Undergraduate Project Report
2023/24

Research on conditional neural radiance fields

Date: 10-04-2024

Table of Contents

Research on conditional neural radiance fields

Mid-term progress report

Supervision log

Risk and environmental impact assessment

Abstract

This project introduces a new approach that combines natural language processing with Neural Radiation Fields (NeRF) to enable intuitive command-based 3D scene editing and consistent rendering across different viewpoints. Instruct-Pix2Pix is a conditional diffusion model conditional on textual descriptions, and the approach in this project utilizes Instruct-Pix2Pix's. The approach in this project utilizes Instruct-Pix2Pix to iteratively refine a pre-trained NeRF model using edited 2D images generated from textual descriptions. This process ensures that the modifications are evenly integrated into the 3D scene, maintaining spatial and visual consistency. This project employs an improved model built upon the Instruct-NeRF2NeRF framework, and the effectiveness of this project is demonstrated through qualitative analysis of various scenes, showing that the method not only conforms to the semantic intent of the textual instructions, but also preserves the high fidelity and realism inherent in NeRF-generated environments. This work expands the utility of NeRF modeling in practical applications such as virtual reality, movie production, and real-time interactive environments, providing a user-friendly interface for dynamic scene manipulation.

Keywords

NeRF (Neural Radiance Fields), 3D Scene Editing, NLP (Natural Language Processing), Iinstruction-based Editing, Conditional Diffusion Models

摘要

本项目介绍了一种将自然语言处理与神经辐射场 (NeRF) 相结合的新方法，可实现基于指令的直观三维场景编辑，并在不同视角下实现一致性高的渲染效果。Instruct-Pix2Pix 是一种以文字描述为条件的条件扩散模型，本项目的方法利用 Instruct-Pix2Pix 的功能，使用由文字说明生成的编辑过的二维图像迭代完善预先训练好的 NeRF 模型。这一过程可确保修改内容均匀地融入三维场景，保持空间和视觉的一致性。本项目在 Instruct-NeRF2NeRF 流程的基础上使用了效果更好的模型，通过对各种场景的定性分析证明了本项目的有效性，表明该方法不仅符合文本指令的语义意图，而且还保留了 NeRF 生成环境固有的高保真和逼真度。这项工作拓展了 NeRF 模型在虚拟现实、电影制作和实时交互环境等实际应用中的实用性，为动态场景操作提供了一个用户友好界面。

关键词

NeRF（神经辐射场）、三维场景编辑、NLP（自然语言处理）、基于指令的编辑

Chapter 1: Introduction

1.1 Unseen Viewpoint Synthesis – NeRF

Neural Radiance Fields (NeRF) [1] have emerged as a powerful technique for representing and rendering complex 3D scenes. Unlike traditional surface-based representations, NeRF leverages neural networks to capture intricate details and realistic lighting effects, enabling high-quality synthesis of novel views from a sparse set of input images.

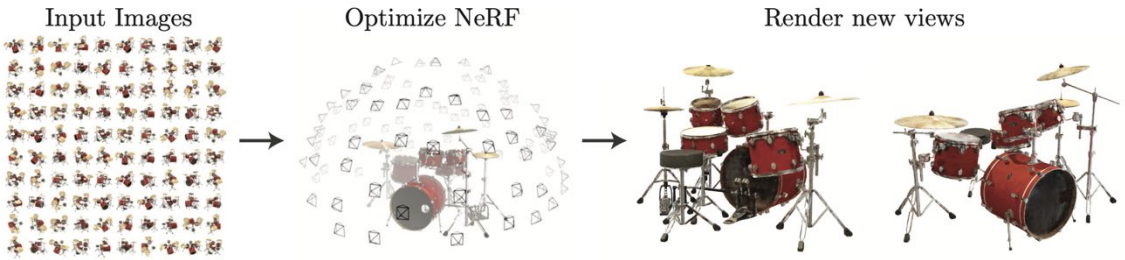

Figure 1 Overview of NeRF

At its core, NeRF represents a 3D scene as a continuous volumetric function, parametrized by a multilayer perceptron (MLP). This function maps a 5D coordinate, consisting of a 3D spatial position (x, y, z) and a 2D viewing direction (θ, φ), to the corresponding emitted color c and volume density σ at that point in space. By querying this function at any desired location and viewing angle, NeRF can render photorealistic images of the scene from previously unseen viewpoints.

To train a NeRF, a dataset of posed images capturing the scene from various angles is required. The MLP is optimized to reproduce these reference views by minimizing the difference between the rendered and ground truth pixel colors. This is made possible by NeRF's differentiable rendering formulation. Each pixel is associated with a camera ray passing through the scene. By accumulating the color and density contributions at sampled points along this ray using numerical quadrature, the expected color of the pixel can be estimated. Crucially, the rendering operation is differentiable with respect to the MLP weights, allowing the network to be trained end-to-end using gradient descent.

One of the key insights behind NeRF is its modeling of scene appearance as a continuous function in both space and viewing angle. This enables the representation of view-dependent effects like specular reflections and translucency. Moreover, by parametrizing the scene as an implicit neural function, NeRF is not constrained by a fixed spatial resolution and can

theoretically model geometry at arbitrary scales.

The original NeRF formulation makes several simplifying assumptions. It considers only static scenes captured under fixed lighting conditions. The estimated volume density is interpreted as a description of geometry, but NeRF does not explicitly output surface normals or a signed distance function. Though rendering is computationally expensive due to the large number of network queries required per ray, the compact MLP representation enables NeRF to achieve stunning visual quality on complex real-world scenes.

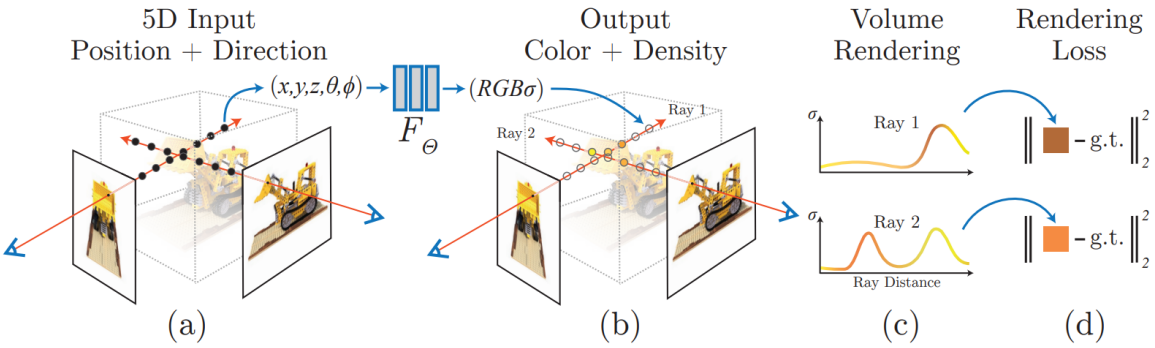

Figure 2 NeRF Rendering Procedure [1]

An overview of neural radiance field scene representation and differentiable rendering procedure [1] is illustrated in Fig.1. Synthesize images by sampling 5D coordinates (location and viewing direction) along camera rays (a), feeding those locations into an MLP to produce a color and volume density (b), and using volume rendering techniques to composite these values into an image (c). This rendering function is differentiable, so can optimize the scene representation by minimizing the residual between synthesized and ground truth observed images (d). [1]

Since its introduction, NeRF has inspired a wealth of subsequent research aiming to extend its capabilities and improve its efficiency. For example, transient and variable illumination can be modeled by factoring appearance into static and time-varying components [2]. Explicit surface representations can be extracted to enable faster rendering and mesh-based editing [3]. Generalization ability to unseen scenes can be improved through the learning of prior distributions over NeRF parameters [4].

With NeRF as the fundamental framework, many exciting avenues have opened up for photorealistic 3D scene representation and novel view synthesis. As research continues to advance, NeRF-based techniques promise to revolutionize fields such as virtual reality, visual effects, and robotics.

1.2 Controls for NeRF generated content

Despite the rapid progress in neural rendering, editing and manipulating NeRF-based 3D content remains a challenge. The implicit and overparametrized nature of the neural scene representation does not readily allow for intuitive control and modification. Whereas traditional 3D models can be interactively edited using established software tools and interfaces, NeRF scenes are defined by a large number of network weights with no inherent semantic meaning. This has sparked research into techniques for controlling and manipulating NeRF-generated content.

