \leftarrow X_0[\mathcal{A}_{final}]$;
$X_0^{final}[\mathcal{R}_{final}] \leftarrow Y[\mathcal{R}_{final}]$;
$Img \leftarrow \text{VAE}(X_0^{final})$;
return Img

Algorithm S2: SpotSelector: LPIPS-like Perceptual Scoring

Input : Reconstructed latent $\hat{X}_0 \in \mathbb{R}^{N \times C}$, Conditional image latent $Y \in \mathbb{R}^{N \times C}$, VAE Decoder shallow layers $\mathcal{L}$, Spatial dimensions (H, W), Patch size p

Output: Regenerate region and non-edited region indices $[\mathcal{A}_{t_i}, \mathcal{R}_{t_i}]$

$F_{\hat{x}} \leftarrow \{\phi_l(\hat{x}_{input}) \mid l \in \mathcal{L}\}$;
$F_y \leftarrow \{\phi_l(y_{input}) \mid l \in \mathcal{L}\}$;
Initialize spatial score map $M \leftarrow \mathbf{0}$;
for *each layer* $l \in \mathcal{L}$ **do**
  $D_l \leftarrow \|\text{Norm}(F_{\hat{x}}^{(l)}) - \text{Norm}(F_y^{(l)})\|_2^2$;
  $D_l^{aligned} \leftarrow \text{Resize}(D_l, \text{size} \leftarrow (H, W))$;
  $M \leftarrow M + D_l^{aligned}$;
end
$M \leftarrow M/|\mathcal{L}|$
$S_{pooled} \leftarrow \text{AvgPool}(M, \text{kernel}, \text{stride})$;
$S_{token} \leftarrow \text{Flatten}(S_{pooled})$;
$[\mathcal{A}_{t_i}, \mathcal{R}_{t_i}] \leftarrow S_{token} < \tau$
return $[\mathcal{A}_{t_i}, \mathcal{R}_{t_i}]$

B. Compatibility with Existing Acceleration Methods

A key advantage of **SpotEdit** is that it is **orthogonal** to existing acceleration techniques for Diffusion Transformers (DiTs). While prior methods accelerate along the *temporal*, *feature*, or *attention* dimensions, SpotEdit accelerates computation along the *spatial* by skipping non-edited regions. Importantly, the acceleration dimensions targeted by these methods are inherently complementary to the spatial acceleration pursued by SpotEdit. Rather than competing with SpotEdit's region spotting and token skipping mechanism, they operate along orthogonal axes, enabling their effects to be combined additively for further speed improvements.

B.1. General compatibility with full-token computation accelerators

Let the set of regenerated tokens selected by SpotSelector be $\mathcal{A}$ and non-edited tokens be $\mathcal{R}$.

SpotFusion reconstructs (K, V) values for computation of all regenerated tokens $\mathcal{A}$:

$$K_{\text{full}} = [K_P, K_{\mathcal{A}}, K_{\mathcal{R}}, K_Y] V_{\text{full}} = [V_P, V_{\mathcal{A}}, V_{\mathcal{R}}, V_Y]$$

This reconstruction ensures that the edited region forms

a **closed and condition-complete subgraph** of the DiT model. Thus, any acceleration operator $\mathcal{O}_{acc}$ that assumes full attention context may be applied to the edited region alone:

$$\mathcal{O}_{acc}(X_{\mathcal{A}}) \oplus X_{\mathcal{R}}^{\text{cache}},$$

where $\oplus$ denotes spatial concatenation.

Since SpotEdit does not change the functional form of the DiT computation and only restricts its spatial domain, it is fully compatible with other temporal accelerators.

B.2. Compatibility with TeaCache and TaylorSeer

To further demonstrate that SpotEdit is orthogonal to temporal accelerators, we integrate **TeaCache** and **TaylorSeer** into the SpotEdit pipeline. In both cases, the edited region subgraph produced by SpotSelector and reconstructed by SpotFusion forms a self-contained full-token region, ensuring that existing reuse-based accelerators operate without modification.

Formally, for any accelerator $\mathcal{O}_{acc}$ applied on the edited tokens, the composite update takes the unified form:

$$\mathcal{F}_{\text{SpotEdit}+\mathcal{O}_{acc}}(X) \leftarrow \mathcal{O}_{acc}\big(X_{\mathcal{A}}\big) \oplus X_{\mathcal{R}}^{\text{cache}},$$

where the cached non-edited tokens remain valid due to SpotFusion's full reconstruction of (K, V).

