Supplementary Material for GenN2N: Generative NeRF2NeRF Translation

Xiangyue Liu[1] Han Xue[2] Kunming Luo[1] Ping Tan[1†] Li Yi[2,3,4†]

[1] Hong Kong University of Science and Technology [2] Tsinghua University
[3] Shanghai Artificial Intelligence Laboratory [4] Shanghai Qi Zhi Institute

1. Overview

To make our GenN2N self-contained, we provide more details in this document, including:

- More details about our method, including 2D image-to-image translator used in our pipeline and the architecture details of our translated NeRF.
- Detailed settings of our experiments, including datasets settings and implementation details.
- More insight experiments of our method, including quality verification of our generation space, comparisons with naive altering parameters in InstructNeRF2NeRF, interpolation of the edit code, and ablation study on hyperparameter M.
- Additional experiment results, including more qualitative and quantitative results and additional applications of our GenN2N.

2. Method Details

2.1. 2D image-to-image Translator

In our proposed GenN2N, we use plug-and-play image-to-image translators to perform editing on the 2D domain and optimize the translated NeRF to lift these 2D edits into the 3D NeRF space. Note that the 2D translator in our pipeline can be changed to support various of NeRF editing tasks, here for convenience of comparing our method with existing task-specific NeRF editing baselines, we different 2D translators to achieve corresponding editing tasks as follows:

- **Text-driven Editing.** To achieve NeRF editing under text instructions, we use InstructPix2Pix [2] as the 2D image-to-image translator in our framework. InstructPix2Pix is a diffusion-based method designed for image editing according to user-provided instructions. Specifically, InstructPix2Pix learns a U-Net to perform denoise diffusion to generate the target edited image based on the given image and the text embedding. While InstructPix2Pix can produce high-quality editing results that highly align with the input instructions, given different initial noise or input image, different content may be generated during the editing process of InstructPix2Pix, which makes it difficult to ensure the 3D consistency in the text-driven NeRF Editing process.
- **Super-resolution.** For the NeRF super-resolution task, we choose ResShift [20] instead of InstructPix2Pix [2] as the 2D image-to-image translator in GenN2N due to the unrobustness of InstructPix2Pix [2] in the super-resoluion task. ResShift [20] is the current state-of-the-art image super-resolution method designed based on diffusion model. With dedicated designs for image super-resolution, such as the residual shifting mechanism and the flexible noise schedule, ResShift [20] can produce super-resolution images with high-quality. Thus, given a set of multi-view images of a NeRF scene, we directly use ResShift [20] to increase the resolution of all these images by the same factor of $\times 4$ as NeRF-SR [16].
- **Inpainting.** For the task of inpainting in NeRF, we aim to replace a certain region in a 3D scene, usually an object, and keep painted contents visually plausible and consistent with the remained context. Following SPIn-NeRF [12], we use LaMa [14] as our 2D image-to-image translator. The input of LaMa is an image and a binary mask that indicates the region to paint. We support various ways to get a mask, but note that multi-view masks must correspond to the same location in the 3D scene. For example, by artificially calculating the position of the part to paint in the 3D scene corresponding to the 2D image, or using the segment anything model [7] to get the mask of the same object. Based on these masked images, LaMa can successfully generate contents in the desired region that remain close to the input image with plausible 3D appearance and geometry.
- **Colorization.** To achieve 3D NeRF colorization, we use DDColor [6] as our plug-and-play 2D image-to-image translaton. Specifically, given a set of gray-scale multi-view images of a NeRF scene, we use DDColor [6] to produce RGB color of each image. While high-quality colorization results can be obtained for each image us-

†Corresponding authors.

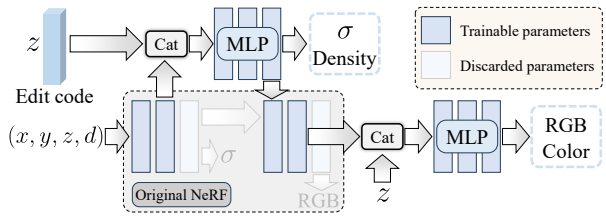

Figure 1. **Detailed structure of the translated NeRF.** Given a pre-trained NeRF model, we concatenate our edit code $\mathbf{z}$ with its intermediate features and produce the density σ and RGB color with two additional MLP networks. In this way, our translated NeRF is optimized to render the translated 3D scene conditioned on the edit code $\mathbf{z}$.

ing DDColor [6], different image may be assigned with different color style, which makes it difficult to generate consistent 3D colorization results. However, GenN2N successfully models the diverse results of colorization in the generated space, and each translated NeRF achieves a high degree of 3D consistency.

