REED-VAE: RE-Encode Decode Training for Iterative Image Editing with Diffusion Models

Gal Almog[1] Ariel Shamir[1] Ohad Fried[1]

[1]Reichman University, Israel

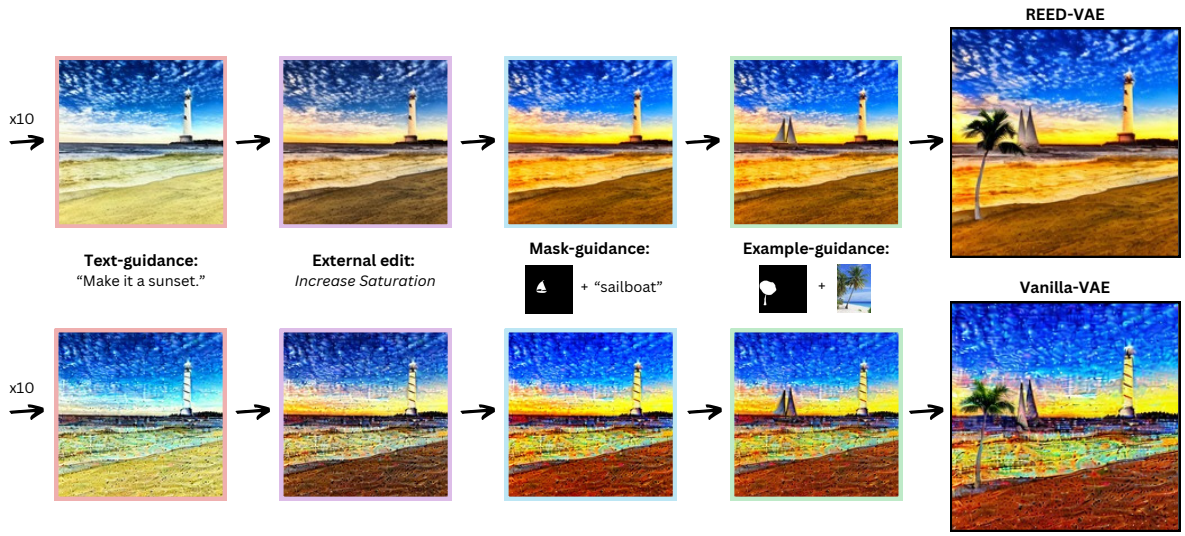

Figure 1: *REED-VAE (top) preserves image quality over multiple editing iterations, allowing users to perform multiple edit operations using a combination of frameworks and techniques. The Vanilla VAE (bottom) accumulates many artifacts and noise along the way, becoming very apparent once multiple iterative edit operations are performed. The total edit sequence consists of 14 steps, of which only the last 4 are shown here for brevity and to highlight the differences in the final picture. Four types of edit operations are performed: text-guided editing [BHE23], external editing (not diffusion-based), mask-guided editing [AFL23], and example-guided editing [YGZ*23].*

Abstract

While latent diffusion models achieve impressive image editing results, their application to iterative editing of the same image is severely restricted. When trying to apply consecutive edit operations using current models, they accumulate artifacts and noise due to repeated transitions between pixel and latent spaces. Some methods have attempted to address this limitation by performing the entire edit chain within the latent space, sacrificing flexibility by supporting only a limited, predetermined set of diffusion editing operations. We present a re-encode decode (REED) training scheme for variational autoencoders (VAEs), which promotes image quality preservation even after many iterations. Our work enables multi-method *iterative image editing: users can perform a variety of iterative edit operations, with each operation building on the output of the previous one using both diffusion-based operations and conventional editing techniques. We demonstrate the advantage of REED-VAE across a range of image editing scenarios, including text-based and mask-based editing frameworks. In addition, we show how REED-VAE enhances the overall editability of images, increasing the likelihood of successful and precise edit operations. We hope that this work will serve as a benchmark for the newly introduced task of multi-method image editing. Our code and models will be available at: https://github.com/galmog/REED-VAE.*

1. Introduction

The ability to edit high-resolution images has long been a fundamental aspect of visual content creation, enabling artists and designers to achieve desired aesthetics and convey specific messages. Traditional image editing techniques range from basic adjustments such as color correction, cropping, and sharpening, to more advanced methods such as applying various filters and layering elements. More recently, diffusion models [HJA20] have led to great advancements not only in high-resolution image generation, but also in editing methods that allow controllable manipulation of existing images. Diffusion-based editing models can receive various conditioning such as text instructions, reference images, and localization masks, and perform a wide range of tasks such as object addition/removal, object replacement, background replacement, and style or texture changes [AFL23, BHE23, CVSC22, HMT*22, NDR*21, YGZ*23]. Although many of these achieved remarkable results, they center around single-operation editing procedures.

Ideally, users should be able to integrate the strengths of both diffusion-based models and traditional editing techniques to manipulate images, while applying and interleaving several different editing frameworks. In practice, there exists an inherent problem in combining diffusion-based operations with traditional methods in the same editing session. This is because diffusion-based models primarily work in the *latent space*, while traditional methods are applied in the *pixel space*. Therefore, each time one wishes to switch between the two types of techniques, it is necessary either to encode or decode the image into the appropriate representation. The variational autoencoder (VAE) [KW13] is the most common model used for this task. As we show in this paper, this iterative cycle of encoding and decoding destroys the quality of the image by accumulating noise and artifacts with each iteration (see Figure 2).

We define *multi-method* iterative image editing as the process of performing multiple (e.g. more than 5) successive edit operations on an input image; each operation uses the previous output as its input, leveraging both diffusion-based models (latent space) and conventional editing techniques (pixel space).

Our work aims to enable such *multi-method* iterative image editing by mitigating the artifacts introduced by the VAE in the iterative autoencoding process. We train a new VAE using a novel re-encode decode (REED) training scheme. Our training procedure utilizes an iterative training process together with dynamic incrementation and a first-step loss, that together improve image quality retention over many encode-decode iterations. We demonstrate the impact of replacing the Vanilla-VAE with our REED-VAE through experiments using a wide range of diffusion-based image editing models. As our REED-VAE is based on the architecture of the vanilla VAE used in Stable Diffusion [RBL*22], it can be easily swapped into the vast majority of models.

In addition to improving multi-method iterative image editing, our REED-VAE facilitates integration between different editing paradigms; for example, between GAN-based editing methods and diffusion-based methods. Furthermore, this improvement facilitates a seamless transition between editing with multiple different diffusion models that may have different latent spaces, for example, SD2 (4-channel) [RBL*22] and SD3 (16-channel) [EKB*24]. Beyond image editing, reducing the noise and artifacts accumulated through iterative VAE use has applications in other domains, such as NeRF editing methods in which it is very common to apply the VAE multiple times. In ED-NeRF [PKY23], it has been demonstrated that performing the NeRF editing process entirely in the latent space, thus avoiding repeated applications of the VAE, leads to significantly better results. We will publish our code and trained REED-VAE model in hope that they can be a useful contribution to the community.

2. Related work

Latent Diffusion Models Diffusion models [HJA20, RDN*22, SCS*22] are a class of deep generative models trained to convert random noise to an image sample from a given distribution. These models generate images in an iterative manner, removing a small amount of noise at each time step. To reduce the computational cost of high-resolution image generation with diffusion models, [RBL*22] introduced their latent diffusion model (LDM), which utilize a separately-trained autoencoding model that learns to map images to a latent space. This latent space is perceptually equivalent to the original pixel space, but significantly reduces computational complexity. LDMs achieve impressive results in both image generation and editing, and thus form the basis of all state-of-the-art editing models we evaluate REED on. We refer to [HJA20] for more details on diffusion models and their implementation.

Variational Autoencoders Variational autoencoders (VAEs), introduced by [KW13], are a class of probabilistic models designed to find a low-dimensional (latent) representation of data, widely used in LDMs for converting images to and from their latent representations. Unlike traditional (deterministic) autoencoders that encode a vector x into a single latent vector z and decode z back to the original space, VAEs encode the input image as a *distribution* over the latent space. This regularizes the latent space and ensures that the model generates data following a specified distribution. Common VAE architectures include the Vector Quantized Variational Autoencoder (VQ-VAE) [VDOV*17] and VQ-GAN [ERO21]. Stable Diffusion models [RBL*22] commonly use a traditional VAE regularized with either KL-divergence [KW13] or vector quantization (VQ-GAN, [ERO21]) in their diffusion models. Since the VAE model with KL loss is the most prevalent in editing models, we use it as the baseline for REED to ensure maximal compatibility with other models.

Image Editing with Diffusion Models Diffusion models [HJA20, RDN*22, SCS*22] have significantly advanced high-resolution image generation and editing methods, enabling tasks such as object addition/removal, object replacement, background changes, and style or texture changes [HHL*24]. These models are typically conditioned on text instructions [BHE23, PKSZ*23, ZMC*23, GAA*22, ZYF*24, PGXH23, KZL*23] or reference images [YGZ*23, MHS*21, JZB*24], sometimes with additional masks for localizing edits or inpainting [AFL23, ALF22, NDR*21, RBL*22, LDR*22, CWQ*23]

Prompt-to-Prompt (P2P) [HMT*22] introduced attention modification as a framework for image editing by identifying that the

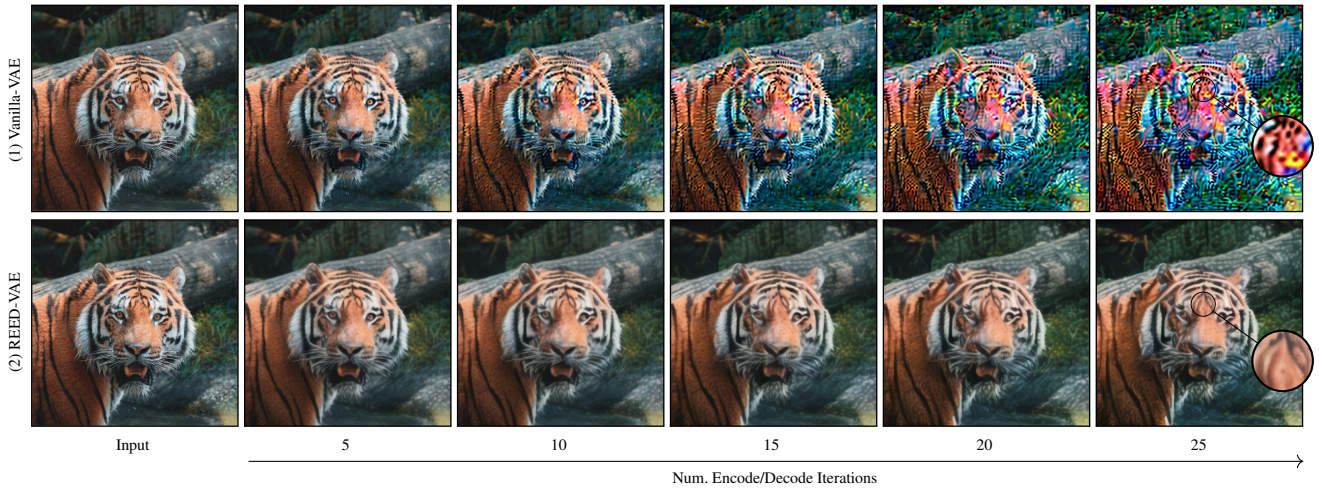

Figure 2: *Even without a diffusion model in the pipeline, the Vanilla-VAE (top row) accumulates artifacts and exhibits significant distortion very quickly throughout encode-decode iterations. The tiger's features lose their distinct shapes and edges, appearing more globular and less defined. The color palette is altered, with a noticeable increase in blue tones and a decrease in the richness of the orange and greens. Fine details such as the grass and the fur are largely lost or blurred. REED-VAE (bottom row), produces successive images that are robust to such artifacts and distortions. The tiger retains its shape, color, and surface details, demonstrating remarkably high fidelity to the original image. The subtle variations in orange and white hues are preserved, and fine elements remain visible.*

cross-attention layers in the diffusion model link prompt tokens to the image layout. By swapping attention masks between the source and target images, P2P allows specific elements to be edited while keeping the rest static. However, P2P was limited to generated (synthetic) images; to enable real-image editing, *inversion* techniques such as DDIM Inversion [DN21, SME20] are required to map real images into the latent space of pre-trained diffusion models. Inversion is the task of finding the latent vector such that denoising it with the pre-trained diffusion model will return the original image, allowing image latents to be edited throughout the denoising process. Originally, DDIM Inversion suffered from notable limitations in preserving high-frequency details and achieving exact image reconstrctions, which are crucial for editing workflows. To address these limtiations, Null-Text Inversion (NTI) [MHA*23] was introduced as an improvement over DDIM Inversion. NTI refines the inversion process by leveraging null-text guidance to achieve highly accurate reconstructions, thus enabling real-image editing with methods such as P2P. InstructPix2Pix [BHE23] introduced instructional image editing [ZMC*23, PKSZ*23] by training a fully supervised diffusion model that can edit based on human instructions — for example, "swap the car with a motorcycle".

