Image Editing Studio with Diffusion Models

Junseok Ko
Statistics, Inha University
juniboy97@naver.com

Abstract

This report explores research on real image editing methods with diffusion models(Rombach et al. (2022)). Since real image editing requires latents of real images, inversion method is necessary. Therefore, this study examines various inversion methods and their applications in different image editing models. Quantitative evaluations were conducted using commonly used metrics such as CLIP Score, LPIPS, PSNR, SSIM, and Structure Distance, following previous studies. The results were also compared with those from Direct Inversion(Ju et al. (2023)).

1 Background

Before explaining the editing methods and inversion methods, I will first introduce the fundamental concepts of diffusion models and latent diffusion models.

1.1 Diffusion Models

Diffusion Models are generative models that model the distribution of the training data $x_0 \sim q(x_0)$. A diffusion model consists of two main processes: the forward process and the reverse process. The forward process gradually adds noise to the data until it transforms into pure Gaussian noise. This process follows a Markovian Gaussian diffusion formulation, where the data at each time step x_t is obtained by adding Gaussian noise to the previous step:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta}x_{t-1}, \beta_t I) \tag{1}$$

where β_t is a predefined noise schedule controlling how much noise is added at each step. As t increases, the data distribution gradually approaches a standard normal distribution $\mathcal{N} \sim (0, I)$. The reverse process aims to recover the original data by removing noise step by step. Instead of directly removing, the model learns to estimate the posterior distribution:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{2}$$

A neural network $\epsilon_\theta(x_t, t)$is trained to predict the noise component ϵ, which allows us to iteratively denoise the image, moving from pure Gaussian noise back to a structured image. This neural network is trained using L2 loss, which ensures that the network accurately predicts the added noise:

$$L(\theta) = \mathbb{E}_{x_0,\epsilon,t}[\|\epsilon - \epsilon_\theta(x_t, t)\|^2] \tag{3}$$

This enables the model to reverse the diffusion process effectively. Figure 1 illustrates the training and sampling algorithm of DDPM.

1.2 Latent Diffusion Models

Diffusion models require computations for every pixel in an image, leading to high computational costs and slow processing. This issue becomes even more significant for high-resolution images, where the computational complexity increases drastically. Instead of performing computations

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Figure 1: Training and sampling of DDPM

directly on the image, LDMs use a Variational Autoencoder (VAE) to encode the image into a lower-dimensional latent space. The diffusion process is then applied in this latent space, reducing the overall computational burden and improving efficiency. LDMs introduce a cross-attention layer in the UNet architecture, enabling the injection of conditions (e.g., text prompts, other images). This allows for conditional image generation, meaning the model can generate images based on specific inputs.

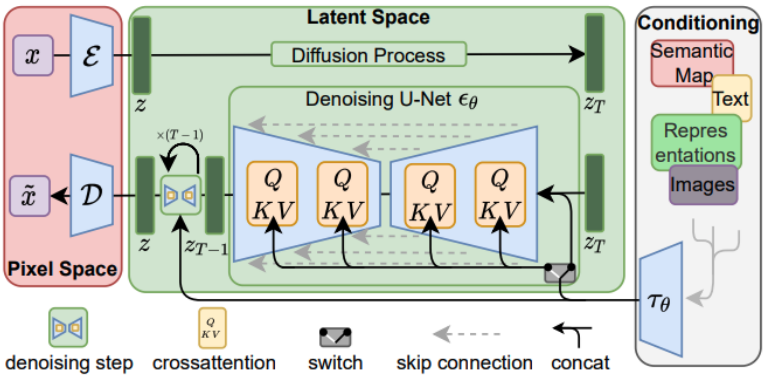


Figure 2: Stable Diffusion Architecture

2 Image Editing Methods with Diffusion Models

The image editing methods include prompt-to-prompt(Hertz et al. (2022)), plug-and-play(Tumanyan et al. (2022b)), and masactrl(Cao et al. (2023)). The commonality among these methods is that they edit images by manipulating the attention layer.

2.1 Prompt-to-Prompt(Hertz et al. (2022))

In prompt-to-prompt, the goal is to obtain an edited image I^* that maintain the content and structure of the original image while using only the modified prompt P^* to exclude the parts corresponding to the modified prompt, given that the text-guided diffusion model generates the image I from the text prompt P and a random seed s. The key method is to edit the image by modifying the pixel-to-text interaction that occurs in the cross-attention layer.

There are three edit functions.