2.2. Network Architecture

After 2D image editing, we then achieve NeRF editing in 3D domain by performing optimization of our translated NeRF using our well-designed loss functions. During the optimization process, we use our latent distil module to extract a latent code named edit code $\mathbf{z}$ from each edited 2D image. Specifically, we employ the off-the-shelf VAE encoder from stable diffusion [13] to extract the feature from the edited 2D image and then apply a tiny MLP network to produce this edit code $\mathbf{z} \in \mathbb{R}^{64}$. Just like a conventional variational encoder, this tiny MLP network is used to estimate the mean and variance of a Gaussian distribution and sample the edit code from it. This tiny MLP network only contains three layers. After extraction of the edit code $\mathbf{z}$, we then render novel views conditioned on the edit code using our translated NeRF. In Fig 1, we provide the detailed structure of the translated NeRF. Our main purpose of this design is to make the translated NeRF render 3D scenes conditioned on the edit code $\mathbf{z}$. As such, the diversity of edits from 2D image-to-image translator can be modeled and represented by a Gaussian distribution of the edit code $\mathbf{z}$. Given a pre-trained original NeRF, we discard its two layers used for density and color estimation. The edit code is concatenated with the intermediate features of the original NeRF and and then fed into two additional MLP networks to obtain the density σ and RGB color for volume rendering. During the optimization process of our GenN2N, the original NeRF parameters except the discared parameters are updated, as well as the newly added MLP networks.

3. Experiment Settings

3.1. Datasets and Evaluation Metrics

Our GenN2N is a unified NeRF-to-NeRF translation framework for various NeRF translation tasks such as text-driven NeRF editing, colorization, super-resolution, inpainting, etc. To verify the effectiveness of our GenN2N, we conduct extensive experiments on various dataset and scenes to compare our GenN2N with existing task-specific specialists, such as Instruct-NeRF2NeRF [4], Palette-NeRF [8], NeRF-SR [16] and SPIn-NeRF [12].

For text-driven NeRF editing, we compare our method with existing methods Instruct-NeRF2NeRF [4] on portrait datasets the face dataset [4], the Fangzhou self-portrait dataset [17] and the Farm and Campsite dataset [4]. The Face dataset [4] comprises 65 images capturing different views of a single person captured by a smartphone. The camera poses are extracted by using the PolyCam app. The Fangzhou self-portrait dataset [17] is collected from users utilizing a front-facing camera, resulting in a total of 100 frames. The Farm and Campsite dataset [4] consists of outdoor 360-degree scenes captured by a camera, containing 250 frames in total, and we only use the former 100 frames for data efficiency. We choose metrics CLIP Text-Image Direction Similarity [4] and CLIP Direction Consistency [4] reported in Instruct-NeRF2NeRF to evaluate editing quality, coupled with Fréchet Inception Distance (FID) [5] to measure generative diversity. More specifically, the reference distribution used for calculating FID is the distribution of 2D edit images, such as InstructPix2Pix [2] editing results in the text-driven editing task. We employ FID to assess how closely our generated results align with the reference distribution.

For colorization, the LLFF dataset [11] for quantitative comparison consists of three large-scale outdoor scenes and five indoor scenes. We also select part of the BlendedMVS dataset [18] for more qulitative results, which covers a variety of scenarios, including cities, buildings, sculptures, and small objects. Following 2D colorization method [6], we use colorfulness score (CF) [9] to measure the richness of color in rgb form and vividness of colorized images.

For NeRF super-resolution, we follow the existing methods NeRF-SR [16] to build high-resolution NeRF with training images down-scaled by $\times 4$. We conduct the comparison with the same datasets, i.e., LLFF dataset [11] mentioned previously and the Realistic Synthetic 360° dataset [10] containing 8 synthetic objects with 100 images. We employ the same metrics as NeRF-SR [16]: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) and Learned Perceptual Image Patch Similarity (LPIPS).

For inpainting, existing methods SPIn-NeRF [12] and OR-NeRF [19] conducted comparative experiments on their

customized dataset [12] of 10 outdoor scenes, including 60 training images with the object and 40 test images without the object for each scene. For fair comparison, we follow these existing methods and also choose the same dataset as well as a statue dataset [12] to train our model and compute test metrics of Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) and Learned Perceptual Image Patch Similarity (LPIPS) on the same test views as previous methods.