In addition to text, other methods utilize masked regions with corresponding text or a reference image for local editing. Blended Latent Diffusion [AFL23] achieves smooth edits by blending the edited region within the mask with the background at each diffusion step. Paint by Example (PbE) [YGZ*23] performs subject-driven editing using an input mask and a reference image, utilizing self-supervised learning to generate training data. DragDiffusion [SXP*23] enables interactive point-based image editing that achieve accurate spatial control.

Each of the described editing models builds upon a latent diffusion model, employing a vanilla-VAE to transition in and out of the latent space for each edit operation. Few works address the limitation of long editing sequences imposed by the VAE [JUS*24, YZLL23], yet their solutions are not *multi-method*, disallowing a combination of edit methods that are diffusion-based and those that operate in the pixel space in the same edit session. In contrast, REED-VAE can be seamlessly integrated into any diffusion-based editing method, replacing the original VAE to facilitate iterative image editing that better retains image quality, while also allowing to interleave non-diffusion-based editing operations. We demonstrate its effectiveness on different types of editing models in Section 5.

Iterative Image Editing To the best of our knowledge, only one prior work has directly addressed iterative image editing. Joseph et al. [JUS*24] extend InstructPix2Pix [BHE23] to support iterative multi-granular editing by staying inside the latent space throughout the entire editing session, only decoding back into pixel space once at the end. At each iterative step, they overcome the autoencoder-induced degradation by using the previous encoded latent and the current edit instruction as input to the diffusion model, rather than decoding it after each step. There are several notable limitations of this approach:

1. **Restriction to diffusion-based editing**: Editing sessions are restricted to diffusion-based editing methods that operate in the latent space, excluding manual image manipulations, traditional editing techniques, and editing models that operate directly in the pixel space.
2. **Restriction to a single model:** Edit operations are limited to models that use the same latent space. This prevents applying both diffusion-based editing and GAN-based editing, for exam-

ple, or even mixing diffusion models that have different latent spaces, such as SD2 [RBL*22] and SD3 [EKB*24]

3. **Rigid and inconvenient workflow**: Users must either predetermine all edits and their locations, limiting the exploratory nature of the creative process, or save an additional latent vector along with each output image, complicating storage and sharing.

Our work addresses these limitations by enabling users to iteratively edit images using any combination of methods that operate in either the pixel space or latent space, without the need to predetermine all edit operations or handle extra latent vectors.

3. Problem Definition

We call our new problem setting *multi-method iterative image editing*. At step i, our goal is to apply the edit operation e^i to the input image x^i, such that the output x^{i+1} can serve as the input for the next iteration while minimizing the accumulation of noise. We use the term *multi-method* to emphasize that each e^i may refer to any type of diffusion-based editing operation, but also any type of operation employed in traditional or commercial image editing tools. Generally, we aim for each step in the editing process to be independent, so that users have the creative freedom to apply each e^i at any stage during the editing session, using a different tool and without the need to pre-define each edit in advance. We believe that this setting better represents real-life workflows and is more conducive to the artistic process of image editing.

Naïvely passing iterative image outputs to the diffusion model accumulates artifacts and renders the images essentially destroyed when performing editing beyond a few operations. As demonstrated in Figure 2, even simply encoding and decoding the same image iteratively (without editing or using the diffusion model) is enough to accumulate significant artifacts after only 5-10 iterations. Similar to previous work [JUS*24, YZLL23], we find that this degradation is a result of the lossy VAE used in the diffusion process. Figure 3(a-b) demonstrates this in the frequency domain: the Vanilla-VAE exhibits significant loss of high-frequency information after several encode-decode cycles, while also accumulating high-frequency noise and artifacts. Therefore, to successfully support multi-method iterative image-editing, we focus our efforts on preventing the degradation that occurs due to the reconstruction error of the VAE in LDMs.

4. REED: RE-Encode Decode Training

We train a VAE that, when paired with a diffusion-based image editing model, can maintain image quality and editability over iterations. VAEs consist of an encoder network that defines a posterior distribution $q(z|x)$, a prior distribution $p(z)$, and a decoder network that models $p(x|z)$. Typically, both the posterior and prior distributions are chosen to be normal with diagonal covariance for efficient parameterization by the Gaussian reparameterization trick [KW13]. Training is regularized with a KL-divergence term between the returned posterior and a standard Gaussian distribution.

Fine-tuning VAE models on specific or niche datasets has been demonstrated to outperform training models from scratch in image generation and editing pipelines. Consequently, we initialize

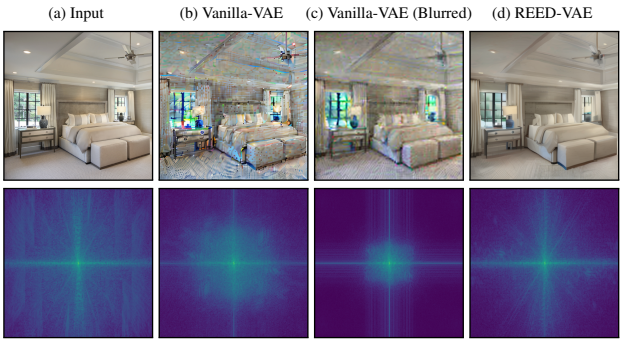

Figure 3: *Given an input image (a) we perform 20 encode-decode iterations and present the the results in image (top) and frequency domain (bottom). Vanilla-VAE (b) exhibits significant loss of high-frequency information (evidenced by the dimming and blurring of the outer regions of the spectrum), and dominance of low-frequency features (evidenced by the enlarged central bright region). In addition, it also introduced new high-frequency features that are not seen in the input image, indicating an introduction of repetitive artifacts. Trying to apply smoothing after each encode-decode iteration (c) solves some of these problems at the cost of blurring the image. REED-VAE (d) demonstrates superior performance in preserving image fidelity across all frequency bands.*

our model's weights with a pretrained VAE checkpoint from Stable Diffusion to leverage its extensive encoding and decoding capabilities. However, to ensure wide compatibility with many image editing models, we fine-tune only the decoder of the VAE, leaving the encoder weights frozen. This ensures that the latent embeddings, on which the diffusion model itself is highly dependent, remain consistent and aligned with the model's training distribution. This strategy also reduces the computational resources required for training. Additional training details are provided in the supplemental material.

4.1. Iterative training

As our goal is to reduce the quality degradation that occurs in the iterative encoding and decoding process, we train our model with this task in mind. We define a parameter k to indicate the number of encode-decode iterations performed on each sample in the training loop. For each training iteration, we begin by encoding ($\mathcal{E}$) the source image x^0, sampling z^0 from the encoded latent distribution, and decoding ($\mathcal{D}$) z^0 to acquire the output image x^1. We define this as one encode-decode iteration, and repeat this process for each x^i for $i = 0$ to k, where x^k is the final output image as follows:

$$\mathcal{D}(\mathcal{E}(x^i)) = x^{i+1} \tag{1}$$

Whereas the reconstruction loss of the Vanilla VAE is computed after one iteration between x^0 and x^1, we perform k encode-decode iterations and compute the loss, detailed in Section 4.4, on x^k. We hypothesize that explicitly training the model to reconstruct images after multiple encode-decode iterations will allow it to generalize to the image editing task, i.e., will also improve the model's ability to reconstruct successive images when edits are performed. First, iter-

ative encoding and decoding is a simpler task than iterative editing, yet the VAE still exhibits a large amount of degradation, as evident in Figure 2. Therefore, this is a good intermediate goal for our model. Second, our experiments show that a similar artifact accumulation occurs in both image editing and simple encode-decode cycles, therefore, it is reasonable to assume that a VAE that overcomes one challenge will also improve in the other. Our results indeed confirm that our model demonstrates improved reconstruction accuracy during both iterative encode-decode cycles *and* iterative edit operations (see Section 5). We further analyze and validate the advantage of having $k > 1$ in our ablation studies (Section 5.1).

4.2. Dynamic incrementation

We initially find that a higher k value imparts a greater reduction in the training loss, however; past $k = 6$, the task becomes too difficult and the model does not converge. To overcome this, we propose a dynamic loss progression that makes use of increasingly higher values of k to achieve better convergence. We begin by initializing $k = 4$ and computing the training loss against x^k in each training iteration. At the end of each epoch, we compute a validation loss on a separate validation set, also against the same x^k (details on the exact loss implementation in section 4.4). If the validation loss does not improve in 5 consecutive iterations, signifying a plateau in the training, we increment $k \leftarrow k+1$. Effectively, we perform a variation of curriculum learning [BLCW09], which attempts to mimic human learning by gradually increasing the complexity of data samples used when training a model. Empirically, we find $k < 4$ to be too trivial of a task, leading to no convergence (see ablations in Section 5.1). We stop training once k passes 20. We find that this training method achieves the best results, allowing the model to learn to reconstruct images almost perfectly after 10 iterations with good generalisability to higher iterations.

4.3. First-step loss

When using any existing model to compress an image into a latent representation, there is an inherent and unavoidable loss of information, such that even after a single encode/decode iteration the reconstruction will not be perfect. We are interested in improving the model's *iterative* performance, and not its general performance (i.e., we are satisfied if we match the original model's performance after one iteration, as long as we improve it for consecutive iterations). For this reason, rather than computing the training loss between the last iteration output x^k and the source image x^0, we instead compute the training loss between x^k and x^1. Our experiments and qualitative analysis show that the first-step loss helps the VAE learn the iterative task more easily, which we believe is due to the reduced complexity of the task (discussed further in our ablation studies, Section 5.1). Note that the validation loss and test metrics are still computed against x^0, as this is the true performance indicator.