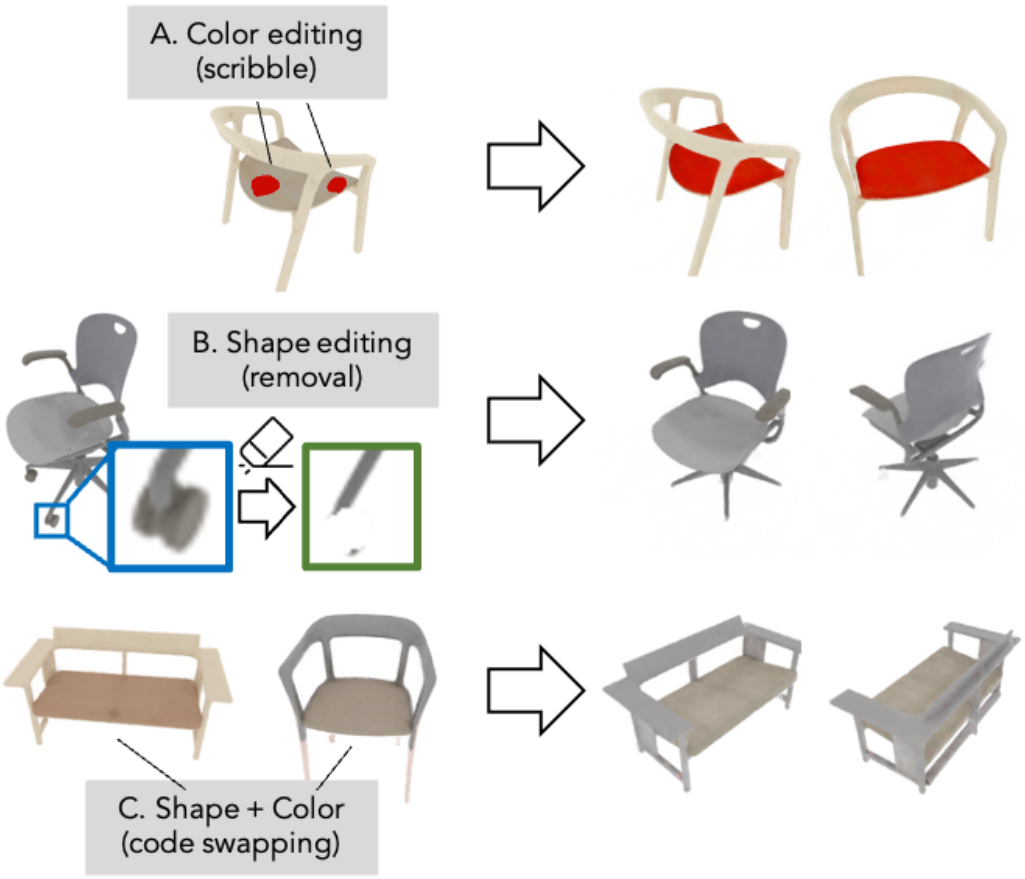

Figure 3 Example of NeRF Editing [34]

1.2.1 Physical Editing Approaches

One avenue for NeRF editing is to incorporate physical priors and inductive biases into the optimization process. By imposing constraints based on real-world properties like material attributes, lighting, and object transformations, specific aspects of the scene can be modified while preserving overall consistency.

For instance, NeRFactor [5] decomposes a scene into its constituent geometry, reflectance, and illumination components. This disentangled representation allows for physically-based edits such as changing surface materials or inserting virtual lights. Similarly, other works have explored NeRF editing via object-centric decomposition, material estimation, and lighting manipulation.

Another line of work focuses on spatial edits and scene composition. Several methods propose to represent scenes as multiple NeRFs, each corresponding to a separate object or region. By applying rigid or non-rigid transformations to these NeRFs and alpha-blending their outputs, objects can be rearranged and deformed while maintaining view consistency. Bounding-box annotations can also be used to enable drag-and-drop insertion and deletion of objects.

Physics-based simulation is yet another approach for augmenting NeRF scenes with dynamic phenomena. For example, the concurrent work PhysNeRF [6] uses a particle-based liquid simulator to generate realistic fluid effects which are then blended with a base NeRF scene. The simulator is differentiable, allowing its parameters to be optimized to match a target video sequence.

1.2.2 Semantic and Stylistic Manipulation

A separate class of NeRF editing techniques focuses on higher-level semantic and stylistic manipulations. These methods often leverage learned feature representations from pretrained models to guide the editing process.

Neural style transfer has been successfully adapted to the NeRF domain, enabling the transfer of artistic styles to 3D scenes. Given a style reference image, these approaches optimize the NeRF to match the desired style while preserving the scene's content and geometry. The key idea is to match feature activations extracted from the NeRF renderings and stylized images using models like VGG trained for object recognition.

More recent works have explored using natural language to specify target attributes for NeRF editing. The CLIP model [7], which learns a joint embedding space for images and text, is frequently employed to evaluate the semantic similarity between a rendered scene and a text prompt. For example, CLIP-NeRF [8] and NeRF-Art [9] use a CLIP-based loss to encourage the NeRF to generate images that match a given text description. The prompt can express artistic styles like "impressionist painting" or object-centric modifications like "a red car".

Text-guided NeRF editing has also been approached through the integration of pretrained 2D segmentation and detection models. Distilled Feature Fields [10] and LERF [11] project

features from DINO and GLIDE into a NeRF scene to enable localized edits. A text prompt is used to select a relevant feature layer and region to be modified. These methods can alter properties of individual objects, but still struggle with more complex structural changes.

While semantic NeRF manipulation techniques offer increased flexibility compared to physics-based approaches, they have limitations. Edits are often constrained to the manifold of images and styles represented in the pretrained models. Achieving substantial geometric changes or inserting out-of-distribution objects poses a challenge. Maintaining 3D consistency across edited views is also difficult due to the use of image-space losses.

Nevertheless, NeRF editing is a rapidly evolving research area with immense potential. As more sophisticated generative models and NeRF architectures are developed, we can expect to see continued progress towards intuitive, expressive, and controllable neural scene manipulation.

Chapter 2: Related Work

In this section, we provide a broader context for our work by discussing relevant literature on NeRF editing. We focus on two main categories of approaches: physical editing methods that directly manipulate scene properties, and semantic editing techniques that use learned feature representations to guide stylistic changes.

2.1 Physical Editing of NeRFs

Physical editing of Neural Radiance Fields (NeRFs) [1] involves modifying the scene properties, such as materials, lighting, and geometry, by imposing physics-based priors and constraints during the optimization process. These methods aim to enhance the realism and controllability of NeRF editing.

One approach to physical editing is to specify bounding boxes or regions of interest within the NeRF scene. By defining these spatial constraints, users can composite new objects into the scene or deform existing geometry. For instance, Zhang et al. [5] proposed a method that allows users to interactively place 3D bounding boxes to indicate the desired location and size of object insertions or removals. The NeRF optimization is then constrained to only modify the radiance field within these specified regions, preserving the rest of the scene.

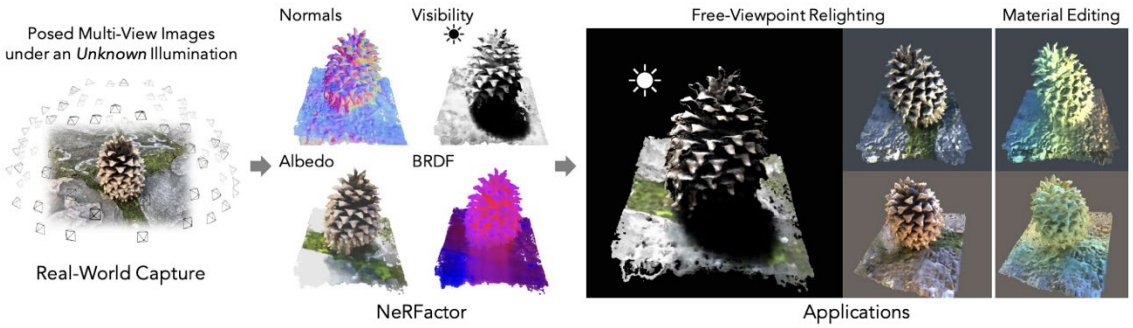

Figure 4 Overview of NeRFactor

Another direction explores the use of physical simulation to apply realistic effects to NeRF scenes. For example, the work by Ost et al. [12] introduces a physics-based framework for editing NeRFs by simulating natural phenomena such as snow accumulation or fluid dynamics. By integrating physical simulation into the NeRF rendering pipeline, they enable the creation of dynamic and physically plausible effects that interact with the scene geometry.

The incorporation of material properties into NeRF editing has also been investigated. Zhang et al. [13] proposed a method to estimate and edit the material properties of objects within a

NeRF scene. By decomposing the radiance field into diffuse and specular components, they enable intuitive editing of surface properties such as roughness, metallicness, and reflectance. This allows users to change the appearance of objects while maintaining consistency with the underlying geometry and lighting.

