Appendix of SMART-Edit: Spectral Mutual Attention Refinement and Fusion for Precise Training-Free Image Editing

Appendix A. Algorithm Overview

In this section, we present the detailed algorithmic procedures of SMART-Edit for both inversion-based and inversion-free editing frameworks. The complete pipelines are summarized in Algorithm 1 and Algorithm 2, respectively.

Appendix B. Implementation Details

Attention Extraction Details. Taking the most frequent 512×512 resolution in our datasets as an example, the joint attention matrix $\mathcal{A}_{\text{joint}}$ comprises 1,024 vision tokens alongside a model-dependent number of text tokens. To perform precise semantic localization, we slice $\mathcal{A}_{\text{joint}}$ into four distinct submatrices: text-to-text self-attention ($\mathcal{A}^{T\to T}$), vision-to-text cross-attention ($\mathcal{A}^{V\to T}$), text-to-vision cross-attention ($\mathcal{A}^{T\to V}$), and vision-to-vision self-attention ($\mathcal{A}^{V\to V}$). The dimensionality of the extracted attention maps is determined by the token layout and text token configuration of each model architecture:

- **Flux.1 Dev**: Employs a single T5 encoder producing $T = 512$ text tokens, which are concatenated with vision tokens to form a joint sequence, resulting in a joint matrix shape of $(1536, 1536)$.
- **Stable Diffusion 3/3.5 (SD3/3.5)**: Employs three text encoders with a total of $T = 333$ text tokens, consisting of 77 concatenated CLIP tokens and 256 T5 tokens. These tokens are positioned after the vision tokens, yielding a joint attention matrix of shape $(1357, 1357)$. Consequently, the text-related attention region is partitioned into independent CLIP and T5 sub-blocks. Following the token selection strategy described in Section 4.1, the extracted CLIP and T5 attention submatrices have matching dimensions and are directly averaged for mask generation. The remaining operations are identical to those of Flux.1 Dev.

Algorithm 1 SMART-Edit Pipeline for Inversion-based Editing Frameworks

Require: Source image I_{src}, Prompts $P_{\text{src}}, P_{\text{tar}}$, Altered token indices $\mathcal{I}$, Total steps T, Injection threshold $\tau_{\text{inj}} \in [0, 1]$

Ensure: Edited target latent z_0

Note: For clarity, the time step t is explicitly denoted as a subscript for variables throughout this algorithm, whereas it is omitted in the previous sections for mathematical conciseness.

$\Delta t \leftarrow \frac{1}{T}$

 # Phase 1: Forward Inversion Stage

for $t \in \{0, \Delta t, \ldots, 1 - \Delta t\}$ **do**

  $v_t, \mathcal{A}_t^{V\to T}, \mathcal{A}_t^{T\to V}, \mathcal{A}_t^{V\to V}, \mathcal{A}_t^{T\to T}, V_{\text{src},t}^{\text{img}} \leftarrow v_\theta(z_t, t, P_{\text{src}})$

  $z_{t+\Delta t} \leftarrow z_t + v_t \cdot \Delta t$

  if $t \geq \tau_{\text{inj}}$ **then**

    $\tilde{\mathcal{A}}_t^{V\to T}, \tilde{\mathcal{A}}_t^{T\to V} \leftarrow \text{DAR}(\mathcal{A}_t^{V\to T}, \mathcal{A}_t^{T\to V}, \mathcal{A}_t^{V\to V}, \mathcal{A}_t^{T\to T}, \mathcal{I})$

    $\tilde{\mathcal{A}}_t^{\text{fused}} \leftarrow \text{SMAF}(\tilde{\mathcal{A}}_t^{V\to T}, \tilde{\mathcal{A}}_t^{T\to V})$

    $\mathbf{M}_t^S \leftarrow \text{SMT}(\tilde{\mathcal{A}}_t^{\text{fused}})$

    Cache $V_{\text{src},t}^{\text{img}}$ and $\mathbf{M}_t^S$

  end if

end for

 # Phase 2: Backward Denoising Stage

for $t \in \{1, 1 - \Delta t, \ldots, \Delta t\}$ **do**

  if $t \geq \tau_{\text{inj}}$ **then**

    Retrieve cached $V_{\text{src},t}^{\text{img}}$ and $\mathbf{M}_t^S$

    $v_{\text{tar},t}, V_{\text{tar},t}^{\text{img}} \leftarrow v_\theta(z_t, t, P_{\text{tar}})$

    $\hat{V}_t^{\text{img}} \leftarrow \mathbf{M}_t^S \odot V_{\text{tar},t}^{\text{img}} + (\mathbf{1} - \mathbf{M}_t^S) \odot V_{\text{src},t}^{\text{img}}$

    $\hat{v}_t \leftarrow v_\theta(z_t, t, P_{\text{tar}} \mid \hat{V}_t^{\text{img}})$ ▷ Velocity prediction with feature injection

  ▷ *Note: Other manipulations in the inversion-based image editing framework can be seamlessly integrated here!*

  else

    $\hat{v}_t \leftarrow v_\theta(z_t, t, P_{\text{tar}})$

  end if

  $z_{t-\Delta t} \leftarrow z_t - \hat{v}_t \cdot \Delta t$

end for

return z_0

Algorithm 2 SMART-Edit Pipeline for Inversion-free Editing Frameworks

Require: Source image I_{src}, Prompts $P_{\text{src}}, P_{\text{tar}}$, Altered token indices $\mathcal{I}$, Total steps T, Injection threshold $\tau_{\text{inj}} \in [0, 1]$

Ensure: Edited target latent z_0

Note: For clarity, the time step t is explicitly denoted as a subscript for variables throughout this algorithm, whereas it is omitted in the previous sections for mathematical conciseness.

