Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Magistrale in
Computer Engineering

CombiNeRF: a Combination of Regularization Techniques for Few-Shot Neural Radiance Field View Synthesis

Advisor:
Prof. Alberto Pretto

Student:
Luigi Sarrocco

Co-Advisor:
Dr. Matteo Bonotto

ID number:
2020634

Company Advisor:
Dr. Marco Imperoli

Academic year 2022/2023
Graduation date 05/09/2023

Thanks

First of all, I want to thank Alberto Pretto, Matteo Bonotto, Daniele Evangelista and Marco Imperoli for the support they gave me, for the time they dedicated to me and for the peace of mind they transmitted during this work. Thank you for contributing to all the satisfactions achieved.

A huge thanks to Giada, for the big support and for being there when I needed it most. Thank you for being proud of me even in problematic times and for motivating me to go on. Your presence was fundamental.

A special thanks to my family, in particular my parents and my sister. Despite the difficulties encountered, they were able to encourage me and put up with me, managing to keep a good family atmosphere.

I want to thank my friends with whom I could share experiences that allowed me to get my mind off. Thanks to all the people at the pizzeria, where I worked for almost the entire duration of my studies at the University of Padua, with whom I had a really good time. Thanks also to all the people in the IAS-Lab, where I spent most of my work time.

Abstract

Neural Radiance Fields (NeRFs) have shown impressive results for novel view synthesis when a sufficiently large amount of views are available. When dealing with few-shot settings, i.e. with a small set of input views, the model could overfit those views, leading to artifacts and geometric and chromatic inconsistencies in the resulting rendering. Regularization is a valid solution that helps NeRF generalization. On the other hand, most of the recent NeRF regularization techniques aim each one to mitigate a specific rendering problem.

Starting from this observation, in this thesis we propose CombiNeRF, a framework that synergically combines several regularization techniques, some of them novel, in order to unify the benefits of each. In particular, we regularize single and neighboring rays distribution and we add a smoothness term to regularize near geometries. After these geometric approaches, we propose to exploit Lipschitz regularization on both NeRF density and color networks and to use encoding masks for input features regularization.

We show that CombiNeRF outperforms the state-of-the-art methods with few-shot settings in several publicly available datasets. We also present an ablation study on the LLFF and Nerf-Synthetic datasets that supports the choices made. The research activity performed within this thesis has been submitted to a major conference in 3D and Computer Vision.

Abstract

Neural Radiance Fields (NeRFs) hanno mostrato risultati impressionanti per la sintesi di immagini inedite quando è inizialmente disponibile un numero sufficientemente elevato di immagini. Nel caso "few-shot", ossia quando viene fornito un numero limitato di immagini di input, il modello potrebbe adattarvisi eccessivamente, causando artefatti e incoerenze geometriche e cromatiche nel rendering risultante. Forme di regolarizzazione possono essere introdotte per mitigare questi problemi. A questo scopo, recenti tecniche di regolarizzazione NeRF sono state introdotte per evitare la convergenza del modello in soluzioni incoerenti.

Partendo da questa osservazione, in questa tesi proponiamo CombiNeRF, un framework che combina sinergicamente diverse tecniche di regolarizzazione, alcune delle quali nuove, al fine di unificarne i benefici. In particolare, regolarizziamo la distribuzione dei raggi e aggiungiamo un termine di smoothness per favorire la generazione di superfici lisce. Oltre questi approcci geometrici, proponiamo di sfruttare la regolarizzazione di Lipschitz sia per la rete NeRF di densità che del colore e di utilizzare maschere di codifica per la regolarizzazione delle features in input alla rete.

Successivamente, mostriamo come CombiNeRF supera i metodi allo stato dell'arte con impostazione few-shot in diversi datasets disponibili pubblicamente. Presentiamo anche un'analisi ulteriore sui datasets LLFF e Nerf-Synthetic che supporta le scelte fatte. L'attività di ricerca condotta in questa tesi è stata proposta ad una delle principali conferenze nel campo 3D e Computer Vision.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

Starting from a set of images together with their corresponding known camera positions, novel view synthesis problem aims to reconstruct novel views, i.e. taken from different points of view with respect to the input ones. Several classes of approaches addressed this problem and made significant progress in predicting geometry and appearance representations from seen images. Novel view synthesis methods use mesh-based representation of the scene [3, 4, 5, 6], light fields [7, 8, 6] or image-based rendering [4, 9, 10, 5, 11]. Another class of approaches uses volumetric representations since they are well-suited for gradient-based optimization. Previous methods [12, 13, 14] worked directly on the voxel grids, while more recent approaches [15, 16, 17, 18, 19, 20, 21, 22] used large datasets to train deep networks. The main problem with these volumetric techniques is the expensive storage required. With the increasing attention on deep learning, other methods in this field have been introduced: NeRF [23] has gained significant attention.

Neural Radiance Field [23] has emerged as a powerful approach for scene reconstruction and photorealistic rendering from a sparse set of 2D images. By leveraging a multilayer perceptron (MLP) to model the volumetric scene representation, NeRF can generate high-quality novel views of scenes. As the MLP networks are queried multiple times for each pixel to be rendered, a lightweight architecture allows to dramatically speed up the whole rendering pipeline. The multiresolution hash encoding proposed in [1] allows for example to improve scene reconstruction efficiency using a shallow MLP without any loss of visual accuracy. However, to achieve high accuracy in its predictions and avoid artifacts, NeRF often relies on a large number of images, which can be impractical in real-world scenarios.

Regularization is a crucial tool for improving the visual fidelity of rendered

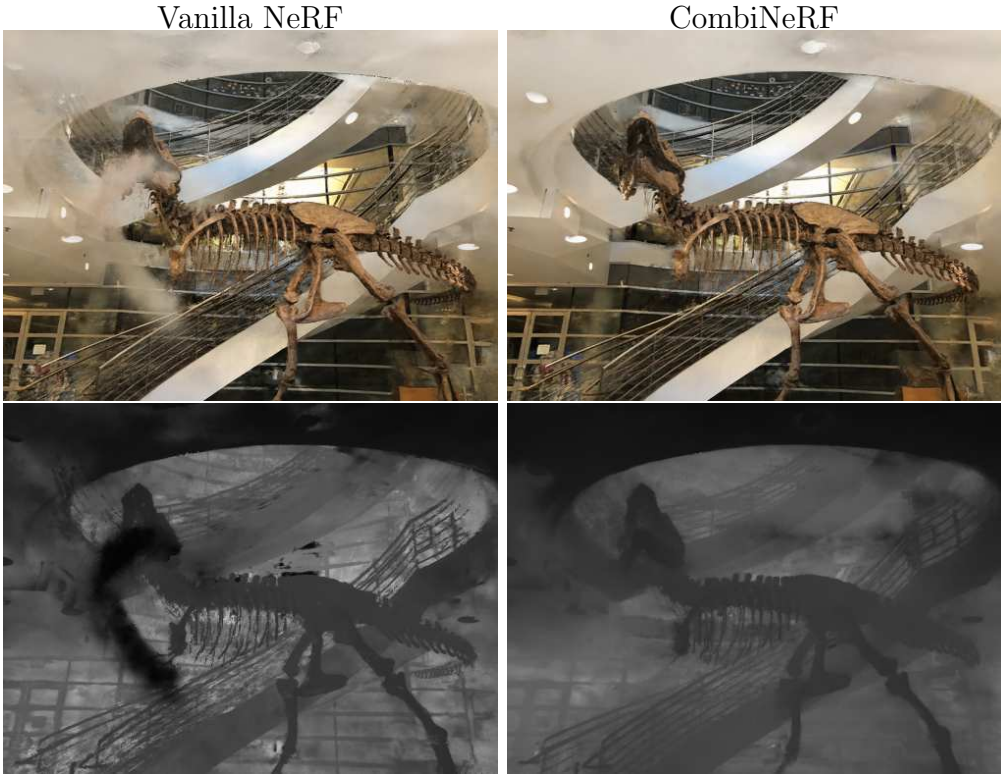


Figure 1.1: The figure shows how the proposed CombiNeRF achieves better results in terms of rendering and reconstruction quality in few-shot settings compared with the Vanilla NeRF [1, 2].

images, ensuring more coherent predictions. Several works [24, 25, 26, 27] have proposed to constrain the MLP network during training by adding regularization terms in the final loss. Instead, Lipschitz regularization [28, 29], is applied directly on network weights to enforce smoothness by controlling the rate of change of the network's outputs concerning its inputs.

In this work, we empirically select a combination of loss components to constrain the MLP network during the training phase. Firstly, (a) we modify the information-theoretic approach of InfoNeRF [25] that includes regularization of neighboring rays distributions, (b) we adopt a smoothness term to regularize near geometries, and (c) we regularize single ray distributions in order to have a specific hitting point on the scene, that reflects real-world scenarios. Additionally, as in [30], we add an encoding mask that encourages the learning of finer details only during the latest stages of the training phase. Finally, we impose Lipschitz regularization on both color and density networks, observing substantial improvements in the overall reconstruction and rendering quality and showing how NeRF benefits from Lipschitz regularization. CombiNeRF avoids the need for pre-training required by similar approaches while showing promising improve-

ments over the state of the art.

In summary, we present the following contributions:

- we propose a modification of the KL-Divergence loss proposed in InfoNeRF,
- we select a combination of regularization losses, available on a unified framework, and we validate its effectiveness through ablation studies,
- to our knowledge we are the first to impose Lipschitz regularization on all the network layers on NeRF, recording a performance increase in all the tested scenarios,
- an extensive performance evaluation on public datasets, showing that CombiNeRF achieves state-of-the-art (SOTA) performance in few-shot setting,
- the research activity performed within this thesis has been submitted to a major conference in 3D and Computer Vision.

In Chapter 2, a theoretical background is presented including preliminaries about Neural Radiance Fields, the multiresolution hash encoding used in our framework, and few-shot rendering, together with other related works in this field. Our method is described in Chapter 3, combining several regularization techniques, while in Chapter 4 are shown the experiments done with different datasets, including comparisons with other methods and an ablation study. After the conclusion in Chapter 5, additional implementations used for preliminary analysis during our work are described in Appendix A.

Chapter 2

Theoretical Background

In this chapter, we provide a preliminary theoretic background of Neural Radiance Fields together with the multiresolution hash encoding adopted in our method to speed up the training process. The few-shot concept will be introduced since our final goal is to propose an efficient method when only a few input images are available.

Related work in this field will be taken into account, including multiple techniques of regularization adopted during the training process, both in the form of additional loss terms and MLP network structure.

2.1 Preliminaries

2.1.1 Neural Radiance Field

Neural Radiance Field addressed the novel view synthesis problem by representing and optimizing a continuous scene through a simple MLP network, as shown in Fig. 2.1. For each input 3D point and view direction, it outputs the density σ of the point and its color value $\mathbf{c}$. Colors of the rays are rendered through classical volume rendering: σ can be interpreted as the probability of a ray hitting an infinitesimal particle at the specified 3D point location. Given the near and far bounds t_n and t_f, the expected color $\hat{C}(\mathbf{r})$ of camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ is

$$\begin{aligned} \hat{C}(\mathbf{r}) &= \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt, \\ \text{where } \ T(t) &= \exp\left(-\int_{t_n}^{t_f} \sigma(\mathbf{r}(s))ds\right) \end{aligned} \tag{2.1}$$

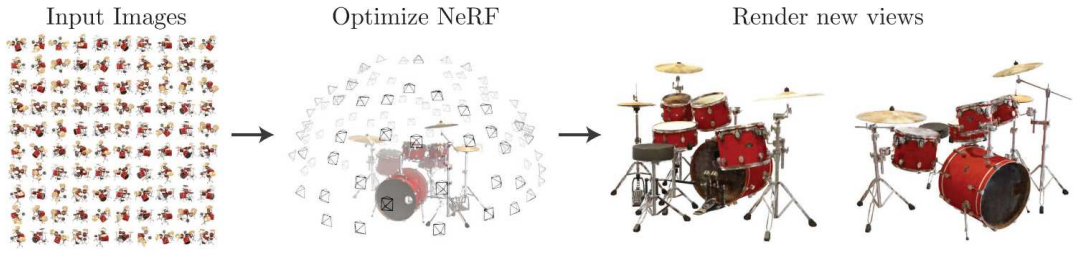


Figure 2.1: NeRF optimizes a continuous neural radiance field representation of a scene. Starting from a set of input views, it exploits an MLP network to encode the scene and it uses classical volume rendering techniques to accumulate samples along the rays in order to render the entire scene from any viewpoint. Together with the input views, it renders novel unseen views from the optimized representation.

