A. Related Work

A.1. Diffusion-based Image Editing

Diffusion-based image editing strives to (1) preserve the visual content of a given source image while (2) modifying specific regions as instructed by text prompts. Prompt-to-Prompt (P2P) [13] pioneered this line of research by introducing *cross-attention scheduling*, which injects the cross-attention maps obtained during the reconstruction of the source image back into the editing process. Subsequent works further refined attention manipulation: MasaCtrl [3] imposes *mutual self-attention control* to maintain spatial consistency, whereas Free-Prompt-Editing (FPE) [19] decomposes cross-attention scheduling layer by layer for finer control.

A complementary thread focuses on *inversion*, operating under the intuition that an accurate inversion of the source image yields higher-quality edits. Several methods optimize text embeddings during inversion [5, 18, 21, 29, 30]; among them, InfEdit [29] proposes a training-free *Virtual Inversion* technique that achieves state-of-the-art results on multiple benchmarks.

Despite these advances, *multi-object* image editing remains under-explored. ZRIS [9] handles multi-object cases, but segments objects for *referring image segmentation*, not editing. Editing multiple target objects sequentially is straightforward but computationally expensive, as each object requires a separate diffusion pass. Our work addresses this gap by proposing a training-free framework that simultaneously handles multiple editing prompts and aligns user intent with the attention mechanism, enabling efficient multi-object edits without sacrificing quality.

A.2. Mitigating Attribute Leakage

Attribute leakage occurs when a diffusion model assigns an attribute to an unintended object. Early work addressed the problem by injecting explicit linguistic structure: StructureDiffusion Guidance [10] constrains generation with a constituency tree or scene graph, while Attend-and-Excite [4] and SynGen [23] refine cross-attention so that each word attends to a single spatial region. These methods focus on the *attention maps* themselves, yet leakage can also stem from *text embeddings*: even a perfect attention map fails if the prompt embedding is semantically entangled.

ToMe [14] tackles embedding-level entanglement via *End Token Substitution* (ETS). It replaces the EOS embedding of the full prompt (e.g., "a yellow cat and a white dog") with the EOS embedding of a stripped prompt that omits attributes (e.g., "a cat and a dog"), thereby suppressing color–attribute leakage ("yellow dog", "white cat"). However, ETS does not address noun-to-noun confusion ("cat" versus "dog") and, being designed for pure image generation, offers no guarantee of consistency with a given source image—an essential requirement for editing. DPL [30] reduces leakage by iteratively optimizing token embeddings at inference time to align cross-attention maps with the prompt, yet this costly optimization still leaves leakage when EOS embeddings remain entangled (see Figure 3).

In summary, existing approaches either leave EOS embeddings untouched or require high-cost optimization. Our method instead offers a lightweight, *optimization-free* pipeline that simultaneously disentangles embeddings and aligns attention, achieving lower attribute leakage while preserving faithfulness to the source image.

B. Benchmark Construction Details

Benchmark overview Our benchmark is designed to evaluate attribute leakage in image editing tasks using diffusion models. Unlike existing benchmarks that focus on image quality and background preservation, our benchmark emphasizes preventing unintended changes in both target-external and target-internal regions. It consists of 20 diverse images, semi-automated object masks, and succinct prompt pairs for various editing types. To comprehensively evaluate models, we generate 10 random edit prompts for each combination of 5 edit types and 1–3 edited objects per image, resulting in a total of 3,000 diverse editing scenarios. By covering diverse editing scenarios and offering precise evaluation metrics, our benchmark provides a robust framework for improving the precision of image editing methods. Figure 6 illustrates examples, showing the source images, object masks, and associated editing prompts.

Image selection We curated a dataset of 20 images, evenly split between natural and artificial scenes, to provide diverse and challenging editing scenarios. All images were drawn from both free image repositories and the PIE-bench dataset [29]. To ensure complexity, we included only images containing at least three distinct objects.

Prompt construction ALE-Bench provides five editing types. The prompt templates for different editing types are as follows:

1. Color change: "{color}-colored {object}" (e.g., "car" → "red-colored car").
2. Object change: "{new object}" (e.g., "car" → "bus").
3. Material change: "{object} made of {material}" (e.g., "car" → "car made of gold").
4. Color and object change: "{color}-colored {new object}" (e.g., "car" → "blue-colored bus").
5. Object and material change: "{new object} made of {material}" (e.g., "car" → "bus made of gold").

We intentionally excluded combinations like "color and material" and "color, object and material" because such cases often lead to unrealistic or ambiguous prompts, such as "**silver**-colored car made of **gold**". These kinds of descrip-

tions are inherently challenging to interpret or generate, even for a human, making them impractical editing scenarios.

