Enhancing Consistency of Flow-Based Image Editing through Kalman Control

Haozhe Chi [1] **Zhicheng Sun** [1] **Yang Jin** [1] **Yi Ma** [2] **Jing Wang** [2] **Yadong Mu** [1*]
[1]Peking University, [2]Central Media Technology Institute, Huawei

Abstract

Flow-based generative models have gained popularity for image generation and editing. For instruction-based image editing, it is critical to ensure that modifications are confined to the targeted regions. Yet existing methods often fail to maintain consistency in non-targeted regions between the original / edited images. Our primary contribution is to identify the cause of this limitation as the error accumulation across individual editing steps and to address it by incorporating the historical editing trajectory. Specifically, we formulate image editing as a control problem and leverage the Kalman filter to integrate the historical editing trajectory. Our proposed algorithm, dubbed Kalman-Edit, reuses early-stage details from the historical trajectory to enhance the structural consistency of the editing results. To speed up editing, we introduce a shortcut technique based on approximate vector field velocity estimation. Extensive experiments on several datasets demonstrate its superior performance compared to previous state-of-the-art methods.

1 Introduction

Diffusion models [44, 47, 17] have revolutionized the field of image and video generation, bringing unprecedented advancements. The recent development of the Diffusion Transformer [39] has enabled current diffusion-based models to scale up their parameters, achieving a higher level of generative capability. Notable works such as Stable Diffusion 3 [10] and Sora [5] demonstrate the remarkable potential of diffusion models in generating complex and intricate scenarios in both images and videos. Furthermore, researchers have explored the potential of diffusion models in editing tasks. For instance, DDIM inversion [45] progressively matches the target image distribution back to the original latent space, allowing for the generation of edited images by incorporating new prompt conditions. Additionally, some studies have modified attention maps [6, 18, 14] during the generation process to directly alter specific characteristics of objects. More recently, efforts have been made to integrate pretrained modules for more precise editing. For example, existing work [29] utilizes SAM [25] to extract specific regions requiring modification. Nevertheless, many diffusion-based editing methods still face challenges related to imprecise editing, primarily due to the non-linear nature of the generation trajectory.

Rectified flow [30, 31, 2], a special class of diffusion models, transforms random noise into the target distribution by linear interpolation between the two distributions. These models achieve distribution matching by constructing a velocity field, resulting in an efficient trajectory. Recent advancements have extended rectified flow models to image editing tasks. However, existing methods often struggle to balance structural consistency and editing quality effectively. For instance, Wang *et al*. [53] employ a new sampler to achieve more precise inversion and freeze specific attention values to preserve the overall semantics of edited images. While this approach maintains structural consistency, it sacrifices editing flexibility, as we will demonstrate later. Similarly, Rout *et al*. [43] utilize Linear Quadratic Regulator (LQR) control [21] to guide the editing process, and Kulikov *et al*. [27] propose using

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

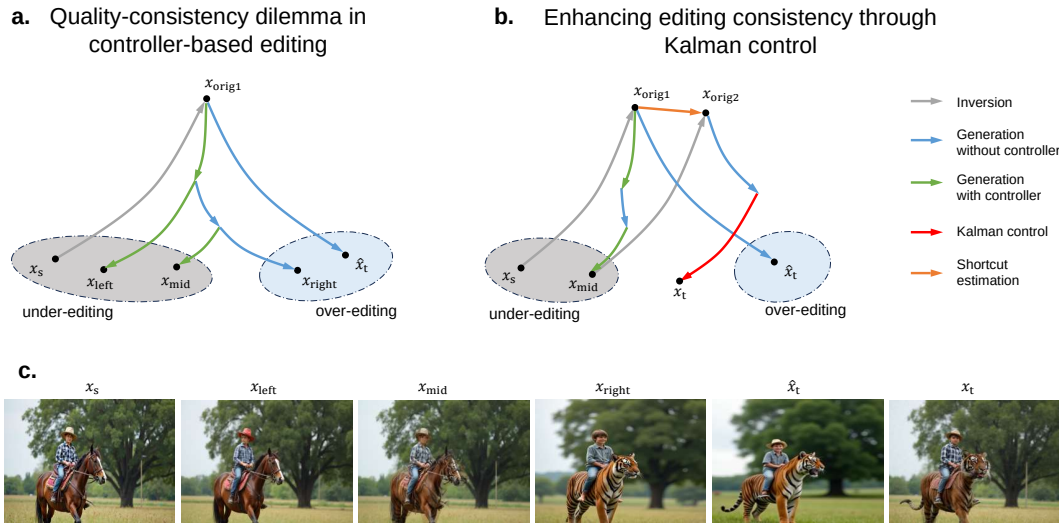

Figure 1: Conceptual illustration of a previously-developed controller-based method [43] (left) and our method (right). x_{s} represents original image, while $\hat{x}_{\mathrm{t}}$ and x_{t} represent edited images. The controller-based method derives an optimal control strategy for different stages of the generation process. However, as illustrated in the figure for x_{left}, excessive control leads to failed edits, whereas x_{right} demonstrates that insufficient control results in significant structural inconsistencies. Additionally, x_{mid} reveals that moderate control intensity produces ambiguous image semantics. Building on these key observations, we introduce Kalman control to further suppress irrelevant semantics while maintaining structural consistency. We also propose a shortcut estimation to eliminate the need for a second inversion process. The x_{t} figure underscores the effectiveness of our approach.

an estimated velocity transform to directly map original images to the target space. However, both methods exhibit limitations in editing quality, highlighting the need for further refinement in this area.

The primary technical contribution of this work lies in pinpointing the cause of inconsistency in image editing (*i.e.*, notable change over non-targeted image areas) as the accumulation of errors (*i.e.*, small inaccuracies can propagate forward and lead to significant deviations in the output) over individual editing steps and mitigating it by integrating the historical editing trajectory. We propose Kalman-Edit (along with its accelerated variant Kalman-Edit*), a method that addresses the Linear Quadratic Gaussian (LQG) problem [23] in optimal control theory. Our approach assumes that each velocity step in the LQG process incorporates a noise term, which can be estimated and reduced through the integration of a Kalman filter [22]. Specifically, we design an inversion algorithm that transforms the original image latent into a mixed (*i.e.*, encapsulates features from both the original and edited images). This mixed latent corresponds to a semantic blend between the source and target prompts. Subsequently, we invert the mixed latent and apply the Kalman filter to preserve structural information in the background. Our contributions are summarized as follows: (1) We introduce the Kalman filter approach to controller-based image editing. (2) Based on observations of direct LQR control, we propose a two-stage method that better unleashes the potential of Kalman control and achieves more flexible control. (3) Through experiments, our method shows high *structural consistency* and good *editing flexibility* on various editing tasks.

2 Related work

Image editing with diffusion models. Following advances of diffusion models [44, 47, 17], remarkable progress has been made in image editing via diffusion model inversion [46]. To address the time-consuming inversion computations and compounding estimation errors, various sampling and estimation strategies are developed [33, 9, 15, 35, 52, 36, 54, 20]. In addition, several work [42, 24, 28] introduce optimization methods to achieve better editing quality. Alternative editing algorithms include attention map controls [16, 50] and masking strategies [37, 8]. However, their editing quality and efficiency remain to be improved.

Image editing with rectified flow. Rectified flow [30, 31, 2] learns linear interpolation between source and target distributions to enable more efficient sampling of diffusion models. In addition to significant advances in rectified flow-based generation models [11, 4], its potential in image editing is gaining increasing attention [27, 38, 53, 43]. Kulikov *et al*. [27] and Patel *et al*. [38] take derivatives to produce more precise edits. Wang *et al*. [53] share attention scores in Transformer blocks to improve editing consistency. Rout *et al*. [43] use optimal controller to guide the generation trajectory. However, these efforts have not yet achieved a good balance between editing quality and consistency, as evidenced in our experiments later.

Image editing with control theory. Inspired by the connections between optimal control and SDEs [51, 48, 12], various sampling strategies in control theory are introduced to diffusion models for more controllable generation. Koo *et al*. [26] apply posterior sampling strategy for linear inverse problems. Rout *et al*. [43] utilize conditional sampling strategy for optimal control in the vector field. Our method advances this line of study with Kalman filter for more precise image editing.

3 Methodology

3.1 Preliminaries

Rectified flow. Flow-based generative models [49, 55, 30] aim to learn a probability path between the source distribution q_0 and the target distribution q_1 with a velocity field v. The flow starting from $x_0 \sim q_0$ to $x_1 \sim q_1$ is formulated as the following ordinary differential equation (ODE):

$$\frac{dx_t}{dt} = V(x_t, t), \tag{1}$$

where the velocity function V takes the timestep $t \in [0, 1]$ and the latent variable x_t as input. Rectified flow [30, 31, 2] is a special class of flow-based models defined by the linear interpolation between start point x_0 and endpoint x_1:

$$x_t = (1 - t)x_0 + tx_1. \tag{2}$$

Note that rectified flow is empirically able to generate high-resolution images with fewer sampling steps [32, 11], making it prevailing in image generation tasks.

Editing with rectified flow. To edit a given image x_{s} with a source prompt c_{s} and a target prompt c_{t}, a straightforward approach is to perform ODE inversion using Eq. (1). Let $V(x_t, t, c)$ denote the velocity function conditioned on prompt c, then editing follows a two-stage procedure:

$$\begin{aligned} \text{Inversion:} \quad & \frac{dx_t^{\mathrm{s}}}{dt} = V(x_t^{\mathrm{s}}, t, c_{\mathrm{s}}), \\ \text{Generation:} \quad & \frac{dx_t^{\mathrm{t}}}{dt} = V(x_t^{\mathrm{t}}, t, c_{\mathrm{t}}), \end{aligned} \tag{3}$$

where we first invert the image x_{s} to structured noise x_0, and then perform sampling with target prompt c_{t} for generation. However, directly applying this could lead to a significant deviation from the desired result (see $\hat{x}_{\mathrm{t}}$ in Fig. 1).