3.2. Implementation Details

In our pipeline, we first perform 2D image-to-image translation and then lift those 2D edits up to 3D domain using our well-designed framework. Note that in our GenN2N, we can flexibly choose a different plug-and-play 2D editor to support various NeRF editing task.

For text-driven NeRF editing, we leverage InstructPix2Pix [2] as the 2D image-to-image translator to generate per-frame edited images corresponding to the unified text prompt. During this step, we follow Instruct-NeRF2NeRF [4] to randomly select the image similarity degree S_I from $\{0.5, 2.5\}$, and text similarity degree S_T from $\{6.0, 8.5\}$, producing edited images with significant diversities under the same text prompt.

After 2D editing, we then perform NeRF editing using our well-designed models and loss functions. We implement all our GenN2N based on PyTorch. Follow Instruct-NeRF2NeRF [4], we use the original NeRF trained by using NeRFStudio [15] on the original scene. Then we modify the original NeRF model as our translated NeRF as described in Sec. 2.2. During the training phase, we efficiently sample one image per iteration and extract $16,384$ rays with 48 points per ray in a batch. Our model is trained using Adam optimizer with a learning rate of $1e-2$, running for $10,000-20,000$ iterations per scene. The total training phase takes about $3-8$ hours on one NVIDIA V100 GPU. During the inference phase, we randomly sample $\mathbf{z}$ from a standard Gaussian distribution and render the generated edited NeRF from arbitrary viewpoints with corresponding style defined by the sampled $\mathbf{z}$. The inference time for a translated NeRF need around 250ms. And the rendered time for a 100-frame scene takes about 3 mins.

Notice that, our method is comparable with Instruct-NeRF2NeRF [4] in training time, and is a lightweight feed-forward model during inference without any heavy components on one NVIDIA V100 GPU, as shown in Table 1. Note that Instruct-NeRF2NeRF [4] does not have the inference phrase and requires retraining every time to get a new result, and the diversity between different results is small. In contrast, our method can directly perform forward inference by sampling different style codes to generate diverse results.

Method	Train			Inference	
	Time(h)	Iteration	Memory(GB)	FLOPs(G)	Latency(s)
IN2N	2.67	20000	18.32	–	–
Ours	3.47	10000	20.92	131	0.35

Table 1. Comparsion with Instruct-NeRF2NeRF [4] on computational intensity.

4. More Insight Experiments

4.1. Quality of the Generation Space

To validate the quality of our generation space, we conducted an experiment as shown in Fig. 3, where we projected many of our generated results to all training viewpoints (top) and performed image retrieval to find the closest match in the training data (bottom). As illustrated in Fig. 3, many of our generated results are not present in the training data (InstructPix2Pix [2]), demonstrating that our generation space is learned well.

4.2. Comparison with Naive Altering Instruct-Pix2Pix Parameters in Instruct-NeRF2NeRF

We conducted this experiment as shown in Fig. 9, which demonstrates that altering the sampling parameters of the underlying 2D edit model cannot effectively increase diversity in Instruct-NeRF2NeRF [4]. Instruct-NeRF2NeRF [4] collapses on diversity due to two reasons. Firstly, the conditioning of InstructPix2Pix [2] on the current NeRF rendering significantly collapses the diversity of Instruct-Pix2Pix [2]'s edit results, resulting in highly homogeneous editing outcomes. Secondly, Instruct-NeRF2NeRF [4] cannot ensure consistent edit directions during each iteration of update and edit, resulting in an average edit mode.

4.3. Interpolation of the Edit Code

We randomly sample two edit code $\mathbf{z}_1$ and $\mathbf{z}_2$ from Gaussian Distribution, and linearly interpolate nine latent code z by $\mathbf{z} = \alpha * \mathbf{z}_1 + (1-\alpha) * \mathbf{z}_2$. Then we use these latent codes to directly inference the translated NeRFs. As shown in Fig. 10, the rendering style of our translated NeRF model is highly related to the edit code and the style changes linearly when the interpolation weight α changes linearly. And the 3D view consistency of the rendering scene is always maintained during the interpolation process.

4.4. Ablation on M

The number of edits per perspective M (default is 3) has little impact on the results. As each edit for every viewpoint is distinct, and we characterize this space by collectively utilizing all edits from different views (around 100). While a small M value can have an impact on the overall data volume, once it reaches a certain threshold (*e.g.* 3), the effect is negligible, as shown in Table 2.

Figure 2. **Application of our GenN2N for NeRF zoom out.** We crop the original scene (right) as the input (left) scene and perform zoom out by our GenN2N. We show our rendered views with different edit codes (inference $1-4$) in our inference stage. As can be seen, our GenN2N can acheive NeRF zoom out application so as to enlarge the original input 3D scene by generating plausible content with 3D geometry consistency.

