

Instruct-NeRF2NeRF: Editing 3D Scenes with Instructions

ICCV 2023 (Oral)

[\[paper\]](#) [\[code\]](#) [\[project\]](#)

Ayaan Haque, Matthew Tancik, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa
(UC Berkeley)

2023.12.29

Mijin Koo

Contents

- **Summary of Instruct-NeRF2NeRF**
- **NeRF**
- **InstructPix2Pix**
- **Instruct-NeRF2NeRF**
 - **Method**
 - **Experiments**

Summary

■ Instruct-NeRF2NeRF

- Propose a method for **editing NeRF scenes with text instructions**.
- Given a NeRF of a scene and the collection of images to reconstruct 3D scene,
- Use an image conditioned diffusion model (InstructPix2Pix) to iteratively edit the input images while optimizing 3D scenes.

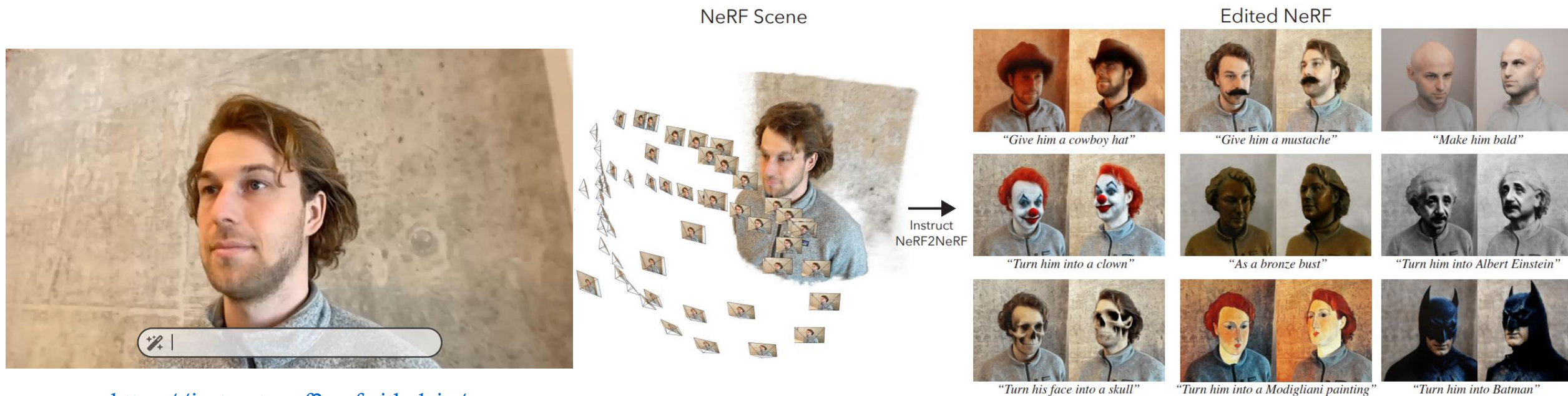


Figure 1: **Editing 3D scenes with Instructions.** We propose Instruct-NeRF2NeRF, a method for consistent 3D editing of a NeRF scene using text-based instructions. Our method can accomplish a diverse collection of local and global scene edits.

Summary

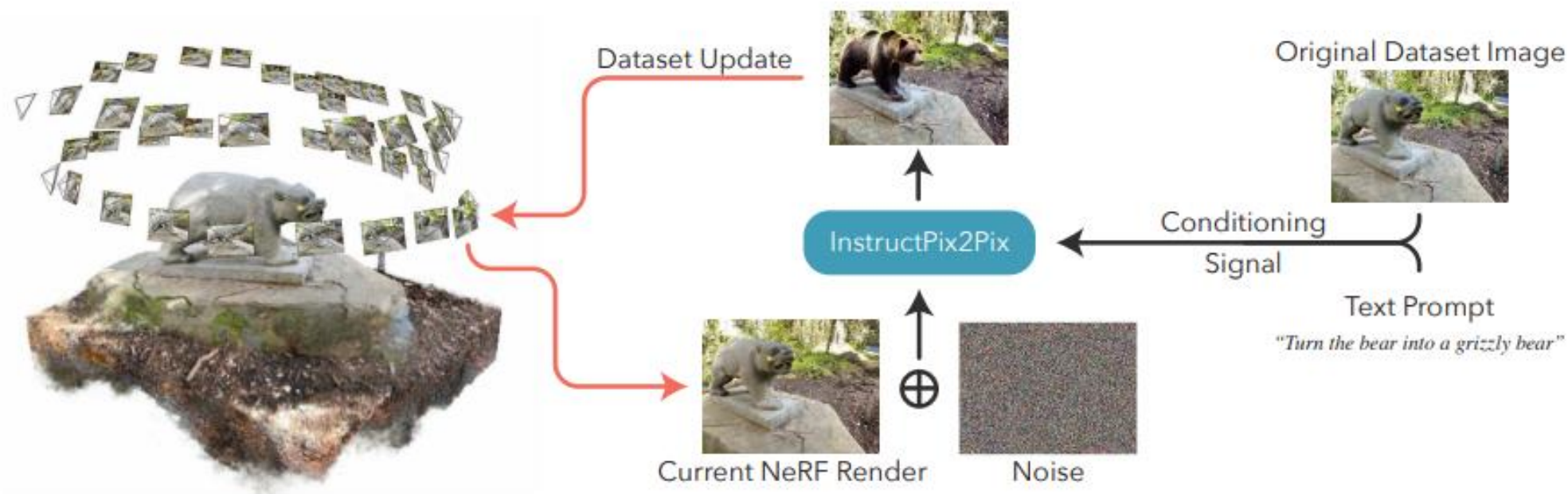


Figure 2: **Overview:** Our method gradually updates a reconstructed NeRF scene by iteratively updating the dataset images while training the NeRF: (1) an image is rendered from the scene at a training viewpoint, (2) it is edited by InstructPix2Pix given a global text instruction, (3) the training dataset image is replaced with the edited image, and (4) the NeRF continues training as usual.

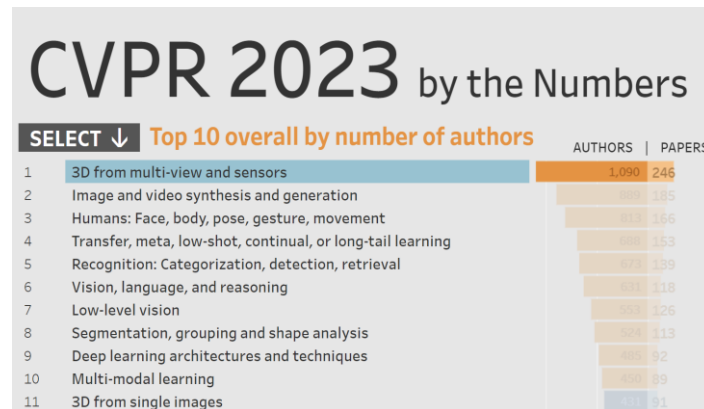
NeRF

■ Emergence of Neural 3D Reconstruction

- Efficient neural 3D reconstruction techniques **have simplified capturing realistic digital representations of real-world 3D scenes**. The process involves capturing images from various viewpoints, reconstructing camera parameters, and optimizing a Neural Radiance Field.