For each image, we generated 10 unique and random edit prompt instances for every combination of edit type and number of objects to edit. These prompts were created using attribute dictionaries containing target instances for colors, objects, and materials, with the assistance of ChatGPT to ensure diversity and consistency. This approach results in a systematic exploration of the attribute space across 20 images, 5 edit types, and varying numbers of objects, covering a total of 3,000 unique editing scenarios. Additionally, we emphasize the importance of user convenience by designing minimal prompt pairs that specify only the intended modification, avoiding the verbosity commonly seen in previous benchmarks.

Evaluation metrics In addition to standard metrics from PIE-bench—such as structural distance, background preservation (PSNR, SSIM, LPIPS, MSE), and editing performance (CLIP similarity)—we propose two novel metrics specifically designed to evaluate attribute leakage. The Target-External-Leakage Score (TELS) metric quantifies unintended changes to background regions during editing. This is calculated by measuring the CLIP scores between the background regions of the edited image and the target prompt. Lower TELS indicate minimal impact on the background, ensuring that non-target regions remain unaffected. The Target-Internal-Leakage Score (TILS) metric captures unintended cross-influence between multiple edited objects. For each edited object, we compute the CLIP scores between its edited region and the prompts intended for other objects, then take the mean scores across all object pairs. Lower TILS indicate that edits are confined to their respective objects without unintended interactions or overlaps.

Comparison with LoMOE-Bench LoMOE-Bench [7] evaluates overall fidelity in multi-object editing using approximately 1k edits across 64 images. In contrast, ALE-Bench focuses on probing *attribute leakage*, generating *3k edits* from just 20 carefully selected images. Rather than scaling the dataset broadly, ALE-Bench emphasizes depth by designing diverse, leakage-prone scenarios for each image. Since each additional image requires new object masks and source–target prompt pairs, annotation costs grow linearly. As a result, the two benchmarks serve complementary purposes: LoMOE-Bench measures broad editing fidelity, while ALE-Bench targets leakage robustness.

C. Experiments Details

Prompt construction For methods such as MasaCtrl and FPE that require only a single target prompt, the target prompt was constructed by concatenating all target object prompts with "and" to form a prompt. For methods like P2P and InfEdit that require both a source and a target prompt, the source prompt was similarly created by concatenating the source object prompts, while the target prompt was constructed by concatenating the target object prompts.

Method	P2P	MasaCtrl	FPE	InfEdit	ALE (Ours)
Runtime (sec) ↓	61.2	63.6	50.9	5.41	**4.31**

Table 4. Average runtime per edit on an RTX 6000 Ada Gen.

Hyperparameters For our method, we set the inference steps to 15 and the mask dilation ratio to 0.01, corresponding to a dilation of seven pixels. The self-attention control schedule was adjusted according to the type of edit: 1.0 for colors, 0.5 for objects, color+object, and material+object, and 0.6 for material. The same self-attention control schedule was applied to InfEdit and P2P, as this hyperparameter is shared. For all other hyperparameters of the baseline methods (MasaCtrl, FPE, P2P, InfEdit), we used the default settings provided in their official implementations.

D. Additional Results

Runtime comparison As shown in Table 4, ALE and InfEdit achieve significantly faster runtimes compared to other baselines, requiring only a few seconds per edit. This efficiency comes from leveraging virtual inversion via DDCM. In contrast, methods like P2P, MasaCtrl, and FPE rely on more expensive DDIM or null-text inversion processes, resulting in runtimes of nearly one minute per edit.

By object count Tables 5, 6, and 7 present the quantitative evaluation of our method and baselines on ALE-Bench across different numbers of editing objects. For the baseline methods, TELS and TILS decrease as the number of edited objects increases, as editing more objects provides a more detailed description of the image, reducing ambiguity. This trend highlights the baselines' dependence on long and detailed prompts. However, their editing performance decreases with an increasing number of edited objects, revealing their limitations in handling complex edits. In contrast, our method demonstrates robust performance across all object counts, consistently achieving the lowest leakage values, preserving structure and background, and maintaining competitive or superior editing performance.

By edit type We compare our methods with baselines across different edit types in Tables 8, 9, 10, 11, and 12. Across all edit types, our method consistently outperforms baselines by achieving lower leakage, better structural and background preservation, and strong editing performance. We provide more qualitative examples on ALE-Bench for each edit type in two objects editing in Figure 14.