$\Delta t \leftarrow \frac{1}{T}$

$z_1^{\text{FE}} \leftarrow I_{\text{src}}$

for $t \in \{1, 1 - \Delta t, \ldots, \Delta t\}$ **do**

  $\epsilon_t \sim \mathcal{N}(0, \mathbf{I})$

  $z_{\text{src},t} \leftarrow (1 - t)I_{\text{src}} + t\epsilon_t$

  $z_{\text{tar},t} \leftarrow z_t^{\text{FE}} + z_{\text{src},t} - I_{\text{src}}$

  if $t \geq \tau_{\text{inj}}$ **then**

    $v_{\text{src},t}, \mathcal{A}_t^{V\to T}, \mathcal{A}_t^{T\to V}, \mathcal{A}_t^{V\to V}, \mathcal{A}_t^{T\to T}, V_{\text{src},t}^{\text{img}} \leftarrow v_\theta(z_{\text{src},t}, t, P_{\text{src}})$

    $v_{\text{tar},t}, V_{\text{tar},t}^{\text{img}} \leftarrow v_\theta(z_{\text{tar},t}, t, P_{\text{tar}})$

    $\tilde{\mathcal{A}}_t^{V\to T}, \tilde{\mathcal{A}}_t^{T\to V} \leftarrow \text{DAR}(\mathcal{A}_t^{V\to T}, \mathcal{A}_t^{T\to V}, \mathcal{A}_t^{V\to V}, \mathcal{A}_t^{T\to T}, \mathcal{I})$

    $\tilde{\mathcal{A}}_t^{\text{fused}} \leftarrow \text{SMAF}(\tilde{\mathcal{A}}_t^{V\to T}, \tilde{\mathcal{A}}_t^{T\to V})$

    $\mathbf{M}_t^S \leftarrow \text{SMT}(\tilde{\mathcal{A}}_t^{\text{fused}})$

    $\hat{V}_t^{\text{img}} \leftarrow \mathbf{M}_t^S \odot V_{\text{tar},t}^{\text{img}} + (\mathbf{1} - \mathbf{M}_t^S) \odot V_{\text{src},t}^{\text{img}}$

    $\hat{v}_{\text{tar},t} \leftarrow v_\theta(z_{\text{tar},t}, t, P_{\text{tar}} \mid \hat{V}_t^{\text{img}})$ ▷ Velocity prediction with feature injection

  else

    $v_{\text{src},t} \leftarrow v_\theta(z_{\text{src},t}, t, P_{\text{src}})$

    $v_{\text{tar},t} \leftarrow v_\theta(z_{\text{tar},t}, t, P_{\text{tar}})$

    $\hat{v}_{\text{tar},t} \leftarrow v_{\text{tar},t}$

  end if

  ▷ *Note: Other manipulations in inversion-free image editing frameworks can be seamlessly integrated here!*

  $v_t^\Delta \leftarrow \hat{v}_{\text{tar},t} - v_{\text{src},t}$

  $z_{t-\Delta t}^{\text{FE}} \leftarrow z_t^{\text{FE}} - v_t^\Delta \cdot \Delta t$

end for

return z_0^{FE}

The detailed configurations regarding the attention types and their respective dimensions are summarized in Table B.1.

Table B.1 Attention configurations. Attention matrix slicing configurations and submatrix dimensions for Flux.1 Dev and Stable Diffusion 3/3.5 (SD3/3.5) at a standardized 512×512 resolution.

Type	Model	Slicing Logic `[Rows, Cols]`	Resulting Shape
$\mathcal{A}^{V\to V}$	Flux.1 Dev	`[512:, 512:]`	(1024, 1024)
	SD3/3.5	`[:1024, :1024]`	(1024, 1024)
$\mathcal{A}^{V\to T}$	Flux.1 Dev	`[512:, :512]`	(1024, 512)
	SD3/3.5 (CLIP)	`[:1024, 1024:1101]`	(1024, 77)
	SD3/3.5 (T5)	`[:1024, 1101:]`	(1024, 256)
$\mathcal{A}^{T\to V}$	Flux.1 Dev	`[:512, 512:]`	(512, 1024)
	SD3/3.5 (CLIP)	`[1024:1101, :1024]`	(77, 1024)
	SD3/3.5 (T5)	`[1101:, :1024]`	(256, 1024)
$\mathcal{A}^{T\to T}$	Flux.1 Dev	`[:512, :512]`	(512, 512)
	SD3/3.5 (CLIP)	`[1024:1101, 1024:1101]`	(77, 77)
	SD3/3.5 (T5)	`[1101:, 1101:]`	(256, 256)