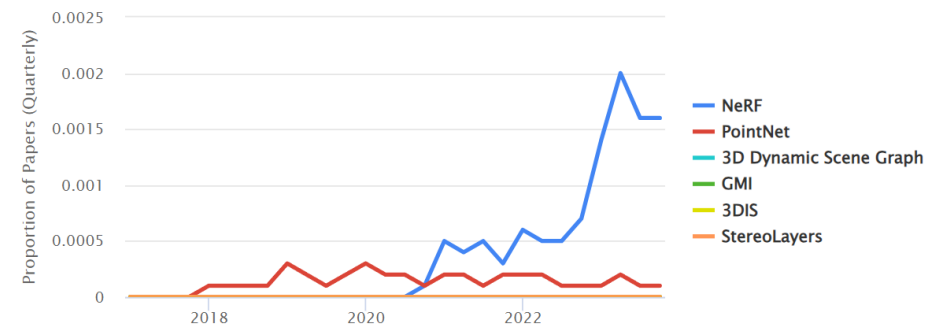


“The First NeRF in a TV commercial, January 18, 2023”
@LumaLabsAI for @McDonalds



- Ego-Body Pose Estimation via Ego-Head Pose Estimation
Jiaman Li · Karen Liu · Jiajun Wu
- 3D Registration with Maximal Cliques
Xiyu Zhang · Jiaqi Yang · Shikun Zhang · Yanning Zhang
- OmniObject3D: Large Vocabulary 3D Object Dataset for Realistic Perception, Reconstruction and Generation
Tong Wu · Jiarui Zhang · Xiao Fu · Yuxin WANG · Jiawei Ren · Liang Pan · Wenyan Wu · Lei Yang · Jiaqi Wang · Chen Qian · Dahua Lin · Ziwei Liu
- **MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures**
Zhiqin Chen · Thomas Funkhouser · Peter Hedman · Andrea Tagliasacchi
- **DynBaR: Neural Dynamic Image-Based Rendering**
Zhengqi Li · Qianqian Wang · Forrester Cole · Richard Tucker · Noah Snavely
- Planning-oriented Autonomous Driving
Yihan Hu · Jiazhi Yang · Li Chen · Keyu Li · Chonghao Sima · Xizhou Zhu · Siqi Chai · Senyao Du · Tianwei Lin · Wenhai Wang · Lewei Lu · Xiaosong Jia · Qiang Liu · Jifeng Dai · Yu Qiao · Hongyang Li

CVPR 2023 Subject Areas by TeamSize

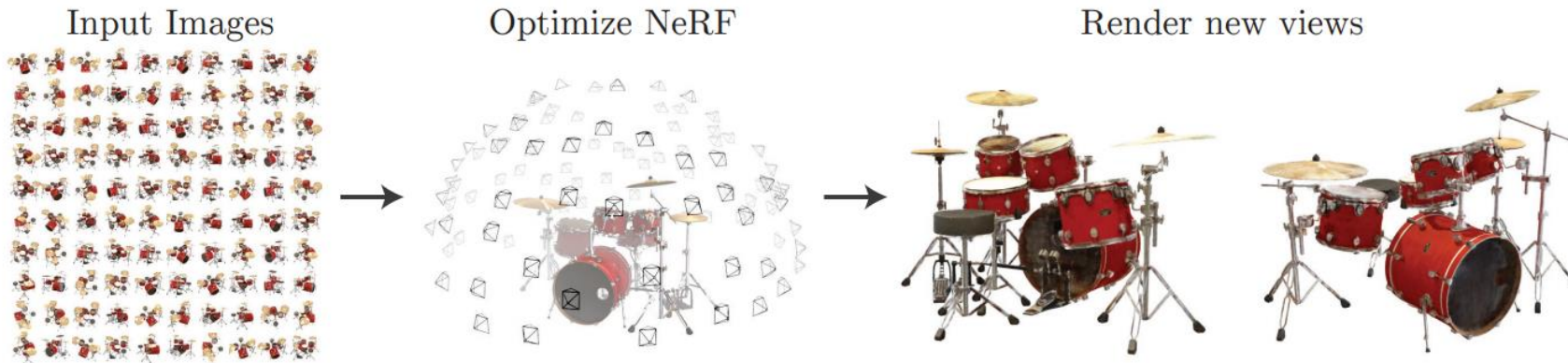


<https://paperswithcode.com/method/nerf>

NeRF

■ Neural Radiance Field (NeRF)

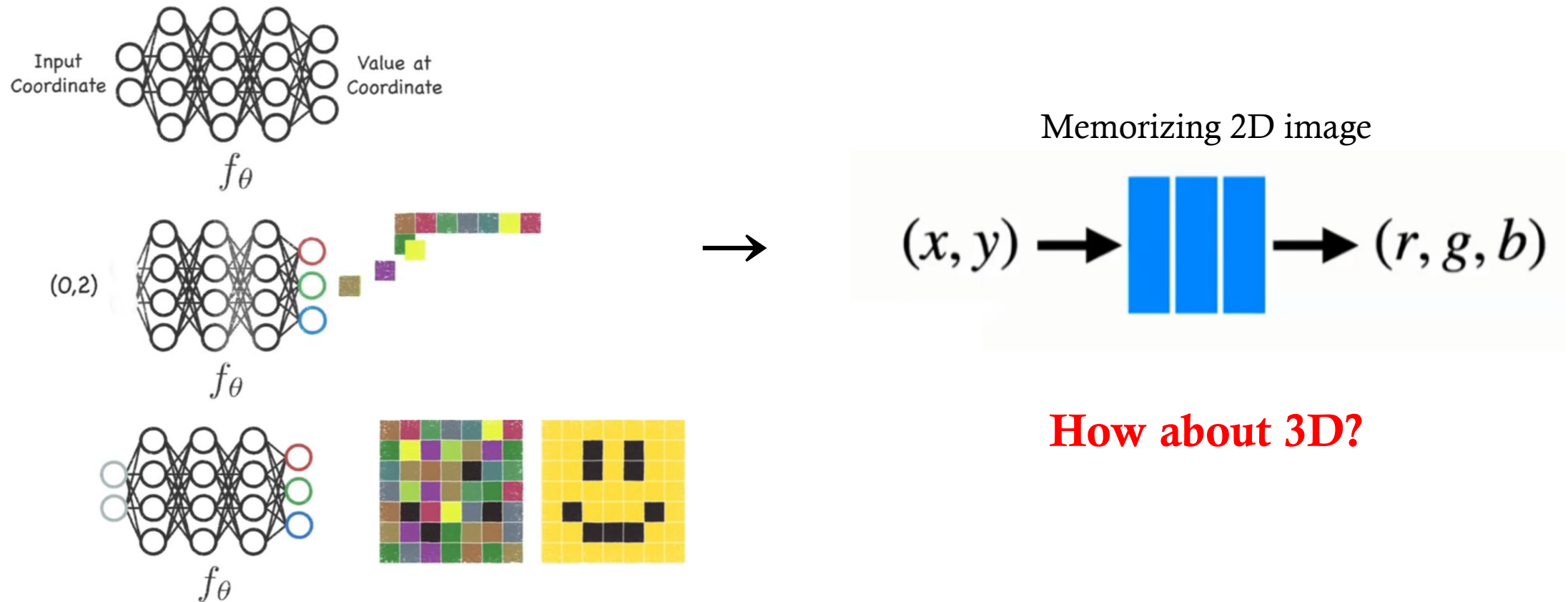
- Novel view synthesis, Image rendering, Scene representation
- Input images rendered from discretized viewpoint
→ Rendering new views which is not included in training dataset



