Optimized RecolorNeRF: Enhancing Efficiency and Performance for Dynamic Color Editing in 3D Scenes

Vyshnav Achuthan
University of Maryland
College Park,MD
vyachu07@umd.edu

Kiran Suvas Patil
University of Maryland
College Park,MD
kpatil27@umd.edu

Abstract

The burgeoning interest in Radiance fields and their diverse applications has propelled them into the mainstream. While considerable research has been devoted to various appearance modifications, the exploration of an optimized, view-consistent recoloring has remained relatively understudied. Existing studies predominantly focus on datasets readily available within the domain, leaving a significant gap in understanding their application in real-time datasets.

Building upon the foundation of Recolor NeRF, a user-friendly color editing approach for neural radiance fields, we present an enhanced iteration of the model fine-tuned specifically for real-time dataset applications. Grounded in the fundamental principle that color manipulation can be achieved by directly altering the color components of a palette, our investigation entailed assessing the robustness of the original model in a custom real-time rendering environment.

The optimized model's performance was meticulously evaluated through photometric reconstruction loss and a comprehensive user study, aimed at validating the method's advantages. Our findings conclusively demonstrate that optimizing the encoder model in the Recolor NeRF backbone outperforms the original model both quantitatively and qualitatively, particularly in the context of color editing within real-time rendering scenarios. Our implementation is available at https://github.com/kirangit27/Optimized-RecolorNeRF.git

1. Introduction

Neural Radiance Fields (NeRF) have emerged as a potent representation for 3D scenes, poised to become a foundational medium comparable to images and videos in the future. However, efficient and high-quality editing within this representation is crucial for widespread adoption. While existing methods have delved into appearance editing, achieving efficient and view-consistent recoloring remains an underexplored domain.

Addressing this challenge, there are current methods for recoloring an image, that use a palette-based color editing [2,38], which involves active user interaction. Palette Color Editing [2], extracts a color palette from an image, assigns colors to pure-colored layers, uses sparse color unmixing (SCU) energy formulation and automatic color model estimation, and decomposes the image based on palette weights, facilitating precise color editing. Developing off these techniques RecolorNeRF [13] was introduced as the first method using a learnable palette for layer-wise decomposition in 3D scenes, advancing photo-realistic PCE on NeRF representations. To address the issue of poor view inconsistency, RecolorNeRf strategically decomposes scenes into pure-colored layers, forming an associated palette. This design allows users to independently modify the color of each layer, facilitating precise and high-fidelity recoloring results. It enables efficient palette-based editing, ensuring the representativeness of each layer's color.

To achieve efficient palette-based editing, we formulate the problem as an optimization task, jointly optimizing layers and their blending weights with the NeRF representation. The optimization process includes regularizers for palette learning and layer sparsity, along with specific loss functions to guarantee high-quality recoloring results. The resulting recolored novel-view renderings showcase RecolorNeRF's effectiveness in surpassing baseline methods both quantitatively and qualitatively, even in complex real-world scenes.

In summary, RecolorNeRF provides a user-friendly and efficient solution for color editing of 3D scenes. Leveraging layer decomposition and learnable palettes, it enables precise and controllable recoloring. The method's effectiveness is substantiated through extensive experiments and comparative evaluations, positioning it as a promising approach for high-quality scene recoloring applications.

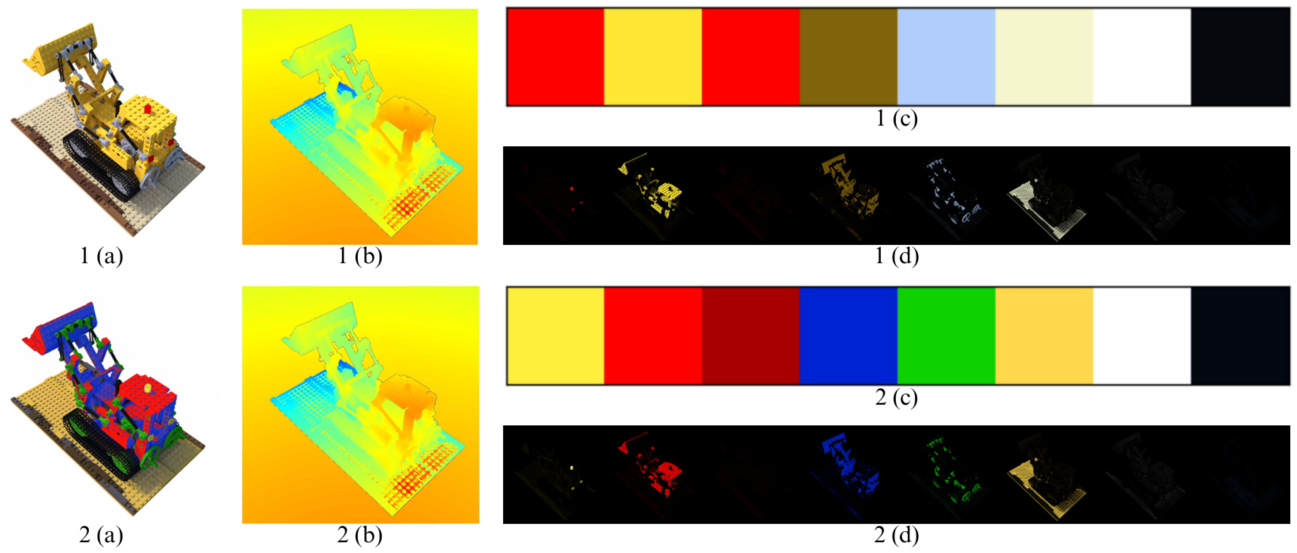


Figure 1. **Layer Decomposed Radiance Fields**. 1(a) Input RGB image, 1(b) Depth-map generated for the input image, 1(c) Optimized color palette 1(d) Layer Decomposed Radiance Fields on input NeRF, 2(a) Output RGB image, 2(b) Depth-map generated for the output, 2(c) Desired color palette, 2(d) Input palette Layer Decomposed Radiance Fields on result.