Optimal control. To improve editing controllability, an effective approach is to formulate it as optimal control [43]. Both inversion and generation processes can be viewed as a continuous-time linear system defined over t in $[0, 1]$.

$$\frac{dx_t}{dt} = Ax_t + Bu_t, \tag{4}$$

where x_t represents the state of the system, u serves as the controller of the system, and A, B are coefficient matrices. In optimal control theory, the objective is to determine an optimal controller u_t that guides the drift path to minimize the energy cost. A choice of the energy cost is a quadratic function, which corresponds to the Linear Quadratic Regulator (LQR) problem [21] as follows:

$$J_1 = x_1^\top F x_1 + \int_0^1 \left(x_t^\top Q x_t + u_t^\top R u_t\right) dt, \tag{5}$$

where F, Q and R are coefficient matrices of the system. As shown in previous work [43], solving the problem for rectified flow in Eq. (2) produces the optimal controller:

$$u_t = \frac{x_{\mathrm{s}} - x_t}{1 - t}, \tag{6}$$

where x_{s} represents the original image for reference. Although this controller reduces the overall error during editing, it fails to preserve detail consistency for background regions, as illustrated in Fig. 1.

3.2 Harnessing history with Kalman filter

To improve the consistency between the edited and original images, we make a key observation that the history sequence in inversion can effectively rectify the generation trajectory. As illustrated in Fig. 1, applying control signals to latents at different steps results in distinct semantic and detail-level changes. A comparison between x_{mid} and x_{right} reveals that latents from later generation steps are particularly effective in recovering fine details of the original image. Additionally, we observe that early inversion latents also contribute significantly to restoring structural information. To fully leverage these latents for improved consistency, it is essential to maintain a sequence of observed latents to guide and refine the editing process. **This approach naturally aligns with Kalman control, where observations are used to refine estimations and achieve more accurate outcomes.**

Inspired by optimal control theory, we reformulate the editing process as a Linear Quadratic Gaussian (LQG) problem [23] and introduce the Kalman control method [22]. To be specific, in LQG control theory, the system's evolution at each timestep includes noise terms, which can be mitigated by refining the trajectory using Kalman filter, *i.e.*,

$$\frac{dx_t}{dt} = Ax_t + Bu_t + w_t, \quad y_t = Hx_t + \sigma_t, \tag{7}$$

where y_t represents the measurement sequence, A, B and H are the coefficient matrices of the system, while w_t and σ_t denote the noise terms of system state estimations. With the application of the Kalman filter, the total cost function to be minimized is given by:

$$J_2 = \mathbb{E}\left[x_1^\top F x_1 + \int_0^1 \left(x_t^\top Q x_t + u_t^\top R u_t\right) dt\right], \tag{8}$$

where $\mathbb{E}$ refers to the expectation of the following terms. Importantly, expectation is necessary in this context, as our goal is to mitigate the impact of noise terms when minimizing the cost function J_2. To accurately estimate the expectation in J_2, we utilize the following Kalman filter equations:

$$\begin{aligned} K_k &= P_{k-1}H^T(HP_{k-1}H^T + T)^{-1}, \\ x_k &= Ax_{k-1} + Bu_k + K_k(y_k - Hx_{k-1}), \\ P_k &= (I - K_kH)P_{k-1}. \end{aligned} \tag{9}$$

In specific, given the covariance matrix P_{k-1}, state x_{k-1}, noise T, controller u_k and measurement y_k, it first computes the Kalman gain K_k. These values are then used recursively to obtain the updated terms x_k and P_k. Consequently, all the filtered latents x_k can be computed using these equations. Here, the noise term is estimated and corrected by multiplying the Kalman gain K_k with the innovation term $(y_k - Hx_{k-1})$. Through proper derivation and analysis, we arrive at the following proposition (see Appendix A for details):

Proposition 1. *With proper initialization of system coefficients (P_0, H, T), the Kalman control process shown in Eq.* (9) *converges.*

As shown in Fig. 1, achieving both high generation quality and structural consistency presents a significant challenge in current controller-based editing methods. Addressing this issue requires establishing an effective measurement sequence. To sufficiently incorporate historical information, we carefully construct the measurement sequence $\{y_k\}_{k=1}^{l}$ of length l, integrating latents from the inversion process:

$$y_k = \mathrm{Inv}(x_{\mathrm{s}}, k), \;\; 1 <= k <= l, \tag{10}$$

where $\mathrm{Inv}(x, k)$ denotes the k^{th} latent obtained during the inversion process described in Eq. (3), and the resulting measurement sequence $\{y_k\}_{k=1}^{l}$ are incorporated in the computation of the Kalman filter to control the generation process.

Algorithm 1 Kalman-Edit and Kalman-Edit*

Input: original image x_{s}, total step N, source prompt c_{s}, target prompt c_{t}, Inversion function $\text{Inv}(\cdot, \cdot)$
 # Stage 1: editing with optimal controller
$x_{\text{orig1}} \leftarrow \text{Inv}(x_{\text{s}}, N)$
Generate x_{mid} from x_{orig1} with controller in Eq. (6)
 # Stage 2: editing with Kalman filter
if Kalman-Edit **then**
$x_{\text{orig2}} \leftarrow \text{Inv}(x_{\text{mid}}, N)$
else if Kalman-Edit* **then**
$x_{\text{orig2}} \leftarrow x_{\text{orig1}} + x_{\text{mid}} - x_{\text{s}}$ ▷ shortcut in Section 3.4
Compute measurement $\{y_k\}_{k=1}^{l}$ according to Eq. (11)
Generate x_{t} from x_{orig2} with Kalman filter in Eq. (12)
Output: x_{t}

Following this formulation, the entire Kalman control process proceeds as follows: First, we construct the measurement sequence using Eq. (10). Next, we iteratively update the Kalman gain K_k and compute the corresponding innovation term $y_k - Hx_{k-1}$ to regulate the generation sequence x_k using Eq. (9). This way, the generation process is conditioned on measurement from the original image before editing, thus preserving more details.

3.3 Proposed algorithm: Kalman-Edit

Applying Kalman control to flow-based editing presents two primary challenges: (1) As shown in Fig. 1, determining the appropriate timesteps for applying the Kalman filter is challenging. If too many timesteps undergo the filtering process, editability may be compromised, leading to edited results that fail to faithfully reflect the target prompt. (2) Artifacts and blurring often arise when directly applying Kalman control through a single inversion and forward process. This occurs because the filtering equations can inadvertently guide the trajectory toward an intermediate state between the original image distribution and the target distribution. Such a middle state often corresponds to poor image quality, resulting in undesirable visual artifacts.

Two-stage image editing. To address both challenges, we propose a two-stage algorithm for generating high-quality edited images using Kalman control, as outlined in Algorithm 1. In the first stage, our goal is to generate an intermediate latent that encapsulates the semantics of both the original and target prompts. Next, we generate the intermediate latent x_{mid} by applying the controller at appropriate timesteps. In the second stage, we first invert x_{mid} to obtain the second original latent x_{orig2}. We then apply the Kalman filter, as described in Eq. (9) to the generation process in order to filter out undesired semantic information irrelevant to the target prompt, and reintroduce history information from the original image x_{s}. This approach results in a target image x_{t} that retains the structural information of x_{s} while adhering to the target prompt c_{t}, as desired.

We further adapt the measurement sequence construction to the two-stage editing scheme. Since the original image x_{s} and edited intermediate image x_{mid} both contain desired information (*i.e.*, the original detail and target semantics), we curate the measurement sequence by collecting inversion trajectories from both images x_{s} and x_{mid}. Specifically, we introduce a hyperparameter δ and define the measurement sequences $\{y_k\}_{k=1}^{\delta-1}$ and $\{y_k\}_{k=\delta}^{l}$ to capture structural information from the first and second inversion processes, respectively. They are computed by the following inversion processes:

$$\begin{aligned} y_k &= \text{Inv}(x_{\text{s}}, k), \;\; 1 <= k <= \delta - 1, \\ y_k &= \text{Inv}(x_{\text{mid}}, k), \;\; \delta <= k <= l. \end{aligned} \tag{11}$$

By incorporating the two inversion sequences and integrating them into our measurement sequence, it enables the recovery of diverse structural and semantic information through Kalman control. Note that this construction scheme is also compatible with our accelerated version without two inversion passes (Section 3.4), in which case we simply set the δ value to $l + 1$.

Kalman filter phases. Another crucial consideration in our approach is selecting the appropriate timesteps for applying both the controller and the Kalman filter effectively. A key observation is that rectified flow models exhibit behavior similar to traditional diffusion models. As noted in prior

Table 1: Quantitative comparison on SFHQ datasets among flow-based editing models. See the main text for the definitions of the performance metrics. The highest value in each column is highlighted in bold.

	Face Rec. ↓	CLIP-I ↑	LPIPS ↓	CLIP-T ↑	DreamSim ↓
RF-Edit	0.4051	0.8984	0.1562	0.2910	0.1591
RF-Inversion	0.4325	0.8927	0.1720	**0.3012**	0.1889
FlowEdit	0.4856	0.8579	0.1687	0.2905	0.2375
FlowChef	0.4013	0.8769	0.1401	0.2832	0.1487
Kalman-Edit	**0.3958**	**0.9167**	**0.1332**	0.2921	**0.1408**
Kalman-Edit*	0.4696	0.8871	0.1892	0.2936	0.2227

Table 2: Quantitative comparison on HQ datasets among flow-based editing models.