4.4. Training objective

Putting the three components (iterative training, dynamic incrementation and first-step loss) together, our full REED training algorithm is presented in Algorithm 1. We use the same training loss as in the vanilla-VAE [RBL*22, KW13], which is composed of an MSE reconstruction term ($\mathcal{L}_{MSE}$) along with a weighted perceptual loss ($\mathcal{L}_{LPIPS}$, [ZIE*18]) term. An additional KL-divergence term (D_{KL}) is computed between the latent vector and the standard normal distribution for regularization. We find that despite only training the decoder, the D_{KL} term still helps to improve the VAE's performance. We believe this is due to the iterative nature of our training method: the output of the decoder indirectly impacts the next latent vector, therefore latent space regularization is still beneficial. We add additional scaling parameters α and β to scale the LPIPS and D_{KL} terms, respectively. The full objectives for training and validating the VAE are as follows:

$$\mathcal{L}_{train} = \mathcal{L}_{MSE}(x^1, x^k) + \alpha \cdot \mathcal{L}_{LPIPS}(x^1, x^k) + \beta \cdot D_{\mathrm{KL}}\left(z^k, \mathcal{N}(0, I)\right) \quad (2)$$

$$\mathcal{L}_{val} = \mathcal{L}_{MSE}(x^0, x^k) + \alpha \cdot \mathcal{L}_{LPIPS}(x^0, x^k) \quad (3)$$

Algorithm 1: Re-Encode Decode Training

Input: Training data $\mathbf{X}$, number of epochs N, pretrained encoder $\mathcal{E}$ and decoder $\mathcal{D}$ of vanilla VAE model
Output: Trained REED-VAE decoder $\mathcal{D}$ parameters

```
Initialize number of encode/decode iterations k ← 4;
for epoch ← 1 to N do
    for each image x^0 ∈ X do
        for i = 0, 1, ..., k − 1 do
            Encode z^i ← E(x^i);
            Decode x^{i+1} ← D(z^i);
        end
        Take gradient descent step on ∇L_train(x^1, x^k, z^k)
    end
    if L_val(x^0, x^k) reaches plateau then
        k ← k + 1;
        if k > 20 then
            End training
        end
    end
end
```

5. Experiments

We conduct a comprehensive evaluation of our proposed REED-VAE across various image editing models, comparing their performance with and without REED-VAE integration. To evaluate the effectiveness of our method, we require a dataset that contains images with ground-truth edits for many (20+) steps. Unfortunately, as we are the first to perform comprehensive iterative image editing, such a dataset does not exist (to the best of our knowledge). We address this by adapting the recently released ImagenHub dataset [KLZ*24] to an iterative editing process. ImagenHub aims to standardize the evaluation process for image editing and generation models, providing a comprehensive dataset of 7 task subsets each with 100–200 images. The dataset images are manually annotated

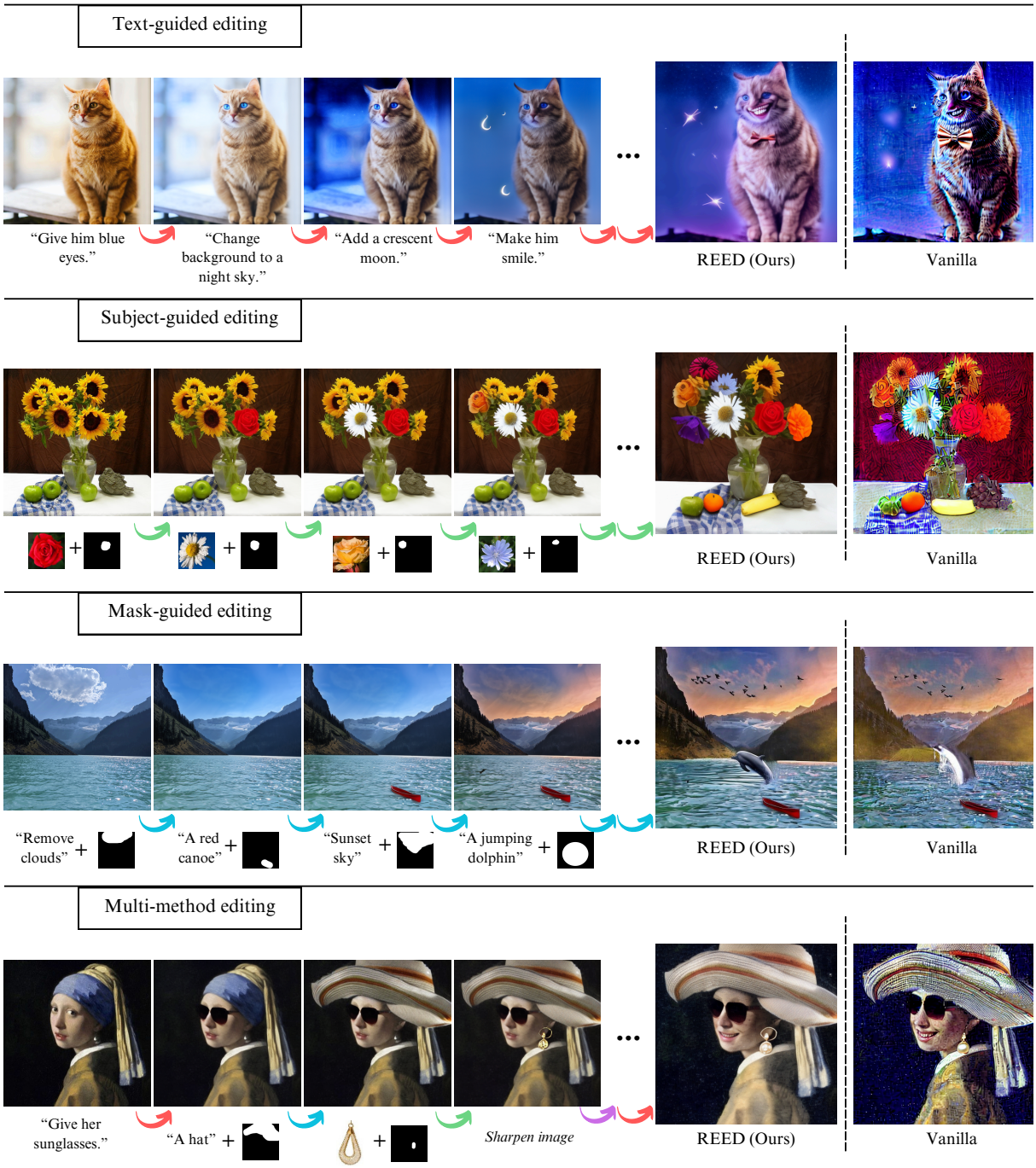

Figure 4: *Examples of types of edit sessions made possible with REED-VAE. Using the Vanilla-VAE (right), significant noise and artifacts accumulate quickly after multiple edit operations. Intermediate edit operations are omitted to highlight the final edited image. Four types of edit operations are performed: text-guided editing [BHE23], external editing (not diffusion-based), mask-guided editing [AFL23], and example-guided editing [YGZ*23].*

for various image editing scenarios: single-turn, multi-turn, mask-guided, text-guided, and subject-guided image editing. We refer to the ImagenHub paper [KLZ*24] for more implementation details. We leverage ImagenHub as a starting point for our evaluations, making adaptations (detailed below) to each evaluation task to accommodate for the iterative nature of our problem setting. We note that these adaptations may sometimes result in edits that are not entirely sensible or visually appealing. However, since our primary concern is quantitatively comparing the performance of REED-VAE to the Vanilla-VAE, the realism of the editing result is less important, and this procedure provides a fair comparison.

Evaluation metrics Our goal is to apply many iterative edit operations on a single image while maintaining the image quality as best as possible. For evaluation, we perform a pair of "inverse edit operations" $\{e^1, e^2\}$ on each test image, iterating back and forth through these operations for multiple cycles. For example, e^1 might be some edit operation to 'change the car into a bus', then e^2 will be to 'change the bus into a car'. For all tasks, we use the mean squared error (MSE), LPIPS [ZIE*18], SSIM [WBSS04], and FID [HRU*17, Sei20] as metrics to evaluate our model's ability to preserve image quality over successive iterations, when compared to the vanilla-VAE. These metrics are commonly used to quantify reconstruction quality of images and generation quality. In all experiments, we observe the improvement that REED-VAE imparts at three different iterative editing stages (5, 15, and 25 iterative edit operations) using a diverse set of editing models.

Notation summary Unless stated otherwise, we use the following notation for all following discussions. Each sample in our evaluation set has a source image $\mathbf{x}_s \in \mathbb{R}^{H\times W\times 3}$ and a target image $\mathbf{x}_t \in \mathbb{R}^{H\times W\times 3}$. The source caption $\mathbf{C}_s$ globally describes $\mathbf{x}_s$, the target caption $\mathbf{C}_t$ globally describes $\mathbf{x}_t$, and the local target caption $\mathbf{C}_t^{local}$ describes the local object to be edited. The sample is also annotated with a human instruction $\mathbf{I}_s$ for transitioning from $\mathbf{x}_s$ to $\mathbf{x}_t$. Some images are additionally annotated with a binary mask $\mathbf{m} \in \{0,1\}^{H\times W}$ representing the pixels to be edited with a value of 1.

5.1. Ablations

First, we validate the contribution of our main components by measuring their performance in the iterative encode-decode task (for now, without editing). We compute metrics for 5, 15 and 25 iterative encode-decode cycles and provide the results in Table 2. Our full REED-VAE model, including all of our main components, is able to best maintain the image quality over many iterative encode-decode operations. Please see the supplementary material for a visual example of the improvement provided by each component. Both the quantitative and qualitative results show a clear improvement from simply introducing iterative training (IT). Here, the iterative training is static, i.e., k does not change during training. When trained with only $(k = 2)$, the model performs marginally better than the Vanilla VAE model — increasing to $(k = 5)$, the results are more significant. The first-step loss (FSL) further improves the model performance. It is worth noting that when evaluated at only 5 iterations, the IT + FSL $(k = 5)$ model exhibits marginally better performance in MSE, FID and LPIPS; this makes sense as the model is optimized specifically for this task (5 encode/decode iterations). However, when evaluated after 15 or 25 encode/decode iterations, it is clear in Table 2 that the dynamic incrementation (DI) component yields a significant improvement in all metrics.

5.2. Text-guided image editing

To evaluate REED-VAE on text-guided image editing, we consider InstructPix2Pix [BHE23], DiffEdit [CVSC22], and MagicBrush [ZMC*23]. DiffEdit takes an input of $\{\mathbf{x}_s, \mathbf{C}_s, \mathbf{C}_t\}$; we treat this as an iterative task by repeatedly editing $\mathbf{x}_s$ in alternating directions of either $\mathbf{C}_t$ or $\mathbf{C}_s$. This is straight-forward as both $\mathbf{C}_s$ and $\mathbf{C}_t$ are provided in the ImagenHub dataset. On the other hand, InstructPix2Pix and MagicBrush take an input of the source image and an instruction prompt $\{\mathbf{x}_s, \mathbf{I}_s\}$. Here, in order to evaluate REED-VAE, we manually add "reverse prompts" to perform each given edit in the opposite direction. For some examples of reverse edit prompts, please see the supplementary material. We will make our full, revised dataset available with our code. Across all evaluated text-editing models, the integration of REED-VAE demonstrates consistent improvements in image quality and editing stability over multiple iterations, as shown in Table 1. These improvements are particularly pronounced in perceptual quality metrics (LPIPS and FID) and become more significant as the number of editing iterations increases.

5.3. Mask-guided image editing

We evaluate our method on the mask-guided image editing model Stable Diffusion (SD) Inpaint [RBL*22]. For this evaluation, we adopt an iterative editing procedure similar to our approach for text-guided editing. Specifically, we repeatedly edit images back and forth between different target states across multiple iterations. Given $\{\mathbf{x}_s, \mathbf{C}_t^{local}, \mathbf{m}\}$, the SD Inpaint model generates an output image that aims to depict the target object $\mathbf{C}_t^{local}$ within the masked region of $\mathbf{x}_s$. To simplify the evaluation, we repeat the same inpainting task across all iterations, inpainting the same object $\mathbf{C}_t^{local}$ into the masked region at each step. Our results show that the SD inpaint model achieves clear improvements across all metrics, with particularly strong enhancements in PSNR and FID scores as the number of iterations increases (Table 1).