2. Related Work

Neural Radiance Field (NeRF) and TensoRF. NeRF [25] stands as a prominent representation of three-dimensional spaces, akin to the traditional counterparts of triangle meshes and polygons. This model is designed to acquire a comprehensive understanding of 3D space by learning to generate both the density and view-dependent radiance attributes for a specific point. Such generation is achieved by inputting the coordinates (x, y, z) and viewing direction (θ, ϕ) into a Multilayer Perceptron (MLP). The success of NeRF has inspired a plethora of research work in implicit representations that extend NeRF to dynamic scenes [29], relighting [5] for 3D models and also introducing generative models [35].

Although NeRF, with its pure MLP-based representation, contributes to realistic rendering and a compact model, it is acknowledged to have limitations, particularly in terms of slow reconstruction and rendering. Recent works [12,39] have leveraged the use of voxel grid features in radiance field modeling, which achieves faster rendering. Nonetheless, grid-based methods [22, 36], despite their utilization, demand extended reconstruction periods and may result in elevated memory expenses, thereby compromising the inherent compactness characteristic of NeRF. Another approach that came off feature grids introduces an innovative tensorial scene representation that capitalizes on tensor factorization techniques [7]. This methodology facilitates swift reconstruction and efficient modeling while preserving the desirable compactness attributes of NeRF. TensoRF achieves state-of-the-art rendering quality while maintaining a significantly lower memory footprint than previous methods.

NeRF Editing NeRF editing is one of the major challenges in the field of implicit scene representation. The first approach to editing implicit representations was a model that encodes the shape and appearance of simple objects into a latent code, allowing for easy interpolation and modification of these objects [23]. Further, NeRF-Editing [40] focuses on geometry content by retrieving manipulable explicit meshes and then inserting the movements of the proxies back into the implicit representations. NVDiffRec [26] uses multi-view captures to separate 3D geometry and lighting effects in order to extract editable BRDF materials. Other text prompt-based approaches like CLIP-NeRF [5] and UPST-NeRF [8], are methods for manipulating the shape and appearance of NeRF based on text prompts and reference images. The architecture utilizes all of CLIP's capabilities to combine text and image-driven manipulations into a single model.

Palette-based Color Editing Addressing color editing challenges in images involves extracting editable color patterns to prevent "color pollution," where edits for different targets become mixed, causing contamination-like artifacts. Methods often rely on user guidance through scribbles or spatial markers for pattern extraction, intending to propagate desired colors in meaningful patterns. Energy minimization, utilizing Euclidean or diffusion distance in feature space, is a common approach, and several algorithms [4,18,27] are designed for user-specified edits. User-scribble-based methods incur higher costs compared to reference-based or palette-based alternatives. Color transfer, a reference-based technique, absorbs color patterns from another image, with early methods [28, 32] focusing on matching means and standard deviations or color probability density functions. Histogram-based representations, like Delon et al.'s method [10], use segmented color histograms to construct palettes for source-to-target color transfer.

Palette-based recoloring methods strike a balance between user control and recoloring capability by enabling users to modify color components within a predefined set of colors. The crucial step involves palette extraction, with methods such as fitting predictive models for human palette preference to generate perceptual palettes [6, 11, 21]. An alternative approach utilizes the k-means algorithm to cluster image colors in RGB space, capturing prominent colors ([14, 30]). Another method involves computing and simplifying the convex hull enclosing all color samples to obtain more "primary" palettes, better representing the image's color gamut ([16]). Once the palette is extracted, the image is decomposed into layers, each corresponding to an important map of a palette color. Layer decomposition, achieved through methods like weighted additive mixing or alpha blending, acts as a reverse procedure of layer composition. Common alpha blending operators, such as "over," are used in recent methods for order-dependent layer composition ([9, 15, 16]). Our method utilizes alpha blending, allowing users control over palette order and facilitating a sparser layer decomposition.

3. Methodology

In this section, we start with the backbone architectural analysis of TensoRF [7] as a 3D modeling and reconstructing radiance fields model. Along with the backbone study, we then present our optimization framework deployed in the TensoRF background which can improve reconstruction quality and efficiency, without compensating the reconstruction speed. Finally, we delve into the understanding of automatic layer breakdown through the activation of alpha blending and simultaneous palette and scene representation optimization, which is an essential part in RecolorNeRF[cite].

3.1. Reconstruction using TensoRF

TensoRF [7] is a novel radiance field representation that models the scene as a 4D tensor, enabling efficient scene reconstruction and modeling. The radiance field is factorized into multiple compact low-rank tensor components, leading to a highly compact and fast reconstruction method. TensoRF achieves state-of-the-art rendering quality while maintaining a significantly lower memory footprint than previous methods.

TensoRF represents the radiance field as a 4D tensor, comprising feature channels and spatial coordinates (X, Y, Z). This tensor is factorized into low-rank components for efficient scene reconstruction. Two 3D voxel grids are employed: one for multi-channel color and another for single-channel density. Density features are used as the volume density σ, obtained from the 3D location x. A predefined function S transforms color features into view-dependent color cc based on the viewing direction dd, creating a grid-based brightness field. The expression for this grid-based brightness field is as follows:

$$\sigma, c = G_\sigma, S(G_c(x), d) \tag{1}$$

where $G_\sigma(x), G_c(x)$ are trilinearly interpolated featured from two grids at a point x, and are factorized tensors. S represents a pre-selected function that converts an appearance feature vector and a viewing direction to color.