NeRF

■ Scene representation

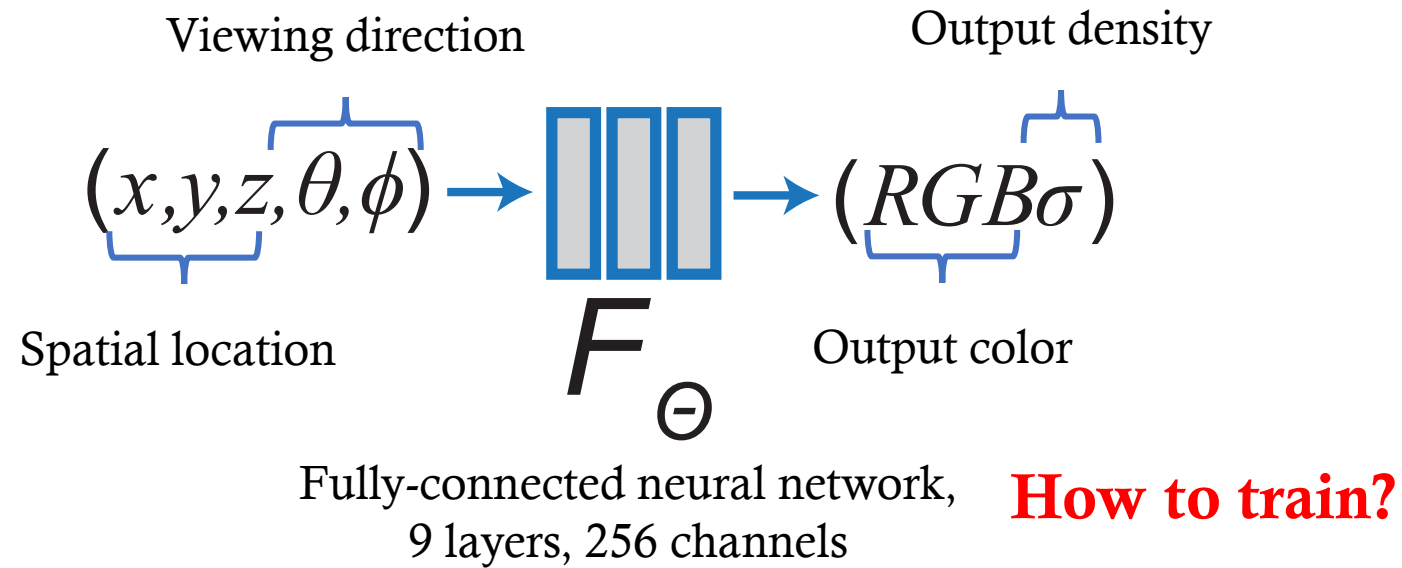
- Continuous functions parameterized by MLP – **Implicit Representation**
- **Coordinate-based Representation:** Takes low dimensional coordinates as input, and trained to output an appropriate representation(values; color, density) of each input



NeRF

■ Scene representation

- It is not enough to represent 3D scene with only spatial location (x y z)



- N of 2D images \rightarrow Synthesizing random view image
- $F_{\theta} : (x, d) \rightarrow (c, \sigma)$ with deep neural network
 - c : with respect to location x and viewing direction d
 - σ : with respect to location x
 - F_{θ} : Fully-connected neural network

NeRF

■ Volume Rendering

- Rendering? 물체의 표면 성질과 빛의 상호 작용을 2D 평면의 RGB 픽셀 값을 결정하는 것
- Volume Ray Casting
 - Computes 2D images from 3D volumetric data sets (3D scalar fields)

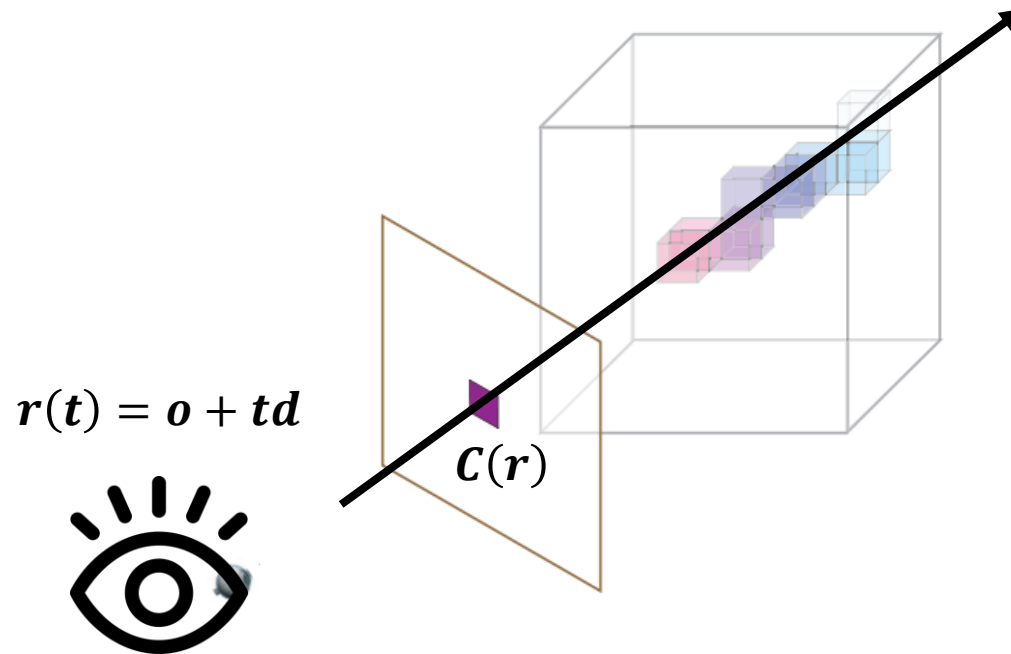
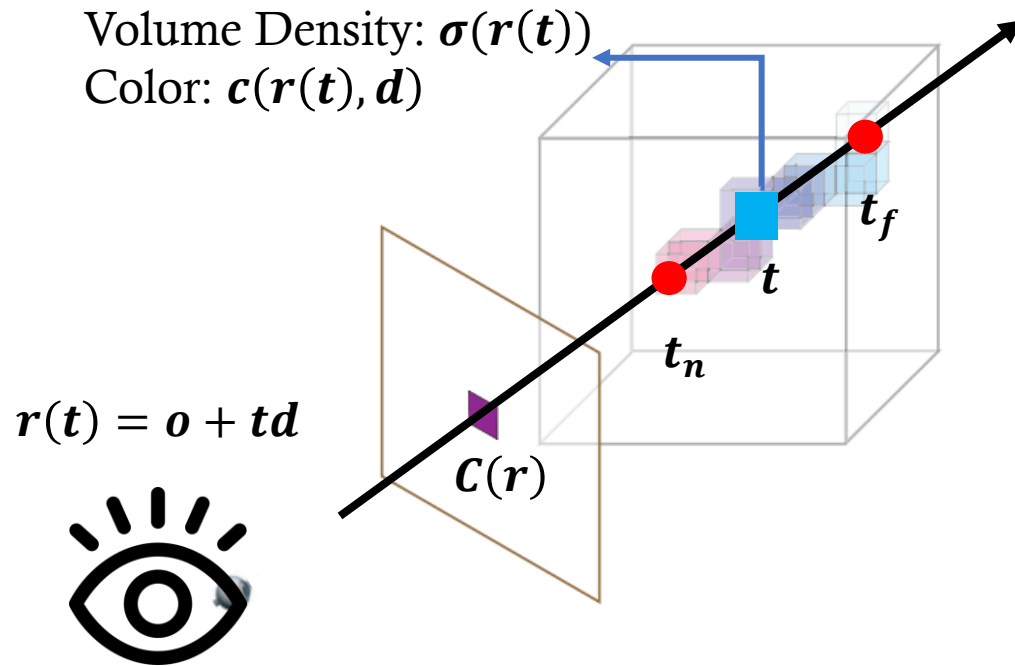


Figure 2: The ray casting integral sums the color and opacity properties of each data voxel that intersects the ray.

NeRF

■ Volume Rendering



$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t) \sigma(\mathbf{r}(t)) \mathbf{c}(\mathbf{r}(t), \mathbf{d}) dt ,$$

$$\text{where } T(t) = \exp\left(-\int_{t_n}^t \sigma(\mathbf{r}(s)) ds\right)$$

Figure 2: The ray casting integral sums the color and opacity properties of each data voxel that intersects the ray.

