Watch Your Steps: Local Image and Scene Editing by Text Instructions

Ashkan Mirzaei[1,2] Tristan Aumentado-Armstrong[1,2,4] Marcus A. Brubaker[1,3,4]
Jonathan Kelly[2] Alex Levinshtein[1] Konstantinos G. Derpanis[1,3,4] Igor Gilitschenski[2,4]
[1]Samsung AI Centre Toronto [2]University of Toronto [3]York University [4]Vector Institute for AI

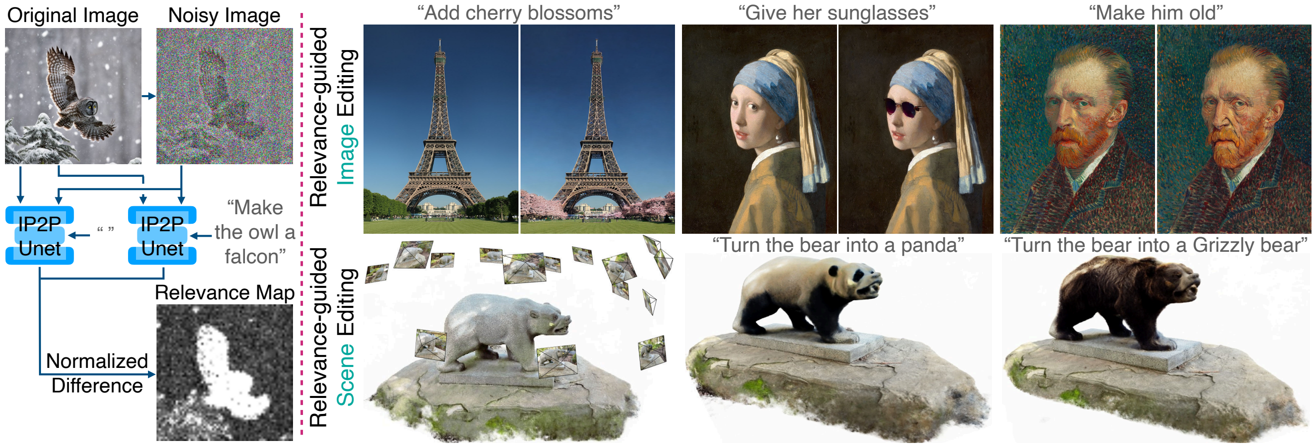

Figure 1. Overview of the calculation of the relevance map (left inset), and sample outputs on image (top-right inset) and neural radiance field (bottom-right inset) editing guided by the relevance. Given an image or a Neural Radiance Field (NeRF), our goal is to change the input according to a textual instruction. The relevance map is the disagreement between noise predictions with and without the instruction. For both image and scene editing, we use the relevance map to confine the changes to the most relevant region, according to the edit text.

Abstract

Denoising diffusion models have enabled high-quality image generation and editing. We present a method to localize the desired edit region implicit in a text instruction. We leverage InstructPix2Pix (IP2P) and identify the discrepancy between IP2P predictions with and without the instruction. This discrepancy is referred to as the relevance map. The relevance map conveys the importance of changing each pixel to achieve the edits, and is used to to guide the modifications. This guidance ensures that the irrelevant pixels remain unchanged. Relevance maps are further used to enhance the quality of text-guided editing of 3D scenes in the form of neural radiance fields. A field is trained on relevance maps of training views, denoted as the relevance field, defining the 3D region within which modifications should be made. We perform iterative updates on the training views guided by rendered relevance maps from the relevance field. Our method achieves state-of-the-art performance on both image and NeRF editing tasks. Project page.

1. Introduction

The crucial role of images in various aspects of modern societies, including social media, marketing, and education, naturally introduces a desire for automated generative approaches for image editing [6, 9, 17, 37, 39]. Neural radiance fields [42] (NeRFs) are increasingly accessible [8, 48, 54] and popular as an intuitive visualization modality, thus editing NeRFs is also receiving significant attention [15, 43, 44]. The remarkable success of denoising diffusion models [18, 63] in generating high-quality images [12, 19, 57, 59, 66] from text [2, 49, 55, 58] has led to diffusion models being adopted for image editing [1, 6, 9, 17, 23, 39, 47, 53]. Recently, InstructNeRF2NeRF (IN2N) [15] demonstrated how to leverage InstructPix2Pix [6] (IP2P) for editing NeRFs [42]. We argue that a notable proportion of image and scene editing tasks can be executed by only local modifications. Consider the top-right example in Figure 1, to *“make him old”*. Technically, any output depicting an old man satisfies the instruction. To maintain fidelity to the input and avoid unnecessary variability, it is crucial to confine modifications within local boundaries (within his face in this example) and naturally prefer parsimonious edits. By keeping changes within the relevant region, the integrity of the original input is better preserved, while we can ensure that the desired edit is accurately reflected in the output.

Despite the promising results, diffusion-based image editors generally lack a mechanism to automatically localize the edit regions. These methods either ask users for a mask [37], rely on the global information kept in a noisy input as a starting point [39], or condition the denoiser on the input [6]. Nevertheless, all of these methods tend to over-edit [6, 9]. Relying on IP2P to iteratively update NeRF's training dataset, IN2N [15] over-edits scenes. Recently, DiffEdit [9] proposed using the difference between the noise predictions conditioned on captions to localize image edits; however, it is slow due to *denoising diffusion implicit model* [64] (DDIM) inversion, its quality is inferior to IP2P [6], and it requires both input and output captions.

In this paper, we provide an approach to predict the scope implicit in edit instructions to localize image edits. We denote the discrepancy between the noise predictions by IP2P conditioned on the instruction and an empty text as the *relevance map*. Binarizing the relevance map gives the mask of the region that should be edited. We force the denoising process to not change the unmasked pixels by replacing the unmasked region after each denoising iteration with the noisy input [37]. For NeRF editing by iterative dataset updates [15], we leverage the power of relevance maps to localize the edits. Note that, across different views, the relevance maps can be slightly inconsistent. To ensure 3D consistency, we train a field on relevance maps from training views, called the *relevance field*. Rendered views from the relevance field are then binarized and used as masks for editing training views. Our approach achieves state-of-the-art performance on both image and NeRF editing tasks.

In summary, (1) we present *relevance maps* to predict the scope of an editing instruction on an image, (2) we use the relevance maps to localize instruction-based image editing, and (3) we lift the maps into 3D by *relevance fields* to leverage the localization in scene editing.

2. Related work

Diffusion models for image editing. Diffusion models have shown impressive performance in image synthesis [12, 18, 19, 57, 59, 63, 66]. Text-to-image diffusion models are able to generate high-quality images based on captions [49, 53, 55, 58]. Motivated by this success, pre-trained diffusion models have been used to edit images based on text descriptions [1, 17, 23, 50, 53]. SDEdit [39] adds noise to input images and denoises them conditioned on a desired description, but lacks a mechanism to keep the details of the original image. DiffEdit [9] uses of the disagreement in predictions of stable diffusion [55] with input and output captions, but can not handle instructions and fails on more complex captions. Recently, IP2P [6] uses Prompt2Prompt [17] to create a dataset, and trains a denoiser conditioned on edit instructions and the original image. IP2P [6] outperforms the previous methods, but tends to over-edit images. Simply increasing the image guidance scale or reducing the text guidance scale has adverse effects on the region that actually should be edited. We propose a method to predict the relevance of each latent pixel to the edit, and use it to limit the scope of the edit.

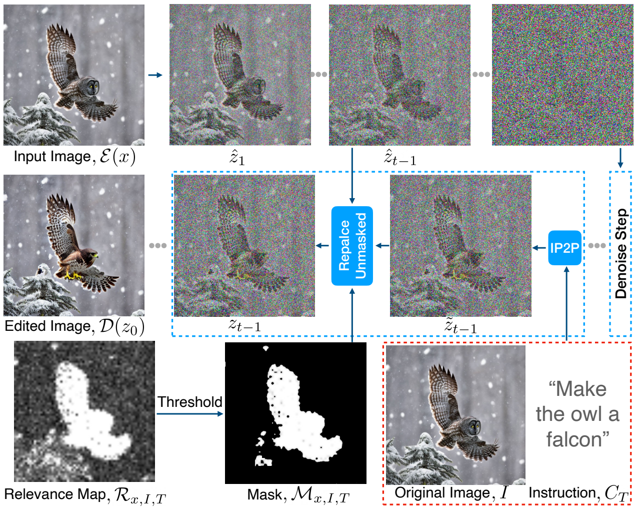

Figure 2. Overview of a denoising step for image editing via relevance-guidance. The relevance map is binarized to get the edit mask. After denoising the output of the last stage with IP2P, the unmasked pixels are swapped with the noisy input to ensure consistency to the input throughout the process.