Figure 3. **Image retrival results of our GenN2N for Text-driven NeRF Editing.** We projected many of the generated results to all training viewpoints (top), and then performed image retrieval to find the closest match in the training data generated by InstructPix2Pix [2] (bottom).

5. More Applications

As we demonstrated before, we can achieve NeRF-to-NeRF text-drive 3D editing, super-resolution, colorization, and inpainting. To further demonstrate the versatility of our framework, we also provide two applications, NeRF-to-NeRF zoom out and text-driven inpainting, of our GenN2N that have not been explored by previous methods.

5.1. Zoom Out of NeRF

Zoom out of NeRF is to extend an input NeRF along the input region to enlarge the NeRF scene. Similar to inpainting, we also use LaMa [14] as the 2D image-to-image translator in our GenN2N to solve the NeRF zoom out problem. Given source multi-view images of a 3D NeRF scene, we set the zoom out ratio as 1.25 for image width and height to enlarge

M	CLIP Text-Image Direction Similarity↑	CLIP Direction Consistency ↑	FID ↓
M=1	0.2635	0.9610	123.505
M=3	0.2807	0.9650	91.823
M=5	0.2835	0.9638	86.377

Table 2. **Ablation of M.**

the source images. We first automatically generate binary masks for the zoom out region and then employ LaMa [14] to recover those zoom out regions. Since different content may be generated for zoom out regions in different 2D images from different viewpoint, it is difficult to ensure the 3D consistency in those zoom out regions. We show qualitative results of our zoom out application in Fig. 2. As we can see that our method can successfully ensure the 3D consistency of those zoom out regions and generate reasonable NeRF scenes.

5.2. Text-driven NeRF Inpainting

Text-driven NeRF Inpainting is similar to inpainting, but with the added restriction of text instructions. For the text-driven inpainting task, we use Blended Latent Diffusion [1] as the 2D image-to-image translator, applying inpainting with text instructions. We get the mask in the same way mentioned in the inpainting task. Moreover, we found that if the text prompt is not provided as a guidance, the diffusion model tends to inpaint unreasonable content or monotonous colors close to the surroundings, instead of drawing meaningful objects. So we artificially set up the required text prompt or used the visual question answering model [3] to get answers to the "imagine what the white area might be" question. For example, in this way, we can generate plausible 2D content in the mask area with the guidance of text instructions. After 2D editing, we then perform our proposed optimization to obtain the translated 3D NeRF scene. Qualitative results of our Text-driven NeRF inpainting results are shown in Fig. 4, where we can see that the area of mask is filled with content that matched the description of the text, such as various sunglasses.

6. Qualitative Results Gallery

We provide more qualitative in Fig 5, Fig. 6, Fig. 7, and Fig. 8. For better visualization, we refer the reader to our project page: https://xiangyueliu.github.io/GenN2N/.

7. More Quantity Results

We provide more qualitative in Table 3 and Table 4. For better visualization, we refer the reader to our project page.

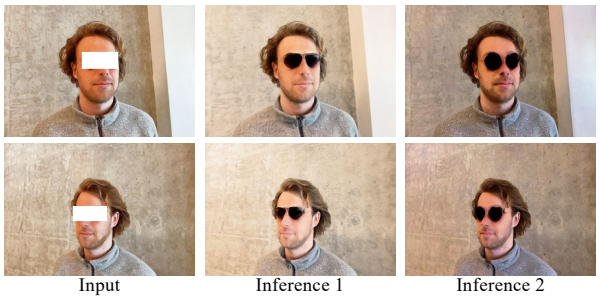

Figure 4. **Application of our GenN2N for text-driven inpainting.** We use "sunglasses" as the text condition to guide the inpainting process.

Method	CLIP Text-Image Direction Similarity↑	CLIP Direction Consistency ↑	FID ↓
Instruct-NeRF2NeRF[4]	0.1383	0.9624	101.219
Ours	**0.1583**	**0.9683**	**93.688**

Table 3. **More quantitative results on text-driven editing.** We compare our method with the state-of-the-art method Instruct-NeRF2NeRF [4] with metrics reported in the latter. Following Instruct-NeRF2NeRF [4], we conduct quantitative evaluation on bear dataset with 3 editing prompts and face dataset with 7 editing prompts.