The radiance field tensors in TensoRF are decomposed into compact components. The volume density is represented by a single-channel grid G_σ, directly depicting volume density. Utilizing VM Decomposition [7], the density at location xx is factorized as:

$$G_\sigma = \sum_{r=1}^{R_\sigma} \sum_{m \in XYZ} A^m_{\sigma,r} \tag{2}$$

Here, $A^m_{\sigma,r}$ are compact components from the factorization, each capturing the interaction of vectors and matrices for specific spatial modes (m) and ranks (r), representing the dot product of vector factors with spatial data and matrix factors with spatial dimension interactions.

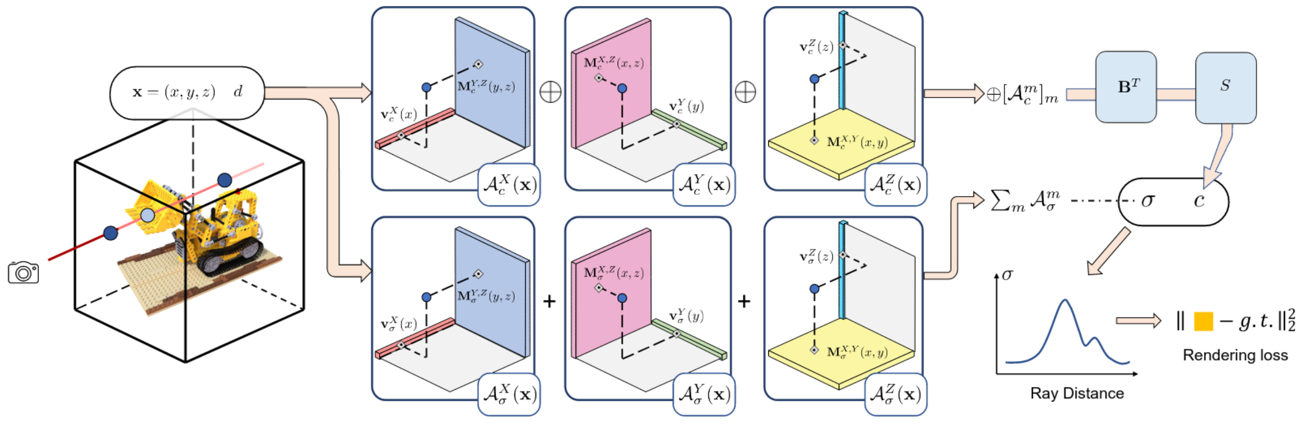
Similarly, the appearance tensor G_c is a multi-dimensional data structure that captures the appearance features and their influence on the color of the scene when viewed from different directions. This can be factorized as

$$G_c = B(\oplus[A^m_{c,r}(x)]_{m,r} \tag{3}$$

where B is the appearance matrix, the concatenated appearance feature vector undergoes transformation before being fed into the function S. Combining Eqn.2 and 3, the continuous volume density and view-dependent color at any 3D location x along with the viewing direction is obtained.Eqn.4 represents the combined form and also uses a function S, that converts an appearance feature vector and a viewing direction to color. Fig 2 represents how tensor decomposition is performed along with radiance field reconstruction.

$$\sigma, c = \sum_{r=1}^{R_\sigma} \sum_{m \in XYZ} A^m_{\sigma,r}, S(B(\oplus[A^m_{c,r}(x)]_{m,r}) \tag{4}$$

$$C = \sum_{q=1}^{Q} \tau_q (i - exp(-\sigma_q \Delta_q)) c_q, \tau_q = exp(-\sum_{p=1}^{q-1} \sigma_p \Delta_p) \tag{5}$$



$\mathbf{x} = (x, y, z) \quad d$
$\mathbf{v}_c^X(x)$
$\mathbf{M}_c^{Y,Z}(y,z)$
$\mathcal{A}_c^X(\mathbf{x})$
$\mathbf{M}_c^{X,Z}(x,z)$
$\mathbf{v}_c^Y(y)$
$\mathcal{A}_c^Y(\mathbf{x})$
$\mathbf{v}_c^Z(z)$
$\mathbf{M}_c^{X,Y}(x,y)$
$\mathcal{A}_c^Z(\mathbf{x})$
$\oplus[\mathcal{A}_c^m]_m$
$\mathbf{B}^T$
S
$\mathbf{v}_\sigma^X(x)$
$\mathbf{M}_\sigma^{Y,Z}(y,z)$
$\mathcal{A}_\sigma^X(\mathbf{x})$
$\mathbf{M}_\sigma^{X,Z}(x,z)$
$\mathbf{v}_\sigma^Y(y)$
$\mathcal{A}_\sigma^Y(\mathbf{x})$
$\mathbf{v}_\sigma^Z(z)$
$\mathbf{M}_\sigma^{X,Y}(x,y)$
$\mathcal{A}_\sigma^Z(\mathbf{x})$
$\sum_m \mathcal{A}_\sigma^m$
$\sigma \quad c$
σ
Ray Distance
$\| \; - g.t.\|_2^2$
Rendering loss

Figure 2. **TensoRF Pipeline**. From the original TensoRF [7] paper, the pipeline explains of how radiance fields are set as tensors of vectors and Matrices. These describe the scene along their corresponding axes and are used for computing volume density and color c.

To generate images, the TensoRF approach employs differentiable volume rendering, used by NeRF [25]. For each pixel, ray marching is performed, sampling Q shading points along its path. The pixel color is then computed based on the sampled density σ_q and color c_q values at locations x_q obtained from tensor decomposition, utilizing the ray step size Δ_q and transmittance as given in equation.5.

3.1.1 Optimizing Shading Function

Architectures like TensoRF [7] and Instant-NGP [37] accelerate the training of luminance fields by converting implicit scene representations into explicit feature grids using spherical harmonics or implicit shading modules. TensoRF's key shading function S transforms an appearance feature vector and viewing direction into color, employing either a shallow Multi-Layer Perceptron (MLP) or spherical harmonics. This feature vector captures texture, and other visual elements, while the viewing direction influences the radiance field's appearance based on the input vector.