Editing neural fields. The advent of NeRFs [42] powered by positional encoding [14, 67, 70] has led to significant popularity of neural rendering models [69]. NeRFs are getting faster [5, 7, 8, 16, 24, 27, 48, 54, 60, 78], and less data-intensive [20, 30, 31, 35, 51, 74, 75, 79], with improved rendering quality [3, 4, 11, 32, 71]. The popularity of NeRFs naturally introduces a desire for editing tools. Recent works [10, 21, 22, 26, 28, 29, 36, 40, 45, 65, 72, 77, 80] provide NeRF editing approaches, but are typically limited to simple scenes or objects, or perform niche editing tasks. More works [33, 43, 44, 76] provide 3D inpainting methods to remove unwanted objects from NeRFs. IN2N [15] introduced leveraging IP2P [6] to edit scenes based on text instructions. Although promising, the outputs lack sharpness due to: i) the spatial ambiguity caused by the IP2P decoder while upsampling latent pixels to patches, and ii) the view-inconsistency due to the 3D-unawareness of IP2P. To alleviate these problems, we leverage our relevance-guided image editor combined with a 3D relevance field to localize the edits in the 3D space and improve consistency.

3. Background

Neural radiance fields. NeRFs represent a 3D scene as a neural field, $f_\theta : (x, d) \rightarrow (c, \sigma)$, mapping a 3D coordinate, $x \in \mathbb{R}^3$ and a view direction, $d \in \mathbb{S}^2$, to a colour, $c \in [0, 1]^3$, and a density, $\sigma \in \mathbb{R}^+$. The field parameters, θ, are optimized to fit the field representation to multiview posed image sets. The field is paired with a rendering operator, implemented as the quadrature approximation of the classical volumetric rendering integral [38]. For a ray, r, parametrized as $r = o + td$, where o is the origin and d is the view-direction, rendering begins with sampling N points, $\{t_i\}_{i=1}^N$, on the ray between near and far bounds. The rendered colour is then obtained via the volumetric rendering equation, $\widehat{C}(r) = \sum_{i=1}^N w_i c_i$, where $w_i = T_i(1 - \exp(-\sigma_i \delta_i))$ is the contribution of the i-th point, $\delta_i = t_{i+1} - t_i$ are the adjacent point distances, and $T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)$ is the transmittance.

InstructPix2Pix. Given an image, I, and a text instruction, C_T, describing the edit, IP2P follows the instruction to edit I. IP2P is trained on a dataset where for each I and C_T, a sample edited image, I_{out}, is given. IP2P is based on latent diffusion [56], where a variational autoencoder [25] (VAE) with encoder, $\mathcal{E}$, and decoder, $\mathcal{D}$, is used for improved efficiency and quality. For training, noise, $\epsilon \sim \mathcal{N}(0, 1)$, is added to $z = \mathcal{E}(I_{\text{out}})$ to get the noisy latent, z_t, where the random timestep, $t \in T$, determines the noise level. The denoiser, ϵ_θ, is initialized with stable diffusion [56] weights, and fine-tuned to minimize the diffusion objective, $\mathbb{E}_{I_{\text{out}}, I, \epsilon, t}\left[\|\epsilon - \epsilon_\theta(z_t, t, I, C_T))\|_2^2\right]$. During training, conditions are randomly removed [34] by setting $I = \emptyset_I$ and $C_T = \emptyset_T$ to further allow unconditional denoising. Thus, the strength of the edit can be controlled by the image guidance scale, s_I, and the text guidance scale, s_T. The modified score estimate is then obtained as

$$\begin{aligned}\widetilde{\epsilon}_\theta(z_t, t, I, C_T) = {} & \epsilon_\theta(z_t, t, \emptyset_I, \emptyset_T) \\ & + s_I\big(\epsilon_\theta(z_t, t, I, \emptyset_T) - \epsilon_\theta(z_t, t, \emptyset_I, \emptyset_T)\big) \\ & + s_T\big(\epsilon_\theta(z_t, t, I, C_T) - \epsilon_\theta(z_t, t, I, \emptyset_T)\big). \quad (1)\end{aligned}$$

After training, the denoiser can be used to either generate edited images from pure noise, or to iteratively denoise a noisy version of an input image to get an output.

4. Method

We describe the calculation of the relevance map in § 4.1. The relevance map is to determine the pixels that should be edited, and is used as a form of mask guidance in § 4.2 for localized image editing. In § 4.3, we introduce *relevance fields* to allow similar edit localization for 3D scene editing. Implementation details can be found in § 4.4.

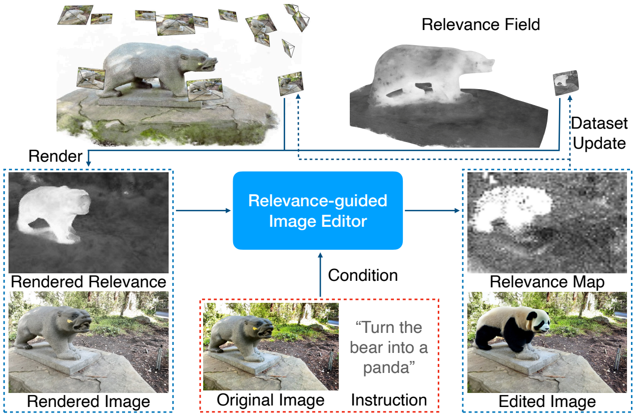

Figure 3. Overview of our relevance-guided NeRF editing method. Iteratively, we take a random view and render it using both the main NeRF and the relevance field. The rendered image is edited guided by the rendered relelvance to only change pixels that are highly relevant to the task. IP2P [6] is used as the backbone of the editing method, and is always conditioned on the initial captures from the scene. This is to prevent drastic drifts from the original scene in the recurrent synthesis process [15]. The relevance-guided image editing module (§ 4.2) returns an edited image and an updated relevance, which are used to update the corresponding training views for the NeRF and the relevance field, respectively.

4.1. Relevance map calculation

Given an image, I, and an edit instruction, C_T, we leverage IP2P [6] to predict the relevance of each pixel to the edit, i.e., the likelihood that a given pixel needs to be changed, based on the editing task. First, we add noise to the encoded image, $\mathcal{E}(I)$, until a fixed timestep, t_{rel}, to obtain the noisy latent,

$$z_{t_{\text{rel}}} = \sqrt{\alpha_{t_{\text{rel}}}}\mathcal{E}(I) + \sqrt{1 - \alpha_{t_{\text{rel}}}}\epsilon, \quad (2)$$

where $\epsilon \sim \mathcal{N}(0, 1)$ is a random noise, and α_t is the noise scheduling factor at timestep t. Note that t_{rel} is a constant noise level used in our method as a hyperparameter. We then use IP2P's noise prediction Unet, ϵ_θ, to get two different predictions: i) the predicted noise conditioned on both the image and the text, $\epsilon_{I,T}(z_{t_{\text{rel}}}) = \epsilon_\theta(z_{t_{\text{rel}}}, t_{\text{rel}}, I, C_T)$, and ii) the predicted noise conditioned only on the image and the empty text as the instruction, $\epsilon_I(z_{t_{\text{rel}}}) = \epsilon_\theta(z_{t_{\text{rel}}}, t_{\text{rel}}, I, \text{“”})$. The difference between $\epsilon_{I,T}(z_{t_{\text{rel}}})$ and $\epsilon_I(z_{t_{\text{rel}}})$ is that only the former is aware of the text prompt. We use the magnitude of the mismatch between these two values as a measurement of the relevance of each pixel to the edit. To this end, we first calculate the absolute difference between the two values, which we call the *relevance map*,

$$\mathcal{R}_{x,I,T} = |\epsilon_{I,T}(z_{t_{\text{rel}}}) - \epsilon_I(z_{t_{\text{rel}}})|. \quad (3)$$

For robustness, we further clamp the outlier values using interquartile range (IQR) with ratio 1.5, and normalize the relevance map between 0 and 1. Figure 1 (left inset) illustrates an overview of the calculation of the relevance mask.