5.4. Exemplar-driven image editing

We use REED-VAE with Paint by Example (PbE) [YGZ*23] to evaluate our method on the task of subject-driven image editing. PbE takes as input $\{\mathbf{x}_s, \mathbf{C}_t, \mathbf{m}\}$, where $\mathbf{x}_r^1$ is an additional reference image, representing the object to be depicted in the masked region of $\mathbf{x}_s$. To transform this into an iterative task, we apply the initial $\mathbf{m}$ to $\mathbf{x}_s$ and generate a tight bounding box around the mask. We use this generated bounding box to create $\mathbf{x}_r^2$, a new reference image containing the original object from $\mathbf{x}_s$ that we replaced in the previous edit operation. Now, we can alternate iteratively in a similar fashion: at each iterative step we provide $\{\mathbf{x}_s, \mathbf{m}\}$, and one of either $\mathbf{x}_r^1$ or $\mathbf{x}_r^2$. As demonstrated in Table 1, enhancing PbE with REED-VAE improved all metrics during this iterative editing procedure. Most notably, at 15 and 25 iterations the LPIPS metric was reduced by 50% or more when REED was used instead of the Vanilla VAE,

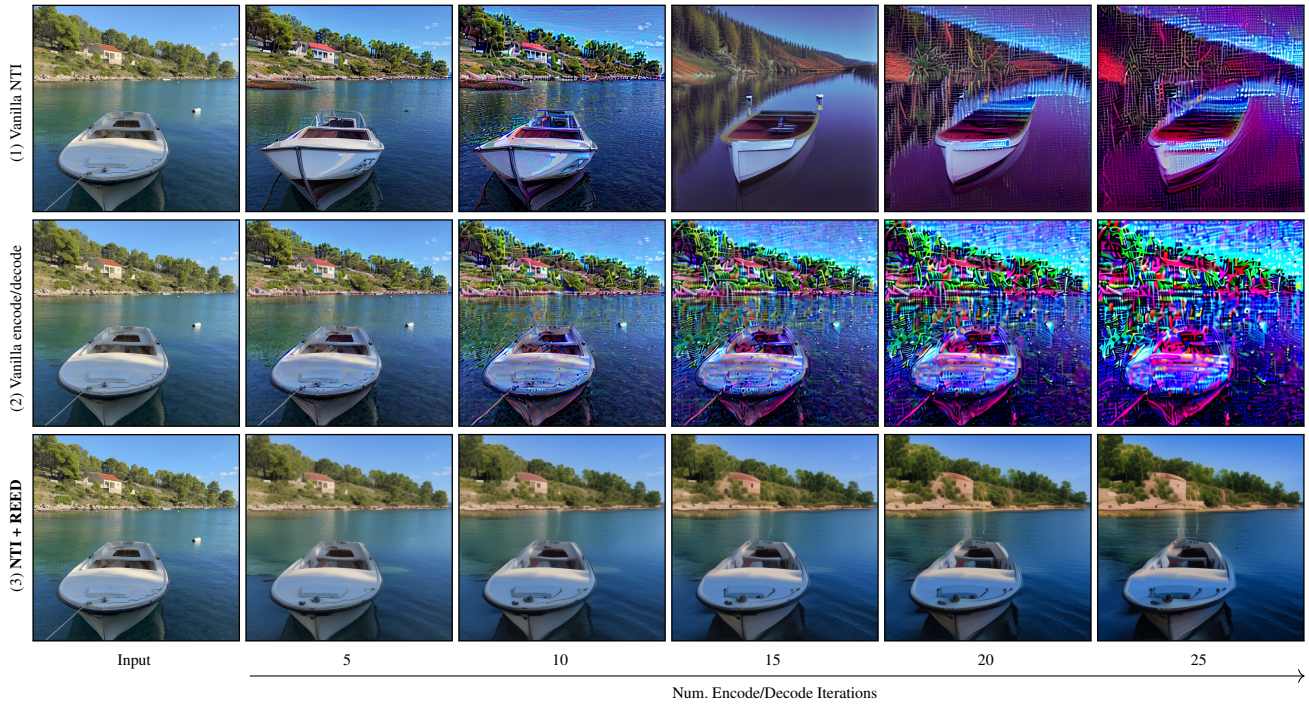

Figure 5: *Row 1: Null-Text Inversion (NTI) is used to iteratively invert the image and regenerate it from the inverted latent. Row 2: the Vanilla-VAE is used to iteratively encode and decode the image. Row 3: NTI is used with REED-VAE to iteratively invert the image and regenerate it from the inverted latent. Vanilla NTI loses fidelity to the original image and is not resilient to iterative degradation. Full sequences for Vanilla NTI and NTI + REED are available in the Supplementary Material.*

Table 1: *Comparisons on image editing models. The addition of REED consistently improves the performance of various image editing models across multiple quality metrics and through different iterative editing stages (5,15, and 25 iterations).MSE, PSNR, and LPIPS are computed with each image sample normalized to the [0,1] range prior to evaluation, ensuring consistency in comparison. Other metrics are computed as per their standard definitions. Metrics are computed based on experiments done on the ImagenHub dataset, consisting of 179 images.*

Method	MSE ↓			LPIPS ↓			SSIM ↑			FID ↓			PSNR ↑		
	5	15	25	5	15	25	5	15	25	5	15	25	5	15	25
IP2P [BHE23]	0.02	0.11	0.15	0.33	0.69	0.76	0.60	0.23	0.18	105.75	246.98	271.72	17.78	10.15	8.59
+ REED	**0.02**	**0.06**	**0.09**	**0.18**	**0.45**	**0.58**	**0.80**	**0.53**	**0.41**	**62.84**	**138.90**	**187.97**	**19.81**	**13.36**	**11.36**
MagicBrush [ZMC*23]	0.02	0.08	0.14	0.31	0.71	0.80	0.65	0.21	0.13	103.55	266.99	295.81	18.84	11.35	8.75
+ REED	**0.01**	**0.03**	**0.05**	**0.19**	**0.51**	**0.69**	**0.81**	**0.60**	**0.45**	**74.70**	**174.69**	**223.75**	**21.53**	**16.66**	**14.09**
DiffEdit [CVSC22]	**0.03**	0.06	0.08	0.34	0.62	0.73	0.65	0.36	0.21	160.73	246.33	301.91	15.99	12.59	11.21
+ REED	**0.03**	**0.07**	0.08	**0.30**	**0.55**	**0.68**	**0.69**	**0.48**	**0.40**	**160.28**	**226.91**	**246.52**	**16.24**	**12.91**	**11.44**
PbE [YGZ*23]	0.02	0.04	0.07	0.26	0.60	0.71	0.66	0.33	0.22	83.49	209.29	253.57	18.55	13.88	11.74
+ REED	**0.02**	**0.03**	**0.04**	**0.20**	**0.44**	**0.59**	**0.77**	**0.61**	**0.54**	**74.24**	**141.09**	**178.29**	**19.62**	**16.19**	**14.43**
SD Inpainting [RBL*22]	0.01	0.06	0.11	0.29	0.69	0.78	0.67	0.22	0.14	95.09	255.78	283.36	20.46	12.31	9.72
+ REED	**0.01**	**0.03**	**0.05**	**0.17**	**0.47**	**0.65**	**0.80**	**0.57**	**0.41**	**72.73**	**166.06**	**210.42**	**23.14**	**16.66**	**13.61**

Table 2: *Ablation on individual components. Static Iterative training (IT) improves metrics more significantly when k is increased from* 2 *to* 5*. First-step loss (FSL) further improves the model performance. The final component, dynamic incrementation (DI), imparts a greater improvements as the number of editing iterations is increased. Overall, the full REED-VAE model most effectively mitigates quality degradation and maintains image fidelity and realism even after numerous edits. MSE, PSNR, and LPIPS are computed with each image sample normalized to the [0,1] range prior to evaluation, ensuring consistency in comparison. Other metrics are computed as per their standard definitions. Metrics are computed based on experiments done on the ImagenHub dataset, consisting of 179 images.*

Model	MSE ↓			LPIPS ↓			SSIM ↑			FID ↓			PSNR ↑		
	5	15	25	5	15	25	5	15	25	5	15	25	5	15	25
Vanilla-VAE	0.0031	0.0126	0.034	0.18	0.54	0.70	0.77	0.49	0.27	2.54	47.68	137.04	26.09	19.32	14.84
IT ($k=2$)	0.0024	0.0069	0.014	0.16	0.35	0.47	0.78	0.62	0.47	5.57	17.61	38.44	27.15	21.89	18.66
IT ($k=5$)	0.0023	0.0059	0.011	0.12	0.24	0.34	0.80	0.67	0.55	4.91	7.60	13.94	27.28	22.71	19.77
IT + FSL ($k=5$)	0.0021	0.0064	0.012	0.13	0.23	0.35	0.79	**0.69**	0.59	3.80	7.37	11.08	27.02	22.39	19.49
IT + FSL + DI ($k=5$)	**0.0019**	**0.0055**	**0.010**	**0.11**	**0.21**	**0.28**	**0.81**	**0.69**	**0.62**	**1.78**	**3.40**	**7.81**	**28.86**	**22.79**	**20.16**

suggesting a substantial improvement in perceptual similarity to the target images.

5.5. Comparison with Inversion-Based Methods

Inversion-based methods, such as DDIM-inversion [SME20, DN21] and Null-Text Inversion (NTI) [MHA*23], are widely used in diffusion-based editing methods. Inversion attempts to find the initial noise vector that will produce the input image when fed into the diffusion model along with the original image prompt. Doing this accurately is crucial for editing real images with methods that function by manipulating the latent vectors throughout the denoising process, such as Prompt-to-Prompt [HMT*22] and DiffEdit [CVSC22]. By regenerating the latent through the denoising process, inversion methods provide an alternative pathway to the image latent, as opposed to directly using the VAE encoder before applying edits (as done in other methods [BHE23, ZMC*23, YGZ*23, RBL*22]. This raises the question of whether inversion-based editing methods can inherently mitigate the degradation observed in iterative editing tasks.

To explore this, we perform iterative inversion using NTI and compare its performance to NTI combined with REED-VAE, as well as to iterative encoding/decoding with the Vanilla-VAE (all without applying edits). For iterative NTI, we first invert the image using NTI with a source prompt and then regenerate the image using the diffusion model with the same source prompt (thus, no edits are performed, the process regenerates the input image). The results, shown in Figure 5, reveal that NTI in fact accumulates considerable noise and artifacts over iterations. As well, the initial reconstructions with NTI are not perfect, while NTI+REED achieves a much more faithful reconstruction at iteration 5. At about iteration 10, Vanilla NTI significantly loses fidelity to the original image, resulting in heavy distortions. We refer to the supplementary material for figures demonstrating the full iterative process which provides more insights into this phenomenon. It is important to note that NTI (as well as regular DDIM-inversion) relies on the VAE to encode the image as the starting point for inversion. The image is then generated by the diffusion model using the inverted latent as the starting point instead of random noise. The VAE decoder is then used as usual to bring the generated image back to pixel space. This highlights that the VAE remains integral to the editing process even when inversion is employed. Furthermore, inversion methods themselves introduce unique noise and challenges that contribute to iterative degradation, explaining why they are not inherently more resilient than the Vanilla-VAE in such scenarios.