	CLIP-T ↑	CLIP-I ↑	LPIPS ↓	DINO ↑	DreamSim ↓
RF-Edit	0.1842	**0.9141**	0.2383	**0.8197**	0.1492
RF-Inversion	0.1825	0.9033	0.3074	0.7963	0.1662
FlowEdit	0.1877	0.8813	0.2846	0.7467	0.2238
FlowChef	0.1928	0.9023	0.2925	0.8053	0.1537
Kalman-Edit	**0.1943**	0.9062	**0.2345**	0.7929	**0.1353**
Kalman-Edit*	0.1870	0.8696	0.3615	0.7123	0.2276

work [56], image generation in diffusion models progresses through distinct phases: the early stages primarily establish semantic content, while finer details are refined in the later stages. This pattern is also evident in rectified flow models. Applying the controller across too many timesteps during the refinement phase would overly constrain the output distribution, making it too similar to the original image and thereby limiting effective editing. To ensure high-quality editing, we apply the controller primarily during the early phase of generation when semantic information is being formed. Likewise, to maximize the effectiveness of Kalman filtering, we apply it during the refinement stage, where it can best aid in recovering structural details.

Implementation for flow models. In the above derivation we consider the evolution of x_k. However, flow-based generative models are often parameterized by velocity instead of x-prediction. Therefore, in practice, we apply the Kalman control updates in the following manner:

$$\begin{aligned} x_{t_{k+1}} &= x_{t_k} + (t_{k+1} - t_k) v'_{t_k}, \\ v'_{t_k} &= \mu v_{t_k} + \lambda u_{t_k} + (1 - \mu - \lambda) K_k (y_k - H x_{t_k}). \end{aligned} \tag{12}$$

where $v_{t_k} = V(x_{t_k}, t_k)$ is the predicted velocity, u_{t_k} is the optimal controller given by Eq. (6), and μ and λ are coefficients balancing their contributions. See Algorithm 2 for complete update details.

3.4 Kalman-Edit*: acceleration with shortcut

To avoid the computational cost of performing the inversion process twice, we accelerate the estimation of x_{orig2} by leveraging the parallelogram law of vectors. This approach is justified in many editing scenarios, such as local area modifications, where the difference between x_{mid} and x_{s} is minimal. Under the assumption that both source and target latents are normalized and exhibit similar variance, a first-order approximation in the vector field yields a sufficiently accurate estimate of x_{orig2}:

$$x_{\text{orig2}} = x_{\text{orig1}} + (x_{\text{mid}} - x_s). \tag{13}$$

This avoids the second inversion process and turns out to be efficient through experiments.

4 Experiments

4.1 Evaluation protocols

Datasets. The experimental evaluation is conducted across four widely used datasets: SFHQ [3], HQ [19], ZONE [29] and DIV2K [1]. The SFHQ dataset consists of 425,000 high-quality human facial images. The HQ dataset[2] is a synthetic editing benchmark containing approximately 200,000

[2] https://thefllood.github.io/HQEdit_web/

Table 3: Quantitative evaluations on ZONE and DIV2K datasets. See main text for more details about the performance metrics.

	CLIP-T ↑	CLIP-I ↑	LPIPS ↓	DINO ↑	DreamSim ↓
SDEdit	0.2754	0.9264	0.1908	0.8547	0.1148
P2P	0.2773	0.9209	0.1568	0.8186	0.1519
MasaCtrl	0.3103	0.9179	0.1580	0.8397	0.1635
DDPM-Inv	0.2847	0.9063	0.1734	0.8215	0.1742
RF-Edit	0.2964	0.8926	0.2039	0.7986	0.1776
RF-Inversion	0.2844	0.8919	0.2491	0.7974	0.1536
FlowEdit	0.3096	0.8687	0.2269	0.7671	0.2319
FlowChef	0.3025	0.8831	0.2563	0.7456	0.2471
Kalman-Edit	0.2957	**0.9492**	**0.1407**	**0.9141**	**0.0793**
Kalman-Edit*	**0.3220**	0.8986	0.2488	0.8237	0.1454

Table 4: Comparison of CLIP-I (left) and LPIPS scores(right) for different Kalman filter strengths and steps evaluated on the ZONE dataset.

Filter strength / Added steps	15-18	15-22	15-27
0.1	0.8770	0.9219	**0.9346**
0.2	0.9043	0.9282	0.9226
0.3	0.9014	0.8921	0.9079

Filter strength / Added steps	15-18	15-22	15-27
0.1	0.2433	0.2035	**0.1487**
0.2	0.2325	0.1944	0.1521
0.3	0.2284	0.2008	0.1886

images generated through DALL-E 3 and GPT-4V. The ZONE dataset features 100 images designed for object insertion, editing, and removal tasks. DIV2K serves as a standard benchmark for super-resolution tasks, comprising 1,000 real-world images. Due to practical computational constraints, we evaluate our approach on the subsets of these benchmarks, including 1,200 images from SFHQ, 320 images from HQ, and 105 images from ZONE and DIV2K. Following the setting of Rout *et al*. [43], we employ an instruction prompt that adds glasses to all face images in the SFHQ dataset.

Metrics. Six metrics are employed to evaluate both editing quality and consistency. For editing quality, CLIP-T [40] is adopted to measure the semantic adherence between edited image and input prompts. Meanwhile, we also use Face Rec. metric to quantify identity similarity on the face-specific SFHQ dataset. Regarding editing consistency, CLIP-I and DINO [7] measure high-level semantic similarity, while LPIPS [57] captures low-level similarity such as pixel-level details. Moreover, Dreamsim [13] is responsible for evaluating mid-level similarity, including image layout.

Baselines. Our method is compared against eight image editing baselines spanning rectified flow and diffusion models. For rectified flow-based editing, we consider RF-Edit [53], RF-Inversion [43], FlowEdit [27] and FlowChef [38]. For diffusion-based editing counterparts, we compare against SDEdit [34], P2P [16], MasaCtrl [6] and DDPM-Inv [18]. To ensure a fair comparison, we follow the original recommended hyperparameter settings (*e.g*., where to add the controller) for all baselines.

Implementation details. For flow-based editing, we use FLUX.1 dev [4] with $N = 28$ sampling steps. For diffusion-based editing, we use Stable Diffusion 1.4 [41] with $N = 50$ sampling steps. The measurement length l is set to 14, and δ is 6 by default. More details are provided in Appendix B.

4.2 Experimental results

Quantitative results. As demonstrated in Table 1 and Table 2, our method maintains high facial similarity after editing on the SFHQ dataset and effectively adheres to complex editing prompts on the HQ dataset. The basic version of Kalman-Edit demonstrates strong performance, achieving state-of-the-art results in most metrics. And Kalman-Edit* (the accelerated variant) produces comparable results on SFHQ and HQ datasets. This discrepancy can be attributed to the fact that Kalman-Edit* directly estimates the second original latent, which may result in the loss of low-level structural information. As shown in Table 3, Kalman-Edit and Kalman-Edit* also outperform the four baseline methods across most metrics on ZONE and DIV2K datasets, showing the effectiveness of our method.

Ablation analysis. In this section, we conduct ablation experiments to determine the optimal filter strength and steps at which the filter is applied, as well as to highlight the importance of Kalman control in structural preservation. Following previous research [56], we observe that the generation process of rectified flow can also be divided into two stages: semantic formation and refinement.

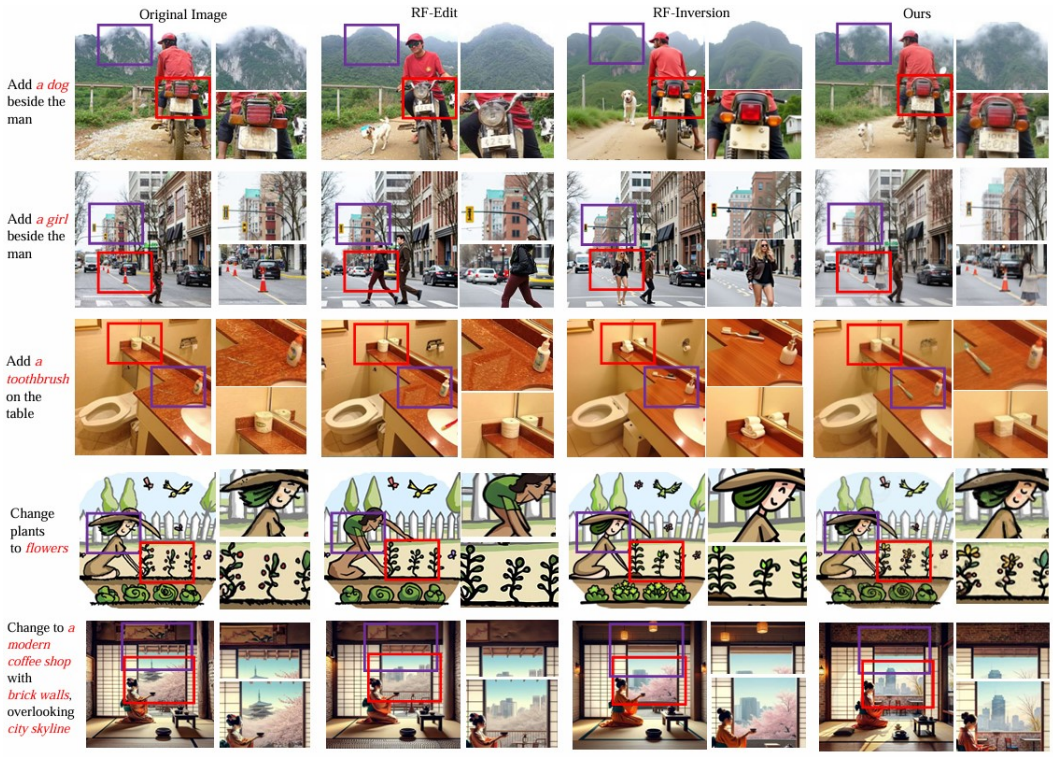

Figure 2: Comparison of structure preservation and editing quality on ZONE and DIV2K dataset. The top three rows demonstrate that our method effectively preserves local details, ensuring strong structural consistency. Meanwhile, the bottom two rows highlight its ability to adhere to the target prompt, accurately incorporating elements such as flowers and brick walls. They illustrate that our method achieves both better structure preservation and editing quality. Better viewing when enlarged.