E. Ablation Study Results

Ablation on EOS embedding methods To evaluate the effect of EOS embeddings, we studied several methods of modifying EOS embeddings: (1) Naive: No modification, using the original EOS embeddings; (2) Zeros: Replacing EOS embeddings with zero-valued vectors; (3) BOS: Substituting EOS embeddings with BOS (beginning-of-sequence) embeddings; (4) Empty String: Using EOS embeddings derived from an empty string. In Figure 4, our method demonstrates robust results across various scenarios, while the other methods often produce images that fail to follow the edit prompt or exhibit attribute leakage. A detailed quantitative comparison is provided in Table 13.

Another EOS modification method is proposed in [14], named End Token Substitution (ETS). ETS substitutes an embedding of EOS in a full prompt into an embedding of EOS in a rephrased prompt, which deletes all attribute expressions, e.g. "a yellow cat and a white dog" into "a cat and a dog". In Figure 10, TI leakage in ETS is observed, e.g. cats are generated instead of a cat and a dog, and jar is generated in a region where ghost should be. RGB-CAM is applied for both methods, therefore the cross-attention mappings are aligned with the prompt.

Our method consistently achieves the best editing performance while maintaining competitive structure and background preservation metrics. In contrast, the other methods reveal a trade-off between reducing leakage and maintaining high editing performance, highlighting the effectiveness of our approach in balancing these objectives.

Ablation on RGB-CAM and BB The results in Table 14 demonstrate the complementary strengths of RGB-CAM and BB in our method. While RGB-CAM effectively reduces TI leakage by confining edits to the targeted objects, its impact on TE leakage and background preservation is limited. Conversely, BB significantly lowers TE leakage by preserving non-target regions, improving background quality but slightly reducing editing performance. Combining all components (Ours) achieves the best overall balance, minimizing leakage while preserving structure and background, and maintaining strong editing performance, highlighting the synergy of these components.

Evaluation on PIE-Bench We also evaluated our method on the existing PIE-Bench [17] in addition to ALE-Bench. Since our method does not support all edit types in PIE-Bench, we conducted experiments on the four edit types that are compatible: *object change*, *content change*, *color change*, and *material change*.

PIE-Bench only considers scenarios with a single object editing, so we excluded the TI Leakage metric. When running our method, we used the blend word provided by PIE-Bench as the SAM prompt for mask generation. In cases where mask segmentation failed, we edited the image without cross-attention masking and background blending.

In Table 15, the results show that our method demonstrated the lowest attribute leakage and high editing performance among all methods, even on PIE-Bench. These findings further validate the robustness and versatility of our approach across different benchmarks. We also provide qualitative examples for each edit type from the PIE-Bench experiments in Figure 15.

Ablation on self-attention injection schedule The degree to which the structure of a source image needs to be preserved varies depending on the edit type. For edits like color changes, maintaining the original structure is crucial, while object changes may require more deviation from the source. Figure 8 shows the effect of the self-attention schedule across various scenarios. Adjusting the schedule from 0.0 to 1.0 shows that higher values preserve more structure, while lower values allow greater flexibility. Thus, selecting the appropriate self-attention schedule depends on the specific goals of the task. The hyperparameters we used were chosen based on these experimental findings.

F. Limitations

ALE-Bench While our benchmark provides a robust framework for evaluating attribute leakage in image editing, it has certain limitations. First, the range of editing tasks is currently limited to basic and mixed edits such as color, object, and material changes. More complex editing types, such as style transfer or pose modifications, are not covered in ALE-Bench. However, defining attribute leakage in edits like style transfer is inherently ambiguous, as such edits often involve holistic changes to the image, making it unclear which regions should remain unaffected. Addressing these challenges would require redefining attribute leakage for these contexts and designing new evaluation metrics tailored to these specific tasks. Second, the dataset size (20 images) may limit the evaluation of models trained on larger or more diverse datasets. Future updates of ALE-Bench could expand its scope by incorporating additional images, and more diverse editing types to overcome these limitations.

Failure cases Our framework leverages two backbone models, a pre-trained diffusion model and a segmentation model, Grounded-SAM. Consequently, it may fail when the task exceeds the capabilities of these backbone models. For instance, overly rare or complex prompts that the pre-trained diffusion model cannot handle (Figure 12), objects that are difficult for the segmentation model to recognize, or incomplete segmentation masks generated by the model (Figure 13) can lead to unsatisfactory results. However, since our method operates in parallel with advancements in these backbone models, we anticipate that such failure cases will decrease as these models continue to improve.