NeRF

■ Volume Rendering

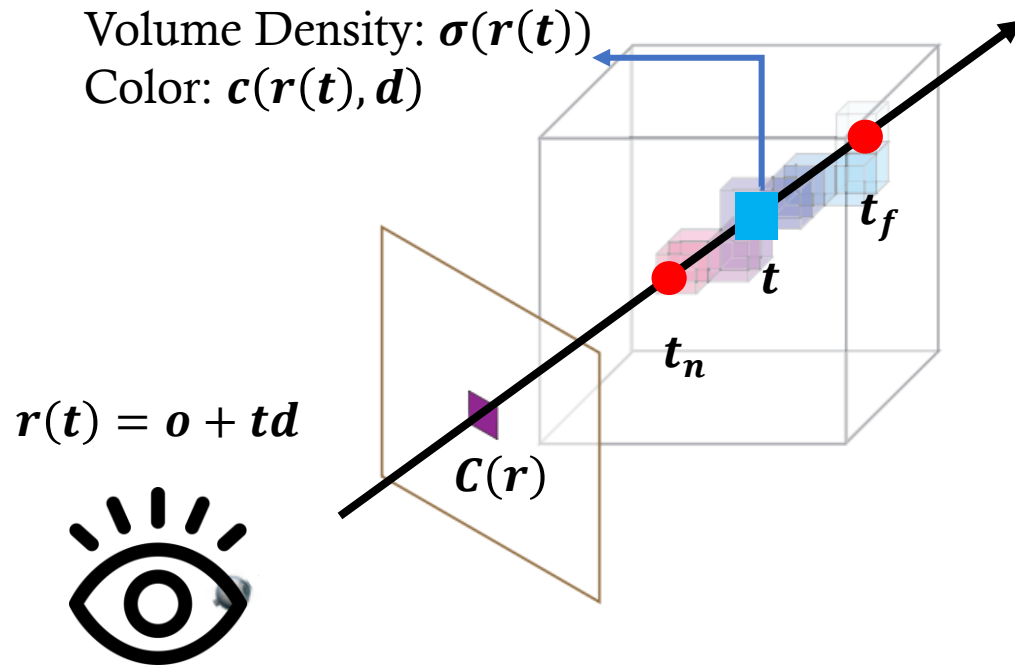


Figure 2: The ray casting integral sums the color and opacity properties of each data voxel that intersects the ray.

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t) \sigma(\mathbf{r}(t)) \mathbf{c}(\mathbf{r}(t), \mathbf{d}) dt ,$$

$$\text{where } T(t) = \exp\left(-\int_{t_n}^t \sigma(\mathbf{r}(s)) ds\right)$$

$$t_i \sim \mathcal{U}\left[t_n + \frac{i-1}{N}(t_f - t_n), t_n + \frac{i}{N}(t_f - t_n)\right]$$

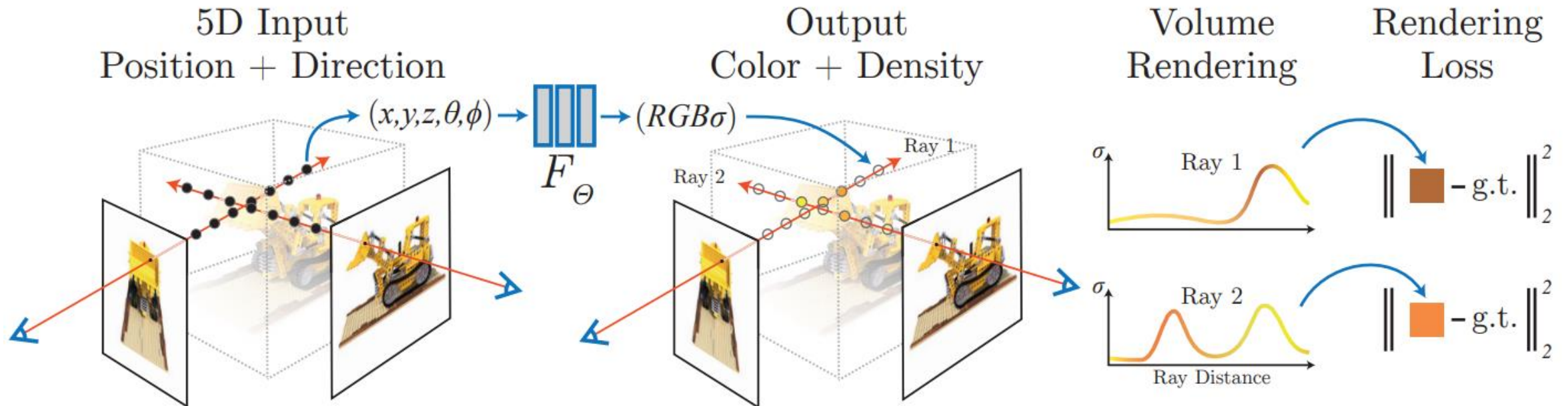
$$\hat{C}(\mathbf{r}) = \sum_{i=1}^N T_i (1 - \exp(-\sigma_i \delta_i)) \mathbf{c}_i, \text{ where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right)$$

- $T(t)$: Accumulated transmittance
- $\sigma(\mathbf{r}(t))$: Volume density or Occupancy at a 3D point
- $\mathbf{c}(\mathbf{r}(t), \mathbf{d})$: Color at a 3D point lying on the ray

NeRF

■ NeRF Training

1. Viewpoint Selection
2. Ray Composition $\rightarrow r(t) = o + td$
3. Select 5D input samples along the ray \rightarrow Hierarchical volume sampling
4. Query into MLP
5. Get predicted Color & Density
6. Render Color using volume ray casting
7. Compute rendering loss (Simply squared error between rendered and true pixel colors)

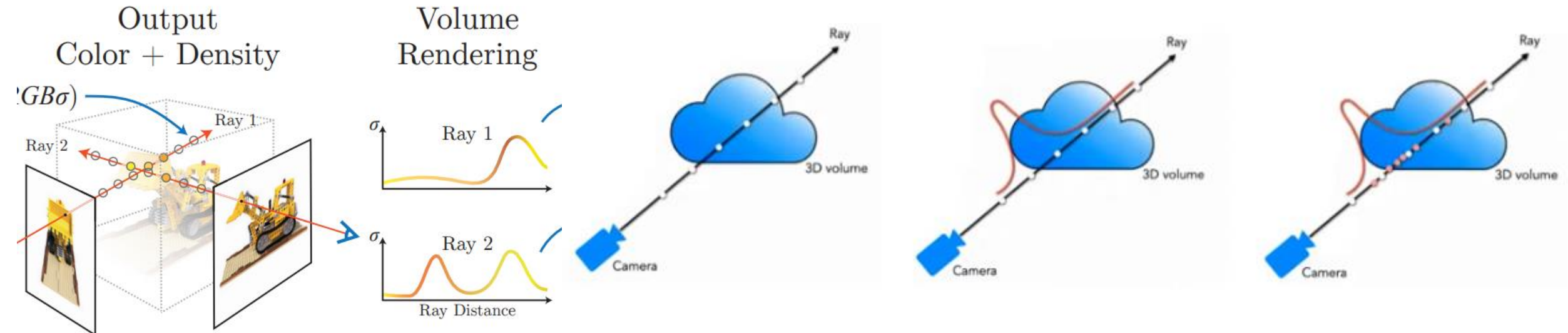


NeRF

■ NeRF Training

- Hierarchical volume sampling

1. **Stratified sampling**: Partition $[t_n, t_f]$ into N_c evenly-spaced bins and then draw one sample uniformly at random from within each bin (Coarse Network)
 - Then feed forward points to MLP and get weights
 - Given coarse network, producing a more informed sampling of points N_f
2. **Inverse transform sampling**: Sample N_f points according to the probability distribution (Fine Network)
 - Then feed forward $N_c + N_f$ sampled points to MLP and get volume density of that points