Appendix C. Additional Ablation Studies

We further conduct ablation studies on FlowEdit and RF-Solver to evaluate the effectiveness of each component in SMART-Edit across different training-free editing methods. Specifically, the quantitative and qualitative ablation results on FlowEdit are presented in Table C.2 and Figure C.1, respectively, while those on RF-Solver are shown in Table C.3 and Figure C.2. Overall, the results align with the observations in the main paper, where removing individual components generally leads to performance degradation. The two cross-modal attention branches provide complementary localization cues, with $\mathcal{A}^{V\to T}$ playing a more critical role in identifying target regions and $\mathcal{A}^{T\to V}$ providing additional semantic guidance for disambiguating the editing area. DAR refines the initial cross-modal attention maps through self-attention, effectively

suppressing noisy and irrelevant activations while enhancing the spatial coherence and structural consistency of the localization responses. SMAF reconciles the complementary semantic and boundary information encoded by the two attention branches through frequency-aware decomposition and sub-band fusion, leading to more robust and precise localization than direct weighted fusion. Centroid initialization establishes a reliable semantic anchor based on the attention distribution, providing a stable starting point for subsequent region expansion and reducing the risk of inaccurate or fragmented localization. SMT further propagates the localized region from the semantic seed while introducing a soft transition between editing and non-editing areas, thereby improving mask continuity, boundary smoothness, and the visual naturalness of the resulting edits. In summary, these ablation results demonstrate that each component contributes a distinct yet complementary function to the localization pipeline, and their effectiveness across different backbones and training-free editing methods further highlights the generalization of SMART-Edit.

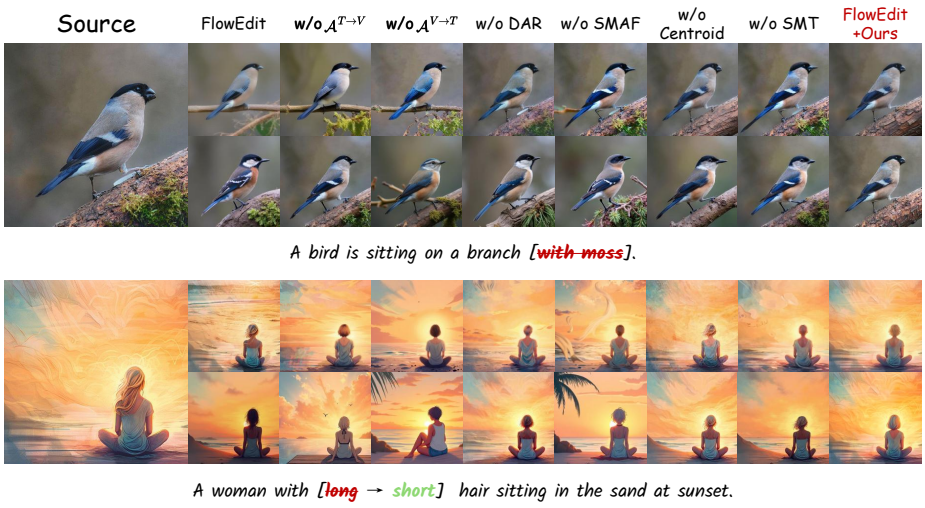


Figure C.1: **Qualitative comparison of ablation variants on FlowEdit.** For each case, results generated by SD3 and Flux.1 Dev are shown in the first and second rows, respectively. Please zoom in for a better view.

Table C.2 Ablation study of SMART-Edit on FlowEdit. We report background preservation, editing fidelity, and aesthetic quality metrics on the PIE-Bench dataset.

Model	Method Variants	Distance$^{\downarrow}_{\times 10^3}$	PSNR↑	Edited↑	AES↑	ImageReward↑
FLUX	FlowEdit$_{+\text{Ours}}$	**24.89**	**23.82**	**22.85**	**5.874**	0.834
	w/o $\mathcal{A}^{T\to V}$	25.96	23.18	22.49	5.801	0.819
	w/o $\mathcal{A}^{V\to T}$	26.13	22.96	22.43	5.743	0.787
	w/o DAR	25.12	23.63	22.76	5.865	0.792
	w/o SMAF	28.07	21.14	22.67	5.670	0.764
	w/o Centroid	25.68	23.51	22.78	5.838	**0.842**
	w/o SMT	25.11	23.67	22.62	5.616	0.755
	FlowEdit	27.82	21.96	22.58	5.527	0.731
SD3	FlowEdit$_{+\text{Ours}}$	**20.17**	**26.74**	**24.32**	**5.841**	**0.831**
	w/o $\mathcal{A}^{T\to V}$	23.26	26.12	24.01	5.776	0.812
	w/o $\mathcal{A}^{V\to T}$	24.73	25.87	24.18	5.721	0.797
	w/o DAR	20.89	26.62	24.19	5.835	0.784
	w/o SMAF	25.36	24.18	23.71	5.603	0.676
	w/o Centroid	21.09	26.31	24.08	5.801	0.820
	w/o SMT	20.52	26.49	23.72	5.673	0.612
	FlowEdit	27.12	22.22	23.57	5.547	0.588

Appendix D. Sensitivity Analysis

We conduct a sensitivity analysis on the five key hyperparameters of SMART-Edit, namely the high-frequency enhancement factor α in spectral recomposition, the low-frequency fusion weight w_{low} in SMAF, the propagation threshold ϕ in SMT, the injection step t_s, and the injection layer L of the backbone. The quantitative results for w_{low}, α, and ϕ are presented in Figure D.3, while the corresponding qualitative results are shown in Figure D.5. The quantitative and qualitative sensitivity analyses of the injection step t_s and injection layer L are presented in Figure D.4 and Figure D.6, respectively.