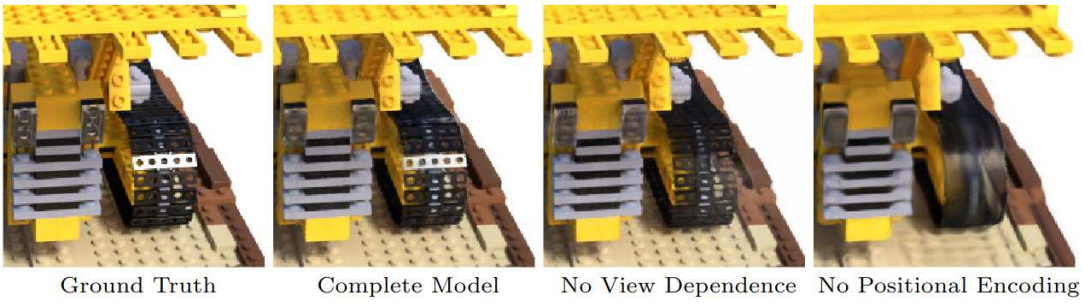


Figure 2.2: NeRF benefits from the high-frequency positional encoding used to map the input of the MLP network. The complete model uses positional encoding in both input position and view direction.

is the accumulated transmittance. In order to estimate this integral, for each pixel, points are sampled on their corresponding camera ray through a stratified sampling approach. $[t_n, t_f]$ is partitioned into N evenly-spaced bins and each sample is drawn uniformly at random within each bin:

$$t_i \sim \mathcal{U}\left[t_n + \frac{i-1}{N}(t_f - t_n), t_n + \frac{i}{N}(t_f - t_n)\right]. \tag{2.2}$$

Using stratified sampling instead of deterministic quadrature enables a continuous scene representation even if the rays are discretized, because points are not sampled in fixed locations. These samples are fed to the MLP network to render the approximated color

$$C(\mathbf{r}) = \sum_{i=1}^{N} T_i \left(1 - \exp(-\sigma_i \delta_i)\right) c_i, \tag{2.3}$$

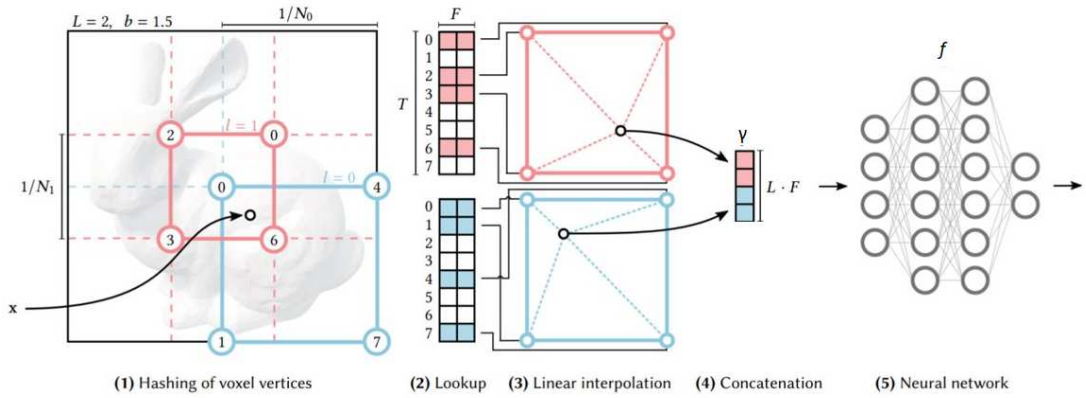


Figure 2.3: Representation of Multiresolution Hash Encoding in 2D.

where $\delta_i = t_{i+1} - t_i$ is is the distance between the i^{th} point and its adjacent sample, N is the number of samples and the transmittance T_i is computed as

$$T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right). \tag{2.4}$$

The rendered pixel color $C(\mathbf{r})$ can be expressed in terms of a weighted sum of the color values $C(\mathbf{r}) = \sum_{i=1}^{N} w_i c_i$ where $w_i = T_i \alpha_i$ with $\alpha_i = (1 - \exp(-\sigma_i \delta_i))$. The final loss for network training is the total squared error between the rendered and true pixel colors:

$$\mathcal{L}_{RGB} = \sum_{\mathbf{r} \in \mathcal{R}} \|C(\mathbf{r}) - C^*(\mathbf{r})\|_2^2. \tag{2.5}$$

Instead of dealing directly with 3D point positions and view directions, NeRF proposes to use a positional-encoding function g to map the input of the MLP network to a high-dimensional space. This encoding exploits sinusoidal functions with different frequencies:

$$g(p) = (\sin(2^0 \pi p), \cos(2^0 \pi p), \ \ldots \ , \sin(2^{V-1} \pi p), \cos(2^{V-1} \pi p)), \tag{2.6}$$

where the parameter V controls the maximum encoded frequency. The g function is applied to the input position and the view direction separately. As shown in Fig. 2.2, this encoding enables to better reconstruct high-frequency geometries and details.

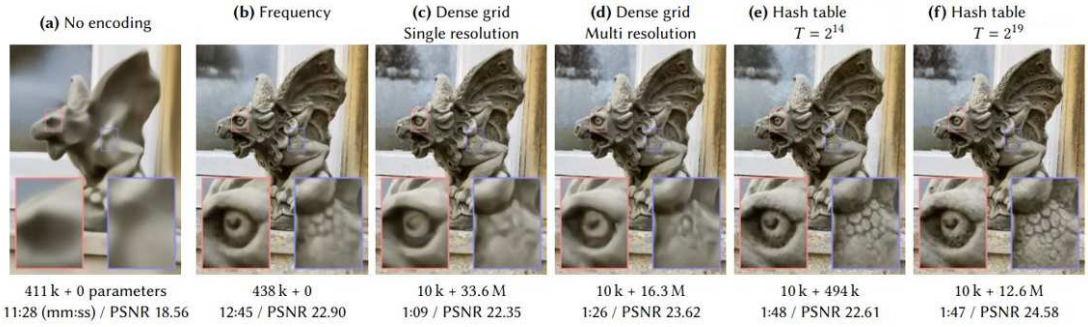


Figure 2.4: Comparisons between different encodings and structures for feature embeddings. "Frequency" is defined as the positional encoding used in NeRF, while in the hash table the hyper-parameter T is defined as its size.

2.1.2 Multiresolution Hash Encoding

Multi-layer perceptrons have been shown the need for high-dimensional representations to capture high-frequency details. In NeRF, a sine-based positional encoding (frequency encoding) is used to map the 3D point position to a high-dimensional feature vector. Other parametric encodings propose to arrange additional trainable parameters in a data structure, such as a grid or a tree. Instead of having a single dense grid, which allocates many features for both empty space and surface areas, other parametric encodings applied to neural approaches propose multiresolution grids maintaining a similar reconstruction quality while reducing the number of parameters required.

Based on these ideas, the multiresolution hash encoding proposed in [1] was able to manage the trade-off between good high-frequency quality details of the reconstructions and memory capacity. In Fig. 2.4, Instant-NGP shows how feature embeddings are essential and how different configurations differ in quality, time, and memory.

The multiresolution hash encoding adopts multi-resolution grids, whose resolutions are chosen between the coarsest and finest ones based on a growth factor b, achieving a similar result while significantly accelerating training and inference times against NeRF. In Fig. 2.3, as illustrated in Instant-NGP, this is accomplished by first identifying the cell of the grid containing the point and then assigning the result of the spatial interpolation of the features defined on the corners. As a result, a lightweight MLP can be employed to process the encoded points. Given a input point $x \in \mathbb{R}^3$, its feature vector $\gamma(x) \in \mathbb{R}^{FL}$ can be expressed as

$$\gamma(x) = [\gamma_{1,1}(x), ..., \gamma_{F,1}(x), ..., \gamma_{F,L}(x)] \tag{2.7}$$

where F is the number of feature dimensions per entry and L is the number of levels.

To minimize memory usage, each grid corner index is mapped into a hash table that holds its corresponding feature vector. Collisions are limited to finer levels where $(N_l + 1)^3 > E$, N_l and E being the resolution at level l and the maximum number of entries in the hash maps respectively, and they are handled through gradient averaging. This ensures that the results maintain their qualitative comparability to using solely a multi-resolution grid without hash encoding.

2.1.3 Few-Shot View-Rendering

Novel view synthesis for Neural Radiance Fields is a challenging problem when dealing with few-shot settings, i.e. when only a small set of input views are available. Starting from a few images representing the scene means to have drastically less information available. For this reason, NeRF struggles to reconstruct unseen viewpoints, in particular for those distant enough from the inputs.

Previous work tried to address this problem, using pre-trained models in order to overcome the lack of input views but they require the collection of additional data and they introduce a non-negligible overhead. In [24] a normalizing flow model is introduced and a denoising diffusion model (DDM) is used in [27]. Prior knowledge from a pre-trained image encoder is adopted in [31], while other methods require depth supervision. Instead, CombiNeRF does not rely on external information or models and it manages to deal with few-shot using multiple forms of regularization techniques.

2.2 Related Work

Novel view synthesis aims to generate images from different viewpoints by utilizing a collection of pre-existing views [32, 33, 34, 35, 36]. Among the novel view synthesis solutions, NeRF emerged as a result of several factors as the compactness offered by the underlying structure, its domain-agnostic nature and the impressive visual quality offered. Subsequent works managed to improve over the fidelity of rendered images [37, 38, 39, 40, 41], reduce required time [42, 43, 44, 45] and extend NeRF capabilities and application domain [46, 47, 48, 49, 50, 51].

2.2.1 NeRF Regularization

NeRF models struggle to render novel high-quality images when supervised with only a few input views during training. To reduce the impact of the artifacts introduced by overfitting on training samples, previous work focused on adding loss terms to provide additional constraints to the model.

In InfoNeRF [25], the entropy of the rays and the KL-divergence between the distribution of neighboring rays are both minimized in order to restrict the range of the prediction of high-density value on the object surface. Similarly, in Mip-NeRF 360 [26], distortion loss was introduced to consolidate weights into as small a region as possible when rays intersect a scene object. In RegNeRF and DiffusioNeRF [24, 27], rendered patches that are less likely to appear, according to a determined image distribution, are penalized. In PixelNerf [52], prior knowledge acquired from different scenes and global context information are both leveraged by concatenating image features extracted by a Convolutional Neural Network Encoder with the input 3D positions.

Depth regularization is considered as a loss term in RegNeRF [24], while DS-NeRF improves quality reconstruction by adding supervision on the depth. Also in [53] a loss term is introduced to regularize rendered depth maps with depths estimated using Structure-from-Motion.

DietNeRF [31] introduced a semantic loss term by extracting semantic representation of renderings using the CLIP Vision Transformer [54] and maximizing the similarity with the representations of the ground truth views. In Aug-NeRF [55], noise is injected into input, feature, and output during training to improve the generalization capabilities.

We take into consideration some of these forms of regularization, combining their benefits during the training phase, ensuring better generalization and reconstruction of the scene with few input views.