4.2. Relevance-guided image editing

We propose to use the relevance map to guide the generation of the edited image, and to localize the edited region. For a pixel, a high relevance value means that the pixel is likely to be relevant to the edit, and we allow it to change. In contrast, a low relevance map value signals that the pixel is unlikely to require change. We apply a mask threshold, $\tau \in [0, 1]$, on the relevance map to get the edit mask, $\mathcal{M}_{x,I,T} = \mathbb{1}(\mathcal{R}_{x,I,T} \geq \tau)$, enclosing the pixels we allow to be edited. To edit an encoded input image, x, the encoded image, $\mathcal{E}(x)$, is diffused to a fixed noise level, t_{edit}, to get the starting noisy latent, $z_{t_{\text{edit}}}$. The edit noise level, t_{edit}, determines the strength of the edit; setting it to 0 results in the input image being unchanged, while setting it to the maximum diffusion timestep starts the generation from pure noise. Each denoising stage takes a noisy latent, z_t, and denoises it to get z_{t-1}. The denoising step begins with predicting the noise via IP2P to get $\widetilde{\epsilon}_t = \widetilde{\epsilon}_\theta(z_t, t, I, C_T)$. Using $\widetilde{\epsilon}_t$ and the DDIM [64] procedure, the mask-unaware prediction at timestep $t - 1$ is

$$\widetilde{z}_{t-1} = \sqrt{\alpha_{t-1}}\Big(\frac{x_t - \sqrt{1-\alpha_t}\widetilde{\epsilon}_t}{\sqrt{\alpha_t}}\Big) + \sqrt{1-\alpha_{t-1}}\widetilde{\epsilon}_t. \quad (4)$$

The unedited noisy latent of the input image, x, at timestep $t - 1$ would have been $\widehat{z}_{t-1} = \sqrt{\alpha_{t-1}}\mathcal{E}(x) + \sqrt{1-\alpha_{t-1}}\epsilon$. To obtain z_{t-1}, we combine the mask unaware prediction, $\widetilde{z}_{t-1}$, and the unedited noisy latent, $\widehat{z}_{t-1}$, as

$$z_{t-1} = \widetilde{z}_{t-1} \odot \mathcal{M}_{x,I,T} + \widehat{z}_{t-1} \odot (1 - \mathcal{M}_{x,I,T}). \quad (5)$$

This way, by replacing the unmasked pixels with the noisy version of the input image, we restrain the generation process from changing any pixel outside of the mask. After iterative denoising, the edited image, $\mathcal{D}(z_0)$, is obtained. Figure 2 presents an overview of our process.

4.3. Relevance field for scene editing

The idea of localizing the edits based on relevance maps can be extended to editing 3D scenes. Given a multiview capture, $\{I_i\}_{i=1}^n$, of a static scene and the corresponding camera poses, the goal is to edit a NeRF, f_θ, fitted to the scene according to a text prompt, C_T. Motivated by IN2N [15], we perform iterative training view updates by replacing one training view, I_t, at a time by its edited counterpart according to a text prompt, C_T. To ensure the consistency of the localization of edits across different views, we fit a 3D neural field, which we call the *relevance field*, to the relevance maps of all the training views. While editing each of the views, we render the corresponding relevance map from the relevance field to guide the edit.

To implement the relevance field, we extend f_θ to return a view-independent relevance, $r(x) \in [0, 1]$, for every point, x, in the 3D space. Notice that the geometry of the main NeRF and the relevance field is shared, and when fitting the relevance field, we always detach the gradients of the densities to ensure that the potential inconsistencies do not affect the geometry of the main scene. For a ray, r, the rendered relevance, $\widehat{R}(r)$, can be simply calculated by replacing the point-wise colours with relevance values in the volumetric rendering equation, as $\widehat{R}(r) = \sum_{i=1}^N w_i r_i$.

During the NeRF editing process, every n_{edit} iterations, we randomly sample a training view I_i. The first time we sample I_i, the relevance map, $\mathcal{R}_{I_i,I,T}$, is calculated and added to the training data of the relevance field. From the same view, the image, $\widehat{I}_i$, and the relevance map, $\widehat{R}_i$, are rendered using f_θ. The relevance-guided image editor from § 4.2 is used to locally edit $\widehat{I}_i$, conditioned on the original captured image, I_i, and the text condition, C_T. To this end, the encoded rendered image, $\mathcal{E}(\widehat{I}_i)$, is diffused until a random timestep, $t_{\text{edit}} \in T$, to obtain $z_{t_{\text{edit}}}$. The noisy latent, $z_{t_{\text{edit}}}$, is iteratively denoised conditioned on the original unedited view, I_i, and the text prompt, C_T, guided by the rendered relevance map, $\widehat{R}_i$, to obtain the edited training view, $\widetilde{I}_i = \mathcal{D}(z_0)$. Since the several-fold upsampling induced by the decoder could lead to inconsistencies in the unedited region, we replace the unedited RGB pixels in $\widetilde{I}_i$ with their counterparts from I_i using a relevance mask rendered in the original image resolution. After editing, $\widetilde{I}_i$ replaces the corresponding training view to supervise the main NeRF (the color field).

4.4. Implementation details

In all of our experiments, we set $t_{\text{rel}} = 0.8$, i.e., we apply 80% of the noise to predict the relevance map. For IP2P, we used the model available on HuggingFace, based on the diffusers package. For NeRF editing, we used the nerfacto model from NeRFStudio [68]. During the iterative dataset updates, we performed edits with noise levels (timesteps) randomly sampled from $[0.02, 0.98]$. We update a single training view every $n_{\text{edit}} = 10$ iterations. Each image is updated using 20 denoising steps for NeRF editing, and 100 denoising steps for image editing. For the relevance field implementation, we borrowed the same hyperparameters as the nerfacto field [68]; however, we never used the densities from this field, and only used the geometry from the main radiance field. The threshold, τ is set between $[0.4, 0.6]$ in all the experiments, unless stated otherwise. Each NeRF is first trained for 30,000 iterations on the original scene, and then edited for 3,000 or 4,000 iterations depending on the number of training views.

5. Experiments

Datasets. For image editing, we follow IP2P [6] and use their dataset of diverse images and editing instructions. The test set we used consists of 5,000 images, paired with instructions, and input and output captions. NeRF editing evaluation is done using scenes from IN2N [15] and LLFF [41]. We use 14 different NeRF editing tasks (i.e., text instructions) for the quantitative experiments. For each, a scene is edited using an instruction, and evaluated against a desired output caption. IN2N and LLFF provide multiview captures of forward-facing and 360° static scenes. Colmap [61, 62] is used to recover camera parameters.

Metrics. Following IP2P [6], for image editing, we use *CLIP image similarity* [52] and *CLIP text-image direction similarity* [13]. The former is the similarity between the CLIP embeddings of the edited and the original images. The latter measures the agreement between the change of the images (in CLIP space) and the change of the text captions. For scene editing, we use CLIP text-image similarity (Txt-Img Sim.) which is the cosine similarity of the CLIP embeddings of output views and the output caption. In addition, CLIP frame similarity (Frame Sim.) measures the cosine similarity of the consecutive frames in CLIP space. CLIP edit similarity (Edit Sim.) measures the directional agreement between the changes applied to neighbouring frames in CLIP space, as described in IN2N [15]. CLIP image similarity (Image Sim.) and edit PSNR measure the consistency of the edited views and the input views, in the CLIP and RGB spaces, respectively. Finally, we use NIQE [46], a no-reference image quality metric, to evaluate the quality and sharpness of the outputs.

Image editing baselines. For image editing, we compare against state-of-the-art methods, including DiffEdit [9], SDEdit [39], and IP2P [6]. Notice that DiffEdit expects input and output captions, and is evaluated with its desired inputs instead of the edit instruction. SDEdit expects the output caption; we evaluate it with the output caption as *SDEdit (out caption)* and with the edit instruction as *SDEdit (instruction)*, separately.

NeRF editing baselines. We quantitatively evaluate against IN2N [15] and per-frame IP2P, which independently edits rendered views of the input NeRF via IP2P [6]. We further compare our model against NeRF-Art [73], which uses CLIP similarity of the scene and a caption to edit scenes. Additional baselines include IN2N [15] with stable diffusion [55] (SD) rather than IP2P, and the Score Distillation Sampling [51] (SDS) loss with IP2P.

5.1. Results

Image editing. We provide quantitative evaluation of our relevance-guided image editing method in Figure 4, based on the IP2P [6] dataset. The figure shows the result of each model based on two competing metrics; similarity to the input image (y-axis), and the agreement with the edit (x-axis). Compared to the baselines, our model achieves higher image consistency with similar directional similarities. Additionally, notice that increasing the mask threshold, τ, increases the image similarity as a smaller region of the images is being edited. However, overly increasing τ has a detrimental effect on successfully achieving the edit.