Effect of w_{low}. The hyperparameter w_{low} balances the low-frequency semantic information from the two cross-modal attentions. Overall, the model exhibits noticeable sensitivity to this parameter, as it directly dictates the cross-modal structural reconcilia-

Table C.3 Ablation study of SMART-Edit on RF-Solver. We report background preservation, editing fidelity, and aesthetic quality metrics on the PIE-Bench dataset.

Model	Method Variants	Distance$^{\downarrow}_{\times 10^3}$	PSNR↑	Edited↑	AES↑	ImageReward↑
FLUX	RF-Solver$_{+\text{Ours}}$	17.74	**26.34**	**23.85**	**5.913**	**1.021**
	w/o $\mathcal{A}^{T\to V}$	19.48	25.39	23.18	5.831	0.968
	w/o $\mathcal{A}^{V\to T}$	19.76	25.34	23.41	5.779	0.837
	w/o DAR	18.92	25.56	23.44	5.809	0.885
	w/o SMAF	20.93	24.73	22.99	5.735	0.878
	w/o Centroid	18.20	26.00	23.60	5.856	0.983
	w/o SMT	**17.52**	26.11	23.04	5.690	0.710
	RF-Solver	21.79	24.83	22.26	5.651	0.692
SD3.5	RF-Solver$_{+\text{Ours}}$	**18.37**	**25.42**	**23.60**	**5.819**	**1.132**
	w/o $\mathcal{A}^{T\to V}$	19.06	24.76	23.29	5.750	1.051
	w/o $\mathcal{A}^{V\to T}$	19.30	24.58	23.17	5.703	1.023
	w/o DAR	18.83	24.86	23.31	5.781	1.087
	w/o SMAF	20.41	23.91	22.96	5.676	0.988
	w/o Centroid	18.76	25.09	23.51	5.773	1.068
	w/o SMT	18.42	25.18	22.80	5.680	0.960
	RF-Solver	20.73	21.12	22.59	5.654	1.009

tion. As illustrated in Figure D.3a, the optimal performance for both prompt adherence and background preservation is achieved at $w_{\text{low}} = 0.7$ across both the SD3.5 and Flux.1 Dev backbones. This indicates that integrating a higher proportion of $\mathcal{A}^{V\to T}$ provides stronger spatial priors, which significantly aids in accurately localizing target objects. Conversely, deviating from this optimal value degrades editing quality: excessively small values lead to poor semantic alignment and incomplete edits, whereas an overly large value over-suppresses the text-to-visual semantic injection, restricting the model's capacity to properly execute the editing instructions, as further corroborated by the qualitative results in Figure D.5a.

Effect of α. The hyperparameter α determines the contribution of high-frequency boundary details in the fused attention map, effectively balancing the high-frequency and low-frequency components during spectral recomposition. Overall, the model is

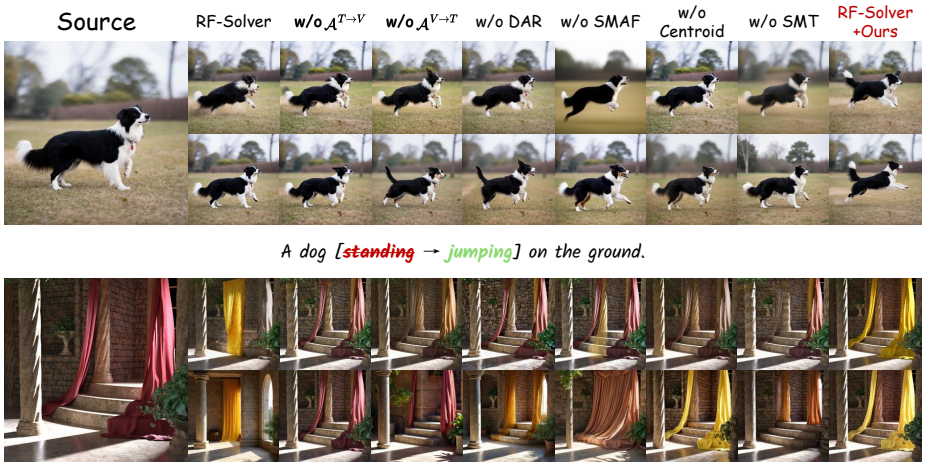
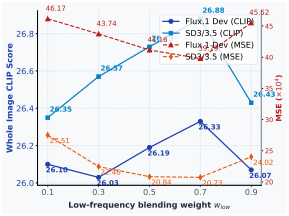


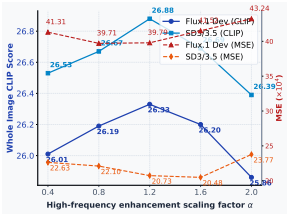
Figure C.2: **Qualitative comparison of ablation variants on RF-Solver.** For each case, results generated by SD3.5 and Flux.1 Dev are shown in the first and second rows, respectively. Please zoom in for a better view.

highly sensitive to this scaling factor, as it directly impacts the sharpness and structural integrity of the edited regions. As depicted in Figure D.3b, the optimal performance for both prompt adherence and background preservation is reached at $\alpha = 1.2$ across both the SD3.5 and Flux.1 Dev backbones. This optimal value, being greater than 1.0, implies that emphasizing high-frequency details is relatively more important for precisely localizing object boundaries and retaining fine-grained textures. Conversely, an inappropriate selection of α leads to noticeable degradation: excessively small values result in blurry boundary transitions and imprecise object localization due to insufficient high-frequency guidance, while an overly large value over-amplifies textural noise, leading to unnatural structural artifacts and compromised overall image quality, as qualitatively validated in Figure D.5b.