2.2.2 Neural Surface Reconstruction

Implicit functions like signed distance functions (SDFs) [56, 57, 58, 59] and occupancy maps [60, 61] are best fitted for representing objects on the scene with a defined surface geometry. Both NeuS and HF-NeuS [62, 63] reparametrize the rendering equation used in NeRF to exploit SDFs properties. In order to increase the training and inference speed and facilitate the learning of high-frequency details, Instant-NSR and Neuralangelo [64, 30] exploit the hash grid encoding

proposed in Instant-NGP [1], while PermutoSDF [65] employs a permutohedral lattice to decrease the memory accesses. To recover smoother surfaces in reflective or untextured areas, Neuralangelo and PermutoSDF add a curvature term loss. Additionally, Neuralangelo discards values from finer levels of the hash grid encoding to encourage the learning of coarse details on the first training iterations. To prevent the color network overfitting due to the injection of the curvature loss term, PermutoSDF employs the Lipschitz constant regularization method proposed in [29].

In our work, we exploit both Lipschitz regularization, leveraged in neural surface reconstruction to avoid geometry over-smoothness, and encoding mask, finding out they provide additional benefits in the few-shot setting.

Chapter 3

Method

A graphical overview of the CombiNeRF method proposed in this work is given in Fig. 3.1. The proposed approach combines multiple regularization techniques in a unified and flexible framework that helps NeRF generalization in few-shot scenarios. In this section, all the combined losses will be described together with the Encoding Mask and Lipschitz regularization techniques. For each technique, our improvements and modifications will be described in detail to better highlight the main contributions of this work.

During the empirical selection of the best implementations used in CombiNeRF, we considered three additional regularization techniques: Sample Space Annealing [24], Adversarial Perturbation [55] and Ray Entropy loss [25]. Such techniques have not been included in CombiNeRF because of their poor contribution. We refer to Appendix A for a detailed description of these techniques together with the results obtained during our preliminary experiments.

3.1 Regularization Losses

When the model is prone to overfit, regularization losses are introduced to limit its capacity and to lead to more realistic solutions, with the further benefit of not increasing the training time. These additional losses are added to the main loss function, in our case to the photometric loss $\mathcal{L}_{RGB}$.

In this section, we regularize ray distributions considering the Distortion loss, the Full Geometry loss and a modification of the KL-Divergence loss. We also take into account the Depth Smoothness loss in order to encourage smooth surfaces.

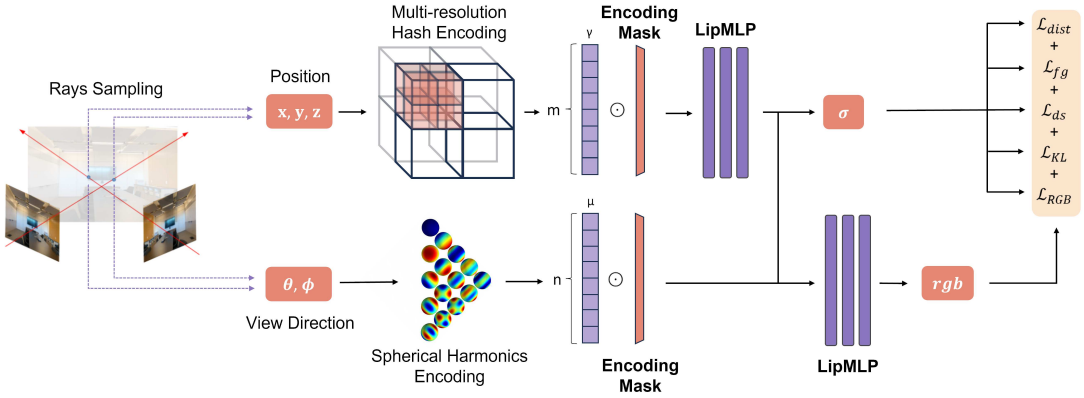


Figure 3.1: Overview of the CombiNeRF framework. We sample 3D points over a batch of rays passing through the scene. Position and view direction are respectively encoded through Multi-resolution Hash Encoding and Spherical Harmonics and fed to the Lipschitz network (LipMLP) after being masked. Networks' outputs are used by volumetric rendering for estimating the expected color C and depth d of each ray, while different loss terms are computed to regularize the training process. CombiNeRF combines *i*) all these regularization losses, *ii*) the Lipschitz network instead of the original MLP, *iii*) the Encoding Mask approach used for masking the networks' input.

3.1.1 KL-Divergence Loss

When dealing with a few training views, the standard models [23, 1] struggle to generalize to unseen views. As 3D scenes exhibit piece-wise smooth surfaces, we enforce similar distributions of weight values among neighboring rays. Differently from [25], we compute the ray density $p(\mathbf{r})$ on weights w_i, instead of alpha values α_i:

$$p(\mathbf{r}_i) = \frac{w_i}{\sum_j w_i} = \frac{T_i \alpha_i}{\sum_j T_j \alpha_j}. \tag{3.1}$$

Given an observed ray $\mathbf{r}$, we sample another neighboring ray $\hat{\mathbf{r}}$ minimizing the KL-Divergence (D_{KL}) between their weight distributions. The corresponding loss is defined as follows:

$$\mathcal{L}_{KL} = D_{KL}\Big(P(\mathbf{r})||P(\hat{\mathbf{r}})\Big) = \sum_{i=1}^{N} p(\mathbf{r}_i) \log \frac{p(\mathbf{r}_i)}{p(\hat{\mathbf{r}}_i)}. \tag{3.2}$$

The choice of the neighboring ray $\hat{\mathbf{r}}$ can be done in two ways: by slightly rotating the camera pose of the ray $\mathbf{r}$ or by choosing one of the possible adjacent rays (defined as near pixels). In our implementation, we deviate from the original contribution [25] by choosing the second option. After sampling uniformly at random one of the four adjacent rays for each training ray, we compute $\mathcal{L}_{KL}$.

3.1.2 Distortion and Full Geometry Loss

A common problem of NeRF is the presence of "floaters", disconnected regions of dense space, usually located close to the camera planes, which cause the presence of blurry cloud artifacts on images rendered from novel views. Previous works [26, 27] showed that those artifacts can be removed by adding a loss term $\mathcal{L}_{dist}$ that takes into account the distances t_i and weights w_i of the N points sampled on the rays, and the depth

$$d(r) = \frac{\sum_{i=1}^{N} w_i t_i}{\sum_{i=1}^{N} w_i} \tag{3.3}$$

of each ray. Thus, the $\mathcal{L}_{dist}$ loss can be written as:

$$\mathcal{L}_{dist} = \frac{1}{d(r)} \left(\sum_{i,j} w_i w_k \left| \frac{t_i + t_{i+1}}{2} - \frac{t_j + t_{j+1}}{2} \right| + \frac{1}{3} \sum_{i=1}^{N} w_i^2 (t_{i+1} - t_i) \right), \tag{3.4}$$

where the depth factor penalizes dense regions near the camera.

The first term minimizes the weighted distances between all pairs of interval midpoints and the second term minimizes the weighted size of each individual interval. In this way, weights on the ray are encouraged to be as compact as possible by pulling distant intervals towards each other, consolidating weights into a single interval or a small number of nearby intervals, and minimizing the width of each interval.

In Eq. (3.4), all possible intervals for each ray are considered. As the total combination of all (i, j) would drastically increase the amount of memory required, to reduce the total load we consider only a subset over the total number of sampled rays on which we compute the loss.

Furthermore, a normalization loss term $\mathcal{L}_{fg}$ is considered, following [27], that encourages the weights to sum to unit, as the ray is expected to be fully absorbed by the scene geometry in real scenes:

$$\mathcal{L}_{fg} = \left(1 - \sum_{i=1}^{N} w_i \right)^2, \tag{3.5}$$

In addition, $\mathcal{L}_{fg}$ enforces $\mathcal{L}_{KL}$ by driving the model to treat the weights w_i as probabilities, $p(\mathbf{r}_i) \approx w_i$ (See Eq. (3.1)).

3.1.3 Depth Smoothness

Similarly to KL-Divergence loss, the depth smoothness constraint encourages smooth surfaces. However, while the former term requires similarity between the compared distributions, the latter term only computes the difference between the expected depth values. As in [24], given the estimated depth $d(r)$ of a pixel ray and a patch of size S_{patch}, the depth smoothness constraint is defined as follows:

$$\mathcal{L}_{ds} = \sum_{p \in P} \sum_{i,j=1}^{S_{patch}-1} \left(\big(d(r_{ij}) - d(r_{i+1j})\big)^2 + \big(d(r_{ij}) - d(r_{ij+1})\big)^2 \right), \tag{3.6}$$

where P is the set of all the rendered patches and r_{ij} is the ray passing through the pixel (i, j) of patch p.

In order to compute this loss term, it is necessary to sample rays through patches instead of sampling them at random, otherwise the depth smoothness prior would not hold anymore because rays could be far away from each other.

3.2 Encoding Mask

Input encoding allows the preserving of high-frequency details [23]. However, when only a few images are available, the network is more sensitive to noise. In this scenario, high-frequency components exacerbate the problem preventing the network from exploring more in depth low-frequency information and consequently learning a coherent geometry. As in [66, 30], we use a mask to filter out high-frequency components of the input in early iterations in order to let the network focus on robust low-frequency information.

Given the length $l = L \cdot F$ of the resulting multiresolution hash encoding given by Eq. (2.7), the mask m is defined as:

$$m = [1_1, ...1_{[\![l \cdot x]\!]}, 0, ...0_l], \tag{3.7}$$

where x is the ratio of features to keep active. The resulting feature vector given in input to the MLP network will be the element-wise product $\gamma \odot m$ between the multiresolution hash encoding of the input and the mask. The ratio x is set to keep only the coarsest features during the initial phase of the training and progressively include the remaining features.

In CombiNeRF, we apply Instant-NGP multiresolution hash encoding on input positions and sphere harmonics encoding on the view directions (see Fig. 3.1).

The progressive mask can be used on both inputs in order to filter high-frequency terms on early training iterations.

3.3 Lipschitz Network

We define $f_\Theta(x, t)$ as the function representing the implicit shape, expressed by means of a neural network, where $\Theta = \{W_i, b_i\}$ is the set of parameters containing weights W_i and biases b_i of each layer l_i of the network.

A function is called Lipschitz continuous if there exists a constant $k \geq 0$ such that:

$$\|f_\Theta(x_0) - f_\Theta(x_1)\| \leq k\|x_0 - x_1\|, \tag{3.8}$$

for all possible inputs x_0, where k is called the Lipschitz constant which bounds how fast the function f_θ can change.

We employ the regularization method proposed in [29], which was already used by PermutoSDF [65] on the color network to balance out the effect introduced by the curvature regularization applied on the SDF network. However, differently from PermutoSDF, we apply the regularization on both NeRF density and color networks, recording an increase in performance with respect to the original MLP network. Given the trainable Lipschitz bound k_i associated to layer l_i, the normalization scheme is described as:

$$\begin{aligned} y &= \sigma(\hat{W}_i x + b_i) \\ \hat{W}_i &= f_n(W_i, \ln(1 + e^{k_i})), \end{aligned} \tag{3.9}$$

where f_n is a normalization function and $\ln(1 + e^{k_i})$ enables to avoid negative values for k_i. The normalization scales each row of W_i to have an absolute value row-sum less or equal to $\ln(1 + e^{k_i})$ in order to satisfy Eq. (3.8).