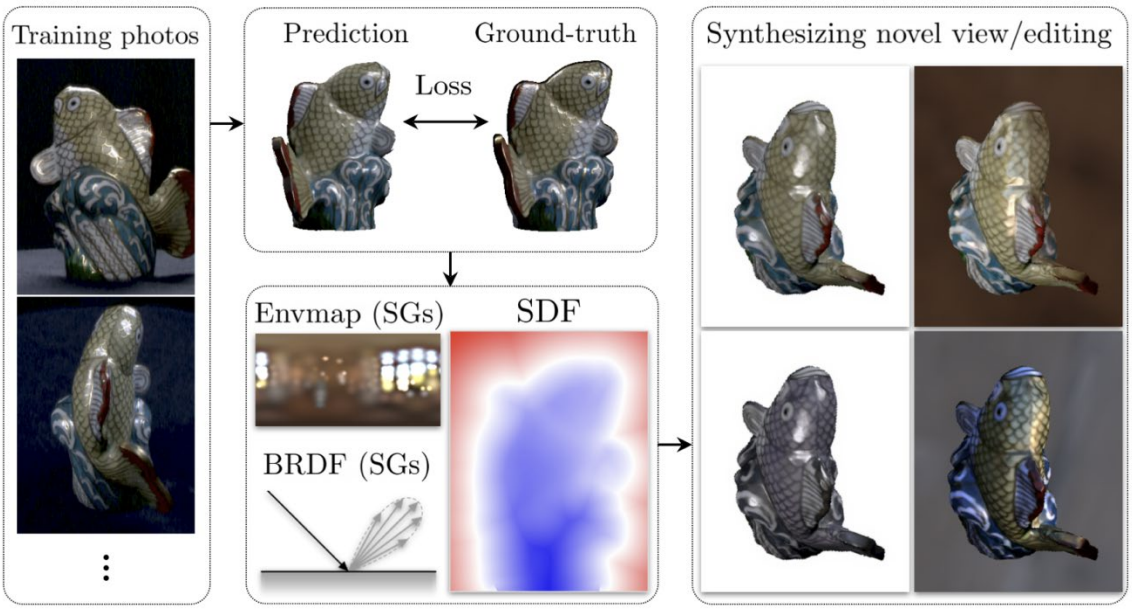

Figure 5 Overview of PhySG

Lighting editing is another important aspect of physical NeRF manipulation. Srinivasan et al. [14] introduced a technique for relighting NeRF scenes by learning a decomposition of the radiance field into direct and indirect illumination components. By disentangling the lighting from the scene representation, they enable flexible editing of illumination conditions, such as changing the direction or color of light sources, or adding and removing shadows.

While these physically-based editing approaches offer powerful tools for modifying NeRF scenes, they often focus on changing specific physical attributes rather than enabling arbitrary creative edits. The user control is typically limited to predefined parameters or constraints, and the editing process may require manual specification of regions or properties. Additionally, the physical accuracy of the edits depends on the fidelity of the simulation or decomposition methods employed.

2.2 Artistic Stylization of NeRFs

Artistic stylization of NeRFs aims to modify the appearance of a scene by transferring the style of a reference image or adapting to a user-specified artistic intent. These methods build upon the success of image stylization techniques and extend them to the domain of 3D scene

representation.

One approach to NeRF stylization is to leverage the power of style transfer networks, such as those based on the pioneering work of Gatys et al. [15]. These networks are trained to extract and recombine the content and style information from input images, allowing the transfer of artistic styles while preserving the overall structure of the content image. Huang et al. [16] proposed a method that applies style transfer to NeRF renderings by optimizing the radiance field to match the style of a given reference image. By minimizing the style loss between the rendered images and the target style, they achieve a consistent stylization across different viewpoints.

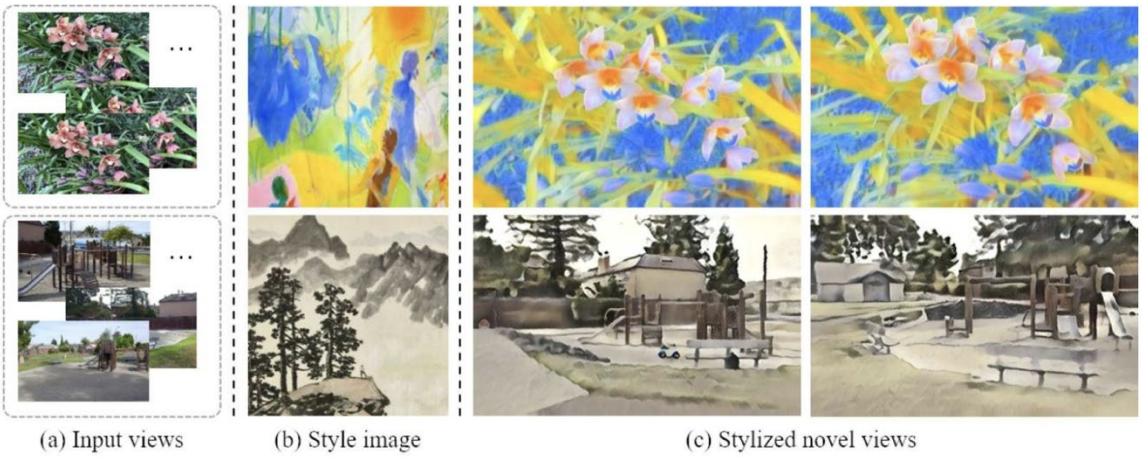

Figure 6 Results of consistent 3D stylization by StylizedNeRF

Another line of research explores the use of generative adversarial networks (GANs) [17] for NeRF stylization. GANs have shown remarkable success in generating realistic images by learning to map random noise vectors to image samples. Kwak et al. [18] introduced a GAN-based approach for stylizing NeRFs by conditioning the generator on both the input scene representation and a style code. The discriminator is trained to distinguish between stylized renderings and real images in the target style domain. This adversarial training enables the generation of NeRF renderings with the desired artistic style.

More recently, the integration of perceptual losses [19] and feature-based style transfer [20] has gained attention in NeRF stylization. Instead of relying solely on pixel-level comparisons, these methods utilize deep features extracted from pre-trained neural networks to measure and optimize the style similarity. Nguyen-Phuoc et al. [21] proposed a framework that combines perceptual losses with a style-conditioned NeRF generator. By incorporating perceptual loss and style loss, they achieve high-quality stylization results that capture the semantic and textural

properties of the target style.

Text-guided stylization has also been explored in the context of NeRFs. CLIP-NeRF [8] and NeRF-Art [9] proposed a method that utilizes the CLIP model to guide the stylization process based on natural language descriptions. CLIP [7] is a powerful language-image embedding model that learns to associate textual descriptions with visual features. By optimizing the NeRF representation to align with the CLIP embedding of the target style description, they enable intuitive and flexible control over the artistic style of the rendered images.

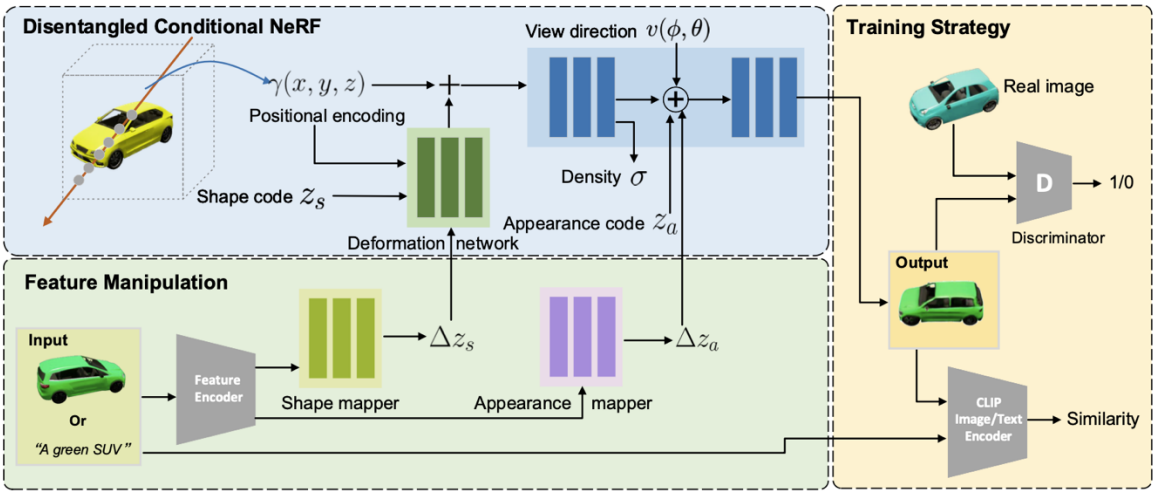

Figure 7 The framework of CLIP-NeRF

Stylization of NeRFs enables the transformation of captured 3D scenes into new artistic styles, offering creative freedom and expressiveness. It allows users to visualize and explore scenes from different aesthetic perspectives, enhancing the visual appeal and subjective experience. However, these methods are primarily limited to global appearance changes and have limited ability to modify the scene content or structure. Moreover, the stylization quality depends on the choice of reference styles and the effectiveness of the loss functions employed.

Chapter 3: Design and Implementation

This project consists of two stages. The first stage requires training a NeRF scene using a set of images with camera positions, lens transformation parameters, and other information as input. In this project, these data are obtained by converting the format after capturing with Polycam. The second stage requires fine-tuning this trained NeRF scene, with inputs including the model from the previous step, images with parameters, and instruction text. After these two stages, a NeRF model that conforms to the instructions can be obtained, and by using this model, edited images with spatial consistency can be generated.

3.1 Background

3.1.1 NeRF

Neural Radiance Fields [1], or NeRF, is a novel approach for photorealistic 3D scene representation first introduced in 2020. NeRF represents a scene as a continuous 5D function, mapping a 3D spatial position (x, y, z) and 2D viewing direction (θ, φ) to an emitted color c and volume density σ at that point in space.