Word swap :

$$\text{Edit}(M_t, M_t^*, t) := \begin{cases} M_t^*, & \text{if } t < \tau \\ M_t, & \text{otherwise.} \end{cases} \tag{4}$$

Algorithm 1: Prompt-to-Prompt image editing

Input: A source prompt $\mathcal{P}$, a target prompt $\mathcal{P}^*$, and a random seed s.
Output: A source image x_{src} and an edited image x_{dst}.
$z_T \sim N(0, I)$ a unit Gaussian random variable with random seed s;
$z_T^* \leftarrow z_T$;
for $t = T, T-1, \ldots, 1$ **do**
  $z_{t-1}, M_t \leftarrow DM(z_t, \mathcal{P}, t, s)$;
  $M_t^* \leftarrow DM(z_t^*, \mathcal{P}^*, t, s)$;
  $\widehat{M}_t \leftarrow Edit(M_t, M_t^*, t)$;
  $z_{t-1}^* \leftarrow DM(z_t^*, \mathcal{P}^*, t, s_t)\{M \leftarrow \widehat{M}_t\}$;
end
Return (z_0, z_0^*)

Figure 3: Overall algorithm of Prompt-to-Prompt

Adding a New Phrase :

$$(\text{Edit}(M_t, M_t^*, t))_{i,j} := \begin{cases} (M_t^*)_{i,j}, & \text{if } A(j) = None \\ (M_t)_{i,A(j)}, & \text{otherwise.} \end{cases} \tag{5}$$

Attention Re-weighting :

$$(\text{Edit}(M_t, M_t^*, t))_{i,j} := \begin{cases} c \cdot (M)_{i,j}, & \text{if } j = j^* \\ (M_t)_{i,j}, & \text{otherwise.} \end{cases} \tag{6}$$

2.2 Plug-and-Play(Tumanyan et al. (2022b))

Plug-and-Play starts by obtaining the initial noise X_T^G from the base image I^G through DDIM inversion. The target image is then generated using a noise X_T^* that is the same as the initial noise. During this process, features and self-attention maps from reconstructing the base image through X_T^G are injected to create target image that maintains the structure of the base image.

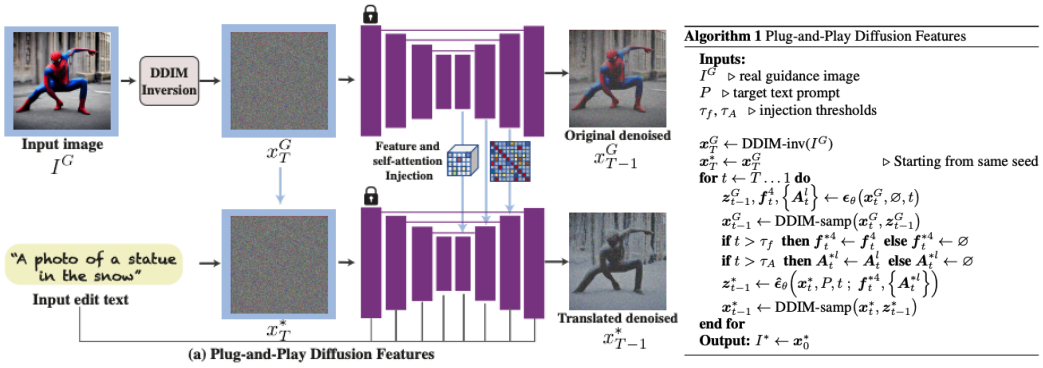


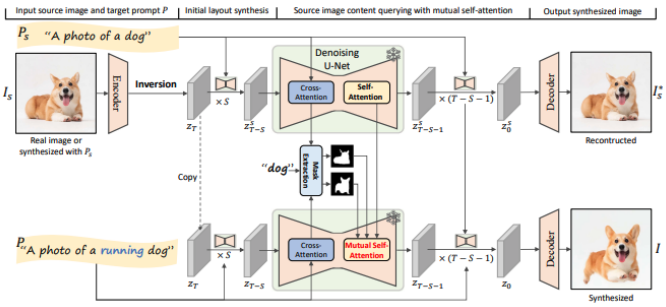
Figure 4: Overall framework and algorithm of plug-and-play diffusion features

Specifically, features from Layer 4 and self-attention map from Layer 4-11 are injected.

2.3 MasaCtrl(Cao et al. (2023))

The goal of MasaCtrl is to generate an image I that follows a modified target prompt P given a source image I_s and a source prompt P_s. In this process, I preserves the object contents of I_s. To achieve this, a mask-guided mutual self-attention mechanism is proposed.

First, the mutual self-attention mechanism uses the key and value features from the self-attention of the source image directly in the process of generating the target image. Specifically, after a few steps,



Algorithm 1 MasaCtrl: Tuning-Free Mutual Self-Attention Control

Input: A source prompt P_s, a modified prompt P, the source and target initial latent noise maps z_T^s and z_T.

Output: Latent map z_0^s, edited latent map z_0 corresponding to P_s and P.

for $t = T, T-1, ..., 1$ **do**
  $\epsilon_s, \{Q_s, K_s, V_s\} \leftarrow \epsilon_\theta(z_t^s, P_s, t)$;
  $z_{t-1}^s \leftarrow \text{Sample}(z_t^s, \epsilon_s)$;
  $\{Q, K, V\} \leftarrow \epsilon_\theta(z_t, P, t)$;
  $\{Q^\star, K^\star, V^\star\} \leftarrow \text{EDIT}(\{Q, K, V\}, \{Q_s, K_s, V_s\})$;
  $\epsilon = \epsilon_\theta(z_t, P, t; \{Q^\star, K^\star, V^\star\})$;
  $z_{t-1} \leftarrow \text{Sample}(z_t, \epsilon)$;
end for