In our baseline, we opted for a shallow MLP over spherical harmonics due to the latter's fixed basis functions, which are less adaptable in capturing detailed or variable features in real time. MLPs, with their flexible weight and bias adjustments, better represent such variations, utilizing a two-layer architecture with 128-channel hidden layers and ReLU activation. However, this MLP architecture, while effective, doesn't explicitly model space, leading to potential shortcomings in handling complex spatial dependencies and details.

This limitation prompted our exploration of stronger decoder functions to enhance detailing and improve alpha blending segmentation. We drew insights from existing works like UNerf [17] and Vision Transformer for NeRF [20], which demonstrate the efficacy of decoder networks in capturing spatial hierarchies and contextual information, guiding our approach towards integrating more sophisticated decoding mechanisms in our model.

We integrate the architecture of UNet [33], renowned for 2D image segmentation, with a shallow MLP, Fig.3, creating a novel architecture that leverages each component's strengths for enhanced neural rendering. The U-Net,Fig.4, with its encoder-decoder structure, excels in extracting and preserving spatial features, efficiently processing hierarchical information to capture both global structures and local details. This spatially enriched feature map from U-Net serves as the input for the MLP renderer, which further refines and renders these features.

Our U-Net encoder consists of convolutional layers with 64 channels, kernel size 3, and padding of 1, followed by ReLU activations and MaxPool2d for downsampling. The decoder, mirroring the encoder, ends in a Conv2d layer with 3 output channels for RGB imaging, coupled with a Sigmoid activation. The MLP Renderer, with U-Net's input channels, utilizes linear layers and ReLU activations, concluding with a Sigmoid for normalized RGB output. Positional encoding on viewing directions augments the feature space for MLP processing, as depicted in Fig.3, creating a cohesive pipeline for rendering tasks. The concatenated input and features, post-positional encoding, are processed through the U-Net encoder-decoder, and then combined with the view directions to feed into the MLP renderer for the final RGB output. This integration capitalizes on U-Net's spatially informed features and MLP's detail rendering capabilities, yielding a comprehensive and visually superior representation.

3.2. Recolor NeRF

Palette-based layer decomposition: The method decomposes a radiance field into K pure-colored layers, each cor-

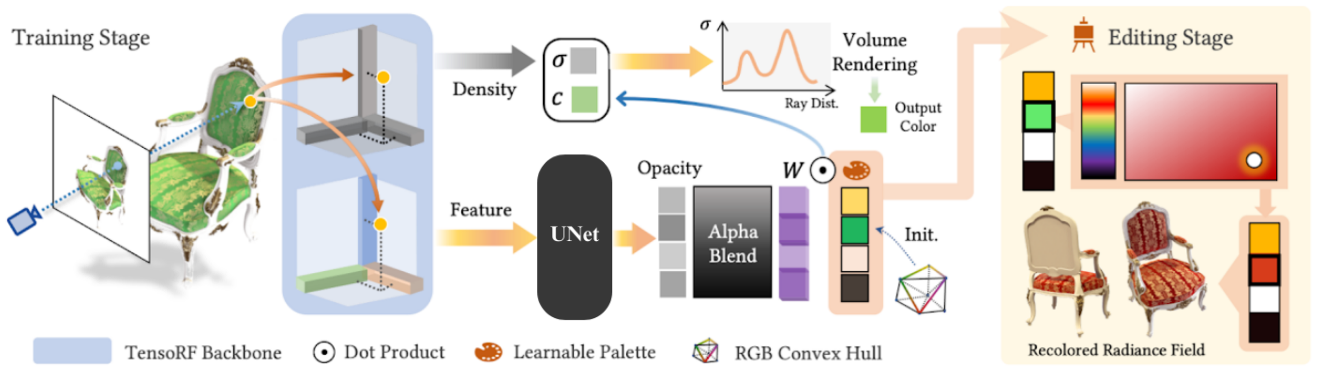


Figure 3. **Optimized RecolorNeRF Pipeline**. from the original RecolorNeRF paper [13] (MLP replaced by UNet+MLP), In the training phase, optimization is performed on the TensoRF [7] backbone and the trainable palette to effectively represent the color of any given query point. This is achieved through alpha compositing over the learned palette. In the editing phase, users have the straightforward ability to recolor the entire radiance field by simply substituting a color in the palette with a new one.

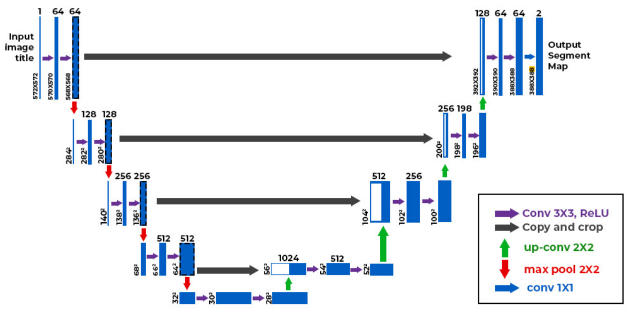


Figure 4. **UNet**. UNet Architecture

responding to an item in a palette p. The model outputs view-dependent opacity values $\alpha_i(x, d)$ for each layer, and composites the layers via alpha blending to obtain the radiance field

$$c(x, d) = \sum_{i=1}^{K} \zeta(\alpha_i(x, d))p_i \tag{6}$$

where $\zeta()$ is an alpha blending activation that maps opacity to generalized barycentric coordinates.

Alpha blending activation: The method defines a function $T(\alpha_i(x, d))$ to transform the opacity values into logarithmic space, and then applies the exponential function to obtain

$$\zeta(\alpha_i(x, d)) = \exp(T(\alpha_i(x, d))) \tag{7}$$

This activation is differentiable and numerically stable and ensures that the radiance field is a convex combination of the palette colors.