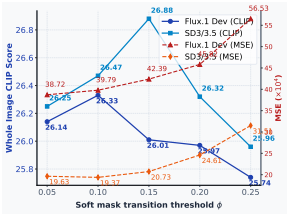
Effect of ϕ. The hyperparameter ϕ serves as the propagation threshold that controls the semantic consistency criterion during the iterative mask expansion. Overall, the model exhibits high sensitivity to this threshold, as it fundamentally dictates the spatial extent and boundary accuracy of the generated soft transition mask. As illustrated in Figure D.3c, the optimal performance for prompt adherence and background preser-



(a) Quantitative sensitivity analysis of hyperparameter w_{low} on PIE-Bench.

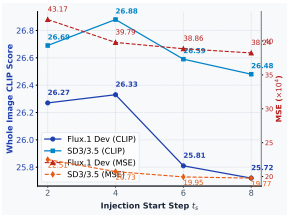


(b) Quantitative sensitivity analysis of hyperparameter α on PIE-Bench.

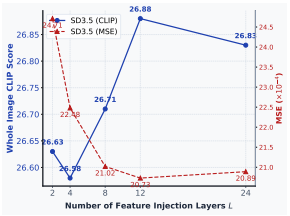


(c) Quantitative sensitivity analysis of hyperparameter ϕ on PIE-Bench.

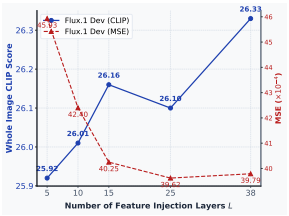
Figure D.3: **Sensitivity analysis of w_{low}, α, and ϕ on PIE-Bench.** We report the Whole Image CLIP Score and MSE.



(a) Quantitative sensitivity analysis of the injection step t_s.



(b) Quantitative sensitivity analysis of the injection layer L (SD3.5).

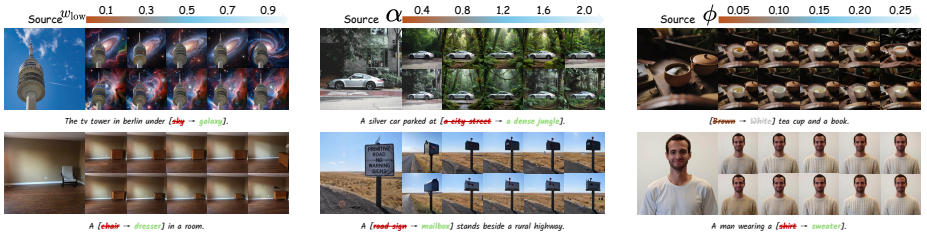


(c) Quantitative sensitivity analysis of the injection layer L (Flux.1 Dev).

Figure D.4: **Sensitivity analysis of SMART-Edit injection step t_s, and layer L on PIE-Bench.** We report the Whole Image CLIP Score and MSE.

vation varies slightly across backbones: Flux.1 Dev peaks at $\phi = 0.10$, while SD3.5 optimizes at $\phi = 0.15$. These specific values represent ideal thresholds for attention homogeneity, ensuring the mask precisely encompasses the target entity without spilling over. Conversely, setting an inappropriate threshold leads to inaccurate spatial editing: an excessively small value imposes an overly strict homogeneity criterion, prematurely halting the mask propagation and resulting in incomplete editing regions. In contrast, an overly large threshold is too permissive, causing the mask to incorrectly absorb adjacent non-target pixels, which leads to over-editing and unintended background alteration, as further verified by the qualitative results in Figure D.5c.

Effect of t_s. The hyperparameter t_s controls the number of early denoising steps for applying SMART-Edit. Since the early diffusion process plays an important role in determining the layout and coarse structural information of the generated image [1, 2],

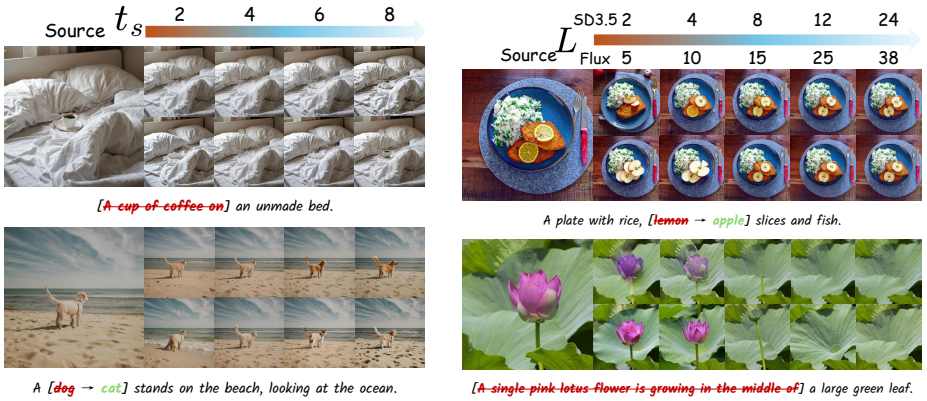


(a) Qualitative sensitivity analysis of hyperparameter w_{low}.