3.4 Overall method

CombiNeRF combines the previously described regularization techniques regarding losses and network structure, hence the name *CombiNeRF*. Thus, we can write the final loss as:

$$\begin{aligned} \mathcal{L}_{CombiNeRF} = \mathcal{L}_{RGB} + \lambda_{dist} \cdot \mathcal{L}_{dist} + \\ \lambda_{fg} \cdot \mathcal{L}_{fg} + \lambda_{ds} \cdot \mathcal{L}_{ds} + \lambda_{KL} \cdot \mathcal{L}_{KL}, \end{aligned} \tag{3.10}$$

where λ are the hyper-parameters controlling the contribution of each loss. In addition, CombiNeRF includes Lipschitz regularization and the Encoding Mask technique. The proposed CombiNeRF offers a unified implementation of all the regularization techniques described above, outperforming current SOTA methods on few-shot scenarios as demonstrated in the following experimental chapter.

Chapter 4

Experiments

In this chapter, we show the datasets and metrics we use for the experiments, we provide relevant details about our implementation and we show quantitative and qualitative comparisons made against the state-of-the art methods. Finally, an ablation study is made to evaluate the contribution of the components that form CombiNeRF.

4.1 Setup

4.1.1 Dataset

Our experiments include the LLFF dataset [18] and the NeRF-Synthetic dataset [23] under few-shot settings. LLFF is composed of 8 complex scenes, representing real-world scenarios, with 20-62 images for each scene captured with a handled camera. Instead, NeRF-Synthetic is composed of 8 synthetic scenes with view-dependent light transport effects, in which each scene is composed of 100 training images and 200 test images. We also consider the Fox dataset, composed of 50 images.

We evaluate LLFF on 3/6/9 input views, following the protocol proposed in RegNeRF [24], while in NeRF-Synthetic we train 8 views and test 25 views following DietNeRF [31]. Fox dataset is evaluated only in the ablation study, including tests under 9 input views and also considering the full dataset.

4.1.2 Metrics

To evaluate the performance of CombiNeRF against the SOTA methods, we use 3 different metrics: peak signal-to-noise ratio (PSNR), structural similarity index

measure (SSIM) [67], and learned perceptual image patch similarity (LPIPS) [68]. We also take into consideration the geometric mean of MSE $= 10^{-\text{PSNR}/10}$, $\sqrt{1-\text{SSIM}}$ and LPIPS following [24], in order to have an easier and unified comparison.

4.1.3 Implementation Details

For our experiments, we used the torch-ngp [2] implementation of Instant-NGP [1] as base code, since from our experience it generally outperforms the original NeRF implementation [23] from both a runtime and accuracy points of view. We refer to torch-ngp as "Vanilla NeRF" in the experiments. We developed CombiNeRF on top of torch-ngp, thus obtaining a unified and unique implementation that embeds all the contributions shown in the previous sections.

In the LLFF dataset, we set $\lambda_{dist} = 0$ for the first 1000 iteration and then $\lambda_{dist} = 2 \cdot 10^{-5}$ until the end and $\lambda_{fg} = 10^{-4}$, $\lambda_{KL} = 10^{-5}$, $\lambda_{ds} = 0.1$ and $S_{patch} = 4$, while we use the encoding mask only on the density network in which x saturates after 90% of the total iterations.

In the NeRF-Synthetic dataset, we set $\lambda_{dist} = 0$ for the first 1000 iterations and then $\lambda_{dist} = 2 \cdot 10^{-3}$ until the end and $\lambda_{fg} = 10^{-3}$, $\lambda_{KL} = 10^{-5}$, $\lambda_{ds} = 0.02$ and $S_{patch} = 4$, x saturates after 20% of the total iterations. We also set the number of sampled rays for each iteration to 4096 and 7008 and the number of levels for the Instant-NGP multiresolution hash encoding to 16 and 32 for LLFF and NeRF-Synthetic, respectively.

In the Fox dataset, we set $\lambda_{dist} = 0$ for the first 1000 iteration when considering 9-view setting or 2000 when considering full dataset and then $\lambda_{dist} = 10^{-3}$ until the end and $\lambda_{fg} = 10^{-2}$, $\lambda_{KL} = 10^{-6}$, $\lambda_{ds} = 1.0$ and $S_{patch} = 4$, while we use the encoding mask only on the density network in which x saturates after 90% of the total iterations when considering full dataset or 30% of the total iterations when considering 9-view setting.

4.1.4 Training Details

For an easier reproduction of the results, we provide the training procedure adopted for the LLFF dataset, which follows [24], and for the NeRF-Synthetic dataset, which follows [31].

In LLFF we use all 8 scenarios ("Fern", "Room", "T-rex", "Flower", "Leaves", "Horns", "Orchids" and "Fortress"). Each view of each scene is 378×504, thus

	PSNR ↑			SSIM ↑			LPIPS ↓			Average ↓		
	3-view	6-view	9-view	3-view	6-view	9-view	3-view	6-view	9-view	3-view	6-view	9-view
Vanilla NeRF	17.71	22.03	24.21	0.544	0.738	0.811	0.303	0.149	0.101	0.155	0.081	0.057
RegNeRF	19.08	23.10	24.86	0.587	0.760	0.820	0.336	0.206	0.161	0.146	0.086	0.067
FreeNeRF	19.63	23.73	25.13	0.612	0.779	0.827	0.308	0.195	0.160	0.134	0.075	0.064
DiffusioNeRF	19.88	**24.28**	25.10	0.590	0.765	0.802	0.192	**0.101**	**0.084**	0.118	0.071	0.060
CombiNeRF	**20.37**	23.99	**25.15**	**0.686**	**0.805**	**0.841**	**0.191**	0.106	**0.084**	**0.101**	**0.060**	**0.049**

Table 4.1: Comparison of CombiNeRF with SOTA methods on the LLFF dataset with 3/6/9 input view few-shot settings.

8× downsampled with respect to the original resolution. Test views are taken every 8th view and input images are evenly sampled among the remaining ones.

In Nerf-Synthetic we use all 8 scenarios ("Lego", "Hotdog", "Mic", "Drums", "Materials", "Ship", "Chair" and "Ficus"). Each view of each scene is 400×400, thus 2× downsampled with respect to the original resolution. We take 25 test views evenly sampled from the original test set, while training views are chosen according to the following IDs: 2, 16, 26, 55, 73, 75, 86, 93.

In Fox we consider views in the original resolution, thus 1080×1920. Considering the full dataset, we take the first view as the validation set, from the second to the fourth as the test set and the other 46 views as the training set. Instead, when considering only 9 input views, the training set is composed of evenly sampled images starting from the previous one.

4.2 Comparison

We compare the performance achieved by CombiNeRF against the SOTA methods in the few-shot scenario. In LLFF we consider RegNeRF [24], FreeNeRF [66] and DiffusioNeRF [27], while in NeRF-Synthetic we take into account DietNeRF [31] and FreeNeRF [66]. All quantitative evaluation results for the other methods, along with images showing the qualitative results, are taken from the respective papers. If some results are not present, it means that the related paper has not reported quantitative or qualitative results on the related dataset.

4.2.1 LLFF Dataset

This dataset is a collection of complex real-world scenes where, from our experiments, we observed that Vanilla NeRF heavily overfits the training images when only a few of them are used as input. In Tab. 4.1 are shown the quantitative results under 3/6/9 training images. For 3 and 9 views, CombiNeRF outperforms the other methods in all the metrics. With 6 views, DiffusioNeRF

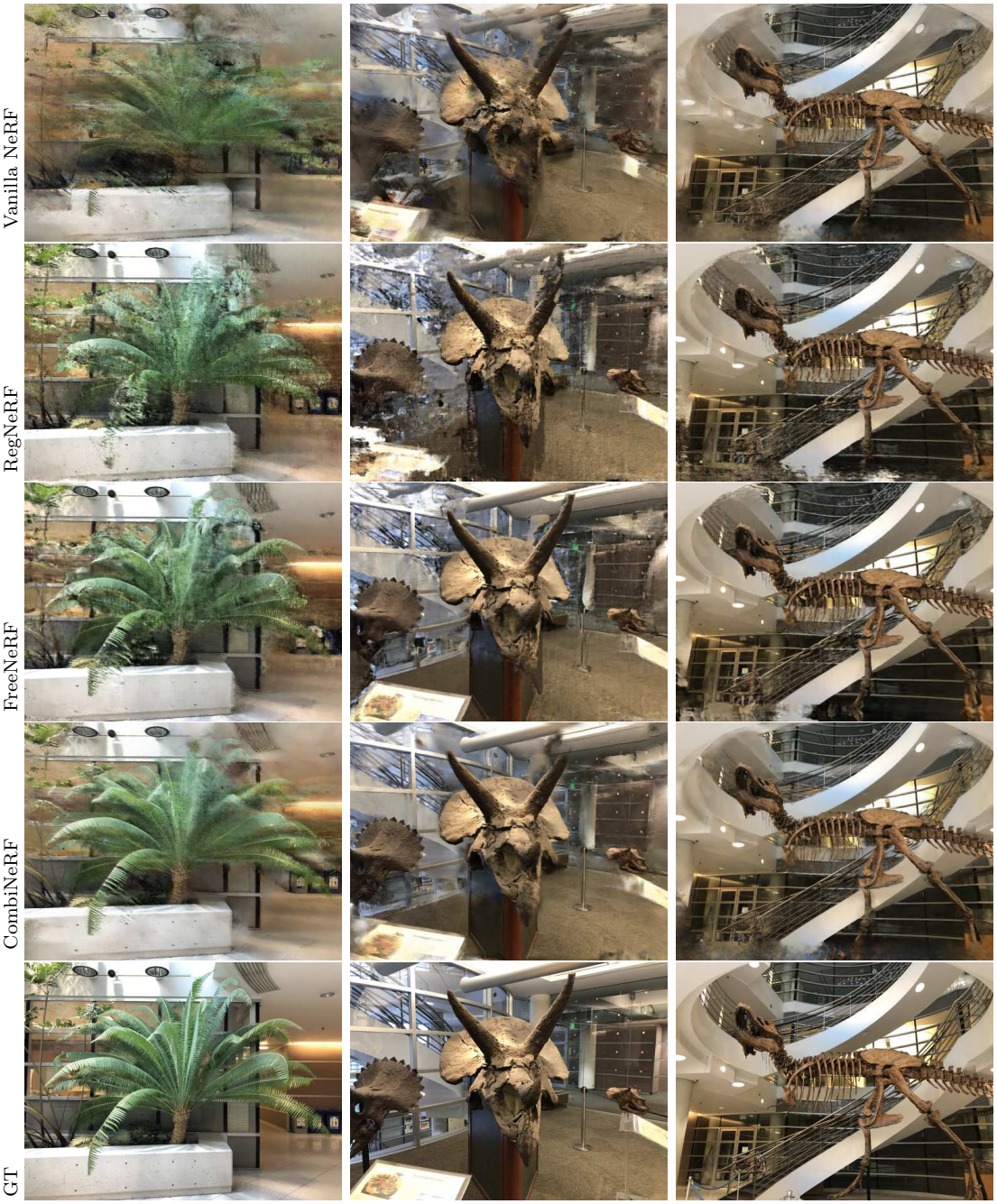
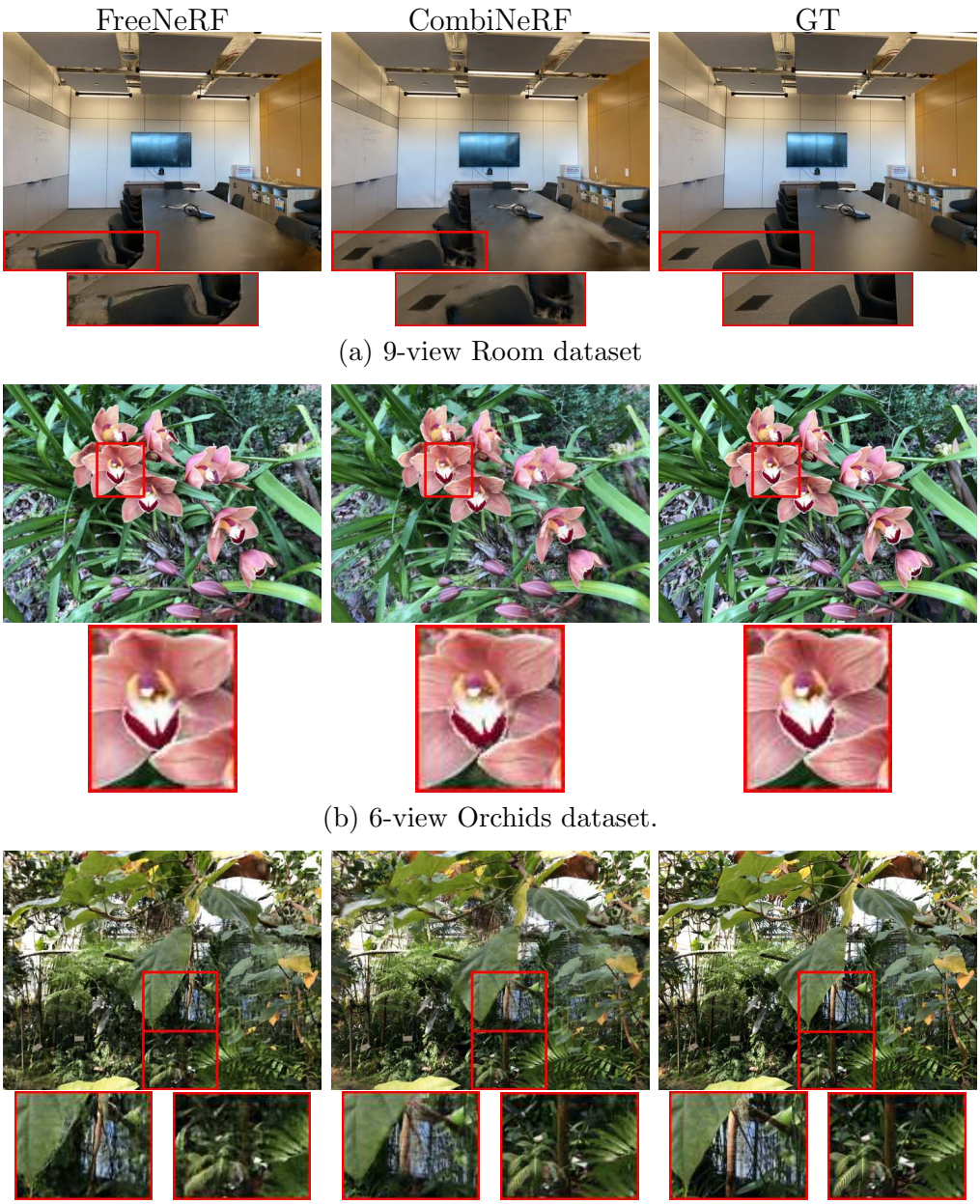