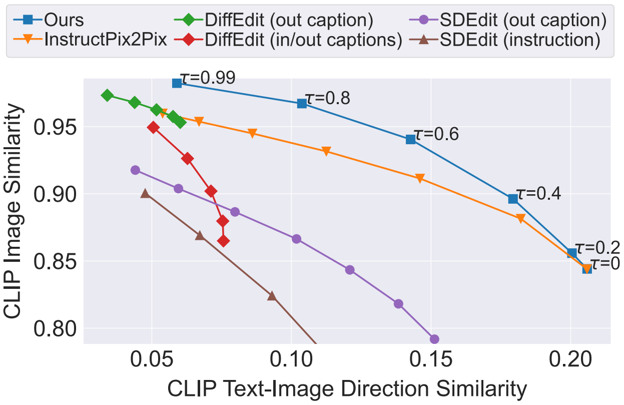

Figure 4. Quantitative image editing evaluation. Our model achieves better text-image direction similarity (x-axis), while maintaining higher fidelity to the input (y-axis). The text-guidance is set to 7.5 for every method. We pick SDEdit's strength from $[0.1, 0.9]$ and Diffedit's encoding-ratio from $[0.5, 0.9]$. For IP2P, S_I is changed between $[1, 2.2]$. For our method, s_I is set to 1.

DiffEdit [9] requires access to both input and output captions. Even with this information, since the captions in IP2P dataset are relatively complex rather than simple class names or high-level descriptions, DiffEdit fails to perform appropriate edits. In particular, when DiffEdit is only given the output caption and an empty text as the input caption, i.e., *DiffEdit (out caption)*, it never achieves higher text-image similarities, and the inputs remain relatively unchanged. This is due in part to the failure of DiffEdit in predicting the right masks; thus, the generative process is unable to apply proper changes to achieve successful edits. For SDEdit [39, 55], the fidelity of the outputs to the input images drop significantly as the strength of the edit is increased. This drop is due to the lack of an explicit mechanism to ensure consistency. In contrast to our model, SDEdit relies on the information kept in the noisy latent; however, in later diffusion stages, the noisy latent retains global information about the input, but lacks local details.

In many editing scenarios, the task can be fulfilled by only changing a local region of the image. Our method outperforms all the baselines by localizing the edits, and is able to produce on-par text-image similarities to IP2P, while keeping the outputs more consistent with the inputs. We provide qualitative comparisons in Figures 5 and 6. Our

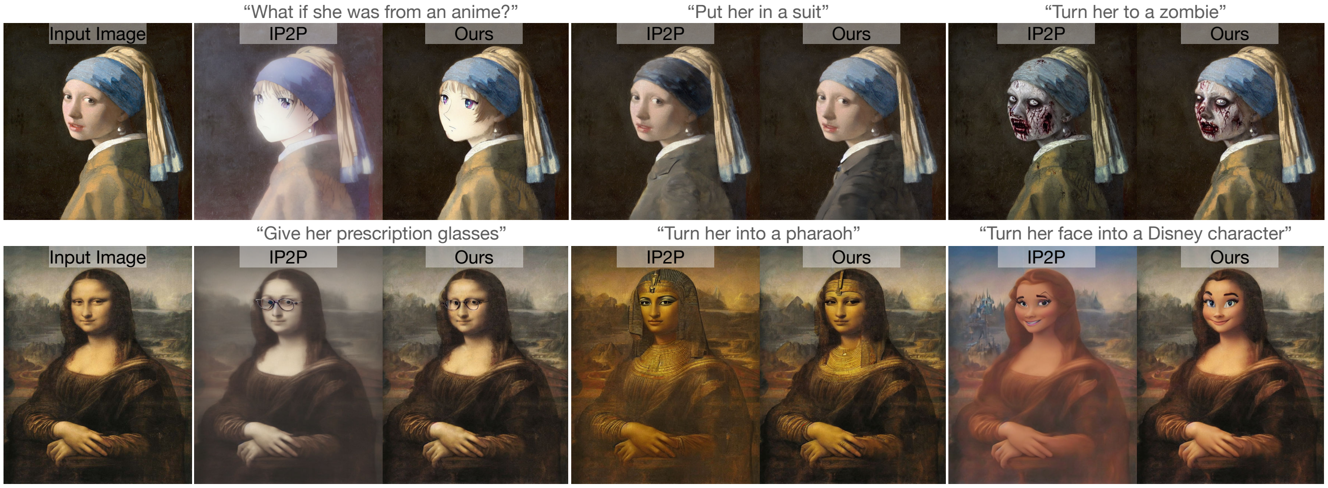
“What if she was from an anime?”
“Put her in a suit”
“Turn her to a zombie”
Input Image
IP2P
Ours
IP2P
Ours
IP2P
Ours
“Give her prescription glasses”
“Turn her into a pharaoh”
“Turn her face into a Disney character”
Input Image
IP2P
Ours
IP2P
Ours
IP2P
Ours

Figure 5. Our image editing results compared to IP2P. For both models, s_T and s_I are set to 7.5 and 1, respectively. IP2P fails to isolate the specified region, and over-edits the input. Our model explicitly predicts the scope of the edit, and limits the edit inside a specific region.

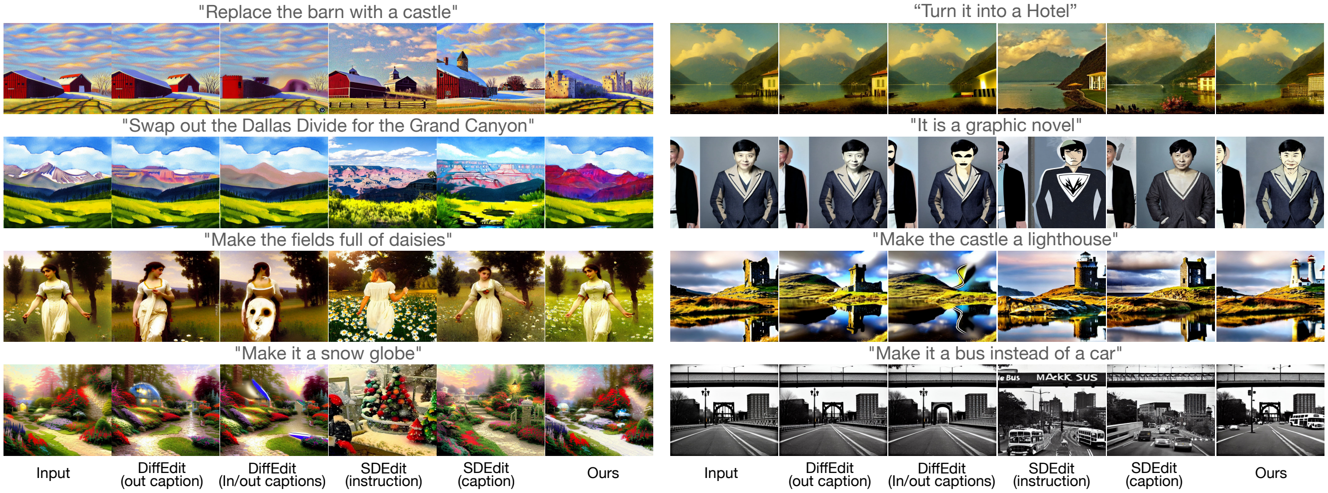
"Replace the barn with a castle"
“Turn it into a Hotel”
"Swap out the Dallas Divide for the Grand Canyon"
"It is a graphic novel"
"Make the fields full of daisies"
"Make the castle a lighthouse"
"Make it a snow globe"
"Make it a bus instead of a car"
Input
DiffEdit (out caption)
DiffEdit (In/out captions)
SDEdit (instruction)
SDEdit (caption)
Ours
Input
DiffEdit (out caption)
DiffEdit (In/out captions)
SDEdit (instruction)
SDEdit (caption)
Ours

Figure 6. Comparison of our image editing method against DiffEdit [9] and SDEdit [39]. DiffEdit requires the captions of both the input and output, but still fails to perform the edit as the captions in IP2P [6] dataset are relatively complex. SDEdit [39] performs better when it is given the output caption. Our model follows the instructions more closely, while maintaining coherence with the input.

outputs closely match the inputs; edits are only applied where necessary. See supplemental for more examples.

NeRF editing. We evaluate our method against the baselines on scenes from LLFF [41] and IN2N [15]. Table 1 contains quantitative results based on 14 scene editing tasks. Both our method and IN2N perform on-par with per-frame IP2P in terms of CLIP [52] text-image similarity, meaning the edited scenes successfully match the output captions. CLIP frame similarity and CLIP edit similarity show that IN2N and our method produce view-consistent results, whereas IP2P independently edits rendered views and is unable to maintain consistency. CLIP image similarity and Edit PSNR compare the cosine similarity and PSNR between each rendered view from the edited NeRF and from the input NeRF. They show that our method keeps the edited scene more consistent with the input scene. Finally, in terms of NIQE [46], a no-reference image quality metric, our method outperforms IN2N by producing sharper and higher quality results. Since per-frame IP2P's outputs are direct returns of a diffusion model rather than a NeRF, they lack typical NeRF artifacts and thus NIQE is higher for IP2P.