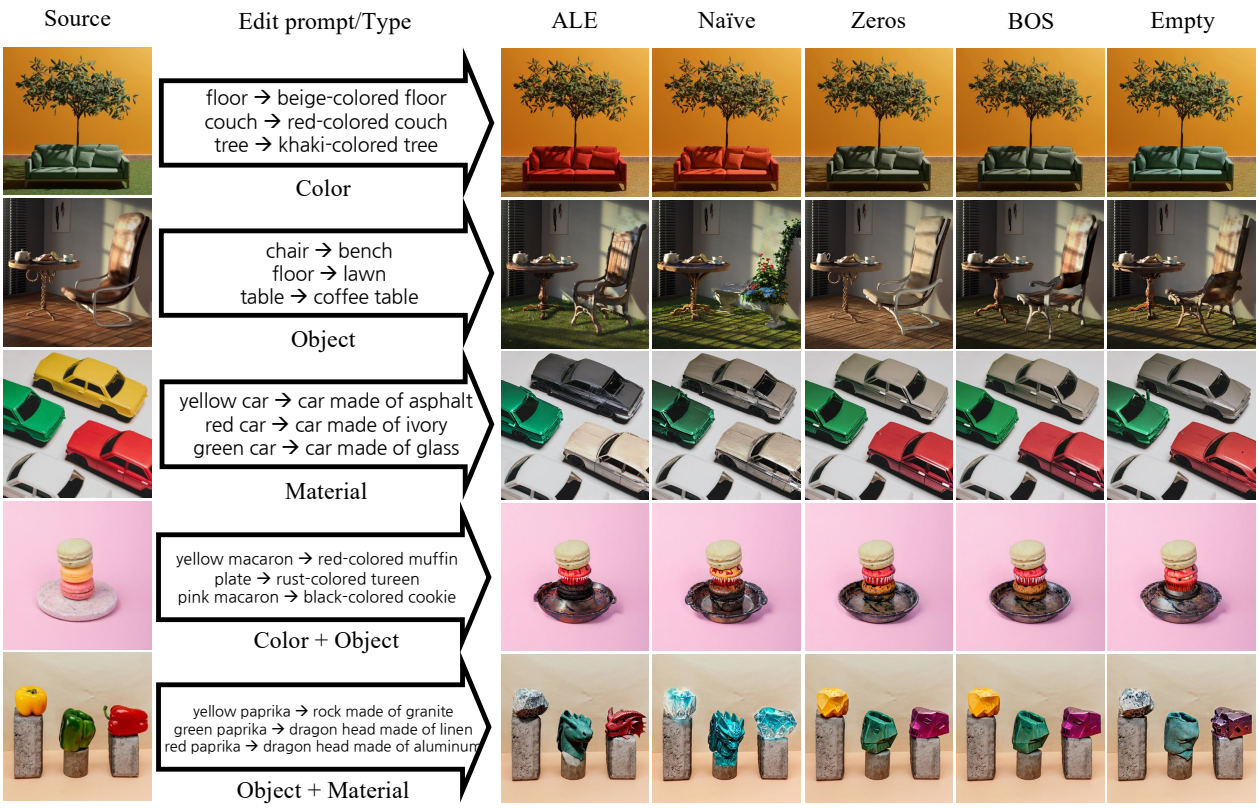

Figure 9. Qualitative examples from the EOS ablation study. While our method produces convincing results, other methods fail to generate the target object or exhibit attribute leakage. For instance, using the naive EOS to edit an object generates plants in place of the chair. This occurs due to attribute leakage from the word lawn to bench, resulting in chair-shaped flowers.

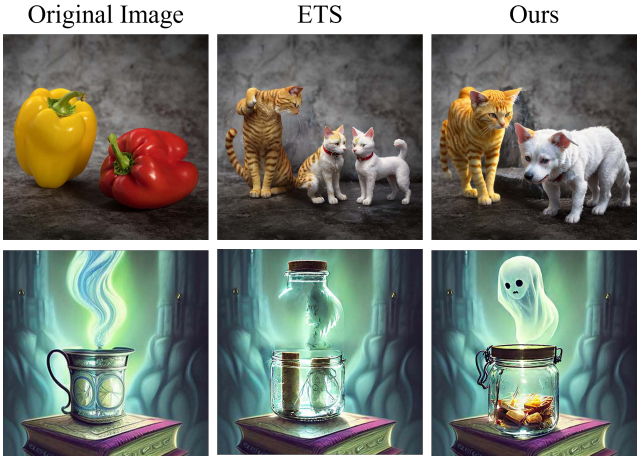

Figure 10. ETS, which is proposed in [14], fails to generate intended results. Edit prompt: (up) yellow paprika → yellow cat, red paprika → white dog. (down) cup → jar, steam → ghost.