Recap

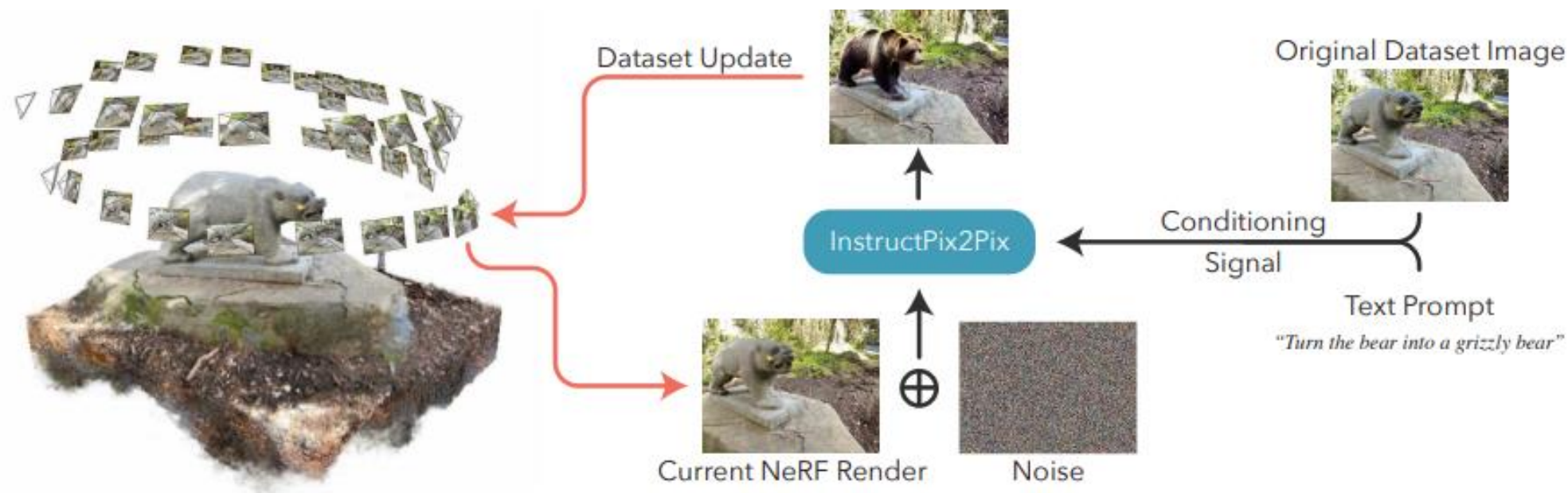


Figure 2: **Overview:** Our method gradually updates a reconstructed NeRF scene by iteratively updating the dataset images while training the NeRF: (1) an image is rendered from the scene at a training viewpoint, (2) it is edited by InstructPix2Pix given a global text instruction, (3) the training dataset image is replaced with the edited image, and (4) the NeRF continues training as usual.

InstructPix2Pix

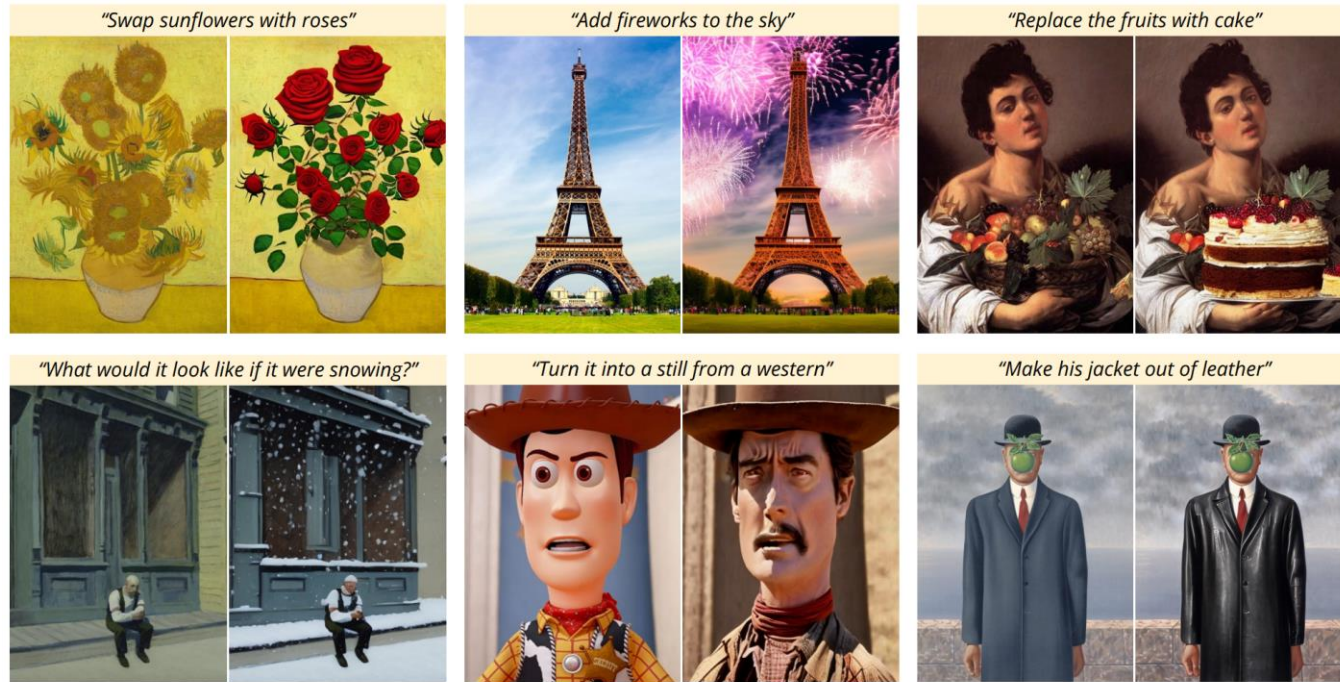


Figure 1. Given **an image** and **an instruction** for how to edit that image, our model performs the appropriate edit. Our model does not require full descriptions for the input or output image, and edits images in the forward pass without per-example inversion or fine-tuning.

Instruction-following Diffusion Model

(d) Inference on real images:

"turn her into a snake lady"



InstructPix2Pix

- A method for editing images based on human instructions

- **Method**


1. Generating an image editing dataset
2. Train a conditional diffusion model
 - Stable Diffusion (latent diffusion model)
 - Conditioned on:
 - RGB image (cI) – input image,
 - Text-based editing instruction (cT),
 - the model takes as input a noisy image called z_t .

Training Data Generation

(a) Generate text edits:

Input Caption: "photograph of a girl riding a horse" → GPT-3 → Instruction: "have her ride a dragon"
Edited Caption: "photograph of a girl riding a dragon"

(b) Generate paired images:

Input Caption: "photograph of a girl riding a horse"
Edited Caption: "photograph of a girl riding a dragon" → Stable Diffusion + Prompt2Prompt → 

(c) Generated training examples:



- The primary goal is to produce an estimate of the edited image z_0
- The diffusion model predicts the amount of noise present in the input image z_t using a denoising U-Net ϵ_θ , formulated as:

$$\hat{\epsilon} = \epsilon_\theta(z_t; t, cI, cT)$$

Instruct-NeRF2NeRF

- **Input**

- Reconstructed NeRF scene.
- Source data, including captured images, camera poses, and camera calibration.
- Natural-language editing instruction (e.g., "turn him into Albert Einstein").

- **Output**

- Edited version of the NeRF scene based on the provided edit instruction.
- Edited versions of the input images.

- **Method**

1. Edits dataset images using InstructPix2Pix
2. Iteratively updates image content at captured viewpoints (Iterative DU)
3. Consolidates these edits in 3D through standard NeRF training.

- **Inspiration**

- Built upon recent advances in diffusion models for image editing, particularly InstructPix2Pix.
- InstructPix2Pix uses an image-and-text conditioned diffusion model trained to edit natural images using human-provided instructions.
 - to make consistent with pre-existing 3D scene

Instruct-NeRF2NeRF

▪ Method

1. Edits dataset images using InstructPix2Pix

- It takes three inputs: Conditioning image(cI), Text instruction(cT), Noisy input(zt)
- To update an image at viewpoint v :
 - The unedited image I_{v0} is used as cI .
 - zt is a noised version of the current render at optimization step i .
- The replacement of an image is represented as:

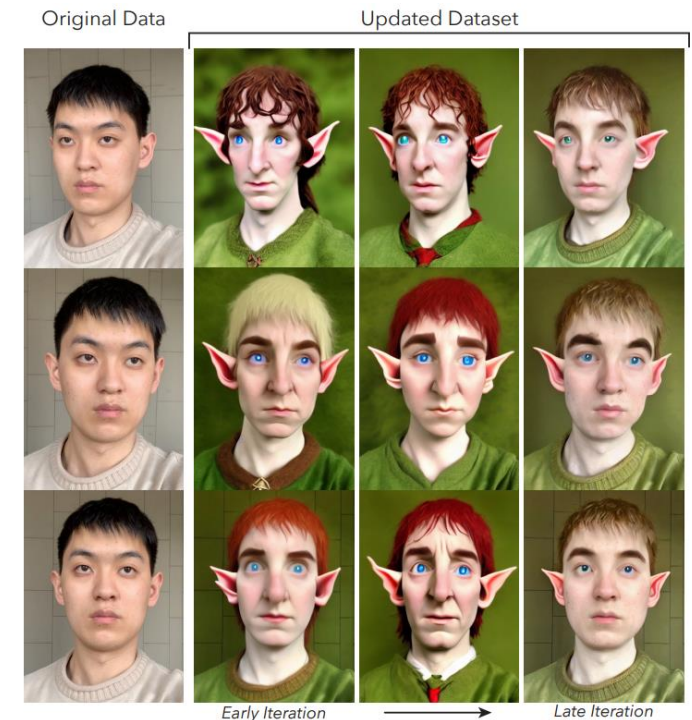
$$I_{vi+1} \leftarrow U_{\theta}(I_{vi}, t; I_{v0}, cT)$$

- U_{θ} is the DDIM sampling process with a fixed number of intermediate steps.