Qualitative examples of our scene editing results are shown in Figure 7. Our model edits the region most relevant to the edit, while keeping the rest of the scene unchanged. For example, while editing the bear statue and changing it to a *panda*, a *grizzly bear*, or a *polar bear*, the background

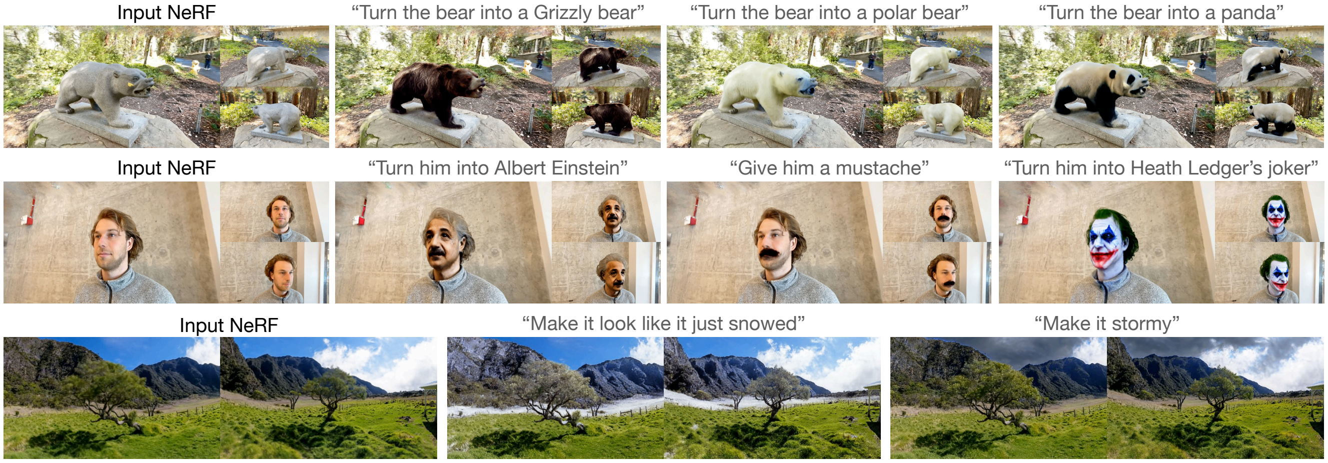
Input NeRF
“Turn the bear into a Grizzly bear”
“Turn the bear into a polar bear”
“Turn the bear into a panda”
Input NeRF
“Turn him into Albert Einstein”
“Give him a mustache”
“Turn him into Heath Ledger’s joker”
Input NeRF
“Make it look like it just snowed”
“Make it stormy”

Figure 7. Qualitative outputs of our NeRF editing method. For each of the 3 scenes and each text instruction, we provide multiview renderings of the edited NeRF to show view consistency. Our method follows the text, yet keeps the regions less relevant to the task intact.

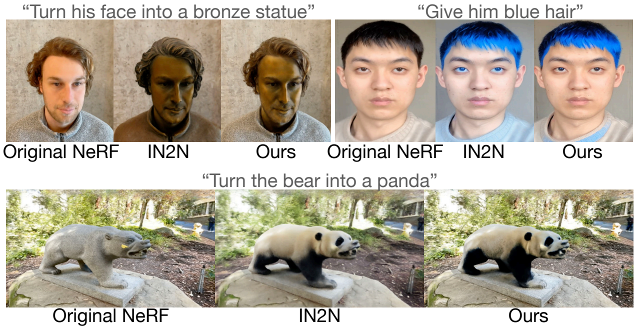

Figure 8. Comparison of our scene editing results against IN2N [15]. The relevance field enables us to localize the edit to the most significant regions. Editing a smaller region reduces the decoder spatial ambiguity problem on unedited pixels. Moreover, it improves the view consistency in the edited region as editing a small part is more likely to produce consistent results across the views. Finally, having a small NeRF reconstruction loss outside of the mask improves the convergence in the masked region.

Table 1. Evaluation of our model on scene editing. Our model achieves similar performance to IN2N [15] and per-frame IP2P [6] in terms of CLIP text-image similarity of the edited frames and output captions. As IP2P is independently editing views, it is inferior to the other methods in terms of consistency between the neighbouring frames and the edits applied to them. In terms of image quality (NIQE), our method outperforms the other 3D baseline, IN2N, by producing sharper and higher quality renderings.

Method	**Txt-Img Sim.↑**	**Frame Sim.↑**	**Edit Sim.↑**	**Image Sim.↑**	**Edit PSNR↑**	**NIQE↓**
IP2P [6]	0.2770	0.9669	0.8082	0.9111	19.44	4.02
IN2N [15]	0.2683	0.9865	0.8822	0.8649	28.70	6.43
Ours	0.2673	0.9876	0.8754	0.8910	31.01	5.53

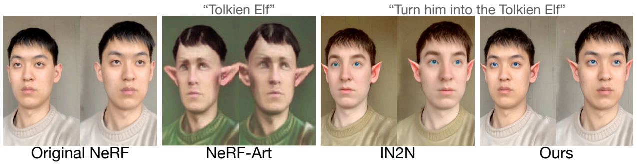

Figure 9. Comparison of our scene editing method against NeRF-Art [73] and IN2N [15]. The baselines modify the background, shirt, and hair of the person, while our model only edits the eyes and ears. The extraneous changes of the baselines can even fail to preserve important scene semantics (in this case, the individual’s identity). In contrast, our method applies only the minimum change required for the desired semantic alteration.

and the stage underneath the statue have remained intact, while the statue itself is changed to desired animals with sharp textures (notice the texture of the fur).

We further provide qualitative comparisons to the baselines. As shown in Figure 8, built directly on IP2P, IN2N has the same tendency to over-edit scenes. In the case of giving *the guy blue hair*, notice how it has also changed the t-shirt, eyes, and background colours. It has also changed the whole torso of the other person to a bronze statue, where the prompt has only asked for changing the face. In the bear scene, the background in IN2N output is blurred. This is due in part to the ambiguity of VAE decoder in upsampling, resulting in minor misalignments between different views. Moreover, since IP2P does not attempt to prevent changes to the background, some of the edited views have an altered background. This inconsistency has resulted in a loss of sharpness. It also disrupts the optimization, as network capacities and loss gradients are allocated to background inconsistencies; hence, IN2N outputs are not as sharp as our result. Moreover, IP2P is constrained to only edit the bear

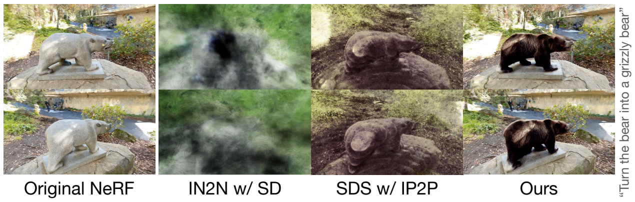

Figure 10. Qualitative comparison of our scene editing method against two baselines. *IN2N w/ SD* performs the same iterative dataset updates as IN2N [15], but with stable diffusion [55] instead of IP2P. *SDS w/ IP2P* performs updates on the NeRF based on the SDS loss [51] calculated via IP2P. Our method results in sharp outputs, while the baselines have failed on the task.

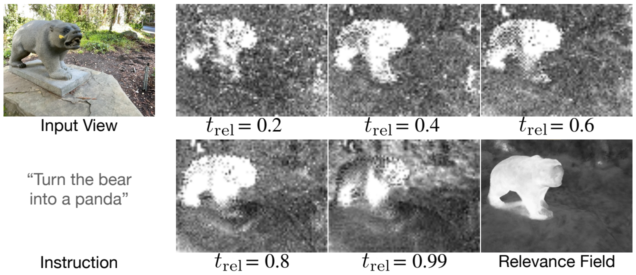

Figure 11. Comparison of the relevance maps calculated with different noise levels and a rendered relevance from a relevance field.

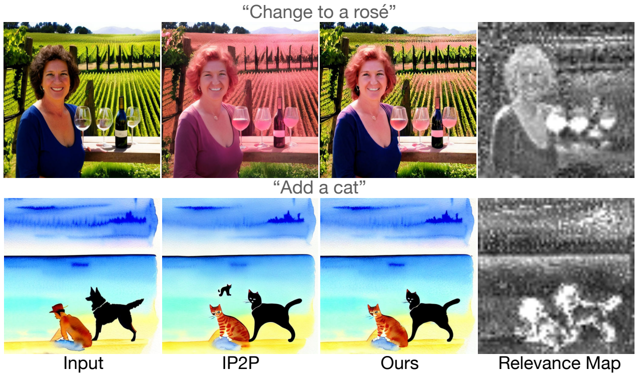

Figure 12. Examples of the failure cases of our image editing model. Due to the reliance of our model on IP2P for predicting the relevance map and the edit, although our model is able to outperform IP2P by localizing the edits, it naturally cannot recover from IP2P's catastrophic failures.

to a panda in our case, rather than trying to edit the whole image to satisfy the instruction. Consequently, the edited views in our method are more likely to be consistent, especially for nearby views, which is another reason that even our edited regions are considerably sharper, e.g., the texture of the panda's fur. In Figure 9, NeRF-Art [73] has followed the instruction and changed the face to the *Tolkien Elf*, but the edited scene has quality artifacts associated with CLIP-based [52] methods, and has changed irrelevant regions of the scene, including the hair, background, and t-shirt. Figure 10 compares our method with additional baselines.