To further explore the degradation patterns associated with DDIM-inversion, we also replicate the iterative text-guided image editing task described previously, using NTI [MHA*23] and P2P editing [HMT*22]. As detailed in the Supplementary Material, DDIM-inversion-based editing methods also suffer from iterative degradation, similar to methods that bypass inversion entirely. While inversion methods are valuable components of many diffusion-based editing pipelines, they do not inherently solve the challenges posed by multi-method iterative image editing, as described in this paper. Conversely, REED-VAE directly mitigates these issues, enabling high-fidelity editing across both pixel and latent spaces without requiring rigid workflows or sacrificing flexibility. This ensures compatibility with a broader range of editing techniques, supporting creative and iterative editing scenarios.

6. Conclusion

We introduce a novel problem setting of *comprehensive iterative image editing* where a user can perform iterative edit operations on a real image, each time using the previous output to perform another edit operation using the same or a different model or conventional editing techniques. We solve the problem of accumulating artifacts with our REED-VAE, which implements a novel iterative training algorithm to enhance the VAE's ability to reconstruct images faithfully over many iterations. We demonstrate the ease of using REED-VAE with any Stable Diffusion-based editing model in place of the vanilla VAE, and its marked effectiveness in improving image quality retention over iterations. We make REED-VAE publicly available, and hope that it will serve as a valuable contribution to the community.

Limitations and future work There are, nevertheless, limitations to our method in its current form. As with any VAE, the reconstruction remains imperfect, and after a very large number of iterations (30+), the REED-VAE will also begin to deteriorate. In

the future, we would like to explore how leveraging REED-VAE to generate synthetic training data for diffusion models can help alleviate the model collapse problem, which is prevalent in iterative generative processes [YHW*24]. By using REED-VAE to better align the diffusion latent space, we anticipate improved stability and performance in downstream editing tasks. As well, applying the REED training algorithm to improve the VAEs of newer latent diffusion models (e.g., SDXL [PEL*23], SD3 [EKB*24], and Flux [Lab23], as discussed in the Supplementary Material) will provide additional insights into the robustness and generalizability of our approach. These models employ advanced architectures and improved, 16-channel latent spaces, which may further benefit from REED-VAE's iterative stability.

Ethical considerations We acknowledge that all diffusion-based image editing techniques inevitably raise ethical concerns, and may reflect biases inherent in the training data used for the underlying model. It follows that the method presented in this paper, which allows more of such edits to be performed on a single image, can amplify these concerns. When implementing and using such models and techniques, it is crucial to establish proper safeguards, particularly concerning permissible prompts and guidance, to prevent malicious use and ensure compliance with legal standards. In preliminary tests, we observe that our model maintains watermarks aiming to detect synthetic images [WKGG24]. We are also actively researching methods for detecting synthetic images and videos [SF24, Kna22, AFFA20]

Acknowledgments We thank Almog Friedlander for the thoughtful suggestions and discussions that helped improve our research. This work was supported by the Israel Science Foundation (Ggrant No. 1574/21) and by the Joint NSFC-ISF Research Grant (no. 3077/23).

References

[AFFA20] AGARWAL S., FARID H., FRIED O., AGRAWALA M.: Detecting deep-fake videos from phoneme-viseme mismatches. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2020), pp. 2814–2822. doi:10.1109/CVPRW50498.2020.00338. 10

[AFL23] AVRAHAMI O., FRIED O., LISCHINSKI D.: Blended latent diffusion. *ACM Trans. Graph. 42*, 4 (jul 2023). URL: https://doi.org/10.1145/3592450, doi:10.1145/3592450. 1, 2, 3, 6

[ALF22] AVRAHAMI O., LISCHINSKI D., FRIED O.: Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 18208–18218. 2

[BHE23] BROOKS T., HOLYNSKI A., EFROS A. A.: Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 18392–18402. 1, 2, 3, 6, 7, 8, 9, 13

[BLCW09] BENGIO Y., LOURADOUR J., COLLOBERT R., WESTON J.: Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning* (2009), pp. 41–48. 5

[CVSC22] COUAIRON G., VERBEEK J., SCHWENK H., CORD M.: Diffedit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427* (2022). 2, 7, 8, 9, 13

[CWQ*23] CAO M., WANG X., QI Z., SHAN Y., QIE X., ZHENG Y.: 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)* (October 2023), pp. 22560–22570. 2

[DN21] DHARIWAL P., NICHOL A.: Diffusion models beat gans on image synthesis. *Advances in neural information processing systems 34* (2021), 8780–8794. 3, 9

[EKB*24] ESSER P., KULAL S., BLATTMANN A., ENTEZARI R., MÜLLER J., SAINI H., LEVI Y., LORENZ D., SAUER A., BOESEL F., ET AL.: Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International Conference on Machine Learning* (2024). 2, 4, 10, 13, 14

[ERO21] ESSER P., ROMBACH R., OMMER B.: Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2021), pp. 12873–12883. 2

[GAA*22] GAL R., ALALUF Y., ATZMON Y., PATASHNIK O., BERMANO A. H., CHECHIK G., COHEN-OR D.: An image is worth one word: Personalizing text-to-image generation using textual inversion, 2022. URL: https://arxiv.org/abs/2208.01618, doi:10.48550/ARXIV.2208.01618. 2

[HHL*24] HUANG Y., HUANG J., LIU Y., YAN M., LV J., LIU J., XIONG W., ZHANG H., CHEN S., CAO L.: Diffusion model-based image editing: A survey. *arXiv preprint arXiv:2402.17525* (2024). 2

[HJA20] HO J., JAIN A., ABBEEL P.: Denoising diffusion probabilistic models. *Advances in neural information processing systems 33* (2020), 6840–6851. 2

[HMT*22] HERTZ A., MOKADY R., TENENBAUM J., ABERMAN K., PRITCH Y., COHEN-OR D.: Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626* (2022). 2, 9, 14

[HRU*17] HEUSEL M., RAMSAUER H., UNTERTHINER T., NESSLER B., HOCHREITER S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems 30* (2017). 7

[JUS*24] JOSEPH K., UDHAYANAN P., SHUKLA T., AGARWAL A., KARANAM S., GOSWAMI K., SRINIVASAN B. V.: Iterative multi-granular image editing using diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (2024), pp. 8107–8116. 3, 4

[JZB*24] JU X., ZENG A., BIAN Y., LIU S., XU Q.: Pnp inversion: Boosting diffusion-based editing with 3 lines of code. *International Conference on Learning Representations (ICLR)* (2024). 2

[KLZ*24] KU M., LI T., ZHANG K., LU Y., FU X., ZHUANG W., CHEN W.: Imagenhub: Standardizing the evaluation of conditional image generation models. In *The Twelfth International Conference on Learning Representations* (2024). URL: https://openreview.net/forum?id=OuV9ZrkQlc. 5, 7, 14

[Kna22] KNAFO G.: *Fakeout: Leveraging out-of-domain self-supervision for multi-modal video deepfake detection*. Master's thesis, Reichman University (Israel), 2022. 10

[KW13] KINGMA D. P., WELLING M.: Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013). 2, 4, 5

[KZL*23] KAWAR B., ZADA S., LANG O., TOV O., CHANG H., DEKEL T., MOSSERI I., IRANI M.: Imagic: Text-based real image editing with diffusion models. In *Conference on Computer Vision and Pattern Recognition 2023* (2023). 2

[Lab23] LABS B. F.: Flux. https://github.com/black-forest-labs/flux, 2023. URL: https://github.com/black-forest-labs/flux. 10, 13, 14

[LDR*22] LUGMAYR A., DANELLJAN M., ROMERO A., YU F., TIMOFTE R., VAN GOOL L.: Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2022), pp. 11461–11471. 2

[MHA*23] MOKADY R., HERTZ A., ABERMAN K., PRITCH Y., COHEN-OR D.: Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 6038–6047. 3, 9, 14, 19, 20

[MHS*21] MENG C., HE Y., SONG Y., SONG J., WU J., ZHU J.-Y., ERMON S.: Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073* (2021). 2

[NDR*21] NICHOL A., DHARIWAL P., RAMESH A., SHYAM P., MISHKIN P., MCGREW B., SUTSKEVER I., CHEN M.: Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741* (2021). 2

[PEL*23] PODELL D., ENGLISH Z., LACEY K., BLATTMANN A., DOCKHORN T., MÜLLER J., PENNA J., ROMBACH R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952* (2023). 10, 13, 14

[PGXH23] PAN Z., GHERARDI R., XIE X., HUANG S.: Effective real image editing with accelerated iterative diffusion inversion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2023), pp. 15912–15921. 2

[PKSZ*23] PARMAR G., KUMAR SINGH K., ZHANG R., LI Y., LU J., ZHU J.-Y.: Zero-shot image-to-image translation. In *ACM SIGGRAPH 2023 Conference Proceedings* (2023), pp. 1–11. 2, 3

[PKY23] PARK J., KWON G., YE J. C.: Ed-nerf: Efficient text-guided editing of 3d scene using latent space nerf. *arXiv preprint arXiv:2310.02712* (2023). 2

[RBL*22] ROMBACH R., BLATTMANN A., LORENZ D., ESSER P., OMMER B.: High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2022), pp. 10684–10695. 2, 4, 5, 7, 8, 9, 13, 14

[RDN*22] RAMESH A., DHARIWAL P., NICHOL A., CHU C., CHEN M.: Hierarchical text-conditional image generation with clip latents. arxiv 2022. *arXiv preprint arXiv:2204.06125* (2022). 2

[SBV*22] SCHUHMANN C., BEAUMONT R., VENCU R., GORDON C., WIGHTMAN R., CHERTI M., COOMBES T., KATTA A., MULLIS C., WORTSMAN M., ET AL.: Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems 35* (2022), 25278–25294. 13

[SCS*22] SAHARIA C., CHAN W., SAXENA S., LI L., WHANG J., DENTON E. L., GHASEMIPOUR K., GONTIJO LOPES R., KARAGOL AYAN B., SALIMANS T., ET AL.: Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems 35* (2022), 36479–36494. 2

[Sei20] SEITZER M.: pytorch-fid: FID Score for PyTorch. https://github.com/mseitzer/pytorch-fid, August 2020. Version 0.3.0. URL: https://github.com/mseitzer/pytorch-fid. 7, 13

[SF24] SINITSA S., FRIED O.: Deep image fingerprint: Towards low budget synthetic image detection and model lineage analysis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (2024), pp. 4067–4076. 10

[SME20] SONG J., MENG C., ERMON S.: Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020). 3, 9

[SXP*23] SHI Y., XUE C., PAN J., ZHANG W., TAN V. Y., BAI S.: Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. *arXiv preprint arXiv:2306.14435* (2023). 3

[VDOV*17] VAN DEN OORD A., VINYALS O., ET AL.: Neural discrete representation learning. *Advances in neural information processing systems 30* (2017). 2

[vPPL*22] VON PLATEN P., PATIL S., LOZHKOV A., CUENCA P., LAMBERT N., RASUL K., DAVAADORJ M., NAIR D., PAUL S., BERMAN W., XU Y., LIU S., WOLF T.: Diffusers: State-of-the-art diffusion models. https://github.com/huggingface/diffusers, 2022. 13

[WBSS04] WANG Z., BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing 13*, 4 (2004), 600–612. 7

[WKGG24] WEN Y., KIRCHENBAUER J., GEIPING J., GOLDSTEIN T.: Tree-rings watermarks: Invisible fingerprints for diffusion images. *Advances in Neural Information Processing Systems 36* (2024). 10