Return z_0^s, z_0

Figure 5: Overall framework and algorithm of MasaCtrl

the mutual self-attention control is applied only to the decoder part of the U-Net. The EDIT function in the MasaCtrl algorithm is formulated as follows.

$$\text{EDIT} := \begin{cases} \{Q, K_s, V_s\}, & \text{if } t > S \text{ and } l > L, \\ \{Q, K, V\}, & \text{otherwise.} \end{cases} \tag{7}$$

When using mutual self-attention mechanism, it fails in cases where the background and objects are too similar. Therefore MasaCtrl utilize a semantic cross-attention map to generate masks that distinguish between the foreground and background in both the source image I_s and the target image I. The object region and background region query content information from corresponding limited areas instead of all features. The masks extracted for the foreground object in I_s and I are denoted as M_s and M, respectively. These masks allow us to restrict the content information queried from the object in I to only that from the object region in I_s:

$$f_0^l = \text{Attention}(Q^l, K_s^l, V_s^l; M_s), \tag{8}$$

$$f_b^l = \text{Attention}(Q^l, K_s^l, V_s^l; 1 - M_s), \tag{9}$$

$$\bar{f}^l = f_0^l * M + f_b^l * (1 - M), \tag{10}$$

3 Inversion Methods

The image editing methods mentioned above are techniques for generating a target image while creating a source image. Therefore, an inversion method is necessary for applying these techniques to real image editing. In this section, I will introduce various inversion methods.

3.1 DDIM Inversion(Song et al. (2022))

Denoising Diffusion Implicit Models(DDIM) extends the DDPM framework by providing a non-Markovian diffusion process that allows for more efficient sampling. Unlike DDPM, DDIM introduces a deterministic process for generating images from noise. While the forward process remains the same as in DDPM, DDIM introduces a different reverse process that no longer follows a Markovian assumption. Reverse process is defined as :

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\left(\frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}\right) + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_t, t), \tag{11}$$

where $\epsilon_\theta$is the same neural network as in DDPM, and $\bar{\alpha}_t$is defined as in DDPM. The key difference is that DDIM removes the stochasticity from the reverse process, making it deterministic and thus allowing for more controlled and efficient sampling.

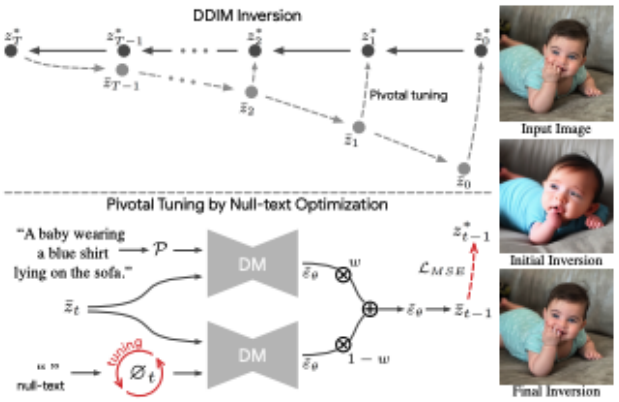
Thanks to this key difference, DDIM allows for the encoding of images into their corresponding latent variables. This encoding process is referred to as DDIM inversion.

3.2 Null-Text Inversion(Mokady et al. (2022))

In the unconditional diffusion model, small errors accumulate at each step through DDIM inversion, but these accumulated errors can be considered negligible. However, in Stable Diffusion, a large guidance scale with $w > 1$ is used. In this context, the guidance scale amplifies the accumulated errors. Therefore, DDIM inversion with classifier-free guidance not only introduces visual artifacts but may also result in the obtained noise vector deviating from a Gaussian distribution. The latter decreases editability.

To address this issue, this work proposes a method for optimizing the null-text $\varnothing$. The null-text $\varnothing$ is optimized at each timestep t through the following objective:

$$\min_{\varnothing_t} \|z^*_{t-1} - z_{t-1}(\bar{z}_t, \varnothing_t, \mathcal{C})\|_2^2 \tag{12}$$



Algorithm 1: Null-text inversion

Input: A source prompt embedding $\mathcal{C} = \psi(\mathcal{P})$ and input image $\mathcal{I}$.
Output: Noise vector z_T and optimized embeddings $\{\varnothing_t\}_{t=1}^T$.

Set guidance scale $w = 1$;
Compute the intermediate results $z^*_T, \ldots, z^*_0$ using DDIM inversion over $\mathcal{I}$;
Set guidance scale $w = 7.5$;
Initialize $\bar{z}_T \leftarrow z^*_T$, $\varnothing_T \leftarrow \psi(\text{""})$;
for $t = T, T-1, \ldots, 1$ **do**
  for $j = 0, \ldots, N-1$ **do**
    $\varnothing_t \leftarrow \varnothing_t - \eta \nabla_{\varnothing} \|z^*_{t-1} - z_{t-1}(\bar{z}_t, \varnothing_t, \mathcal{C})\|_2^2$;
  end
  Set $\bar{z}_{t-1} \leftarrow z_{t-1}(\bar{z}_t, \varnothing_t, \mathcal{C})$, $\varnothing_{t-1} \leftarrow \varnothing_t$;
end
Return $\bar{z}_T, \{\varnothing_t\}_{t=1}^T$

Figure 6: Overall framework and algorithm of Null-Text Inversion

This process yields $\bar{z}_t$ and the null text $\varnothing$, which can be used to edit real images.