Relevance noise level. The relevance noise level, t_{rel}, is a hyperparameter we use for the calculation of the relevance field. Figure 11 shows a comparison between the maps calculated using different noise levels. In our experiments, we found $t_{\text{rel}} = 0.8$ to be reliable. This way, the relevance map is calculated using predictions in the higher-noise stages. As a result, the denoising process is fixating on the global structure of the generated images, rather than the fine details [2]. Thus, the predicted relevance masks encapsulate the global boundaries of the relevant regions. Moreover, Figure 11 shows the rendering of the relevance field from the same view. Since the relevance field is supervised using maps from multiple views, it is effectively an ensemble over multiple predictions, and is more accurate than each single map. In addition to this ensemble nature, the inductive bias of the NeRF architecture limits high-frequency field variations; hence, relevance renders provide a smooth consensus over the global scene structure, with minimal noise.

Failure cases. The backbone of our method is a pre-trained IP2P [6]. As a result, although our mask-guidance is able to alleviate the over-editing problem of IP2P, and to fix the upsampling ambiguity issue, it still is unable to recover from the cases in which IP2P fails badly. In Figure 12, we provide examples of such failures. For instance, in the first row, the prompt is *"change to a rosé"*. Given the context of the image, the goal is to only change the drinks. However, IP2P has completely changed the field in the background and the hair colour to pink. This failure is reflected in the predicted relevance mask, which superfluously highlights those areas. Although the result of our model is arguably better, it has still edited parts that were unnecessary to change. In the second example, *"add a cat"*, localizing the edit with respect to the prompt is an ambiguous problem. The relevance map has failed to localize a certain position for the cat to be added, and instead, the person and the dog have been replaced with cats. Our method is agnostic to the underlying instruction-conditioned diffusion model, and can benefit from swapping IP2P with a better one in the future.

6. Conclusion

We propose a method for predicting the relevance of each image pixel to an editing task based on a text instruction. This is done by looking at the discrepancy between a conditional and an unconditional pass over a diffusion-based image editor. We use this relevance as a mask to guide the generation and force the unmasked pixels to not change, resulting in a localized image editor. We further show that training a relevance field on the relevance maps of the training views of a NeRF achieves similar localizations when editing 3D scenes. Our method shows superior performance compared to the baselines in both image and scene editing tasks.

Acknowledgments. This work was conducted at Samsung AI Centre Toronto and it was funded by Mitacs and Samsung Research, Samsung Electronics Co., Ltd.

References

[1] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *CVPR*, 2022. 1, 2

[2] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Qinsheng Zhang, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediff-i: Text-to-image diffusion models with ensemble of expert denoisers. *arXiv*, 2022. 1, 8

[3] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. *ICCV*, 2021. 2

[4] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 2

[5] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *ICCV*, 2023. 2

[6] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 1, 2, 3, 5, 6, 7, 8, 12, 13

[7] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, 2022. 2

[8] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. MobileNeRF: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In *CVPR*, 2023. 1, 2

[9] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. *ICLR*, 2023. 1, 2, 5, 6

[10] Angela Dai, Yawar Siddiqui, Justus Thies, Julien Valentin, and Matthias Niessner. SPSG: Self-supervised photometric scene generation from RGB-D scans. In *CVPR*, 2021. 2

[11] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *CVPR*, June 2022. 2

[12] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021. 1, 2

[13] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *TOG*, 2022. 5

[14] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. Convolutional sequence to sequence learning. In *ICML*, 2017. 2

[15] Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. *ICCV*, 2023. 1, 2, 3, 4, 5, 6, 7, 8

[16] Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *ICCV*, 2021. 2

[17] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *ICLR*, 2023. 1, 2

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 1, 2

[19] Jonathan Ho, Chitwan Saharia, William Chan, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *JMLR*, 2022. 1, 2

[20] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. *CVPR*, 2022. 2

[21] Ru-Fen Jheng, Tsung-Han Wu, Jia-Fong Yeh, and Winston H Hsu. Free-form 3D scene inpainting with dual-stream GAN. *BMVC*, 2022. 2

[22] Kacper Kania, Kwang Moo Yi, Marek Kowalski, Tomasz Trzciński, and Andrea Tagliasacchi. CoNeRF: Controllable neural radiance fields. In *CVPR*, 2022. 2

[23] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models, 2023. 1, 2

[24] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *TOG*, 2023. 2

[25] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv*, 2022. 3

[26] Zhengfei Kuang, Fujun Luan, Sai Bi, Zhixin Shu, Gordon Wetzstein, and Kalyan Sunkavalli. PaletteNeRF: Palette-based appearance editing of neural radiance fields. In *arXiv*, 2022. 2

[27] Andreas Kurz, Thomas Neff, Zhaoyang Lv, Michael Zollhöfer, and Markus Steinberger. AdaNeRF: Adaptive sampling for real-time rendering of neural radiance fields. In *ECCV*, 2022. 2

[28] Verica Lazova, Vladimir Guzov, Kyle Olszewski, Sergey Tulyakov, and Gerard Pons-Moll. Control-NeRF: Editable feature volumes for scene rendering and manipulation. In *WACV*, 2023. 2

[29] Zuoyue Li, Tianxing Fan, Zhenqiang Li, Zhaopeng Cui, Yoichi Sato, Marc Pollefeys, and Martin R Oswald. Comp-NVS: Novel view synthesis with scene completion. In *ECCV*, 2022. 2

[30] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *CVPR*, 2023. 2

[31] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. BARF: Bundle-adjusting neural radiance fields. In *ICCV*, 2021. 2

[32] David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. BACON: Band-limited coordinate networks for multiscale scene representation. In *CVPR*, 2022. 2

[33] Hao-Kang Liu, I-Chao Shen, and Bing-Yu Chen. NeRF-In: Free-form NeRF inpainting with RGB-D priors. In *arXiv*, 2022. 2

[34] Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B. Tenenbaum. Compositional visual generation with composable diffusion models. *ECCV*, 2023. 3

[35] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object, 2023. 2

[36] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. In *ICCV*, 2021. 2

[37] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022. 1, 2

[38] Nelson Max and Min Chen. Local and global illumination in the volume rendering integral. Technical report, Lawrence Livermore National Lab (LLNL), Livermore, CA (United States), 2005. 3

[39] 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, 2021. 1, 2, 5, 6

[40] Aryan Mikaeili, Or Perel, Mehdi Safaee, Daniel Cohen-Or, and Ali Mahdavi-Amiri. Sked: Sketch-guided text-based 3d editing, 2023. 2

[41] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ToG*, 2019. 5, 6

[42] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2

[43] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Marcus A. Brubaker, Jonathan Kelly, Alex Levinshtein, Konstantinos G. Derpanis, and Igor Gilitschenski. Reference-guided controllable inpainting of neural radiance fields. In *ICCV*, 2023. 1, 2

[44] Ashkan Mirzaei, Tristan Aumentado-Armstrong, Konstantinos G. Derpanis, Jonathan Kelly, Marcus A. Brubaker, Igor Gilitschenski, and Alex Levinshtein. SPIn-NeRF: Multiview segmentation and perceptual inpainting with neural radiance fields. In *CVPR*, 2023. 1, 2

[45] Ashkan Mirzaei, Yash Kant, Jonathan Kelly, and Igor Gilitschenski. LaTeRF: Label and text driven object radiance fields. In *ECCV*, 2022. 2

[46] Anish Mittal, Rajiv Soundararajan, and Alan C. Bovik. Making a “completely blind” image quality analyzer. *IEEE Signal Processing Letters*, 2013. 5, 6

[47] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. *CVPR*, 2023. 1

[48] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *TOG*, 2022. 1, 2

[49] 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 *ICML*, 2022. 1, 2

[50] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. In *SIGGRAPH*, 2023. 2

[51] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. In *ICLR*, 2023. 2, 5, 8

[52] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *ICML*, 2021. 5, 6, 8

[53] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *arXiv preprint arXiv:2204.06125*, 2022. 1, 2

[54] Christian Reiser, Richard Szeliski, Dor Verbin, Pratul P. Srinivasan, Ben Mildenhall, Andreas Geiger, Jonathan T. Barron, and Peter Hedman. MERF: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. In *arXiv*, 2023. 1, 2