Instruct-NeRF2NeRF

2. Iteratively updates image content at captured viewpoints (Iterative DU)

- Initial Dataset: Optimization starts with a dataset of originally captured images, represented as (I_{v0}) .
- At each iteration:
 - Performs a number of image updates (d) and NeRF updates (n).
 - Image updates occur sequentially in a random order.
 - NeRF updates sample random rays from the entire training dataset.
- Editing process results in the replacement of dataset images with edited versions.
- Over time, edited images converge on a globally consistent depiction of the edited scene.



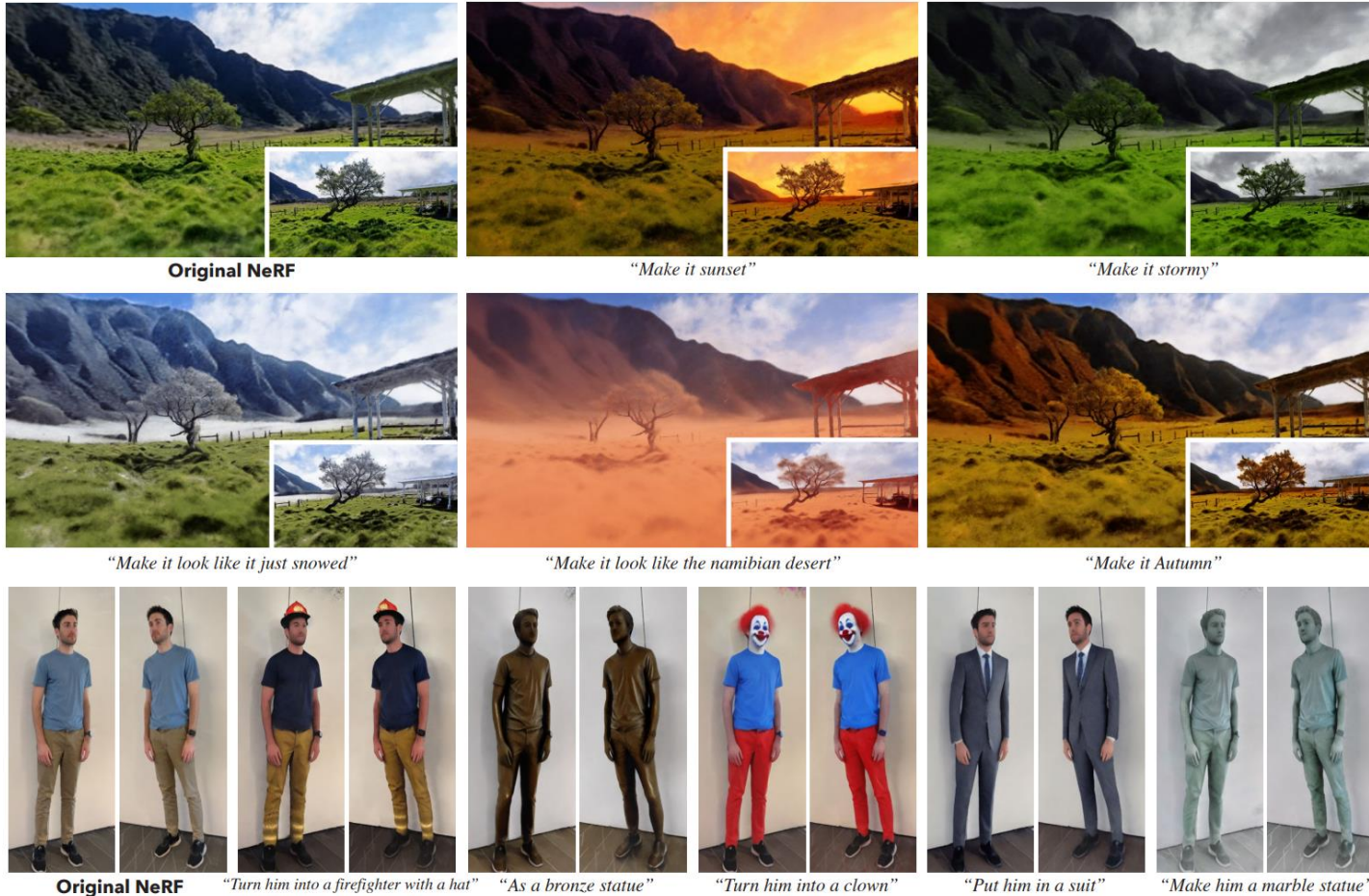
Instruct-NeRF2NeRF

■ Implementaion details

- Underlying NeRF Model: 'nerfacto' model from NeRFStudio is used as the base NeRF implementation.
- Parameters Affecting Diffusion Model:
 - $[t_{\min}, t_{\max}] = [0.02, 0.98]$ values define the amount of noise and signal retained from original images.
 - Denoised images are always sampled with 20 denoising steps.
- Diffusion Model Parameters:
 - Classifier-free guidance weights for text and image conditioning signals.
 - Default values: $s_I = 1.5$ and $s_T = 7.5$.
 - Users can hand-tune the guidance weight on an image for optimal edit strength before NeRF optimization.
- Manual Guidance Values: The paper's results use manually selected guidance values, but adjusting these can result in varying scene edits.
- Fixed 3D Hyperparameters: All other 3D hyperparameters are consistent across all experiments.
- NeRF Training Losses: L1 and LPIPS [54] losses are used for NeRF training.
- Training Iterations: The method is trained for a maximum of 30,000 iterations, which takes approximately an hour on a single NVIDIA Titan RTX with 15GB of memory.

Instruct-NeRF2NeRF

Qualitative Evaluation



Instruct-NeRF2NeRF

- Ablation study

Instruct-NeRF2NeRF

■ Limitations

- The method inherits limitations from InstructPix2Pix,
 - The method inherits limitations from InstructPix2Pix, including the inability to perform large spatial manipulations.
 - Adding entirely new objects or removing content to the scene is challenging.
- Two types of failure cases are demonstrated:
 1. When InstructPix2Pix fails in 2D, the method also fails in 3D
 2. When InstructPix2Pix succeeds in 2D but has large inconsistencies, the method cannot consolidate them in 3D.



Figure 9: **Limitations:** InstructPix2Pix cannot always perform the desired edit (top), and thus our method does not perform an edit. Sometimes InstructPix2Pix produces correct, but inconsistent edits in 2D that our method fails to consolidate in 3D (bottom).

Thank you!