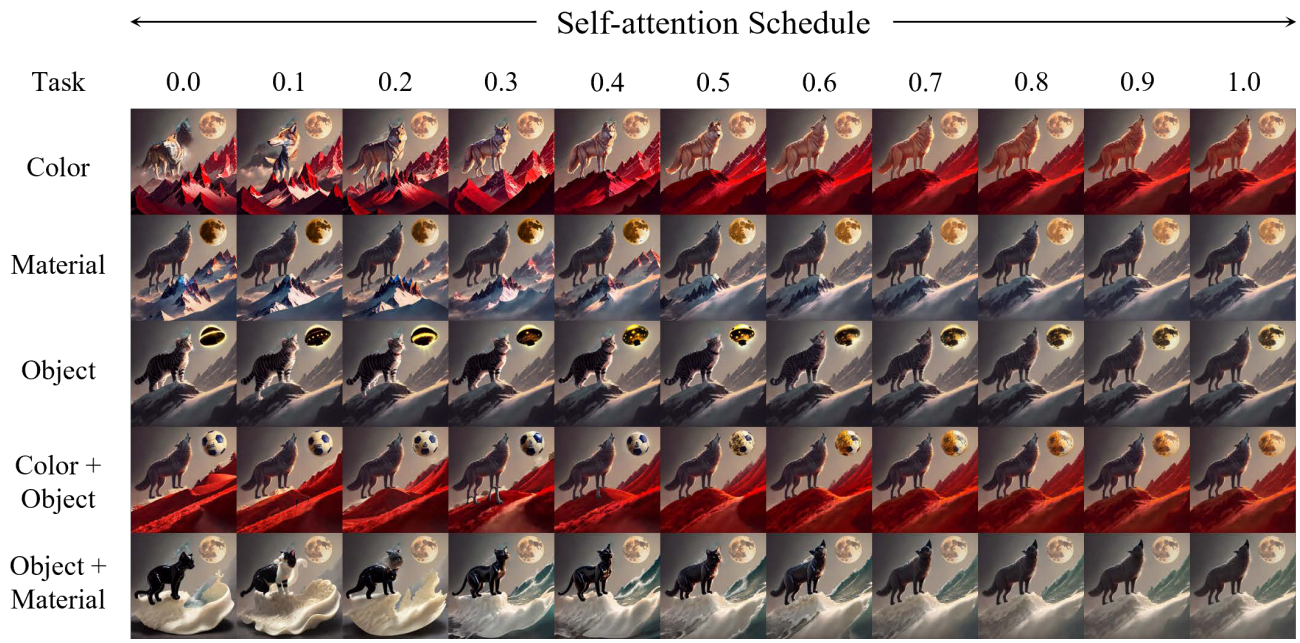

Figure 11. Ablation study on self-attention injection schedule. A schedule value specifies the fraction of early denoising steps during which self-attention maps from the source image are injected (e.g., 0.3 → first 30 % of steps). Larger values preserve more of the source structure and content, whereas smaller values grant greater freedom to satisfy the edit. The optimal schedule therefore varies by edit type. Prompts for each editing type are: (1) color: wolf → cream-colored wolf, mountain → crimson-colored mountain, (2) material: mountain → mountain made of crystal, moon → moon made of gold, (3) object: wolf → cat, moon → UFO, (4) color + object: moon → navy-colored soccer ball, mountain → crimson-colored hill, (5) object + material: cat → wolf made of rubber, mountain → wave made of ivory.

(a) Source image (b) Editing result (c) Generation result

Figure 12. Failure case due to the base model's inability. Editing prompt: cloud → cloud made of chrome. Figure 12c illustrates the generation result when given the prompt "cloud made of chrome".

(a) Source image (b) Editing result (c) Segmentation mask

Figure 13. Failure case due to SAM segmentation fail. Editing prompt: . . . cat . . . → . . . panda . . .. Figure 13c shows the unsuccessful segmentation of SAM.

Method	TELS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
				PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	24.91	0.1513	21.53	10.29	0.5306	0.10342	0.4737
MasaCtrl	23.36	0.1012	20.58	13.74	0.3671	0.05250	0.6645
FPE	24.42	0.1172	**22.94**	11.66	0.4677	0.08009	0.5194
InfEdit	21.98	0.0504	22.71	15.34	0.2495	0.04359	0.7057
ALE	**16.41**	**0.0088**	22.62	**30.01**	**0.0405**	**0.00167**	**0.9049**

Table 5. Quantitative evaluation of **editing one object** for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	20.87	17.52	0.1499	20.41	11.11	0.4506	0.08699	0.5560
MasaCtrl	19.58	16.90	0.0911	19.92	14.99	0.2886	0.04058	0.7357
FPE	20.44	17.68	0.1141	21.72	12.81	0.3880	0.06439	0.6050
InfEdit	19.16	16.86	0.0485	21.52	16.69	0.2026	0.03288	0.7719
ALE	**16.00**	**15.42**	**0.0165**	**22.06**	**30.06**	**0.0360**	**0.00146**	**0.9235**