Table 5: Ablation study of Kalman filter in LQR-based control method on ZONE dataset.

Metrics	CLIP-T ↑	CLIP-I ↑	LPIPS ↓
w/o Kalman filter	0.2952	0.8784	0.2695
w/ Kalman filter	**0.2961**	**0.9346**	**0.1487**

Since the Kalman filter is designed to refine structural details, we apply Kalman control during the refinement stage, which corresponds to the latter half of the generation steps. As shown in Tables 4 and 5, the best CLIP-I and LPIPS scores are achieved when using a relatively small filter strength combined with a large number of added steps. Also, from each row of Table 4 we conclude that more steps of Kalman control help recover more pixel-level details. From each column of Table 4, we observe that small strength helps to generate structural details more efficiently and smoothly, while large strength causes significant performance drop. This indicates that a longer measurement sequence enhances structural details, while lower filter strength steers the trajectory toward a higher-quality distribution. Overall, our method is effective across a fairly wide range of hyperparameters.

To validate the effectiveness of our approach, we conduct ablation experiments to assess the impact of Kalman control. Specifically, we compare our method against the RF-Inversion baseline without the Kalman control. As shown in Table 5, our method achieves higher CLIP-I and LPIPS scores, indicating superior structural consistency. Furthermore, the CLIP-T metric confirms that our approach maintains high editing quality, demonstrating its advantages in both structure and semantics.

Qualitative results. Our qualitative comparison results are presented in Figures 2 and 3, showcasing both real-world and synthetic images to demonstrate our method's ability to maintain high structural consistency and editing quality. In particular, Fig. 2 compares the structural preservation capabilities of our method against baseline approaches. As shown, our method retains more local details than the baselines. For instance, in the second row of Fig. 2, only our method successfully recovers the

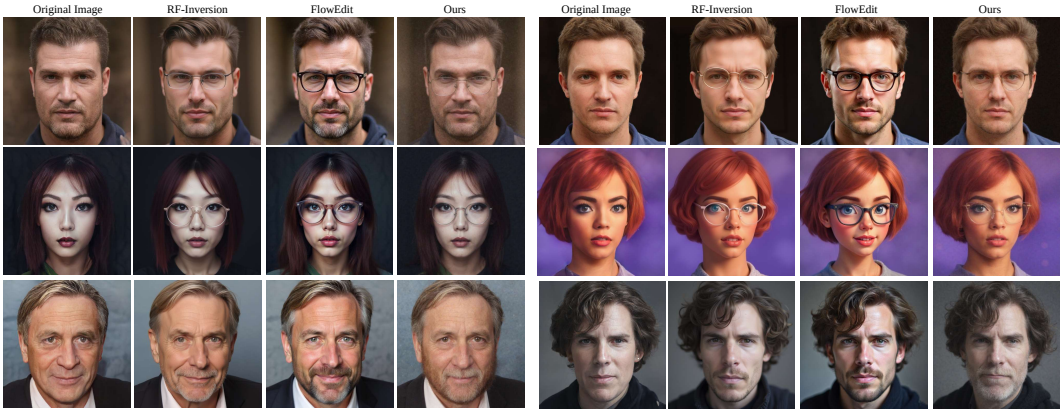

Figure 3: Qualitative results obtained using human face images from SFHQ dataset. The first two rows are edited with target prompt “A person wearing glasses" and the last row is edited with target prompt “A person with beard". The edited results are compared with baseline methods, demonstrating our approach’s superior ability to preserve the structural details of human faces (*i.e.*, our method produces edited images of higher fidelity, recovering facial features more accurately than baseline approaches).

traffic cones on the road. Similarly, in the last row, our edited results accurately incorporate multiple elements specified in the target prompt, demonstrating our method’s flexibility in handling longer and more complex prompts. In contrast, baseline methods struggle to recover structural details while maintaining overall editing quality. Compared to the baselines, our method significantly outperforms in preserving the structure of the original images while effectively editing the desired areas. Furthermore, to demonstrate the effectiveness of our method in preserving structural information across various image structures, we present editing results on human faces in Fig. 3. Specifically, we first modify the images by adding glasses to each face, following the approach in [43]. To further validate the generality of our method, we then edit the images by adding beards to each face. The edited results maintain a high degree of structural similarity to the original faces, highlighting our method’s ability to preserve facial structural consistency. For additional qualitative results on tasks such as style transfer and scene editing, please refer to Appendix D.

Computational efficiency. Figure 4 compares the time usage of our approach against flow-based editing methods. Since RF-Edit takes much time to load models, we report the time cost from the starting point of inversion to the time editing is completed to ensure the fairness. From the comparison, we observe that FlowChef has the fastest editing speed. Our proposed Kalman-Edit* is slower than RF-Inversion and FlowChef, but faster than all other baselines. This indicates that our algorithm strikes a competitive balance between editing performance and efficiency.

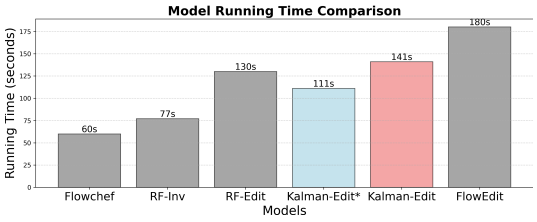

Figure 4: Running time comparison of different flow-based methods highlighting efficiency.

Failure cases. For tasks such as object removal, our method requires hyperparameter tuning to achieve optimal results. This necessity arises from our design that leverages historical inversion latents as the measurement sequence to rectify the final generation process. As a consequence, the original details and residual artifacts from the original image tend to appear in edited images at the default control strength settings, as illustrated by the boat example in Appendix G.

Style transfer and Scene editing. To further evaluate our method on a wider range of tasks, such as style transfer and scene editing, we present additional qualitative results. As illustrated in Fig. 5, our method demonstrates strong capability in transforming an old rusty room with a cement floor into a simple and elegant room with a wooden floor, converting a castle into a Disney-style cartoon scene, transforming a house into a white church with stained-glass windows, and replacing a beach background with snow-covered mountains. Moreover, our method effectively preserves structural consistency, maintaining both local details and the overall spatial layout. For example,

[Simple and elegant style] an empty room with wooden floor and a view of the *[impressionism beach]*

Change the *beach* in the background to a *snow mountain*

Change the castle to *Disney cartoon style* and add *a bridge*

[Stained glass window of] a white *[cartoon]* church sits on a hill in a field

Figure 5: High-resolution qualitative results of style transfer and scene editing tasks. The left image is the original input and the right one is the edited result. As illustrated above, our method achieves precise prompt adherence and delivers high-quality editing outcomes.

the edited result in the top case of Fig. 5 retains the room's geometric structure, while in the car case, it accurately preserves the road direction and car position. These results demonstrate that our method can successfully handle a broad variety of complex tasks and produce high-quality, structure-preserving outputs. Additional discussions on broader tasks and high-resolution visualization results are provided in Appendix D and Appendix E.

5 Conclusion

In this paper, we propose Kalman-Edit, a training-free flow-based image editing method based on optimal control theory. Existing rectified flow editing methods struggle to balance structural consistency and editing quality. To address this challenge, we derive fundamental equations from Linear Quadratic Gaussian (LQG) control, effectively utilizing history information in the editing trajectory with a Kalman filter-based algorithm. Through extensive experiments, we demonstrate that Kalman-Edit achieves superior structural consistency while maintaining high editing quality.

Acknowledgement: This work is supported by a grant from Huawei (No. TC20240821013_01).

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135, 2017. 6

[2] Michael Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations*, 2023. 1, 3

[3] David Beniaguev. Synthetic faces high quality (sfhq) dataset, 2022. 6

[4] Black Forest Labs. FLUX. `https://github.com/black-forest-labs/flux`, 2024. 3, 7, 22

[5] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. `https://openai.com/research/video-generation-models-as-world-simulators`, 2024. 1

[6] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang 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 (ICCV)*, pages 22560–22570, October 2023. 1, 7

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 7

[8] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. In *ICLR 2023 (Eleventh International Conference on Learning Representations)*, 2023. 2

[9] Adham Elarabawy, Harish Kamath, and Samuel Denton. Direct inversion: Optimization-free text-driven real image editing with diffusion models. *arXiv preprint arXiv:2211.07825*, 2022. 2

[10] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning*, 2024. 1

[11] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, pages 12606–12633, 2024. 3

[12] Wendell H Fleming and Raymond W Rishel. *Deterministic and stochastic optimal control*, volume 1. Springer Science & Business Media, 1975. 3

[13] Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. *Advances in Neural Information Processing Systems*, 36, 2024. 7

[14] Michal Geyer, Omer Bar-Tal, Shai Bagon, and Tali Dekel. Tokenflow: Consistent diffusion features for consistent video editing, 2023. 1

[15] Chunming He, Kai Li, Yachao Zhang, Longxiang Tang, Yulun Zhang, Zhenhua Guo, and Xiu Li. Camouflaged object detection with feature decomposition and edge reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22046–22055, 2023. 2

[16] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-or. Prompt-to-prompt image editing with cross-attention control. In *The Eleventh International Conference on Learning Representations*, 2022. 2, 7, 22

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851, 2020. 1, 2

[18] Inbar Huberman-Spiegelglas, Vladimir Kulikov, and Tomer Michaeli. An edit friendly ddpm noise space: Inversion and manipulations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12469–12478, June 2024. 1, 7