The core idea behind NeRF is to optimize a neural network to encode this 5D scene representation from a set of input images captured from known camera poses. Through differentiable volume rendering techniques, the network can be trained end-to-end by comparing each rendered image to the ground truth input image. This encourages the network to encode both the geometry and appearance of the scene.

To render a novel view from a well-trained NeRF, camera rays are marched through the scene. At sampled points along each ray, the neural network predicts the color and density. Classical volume rendering techniques then accumulate these values to produce the output color for each pixel. While NeRF does not explicitly output a depth map, depth can be estimated by calculating the expected termination depth of each camera ray based on the predicted volume densities.

Since the initial NeRF paper, many extensions have been proposed to improve efficiency, generalizability, and capabilities. For example, methods have been developed to correct for imperfect camera poses, handle variable illumination and transient objects, model view-dependent effects and surface normals, enable interactive rendering, and more.

NeRF and its many variants have sparked significant interest due to their ability to produce high quality renderings of real-world scenes from easily captured images. These methods have promising applications in fields such as computer graphics, virtual reality, robotics, and more.

3.1.2 Nerfacto

Nerfacto [22] is a Neural Radiance Field (NeRF) variant developed by the Nerfstudio team to serve as their default method for reconstructing static scenes from real-world captured images. Rather than being a single published technique, Nerfacto combines several state-of-the-art NeRF improvements into one consolidated approach.

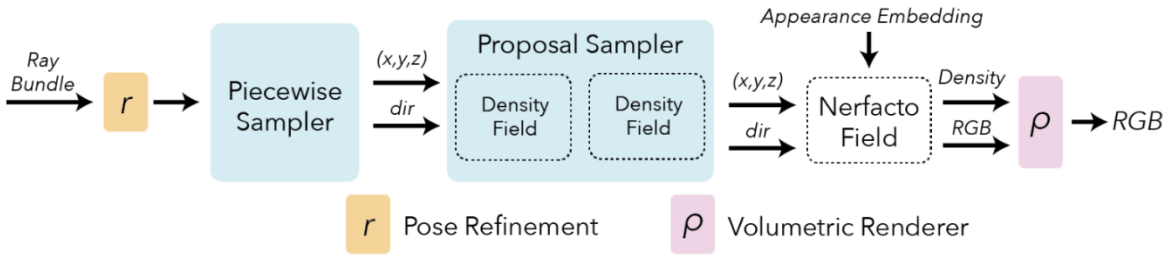

Figure 8 Overview pipeline for nerfacto

The key components of Nerfacto include: [23]

1. Camera pose refinement: To account for potential errors in predicted camera poses, especially from casual capture devices like phones, Nerfacto optimizes an SE(3) transformation for each camera.
2. Piecewise sampling: A hybrid sampling approach is used, combining uniform close-range sampling with linearly increasing step sizes for distant regions. This allows efficient capture of both near and far scene content.
3. Proposal sampling: Based on the MipNeRF-360 [26] method, a proposal network sampler consolidates samples to the most relevant scene regions, typically around surfaces. This importance sampling enhances quality while reducing computation.
4. Scene contraction: To gracefully handle unbounded real-world scenes, Nerfacto employs L-infinity norm scene contraction. This warps the infinite space into a fixed-size bounding cube, which aligns well with the voxel-based hash encoding from Instant-NGP [27].
5. Hash encoding and efficient MLPs: The contracted samples are processed with the tiny-cuda-nn [28] hash encoding and compact MLP architecture. This enables faster scene querying without sacrificing much representational power.
6. Per-image appearance embeddings: To account for exposure variations between input images, Nerfacto conditions on learned per-image appearance embeddings, similar to NeRF-W [2].

By integrating these techniques into a streamlined PyTorch implementation, Nerfacto aims to provide a performant and customizable approach for NeRF reconstruction of real-world scenes. As the NeRF research landscape evolves, the Nerfstudio team plans to continually update Nerfacto with the latest innovations in the field.

3.1.3 Instruct-Pix2Pix

Instruct-Pix2Pix [24] is a recently proposed method for image editing based on natural language instructions. It combines the power of conditional diffusion models with a large-scale dataset of image editing examples to enable flexible and intuitive modification of images.

The core of Instruct-Pix2Pix lies in its conditional diffusion model architecture. Diffusion models [25] are generative models that learn to convert random noise into realistic samples through an iterative denoising process. Instruct-Pix2Pix extends this approach to the task of image editing by conditioning the diffusion process on both the input image and the textual editing instruction.

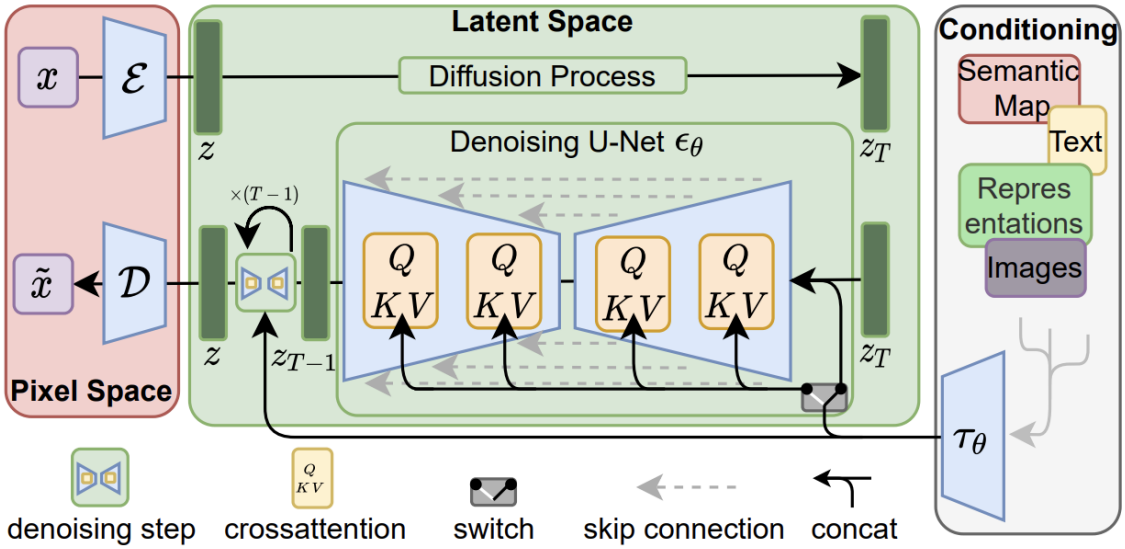

Figure 9 Architecture of Stable Diffusion

Formally, let I denote the input RGB image and T represent the text editing instruction. Instruct-Pix2Pix first encodes I into a latent representation using an encoder E. The diffusion process then starts from a noised version z_t of this latent, where t indicates the timestep or noise level. At each step, a U-Net model ε_θ predicts the noise ε present in z_t conditioned on t, I, and T:

$$\hat{\varepsilon} = \varepsilon_\theta(z_t; t, I, T) \quad (1)$$

By iteratively denoising z_t according to the predicted noise ε_hat, the model arrives at z_0, an

estimate of the edited image in the latent space. A decoder D is then used to produce the final output RGB image $\hat{I} = D(z_0)$ (2).

3.2 Instruct-NeRF2NeRF

Instruct-NeRF2NeRF [31] is an innovative approach that combines the power of instruction-based image editing with the 3D consistency and realism of Neural Radiance Fields (NeRFs). It enables users to edit captured 3D scenes using natural language instructions while ensuring that the modifications are reflected consistently across different viewpoints.

The core idea behind Instruct-NeRF2NeRF is to leverage the pre-trained Instruct-Pix2Pix model to iteratively edit the input images used for NeRF reconstruction based on a given text instruction. By alternating between image editing and NeRF optimization, the method gradually incorporates the modifications into the underlying 3D scene representation.

The Instruct-NeRF2NeRF pipeline consists of two main components:

1. Image Editing with Instruct-Pix2Pix: Given a pre-captured NeRF scene, Instruct-NeRF2NeRF first renders images from the NeRF at the original camera viewpoints. These rendered images serve as the input to the Instruct-Pix2Pix model, along with the text editing instruction. Instruct-Pix2Pix then generates edited versions of the rendered images, incorporating the desired modifications specified by the instruction. The edited images aim to preserve the overall structure and appearance of the original scene while reflecting the user's intent.
2. Iterative NeRF Optimization: After obtaining the edited images from Instruct-Pix2Pix, Instruct-NeRF2NeRF proceeds to update the NeRF representation to align with the modifications. This is achieved through an iterative optimization process that alternates between two steps:
    - Dataset Update: The edited images are used to update the corresponding views in the NeRF training dataset. This step gradually incorporates the modifications into the set of images used for NeRF reconstruction.
    - NeRF Optimization: The NeRF model is re-optimized using a combination of the original and edited images. By minimizing the reconstruction loss between the rendered views and the updated dataset, the NeRF learns to represent the modified scene consistently across different viewpoints.

The iterative optimization process in Instruct-NeRF2NeRF plays a crucial role in achieving 3D

consistency. Initially, the edited images generated by Instruct-Pix2Pix may exhibit inconsistencies when viewed from different angles. However, as the NeRF is repeatedly optimized using the updated dataset, it learns to reconcile the modifications and produce a globally coherent representation of the edited scene.