Table 6. Quantitative evaluation of **editing two objects** for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	18.80	17.00	0.1531	20.05	12.05	0.3674	0.07318	0.6469
MasaCtrl	17.60	16.58	0.0866	19.53	16.26	0.2231	0.03245	0.8037
FPE	18.36	17.09	0.1180	20.99	14.00	0.3153	0.05247	0.6911
InfEdit	17.62	16.51	0.0463	21.10	18.18	0.1580	0.02540	0.8350
ALE	**15.89**	**15.36**	**0.0246**	**22.19**	**30.01**	**0.0323**	**0.00154**	**0.9426**

Table 7. Quantitative evaluation of **editing three objects** for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	23.20	18.02	0.1467	21.94	11.03	0.4529	0.0898	0.5753
MasaCtrl	21.70	17.35	0.0964	21.64	14.59	0.3113	0.04557	0.7283
FPE	22.33	17.94	0.1065	**23.23**	12.66	0.3926	0.06901	0.6266
InfEdit	19.68	17.31	0.0343	23.16	18.54	0.1401	0.02695	0.8347
ALE	**17.63**	**16.21**	**0.0089**	23.12	**32.97**	**0.0288**	**0.00079**	**0.9309**

Table 8. Quantitative evaluation of the **color change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	20.65	17.78	0.1535	20.38	11.23	0.4457	0.08750	0.5485
MasaCtrl	19.52	17.33	0.0901	19.72	15.48	0.2744	0.03801	0.7417
FPE	19.76	17.58	0.1221	21.30	13.12	0.3732	0.06236	0.6103
InfEdit	18.67	17.12	0.0504	21.10	16.59	0.2114	0.03381	0.7607
ALE	**15.86**	**16.25**	**0.0197**	**21.82**	**29.03**	**0.0386**	**0.00182**	**0.9218**

Table 9. Quantitative evaluation of the **object change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	21.40	17.07	0.1549	20.75	11.07	0.4417	0.08674	0.5429
MasaCtrl	20.82	17.00	0.0866	20.93	15.39	0.2755	0.03763	0.7418
FPE	21.91	17.68	0.1151	22.64	13.14	0.3781	0.05908	0.5943
InfEdit	20.33	16.87	0.0387	22.59	17.71	0.1753	0.02559	0.7862
ALE	**17.15**	**15.96**	**0.0118**	**22.94**	**30.63**	**0.0339**	**0.00120**	**0.9248**

Table 10. Quantitative evaluation of the **material change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	21.62	17.10	0.1483	20.71	11.12	0.4586	0.0896	0.5786
MasaCtrl	19.59	16.15	0.0986	19.18	14.43	0.3142	0.04634	0.7281
FPE	20.82	17.11	0.1159	21.40	12.42	0.4079	0.07175	0.6081
InfEdit	19.92	16.11	0.0619	21.23	15.04	0.2543	0.04503	0.7333
ALE	**15.30**	**14.01**	**0.0231**	**22.15**	**28.60**	**0.0407**	**0.00205**	**0.9206**

Table 11. Quantitative evaluation of the **color and object change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	20.76	16.31	0.1538	19.54	11.29	0.4488	0.0857	0.5490
MasaCtrl	19.28	15.86	0.0929	18.57	15.07	0.2893	0.04166	0.7332
FPE	20.55	16.62	0.1224	20.86	12.77	0.3998	0.06604	0.5865
InfEdit	19.34	16.03	0.0567	20.80	15.81	0.2356	0.03842	0.7394
ALE	**14.55**	**14.51**	**0.0196**	**21.42**	**28.88**	**0.0393**	**0.0019**	**0.9201**

Table 12. Quantitative evaluation of the **object and material change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
Naive	16.02	15.81	0.0156	21.86	30.14	0.0359	0.0014	0.9232
Zeros	**15.74**	**15.23**	**0.0107**	20.78	**31.22**	**0.0327**	**0.0011**	**0.9254**
BOS	15.76	15.27	0.0115	20.87	31.09	0.0334	0.0011	0.9241
Empty String	15.86	15.33	0.0139	21.25	30.61	0.0342	0.0013	0.9248
ALE	16.03	15.28	0.0167	**22.20**	30.04	0.0361	0.0014	0.9228

Table 13. Ablation study on different strategies for handling EOS embeddings in the prompt. While ALE shows slightly higher leakages compared to others, it achieves the best editing performance. All experiments were conducted with both RGB-CAM and BB applied.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
ORE	20.05	16.87	0.0521	21.81	16.16	0.2182	0.0380	0.7591
ORE+RGB	18.99	15.46	0.0436	**22.42**	17.48	0.1805	0.0291	0.7887
ORE+BB	16.12	16.58	**0.0164**	21.56	29.88	0.0368	0.0015	0.9219
ALE	**16.03**	**15.28**	0.0167	22.20	**30.04**	**0.0361**	**0.0014**	**0.9228**