[55] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 2, 5, 8

[56] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *CVPR*, 2022. 3

[57] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *SIGGRAPH*, 2022. 1, 2

[58] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022. 1, 2

[59] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *TPAMI*, 2022. 1, 2

[60] Sara Fridovich-Keil and Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022. 2

[61] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 5

[62] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 5

[63] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised

learning using nonequilibrium thermodynamics. *PMLR*, 2015. 1, 2

[64] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ICLR*, 2021. 2, 4

[65] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, 2017. 2

[66] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *NeurIPS*, 2020. 1, 2

[67] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020. 2

[68] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *SIGGRAPH*, 2023. 4

[69] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Niessner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhoefer, and Vladislav Golyanik. Advances in neural rendering. In *SIGGRAPH*, 2021. 2

[70] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2

[71] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022. 2

[72] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. CLIP-NeRF: Text-and-image driven manipulation of neural radiance fields. *CVPR*, 2022. 2

[73] Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Nerf-art: Text-driven neural radiance fields stylization. *TVCG*, 2023. 5, 7, 8

[74] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. IBRNet: Learning multi-view image-based rendering. In *CVPR*, 2021. 2

[75] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF--: Neural radiance fields without known camera parameters. In *arXiv*, 2021. 2

[76] Silvan Weder, Guillermo Garcia-Hernando, Aron Monszpart, Marc Pollefeys, Gabriel Brostow, Michael Firman, and Sara Vicente. Removing objects from neural radiance fields. In *CVPR*, 2023. 2

[77] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *ICCV*, 2021. 2

[78] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *ICCV*, 2021. 2

[79] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 2

[80] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. NeRF-editing: geometry editing of neural radiance fields. In *CVPR*, 2022. 2

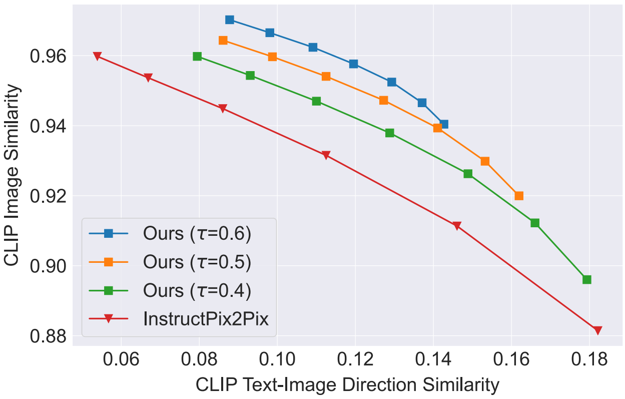

Figure 13. Quantitative comparison of our image editing method with different mask threshold values, τ, against IP2P [6]. The text-guidance is set to 7.5 for both method. For IP2P and our method, S_I is changed between $[1.2, 2.2]$ and $[1.0, 2.2]$, respectively.

A. Mask threshold effect

Figure 13 provides a quantitative comparison of our image editing method with different mask thresholds against IP2P [6]. Our method produces outputs that closely reflect the desired edits (x-axis) while remaining consistent with the inputs (y-axis) by confining the edits in the relevant region. However, while increasing the mask threshold results in higher fidelity to the input, overly increasing it can prevent the model to edit the parts that actually matter; the lines in Figure 13 cover a smaller text-image direction similarity region as the mask threshold, τ, is increased. Based on this experiment, we've typically set τ between $[0.4, 0.5]$ throughout the paper.

We further provide a qualitative example to showcase the effect of the mask threshold on the generation of the edited image (Figure 14). Setting the mask threshold, τ, to 0 results in every pixel to be masked. As a result, our model with $\tau = 0$ is equivalent to IP2P [6]. For each τ, we provide results with different image guidance scales, S_I. As evident in the results, in IP2P, simply increasing the image guidance scale is not enough to localize the edits; with $S_I = 1.0$, the background and the clothes are drastically changed. When setting $S_I = 3.0$ in IP2P, the woman's collar and the man's shirt are still changed to yellow, while the faces no more look like the Simpsons characters; increasing S_I has an adverse effect on the text-image similarity, which is consistent with our quantitative findings in Figures 4 and 13. On the other hand, changing τ provides a different guidance knob to the user, and allows them to control the region to be edited, with a minimal damage to the regions that actually need to be modified.

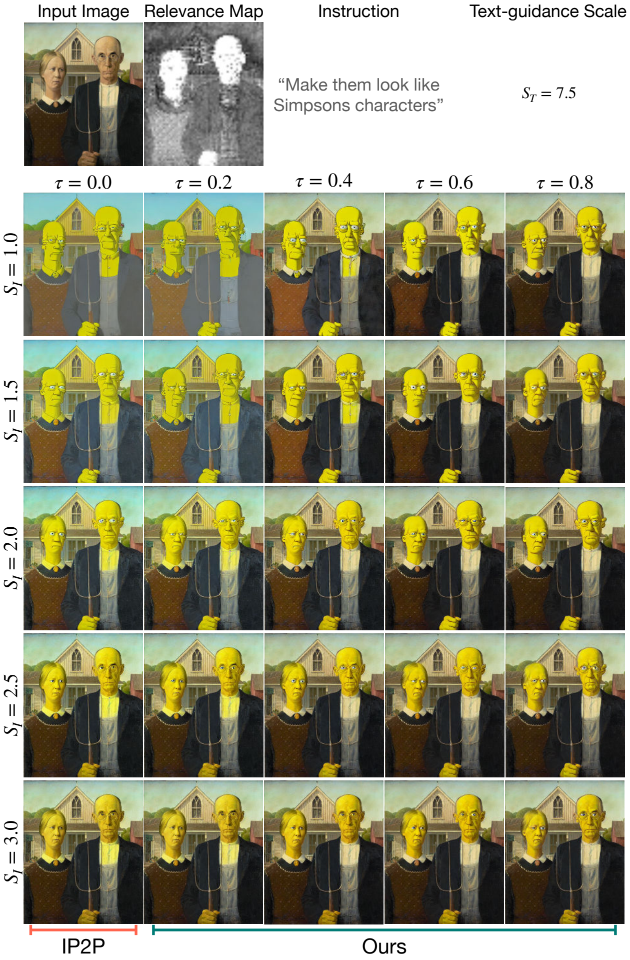

Figure 14. Qualitative comparison of the effects of the mask threshold, τ, and the image guidance scale, S_I. Increasing the image guidance scale improves the similarity of the output and the input image, but significantly decreases the intensity of the edit, resulting in a failure. In contrast, τ provides a knob to the user to control the region of the edit, with reduced effects on the quality of the edit.

B. Additional qualitative comparisons

In Figure 15, we provide additional example to compare our localized image editing method with IP2P. In all of these examples, S_I is between $[0.8, 1.0]$, and S_T is always 7.5. Mask thresholds are either 0.4 or 0.5. Foe each of the examples, we further provide the relevance map predicted by our method. Our results are more consistent with the input image by only locally changing the inputs in the regions with high relevance values. Meanwhile, our method has followed the instructions closely, and has resulted in images with similar or better edit qualities compared to IP2P.