Introduction

- Instruct-Pix2Pix

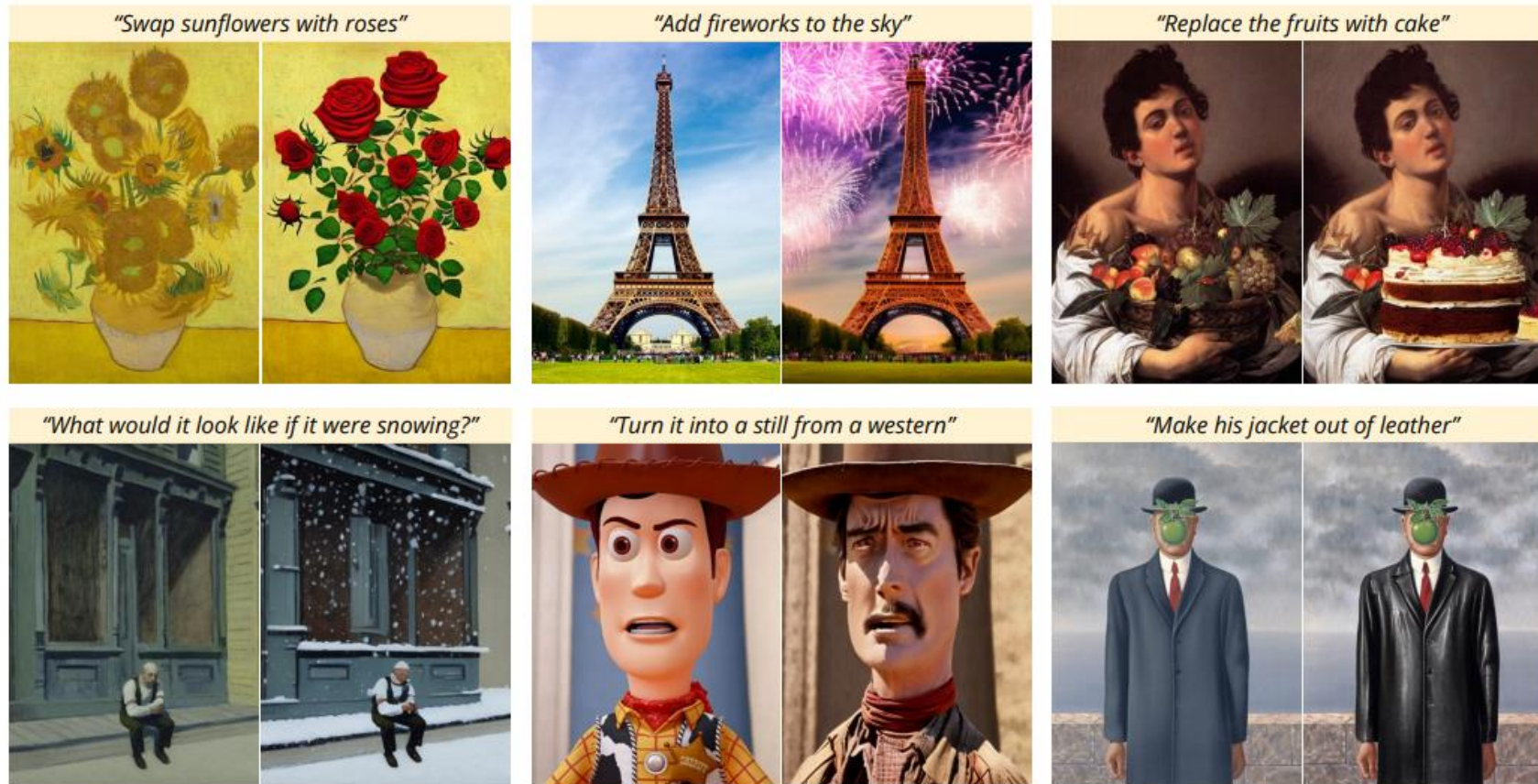


Figure 1. Given **an image** and **an instruction** for how to edit that image, our model performs the appropriate edit. Our model does not require full descriptions for the input or output image, and edits images in the forward pass without per-example inversion or fine-tuning.

Introduction

- **Limitations in 3D Asset Creation Tools**

- While creating 3D representations from real scenes is now more accessible, **the tools needed for editing these 3D assets are underdeveloped**. Traditional 3D model editing requires specialized tools and extensive training.

- **Challenges with Neural Representations**

- Neural representations often lack explicit surfaces, complicating the editing process and highlighting the need for modern 3D editing methods that are as accessible as the capture techniques.

Introduction

▪ **Limitations in 3D Asset Creation Tools**

- Traditional 3D model editing requires specialized tools and training.
- Accessibility issues in editing 3D representations derived from real scenes.

▪ **Challenges with Neural Representations**

- Neural representations often lack explicit surfaces, complicating editing.
- Need for modern, accessible 3D editing methods.

▪ **Instruct-NeRF2NeRF Methodology**

- Allows editing of 3D NeRF scenes using simple text instructions.
- Ensures edits are consistent in 3D, making the process intuitive for everyday users.

▪ **Use of 2D Diffusion Model**

- Extracts shape and appearance priors from a 2D diffusion model, InstructPix2Pix.
- Facilitates instruction-based 2D image editing due to limited 3D training data.

▪ **Overcoming Inconsistencies Across Viewpoints**

- Addresses inconsistencies caused by editing individual NeRF-rendered images.
- Iterative Dataset Update method to alternate between editing inputs and updating the 3D representation.

Introduction

- **Physical Editing of NeRFs**
 - NeRFs create photorealistic scenes but are hard to edit.
 - Techniques include physics-based optimization and bounding boxes.
 - ClimateNeRF adds weather effects like snow and flooding.
 - Focus on enabling diverse, arbitrary creative edits.
- **Artistic Stylization of NeRFs**
 - Developments in 3D stylization of NeRFs based on image stylization literature.
 - EditNeRF and ClipNeRF manipulate latent codes but struggle with localized edits.
 - Distilled Feature Fields and Neural Feature Fusion Fields for localized, guided edits.
 - Introduction of a language-based interface for intuitive 3D scene editing.
- **Generating 3D Content**
 - Large-scale models now generate 3D content from text prompts.
 - Challenges include limited control, focus on single objects, and lack of realism.
 - DreamFusion creates 3D models; RealFusion and SparseFusion use input images for realism.
 - This approach focuses on editing real NeRF scenes for consistent, realistic outcomes.
- **Instruction as an Editing Interface**
 - Rise of large language models like GPT and ChatGPT.
 - Natural language used as an interface for complex tasks.
 - Demonstrated effectiveness in 2D image tasks and beyond.
 - Pioneering instructional guidance in 3D editing for easy access by novices.

Introduction

■ **Instruct-NeRF2NeRF Methodology**

- The proposed Instruct-NeRF2NeRF method enables editing 3D NeRF scenes using simple text instructions. It operates on pre-captured 3D scenes to ensure edits are consistent in 3D, making 3D scene editing intuitive and accessible for everyday users.

■ **Use of 2D Diffusion Model**

- Due to limitations in training data for 3D generative models, shape and appearance priors are extracted from a 2D diffusion model, specifically InstructPix2Pix. This allows for instruction-based 2D image editing.

■ **Overcoming Inconsistencies Across Viewpoints**

- Applying InstructPix2Pix to individual NeRF-rendered images leads to inconsistent edits. To address this, an Iterative Dataset Update (Iterative DU) method alternates between editing NeRF input images and updating the 3D representation to integrate these edits.

■ **Evaluation and Comparison**

- The approach is evaluated on various captured NeRF scenes. It is compared with ablated versions of the method and naive implementations of the score distillation sampling (SDS) loss from DreamFusion. Additionally, it is qualitatively compared with a concurrent text-based stylization approach, demonstrating its ability to perform a wide range of edits on people, objects, and large-scale scenes.

Introduction

- **Physical Editing of NeRFs**
 - NeRFs create photorealistic scenes but are hard to edit.
 - Techniques include physics-based optimization and bounding boxes.
 - ClimateNeRF adds weather effects like snow and flooding.
 - Focus on enabling diverse, arbitrary creative edits.
- **Artistic Stylization of NeRFs**
 - Developments in 3D stylization of NeRFs based on image stylization literature.
 - EditNeRF and ClipNeRF manipulate latent codes but struggle with localized edits.
 - Distilled Feature Fields and Neural Feature Fusion Fields for localized, guided edits.
 - Introduction of a language-based interface for intuitive 3D scene editing.

Introduction

▪ **Instruct-NeRF2NeRF Methodology**

- The proposed Instruct-NeRF2NeRF method enables editing 3D NeRF scenes using simple text instructions. It operates on pre-captured 3D scenes to ensure edits are consistent in 3D, making 3D scene editing intuitive and accessible for everyday users.

▪ **Use of 2D Diffusion Model**

- Due to limitations in training data for 3D generative models, shape and appearance priors are extracted from a 2D diffusion model, specifically InstructPix2Pix. This allows for instruction-based 2D image editing.

▪ **Overcoming Inconsistencies Across Viewpoints**

- Applying InstructPix2Pix to individual NeRF-rendered images leads to inconsistent edits. To address this, an Iterative Dataset Update (Iterative DU) method alternates between editing NeRF input images and updating the 3D representation to integrate these edits.