Palette Learning: The goal is to optimize the palette p to express the entire color space of the 3D scene1. The palette is modeled as a set of convex hull vertices that enclose all the pixel colors from the input images. The palette size is reduced by jointly optimizing the palette and the scene representation with novel convex hull regularization.

Convex Hull Bounding Regularization: This term penalizes out-of-bound palette colors by measuring the distance from the palette color p_i to its closest point on the simplex facets of the bounding convex hull U which is given by,

$$L_{bd} = \sum_{i=1}^{K} \omega_{out} ⊮\{p_i \notin U\} \min_{s \in S(U)} \|p_i - \text{Nearest}(p_i|s)\|^2 \tag{8}$$

where ω_{out} is the penalty strength and Nearest$(p_i|s)$ finds the closest point on the simplex facet s to the color p_i.

Projection-based Convex Regularization: This term projects the palette colors to the vertices of the bounding convex hull, imposing a constraint $P \in \text{hull}(\text{vertices}(U))$.

$$L_{proj} = \sum_{i=1}^{K} \omega_{in} ⊮\{p_i \in U\} \min_{v \in HullVertices(U)} \|p_i - v\|^2 \tag{9}$$

Palette Initialization: The palette can be randomly initialized or user-designed. A prior palette can promote the editability of the scene by allowing the user to control the order and selection of the palette colors.

4. Experiments

Datasets Baseline evaluations are done on 360° synthesis objects as the RecolorNeRF [13] paper. The three regulation datasets include the Synthetic NeRF Blender dataset [25], which includes Lego, Drums, Chairs, and Ship; the forward-facing real word scenes LLFF dataset [24], containing Horns, Flower, and Fortress; and the real-world objects like tanks and temples dataset.

This study includes the production of a unique dataset (Fig 5) utilizing an item captured with a simple mobile camera, in addition to baseline datasets that have already been generated. The main goal of this dataset is to assess the

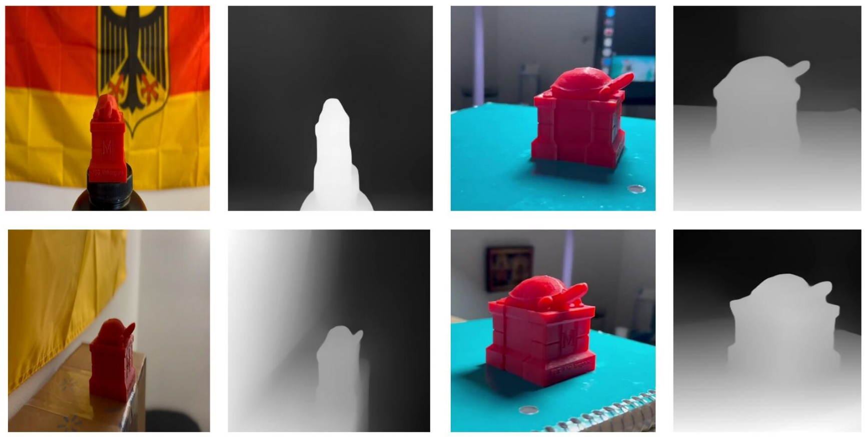


Figure 5. **Terp Dataset**. The above images show some of the frames we captured using the testudo 3D printed model on various colored backgrounds, along with their depth maps generated using DPT [31]

segmented colorization of the item and the authenticity of the scene reconstruction while maintaining the background information's integrity.

For the dataset generation, we select one object projected in two different scenes to test how much background color information is learned by the learnable palette. The dataset generation pipeline is majorly done in accordance with the Instant-NGP [37] method, where the video of the scene is split into n-second frames. The initial camera parameters which are required for Nerf transforms, are extracted using COLMAP [34]. COLMAP uses a sequential matcher by default when launching, which works well for photos that were captured from a smoothly shifting camera path.

In our initial attempts at generating depth maps, we found traditional methods like setting minimum depth thresholds and using OpenCV's disparity techniques for stereo images insufficient. This led us to explore monocular depth estimation using deep learning approaches, as suggested in research by [3] and [19]. However, these were outperformed by the Dense Prediction Transformer (DPT) [31] model, a notable development in depth estimation. The DPT model's transformer architecture is particularly adept at depth prediction tasks and managing sequential data. Its proficiency in maintaining depth consistency across varied scenes and capturing intricate details proved highly effective for our dataset, which includes recordings with camera shake. Consequently, the depth maps produced, as exemplified in our figures, Fig.5, showed considerable improvement in accurately interpreting and reconstructing depth from single images, underscoring the DPT model's suitability for our specific needs.

Implementation Details Our model, developed in PyTorch [1] and based on the TensoRF framework, applies a novel approach to 3D space geometry factorization using the TensoRF-VM backbone model. This model distinguishes geometry and appearance by employing separate density and opacity fields, initially set with vector and matrix resolutions of 128, which are progressively increased to 300, and for forward-facing scenes, up to 640. The training process spans 30,000 iterations using the Adam optimizer, conducted on an NVIDIA RTX 2080 GPU. We adopt a large batch size of 1024, enhancing our model's learning efficiency. To refine the model, we apply a Total Variation (TV) loss and an L1 norm penalty, with the L1 norm weight starting at 8e-5 and halved post the first resolution upsampling. Our dataset is partitioned into 70/10/20 training, validation, and testing sequences, ensuring a comprehensive evaluation. This methodological combination results in a robust and efficient pipeline for 3D scene rendering and understanding.