(b) Qualitative sensitivity analysis of hyperparameter α.

(c) Qualitative sensitivity analysis of hyperparameter ϕ.

Figure D.5: **Qualitative sensitivity analysis of w_{low}, α, and ϕ on PIE-Bench.** We analyze the sensitivity of w_{low}, α, and ϕ on SD3.5 (top row) and Flux.1 Dev (bottom row). Please zoom in for a better view.



(a) Qualitative sensitivity analysis of the injection step t_s.

(b) Qualitative sensitivity analysis of the injection layer L.

Figure D.6: **Qualitative sensitivity analysis of t_s and L on PIE-Bench.** We analyze the sensitivity of t_s and L on SD3.5 (top row) and Flux.1 Dev (bottom row). Please zoom in for a better view.

t_s directly controls the trade-off between preserving the source image structure and adapting to the target editing semantics. As shown in Figure D.4a and Figure D.6a, the choice of t_s plays a critical role in determining the overall performance. When t_s is too small (*e.g.*, $t_s = 2$), the source-image information is insufficiently preserved during the early denoising stage, leading to degraded editing consistency. On the other hand, as t_s increases, the MSE consistently decreases, indicating improved source-image preservation, but the Whole Image CLIP Score drops noticeably, suggesting that excessive injection of inversion features increasingly constrains the generation toward the source-image semantics and interferes with the target layout and semantics. Empir-

ically, $t_s = 4$ provides the best overall balance between source-structure preservation and target-semantic alignment, achieving the highest Whole Image CLIP Score for both backbones while maintaining a relatively low MSE. We therefore adopt $t_s = 4$ as the default setting.

Effect of L. The hyperparameter L controls the number of backbone layers over which SMART-Edit is applied, counting from the last layer. Existing studies [3] have shown that different layers in diffusion transformers capture different levels of information, with middle and deeper layers tending to encode richer semantic features. We therefore analyze applying SMART-Edit to the last L layers, focusing on the joint attention layers of SD3.5 and the single-stream blocks of Flux.1 Dev. As shown in Figure D.4b, Figure D.4c, and Figure D.6b, the performance exhibits a clear saturation trend as L increases. For SD3.5, the performance reaches its best trade-off at $L = 12$, indicating that controlling the last 12 layers is already sufficient to meet the structural and semantic injection requirements of SMART-Edit. In contrast, Flux.1 Dev requires a larger number of layers, with $L = 38$ achieving the best overall performance. Since SMART-Edit operates in a layer-wise manner, controlling too few layers may not sufficiently cover the layers required for effective structural and semantic feature injection. In contrast, increasing L beyond the effective range does not lead to a significant performance degradation, demonstrating that SMART-Edit is relatively robust to the choice of the injection layer. We therefore adopt $L = 12$ for SD3.5-Medium and $L = 38$ for Flux.1 Dev as the default settings.

Appendix E. Complexity Analysis

To evaluate computational efficiency, we randomly select 100 samples with a resolution of 512×512 from PIE-Bench and report the average inference time per image. The results are summarized in Table E.4. The overall runtime is largely dominated by the underlying editing framework. Integrating SMART-Edit introduces only a minor latency overhead across all evaluated methods while improving editing performance. The low computational overhead stems from the fact that SMART-Edit does not introduce additional forward passes. Instead, it relies primarily on caching intermediate attention

Table E.4 Computation efficiency comparison. The average per-sample time (s) for generating images based on 100 random prompts. The inference speed is evaluated on a single NVIDIA A100 40GB GPU.

Model	Method Variants	Time (s)$^{\downarrow}$
Flux.1 Dev	FlowEdit	9.18
	FlowEdit$_{+\text{Ours}}$	9.52
	RF-Solver	21.78
	RF-Solver$_{+\text{Ours}}$	22.80
	DNAEdit	9.31
	DNAEdit$_{+\text{Ours}}$	9.64
SD3	FlowEdit	3.72
	FlowEdit$_{+\text{Ours}}$	3.88
SD3.5	RF-Solver	8.37
	RF-Solver$_{+\text{Ours}}$	8.86
	DNAEdit	4.09
	DNAEdit$_{+\text{Ours}}$	4.41

features and performing lightweight attention modulation at selected timesteps during inversion and denoising. Consequently, SMART-Edit achieves an effective balance between computational efficiency and editing performance, demonstrating its practicality for real-world image editing applications.

Appendix F. Failure Case Analysis and Future Directions

In this section, we analyze the failure cases of SMART-Edit across five challenging tasks, including object addition, attribute addition, small-object editing, occluded-object editing, and non-rigid editing. We also compare the performance of SMART-Edit with the baseline, DNAEdit [4]. The results are shown in Figure. F.7.