Table 14. Ablation study comparing the components of our method: object-restricted embeddings (ORE), region-guided blending cross-attention masking (RGB), and background blending (BB). RGB markedly reduces TILS, whereas BB substantially lowers TELS. When ORE is used *without* RGB, it relies solely on the base embedding E'_{base} (i.e., the ORE and ORE + BB cases). Integrating all three components (ALE) yields the best overall performance across nearly every metric, underscoring their complementary strengths.

Method	TELS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
				PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	26.20	0.1571	23.74	11.11	0.4270	0.0919	0.4600
MasaCtrl	24.48	0.0856	22.16	15.81	0.2540	0.0334	0.6803
FPE	25.64	0.1265	**23.89**	13.35	0.3499	0.0581	0.5346
InfEdit	24.51	0.0446	22.92	19.41	0.1519	0.0168	0.7581
ALE	**22.94**	**0.0238**	22.87	**28.77**	**0.0580**	**0.0046**	**0.8865**

Table 15. Evaluation results on PIE-Bench for compatible edit types (object change, content change, color change, and material change). Our method achieves the lowest TELS and demonstrates the best structure and background preservation while maintaining competitive editing performance.

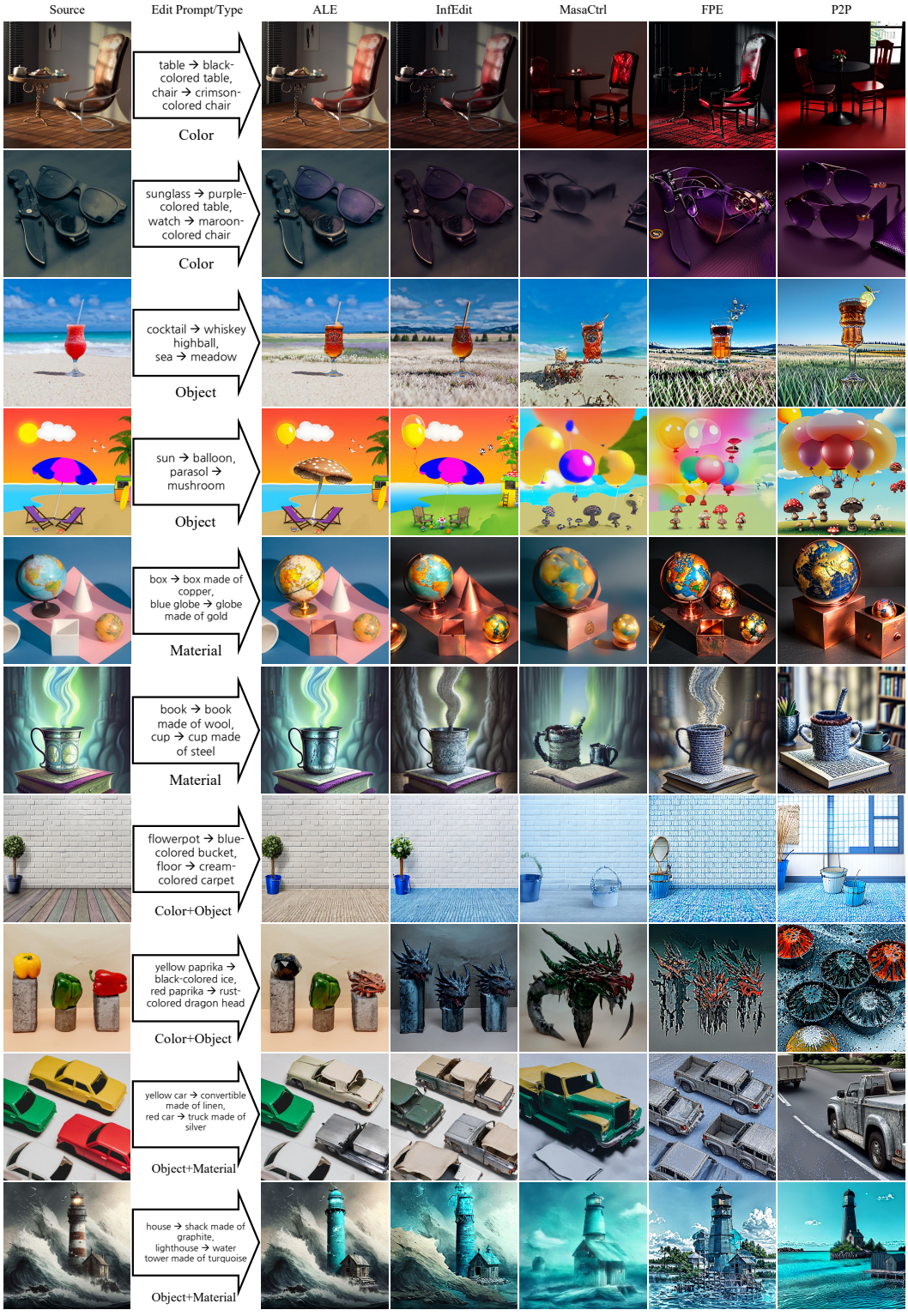

Figure 14. Qualitative examples of editing results for each edit type on ALE-Bench. Two examples are provided for each edit type. The left side of $\rightarrow$ represents the source prompt, and the right side represents the target prompt.

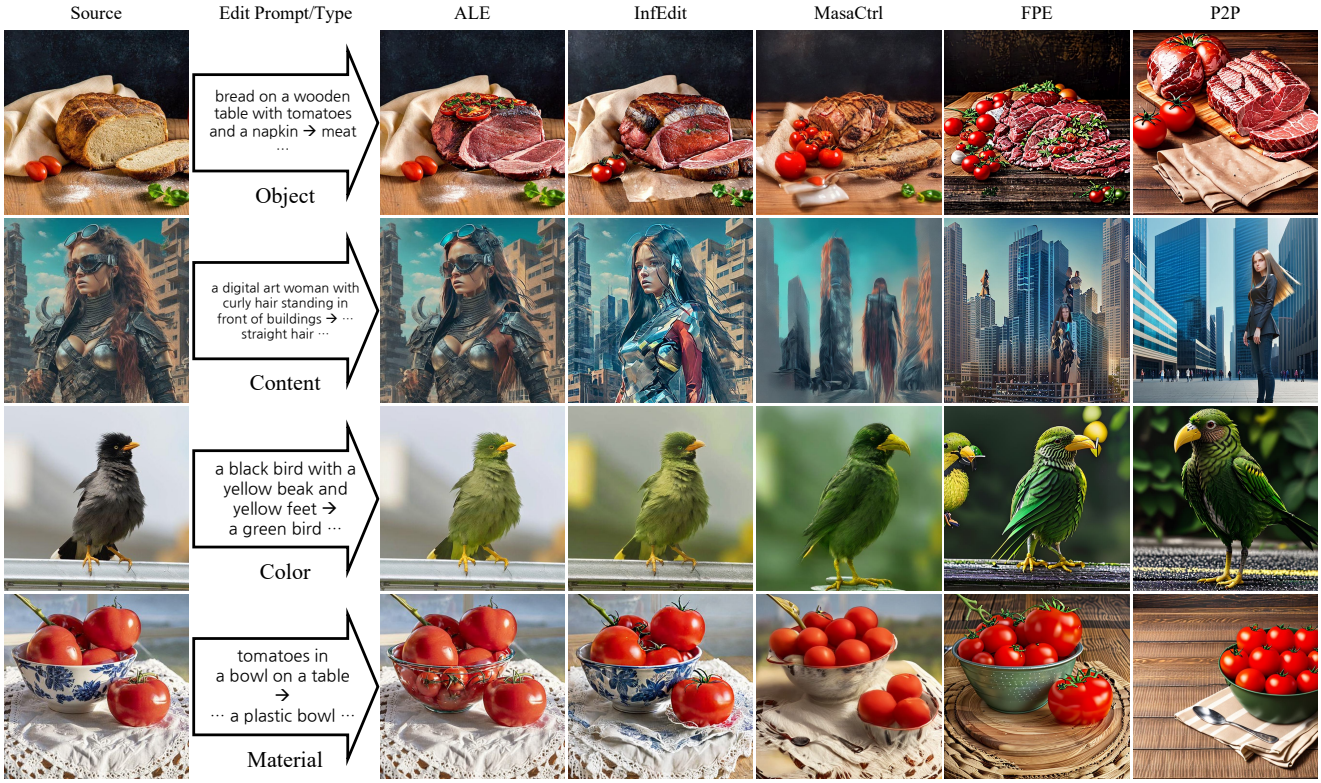

Figure 15. Qualitative examples of editing results for the four compatible edit types on PIE-Bench: object change, content change, color change, and material change. In edit prompt column, the left side of the arrow → represents the source prompt, and the right side represents the target prompt, with unchanged parts omitted as "..." for brevity. Baseline methods exhibit attribute leakage or fail to preserve the source image structure, while our method achieves more precise edits with minimal leakage.