TeaCache TeaCache performs timestep feature reuse through caching. Since SpotFusion regenerates complete attention states for edited tokens, TeaCache cached residuals remain fully compatible and can be directly reused inside the edited subgraph.

TaylorSeer TaylorSeer approximates residuals via local Taylor-series predictions. Because the edited subgraph satisfies the same continuous-time latent dynamics, the Taylor approximation computed on $X_{\mathcal{A}}$ remains valid, while non-edited tokens continue using cached features.

B.3. Experimental results

Table S1 and Table S2 report speed and quality metrics for SpotEdit combined with TeaCache and TaylorSeer. Both integrations remain stable and improve efficiency while preserving editing quality.

These results empirically confirm TeaCache and TaylorSeer integrate seamlessly into SpotEdit.

Baselines	CLIP↑	SSIMc↑	PSNR↑	DISTS↓	Speedup↑	Latency (s)↓
Original Inference	0.699	0.67	16.40	0.17	1.00×	27.10
TeaCache[20]	**0.698** (↓0.001)	0.60 (↓0.07)	15.02 (↓1.38)	0.21 (↑0.04)	3.43× (↑2.43)	7.90
TaylorSeer[21]	0.666 (↓0.033)	0.52 (↓0.15)	14.36 (↓2.04)	0.37 (↑0.20)	3.61× (↑2.61)	7.51
SpotEdit–TeaCache	0.695 (↓0.004)	**0.62** (↓0.05)	**15.57** (↓0.83)	**0.19** (↑0.02)	**3.94**× (↑2.94)	**6.88**
SpotEdit–TaylorSeer	0.698 (↓0.001)	0.61 (↓0.06)	15.50 (↓0.90)	0.19 (↑0.02)	3.85× (↑2.85)	7.04

Table S1. Comparison of models on **imgEdit-Benchmark**.

Baselines	CLIP↑	SSIMc↑	PSNR↑	DISTS↓	Speedup↑	Latency (s)↓
Original Inference	0.741	0.791	18.76	0.136	1.00×	27.10
TeaCache[20]	0.735 (↓0.006)	0.764 (↓0.027)	18.89 (↑0.13)	0.144 (↑0.008)	3.59× (↑2.59)	7.55
TaylorSeer[21]	0.741 (0.00)	0.755 (↓0.036)	17.81 (↓0.95)	0.159 (↑0.023)	3.86× (↑2.86)	7.02
SpotEdit–TeaCache	0.740 (↓0.0005)	**0.797** (↑0.006)	**18.98** (↑0.22)	**0.133** (↓0.003)	**4.28**× (↑3.28)	**6.33**
SpotEdit–TaylorSeer	**0.743** (↑0.002)	0.783 (↓0.008)	18.50 (↓0.26)	0.142 (↑0.006)	4.16× (↑3.16)	6.51

Table S2. Comparison of models on **PIE-Bench++**.

C. Discussion between ℓ_2-distance and LPIPS-like score

We further justify the use of the LPIPS-like score in **SpotSelector** by analyzing the spectral bias inherent to different similarity metrics.

Figure S1. ℓ_2 **Distance vs. LPIPS-like Score.** Low-frequency changes (e.g., brightness) produce overly large ℓ_2 responses, while subtle high-frequency texture edits barely affect it. As shown in the first sample, ℓ_2 incorrectly preserves the spaceship that should have been removed; in the second sample, it misclassifies background tokens as regenerate tokens, causing unnecessary regeneration and background degradation. LPIPS-like features avoid these failures by operating in a perceptually aligned feature space. For a fair comparison, the threshold τ is set to 0.2 for both methods.

Limitations of latent ℓ_2 distance. Latent representations in diffusion models are highly compressed. A pointwise ℓ_2 distance is dominated by low-frequency components such as global brightness and color statistics. As noted by Zhang et al. [54], pixel-wise metrics assume independence across dimensions and thus fail to capture structural degradations like blur, which primarily remove high-frequency content but induce only mild ℓ_2 deviations. In selective editing, this bias produces two failure modes: (1) low-frequency shifts, such as brightness changes, disproportionately inflate ℓ_2 and falsely mark a region as 'regenerated', and (2) high-frequency feature changes with similar low-frequency features remain undetected, causing truly edited regions to be misclassified as 'non-edited'.

Perceptual transferability via the VAE decoder. Although the observations of Zhang et al. were established in the pixel domain, their core principle, deep features reflect perceptual similarity better than raw vectors extends natu-

rally to latent diffusion models. The VAE decoder provides a non-linear mapping from compressed latent space back to perceptual image manifolds. Its intermediate activation maps recover spatial structure and high-frequency cues that are not explicitly represented in raw latents.