Figure 4.1: Comparison of our CombiNeRF against RegNeRF, FreeNeRF and Vanilla NeRF on Fern, Horns and T-rex scenarios with 3-view setting.

achieves a better LPIPS score, however, CombiNeRF scores higher on average. Interestingly, with more views (6 or 9), Vanilla NeRF approaches and sometimes outperforms all other methods except CombiNeRF, while its performance heavily degrades with a reduced number of views. CombiNeRF, on the other hand, shows cutting-edge and consistent performance regardless of the number of views.

Qualitative results on LLFF are shown in Fig. 4.1. CombiNeRF is able to



(a) 9-view Room dataset

(b) 6-view Orchids dataset.

(c) 3-view Leaves dataset.

Figure 4.2: In-depth comparison of CombiNeRF against FreeNeRF on some LLFF scenes with 3/6/9 input views.

reconstruct better high-frequency details maintaining the geometry of the scene. Other methods generate lots of artifacts, like in the "Fern" scene, and they struggle to reconstruct the background. In the "Horns" scene, the environment is noisy in RegNeRF and FreeNeRF while it becomes sharper in CombiNeRF. We also notice that some fine details like the stair handrail in the "T-rex" scene are preserved. In Fig. 4.2 we additionally compare CombiNeRF against FreeNeRF. In "Room" with the 9-view setting, the near part of the table gets deformed while

NeRF-Synthetic 8-views	PSNR ↑	SSIM ↑	LPIPS ↓
Vanilla NeRF	22.335	0.845	0.144
DietNeRF	23.147	0.866	0.109
DietNeRF, $\mathcal{L}_{MSE}$ ft	23.591	0.874	0.097
FreeNeRF	24.259	**0.883**	0.098
CombiNeRF	**24.394**	**0.883**	**0.088**

Table 4.2: NeRF SOTA comparison on NeRF-Synthetic dataset with 8 input views. "$\mathcal{L}_{MSE}$ ft" is the fine-tuned version of DietNeRF.

in CombiNeRF some details on the floor are still visible. In "Orchids", with the 6-view setting, CombiNeRF is able to render more high-frequency details, for example, stripes in the orchid petals (Fig. 4.2). In "Leaves" with the 3-view setting, the central leaf rendered in FreeNeRF contains several artifacts and, in general, even in this case, CombiNeRF is able to reconstruct better high-frequency details.

In Fig. 4.8, Fig. 4.9 and Fig. 4.10 are shown additional qualitative results on all scenarios of the LLFF dataset under 3-view, 6-view and 9-view settings, respectively. Under the 3-view setting, we can see the effectiveness of CombiNeRF against Vanilla NeRF in both RGB images and depth images. When the number of input images increases, the performance gap between Vanilla NeRF and CombiNeRF decreases, but the depth quality still remains definitely better in the latter. In Tab. 4.7, Tab. 4.8 and Tab. 4.9 are shown per-scene quantitative results on PSNR, SSIM, and LPIPS metrics, respectively, including 3/6/9-view settings. CombiNeRF clearly outperforms Vanilla NeRF in all the scenarios.

4.2.2 NeRF-Synthetic Dataset

This dataset contains synthetic renderings of objects exhibiting high-frequency geometric details and reflective effects, thus making this dataset particularly challenging. In Tab. 4.2 we show the quantitative results of CombiNeRF and the compared methods. We can see that CombiNeRF outperforms the other SOTA methods on the PSNR and LPIPS metrics while achieving the same result on the SSIM. CombiNeRF is also the overall best-performing method on the NeRF-Synthetic dataset, performing better on scenes like "Lego", "Mic", and "Ficus" which exhibit complex geometries, while "Materials" represents the most challenging scenario due to the presence of strong view-dependent reflection.

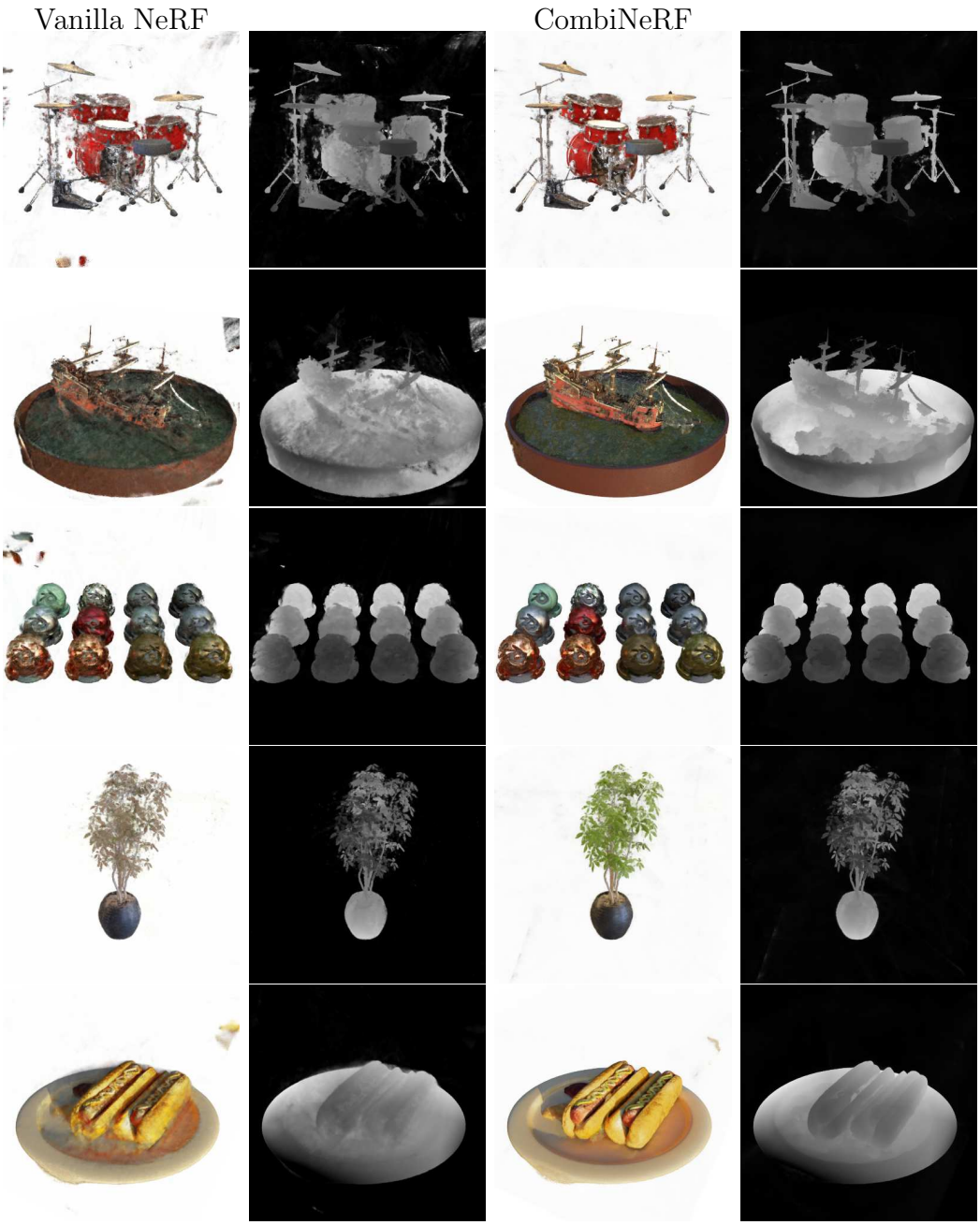


Figure 4.3: Comparison of CombiNeRF against the Vanilla NeRF method in Drums, Ship, Materials, Ficus and Hotdog scenarios.

In Fig. 4.3 we show qualitative results of CombiNeRF in different scenes. The reconstructed depth is more consistent, presenting fewer artifacts. Rendered images are far less noisy with respect to the ones rendered by the Vanilla NeRF implementation. Colors are better preserved as can be seen in the "Hotdog" and in "Ficus" scenes where the color of the leaves is more realistic and closer to the ground truth.

In Fig. 4.11 are shown additional qualitative results on all scenarios of the

NeRF-Synthetic dataset under the 8-view setting. CombiNeRF is able to better reconstruct the scenarios, this is particularly visible in the geometry of "Ship" and in the color of "Ficus". Finer details are less visible in Vanilla NeRF, as we can see in "Mic" and "Chair", while a lot of floaters and noise are removed in "Drums" and "Materials" with CombiNeRF. In Tab. 4.10, Tab. 4.11 and Tab. 4.12 are shown per-scene quantitative results on PSNR, SSIM, and LPIPS metrics, respectively. We can observe that in "Lego", "Chair", "Mic", "Hotdog" and "Ficus" CombiNeRF gets better results in all the metrics. Instead, in "Drums", "Materials" and "Ship", which are the most challenging scenarios, CombiNeRF outperforms the other methods in the overall metric scores.

4.3 Ablation Study

In this section, we assess the contribution of each of the regularization techniques used in CombiNeRF, thus highlighting their importance. Instead of showing all the possible combinations, we empirically select only the more relevant regularization techniques. We conducted an ablation study of our CombiNeRF in the 3-view and 8-view few-shot settings for the LLFF and NeRF-Synthetic datasets respectively. For these datasets, we used the same parameters as the experiments previously described. We also consider the Fox dataset both under 9-view few-shot setting and with full dataset.