[YGZ*23] YANG B., GU S., ZHANG B., ZHANG T., CHEN X., SUN X., CHEN D., WEN F.: Paint by example: Exemplar-based image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2023), pp. 18381–18391. 1, 2, 3, 6, 7, 8, 9, 13

[YHW*24] YOON Y., HU D., WEISSBURG I., QIN Y., JEONG H.: Model collapse in the self-consuming chain of diffusion finetuning: A novel perspective from quantitative trait modeling. *arXiv preprint arXiv:2407.17493* (2024). 10

[YZLL23] YANG S., ZHOU Y., LIU Z., LOY C. C.: Rerender a video: Zero-shot text-guided video-to-video translation. In *SIGGRAPH Asia 2023 Conference Papers* (2023), pp. 1–11. 3, 4

[ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 586–595. 5, 7

[ZMC*23] ZHANG K., MO L., CHEN W., SUN H., SU Y.: Magicbrush: A manually annotated dataset for instruction-guided image editing. In *Advances in Neural Information Processing Systems* (2023). 2, 3, 7, 8, 9

[ZYF*24] ZHANG S., YANG X., FENG Y., QIN C., CHEN C.-C., YU N., CHEN Z., WANG H., SAVARESE S., ERMON S., ET AL.: Hive: Harnessing human feedback for instructional visual editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2024), pp. 9026–9036. 2

REED-VAE: RE-Encode Decode Training for Iterative Image Editing with Diffusion Models
Supplementary Material

Appendix A: Implementation details

Training

All experiments conducted are based on the released `v2.1` of Stable Diffusion [RBL*22] along with the default VAE using a single NVIDIA A100 GPU card. We finetune the VAE using the same Diffusers [vPPL*22] implementation named `"AutoencoderKL"`. In accordance with the training setting used for both the original VAE and the original diffusion model, we finetune our REED-VAE on a subset of the LAION-5B dataset [SBV*22]. During training, we preprocess the image resolution to 512×512 and train for 35 epochs, which took approximately 1 day. Scaling parameters α and β were used to scale the LPIPS and D_{KL} terms of the training loss, respectively. To train the final model, we set $\alpha = 0.01$ and $\beta = 1$.

Backpropagation strategy

Initially, we backpropagated through all iterations, calculating gradients for each intermediate step. Although this approach achieved maximal learning at each iteration, it was very memory-intensive and limited the maximum number of iterations (k) to approximately 7 on our A100 GPU, even with gradient checkpointing. To address this, we optimized memory usage by computing gradients only for the final iteration, significantly reducing computational overhead and enabling training with larger k values. This adjustment leverages the dynamic incrementation in our loss design, which encourages the model to learn progressively from intermediate iterations without requiring full-gradient computation at each step.

Experiments and comparisons

When calculating metrics, we try to isolate errors and noise that occur due to the iterative autoencoding process from those that occur due to imperfect performance by the editing model. To do this, we compute metrics between the given target image (one of iterations 5,15, or 25) and x^1 — *not* to the source image. This guarantees that (1) we always compare images from aligned edit operations (i.e., edits in the same direction, such as changing the bus into a car and not the other way around) and (2) metrics are more dependent on the model's ability to maintain image quality over iterations than the model's general editing capabilities. In other words, if a model performs a non-sensical edit operation from the given inputs, as long as it is consistent (which it should be if it does not degrade images), then this alone should not harm its performance in our experiments.

Metric calculations

For all reported metrics, MSE, PSNR, and LPIPS are computed with each image sample normalized to the [0,1] range prior to evaluation, ensuring consistency in comparison. Other metrics are computed as per their standard definitions. FID is calculated using the pytorch-fid implementation [Sei20].

Iterative prompt list used for iterative text-guided image editing

We provide some examples of iterative prompts used in the iterative text-guided image editing task in Table. A1. The full enhanced dataset will be made public along with our code.

Table A1: *Example of edit prompts and corresponding reverse edit prompts used iteratively to evaluate InstructPix2Pix [BHE23] and DiffEdit [CVSC22]*

	Prompt	Reverse Prompt
1	Change the frisbee into a ball	Change the ball into a frisbee
2	Put a lion in the place of the donkey	Put a donkey in the place of the lion
3	Add a pedestrian	Remove the pedestrian
4	Make it a black sheep	Make it a white sheep
5	Replace the coffee with beer	Replace the beer with a coffee

Varying metric scales across editing methods

In our main editing experiments, the scale of improvements observed with REED may vary due to editing methods differing in conditioning and inputs (text/mask/image), scope (local/global edits), and VAE use. For instance, DiffEdit [CVSC22] automatically generates latent masks from text prompts, introducing ambiguity regarding the edit location especially as noise increases in higher iterations. Such ambiguity can result in edits being applied to different regions of the image, potentially inflating computed metrics - this will be noticeable even when REED is used, as metrics are evaluated against the first edit iteration. In contrast, PbE [YGZ*23] employs predefined masks, ensuring that edits remain localized and consistent regardless of accumulated noise. This provides a more controlled editing scenario, reducing variability in computed metrics.

Appendix B: Additional Experiments

Comparison with newer latent diffusion models

At the time of writing, Stable Diffusion 2.1 (SD 2.1) [RBL*22] was one of the most advanced and widely-used diffusion models for image generation and editing. While SD2 remains an important and widely-used model, more recent LDM variants such as SDXL [PEL*23], SD3 [EKB*24], and Flux [Lab23], have since been released. Specifically, SD3 and Flux use more advanced latent spaces with improved, 16-channel VAEs that may behave differently from the 4-channel VAE used in SD2. We conduct additional experiments to confirm that a similar iterative degradation problem does occur in these newer models as well. We perform an iterative encode/decode task on the images in the ImagenHub dataset (179 images) and compute metrics with the original image. The results (Table A2, Figure 7, Figure 8) show that in all the newer models, noise and artifacts still accumulate after 5+ iterations. Flux seems to be the most resistant, yet still accumulates a

Table A2: *Comparison of performance metrics for several state-of-the-art latent diffusion models and Stable Diffusion 2.1 with/without our REED-VAE. The metrics are calculated on the ImagenHub dataset [KLZ*24] (179 images) and are reported for various iteration steps (5,15,25) on an iterative encode/decode task (without editing). Despite the more advanced latent spaces in models such as SD3 and Flux (with 16 channels), these newer models still exhibit the problem of iterative degradation.*

Model	MSE ↓			LPIPS ↓			SSIM ↑			FID ↓			PSNR ↑		
	5	15	25	5	15	25	5	15	25	5	15	25	5	15	25
SD3 [EKB*24]	0.0025	0.013	0.031	0.11	0.55	0.76	0.83	0.51	0.26	2.53	24.67	68.79	26.7	19.07	15.19
SDXL [PEL*23]	0.0026	0.0067	0.013	0.19	0.45	0.61	0.78	0.64	0.52	7.33	18.27	29.71	26.9	22.12	19.20
Flux.1 [Lab23]	0.0014	0.0075	0.017	**0.064**	0.27	0.56	**0.90**	0.74	0.53	**1.02**	9.725	28.62	28.9	21.51	17.88
Vanilla SD 2.1 [RBL*22]	0.0031	0.013	0.034	0.19	0.55	0.71	0.76	0.49	0.26	2.35	45.95	133.9	26.0	19.30	14.83
+ REED	**0.0011**	**0.0042**	**0.0086**	0.075	**0.18**	**0.25**	0.89	**0.76**	**0.68**	1.08	**2.790**	**3.873**	**30.5**	**24.10**	**20.93**

fair amount of noise. When SD2 is paired with our REED-VAE, it consistently outperforms even these newer LDM variants in most metrics. Flux is the only model that surpasses SD2 + REED-VAE in certain cases. Specifically, the updated Flux model demonstrates improved performance in terms of LPIPS and FID at the 5 iteration mark, though this advantage is not sustained at later iterations. Therefore, despite their more advanced latent spaces, at higher iterations these newer models still exhibit the problem of iterative degradation, and the REED training algorithm trained on their respective VAEs will likely improve performance in them as well. Even when trained on the simpler SD2, the pipeline with REED-VAE is able to outperform all models at higher iterations.

More ablation results

We provide a visual example of the improvement provided by each component in our final REED-VAE in Figure.9.

Comparison with Inversion-Based Methods

In addition to the iterative inversion experiment discussed in the main paper, we also provide results for iterative NTI [MHA*23] combined with P2P editing [HMT*22]. As shown in Figure 6, NTI introduces significant artifacts over iterations that are noticably reduced when NTI is paired with REED-VAE. The experiment follows a similar iterative setup: NTI is first used to invert the image back to its latent representation, after which P2P editing is applied based on the provided text prompt. These iterative steps are repeated for 6 iterations in the example figure. The results demonstrate that NTI-based methods struggle to maintain fidelity across iterations, accumulating noise and distortions. These findings highlight the limitation of iterative editing that exists in DDIM-Inversion-based methods as well as non-inversion diffusion-based editing methods, demonstrating the importance and relevance of REED-VAE even with such newer models.

We also provide more extensive results from the iterative inversion experiment in Figure 10 and Figure 11. In the full sequence, it is evident that using NTI with the Vanilla-VAE (Figure 10) causes artifacts and noise patterns to progressively worsen in the first 10 iterations, until arriving at near complete noise at iteration 13. Likely due to the involvement of the source prompt during inversion, the image is able to "bounce back", but this time severely diverged from the original imaging, displaying heavy distortions and color shifts. Through iteration 25, the image continues to build noise in a more typical manner. When NTI is combined with REED-VAE Figure 11, the fidelity to the input image is maintained throughout the entire 25 iterations. Although there are some minor color distortions, the overall level of noise and artifacts is significantly reduced. The improved fidelity to the input image with REED-VAE already visible in Inversion 1, as well as the lack of noise dominance around Inversion 13, suggest REED-VAE may contribute to a more efficient latent space organization that is more conducive to image editing and resilient to iterative operations.