[19] Mude Hui, Siwei Yang, Bingchen Zhao, Yichun Shi, Heng Wang, Peng Wang, Yuyin Zhou, and Cihang Xie. Hq-edit: A high-quality dataset for instruction-based image editing, 2024. 6

[20] Xuan Ju, Ailing Zeng, Chenchen Zhao, Jianan Wang, Lei Zhang, and Qiang Xu. Humansd: A native skeleton-guided diffusion model for human image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15988–15998, 2023. 2

[21] RE Kalman. Contributions to the theory of optimal control. *Boletin Sociedad Matematica Mexicana*, 5:102–109, 1960. 1, 3

[22] R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960. 2, 4

[23] RE Kalman and RS Bucy. New results in linear filtering and prediction theory. *Journal of Basic Engineering*, 83(1):95–108, 1961. 2, 4

[24] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2435, 2022. 2

[25] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 1

[26] Juil Koo, Chanho Park, and Minhyuk Sung. Posterior distillation sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13352–13361, 2024. 3

[27] Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. Flowedit: Inversion-free text-based editing using pre-trained flow models. *arXiv preprint arXiv:2412.08629*, 2024. 1, 3, 7

[28] Gihyun Kwon and Jong Chul Ye. Diffusion-based image translation using disentangled style and content representation. In *The Eleventh International Conference on Learning Representations*, 2023. 2

[29] Shanglin Li, Bohan Zeng, Yutang Feng, Sicheng Gao, Xiuhui Liu, Jiaming Liu, Lin Li, Xu Tang, Yao Hu, Jianzhuang Liu, et al. Zone: Zero-shot instruction-guided local editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6254–6263, 2024. 1, 6

[30] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. 1, 3

[31] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023. 1, 3

[32] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, et al. InstaFlow: One step is enough for high-quality diffusion-based text-to-image generation. In *International Conference on Learning Representations*, 2024. 3

[33] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022. 2

[34] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. 7, 22

[35] Daiki Miyake, Akihiro Iohara, Yu Saito, and Toshiyuki Tanaka. Negative-prompt inversion: Fast image inversion for editing with text-guided diffusion models. *arXiv preprint arXiv:2305.16807*, 2023. 2

[36] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023. 2

[37] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning*, pages 16784–16804. PMLR, 2022. 2

[38] Maitreya Patel, Song Wen, Dimitris N Metaxas, and Yezhou Yang. Steering rectified flow models in the vector field for controlled image generation. *arXiv preprint arXiv:2412.00100*, 2024. 3, 7

[39] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 1

[40] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 7

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 7, 22

[42] Litu Rout, Yujia Chen, Abhishek Kumar, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Beyond first-order tweedie: Solving inverse problems using latent diffusion. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2

[43] Litu Rout, Yujia Chen, Nataniel Ruiz, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Semantic image inversion and editing using rectified stochastic differential equations. In *International Conference on Learning Representations*, 2025. 1, 2, 3, 4, 7, 9, 24

[44] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265, 2015. 1, 2

[45] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 1

[46] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 2

[47] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pages 11918–11930, 2019. 1, 2

[48] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. 3

[49] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. 3

[50] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1921–1930, 2023. 2

[51] Belinda Tzen and Maxim Raginsky. Theoretical guarantees for sampling and inference in generative models with latent diffusions. In *Conference on Learning Theory*, pages 3084–3114. PMLR, 2019. 3

[52] Bram Wallace, Akash Gokul, and Nikhil Naik. Edict: Exact diffusion inversion via coupled transformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22532–22541, 2023. 2

[53] Jiangshan Wang, Junfu Pu, Zhongang Qi, Jiayi Guo, Yue Ma, Nisha Huang, Yuxin Chen, Xiu Li, and Ying Shan. Taming rectified flow for inversion and editing. *arXiv preprint arXiv:2411.04746*, 2024. 1, 3, 7

[54] Chen Henry Wu and Fernando De la Torre. A latent space of stochastic diffusion models for zero-shot image editing and guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7378–7387, 2023. 2

[55] Yilun Xu, Ziming Liu, Max Tegmark, and Tommi Jaakkola. Poisson flow generative models. In *Advances in Neural Information Processing Systems*, pages 16782–16795, 2022. 3

[56] Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23174–23184, 2023. 6, 7

[57] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 7

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We summarize our contributions correctly in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations as failure cases in the experiment section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We have provided detailed proof of theoretical results in Appendix A.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have disclosed all necessary settings in Appendices B and C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We have released the link to our code in Appendix G.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Experimental setting could be found in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The datasets we use are large, and conducting a comprehensive statistical analysis over the entire datasets would be prohibitively time-consuming and computationally expensive.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the information of compute resources in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research follows the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impacts in Appendix G.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We would include safeguards in our released data or models.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly cited all relevant works and follow the public licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: Our core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Additional theoretical results: Proof of convergence for Kalman control iteration

We show that the error covariance sequence P_k decreases at each iteration, ensuring convergence. According to the Kalman iteration in Eq. (9), the update rule is given by:

$$P_k = (I - K_k H)P_{k-1}, \tag{14}$$

$$K_k = P_{k-1}H^T S_k^{-1}, \tag{15}$$

$$S_k = HP_{k-1}H^T + T. \tag{16}$$

Substituting Eq. (15) into Eq. (14), we obtain:

$$P_k = P_{k-1} - P_{k-1}H^T S_k^{-1} HP_{k-1}. \tag{17}$$

Defining $M = P_{k-1}H^T S_k^{-1} HP_{k-1}$, we need to show M is positive semidefinite to conclude $P_k \preceq P_{k-1}$.

Assuming P_0 is positive definite and P_{k-1} remains positive definite, and also that H is positive semidefinite and T is positive definite with proper initialization, we first show that S_k^{-1} is positive definite. According to Eq. (16), since P_{k-1} is positive definite, $HP_{k-1}H^T$ is positive semidefinite. As T is positive definite, their sum S_k is also positive definite, implying S_k^{-1} is positive definite.

To establish $M \succeq 0$, for any nonzero vector x, we compute

$$x^T Mx = x^T P_{k-1}H^T S_k^{-1} HP_{k-1}x. \tag{18}$$

Letting $y = HP_{k-1}x$, this simplifies to

$$x^T Mx = y^T S_k^{-1} y. \tag{19}$$

Since S_k^{-1} is positive definite, $y^T S_k^{-1} y \geq 0$, proving $M \succeq 0$. This guarantees $P_k \preceq P_{k-1}$, showing that the error covariance sequence decreases. If the system is stable, P_k converges to a steady-state value.

Table 6: **Experiment hyperparameters.**

	Steps	Base model	CFG scale	Control strength
SDEdit	50	SD 1.4	4.0	-
P2P	50	SD 1.4	7.5	-
MasaCtrl	50	SD 1.4	7.5	-
DDPM-Inv	50	SD 1.4	9	-
RF-Edit	28	FLUX.1 dev	2	-
RF-Inversion	28	FLUX.1 dev	3.5	(0.7, 0.95)
FlowEdit	28	FLUX.1 dev	(1.5,5.5)	-
FlowChef	28	FLUX.1 dev	2	-
Ours	28	FLUX.1 dev	3.5	0.95

B Additional details for experiment settings

First, we show the detailed hyperparameters setting for all baseline methods in Table 6. As demonstrated, for diffusion-based editing methods SDEdit [34] and P2P [16], we test both of them with 50 sampling steps and use StabeDiffusion 1.4 [41] as the base model. For all flow-based editing methods, we test all of them with 28 steps and use FLUX.1 dev [4] as base model. The timesteps are determined by Euler discrete scheduler. We also summarize the CFG scale and control strength in the table. These CFG values follow the default settings recommended in each baseline's implementation.

Next, we explain the hyperparameter settings for our proposed method. We set the steps to add the Kalman filter l to 14, which is half of the total steps. And we set the steps L to be the later half steps of the generation (i.e., steps 15 to 28). The hyperparameter δ determining the two types of measurement sequences is 6 by default. The coefficient hyperparameters μ and λ are set to 0.7 and 0.1 by default.

We also set the matrices A, B, and H in Eq. (9) to be identity matrices, which is computationally efficient and has proven to be effective in experiments. We also set the initial covariance matrix P_0 to be an identity matrix. The noise term is approximated by an identity matrix multiplied by a small coefficient 0.1 or 0.01. The complete procedure of our algorithm is given in Algorithm 2. All of our experiments are conducted on a single NVIDIA A40 GPU.

Algorithm 2 Detailed procedure of Kalman-Edit

Input: original image x_{s}, timesteps $\{t_i\}_{i=0}^{T}$, source prompt c_s, target prompt c_t, strength coefficients $\{\lambda\}_{i=0}^{T}$ and $\{\mu\}_{i=0}^{T}$, step sets S_1, S_2 for adding controller, and L for adding Kalman filter (with $|L| = l$).