In Case 1, for the object addition task, we cannot find any explicit tokens in the original prompt. Therefore, SMART-Edit cannot semantically localize the editing region or generate a mask for background preservation. Consequently, its performance

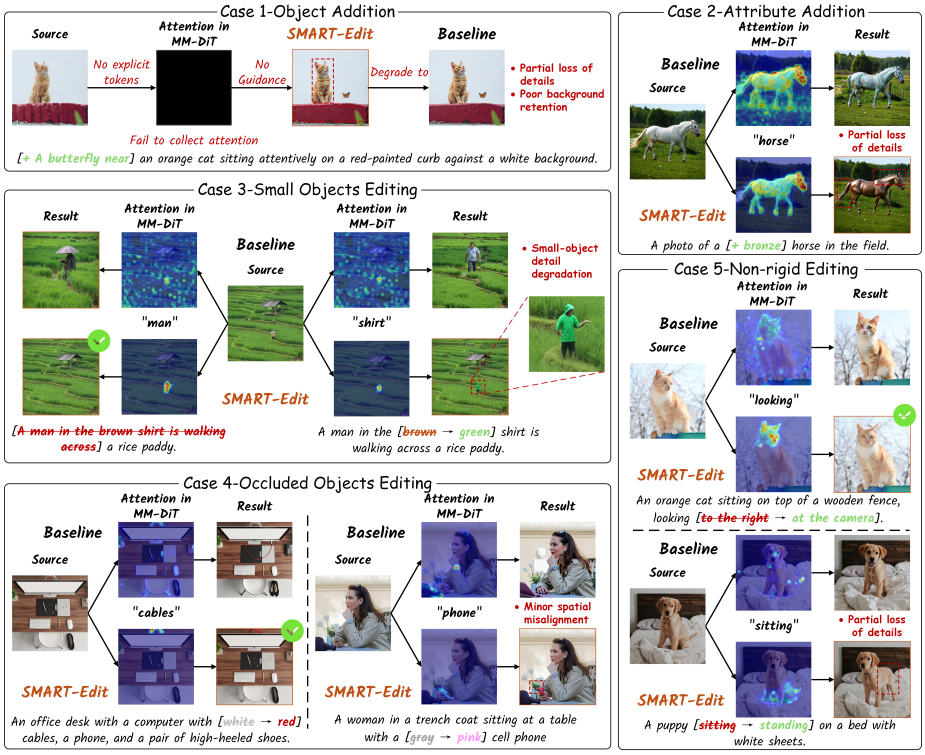


Figure F.7: **Qualitative failure case analysis of SMART-Edit on complex samples.** For object addition, SMART-Edit degrades to the baseline while successfully handling general attribute addition, small-object editing, occluded-object editing, and non-rigid editing. Although SMART-Edit still exhibits limitations on more complex samples, it accurately localizes the key editing tokens, leading to better editing quality than the baseline. The red dashed boxes highlight the regions where editing failures occur.

degrades to that of the baseline. Although the butterfly is successfully added, fine-grained details of the cat are lost, while the background, particularly the red-painted curb, is not fully preserved.

In Case 2, which involves an attribute addition task, SMART-Edit achieves more precise semantic localization of the horse than the baseline and effectively preserves the background information, resulting in better editing quality. However, we also observe a partial loss of fine-grained hair details around the horse's neck and tail. This may be attributed to the limitations of our method in precisely localizing fine-grained details such as hair. Meanwhile, accurately rendering and preserving such fine-grained

textures remains challenging for the underlying generative model itself [5].

In Case 3, for the small-object editing task, we evaluate both the removal of a tiny person in the image and the modification of the person's clothing color. We observe that the baseline struggles to effectively localize the person and the clothing to be edited, whereas SMART-Edit can accurately identify the corresponding editing regions. Moreover, even when the person occupies only a very small portion of the image, SMART-Edit can still remove the person precisely. However, we also observe that when modifying the color of the person's clothing, SMART-Edit inadvertently changes the color of the hat as well, and the fine-grained details of the person's body exhibit slight deviations from the original image. These observations suggest that editing extremely small objects remains constrained by the capabilities of the pretrained generative model. This limitation is consistent with the challenges observed in fine-grained and region-based generation and editing with flow-matching models [6, 7].

In Case 4, we evaluate the editing effectiveness of SMART-Edit on occluded objects. In the left sample, the cable is partially occluded by the computer monitor. Although the baseline can partially localize the cable, it suffers from severe semantic leakage to other white objects, resulting in an additional red cable being rendered and the background being corrupted. In contrast, SMART-Edit accurately localizes the cable and successfully edits it while preserving the surrounding background. In the right sample, the baseline directly leaks the semantics of the phone to the watch and the lower-left region, resulting in an additional pink laptop being generated. In comparison, SMART-Edit accurately localizes and edits the phone despite the presence of occlusion. A minor imperfection is that the phone in the original image is occluded by both fingers and leaves. After editing, SMART-Edit preserves the occlusion caused by the finger but incorrectly occludes leaves, resulting in minor spatial misalignment. This may be attributed to the limited spatial awareness of current generative models and the lack of explicit spatial information modeling during the editing process [8, 9].