3.3 Edit-Friendly DDPM Inversion(Huberman-Spiegelglas et al. (2024))

DDIM inversion has been primarily used, as it is accurate when the number of diffusion timesteps is large; however, it often results in sub-optimal outcomes in text-guided editing. To address this issue, edit-friendly DDPM inversion uses DDPM latents instead of DDIM latents. The existing DDPM latents in previous works are not edit-friendly, so an edit-friendly DDPM latent is proposed.

DDPM forward process is as follows :

$$x_t = \sqrt{1-\beta_t} x_{t-1} + \sqrt{\beta_t} n_t, \; t = 1, \cdots, T \tag{13}$$

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t} \epsilon_t, \tag{14}$$

It is important to note that is the Equation (14), the vectors $\{\epsilon_t\}$ are not independent. This is because each ϵ_t corresponds to the accumulation of the noises $n_1, \cdots, n_t$ so that ϵ_t and ϵ_{t-1} are highly correlated for all t.

Instead of using Equation 14, this work propose constructing $x_1, \cdots, x_T$ from x_0 as follows :

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t} \tilde{\epsilon}_t, t = 1, \cdots, T \tag{15}$$

where $\tilde{\epsilon} \sim \mathcal{N}(0, I)$ are statistically independent. Equation (15) and (14) have similar forms but fundamentally represent different probabilistic processes. In Equation (14), all pairs of consecutive ϵ_t are highly correlated, whereas in Equation (15), $\tilde{\epsilon}_t$ is independent.

3.4 Direct Inversion(Ju et al. (2023))

Optimization-based methods, such as Null-Text Inversion, do not affect the distribution in DDIM sampling while maintaining sufficient guidance for text conditions and preserving the editability

Algorithm 1 Edit-friendly DDPM inversion

Input: real image x_0
Output: $\{x_T, z_T, \ldots, z_1\}$
for $t = 1$ **to** T **do**
  $\tilde{\epsilon} \sim \mathcal{N}(0,\, 1)$
  $x_t \leftarrow \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\tilde{\epsilon}$
end for
for $t = T$ **to** 1 **do**
  $z_t \leftarrow (x_{t-1} - \hat{\mu}_t(x_t))/\sigma_t$
  $x_{t-1} \leftarrow \hat{\mu}_t(x_t) + \sigma_t z_t$ // to avoid error accumulation
end for
Return: $\{x_T, z_T, \ldots, z_1\}$

Figure 7: Overall algorithm of Edit-friendly DDPM inversion

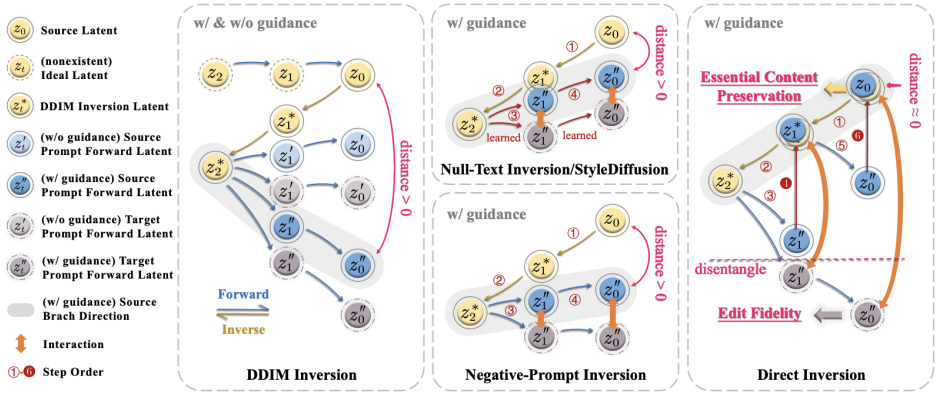


Figure 8: Comparisons among different inversion methods

of the diffusion model. So optimization-based methods perform better among previous inversion methods.

Optimization-based inversion learns a specific latent variable by minimizing the loss function, $z_t^{''} - z_t^*$, with the target of modifying $z_t^{''}$ to match z_t^*. However, optimizing a unified variable for both the source and target branches introduce several issues: (1) Increased inference time: The process duration is extended during inference. (2) Trade-off between optimization steps and preservation: To reduce process time, optimization is performed only for a limited number of iterations on the target latent. This results in a learned latent space with a large gap between $z_t^{''}$ and z_0, leading to poor preservation. (3) Misalignment with the expected input distribution: The learned variable acts as an input parameter for the generative model, which does not align with the expected input distribution of the diffusion model. This misalignment negatively impacts the integrity of the diffusion model. These issues hinder both practicality and editability.