Init: $x_{t_N} \leftarrow x_{\mathrm{s}}, \quad M \leftarrow \emptyset$

/* Phase 1: Backward Denoising with Source Prompt */
for $i = N$ **to** 1 **do**
  $v_{t_i} \leftarrow V_\theta(x_{t_i}, t_i, c_s)$
  if $t_i \in S_1$ **then**
    $x_{t_{i-1}} \leftarrow x_{t_i} + (t_{i-1} - t_i)\Big(\lambda\, v_{t_i} + (1-\lambda)\, u_{t_i}\Big)$
  else
    $x_{t_{i-1}} \leftarrow x_{t_i} + (t_{i-1} - t_i)\, v_{t_i}$
  if $t_i \in L$ **then**
    Add t_i to M ▷ Construct measurement sequence $\{y_i\}_{i=0}^{l}$
$x_{\text{orig1}} \leftarrow x_{t_0}$

/* Phase 2: Forward Denoising with Target Prompt */
for $i = 0$ **to** $N-1$ **do**
  $v_{t_i} \leftarrow -V_\theta(x_{t_i}, t_i, c_t)$
  if $t_i \in S_1$ **then**
    $x_{t_{i+1}} \leftarrow x_{t_i} + (t_i - t_{i+1})\Big(\lambda\, v_{t_i} + (1-\lambda)\, u_{t_i}\Big)$
  else
    $x_{t_{i+1}} \leftarrow x_{t_i} + (t_i - t_{i+1})\, v_{t_i}$
$x_{\text{mid}} \leftarrow x_{t_N}$ ▷ Middle latent

/* Phase 3: Backward Refinement with Controller S_2 */
for $i = N$ **to** 1 **do**
  $v_{t_i} \leftarrow V_\theta(x_{t_i}, t_i, c_s)$
  if $t_i \in S_2$ **then**
    $x_{t_{i-1}} \leftarrow x_{t_i} + (t_{i-1} - t_i)\Big(\lambda\, v_{t_i} + (1-\lambda)\, u_{t_i}\Big)$
  else
    $x_{t_{i-1}} \leftarrow x_{t_i} + (t_{i-1} - t_i)\, v_{t_i}$
$x_{\text{orig2}} \leftarrow x_{t_0}$ ▷ Alternatively, one can use:
  $x_{\text{orig2}} \leftarrow x_{\text{orig1}} + (x_{\text{mid}} - x_{\mathrm{s}})$

/* Phase 4: Forward Refinement with Target Prompt and Kalman Filter */
for $i = 0$ **to** $N-1$ **do**
  $v_{t_i} \leftarrow -V_\theta(x_{t_i}, t_i, c_t)$
  if $t_i \in S_2$ **then**
    $x_{t_{i+1}} \leftarrow x_{t_i} + (t_i - t_{i+1})\Big(\lambda\, v_{t_i} + (1-\lambda)\, u_{t_i}\Big)$
  else if $t_i \in L$ **then**
    $x_{t_{i+1}} \leftarrow x_{t_i} + (t_i - t_{i+1})\Big(\mu\, v_{t_i} + (1-\mu)\, k_{t_i}\Big)$ ▷ k_{t_i}: Kalman filter terms (see Eq. (12))
  else
    $x_{t_{i+1}} \leftarrow x_{t_i} + (t_i - t_{i+1})\, v_{t_i}$

Output: edited image $x_{\mathrm{t}} \leftarrow x_{t_N}$

C Detailed Kalman-Edit algorithm

The detailed editing algorithm of our approach is presented in Algorithm 2. Phases 1 and 2 correspond to the first stage, where we construct the measurement sequence, while Phases 3 and 4 make up the second stage, where Kalman control is applied. Concretely, in Phase 1, we apply ODE inversion using rectified flow to construct the first part of the measurement sequence, $\{y_i\}_{i=0}^{\delta-1}$, from early inversion latents and obtain x_{orig1}. In Phase 2, we build the second part of the measurement sequence, $\{y_i\}_{i=\delta}^{l}$, using generation latents from later timesteps and acquire x_{mid}. We then compute x_{orig2} either via shortcut estimation or a second inversion. In Phase 4, the Kalman filter is applied to produce more accurate results that align with both the target prompt and the structural integrity of the original image. For hyperparameters, we define filter strengths $\{\lambda\}_{i=0}^{T}$ and $\{\mu\}_{i=0}^{T}$ to control the guidance strength in the Kalman control process. We also specify step sets S_1 and S_2 to indicate where controllers should be applied, and step set L to identify where the Kalman filter should be used. All hyperparameters are set to default values but may require tuning for different editing tasks.

D Illustration of more capabilities: Style transfer, Scene editing and large area modification

We showcase additional capabilities of Kalman-Edit, including style transfer, complex scene editing, and large-area modifications. Our method effectively adapts images to various styles, edits intricate backgrounds, and modifies extensive regions within an image. For instance, in Fig. 6, all four cases demonstrate our method's capability to edit complex background scenes while producing high-quality results. Figure 7 presents additional style transfer examples, such as converting a room into a medieval setting and performing large-area modifications, exemplified by the parrot case. In Fig. 8, we further highlight another large-area modification involving a giraffe. Moreover, Fig. 9, Fig. 10, and Fig. 11 illustrate various high-resolution style transfer cases, including transforming a picnic scene into a cartoon style and rendering a woman in the style of Van Gogh. Overall, these examples demonstrate the flexibility and effectiveness of our method in handling diverse and complex image editing tasks.

E Additional qualitative results

We present additional qualitative results, including real-world image edits and diverse editing tasks. We provide additional examples to further demonstrate the strong structural consistency and high editing quality achieved by our method. In Fig. 12, we present edited results on real-world images sampled from the DIV2K dataset. Our method effectively preserves the overall structure in non-target regions while accurately adhering to the editing prompts. For instance, in the boat insertion example, the foggy atmosphere is well preserved as a boat is naturally integrated into the scene. Other examples similarly exhibit high editing fidelity and strong structural consistency. Furthermore, Fig. 13 includes more diverse editing tasks, such as changing the breed of a dog. These results highlight the versatility and robustness of our approach in handling various types of image edits. In addition, Fig. 14 presents ablation studies on the effect of controller placement. The first row of images shows that applying controllers to early generation steps primarily influences the semantic structure, steering the output toward the original content (e.g., generating a cat) but missing background details. Conversely, the second row demonstrates that applying controllers to later steps refines visual details more effectively. These observations support our understanding of the controller's impact at different stages and help guide the selection of hyperparameters in Algorithm 2.

F Discussion of controller-based methods

In this section, we briefly discuss controller-based approaches, which are grounded in optimal control theory. As noted in [43], such methods can be applied to both diffusion-based and flow-based generative models (*i.e.*, they can operate on both stochastic differential equations (SDEs) and ordinary differential equations (ODEs)). With recent advances in flow-based models like Flux, there has been growing interest in applying control techniques specifically to ODE-based systems. From the perspective of optimal control theory, ODEs offer more favorable mathematical properties than SDEs, often enabling the derivation of more effective optimal controllers. For this reason, our proposed

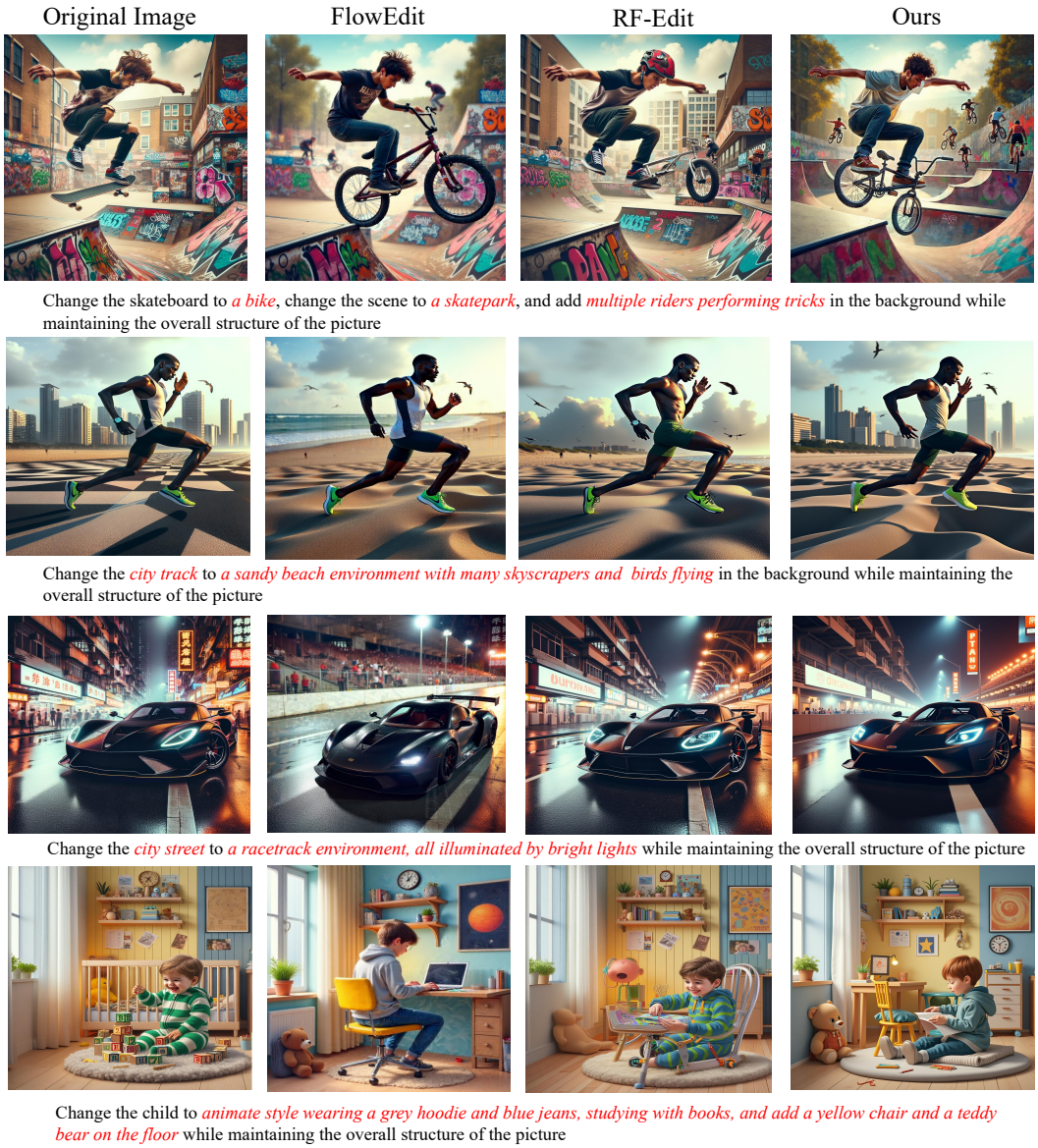

Figure 6: Illustration of intricate scene editing. We present complex scene editing cases, such as adding and modifying multiple room elements in the last-row example, and altering background scenes from a city track to a sandy beach and from a city street to a racetrack in the second-row example. Compared with baseline methods, our approach exhibits significantly stronger prompt adherence. These cases further demonstrate the robustness and versatility of our method in handling intricate editing tasks.

method adopts a flow-based framework. Nevertheless, similar to other controller-based approaches, our method can also be extended to diffusion-based models. Kalman-Edit presents a principled solution for achieving more accurate and consistent image editing through optimal control.