Building on this insight, our LPIPS-like score measures distances between decoder features, analogous to evaluating perceptual differences through VGG features in standard LPIPS. This grants two advantages:

It captures high-frequency discrepancies essential for determining whether a region was genuinely edited. And it ensures alignment between a non-edited region $\mathcal{R}_t$ and the condition image Y not only in coarse color tone but also in fine-grained spatial patterns.

By leveraging decoder deep features, LPIPS-like score provides a perceptually faithful descriptor for region stability and editing consistency. This resolves the spectral bias of ℓ_2 and enables robust token selection in SpotSelector.

D. User Study

To further evaluate the perceptual quality of our method, we conducted a User Study with 33 participants, as shown in Table S3. SpotEdit consistently outperforms baselines across Instruction Alignment, Image Preservation, Perceptual Quality, and Overall preference.

Method	Instr. Align.	Image Pres.	PQ	Overall
TeaCache	0.85	0.61	0.71	0.66
TaylorSeer	0.81	0.61	0.61	0.59
FollowYourShape	0.58	0.19	0.59	0.42
SpotEdit (Ours)	**0.91**	**0.89**	**0.91**	**0.89**

Table S3. Results of User Study. Scores indicate user preference across different models.

E. Sensitivity Analysis

To provide a deeper understanding of the SpotSelector mechanism, we visualize the editing outcomes under varying threshold values τ in Figure S2. The results illustrate a clear trade-off: setting τ below 0.15 makes the model overly sensitive to negligible feature variations, causing fragmented region masks and suboptimal preservation. Conversely, values above 0.25 impose excessively strict criteria, inadvertently excluding essential editing context and disrupting the generated content. Our chosen $\tau = 0.2$ consistently avoids these extremes, ensuring precise semantic localization and high fidelity across diverse editing scenarios.

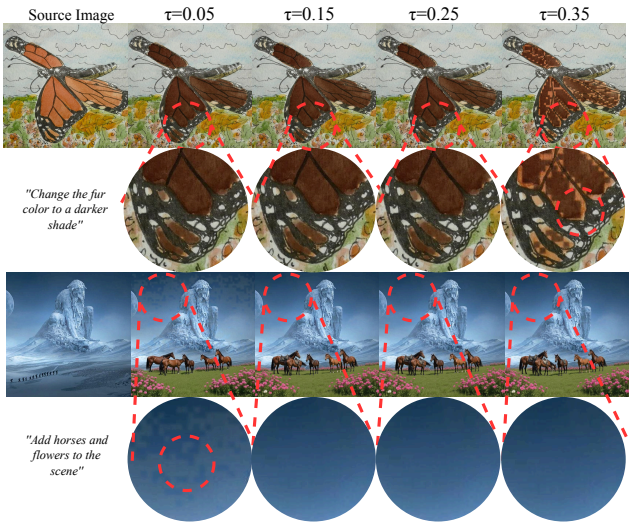

Figure S2. Sensitivity analysis of the SpotSelector threshold τ.

F. Experimental comparison with RegionE

To ensure a fair comparison with the concurrent work RegionE [5], we evaluate both methods under a 28-step sampling configuration, aligning directly with RegionE experimental settings. As demonstrated in Table S4 and Table S5, our approach consistently achieves better preservation of the original image region. Specifically, our method yields higher structural similarity (SSIMc), higher peak signal-to-noise ratio (PSNR), and lower perceptual differences (DISTS) across both the PIE-Bench++ and imgEdit benchmarks, indicating the better preservation of the original image regions.

Method	CLIPScore ↑	SSIMc ↑	PSNR ↑	DISTS ↓
Kontext (Original)	0.7406	0.7821	18.8383	0.1331
RegionE [5]	0.7395	0.7860	18.8759	0.1332
Ours	**0.7398**	**0.7890**	**19.0408**	**0.1298**

Table S4. Comparison with RegionE on **PIE-Bench++**.

Method	CLIPScore ↑	SSIMc ↑	PSNR ↑	DISTS ↓
Kontext (Original)	0.6974	0.6731	16.7197	0.1586
RegionE [5]	**0.6958**	0.6761	16.7530	0.1580
Ours	0.6946	**0.6805**	**17.0020**	**0.1551**

Table S5. Comparison with RegionE on **imgEdit-Benchmark**.

G. Limitations

While SpotEdit excels at local editing and strict background preservation, it is specifically tailored for localized edits

where large portions of the image are meant to be kept identical. Consequently, it is unsuitable for global editing tasks (e.g., global style transfer) that inherently require widespread feature changes across the entire image.