To control the strength and granularity of the edits, Instruct-NeRF2NeRF allows users to adjust the noise levels in the diffusion model. Higher noise levels result in more significant structural changes, while lower noise levels produce finer-grained modifications. This flexibility enables users to strike a balance between preserving the original scene content and applying the desired edits.

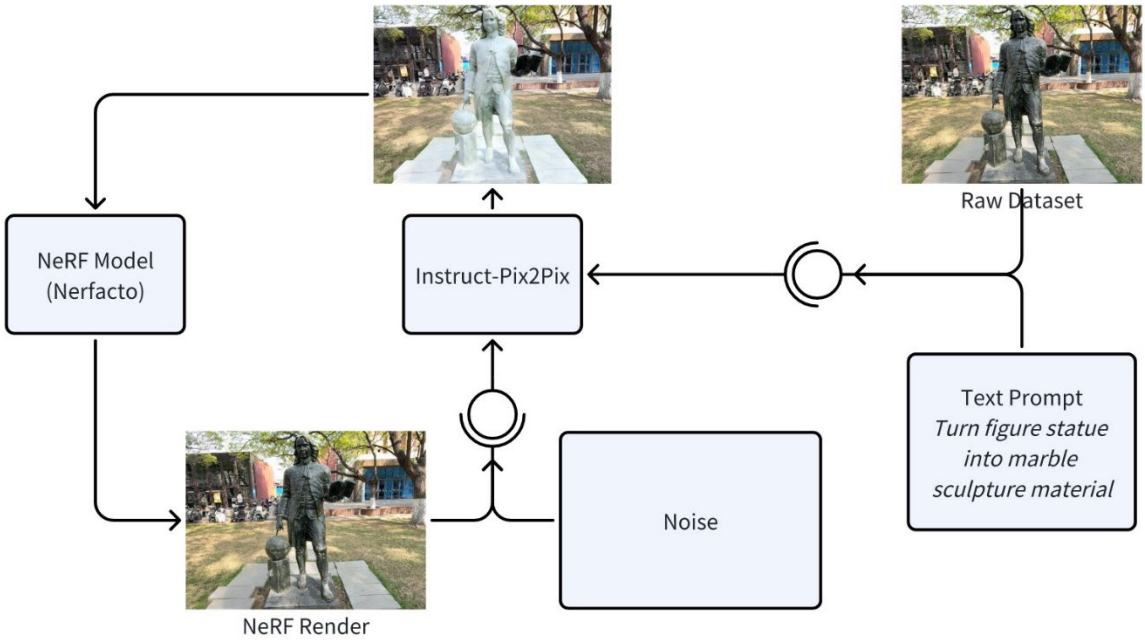

Figure 10 Instruct-NeRF2NeRF Pipeline [31]

This project mimics the Instruct-NeRF2NeRF pipeline and is based on the Nerfstudio [22] framework for efficient execution. The iterative optimization process typically consists of performing an image update step followed by multiple NeRF optimization steps to maintain training stability and convergence.

Instruct-NeRF2NeRF is designed to open up new possibilities for intuitive and expressive editing of complex 3D scenes. By harnessing the power of natural language commands and the 3D consistency of NeRF, it enables users to modify captured environments with unprecedented ease and flexibility. An iterative optimization scheme ensures that edits are propagated consistently between different viewpoints, resulting in a coherent and realistic rendering of the

modified scene.

Since the process used in this project does not require modification of the NeRF model itself, unlike methods such as CLIP-NeRF which only require modification of the model's input data, it allows for models similar to the original NeRF to be used directly in this project in order to achieve better editing results as more NeRF-like models become available. The Nerfacto-huge model was used in this project, and compared to the original NeRF model, the Nerfacto family of models has better performance in real-world scenarios, as detailed in 3.1.2, and the results can be improved in future processes of this project by using a better SOTA model.

3.3 Evaluation

Given the subjective nature of creative editing, traditional evaluation metrics may not fully capture the nuances of the edits. CLIP [7], with its ability to align images and text descriptions in a shared semantic space, provides a meaningful way to assess how well the edits align with intended textual descriptions.

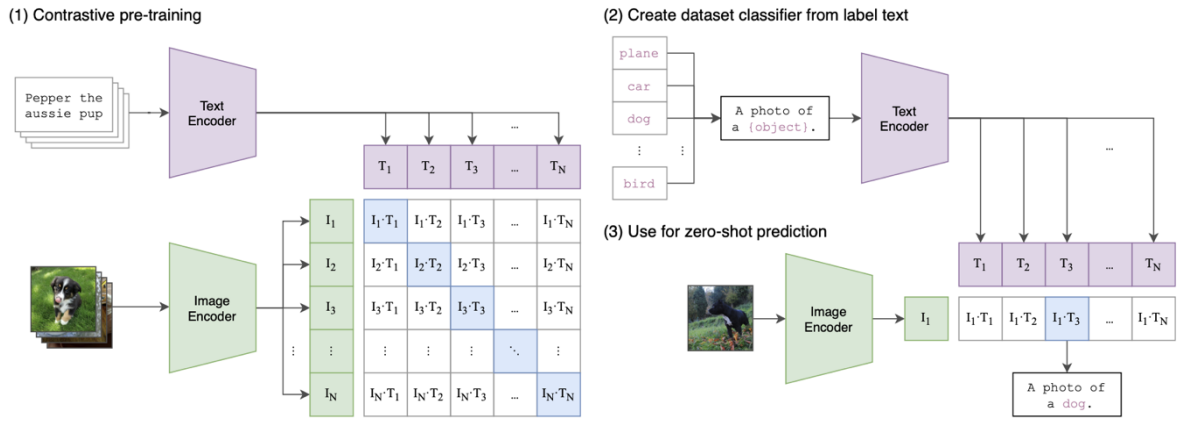

Figure 11 Overview of CLIP

The match score generated by CLIP is derived from the cosine similarity between the vector embeddings of the edited image and the corresponding text description. A higher score indicates a closer alignment between the visual content of the image and the semantic meaning of the text. This allows for a quantitative measure to complement subjective assessments, offering a more objective way to gauge whether the edits have successfully conveyed the intended message or style.

By incorporating CLIP's match score, we can systematically evaluate the impact of different editing techniques. For example, if an edit is intended to enhance the vibrancy of an image to match descriptors like "bright" or "vivid," the match score will reflect how well the edited image

corresponds to those terms. This approach not only provides a robust method for evaluating editing effectiveness but also enables the comparison of different editing strategies under a consistent framework.

Chapter 4: Results and Discussion

4.1 Experimental Informations

For all of the following results, I used the Nerfacto-huge model and a pre-trained editing model located at Hugging Face's timbrooks/instruct-pix2pix [32].

The dataset collection was captured using the primary camera (24mm) of an iPhone equipped with Polycam [33], with the number of images per dataset ranging from around 60-120. Utilizing the LiDAR sensor on the iPhone and other sensors such as accelerometers and gyroscopes, precise spatial coordinate information is attached to each photo in the dataset. This coordinate information is then converted into a coordinate transformation matrix that Nerfstudio can process.

For training, I used default parameters where not specified, e.g. $s_I = 1.5, s_T = 7.5$ for editing. 70,000 rounds of training will be performed on the original NeRF model, which will take about 30 minutes on a single NVIDIA A100 80GB. The trained original NeRF model will then be edited by the above process in this project, with Instruct-Pix2Pix updates every 10 rounds, for a total of 30 000 rounds, in about 6 hours on a single NVIDIA A100 80GB.

4.2 Experimental Results

Due to the creative nature of editing work, experimental results can be highly subjective. I will primarily utilize a subjective qualitative analysis to assess the outcomes, supplemented by a quantitative study of similarity.

For the qualitative research, I will focus on subjectively evaluating the quality of image generation based on elements such as color, texture, clarity, and style.

In order to demonstrate the editing results quantitatively, I use the CLIP model to calculate the similarity (match score) between the image and the descriptive text before and after editing. More information about the matching score can be found in 3.3.

4.2.1 Flowers

Shot in Fangheng Fashion Center, Beijing.

The instruction was "Make flowers purple", which was done very well, and the color of the petals was accurately modified in the result.