Method	CF ↑
PaletteNeRF[8]	58.065
Ours	**75.960**

Table 4. **More quantitative results on colorization.** We compare our method with the state-of-the-art method PaletteNeRF [8]. Since the latter does not provide an appropriate metric for comparison, we use colorfulness score (CF) [9] to measure the vividness of colorized images. We choose the dataset used by PaletteNeRF [8], namely, Fern, Horns, Flower and Orchids from the forward-facing LLFF dataset [11].

References

[1] Omri Avrahami, Ohad Fried, and Dani Lischinski. Blended latent diffusion. *ACM Transactions on Graphics (TOG)*, 42 (4):1–11, 2023. 5

[2] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. *arXiv preprint arXiv:2211.09800*, 2022. 1, 2, 3, 4, 9

[3] Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*, 2023. 5

[4] Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. *arXiv preprint arXiv:2303.12789*, 2023. 2, 3, 5, 9

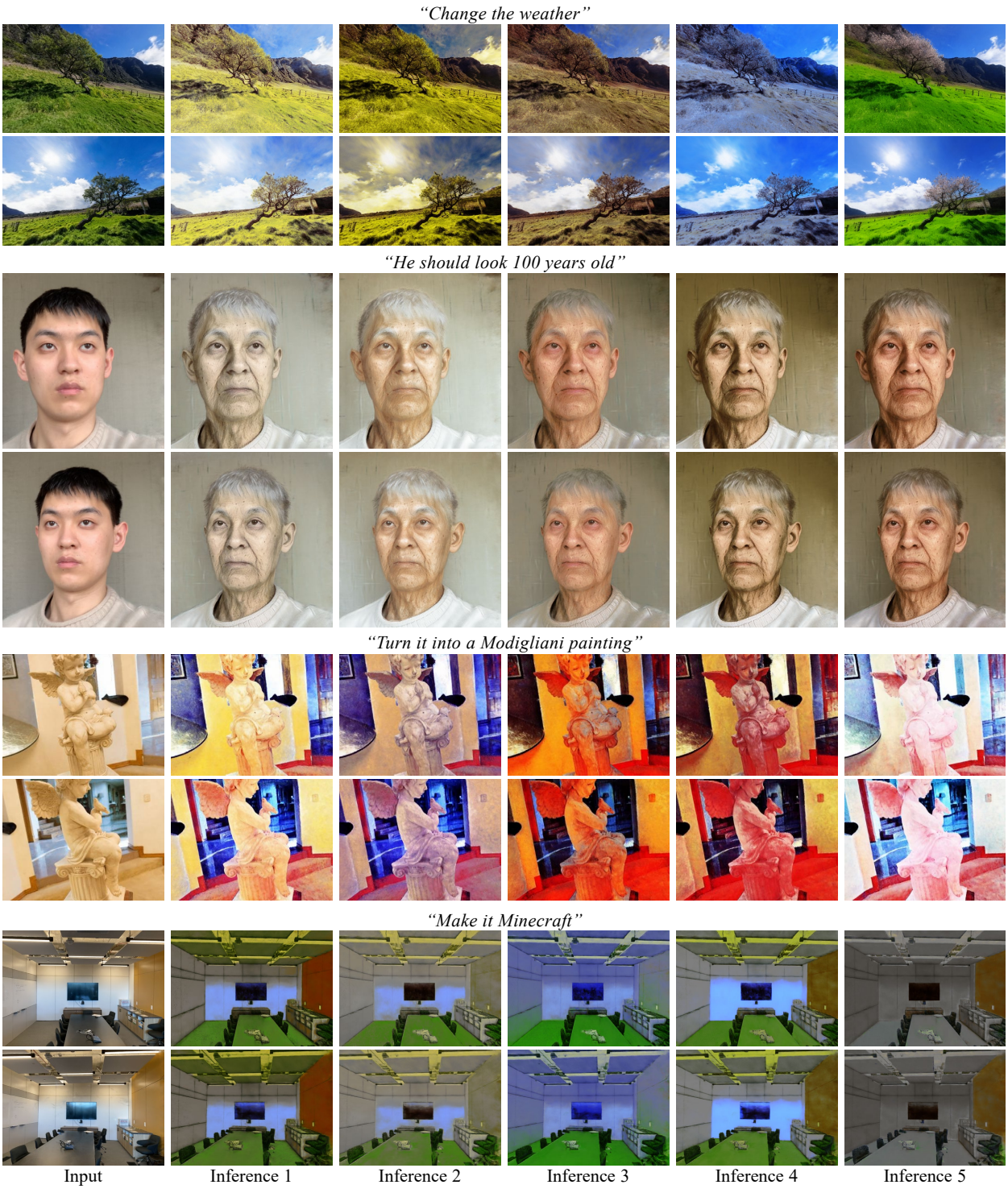

Figure 5. **Qualitative results for Text-driven Editing.** We show the original 3D scene (left), our inference results (right) with different edit codes to show the diversity ability of our method. We can see that under different edit code, the edited scene with different styles can be rendered with plausible 3D geometry consistency.