Considering these issues, Direct Inversion was proposed. The key idea of Direct Inversion is to disentangle the source and target branches. For the source branch, they directly add $z_t^* - z_t^{''}$ back to $z_t^{''}$. This is a simple approach that directly corrects the deviation path and can be easily integrated into various editing methods. For the target branch, they leave it unchanged to maximize the potential of the diffusion model for target image generation. Figure 9 illustrates the overall algorithm of Direct Inversion. This approach addresses the three major issues of optimization-based inversion as follows:

- Since no optimization is required, it introduces minimal additional time overhead.
- Adding $z_t^* - z_t^{''}$ eliminates the discernible gap between $z_0^{''}$ and z_0
- It does not impact the expected distribution of the diffusion model's input.

Algorithm 1: Real Image Editing Pipeline with Direct Inversion

Input: A source prompt embedding C^{src} (or embedding of null for some editing methods), a target prompt embedding C^{tgt}, a real image or latent embedding z_0^{src}

Output: An edited image or latent embedding z_0^{tgt}

Part I : Inverse z_0^{src}

$z_0^* = z_0^{src}$;

for $t = 1, \ldots, T-1, T$ **do**

  $z_t^* \leftarrow \text{DDIM\_Inversion}\left(z_{t-1}^*, t-1, \left[C^{src}, C^{tgt}\right]\right)$;

end

Part II: Perform editing on z_T^* with *Direct Inversion*

$z_T^{tgt} = z_T^*$; $z_T^{''} = z_T^*$;

for $t = T, T-1, \ldots, 1$ **do**

  $\left[o_{t-1}^{src}, o_{t-1}^{tgt}\right] \leftarrow z_{t-1}^* - \text{DDIM\_Forward}\left(z_t^{''}, t, \left[C^{src}, C^{tgt}\right]\right)$; // 1 calculate distance

  $z_{t-1}^{''} = \text{DDIM\_Forward}\left(z_t^{''}, t, \left[C^{src}, C^{tgt}\right]\right) + \left[o_{t-1}^{src}, 0\right]$; // 2 update $z_{t-1}^{''}$

  $z_{t-1}^{tgt} \leftarrow \text{DDIM\_Forward}_{\text{Editing\_Model}}\left(z_t^{tgt}, t, \left[C^{src}, C^{tgt}\right]\right) + \left[o_{t-1}^{src}, 0\right]$; // 3 add distance

end

Return z_0^{tgt}

Figure 9: Algorithm of Direct Inversion

4 Experiments & Results

All experiments were conducted using Stable Diffusion v1.4 with the same random seed. Evaluation was performed using PIE-Bench(Ju et al. (2023)) with seven different metrics : structure distance(Tumanyan et al. (2022a)), background preservation (PSNR, LPIPS(Zhang et al. (2018)), SSIM), text-image consistency (CLIP score of the whole image and regions in the editing mask). Figure 10 illustrates examples from PIE Bench.

Editing Type	Source Image	Source Prompt	Target Prompt	Editing Instruction	Editing Mask	Editing Type	Source Image	Source Prompt	Target Prompt	Editing Instruction	Editing Mask
0 Random		A [dog] sitting on a wooden chair	A [cat] sitting on a wooden chair	Change the animal from a cat to a dog		5 Change Pose		A greyhound [walking] through the grass	A greyhound [jumping] through the grass	Change the greyhound from walking to jumping	
1 Change Object		[Fishes] in the ocean	[Sharks] in the ocean	Change the fishes to sharks		6 Change Color		A [white] shirt	A [yellow] shirt	Change the color of the shirt from white to yellow	
2 Add Object		Asian woman with black hair	Asian woman with [flowers on her] black hair	Add flowers to the woman's hair		7 Change Material		Camera, polaroid, and travel photography	[Wooden toy] camera, polaroid, and travel photography	Make the camera a wooden toy	
3 Delete Object		A painting of [an orange chair and] a lamp in a living room	A painting of a lamp in a living room	Remove the chair		8 Change Backrgound		A girl sitting in the [ruins]	A girl sitting in the [beach]	Change the background from ruins to a beach	
4 Change Content		A cave with trees and [sparkling] river	A cave with trees and [blood] river	Change the river from sparkling to blood		9 Change Style		[Painting of] a bird standing on tree branch	[A real photo of] a bird standing on tree branch	Change the image from illustration to photo	

Figure 10: PIE-Bench

Below, I present the quantitative and qualitative evaluation results of edited images using DDIM inversion (DDIM), Null-Text Inversion (NT), Edit-Friendly DDPM Inversion (EF), and Direct Inversion (Direct).

4.1 Qualitative Results

This section presents a qualitative comparison of the results obtained by applying different inversion methods to various editing methods. As seen in Figure 11, Figure 12, Figure 13, DDIM inversion fails to achieve accurate reconstruction, resulting in edited images that deviate significantly from the original. On the other hand, while the edit-friendly DDPM achieves good reconstruction, its editing

results are suboptimal. Comparing null-text inversion and direct inversion, direct inversion produces better results in both reconstruction and editing.

4.2 Quantitative Results

I conducted a quantitative evaluation using PIE Bench by applying different inversion methods to each editing method. I compared the results to those obtained with Direct Inversion to assess how well they were reproduced. As seen in Table 1, the results were nearly identical to those from Direct Inversion.