4.3.1 LLFF Dataset

In Tab. 4.3 is shown the ablation on the LLFF dataset with the 3-view setting. Starting from the Vanilla NeRF solution, we notice that each contribution in CombiNeRF brings an increase in performance, with Lipschitz regularization and $\mathcal{L}_{dist} + \mathcal{L}_{fg}$ losses playing a crucial role in improving the overall performance. Applying Lipschitz regularization on both density and color networks also allows to further improve the quality of the obtained results. From the same table, we can observe that, when using the term losses $\mathcal{L}_{dist} + \mathcal{L}_{fg}$, they greatly benefit from sampling patches of rays instead of the single rays.

In Fig. 4.4 we show qualitative results of CombiNeRF using Lipschitz regularization in both sigma and color network against the only color network. The latter tentative approach is named CombiNeRF$^\sharp$ in the figures and tables. CombiNeRF is able to model in a more coherent way the geometry of the scenarios, removing also the blurry effects that characterize the renderings of the implementation

LLFF 3-views	PSNR ↑	SSIM ↑	LPIPS ↓	Average ↓
Vanilla NeRF	17.71	0.544	0.303	0.155
L	18.74	0.608	0.249	0.130
$\mathcal{L}_{dist}$+$\mathcal{L}_{fg}$	17.93	0.554	0.288	0.151
$\mathcal{L}_{dist}$+$\mathcal{L}_{fg}(S_{patch=4})$	19.10	0.608	0.249	0.128
$\mathcal{L}_{dist}$+$\mathcal{L}_{fg}$+$\mathcal{L}_{ds}$	19.50	0.626	0.237	0.122
$\mathcal{L}_{dist}$+$\mathcal{L}_{fg}$+$\mathcal{L}_{ds}$+L	20.09	0.674	0.198	0.105
$\mathcal{L}_{dist}$+$\mathcal{L}_{fg}$+$\mathcal{L}_{ds}$+L+EM	20.27	0.683	0.194	0.103
CombiNeRF$^\sharp$	19.52	0.637	0.236	0.119
CombiNeRF	**20.37**	**0.686**	**0.191**	**0.101**

Table 4.3: Ablation on LLFF dataset with 3-view setting. Lipschitz and Encoding Mask are named L and EM respectively. When using $\mathcal{L}_{ds}$, we set the patch size $S_{patch} = 4$ by default. We call CombiNeRF$^\sharp$ our method using Lipschitz only in the color network.

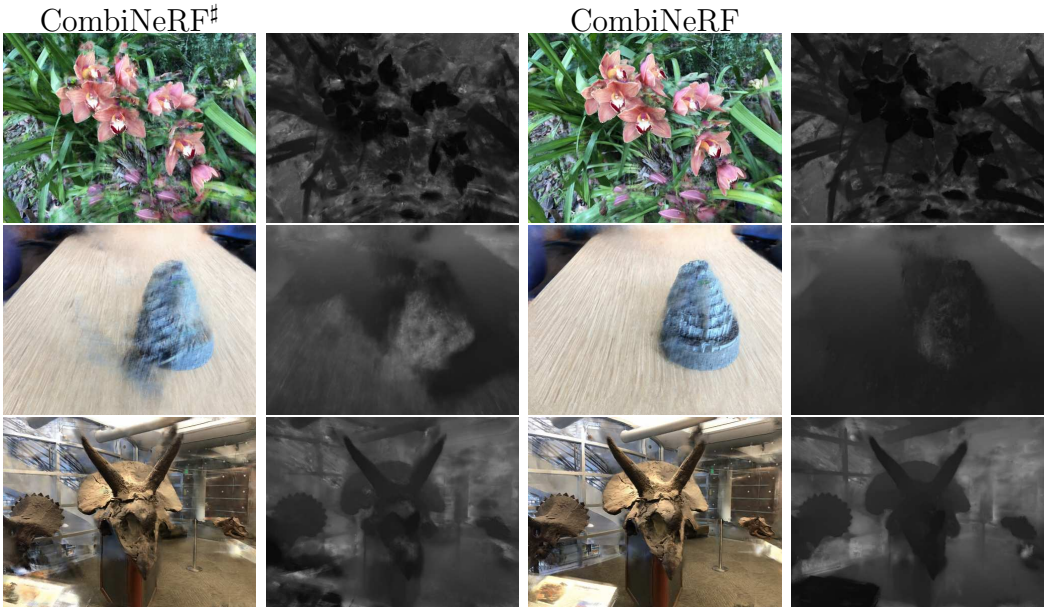


Figure 4.4: Qualitative result on the ablation study of LLFF with 3-view setting. We compare our CombiNeRF with CombiNeRF$^\sharp$ (our method using Lipschitz only in the color network).

using only Lipschitz regularization in the color network.

4.3.2 NeRF-Synthetic Dataset

As already done for the LLFF dataset, we also provide an ablation study on the NeRF-Synthetic dataset. In Tab. 4.4 is shown the result of this study on the with the 8-view setting. In this study, we additionally focused on the $\mathcal{L}_{KL}$ loss. In particular, we both tested the same implementation of the KL-Divergence

NeRF-Synthetic 8-views	PSNR ↑	SSIM ↑	LPIPS ↓	Average ↓
Vanilla NeRF	22.34	0.845	0.144	0.073
$\mathcal{L}_{dist}$+$\mathcal{L}_{fg}$+$\mathcal{L}_{ds}$+L	23.61	0.865	0.115	0.059
$\mathcal{L}_{dist}$+$\mathcal{L}_{fg}$+$\mathcal{L}_{ds}$+$\mathcal{L}^{\dagger}_{KL}$+L	22.93	0.863	0.126	0.064
$\mathcal{L}_{dist}$+$\mathcal{L}_{fg}$+$\mathcal{L}_{ds}$+L+EM	24.04	0.871	0.105	0.056
CombiNeRF†	24.36	**0.883**	**0.088**	**0.050**
CombiNeRF	**24.39**	**0.883**	**0.088**	0.051

Table 4.4: Ablation on NeRF-Synthetic dataset with 8-view setting. Lipschitz is defined as L and Encoding Mask is defined as EM. We call $\mathcal{L}^{\dagger}_{KL}$ the KL-Divergence loss presented by InfoNeRF and we call CombiNeRF† our method using their $\mathcal{L}^{\dagger}_{KL}$.

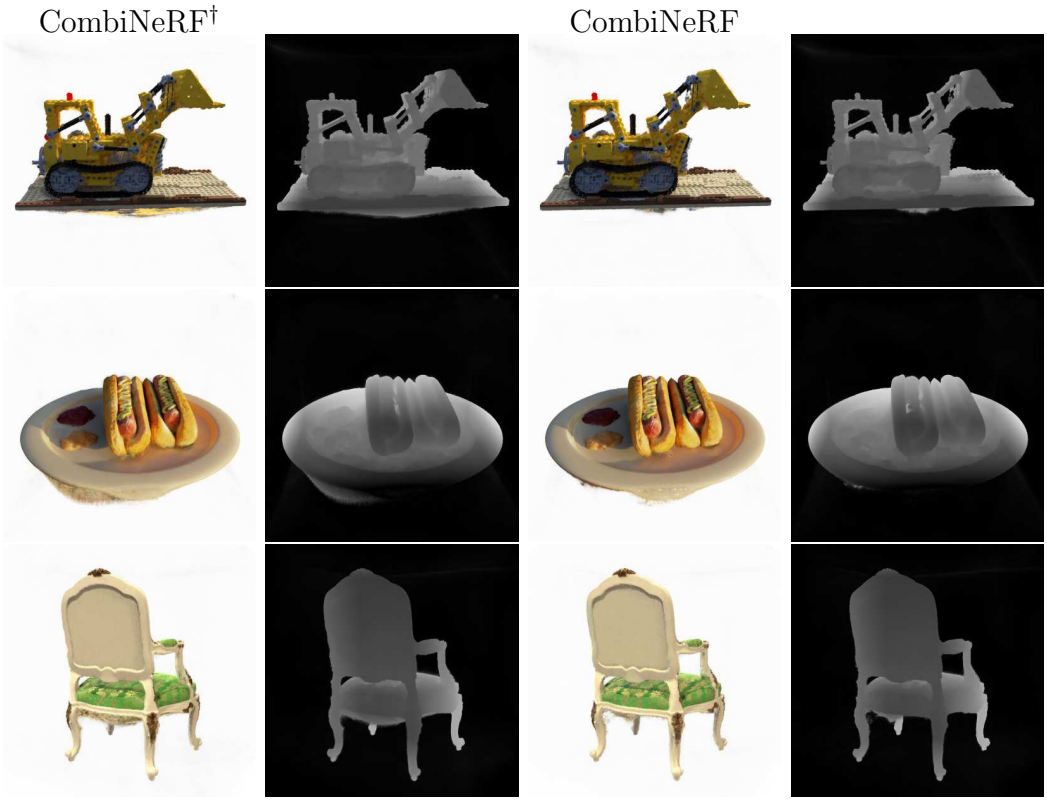


Figure 4.5: Qualitative result on the ablation study of NeRF-Synthetic with 8-view setting. We compare our CombiNeRF with CombiNeRF† (our method with KL-Divergence as in InfoNeRF).

loss used in [25] (implemented in an intermediate version of CombiNeRF called CombiNeRF†) and our modified version (as described in Sec. 3.1) used in the final CombiNeRF.

In Fig. 4.5 we show the qualitative results of both the two implementations. While quantitative results remain almost the same (see Tab. 4.4), CombiNeRF is able to remove most of the artifacts generated below the objects, thus indicating

Fox 9-views	PSNR ↑	SSIM ↑	LPIPS ↓	Average ↓
Vanilla NeRF	17.06	0.679	0.613	0.190
$\mathcal{L}_{dist} + \mathcal{L}_{fg}$+L	22.07	0.775	**0.348**	0.101
$\mathcal{L}_{dist} + \mathcal{L}_{fg} + \mathcal{L}_{ds}$+L	22.69	**0.791**	0.376	0.097
$\mathcal{L}_{dist} + \mathcal{L}_{fg} + \mathcal{L}_{ds}$+L+EM	23.89	0.784	0.402	0.091
CombiNeRF*	**24.21**	**0.791**	0.386	**0.087**

Table 4.5: Ablation on the Fox dataset with 9 input views. Lipschitz is defined as L and Encoding Mask is defined as EM. CombiNeRF* is our method using near-pose sampling instead of near-pixel sampling in the computation of $\mathcal{L}_{KL}$.

Fox	PSNR ↑	SSIM ↑	LPIPS ↓	Average ↓
Vanilla NeRF	25.77	0.842	0.347	0.072
$\mathcal{L}_{ds}$	26.43	0.839	0.385	0.071
EM	27.70	0.852	0.339	0.061
$\mathcal{L}_{dist} + \mathcal{L}_{fg}$	29.82	0.863	0.279	0.048
$\mathcal{L}_{dist} + \mathcal{L}_{fg}$+EM	28.75	0.857	0.290	0.053
$\mathcal{L}_{dist} + \mathcal{L}_{fg}$+L	**30.22**	**0.869**	**0.277**	**0.046**

Table 4.6: Ablation on the full Fox dataset. Lipschitz is defined as L and Encoding Mask is defined as EM.

an improved reconstruction of the final geometries.