In Case 5, we investigate the capability of SMART-Edit for non-rigid editing. In the upper sample, when the baseline attempts to localize the cat's motion through attention, the editing semantics leak into the background, resulting in substantial modifications to the background as well as changes in the cat's pose and appearance. In contrast,

SMART-Edit accurately localizes the joints associated with the desired motion and only adjusts the orientation of the cat's head while preserving the rest of the image. In the lower strong non-rigid editing sample, SMART-Edit successfully localizes the editing joints that are difficult for the baseline to identify and performs the desired edit according to the prompt. Despite the well-preserved background, the editing process requires synthesizing details of the dog's back and side that are not visible in the original image. Due to the lack of sufficient semantic references, some fine-grained details are lost during the editing process.

In summary, when no explicit token is available to drive the editing process in the object addition task, SMART-Edit degrades to the baseline. For the other tasks, including attribute addition, small-object editing, occluded-object editing, and non-rigid editing, SMART-Edit can accurately localize the key regions to be edited, leading to better editing results than the baseline. Although some challenging cases still exhibit limitations, such as the loss of fine-grained details or minor spatial misalignment, SMART-Edit generally preserves the surrounding background and remains faithful to the intended editing semantics.

These failure cases also point to several promising directions for future work. First, although MM-DiT can successfully localize the target regions with the aid of SMART-Edit, some fine-grained details remain visible but difficult to faithfully regenerate, mainly because the VAE-based latent representation inevitably discards high-frequency visual information. Recent pixel-space generation approaches [10, 11, 12] therefore provide a potential direction for better preserving and reconstructing such details. Second, existing image editing methods rarely incorporate explicit spatial awareness into the editing process, highlighting the importance of enhancing the spatial perception and spatial consistency of generative models [13]. Finally, reducing the reliance on explicit editing tokens [14, 15, 16, 17] and developing task-agnostic guidance mechanisms could enable more general and flexible image editing across different editing scenarios.

Appendix References

[1] X. Ma, G. Fang, M. Bi Mi, X. Wang, Learning-to-cache: Accelerating diffusion transformer via layer caching, Advances in Neural Information Processing Systems 37 (2024) 133282–133304.

[2] J. Liu, C. Zou, Y. Lyu, J. Chen, L. Zhang, From reusing to forecasting: Accelerating diffusion models with taylorseers, in: 2025 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE, 2025, pp. 15853–15863.

[3] O. Avrahami, O. Patashnik, O. Fried, E. Nemchinov, K. Aberman, D. Lischinski, D. Cohen-Or, Stable flow: Vital layers for training-free image editing, in: Proceedings of the Computer Vision and Pattern Recognition Conference, 2025, pp. 7877–7888.

[4] C. Xie, M. Li, S. Li, Y. Wu, Q. Yi, L. Zhang, Dnaedit: Direct noise alignment for text-guided rectified flow editing, Advances in Neural Information Processing Systems 38 (2026) 124456–124474.

[5] N. Matsunaga, M. Ishii, A. Hayakawa, K. Suzuki, T. Narihira, Fine-grained image editing by pixel-wise guidance using diffusion models, AI for Content Creation workshop at CVPR2023 (2023).

[6] J. Lee, D. S. Jung, K. Lee, K. M. Lee, Semanticdraw: towards real-time interactive content creation from image diffusion models, in: 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2025, pp. 13021–13030.

[7] V. T. Hu, W. Zhang, M. Tang, P. Mettes, D. Zhao, C. Snoek, Latent space editing in transformer-based flow matching, in: Proceedings of the AAAI conference on artificial intelligence, Vol. 38, 2024, pp. 2247–2255.

[8] T. Gokhale, H. Palangi, B. Nushi, V. Vineet, E. Horvitz, E. Kamar, C. Baral, Y. Yang, Benchmarking spatial relationships in text-to-image generation, arXiv preprint arXiv:2212.10015 (2022).

[9] Z. Wang, J. Xu, Z. Zhang, T. Pang, C. Du, H. Zhao, Z. Zhao, Genspace: Benchmarking spatially-aware image generation, Advances in Neural Information Processing Systems 38 (2026).

[10] Y. Yu, W. Xiong, W. Nie, Y. Sheng, S. Liu, J. Luo, Pixeldit: Pixel diffusion transformers for image generation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2026, pp. 14273–14282.

[11] S. Wang, Z. Gao, C. Zhu, W. Huang, L. Wang, Pixnerd: Pixel neural field diffusion, in: International Conference on Learning Representations, Vol. 2026, 2026, pp. 43559–43580.

[12] T. Li, K. He, Back to basics: Let denoising generative models denoise, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2026, pp. 36115–36125.

[13] H. Lee, S. Choi, I. Hwang, B.-T. Zhang, Towards spatially consistent image generation: On incorporating intrinsic scene properties into diffusion models, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 40, 2026, pp. 5863–5871.

[14] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, D. Cohen-Or, Prompt-to-prompt image editing with cross attention control, in: International Conference on Learning Representations, 2023.

[15] M. Cao, X. Wang, Z. Qi, Y. Shan, X. Qie, Y. Zheng, Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing, in: Proceedings of the IEEE/CVF international conference on computer vision, 2023, pp. 22560–22570.

[16] Z. Zhou, Y. Deng, X. He, W. Dong, F. Tang, Multi-turn consistent image editing, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2025, pp. 15792–15801.

[17] Y. Hu, J. Peng, Y. Lin, T. Liu, X. Qu, L. Liu, Y. Zhao, Y. Wei, Dcedit: Dual-level controlled image editing via precisely localized semantics, arXiv preprint arXiv:2503.16795 (2025).