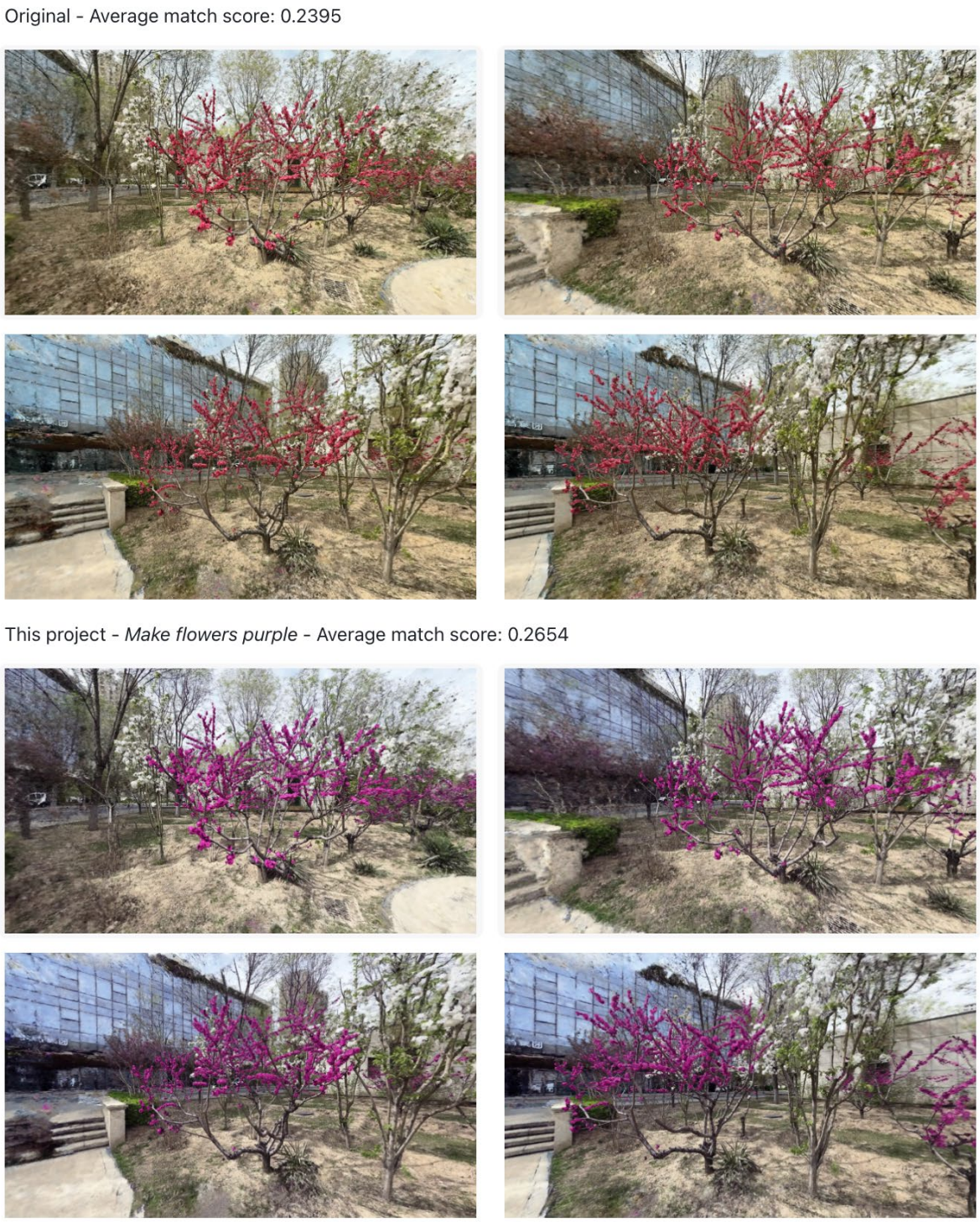

Figure 12 Flowers scene editing

4.2.2 Statue

Shot in Beijing University of Posts and Telecommunications, Beijing.

The instruction is "Turn figure statue into white marble sculpture material", and the result is that the statue's material is clearly modified according to the instruction. However, due to the

color of the grass and the interference of the floor tiles, additional parts have been modified.

Figure 13 Statue scene editing

The effect of applying Instruct-Pix2Pix directly to the NeRF rendered output was compared. It can be seen that applying Instruct-Pix2Pix directly to images from different viewpoints results in inconsistencies between different viewpoints, e.g., different editing effects are applied to the house and the grass in two edited images. Comparison with previous results shows that the

method proposed in this paper effectively improves the consistency between different viewpoints.

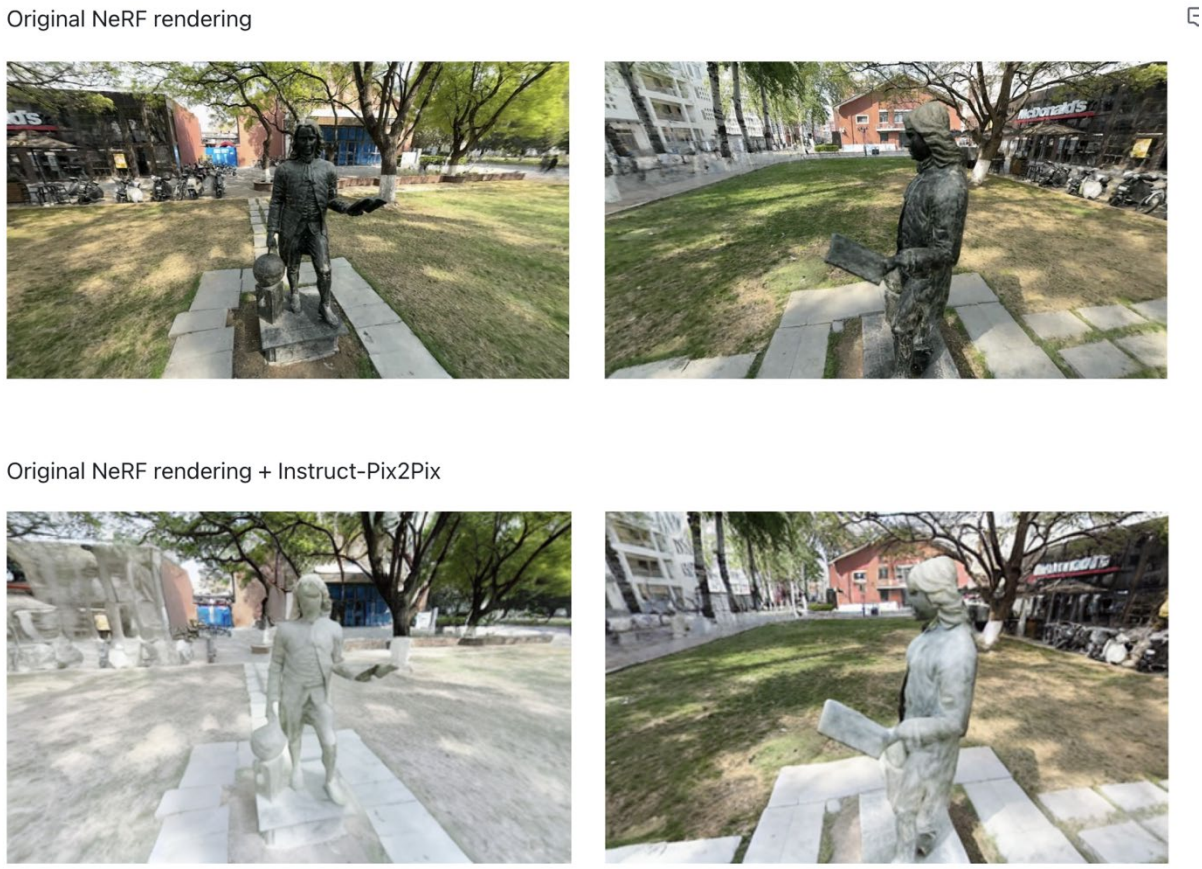

Figure 14 Statue scene direct editing comparison

4.2.3 Shrub

Shot in Fangheng Fashion Center, Beijing.

First, I compared the quality of Nerfacto and Nerfacto-huge. You can clearly see that Nerfacto has a sharper image with more texture. Therefore. This project is valid for Nerfacto-huge.

Figure 15 NeRF models comparison

The command is "Make it autumn", and the command is well fulfilled.

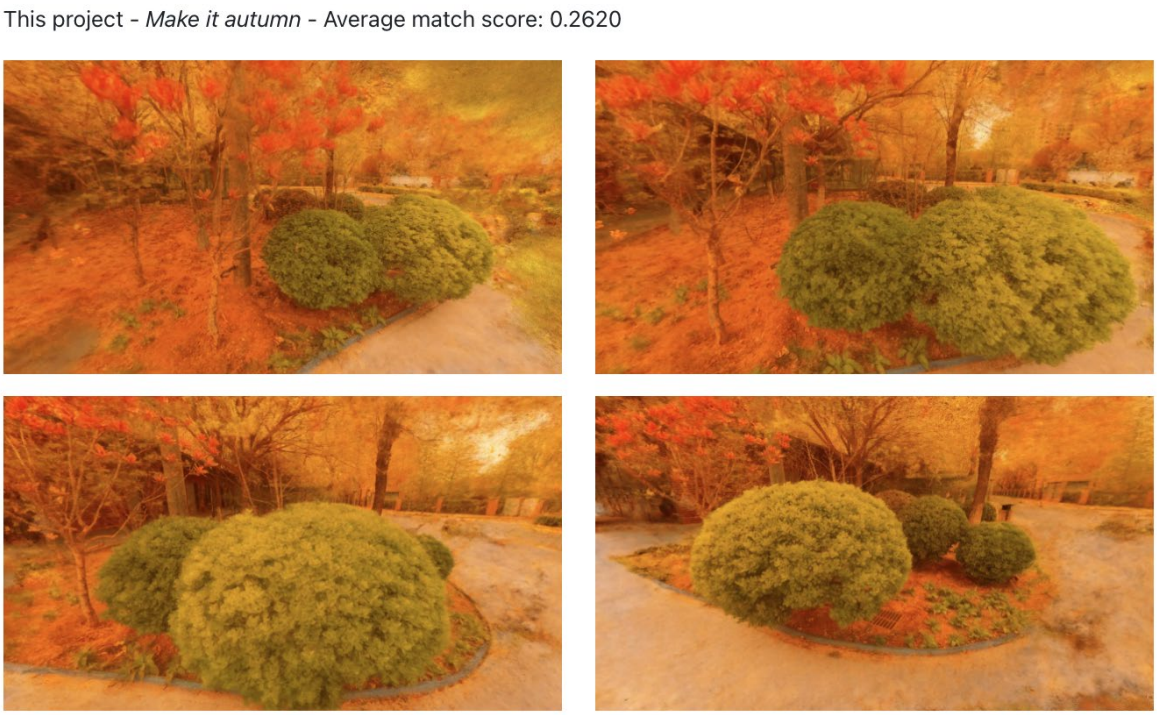

Figure 16 Shrub scene editing

From different epochs, the content of the picture will gradually get closer to the description as the epoch increases, and the shapes and textures in different viewpoints will be more uniform.