4.3.3 Fox Dataset

Starting from the full dataset experiment, in Tab. 4.6 are shown quantitative results considering some of the implementations of CombiNeRF. Lipschitz and $\mathcal{L}_{dist} + \mathcal{L}_{fg}$ are the best-performing forms of regularization, in fact also their combination increases the scores of each metric, especially the PSNR and LPIPS. The Encoding Mask technique is able to improve the result with respect to Vanilla NeRF, but it gets worse together with the others. In Fig. 4.6 are shown the qualitative results. Lipschitz with $\mathcal{L}_{dist} + \mathcal{L}_{fg}$ enable a smoother depth reconstruction, also avoiding the wrong reconstructed floral pattern on the wall. Moreover, the RGB images have high-frequency details and light effects that are closer to the ground truth.

Instead, in Tab. 4.5 are shown quantitative results of the Fox dataset under 9-view setting. Vanilla NeRF clearly struggles more when dealing with fewer input views. The combinations of multiple techniques definitely improve the performance even if there is not a single clear winner in all the scores. Anyway, CombiNeRF*, which deviates from CombiNeRF by only changing the sampling

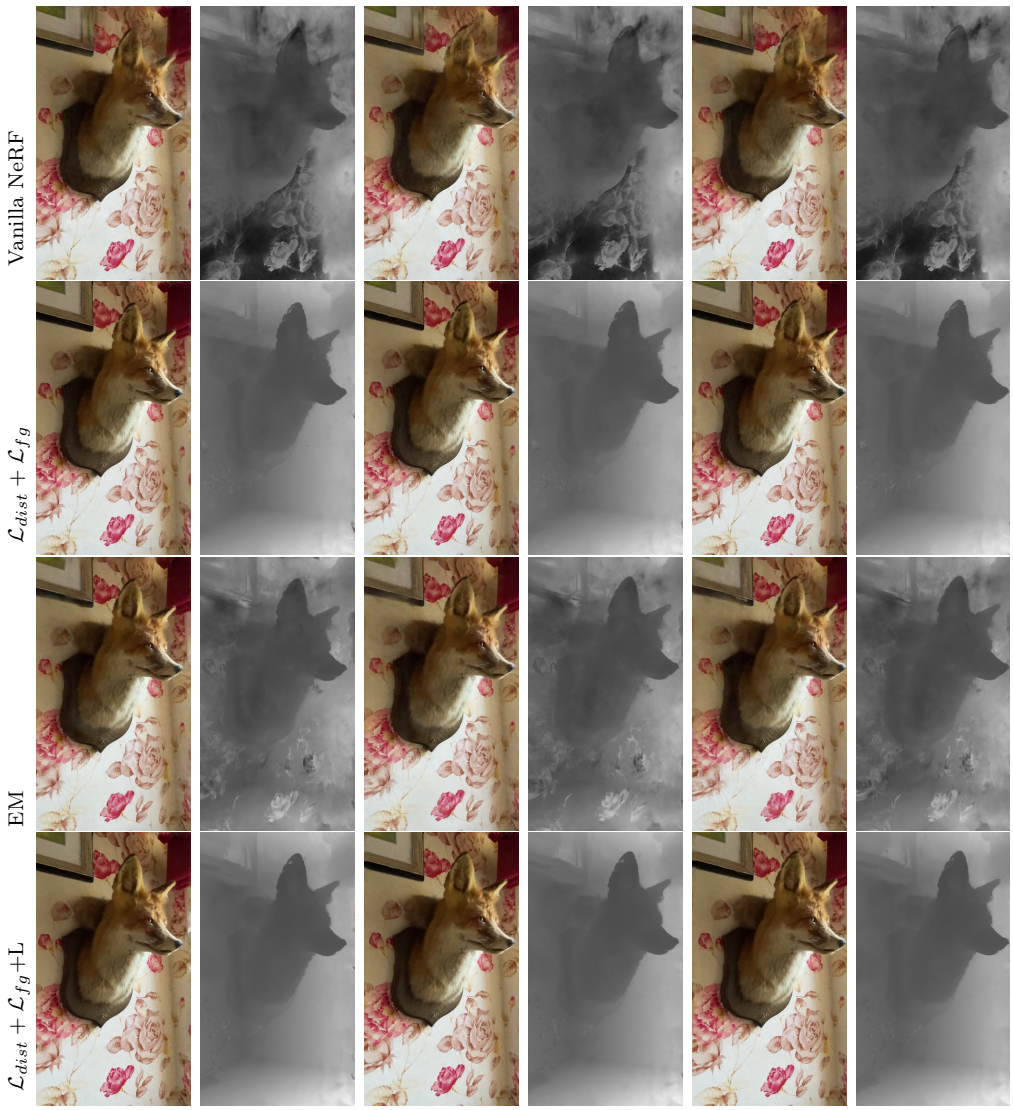


Figure 4.6: Qualitative result on the ablation study of Fox with full dataset in different implementations. Lipschitz is defined as L and Encoding Mask is defined as EM.

method of $\mathcal{L}_{KL}$ from near-pixel to near-pose, has the overall best performance and outperforms Vanilla NeRF. In Fig. 4.7 are shown the qualitative results of Fox under 9-view setting, comparing them together with the one using no few-shot, as described before. It's interesting to notice how the regularization technique enables better reconstruction in the RGB images against Vanilla NeRF, especially in few-shot settings where, however, depths are badly reconstructed in all the cases.

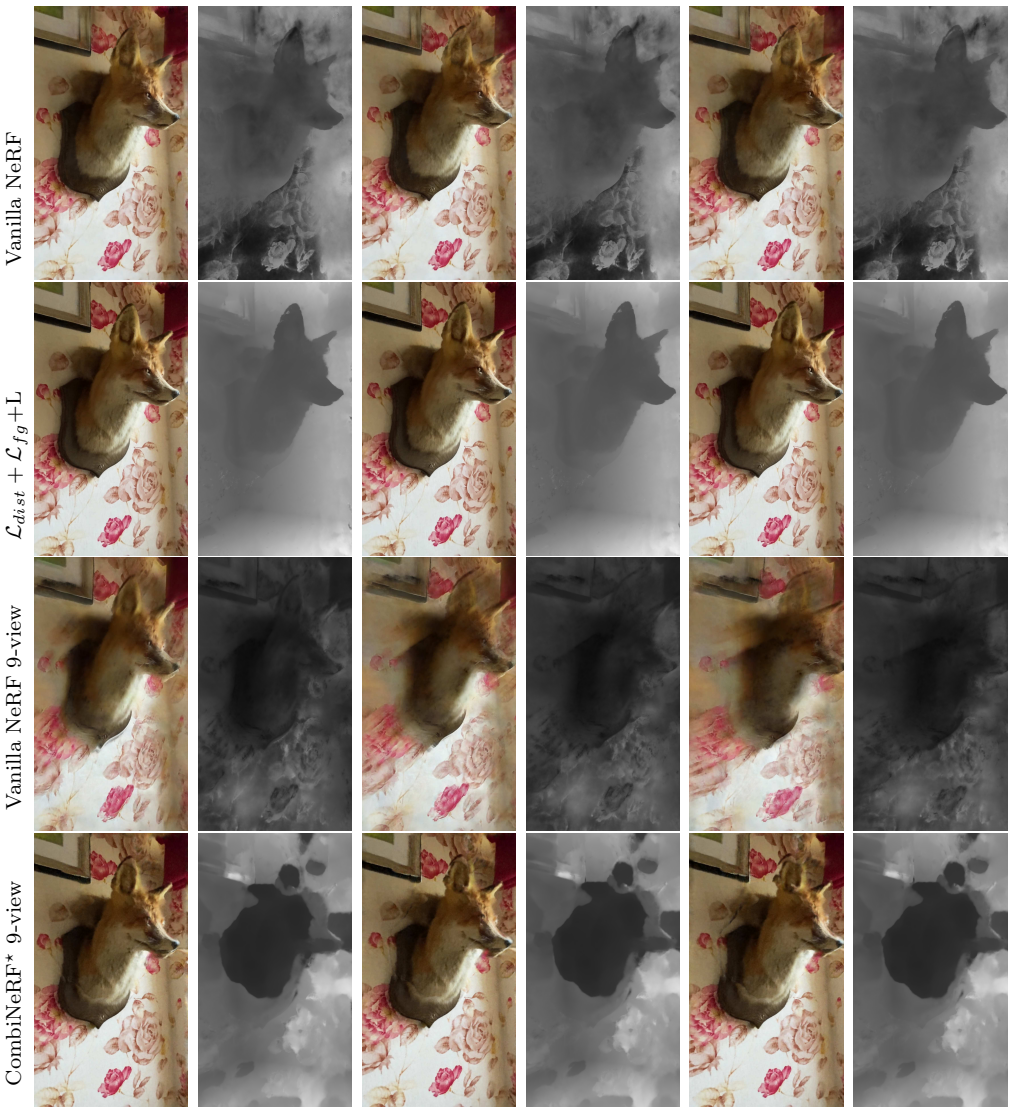


Figure 4.7: Qualitative comparison on the ablation of Fox dataset against Vanilla NeRF with full dataset (top two rows) and under 9-view setting (bottom two rows). CombiNeRF* is our method using near-pose sampling instead of near-pixel sampling in the computation of $\mathcal{L}_{KL}$, while Lipschitz technique is defined as L.

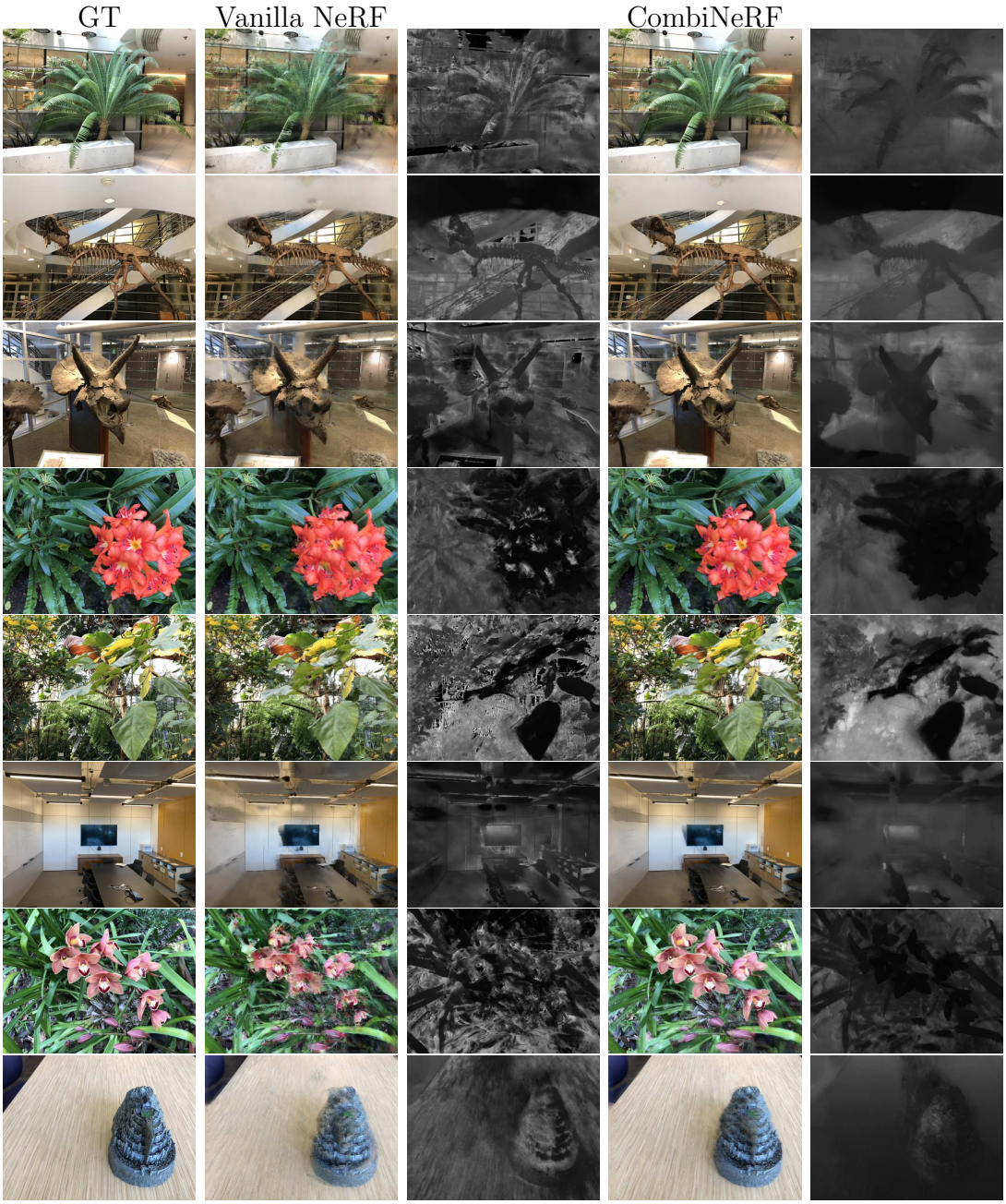


Figure 4.8: Additional qualitative results on all the scenarios of LLFF dataset with 3-view setting.

