Supplementary Material for Instruct-NeRF2NeRF: Editing 3D Scenes with Instructions

Ayaan Haque, Matthew Tancik, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa

UC Berkeley

1. Additional implementation details

The primary input to our method is a NeRF reconstruction of a real scene. This reconstruction is obtained using the 'nerfacto' model from NeRFStudio [3], trained for $30,000$ iterations per scene. Before running Instruct-NeRF2NeRF, we re-initialize the optimizers in NeRFStudio. In order to specify edits, we use InstructPix2Pix [1], where users specify the classifier-free guidance weights in order to explore the desired amount of change for a given edit. In our method, we inherit this parameter. Below we list these chosen values for the sequences shown in the paper:

1. *Fig. 4 Tree*: s_I=1.5, $s_T \in [7.5, 12.5]$
2. *Fig. 8 Tents*: s_I=1.5, s_T=10.0
3. *Fig. 4 Person*: $s_I \in [1.5, 1.75]$, $s_T \in [6.5, 7.5]$
4. *Fig. 1*: s_I=1.5, $s_T \in [6.5, 7.5]$
5. *Fig. 4 Bear*: $s_I = 1.5$, $s_T = 6.5$
6. *Fig. 7*: $s_I \in [1.3, 1.5]$, $s_T \in [6.5, 8.5]$

Our process of optimizing for an edited NeRF uses a diffusion model as guidance, which can produce a collection of temporally varying images (i.e., varying over the course of optimization). As a result, the optimization process does not have a single convergence point, as standard NeRF optimization does, where all images are sufficiently well explained by the reconstructed model. Therefore, the edited NeRF also varies in type and strength of edit over the course of optimization, and one must select an iteration at which to terminate optimization and visualize the edited scene. In practice, the optimal choice for training length is a subjective decision — a user may prefer more subtle or more extreme edits that are best found at different stages of training. The results in the paper are shown after a varying number of iterations, $3000 - 4000$ for smaller scenes (e.g., Fig. 1 and Fig. 4 bear, which both have under 100 images each), and $7000 - 8000$ for larger scenes (the remaining scenes, with over 200-300 images each).

For InstructPix2Pix, we use the public implementation included in the Diffusers library [4].

2. Limitations

As mentioned in the main paper, our method inherits many of the limitations of InstructPix2Pix. This includes (1) the inability to perform large spatial manipulations, (2) the occasional inability to perform binding between objects referred to in the instruction and the corresponding scene objects, and (3) adding or removing large objects. Furthermore, as in DreamFusion, our method uses a diffusion model on a single view at a time, and thus may suffer from similar artifacts, such as double faces on added objects. Finally, it's worth noting that the edit instructions provided to InstructPix2Pix are sometimes more relevant to certain views than others. For example, if the instruction is to *"turn the man into a bear"*, not all views may prominently feature the man, and therefore, certain views may consistently produce less of an edit or no edit at all. Our framework can easily incorporate improvements made in the InstructPix2Pix and its follow up works.

We additionally note that the edited NeRF scenes often contain slightly blurrier textures when compared to the originally reconstructed NeRF. Some initial experimentation suggests that this may be a result of the Stable Diffusion autoencoder, which often does not produce an entirely faithful copy of the original image, even in unedited regions. The autoencoder, while producing visually comparable results, often creates images with locally similar but non-identical textures that are not globally 3D-consistent. To validate this hypothesis, we show an experiment in Figure 1, where we simply autoencode the input images using Stable Diffusion (i.e., the same autoencoder used by InstructPix2Pix) and continue training the NeRF. As a result, the scene becomes gradually blurrier.

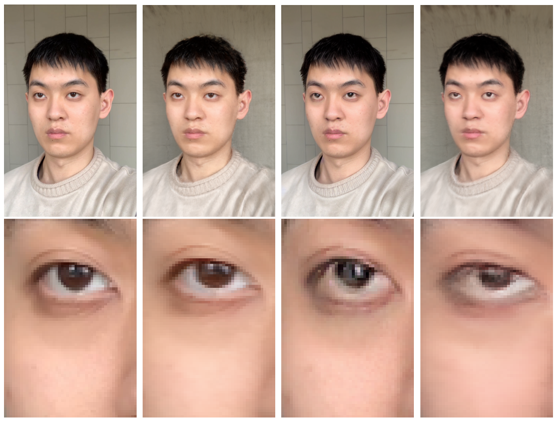

Figure 1: **Effects of autoencoder**: From left to right: (1) the input captured image, (2) the original reconstructed NeRF, rendered from the same viewpoint, (3), the original image 1, encoded and decoded by Stable Diffusion's autoencoder, (4) a NeRF reconstruction of the autoencoded images, rendered from the same viewpoint. Below, we show a zoomed-in patch of the eye. We note that while the input sequence can be reasonably reconstructed without much loss of detail, and the autoencoded images are similarly sharp as the input captured sequence, there are slight perturbations in local textures that end up not being 3D-consistent, resulting in blurry renders in the final recontruction.

Furthermore, over much longer optimization runs (i.e., when the process of rendering, editing, and propagating the edited images back into the NeRF has been repeated many times), we note a decrease in visual quality. We also largely attribute this to the effects of the autoencoder.

3. Metrics

In the quantitative evaluation, we report two metrics, a *CLIP Directional Score* [2], and a metric we name *CLIP Direction Consistency Score*. The CLIP Directional score measures how much the change in text captions agrees with the change in the images, and the CLIP Consistency score measures the cosine similarity of the CLIP embeddings of each pair of adjacent frames in a render novel camera path.

More formally, for the directional score, we encode a pair of images (the original and edited NeRFs, rendered at a given viewpoint), as well as a pair of text prompts that describe the original and edited scenes, e.g., *"a photograph of a man"* and *"a photograph of a Tolkien Elf"*. Using these, we compute the directional score described in InstructPix2Pix [1] and StyleGAN-NADA [2].

For the temporal consistency loss, we encode a pair of consecutive rendered frames from a novel trajectory, using both the original NeRF and the edited NeRF. In all, we are left with four CLIP embeddings, $C(o_i)$, $C(o_{i+1})$, $C(e_i)$, $C(e_{i+1})$, corresponding to original NeRF renderings o_i, o_{i+1} and edited NeRF renderings e_i, e_{i+1} for consecutive novel views i and $i+1$. We define the consistency loss as:

$$(C(e_i) - C(o_i)) \cdot (C(e_{i+1}) - C(e_i)) \tag{1}$$

This effectively amounts to measing the change in the CLIP-space edit direction from frame to frame.

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023.

[2] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *arXiv preprint arXiv:2108.00946*, 2021.

[3] 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. *arXiv preprint arXiv:2302.04264*, 2023.

[4] Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, and Thomas Wolf. Diffusers: State-of-the-art diffusion models. `https://github.com/huggingface/diffusers`, 2022.