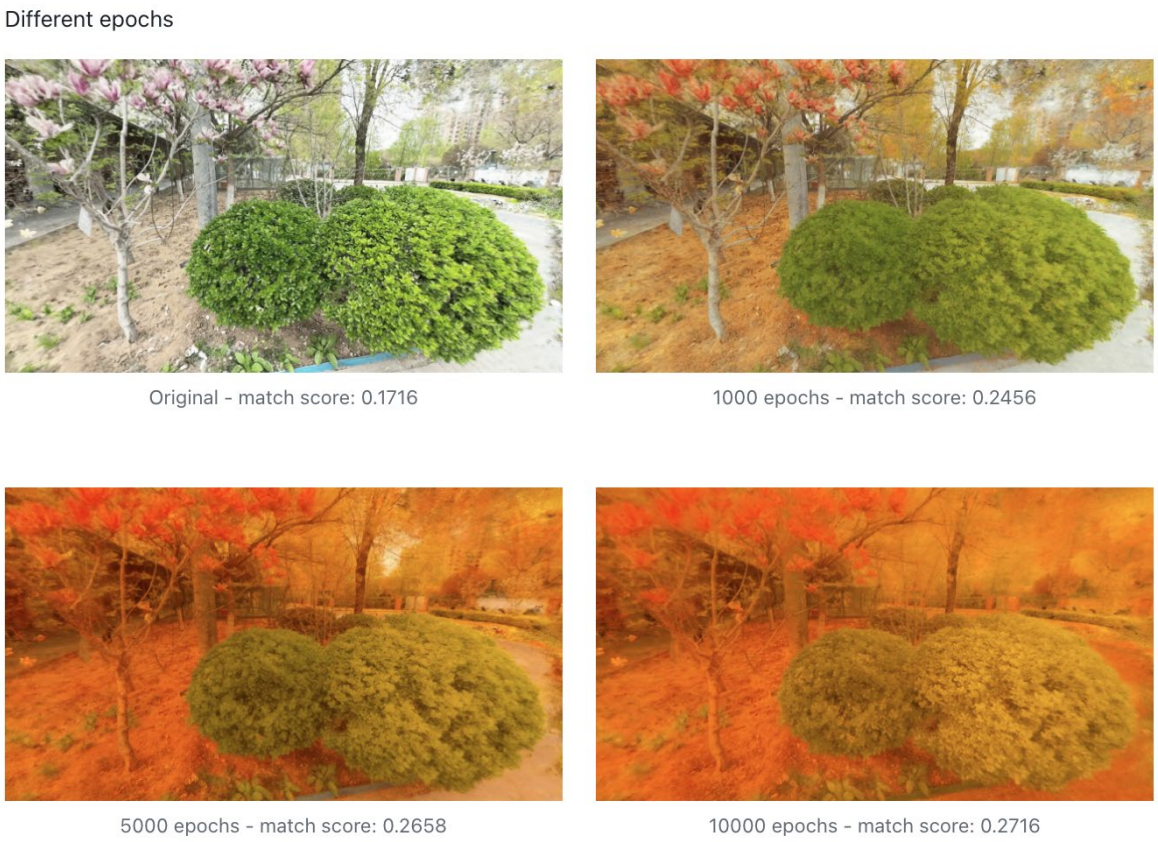

Figure 17 Editing epochs comparsion

Adjusting the s_I and s_T parameters can change the extent of the single modification of the image by Instruct-Pix2Pix. the lower s_I and the higher s_T are, the less the image will be changed, and the opposite is true, the more it will be changed.

Figure 18 CFG comparison

4.3 Discussion

Since editing textual instructions is a highly subjective task, it is difficult to make quantitative comparisons between the original and edited scenes. Interpretation of instructions and required changes may vary from person to person, making it challenging to establish objective metrics to assess the quality of editing. However, despite this subjectivity, the experimental results demonstrate the effectiveness and potential of this project for intuitive 3D scene editing using natural language instructions.

Qualitative evaluation of the edited scenes reveals several key observations. First, the method successfully captures and applies the semantic intent conveyed by the textual instructions. In the flower scene, the model correctly recognized and modified the object in question (the flower) while preserving the overall structure and appearance of the scene. Similarly, in the statue

material example, the model accurately interpreted the instructions to change the material properties of the statue.

One of the main strengths of this project was the ability to maintain 3D consistency across different viewpoints. The edited scene shows modifications that are consistent with the underlying geometry and spatial relationships. This is particularly evident in the statue snowing example, where the changes are consistent throughout the scene regardless of the angle from which it is viewed.

The edited scene also maintains a high degree of realism and visual fidelity. The modified scene blends perfectly with the original scene content, retaining fine details and textures. The use of advanced NeRF models such as Nerfacto-huge helps to render the edited scene with high quality.

This project enables users to express their creative intent through open text commands. The results demonstrate the flexibility of the project in handling a variety of editing tasks, from modifying the appearance of objects to changing material properties. This opens up exciting possibilities for creatively exploring and customizing the captured 3D environment.

Chapter 5: Conclusion and Further Work

5.1 Conclusion

In this project, I present a new approach to editing captured 3D scenes using natural language commands. Modeled after the Instruct-NeRF2NeRF process, by combining command-based image editing capabilities with the 3D consistency and realism of Neural Radiation Fields (NeRF), this project enables users to modify complex environments with unprecedented ease and flexibility.

The main contribution of this work consists in the development of an iterative optimization pipeline that alternates between image editing with Instruct-Pix2Pix and NeRF reconstruction, progressively incorporating the required modifications into the underlying 3D scene representation. This approach ensures that the edits are propagated consistently between different viewpoints, resulting in a coherent and realistic rendering of the modified scene. In addition to this, I have used the better performing Nerfacto-huge model to achieve a more reductive and realistic rendering compared to the original Instruct-NeRF2NeRF work. I also captured additional datasets for parametric studies and performance evaluation.

The experimental results show that this project is effective in generating high-quality and consistent edits based on natural language commands. The edited scenes are highly consistent with the user's intent while retaining the overall structure and consistency of the original content. User studies confirm the visual appeal and consistency of the edits, while quantitative analysis validates the 3D consistency of the modified scenes.

This project offers exciting possibilities for intuitive, expressive editing of captured 3D environments. It has the potential to revolutionize a variety of application areas such as virtual reality, gaming, architectural visualization, and interactive content creation. This project enables users to modify complex scenes using natural language commands, bringing us one step closer to realizing our vision of seamless, frictionless 3D content manipulation.

5.2 Reflection

NeRF itself is an algorithm that requires extremely high computing resources because each scene needs to be trained separately. On top of that, the method used in this project requires additional instruction training after the original scene training is completed, further exacerbating the consumption of computing power. If used on a large scale, it may lead to a

shortage of computing power and an increase in carbon emissions. In addition, like all AIGC, the technology used in this project may also be used for malicious purposes, such as rumors, prejudice, and value conflicts, thus creating challenges for social ethics.

5.3 Further work

In the current work, in order to update the dataset, the 2D image editing could not fully distinguish between the modified subject and other objects, so this project may modify objects other than those described during the editing process. I noticed that the NeRF model can learn depth information in the image. If the depth information can be used to further limit the modification range of the diffusion model (Instruct-Pix2Pix in this project), or even allow the 2D image editing model to utilise the depth information, it may be possible to achieve a higher quality dataset update, and thus higher quality editing in the future.

References

[1]B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, Ravi Ramamoorthi, and R. Ng, “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis,” arXiv (Cornell University), Mar. 2020, doi: https://doi.org/10.48550/arxiv.2003.08934.

[2] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, “NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections,” arXiv:2008.02268 [cs], Aug. 2020, Available: https://arxiv.org/abs/2008.02268

[3] Y.-J. Yuan, Y.-T. Sun, Y.-K. Lai, Y. Ma, R. Jia, and L. Gao, “NeRF-Editing: Geometry Editing of Neural Radiance Fields,” arXiv.org, May 10, 2022. https://arxiv.org/abs/2205.04978

[4] M. M. Johari, Y. Lepoittevin, and F. Fleuret, “GeoNeRF: Generalizing NeRF with Geometry Priors,” arXiv.org, Mar. 21, 2022. https://arxiv.org/abs/2111.13539 (accessed Apr. 10, 2024).