G Broader impact and limitations

Broader impact. While our Kalman-based method advances the quality and consistency of image editing methods, such techniques should be treated with caution due to their increasing potential for malicious use. Noteworthy, our method is training-free and does not rely on any private datasets for evaluation, thereby posing no data privacy concerns or associated negative impacts. To facilitate

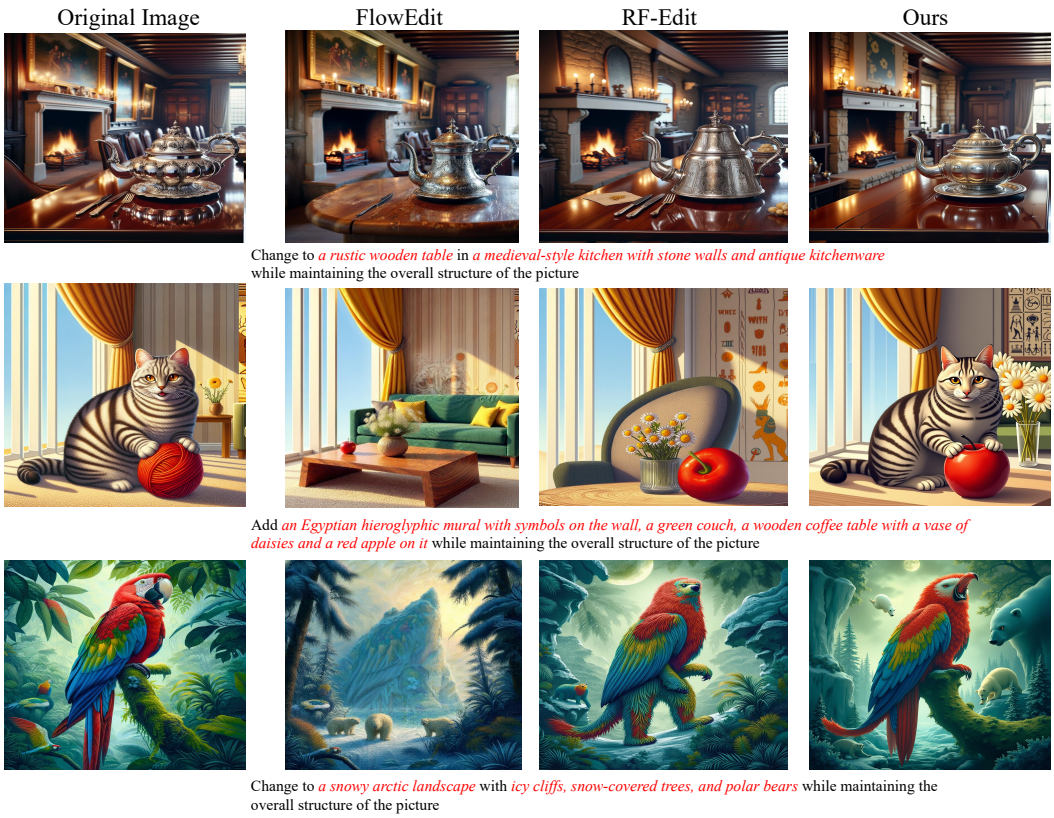

Figure 7: Illustration of large-area modification. We present examples including room style transformation (first-row case) and the addition of multiple new elements (second-row case). Furthermore, we demonstrate a case involving substantial background modification, where a forest scene is replaced with icy cliffs and polar bears (last-row case).

further open research into its practical uses and any potential societal impacts, our code would be open sourced at https://github.com/anonymous-138384/Kalman-Edit-Pytorch/.

Limitation. Due to our design focus on leveraging historical inversion latents as the measurement sequence to rectify the final generation process, the method requires additional hyperparameter tuning for tasks such as object removal. We illustrate these failure cases in Fig. 15. In addition, the noise and artifacts from the original image tend to appear in the edited images at our default control strength.

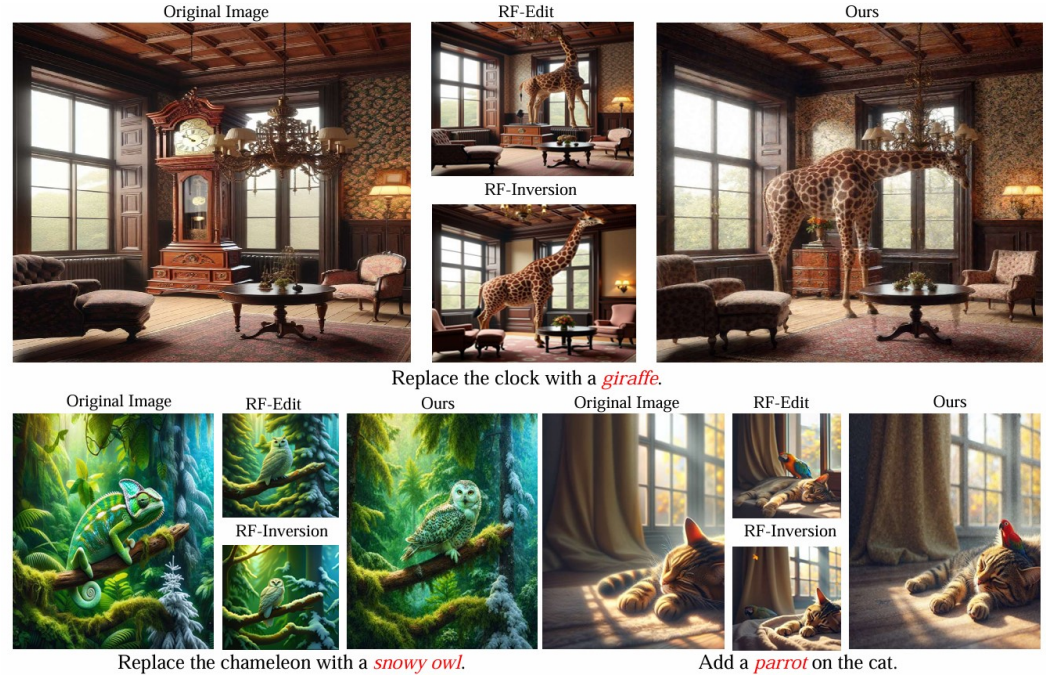

Figure 8: Illustration of large-area modification and complex scene editing. The giraffe example demonstrates our method's ability to preserve image structure, as evidenced by the chandelier remaining correctly positioned. The owl example features a complex background, and our result retains most structural details in the non-target regions. In the cat example, our method maintains strong structural consistency while accurately adhering to the target prompt.

Figure 9: Additional qualitative results on the style transfer task. We present examples of style transfer in surrealism and cartoon styles.

Figure 10: Additional qualitative results on the style transfer task. We present examples of style transfer in realism and van gogh styles.

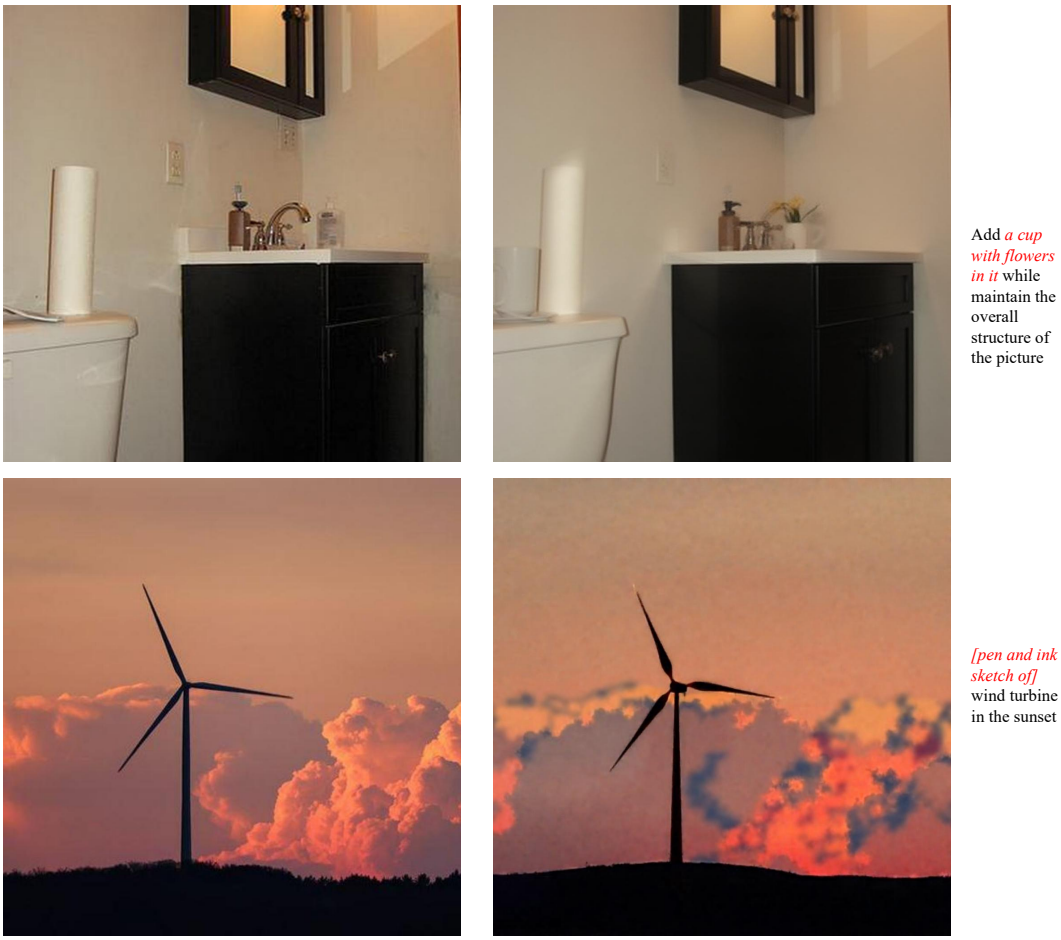

Figure 11: Additional qualitative results on the style transfer task. We present an example of style transfer to a pen-and-ink style, and additionally include a bathroom editing case.