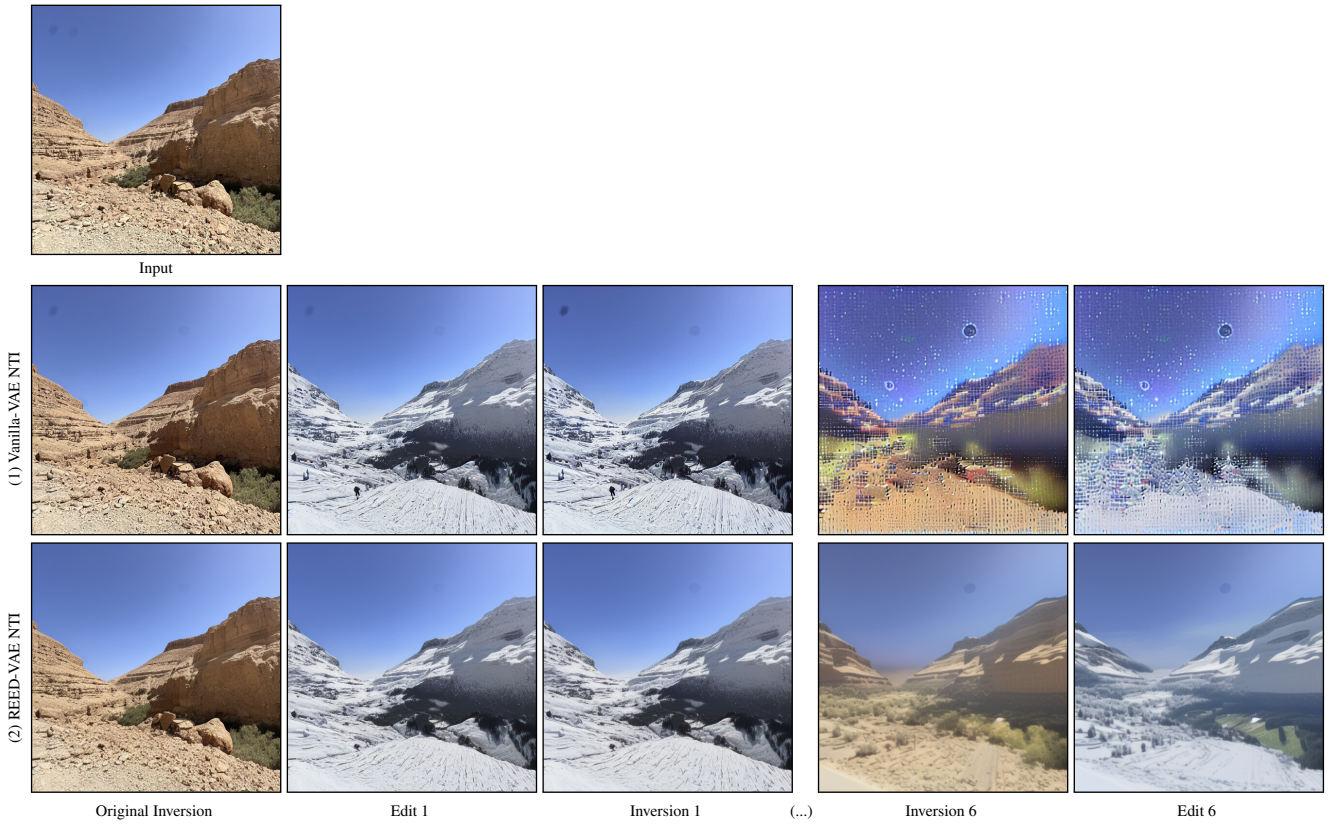

Figure 6: *Iterative edits using Null-Text Inversion. Prompts: "a landscape with **desert** mountains" → "a landscape with **snowy** mountains". Despite regenerating latents through the inversion process, visual artifacts accumulate, particularly in later iterations (e.g. noise patterns and loss of fidelity to the original image). This illustrates that DDIM inversion-based methods do not mitigate the degradation that occurs in iterative editing tasks, underscoring the need for REED-VAE.*

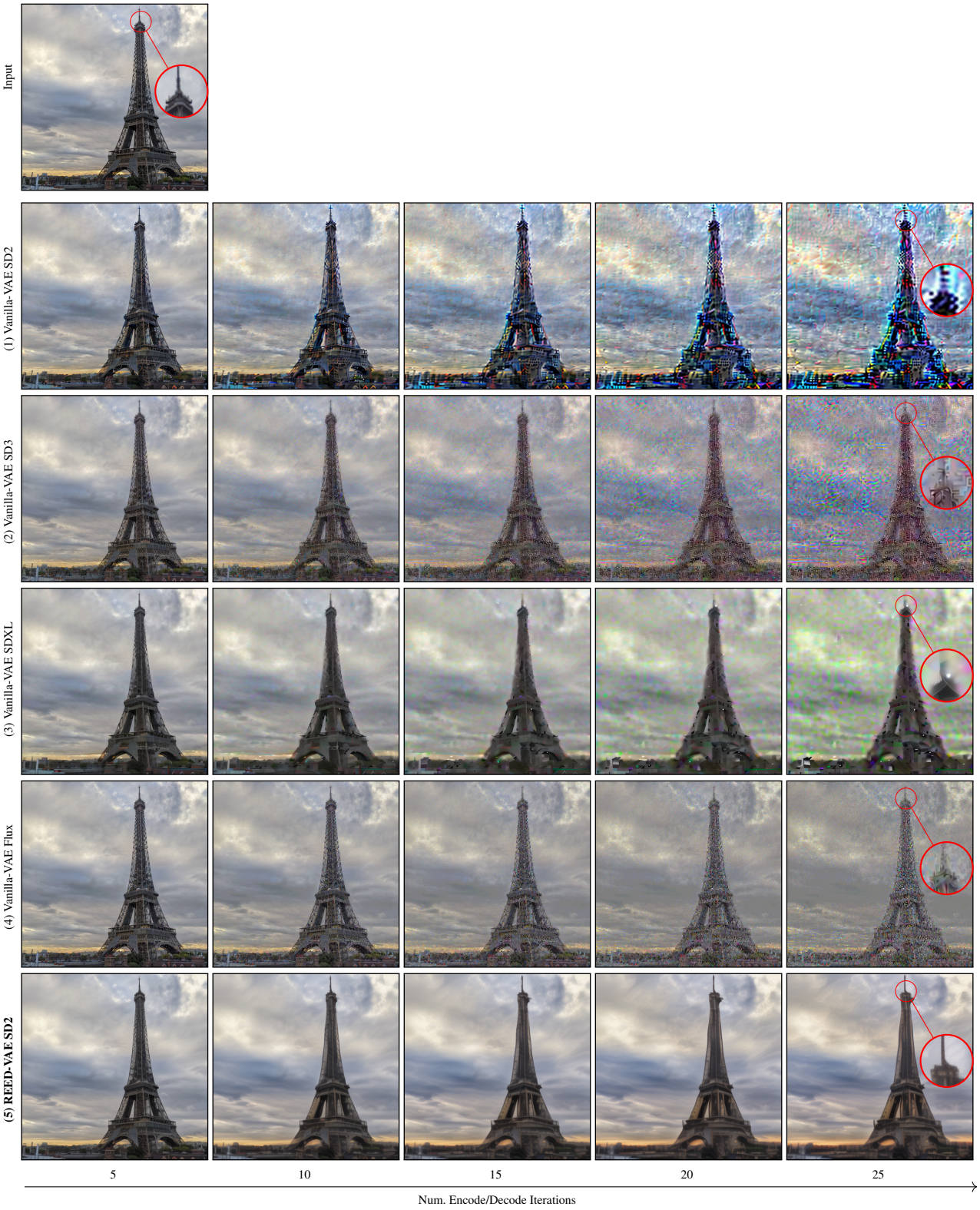

Figure 7: *Comparison on iterative encode/decode task with more recent latent diffusion models, reported at 5,10,15,20,25 iterations. REED-VAE is able to outperform even the newest models with 16-channel latent spaces, suggesting training these new model's VAEs with the REED algorithm can improve them even further.*

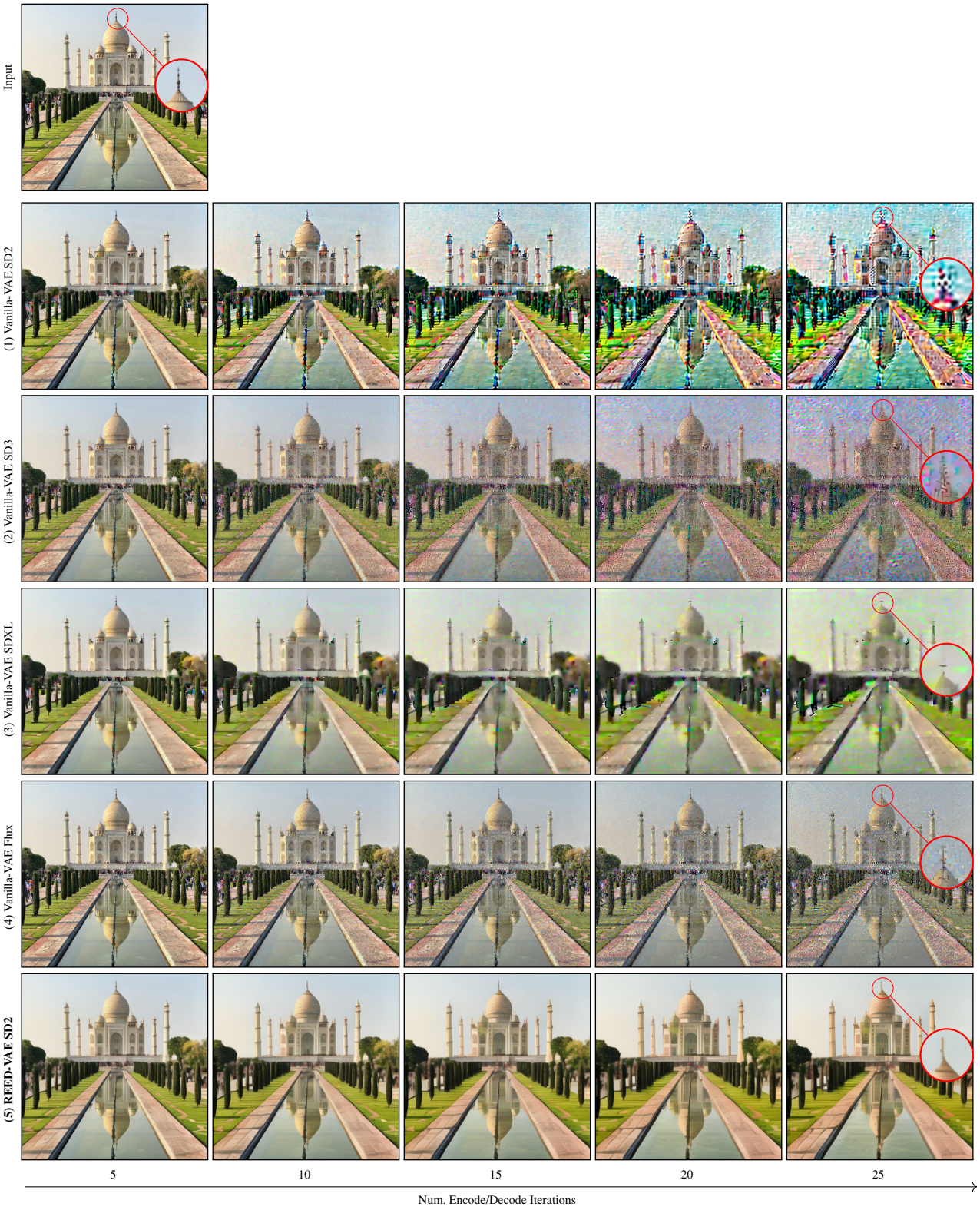

Figure 8: *Additional comparison on iterative encode/decode task with more recent latent diffusion models, reported at 5,10,15,20,25 iterations.*