H. More Visualization Results

As shown in Fig. S4, we provide additional qualitative results of SpotEdit across a wide range of instruction-based editing tasks, including *add*, *action*, *adjust*, *background*, *remove*, *replace*, and *extract*. These examples further demonstrate the versatility and robustness of our framework in handling diverse semantic manipulations while maintaining high fidelity in non-edited regions.

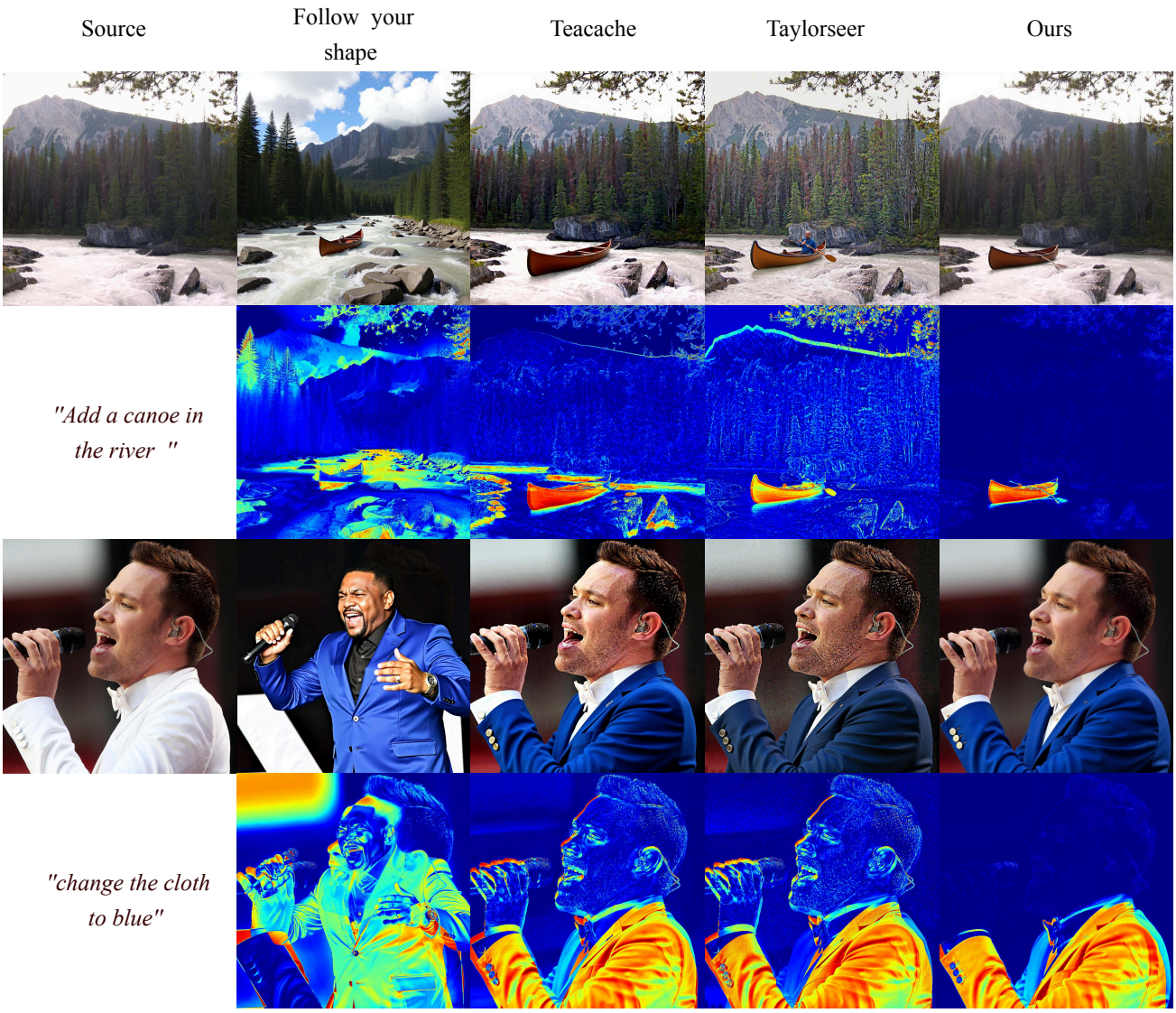

Figure S3. Additional qualitative comparisons and residual heatmaps against baselines.

Figure S4. More results of SpotEdit with various editing tasks