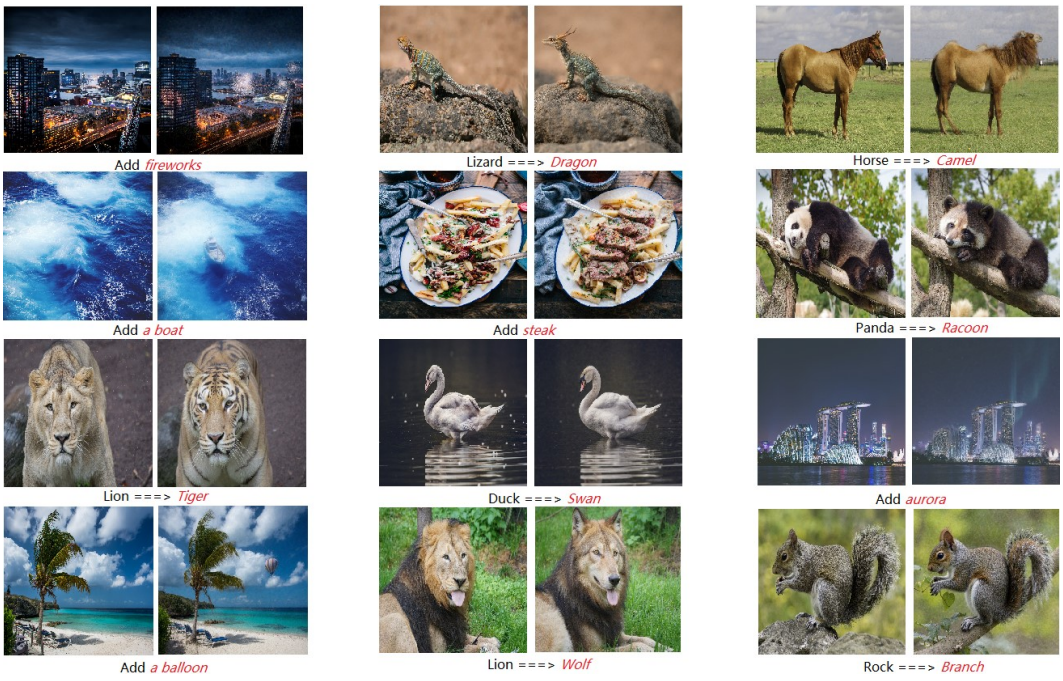

Figure 12: Additional results on real-world image editing. We present examples involving complex scenes such as urban environments and beaches, as well as images with intricate structures like the boat and the panda. These results highlight the flexibility and effectiveness of our method in handling diverse and challenging editing tasks.

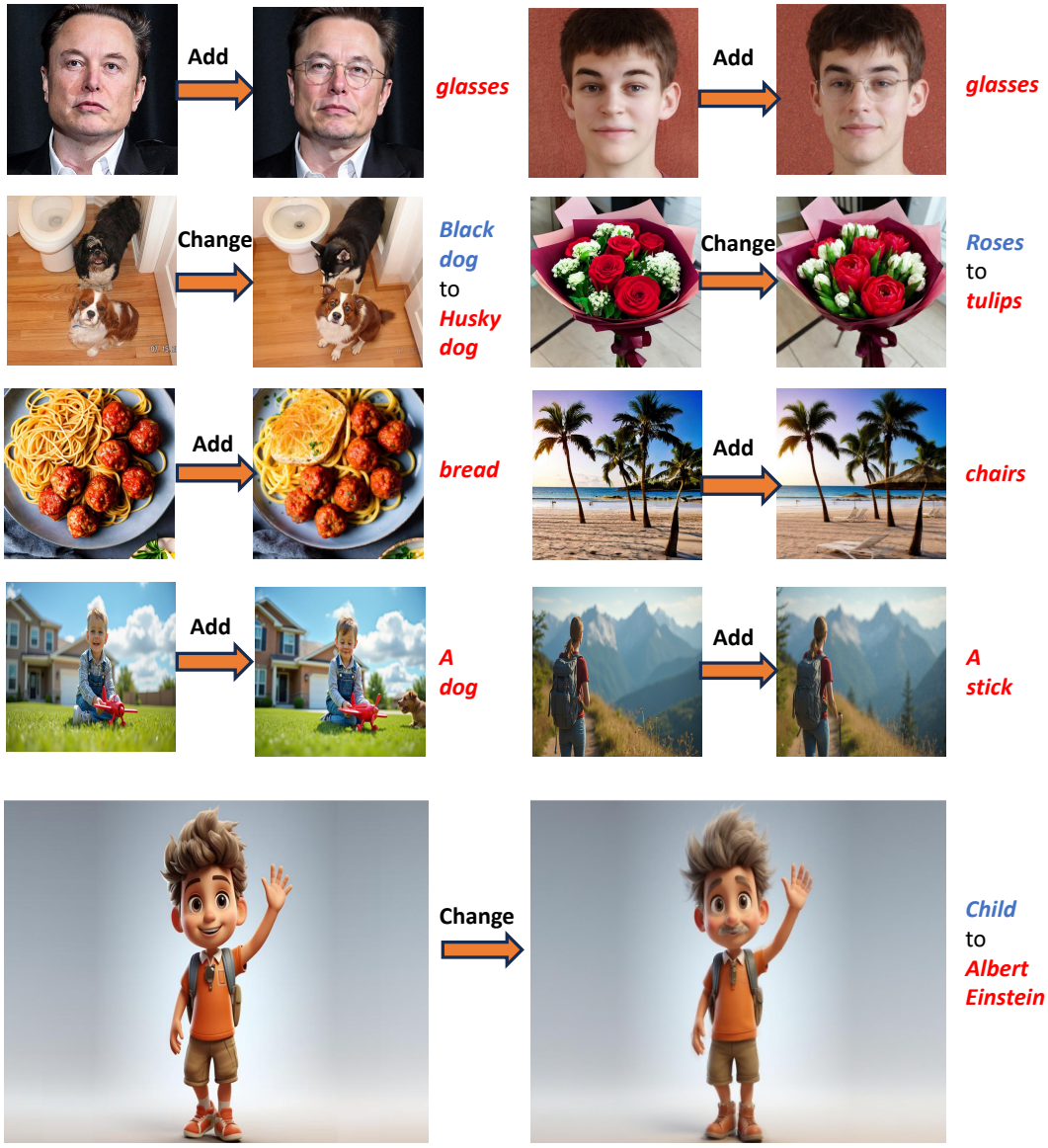

Figure 13: Additional results on diverse image editing tasks. We present more examples with a wide range of target prompts, such as changing a dog's breed and transforming roses into tulips. These results further demonstrate the strong performance and versatility of our approach.

Figure 14: Additional ablation results on controller placement. We evaluate the effects of adding controllers at different stages of the generation process. These insights inform the hyperparameter choices in our proposed algorithm. For a detailed analysis, please refer to Appendix E.

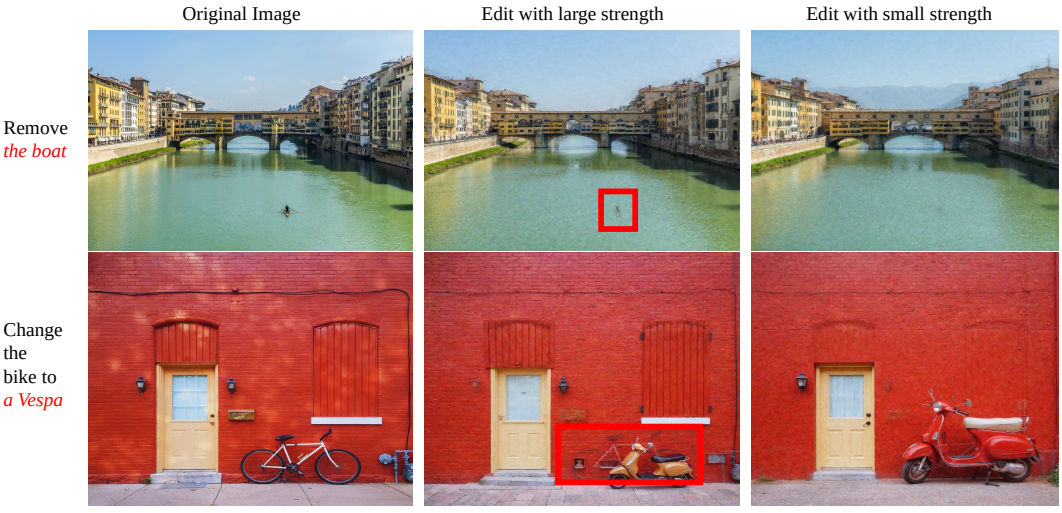

Figure 15: Limitation of our method. Improper hyperparameter settings may cause our method to fail in some editing cases, such as certain object removal tasks. See Appendix G for detailed explanation.