	Inversion	Method	Structure Distance$_{\times 10^3}$ ↓		PSNR ↑		LPIPS$_{\times 10^3}$ ↓		MSE$_{\times 10^4}$ ↓		SSIM$_{\times 10^2}$ ↑		CLIP$_{score}$(Edit) ↑		CLIP$_{score}$(Whole) ↑	
			Paper	Our	Paper	Our	Paper	Our	Paper	Our	Paper	Our	Paper	Our	Paper	Our
PIE Bench	DDIM	PnP	28.22	28.18	22.28	22.27	113.46	113.47	83.64	82.83	79.05	79.33	22.55	22.58	25.41	25.36
	NT	PnP	–	26.76	–	22.20	–	112.73	–	84.83	–	79.47	–	22.57	–	25.39
	EF	PnP	–	28.62	–	22.13	–	121.2	–	118.13	–	79.17	–	22.34	–	25.55
	Direct	PnP	24.29	24.03	22.28	22.49	113.46	105.34	80.45	79.15	79.68	80.10	22.62	22.6	25.41	25.5
	DDIM	P2P	69.43	69.49	17.87	17.85	208.80	209.01	219.88	220.79	71.14	71.45	22.44	22.47	25.01	25.13
	NT	P2P	13.44	13.61	27.03	27	60.67	61.78	35.86	36.65	84.11	84.32	21.86	21.89	24.75	24.81
	EF	P2P	18.05	18.29	24.55	24.32	91.88	96.1	94.58	99.38	81.57	81.43	21.03	21.05	23.97	24.17
	Direct	P2P	11.65	11.64	27.22	27.2	54.55	54.44	32.86	33.04	84.76	85.02	22.10	22.12	25.02	25.04
	DDIM	MasaCtrl	28.38	28.04	22.17	22.16	106.62	106.0	86.97	86.90	79.67	80.08	21.16	21.16	23.96	24.03
	NT	MasaCtrl	–	32.75	–	22.1	–	105.64	–	102.09	–	79.64	–	21.19	–	24.18
	EF	MasaCtrl	–	26.26	–	22.81	–	109.82	–	102.50	–	80.11	–	21.9	–	25.21
	Direct	MasaCtrl	24.70	24.68	22.64	22.62	87.94	87.89	81.09	81.43	81.33	81.62	21.35	21.34	24.38	24.45

Table 1: Comparison on PIE Bench

5 Discussion

5.1 Limitations

Diffusion-based inversion methods require multiple steps during the inversion process. As a result, errors accumulate, and perfect reconstruction is not achieved. This point also affects editing. Additionally, the multiple steps involved in the inversion process and the subsequent reconstruction or editing process consume a significant amount of time.

Diffusion-based image editing methods share the drawbacks of diffusion models since they are based on them. Therefore, these methods also exhibit common limitations regarding the challenges faced by diffusion models in generation.

5.2 Future Works

Recently, InstaFlow(Liu et al. (2024)), a one-step image generation model using rectified flow, was proposed. If inversion is performed using such one-step models, it could potentially address the limitations mentioned above.

Furthermore, image generation models are continuously evolving, with advancements like SDXL and Flux building upon the existing stable diffusion framework. Therefore, it is anticipated that the limitations of image editing caused by the drawbacks of the original diffusion models will be overcome.