5. Experiments

5.1. Quantitative Analysis

In our quantitative analysis, we evaluate the performance of the UNeT + MLP rendering architecture with TensoRF against the baseline TensoRF with MLP model by assessing photometric errors. This comparison will first be conducted using the Synthetic NeRF dataset, followed by an analysis with our custom dataset, providing a comprehensive evaluation of the architectures' effectiveness. Table 1 represents the PSNR, SSIM, and LPIPS metrics and comparisons done on the Lego NeRF dataset along with the custom dataset.

The table 1 demonstrates that on a custom dataset, optimized TensoRF models with AlexNet and SqueezeNet surpass their standard versions in PSNR and SSIM, delivering clearer and more accurate images. However, the optimized TensoRF with SqueezeNet shows a higher LPIPS score, indicating a small reduction in perceptual quality. This is attributed to the Unet architecture in the optimized models, which enhances resolution and detail but can lead to slight perceptual differences.

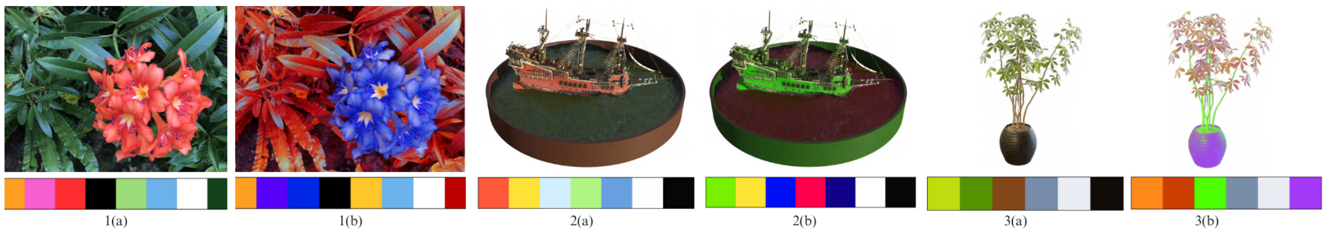


Figure 6. **Color Editing Results**. Gallery of our color editing results. The images 1(a), 2(a), and 3(a) are the reference images before editing along with the optimized palette beneath. The other 3 images showcase examples of color editing with the corresponding edited palettes.

In Table 2, experiments were done on the NeRF synehtic Lego Model. The Optimized TensoRF models, both with AlexNet and SqueezeNet, demonstrate superior performance in all three metrics compared to their standard counterparts, indicating higher image quality and better perceptual similarity. Specifically, the Optimized TensoRF with SqueezeNet achieves the highest scores across all metrics, showcasing its effectiveness in rendering high-quality, realistic images on synthetic datasets.

These results highlight the improvements in image clarity, structural accuracy, and perceptual likeness achieved through optimization in these advanced 3D rendering models.

5.2. Qualitative Analysis

In Fig.7, we present a qualitative analysis of our custom-generated dataset, the Terp Dataset. The evaluation encompasses both the original RecolorNeRF and our optimized method. Fig.7 1(a) illustrates an RGB image from the set used to train the RecolorNeRF model. Notably, the optimized palette for the input image (Fig.7 1(c)) exhibits a misalignment with the Layer Decomposed Radiance Fields generated by RecolorNeRF (Fig.7 1(b)). Conversely, our method demonstrates a well-aligned relationship between the optimized palette (Fig.7 1(c)) and the corresponding Layer Decomposed Radiance Fields (Fig.7 1(f)).

This discrepancy in the interpretation of input Layer Decomposed Radiance Fields leads to suboptimal outcomes for the original RecolorNeRF in generating desired edited color palettes (Fig.7 2(a) and 2(b)), while our method excels, yielding excellent results (Fig.7 2(e) and 2(f)). The misalignment is evident when comparing the Layer Decomposed Radiance Fields for the edited palette, where our method (Fig.7 2(d) and 2(f)) showcases alignment, while RecolorNeRF (Fig.7 2(d) and 2(b)) falls short in achieving this synchronization.

6. Conclusion

RecolorNeRF proves to be a highly efficient approach for generating photo-realistic novel views through the decomposition of neural radiance fields into distinct, pure-colored layers. This method involves the joint optimization of decomposed layers alongside a learnable palette, promoting a more separated decomposition and enhancing the representativeness of colors in the palette. The process involves stacking these optimized layers with alpha blending to generate color radiance, ultimately resulting in the production of final, photo-realistic images. Despite the positive aspects mentioned, it is evident from the experiments detailed in the above section that additional optimization of the model can be achieved by incorporating UNet with MLP and employing various weight configurations.

Limitations arise from RecolorNeRF's exclusive focus on overall color editing rather than instance-level editing. The decomposition process relies solely on colors, potentially causing two distinct objects with similar colorations to be amalgamated into a single layer. A prospective enhancement for RecolorNeRF involves integrating semantics into the layer decomposition process to address this limitation.

Method	PSNR ↑	SSIM ↑	LLIPS ↓
TensoRF with AlexNet	15.463	0.8192	**0.1857**
Optimised TensoRF with AlexNet	17.285	0.8467	0.1756
TensoRF with SqueezeNet	16.932	**0.8551**	0.1917
Optimised TensoRF with SqueezeNet	**18.134**	0.8004	0.1960

Table 1. Qualitative Analysis. Comparisons of view synthesis between TensoRF backbone and TensoRF optimized backbone on Custom Dataset

Method	PSNR ↑	SSIM ↑	LLIPS ↓
TensoRF with AlexNet	32.833	0.9621	0.4698
Optimised TensoRF with AlexNet	32.95	0.9627	0.3565
TensoRF with SqueezeNet	33.02	0.9631	0.3397
Optimised TensoRF with SqueezeNet	33.134	0.9635	0.2780

Table 2. Qualitative Analysis. Comparison of view synthesis between TensoRF backbone and TensoRF optimized backbone on Lego Dataset

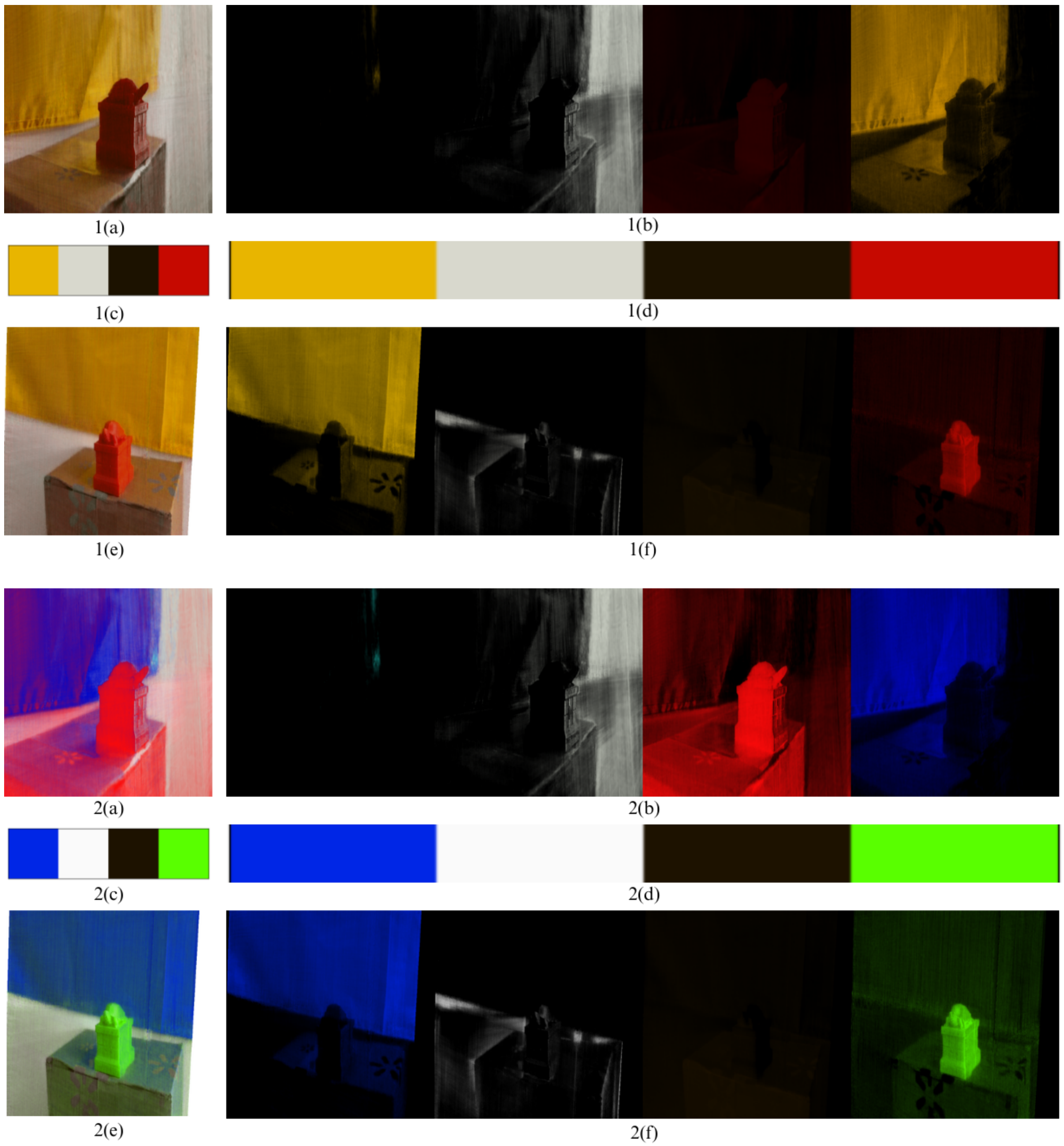


Figure 7. **Terp Dataset Color Editing Results**. 1(a) and 1(e) are input reference images before editing. 1(c) and 1(d) are the optimized palette for the input images. 1(b) and 1(f) are the Layer Decomposed Radiance Fields generated for the input by RecolorNeRF [13] and Our method respectively. 2(a) and 2(e) are output images. 2(c) and 2(d) are the desired edited color palettes. 2(b) and 2(f) are the Layer Decomposed Radiance Fields generated for the edited palette by RecolorNeRF [13] and Our method respectively.

7. Individual Contributions

Kiran Suvas Patil: Setup environment, Trained and evaluated recoloNeRF on baseline datasets, Generated Palette for evaluation, Report, and Presentation.

Vyshnav Achuthan: Worked on UNeT + MLP render model, Generated Custom Dataset, and Implemented DPT Lite to extract depth, Report and Presentation.

References

[1] Francisco Massa Adam Lerer James Bradbury Gregory Chanan Trevor Killeen Zeming Lin Natalia Gimelshein Luca Antiga Adam Paszke, Sam Gross. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems 32 (2019)*, 2019. 6

[2] Yağiz Aksoy, Tunç Ozan Aydin, Aljoša Smolić, and Marc Pollefeys. Unmixing-based soft color segmentation for image manipulation. *ACM Trans. Graph. 36, 4, Article 61c*, page 19, 2017. 1

[3] Ibraheem Alhashim and Peter Wonka. High quality monocular depth estimation via transfer learning. *arXiv e-prints*, abs/1812.11941, 2018. 6

[4] Xiaobo An and Fabio Pellacini. Appprop: all-pairs appearance-space edit propagation. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008. 2

[5] Mingming He Dongdong Chen Jing Liao Can Wang, Menglei Chai. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. *Computer Vision and Pattern Recognition*, 2021. 2

[6] Ying Cao, Antoni B. Chan, and Rynson W. H. Lau. Mining probabilistic color palettes for summarizing color use in artwork collections. *SIGGRAPH Asia 2017 Symposium on Visualization (Bangkok, Thailand) (SA '17)*, 2017. 3

[7] Anpei Chen, Zexiang Xu, Andreas Geiger Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. *arXiv:2203.09517v*, 2022. 2, 3, 4, 5

[8] Yaosen Chen, Qi Yuan, Zhiqiang Li, Yuegen Liu Wei Wang Chaoping Xie, Xuming Wen, , and Qien Yu. Upst-nerf: Universal photorealistic style transfer of neural radiance fields for 3d scene. *arXiv preprint arXiv:2208.07059*, 2022. 2

[9] Adrien Bousseau Maneesh Agrawala Christian Richardt, Jorge Lopez-Moreno and George Drettakis. Vectorising bitmaps into semi-transparent gradient layers. *Computer Graphics Forum 33, 4 (2014), 11–19.*, 2014. 3

[10] Julie Delon, Agnes Desolneux, Jose Luis Lisani, and Ana Belen Petro. Automatic color palette. *IEEE International Conference on Image Processing 2005, Vol. 2. II–706*, 2005. 3

[11] Zunlei Feng, Wolong Yuan, Chunli Fu, Jie Lei, and Mingli Song. Finding intrinsic color themes in images with human visual perception. *Neurocomputing 273 (2018), 395–402*, 2018. 3

[12] Sara Fridovich-Keil, Alex Yu, Tancik M, Q. Chen, B. Recht, and A Kanazawa. Plenoxels: Radiance fields without neural network. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[13] Bingchen Gong, Yuehao Wang, Xiaoguang Han, and Qi Dou. Recolornerf - layer decomposed radiance fields for efficient color editing of 3d scenes. *arXiv preprint arXiv:2301.07958*, 2023. 1, 5, 8

[14] Yiming Liu Stephen DiVerdi Huiwen Chang, Ohad Fried and Adam Finkelstein. Palette-based photo recoloring. *ACM Trans. Graph. 34, 4, Article 139 (Jul 2015), 11 pages*, 2015. 3

[15] Daniel Sýkora Jianchao Tan, Marek Dvorožňák and Yotam Gingold. Decomposing time-lapse paintings into layers. *ACM Trans. Graph. 34, 4, Article 61 (Jul 2015), 10 pages*, 2015. 3

[16] Jyh-Ming Lien Jianchao Tan and Yotam Gingold. Decomposing images into layers via rgb-space geometry. *ACM Trans. Graph. 36, 1, Article 7 (nov 2016), 14 pages*, 2016. 3

[17] Abiramy Kuganesan, Shih yang Su, James J. Little, and Helge Rhodin. Unerf: Time and memory conscious u-shaped network for training neural radiance fields. *arXiv:2206.11952v1*, 2022. 4

[18] Tao Ju Shi-Min Hu Kun Xu, Yong Li and Tian-Qiang Liu. Efficient affinity-based edit propagation using k-d tree. *ACM Trans. Graph. 28, 5 (Dec 2009)*, 2009. 2

[19] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016. 6

[20] Kai-En Lin, Lin Yen-Chen, Wei-Sheng Lai, Tsung-Yi Lin, Yi-Chang Shih, and Ravi Ramamoorthi. Vision transformer for nerf-based view synthesis from a single input image. *WACV 2023*, 2023. 4

[21] Sharon Lin and Pat Hanrahan. Modeling how people extract color themes from images. *SIGCHI Conference on Human Factors in Computing Systems (Paris, France) (CHI '13). Association for Computing Machinery, New York, NY, USA, 3101–3110*, 2013. 3

[22] L. Liu, J. Gu, K.Z. Lin, T.S. Chua, and C. Theobalt. Neural sparse voxel fields neurips. 2020. 2

[23] Steven Liu, Xiuming Zhang, Zhoutong Zhang, Richard Zhang, Jun-Yan Zhu, and Bryan Russell. Editing conditional radiance fields. 2021. 2

[24] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines, 2019. 5

[25] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren N. Nerf: Representing scenes as neural radiance fields for view synthesis. *ECCV*, 2020. 2, 4, 5

[26] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting triangular 3d models, materials, and lighting from images. 2022. 2

[27] Fabio Pellacini and Jason Lawrence. Appwand: editing measured materials using appearance-driven optimization. *ACM Trans. Graph. 26, 3 (Jul 2007)*, 2007. 2

[28] Francois Pitie, Anil C Kokaram, and Rozenn Dahyot. N-dimensional probability density function transfer and its application to color transfer. *Tenth IEEE International Conference on Computer Vision (ICCV'05)*, 2005. 3

[29] Albert Pumarolaz, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scene. 2020. 2

[30] Hanqiu Sun Qing Zhang, Chunxia Xiao and Feng Tang. Palette-based image recoloring using color decomposition optimization. *IEEE Transactions on Image Processing 26, 4 (April 2017)*, 2017. 3

[31] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ArXiv preprint*, 2021. 6

[32] Erik Reinhard, Michael Adhikhmin, Bruce Gooch, and Peter Shirley. Color transfer between images. *IEEE Computer Graphics and Applications 21, 5 (July 2001), 34–41*, 2001. 3

[33] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 4

[34] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 6

[35] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Generative radiance fields for 3d-aware image synthesis. *arXiv preprint arXiv:2007.02442*, 2020. 2

[36] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[37] Christoph Schied Thomas Müller, Alex Evans and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans.Graph. 41, 4, Article 102 (Jul 2022)*, 2022. 4, 6

[38] Yili Wang, Yifan Liu, and Kun Xu. An improved geometric approach for palette-based image decomposition and recoloring. *Computer Graphics Forum*, pages 11–22, 2019. 1

[39] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. *arXiv preprint arXiv:2103.14024*, 2021. 2

[40] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: Geometry editing of neural radiance fields. 2022. 2