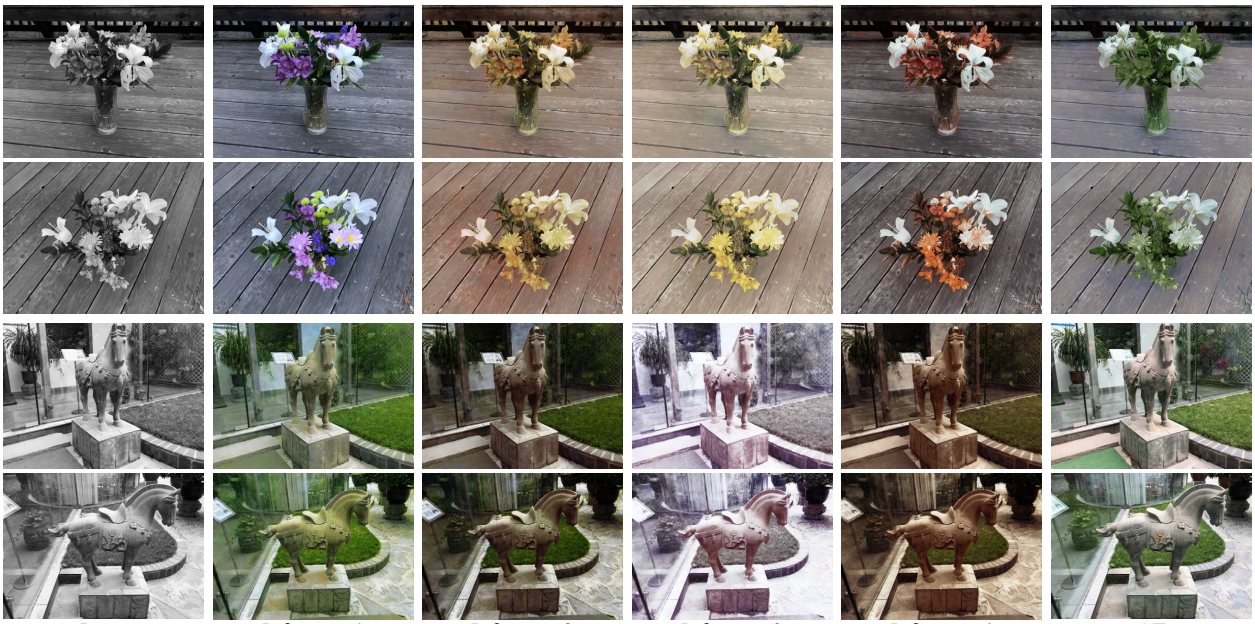

Figure 6. **Qualitative results for NeRF colorization.** For each scene, we show two views of the input gray-scale scene (left), five of our inference edited NeRF rendering results from different edit code (middle), and the Ground-truth scene (right). As can be seen, our GenN2N can produce plausible colorization results while maintaining the 3D multi-view consistency.

[5] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 2

[6] Xiaoyang Kang, Tao Yang, Wenqi Ouyang, Peiran Ren, Lingzhi Li, and Xuansong Xie. Ddcolor: Towards photo-realistic and semantic-aware image colorization via dual decoders. *arXiv preprint arXiv:2212.11613*, 2022. 1, 2

[7] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023. 1

[8] Zhengfei Kuang, Fujun Luan, Sai Bi, Zhixin Shu, Gordon Wetzstein, and Kalyan Sunkavalli. Palettenerf: Palette-based appearance editing of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20691–20700, 2023. 2, 5

[9] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 577–593. Springer, 2016. 2, 5

[10] 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. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 2

[11] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2, 5

[12] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20669–20679, 2023. 1, 2, 3

[13] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 2

[14] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2149–2159, 2022. 1, 4, 5

[15] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, et al. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages

Figure 7. **Qualitative results for NeRF super-resolution.** We show the original low-resolution input (left), our super-resolution result (middle) and the ground-truth (right) all with their zoom in results for better visualization.