▪ **Evaluation and Comparison**

- The approach is evaluated on various captured NeRF scenes. It is compared with ablated versions of the method and naive implementations of the score distillation sampling (SDS) loss from DreamFusion. Additionally, it is qualitatively compared with a concurrent text-based stylization approach, demonstrating its ability to perform a wide range of edits on people, objects, and large-scale scenes.

▪ Volume Rendering

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t) \sigma(\mathbf{r}(t)) \mathbf{c}(\mathbf{r}(t), \mathbf{d}) dt, \quad \text{where } T(t) = \exp\left(-\int_{t_n}^t \sigma(\mathbf{r}(s)) ds\right)$$

- $T(t)$: Accumulated transmittance
- $\sigma(\mathbf{r}(t))$: Volume density or Occupancy at a 3D point
- $\mathbf{c}(\mathbf{r}(t), \mathbf{d})$: Color at a 3D point lying on the ray

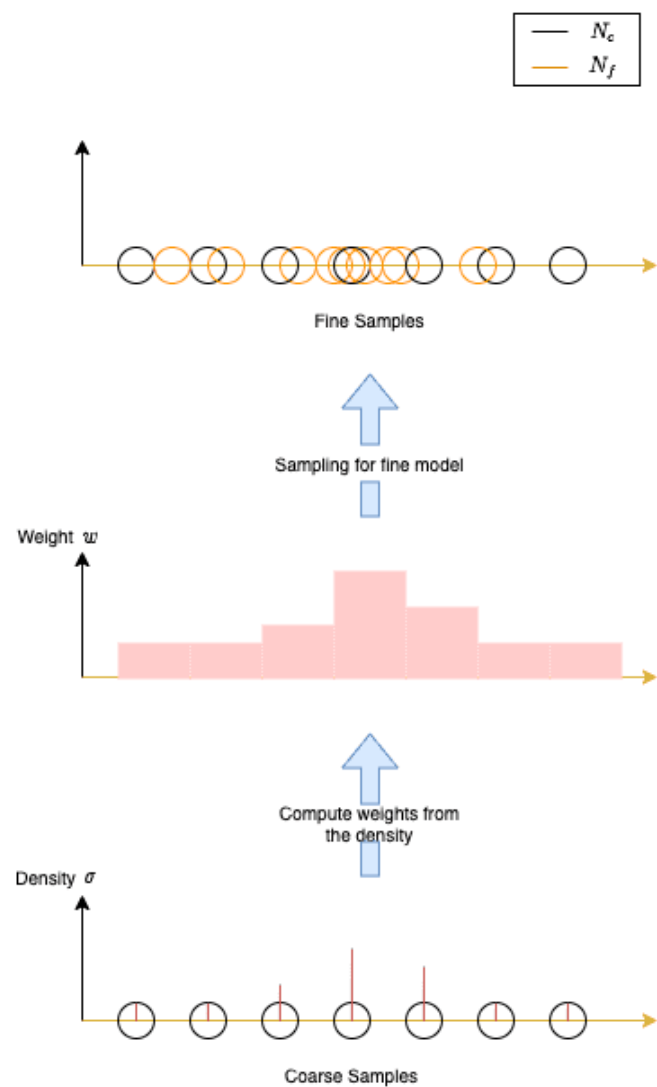
▪ Volume Rendering

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t) \sigma(\mathbf{r}(t)) \mathbf{c}(\mathbf{r}(t), \mathbf{d}) dt, \quad \text{where } T(t) = \exp\left(-\int_{t_n}^t \sigma(\mathbf{r}(s)) ds\right)$$

$$t_i \sim \mathcal{U}\left[t_n + \frac{i-1}{N}(t_f - t_n), t_n + \frac{i}{N}(t_f - t_n)\right]$$

- $T(t)$: Accumulated transmittance
 - $\sigma(\mathbf{r}(t))$: Volume density or Occupancy at a 3D point
 - $\mathbf{c}(\mathbf{r}(t), \mathbf{d})$: Color at a 3D point lying on the ray
-
- $T(t)$: Accumulated transmittance
 - $\sigma(\mathbf{r}(t))$: Volume density or Occupancy at a 3D point
 - $\mathbf{c}(\mathbf{r}(t), \mathbf{d})$: Color at a 3D point lying on the ray

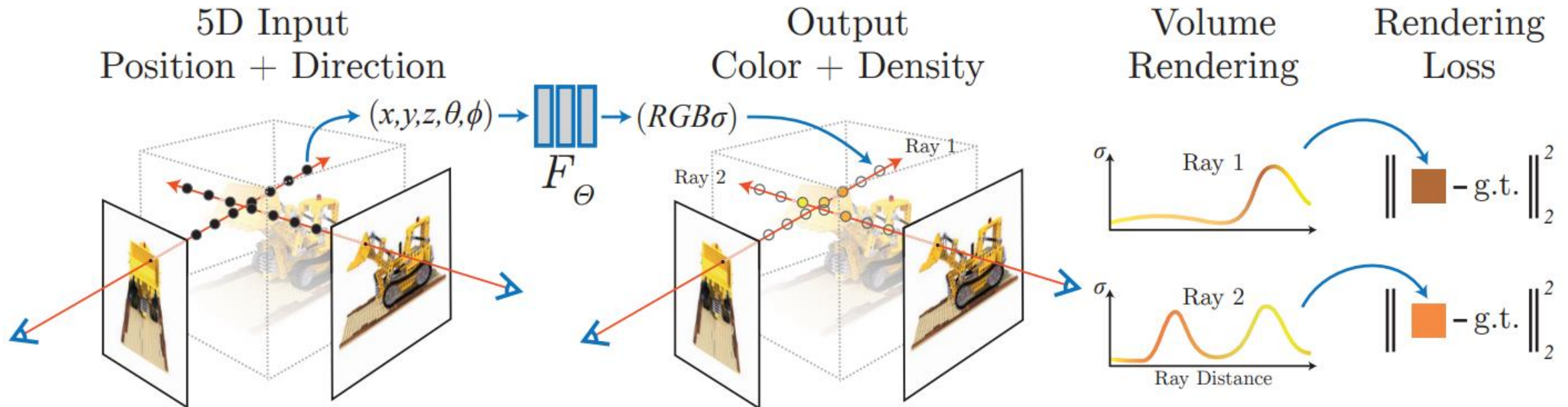
NeRF



NeRF

■ NeRF Training

1. Viewpoint Selection
2. Ray Composition
3. Select 5D input samples along the ray \rightarrow Hierarchical volume sampling
4. Query into MLP
5. Get predicted Color & Density
6. Render Color using volume ray casting
7. Compute rendering loss (Simply squared error between rendered and true pixel colors)



InstructPix2Pix

- Diffusion-based method for image editing.
 - Denoising diffusion models, which transform noisy samples towards a modeled data distribution.
- Conditioned on:
 - RGB image (cI),
 - Text-based editing instruction (cT),
 - the model takes as input a noisy image (or pure noise) called z_t .
- The primary goal is to produce an estimate of the edited image z_0 , which is an edited version of the input RGB image cI based on the provided text instruction cT .
- The diffusion model predicts the amount of noise present in the input image z_t using a denoising U-Net ϵ_θ , formulated as:
 - $\hat{\epsilon} = \epsilon_\theta(z_t; t, cI, cT)$
- The predicted noise $\hat{\epsilon}$ is used to derive \hat{z}_0 , the estimate of the edited image.
- The denoising process can be applied at any timestep t within the range $[0, T]$, allowing for images with varying degrees of noise.
 - Larger t values produce estimates with more variance, while smaller t values result in lower variance estimates that closely follow the visible image signal in z_t .
- InstructPix2Pix operates based on a latent diffusion model, meaning the diffusion process entirely operates within an encoded latent domain.
 - Variables cI and z_0 are latent images created by encoding an RGB image ($E(I)$).
 - To obtain an RGB image from the diffusion model, the predicted latents \hat{z}_0 are decoded using a decoder, giving $\hat{I} = D(\hat{z}_0)$.