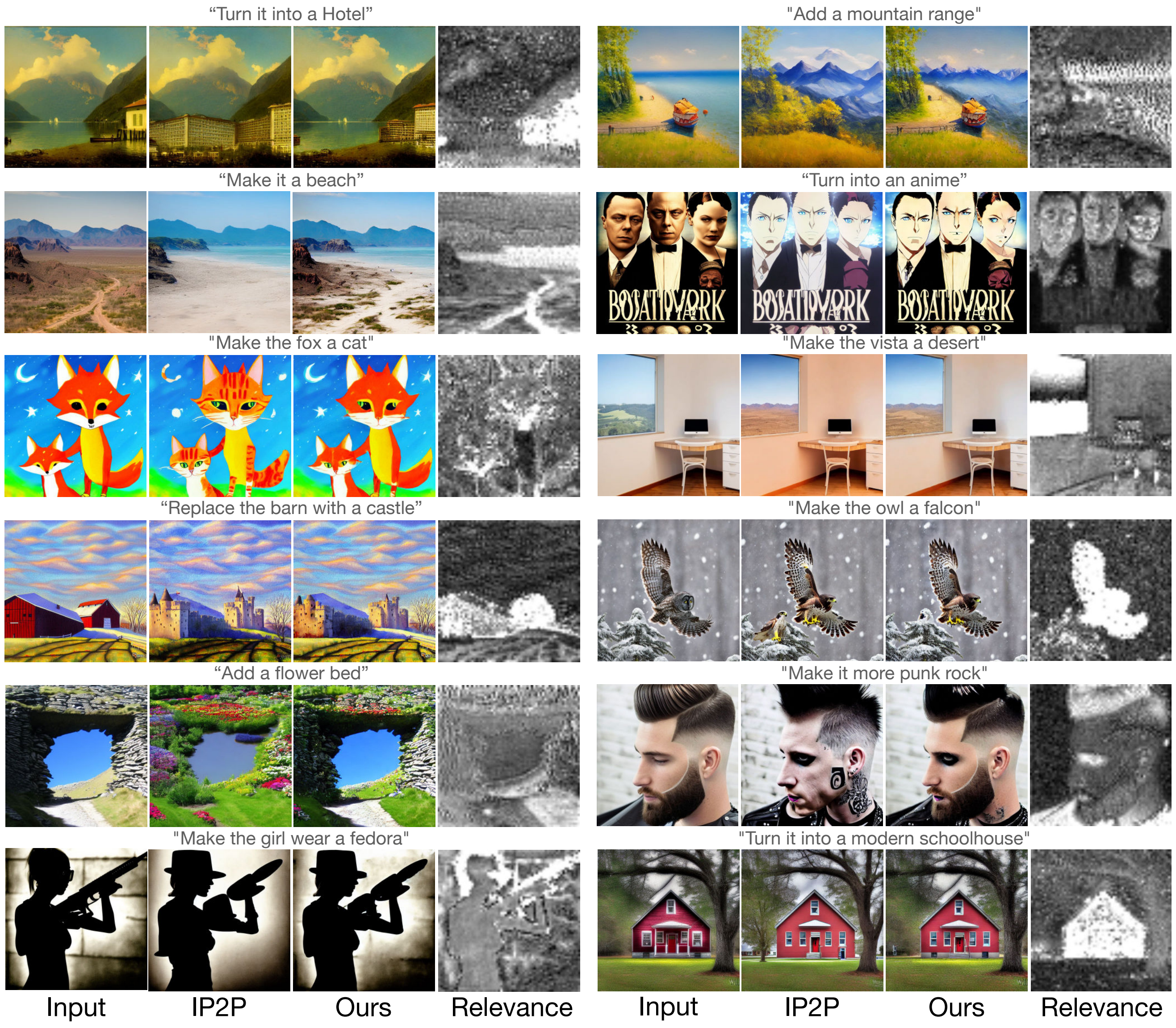

Figure 15. Additional Qualitative comparison of our image editing results against IP2P [6]. For each image, we show the instruction, outputs of IP2P and our model, and the relevance map predicted by our model. Our model follows the instruction, while maintaining more consistency with the input image. This is due to the relevance-guidance, as only pixels with high relevance values are allowed to be modified.

C. Sample relevance field renderings

In Figure 16, we visualize sample relevance fields trained on different scenes and different edit instructions. Each relevance field is trained via 2D supervisions from relevance maps of the training views. As shown in the results, the relevance fields are mainly activated around the region that should be edited. To edit a NeRF, we use the rendered views from the relevance field as relevance-guidance for editing training views. As a result, the updates of training views during the iterative update process are only locally changed, and the changed region is consistent across different views.

Note that the relevance field's densities (geometry) is always queried from the main NeRF model that is being edited. This is to ensure that potential inconsistencies between 2D relevance maps of different views does not hurt the main NeRF, and to enforce 2D relevance maps to be projected to the actual geometry of the scene and to become 3D-consistent; otherwise, the relevance field's geometry might converge to a degenerate solution to justify the inconsistencies. As the main NeRF is being updated towards the desired edited NeRF, its geometry might change. In that case, since the relevance field shares the same geometry, its 2D maps will be projected to the updated NeRF. Thus, during each update, the relevance field localizes the edits on the current version of the main NeRF. This allows slow changes in the geometry of the scene while only locally updating the views at each step, e.g., the addition of the mustache or the sunglasses, which require changes to the densities.

D. Additional Details

For Figure 5, we set S_T and S_I to 7.5 and 1 respectively, while selecting τs proper to each edit. In Figure 6, for our method, we set $S_T = 7.5$, $S_I = 0.8$, and $\tau = 0.4$. For the other methods we used their default set of hyperparameters. In Figures 12 and 15, we set $S_T = 7.5$, $S_I = 0.8$ for both methods and $\tau = 0.4$ for our method. For NeRF editing experiments, τ is always set to 0.5, and the guidance scales are as follows:

- **Bear**: $S_T = 6.5, S_I = 1.5$
- **Face**: $S_T = 7.5, S_I = 1.5$
- **Farm**: $S_T = 12.5, S_I = 1.5$
- **Fangzhou**: $S_T = 6.5, S_I = 1.5$

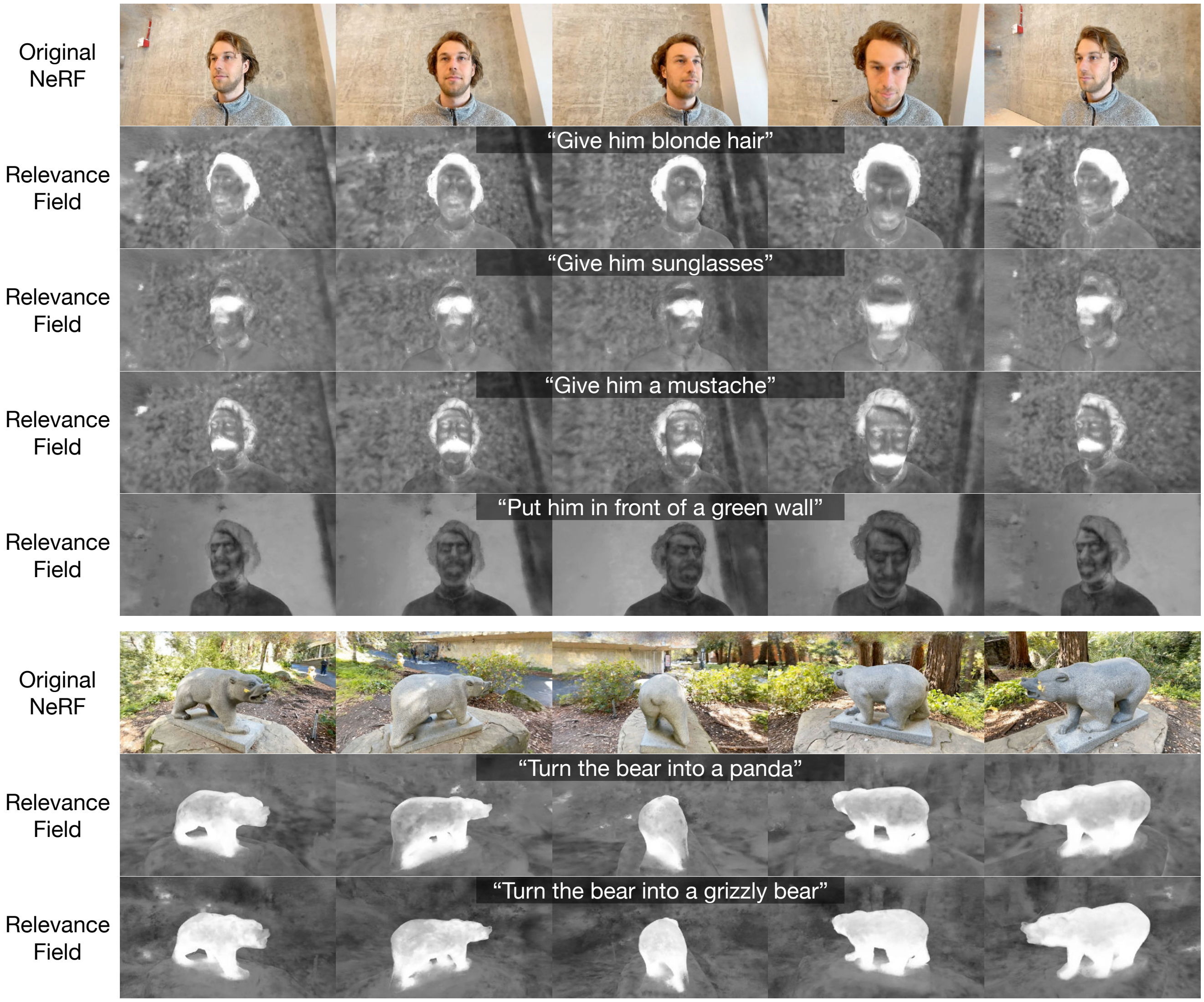

Figure 16. Sample rendered relevances from relevance fields trained on different scenes and different edit instructions. Each relevance field is visualized from multiple views, in addition to the corresponding views from the original NeRF model of the scene. Notice how each relevance field is mostly activated around the region that is highly relevant to the edit. For example, in the face scene and with the instruction *"Give him blonde hair"*, only the hair is given high values in the field. This field allows to localize edits of the training views during each iterative update in a 3D consistent manner.