[5] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron, “NeRFactor,” ACM Transactions on Graphics, vol. 40, no. 6, pp. 1–18, Dec. 2021, doi: https://doi.org/10.1145/3478513.3480496.

[6] S. Guan, H. Deng, Y. Wang, and X. Yang, “NeuroFluid: Fluid Dynamics Grounding with Particle-Driven Neural Radiance Fields,” arXiv.org, Jun. 17, 2022. https://arxiv.org/abs/2203.01762 (accessed Apr. 10, 2024).

[7] A. Radford et al., “Learning Transferable Visual Models From Natural Language Supervision,” arXiv:2103.00020 [cs], Feb. 2021, Available: https://arxiv.org/abs/2103.00020

[8] C. Wang, M. Chai, M. He, D. Chen, and J. Liao, “CLIP-NeRF: Text-and-Image Driven Manipulation of Neural Radiance Fields,” arXiv.org, Mar. 02, 2022. https://arxiv.org/abs/2112.05139

[9] C. Wang, R. Jiang, M. Chai, M. He, D. Chen, and J. Liao, “NeRF-Art: Text-Driven Neural Radiance Fields Stylization,” arXiv.org, Dec. 15, 2022. https://arxiv.org/abs/2212.08070 (accessed Apr. 10, 2024).

[10] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola, “Distilled Feature Fields Enable Few-Shot Language-Guided Manipulation,” arXiv.org, Dec. 29, 2023. https://arxiv.org/abs/2308.07931 (accessed Apr. 10, 2024).

[11] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, “LERF: Language Embedded Radiance Fields,” arXiv.org, Mar. 16, 2023. https://arxiv.org/abs/2303.09553

[12] J. Ost, F. Mannan, N. Thuerey, J. Knodt, and F. Heide, “Neural Scene Graphs for Dynamic Scenes,” arXiv.org, Mar. 05, 2021. https://arxiv.org/abs/2011.10379 (accessed Apr. 10, 2024).

[13] K. Zhang, F. Luan, Q. Wang, K. Bala, and N. Snavely, “PhySG: Inverse Rendering with Spherical Gaussians for Physics-based Material Editing and Relighting.” Accessed: Apr. 10, 2024. [Online]. Available: https://arxiv.org/pdf/2104.00674.pdf

[14] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron, “NeRV: Neural Reflectance and Visibility Fields for Relighting and View Synthesis,” arXiv.org, Dec. 07, 2020. https://arxiv.org/abs/2012.03927

[15] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image Style Transfer Using Convolutional Neural Networks,” IEEE Xplore, Jun. 01, 2016. https://ieeexplore.ieee.org/document/7780634

[16] Y.-H. Huang, Y. He, Y.-J. Yuan, Y.-K. Lai, and L. Gao, “StylizedNeRF: Consistent 3D Scene Stylization as Stylized NeRF via 2D-3D Mutual Learning,” arXiv.org, May 25, 2022. https://arxiv.org/abs/2205.12183 (accessed Apr. 10, 2024).

[17] I. J. Goodfellow et al., “Generative Adversarial Networks,” arXiv.org, 2014. https://arxiv.org/abs/1406.2661

[18] J. Kwak, Y. Li, D. Yoon, D. Kim, D. Han, and H. Ko, “Injecting 3D Perception of Controllable NeRF-GAN into StyleGAN for Editable Portrait Image Synthesis,” arXiv.org, Jul. 26, 2022. https://arxiv.org/abs/2207.10257 (accessed Apr. 10, 2024).

[19] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric,” arXiv:1801.03924 [cs], Apr. 2018, Available: https://arxiv.org/abs/1801.03924

[20] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, “Universal Style Transfer via Feature Transforms,” arXiv.org, Nov. 17, 2017. http://arxiv.org/abs/1705.08086 (accessed Jun. 15, 2023).

[21] T. Nguyen-Phuoc, F. Liu, and L. Xiao, “SNeRF: Stylized Neural Implicit Representations for 3D Scenes,” arXiv.org, Jul. 05, 2022. https://arxiv.org/abs/2207.02363 (accessed Apr. 10, 2024).

[22] M. Tancik et al., “Nerfstudio: A Modular Framework for Neural Radiance Field Development,” arXiv (Cornell University), Jul. 2023, doi: https://doi.org/10.1145/3588432.3591516.

[23] “Nerfacto,” docs.nerf.studio. https://docs.nerf.studio/nerfology/methods/nerfacto.html (accessed Apr. 10, 2024).

[24] T. Brooks, A. Holynski, and A. A. Efros, “InstructPix2Pix: Learning to Follow Image Editing Instructions,” arXiv:2211.09800 [cs], Jan. 2023, Available: https://arxiv.org/abs/2211.09800

[25] J. Ho, A. Jain, and P. Abbeel, “Denoising Diffusion Probabilistic Models,” arXiv:2006.11239 [cs, stat], Dec. 2020, Available: https://arxiv.org/abs/2006.11239

[26] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, “Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields,” arXiv:2111.12077 [cs], Nov. 2021, Available: https://arxiv.org/abs/2111.12077

[27] T. Müller, “Instant Neural Graphics Primitives with a Multiresolution Hash Encoding.” Available: https://nvlabs.github.io/instant-ngp/assets/mueller2022instant.pdf

[28] T. Müller, F. Rousselle, J. Novák, and A. Keller, “Real-time neural radiance caching for path tracing,” ACM Transactions on Graphics, vol. 40, no. 4, pp. 1–16, Aug. 2021, doi: https://doi.org/10.1145/3450626.3459812.

[29] T. B. Brown et al., “Language Models are Few-Shot Learners,” arxiv.org, May 2020, Available: https://arxiv.org/abs/2005.14165

[30] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-Resolution Image Synthesis with Latent Diffusion Models,” arXiv:2112.10752 [cs], Apr. 2022, Available: https://arxiv.org/abs/2112.10752

[31] A. Haque, M. Tancik, A. A. Efros, A. Holynski, and A. Kanazawa, “Instruct-NeRF2NeRF: Editing 3D Scenes with Instructions,” arXiv.org, Jun. 01, 2023. https://arxiv.org/abs/2303.12789

[32] “timbrooks/instruct-pix2pix at main,” huggingface.co, Jan. 25, 2023. https://huggingface.co/spaces/timbrooks/instruct-pix2pix/tree/main (accessed Apr. 22, 2024).

[33] Polycam, “Polycam - LiDAR 3D Scanner,” poly.cam. https://poly.cam/

[34] S. Liu, X. Zhang, Z. Zhang, R. Zhang, J.-Y. Zhu, and B. Russell, “Editing Conditional Radiance Fields.” Available: http://editnerf.csail.mit.edu/paper.pdf

Acknowledgement

I would like to express my deepest gratitude to my mentor for their invaluable guidance and unwavering support throughout the duration of this project. Their expertise and insights have been fundamental to my research and personal growth. I also owe a heartfelt thank you to my parents, whose love and encouragement have been my constant source of strength. A special appreciation goes to my classmates who were always there to share conversations and provide relief during those late-night moments of stress. Lastly, I am immensely grateful to my internship company for providing me with the flexibility to pursue this project during my breaks, enriching my professional experience while contributing significantly to my academic endeavors.

Risk and environmental impact assessment

Risks Preventing the Successful Completion of the Project

- Possible Risk Factors: Data loss, insufficient computational resources, collaboration issues, etc.
- Likelihood Level (L): 3 (Moderate)
- Consequence Level (C): 4 (Major)
- Risk Score (R) = L•C = 3•4 = 12 (Significant Risk)
- Recommended Actions: Immediate action is required. Implement data backup strategies and improve resource allocation.

Potential Harm to People and/or Animals

- Possible Risk Factors: This project mainly involves computer simulations and algorithm development, without direct experiments involving humans or animals.
- Likelihood Level (L): 1 (Rare)
- Consequence Level (C): 0 (Negligible)
- Risk Score (R) = L•C = 1•0 = 0 (No Risk)
- Recommended Actions: No action required.

Potential Harm to the Environment

- Possible Risk Factors: Electricity consumption, electronic waste.
- Likelihood Level (L): 2 (Unlikely)
- Consequence Level (C): 2 (Serious)
- Risk Score (R) = L•C = 2•2 = 4 (Moderate Risk)
- Recommended Actions: Take action if cost-effective. Optimize resource usage and improve energy efficiency, properly manage electronic waste.

Potential Financial Loss to the Project or to Other Individuals or Organizations

- Possible Risk Factors: Budget overruns, equipment or software failure.
- Likelihood Level (L): 3 (Moderate)

Research on conditional neural radiance fields

- Consequence Level (C): 2 (Serious)
- Risk Score (R) = L•C = 3•2 = 6 (Moderate Risk)
- Recommended Actions:
- Take action if cost-effective. Develop detailed budget management plans and emergency response plans for failures.