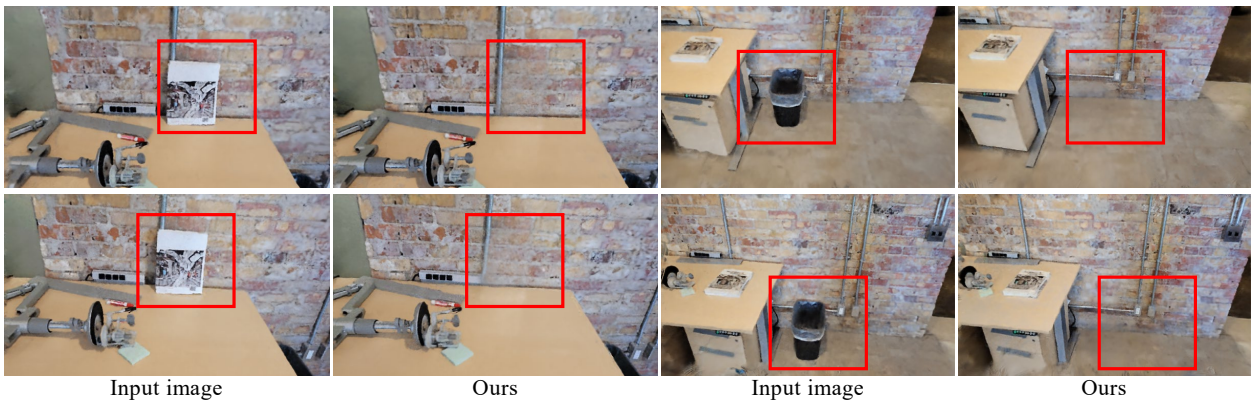

Figure 8. **Qualitative results for NeRF inapinting.** We highlight the removed objects and our inpainting regions using the red boxes.

1–12, 2023. 3

[16] Chen Wang, Xian Wu, Yuan-Chen Guo, Song-Hai Zhang, Yu-Wing Tai, and Shi-Min Hu. Nerf-sr: High quality neural radiance fields using supersampling. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 6445–6454, 2022. 1, 2

[17] Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Nerf-art: Text-driven neural radiance fields stylization. *IEEE Transactions on Visualization and Computer Graphics*, 2023. 2

[18] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1790–1799, 2020. 2

[19] Youtan Yin, Zhoujie Fu, Fan Yang, and Guosheng Lin. Or-

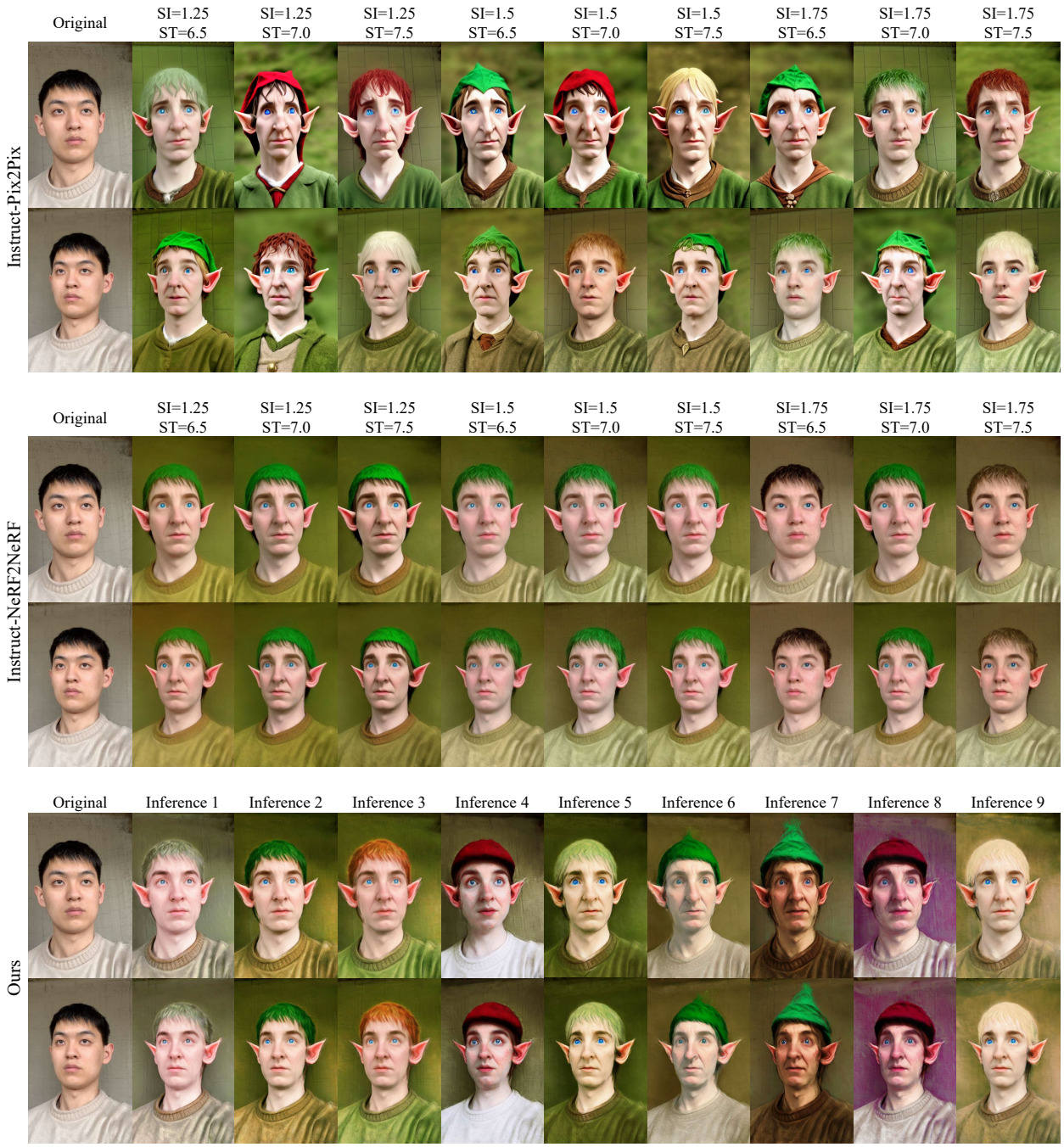

Figure 9. **Diversity and view consistency comparisons of our GenN2N with existing methods.** The same text instruction of "Turn him into the Tolkien Elf" is used for all the methods. As can be seen, InstructPix2Pix [2] can produce diverse results in 2D but 3D multi-view consistency is not ensured. Though Instruct-NeRF2NeRF [4] can ensure the multi-view consistency, its results show little variance. In contrast, our GenN2N can produce diverse editing results and address the 3D geometry consistency at the same time.

nerf: Object removing from 3d scenes guided by multiview segmentation with neural radiance fields. *arXiv preprint arXiv:2305.10503*, 2023. 2

[20] Zongsheng Yue, Jianyi Wang, and Chen Change Loy. Resshift: Efficient diffusion model for image super-resolution by residual shifting. *arXiv preprint arXiv:2307.12348*, 2023. 1

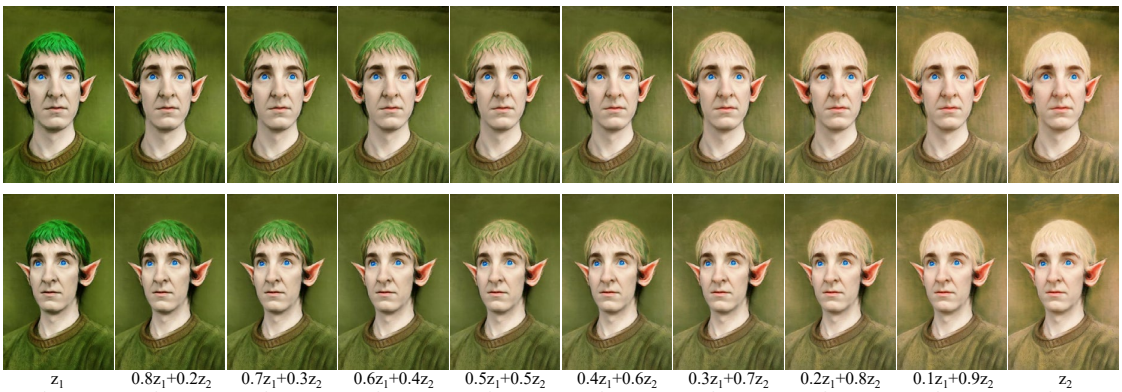

Figure 10. **Interpolation of the edit code.** We render the same views using the edit code $\mathbf{z}$ produced by interpolation of two random sampled edit code $\mathbf{z}_1$ and $\mathbf{z}_2$ by: $\mathbf{z} = \alpha * \mathbf{z}_1 + (1 - \alpha) * \mathbf{z}_2$. As can be seen, the rendering style of our translated NeRF model is highly related to the edit code and the style changes linearly when the interpolation weight α changes linearly. Note that, the 3D view consistency of the rendering scene is always maintained during the interpolation process.