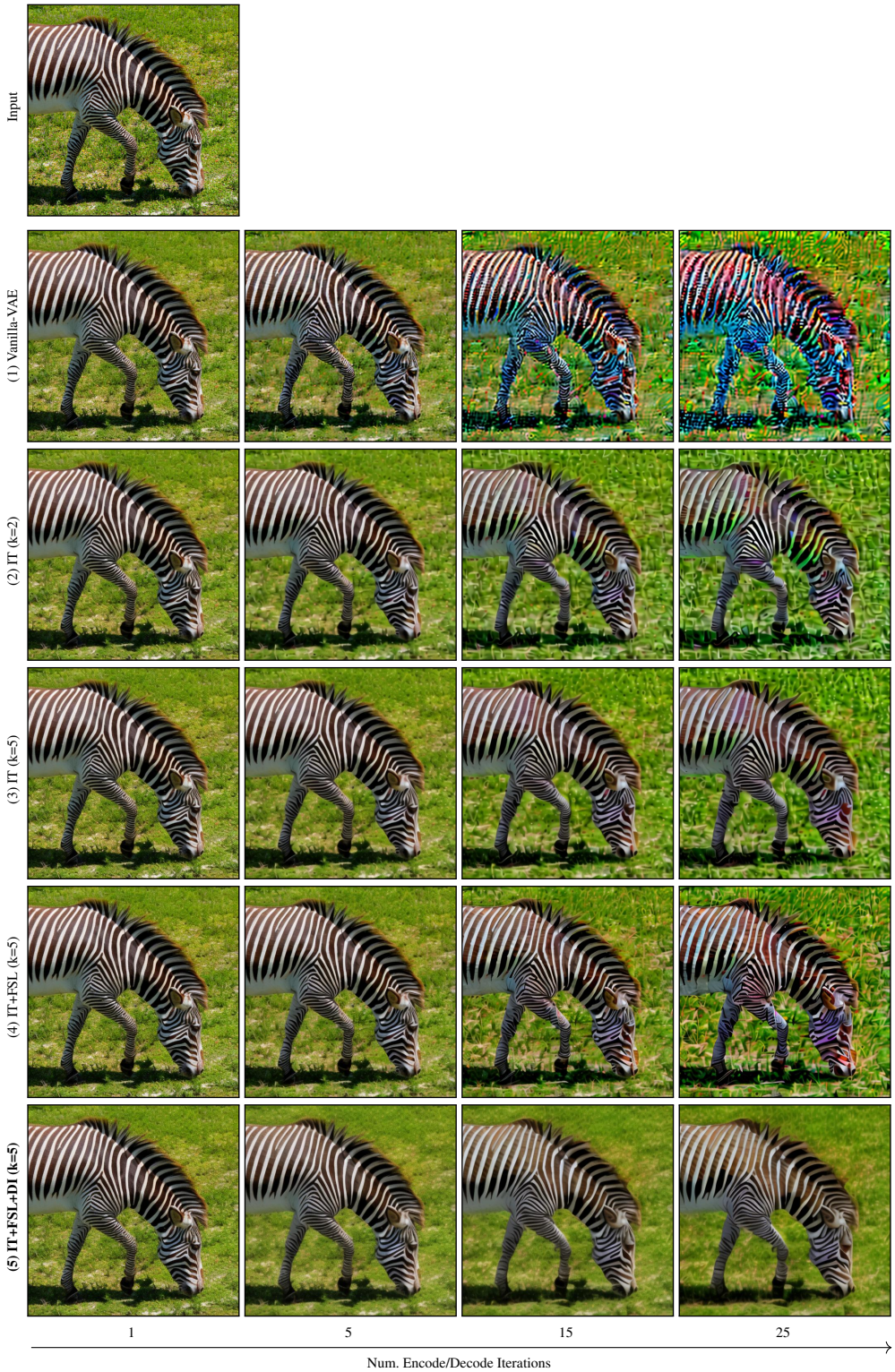

Figure 9: *Ablation on individual components of REED-VAE on a sample image from our evaluation set. We compare the Vanilla-VAE (1), REED with static Iterative Training (IT) at $k = 2$ (2) and $k = 3$ (3), REED with IT at $k = 5$ and the First-Step Loss (FSL) (4), and the full REED-VAE model with IT at $k = 5$, FSL, and Dynamic Incrementation (DI). It can be seen that the full REED-VAE model (IT+DI+FSL ($k = 5$)) is best able to maintain image features and colors, even at 25 iterations.*

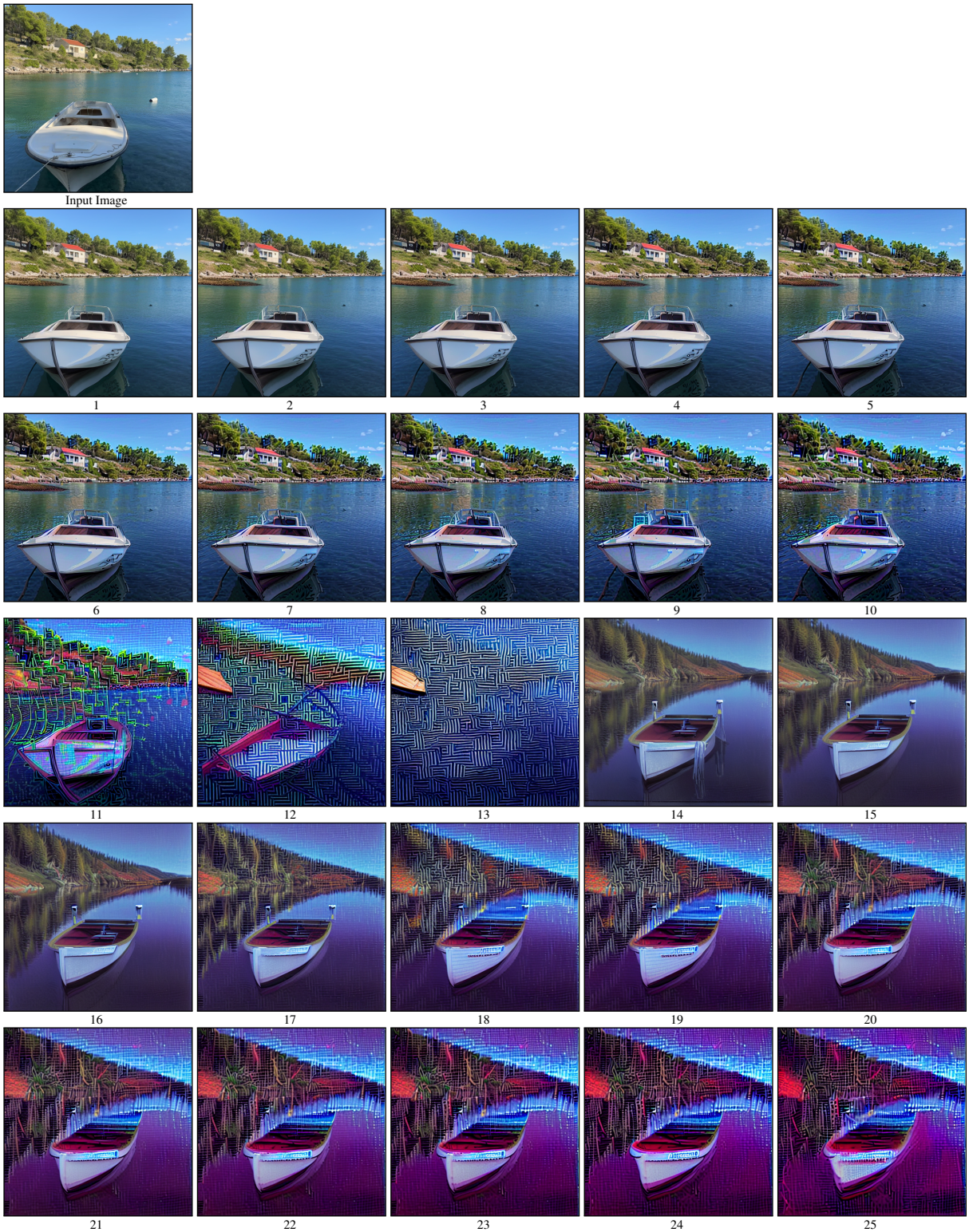

Figure 10: *Full sequence of iterative inversion reconstructions using Vanilla-VAE with Null-Text Inversion (NTI) [MHA*23]. The input image undergoes NTI-based inversion followed by reconstruction with the same source prompt for 25 iterations. Early iterations (1-10) retain reasonable fidelity, but progressive iterations introduce artifacts, noise patterns, and distortions, culminating in severe degradation by iteration 25.*

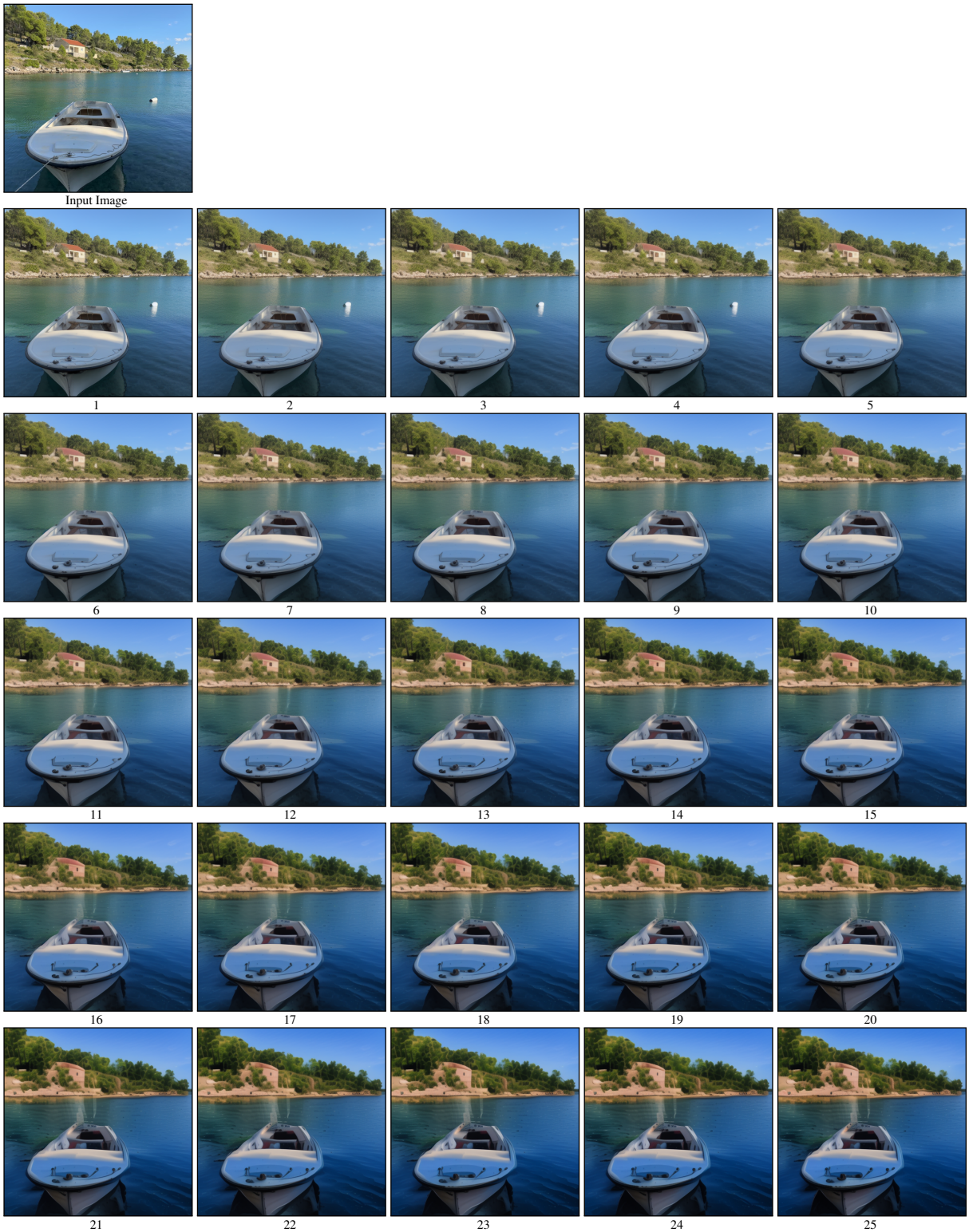

Figure 11: *Full sequence of iterative inversion reconstructions using REED-VAE with Null-Text Inversion (NTI) [MHA*23]. The input image undergoes NTI-based inversion followed by reconstruction with the same source prompt for 25 iterations. REED-VAE maintains high fidelity to the input image across all iterations, with significantly reduced artifacts, noise, and distortions even at high iterations.*