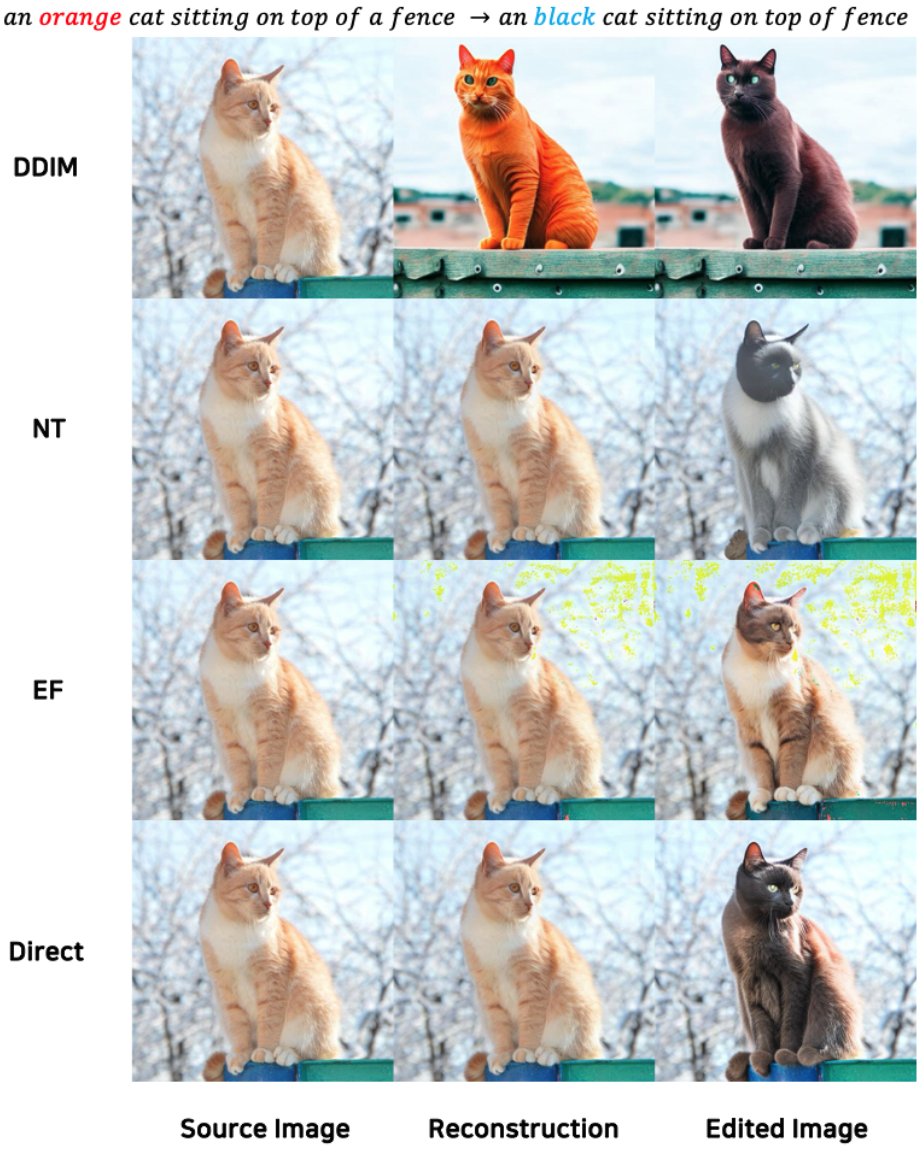


Figure 11: Comparison on Prompt-to-Prompt

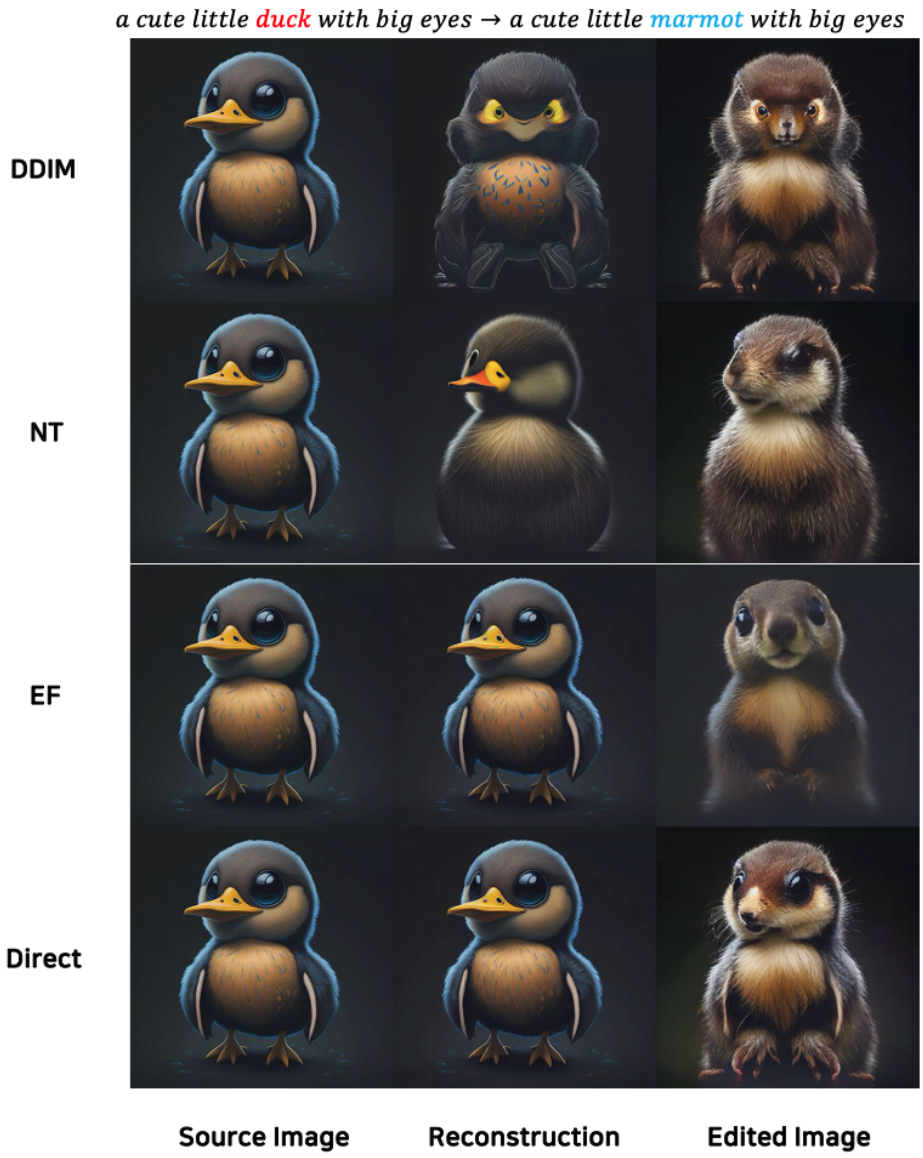


Figure 12: Comparison on Plug-and-Play

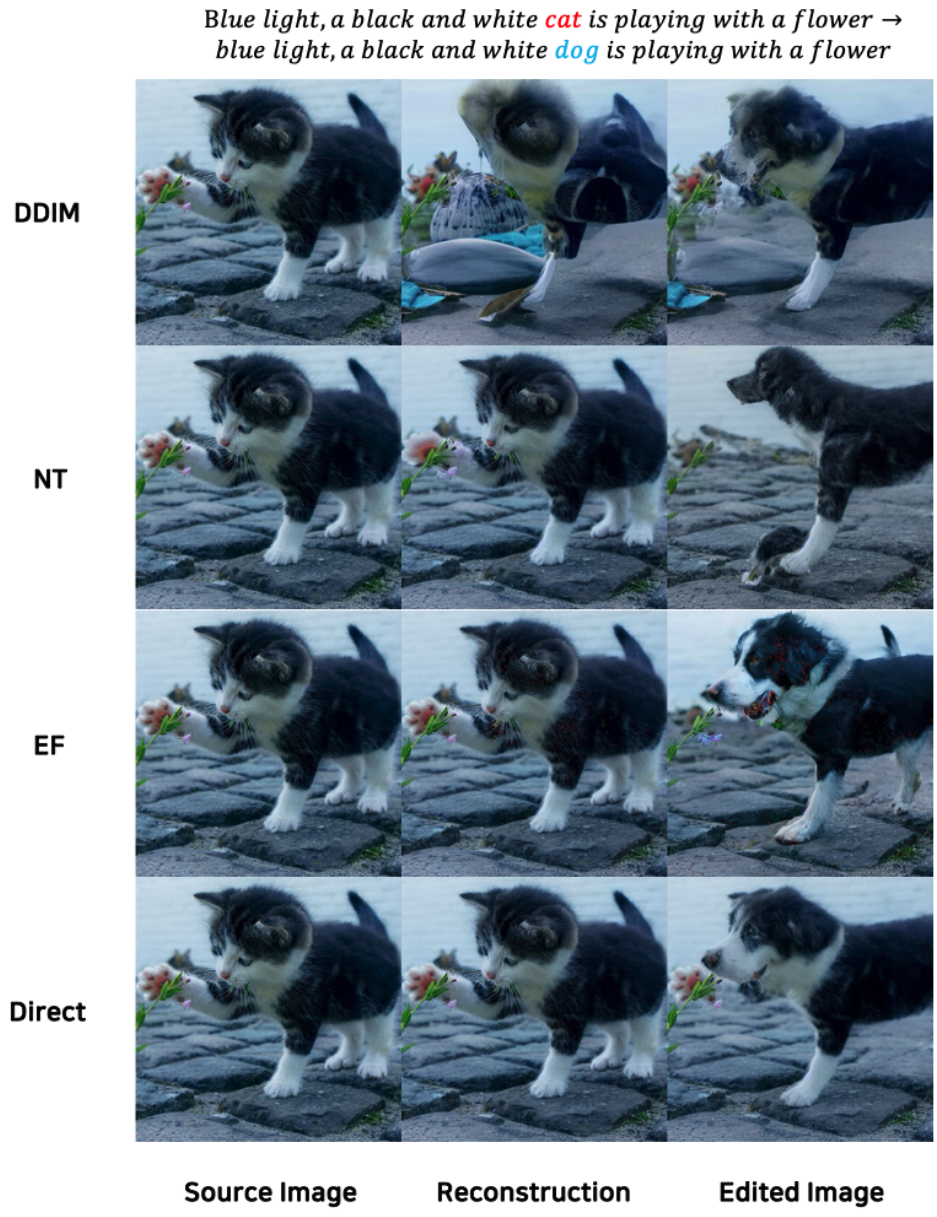


Figure 13: Comparison on MasaCtrl

References

Cao, M., Wang, X., Qi, Z., Shan, Y., Qie, X., and Zheng, Y. (2023). Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing.

Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., and Cohen-Or, D. (2022). Prompt-to-prompt image editing with cross attention control.

Huberman-Spiegelglas, I., Kulikov, V., and Michaeli, T. (2024). An edit friendly ddpm noise space: Inversion and manipulations.

Ju, X., Zeng, A., Bian, Y., Liu, S., and Xu, Q. (2023). Direct inversion: Boosting diffusion-based editing with 3 lines of code.

Liu, X., Zhang, X., Ma, J., Peng, J., and Liu, Q. (2024). Instaflow: One step is enough for high-quality diffusion-based text-to-image generation.

Mokady, R., Hertz, A., Aberman, K., Pritch, Y., and Cohen-Or, D. (2022). Null-text inversion for editing real images using guided diffusion models.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022). High-resolution image synthesis with latent diffusion models.

Song, J., Meng, C., and Ermon, S. (2022). Denoising diffusion implicit models.

Tumanyan, N., Bar-Tal, O., Bagon, S., and Dekel, T. (2022a). Splicing vit features for semantic appearance transfer.

Tumanyan, N., Geyer, M., Bagon, S., and Dekel, T. (2022b). Plug-and-play diffusion features for text-driven image-to-image translation.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric.