SIn-NeRF2NeRF: Editing 3D Scenes with Instructions through Segmentation and Inpainting
Performance Improvement Track

Jiseung Hong
KAIST
Computer Science
jiseung.hong@kaist.ac.kr

Changmin Lee
KAIST
Computer Science
lcm914@kaist.ac.kr

Gyusang Yu
KAIST
Mechanical Engineering
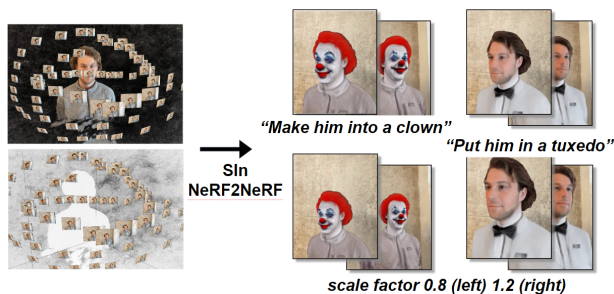peter.yu@kaist.ac.kr

Figure 1. **Overview**: We propose the method SIn-NeRF2NeRF, enabling wide range of object edition including translation, rotation, and scale changes.

Abstract

TL;DR Perform 3D object editing selectively by disentangling it from the background scene.

Instruct-NeRF2NeRF [3] (in2n) is a promising method that enables editing of 3D scenes composed of Neural Radiance Field (NeRF) [7] using text prompts. However, it is challenging to perform geometrical modifications such as shrinking, scaling, or moving on both the background and object simultaneously. In this project, we enable geometrical changes of objects within the 3D scene by selectively editing the object after separating it from the scene. We perform object segmentation and background inpainting respectively, and demonstrate various examples of freely resizing or moving disentangled objects within the three-dimensional space. Code is available at:
https://github.com/KAISTChangmin/SIn-NeRF2NeRF

1. Introduction

The current leading method for representing 3D scenes is Neural Radiance Fields (NeRF), which can generate realistic novel views from a sparse set of images, given the camera parameters. The ability to freely and stably edit such 3D scenes is one of the most critical technologies for implementing the real world in VR/AR applications.

The introduced Instruct-NeRF2NeRF is a powerful tool that allows humans to edit the scene itself based on given input. Additionally, SPIn-NeRF [8] has created a background scene by interactively removing selected objects from the entire scene. InpaintNeRF360 [11] is a recent work that appropriately fuses the concepts of both. Unlike SPIn-NeRF, which is limited to inpainting results of frontal scenes, InpaintNeRF360 demonstrated the ability to perform inpainting across 360-degree scenes using human prompts.

The 3D scene edition results from Instruct-NeRF2NeRF shows promising results in geometrical addition and minor perturbations, while some of the updates in objects were not reflected in the background scene. For instance, when an object goes through geometrical shrinking, the original part of the background wouldn't fill the perturbed part of the object.

In this project, our objective is to achieve a refined object edition by separating the object from the scene, restoring the original portion of the background without the object through inpainting, and selectively editing the object. Ultimately, this approach enables the disentanglement of the object and background, facilitating more precise and effective modifications in 3D scene reconstruction.

Therefore, we propose SIn-NeRF2NeRF (sn2n) that modifies Instruct-NeRF2NeRF (in2n) based on the code of SPIn-NeRF to perform object edition. The key challenge lied in the modification of in2n framework to get RGBA image as input, and yield the object scene with no background. We elaborate the detailed implementation of this and other processes in the subsequent sections. We verify the fidelity of our model based on the datasets provided by in2n and SPIn-NeRF. Furthermore, we confirm its functionality on a custom dataset, thereby demonstrating its robustness and scalability.
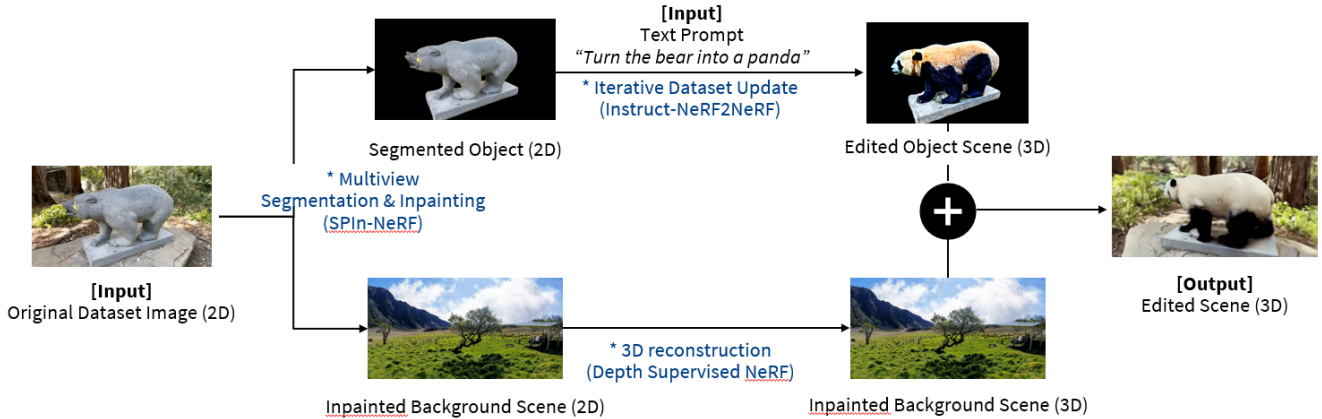
Figure 2. **Main Framework**: Our method processes an input consisted of original NeRF scene and a text prompt, yielding an edited NeRF scene with the disentangled object. Detailed information regarding the implementation can be found in Section 3.

2. Method Summary

As shown in Figure 2, sn2n receives images of a scene from multiple views and object masks as input. Additionally, it takes a text prompt as an input to output an edited 3D scene where the object is modified. During this process, techniques such as multiview segmentation and SPIn-NeRF were utilized to disentangle the object and the background.

2.1. Problem Setup

As specified above, sn2n takes a text prompt and a NeRF scene as inputs to separate the scene into object and background components. For the object, it performs scene editing and scaling, while for the background, it applies 3D inpainting, and then merges the two scenes. In the process of editing the object scene, it employs a method $I^v_{object,i+1} \leftarrow U_\theta(I^v_{object,i}, t; I^v_{object,0}, c^T)$, for an unedited object image $I^v_{object,0}$, a text instruction c^T, and a noisy input z_t, where $z_0 = \varepsilon(I^v_{object,i})$. Here, U_θ is defined as DDIM sampling, consistent with the definition in in2n. Meanwhile, for the background scene, a 3D inpainting method called SPIn-NeRF was used. For further details in SPIn-NeRF, refer to Section 2.3. Consequently, we merge two NeRF scenes where the object is disentangled from the background.

2.2. Baseline Inspection

Instruct-NeRF2NeRF is a novel work performing object editing within NeRF scenes and serves as the baseline method for our project. In the Section 4, we compare the outcomes of applying in2n after separately learning the object and background scene using the SPIn-NeRF framework, against when in2n applied to the original scene. Instruct-NeRF2NeRF takes a typical NeRF scene as input and follows an iterative dataset update pipeline to update the scene. Per each iteration step, this dataset update pipeline takes the novel scene image with added random gaussian noise and prompt from the rendered view as input, and outputs an image updated via InstructPix2Pix [1] (ip2p). Initially the difference between the text prompt and the original scene is huge, and 2D images reconstructed according to the prompt turns out 3D inconsistent. However as learning continues, the scene gradually changes starting from aspects across views. This can be seen in Figure 3. Iterative dataset update is showing incremental changes to (such as pointy ears, blue eyes, green t-shirts, etc.) per iteration.

(a) Original   (b) Iter-5000   (c) Iter-10000   (d) Iter-15000

Figure 3. Iterative Dataset Update (IDU)

2.3. SPIn-NeRF

SPIn-NeRF plays a pivotal role in our project by inpainting the background of the original scene with sparse object selection. SPIn-NeRF starts with training the given NeRF scene and acquires disparity maps for novel views. Consequently, the the object mask is given as input and it perform LaMA inpainting in every training views. Additional LaMA inpainting is done to the initially acquired disparity map. Finally the background scene is constructed using these two inpainted rgb image and disparity inpainted image sets.

3. Implementation Details

We have implemented the entire process from data preparation/preprocessing (2D image multiview segmentation) to applying in2n on the object scene and synthesiz-

ing the 3D object scene with the 3D inpainted background scene. We borrowed 3D inpainting of the background scene algorithm from LaMa [10]. The detailed methods and algorithms applied at each step are as follows.

3.1. 2D Image Multiview Segmentation

We implemented code that allows interactive 2D image multiview segmentation using the Segment Anything Model (SAM) [5]. It takes the original 2D image set as input, separates the object and background by providing sparse annotations for each frame, and retrieves the object mask.

3.2. Object Scene Reconstruction & Applying in2n

For the object scene, we train a DSNeRF [2] using an RGBA image set concatenated from the object mask and object image. During this process, we implement the random background color technique borrowed from the instant-ngp [9] code to ensure effective training of the object scene with a transparent background. The random background color technique involves alpha blending each view with a random color. Figure 4 shows significant difference in results compared to when technique is not applied.
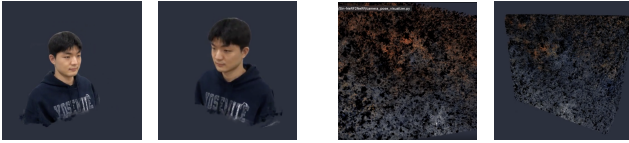

Figure 4. Object scene with (left two) and without (right two) the random background color method.

We develop a code to edit the object scene based on text prompts by porting the in2n pipeline, especially the Iterative Dataset Update (IDU) algorithm, from Nerfacto to DSNeRF. The IDU algorithm used in our team's code is specified in Algorithm 1. Our novelty lies in the manipulation of object scenes, since we modify the in2n framework to get input of RGBA images, as opposed to the conventional RGB images. As mentioned, alpha blending is applied and outputs the object scene within the RGBA images, with no background in 3D space when reconstructed. The modified framework preserves the integrity of the original format while stably incorporate our targeted modifications.

3.3. Background Scene Reconstruction

Using the code provided by SPIn-NeRF, we train the DSNeRF for the inpainted background scene. We perform 2D inpainting on the original 2D background images, where the object part is segmented out, using LaMa, and then trained the 3D inpainted background scene using RGB loss, disparity loss, LPIPS loss, and true depth loss.

Algorithm 1: Iterative Dataset Update Process

1 Dataset $\leftarrow$ original images I_0^v for viewpoints v;
2 **for** *each iteration* **do**
3   Randomly order viewpoints v;
4   **for** d *image updates* **do**
5   **for** *each viewpoint in ordered* v **do**
6   1. Alpha blend RGBA image with black background (RGBA $\rightarrow$ RGB);
7   2. Update image using ip2p;
8   3. Segment the object (RGB $\rightarrow$ RGBA);
9   **end**
10   **end**
11   **for** n *NeRF updates* **do**
12   Sample random rays from entire training dataset;
13   Update NeRF with a mix of old and newly updated images;
14   **end**
15 **end**

3.4. 3D NeRF Scene Synthesis

The method for synthesizing the edited object scene from Section 3.2 with the inpainted background scene using SPIn-NeRF was adopted from ml-neuman [4]. Since both scenes share the same camera parameters, we sort the sampled points for the same rays generated in the object and background scenes by their depth values Z as in Equation 1.

$$P_{\text{sorted}} = \{t_i \mid t_i \in P_{\text{object}} \cup P_{\text{bkg}} \text{ and } Z(t_i) \text{ is sorted}\} \quad (1)$$

3.5. Object Transformation

The process of transforming (scaling, translation, and rotation) a disentangled object scene from the background commences with the utilization of COLMAP to ascertain the coordinates P of the 3D object. Subsequently, these coordinates are transformed around the centroid O using a transformation factor $scale$, $rotate$, and $trans$, thereby acquiring the 3D coordinates of the transformed object P'. This transformation doesn't change the actual 3D model but alters how it is perceived from the camera's viewpoint during the rendering process.

4. Experimental Results

We confirm the robustness of our framework by dataset provided by SPin-NeRF team and in2n team. We focus primarily on presenting qualitative results, complementing these with quantitative results using CLIP metric. The face scene is mainly presented due to its intuitiveness. The method can be also applied for other datasets. We train

our Segmented NeRF (object scene) for 2k iterations, which takes about 30 minutes on a single RTX 4060 Ti. Running multiview inpainter (background scene) also takes about 30 minutes on a same device.

4.1. Qualitative Results

The advantage of our method is that by separately training the object and background and then merging them, it allows for not only editing through prompt input but also free adjustment of position and scaling. In Figure 5, we can observe the free panning of an object in the scene.
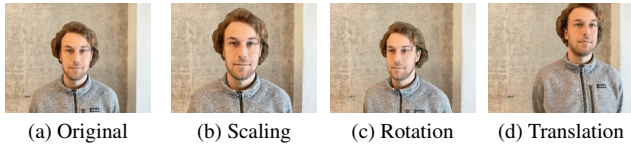

(a) Original   (b) Scaling   (c) Rotation   (d) Translation

Figure 5. **Qualitative Results**: Object Transformation.

Baseline Comparison

It can be confirmed from Figure 6 that in2n has been stably implemented within the SPIn-NeRF framework. When comparing the results of sn2n with in2n for various prompts, it can be observed that the outcomes are similar because both employ the same instruct pix2pix's 2D diffusion image editing. However, in the detailed aspects below, it is evident that sn2n has a geometric advantage over in2n, such as smoother background reconstruction.
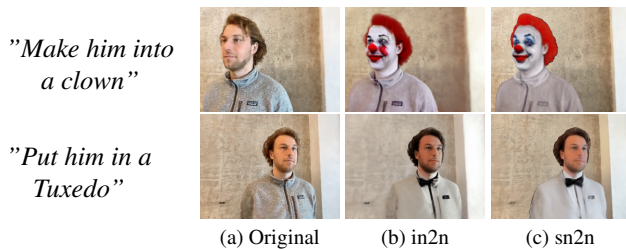

"Make him into a clown"

"Put him in a Tuxedo"

(a) Original   (b) in2n   (c) sn2n

Figure 6. Comparison between in2n & sn2n

4.2. Quantitative Results

As mentioned in the baseline paper [3], editing is a subjective task, so the focus should be on qualitative results rather than quantitative results when measuring performance improvement. However, as in ip2p and in2n, we analyze quantitatively in two ways. The results of comparing sn2n to the baseline model in2n based on the alignment of 3D edits with text instructions and the temporal consistency of edits across views are shown in Table 1. We evaluated the metrics based on the face scene with the text prompt *"Make him into a clown"* and *"Put him in a Tuxedo"*.

Face Scene \	CLIP Text-Image Direction Similarity ↑		CLIP Direction Consistency ↑	
	in2n	Ours	in2n	Ours
→ Clown	**0.2372**	0.2081	0.9071	**0.9117**
→ Tuxedo	0.0251	**0.0481**	0.8451	**0.8599**

Table 1. **Quantitative Results**: We offer quantitative measures to assess how well the edits align with the text and the consistency of subsequent frames in CLIP space.

The CLIP values of sn2n are comparable to, or even higher than those of in2n. It confirms the robust performance of the scene reconstruction process of our method.

5. Conclusion

In this project, we aim to achieve high-quality object editing within 3D scene editing, utilizing interactively segmented input and human prompts. We perform parallel NeRF learning for the received object and scene, where the object is edited using the Instruct-NeRF2NeRF technique as desired, and the scene undergoes inpainting using the SPIn-NeRF method. We demonstrate enhanced object edition (such as translation, scaling, etc.) on people, objects, and large-scale scenes as observed in in2n.

However, the results of object editing significantly varied depending on the performance of iterative dataset update using ip2p. Also, due to the gradual nature of updates, changes are limited to texture and feature modifications rather than dynamic changes like altering the pose. Advancing towards more robust and dynamic object edition capabilities is a direction for future research in this field, and we expect this can be done with RGBA image updating methods like LayerDiffusion [6].

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions, 2023. 2

[2] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free, 2022. 3

SPIn-NeRF	Multiview inpainting
Instruct-NeRF2NeRF	Iterative dataset update
SAM	Interactive object segmentation and mask generation
COLMAP	Get true depth using camera parameters
ML-Neuman	Merge two different NeRF scenes
Instant-ngp	Random background color

Table 2. **Acknowledgement**: Borrowed codes.

[3] Ayaan Haque, Matthew Tancik, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions, 2023. 1, 4

[4] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video, 2022. 3

[5] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023. 3

[6] Pengzhi Li, QInxuan Huang, Yikang Ding, and Zhiheng Li. Layerdiffusion: Layered controlled image editing with diffusion models, 2023. 4

[7] 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, 2020. 1

[8] 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, 2023. 1

[9] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4):1–15, July 2022. 3

[10] 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, 2021. 3

[11] Dongqing Wang, Tong Zhang, Alaa Abboud, and Sabine Süsstrunk. Inpaintnerf360: Text-guided 3d inpainting on unbounded neural radiance fields, 2023. 1