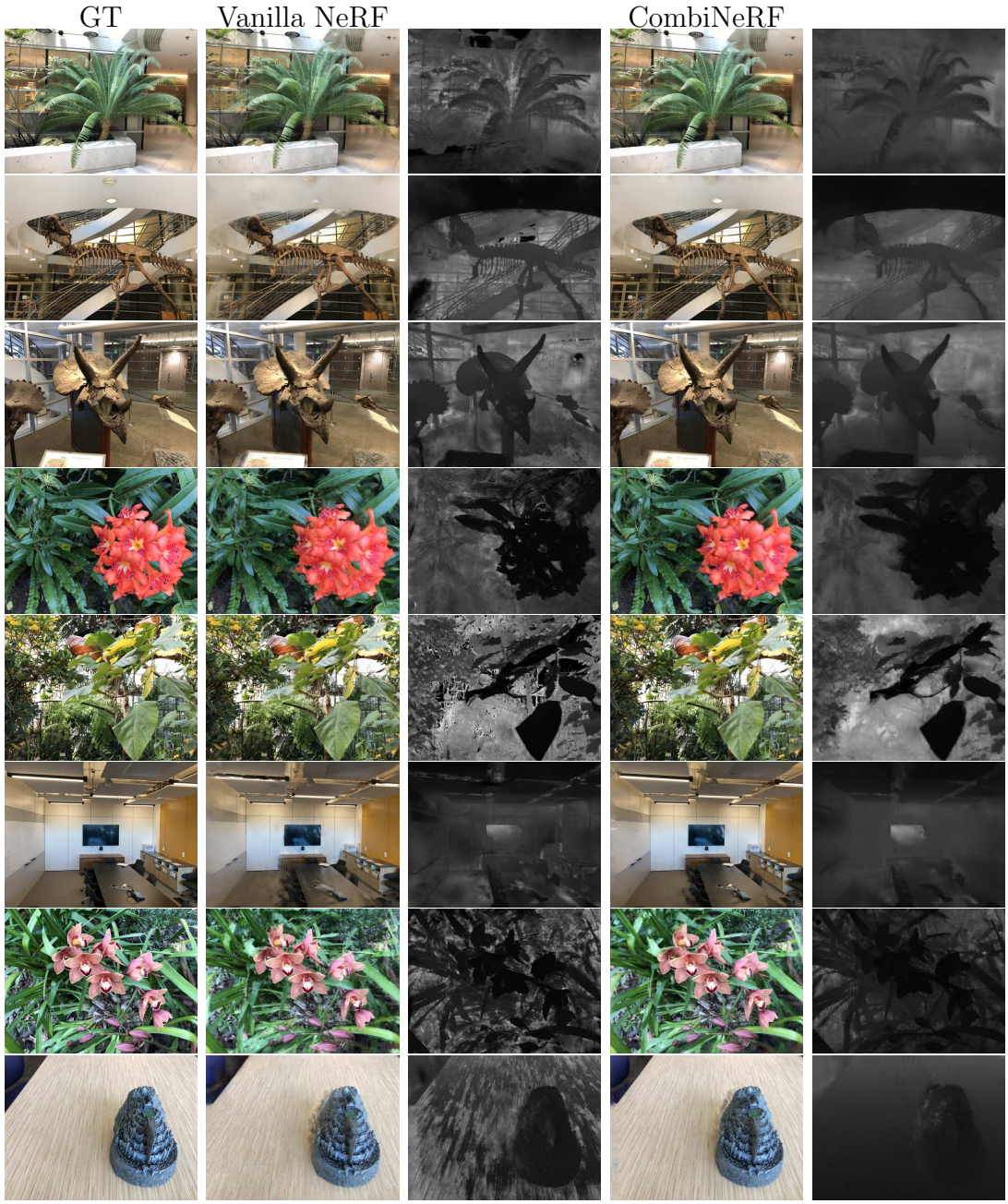


Figure 4.9: Additional qualitative results on all the scenarios of LLFF dataset with 6-view setting.

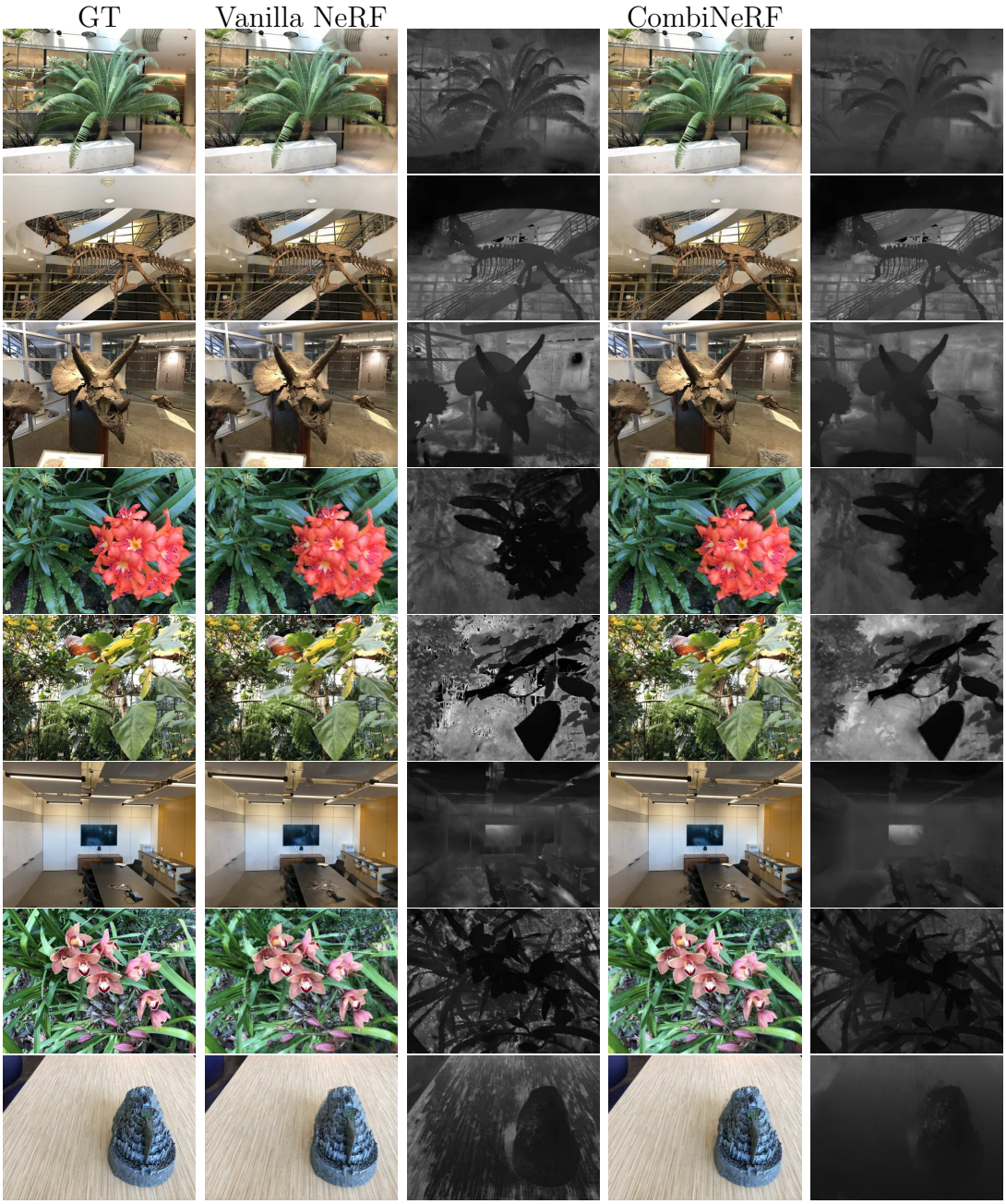


Figure 4.10: Additional qualitative results on all the scenarios of LLFF dataset with 9-view setting.

(a) 3 input views.

LLFF 3-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	12.47	17.72	15.65	18.54	20.54	19.52	17.79	19.47
CombiNeRF	**16.04**	**18.46**	**18.84**	**21.01**	**21.21**	**21.47**	**22.21**	**23.75**

(b) 6 input views.

LLFF 6-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	16.74	20.1	22.32	22.25	21.66	25.58	23.18	24.42
CombiNeRF	**18.11**	**20.46**	**23.26**	**24.52**	**23.91**	**28.72**	**25.07**	**27.9**

(c) 9 input views.

LLFF 9-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	18.45	21.07	24.59	24.88	25.27	27.11	25.54	26.76
CombiNeRF	**19.1**	**21.22**	**25.15**	**26.13**	**26.35**	**28.57**	**26.17**	**28.51**

Table 4.7: Per-scene quantitative results on LLFF dataset w.r.t. **PSNR**↑ metric.

(a) 3 input views.

LLFF 3-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	0.199	0.61	0.523	0.548	0.719	0.755	0.509	0.486
CombiNeRF	**0.461**	**0.666**	**0.692**	**0.668**	**0.773**	**0.821**	**0.713**	**0.692**

(b) 6 input views.

LLFF 6-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	0.499	0.738	0.797	0.725	0.816	0.894	0.744	0.688
CombiNeRF	**0.584**	**0.753**	**0.823**	**0.815**	**0.863**	**0.921**	**0.811**	**0.873**

(c) 9 input views.

LLFF 9-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	0.605	0.777	0.86	0.82	0.885	0.912	0.827	0.805
CombiNeRF	**0.647**	**0.782**	**0.876**	**0.857**	**0.908**	**0.93**	**0.85**	**0.877**

Table 4.8: Per-scene quantitative results on LLFF dataset w.r.t. **SSIM**↑ metric.

(a) 3 input views.

LLFF 3-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	0.435	0.181	0.35	0.3	0.195	0.269	0.402	0.292
CombiNeRF	**0.25**	**0.155**	**0.212**	**0.226**	**0.15**	**0.184**	**0.194**	**0.157**

(b) 6 input views.

LLFF 6-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	0.23	0.124	0.137	0.156	0.133	0.112	0.166	0.137
CombiNeRF	**0.178**	**0.12**	**0.122**	**0.095**	**0.094**	**0.08**	**0.105**	**0.054**

(c) 9 input views.

LLFF 9-view	Orchids	Leaves	Horns	Flower	T-rex	Room	Fern	Fortress
Vanilla NeRF	0.171	**0.106**	0.091	0.093	0.078	0.083	0.101	0.084
CombiNeRF	**0.151**	0.112	**0.078**	**0.071**	**0.06**	**0.066**	**0.082**	**0.05**

Table 4.9: Per-scene quantitative results on LLFF dataset w.r.t. **LPIPS**↓ metric.

NeRF-Synthetic 8-views	Drums	Material	Ship	Ficus	Lego	Chair	Mic	Hotdog
Vanilla NeRF	18.07	19.36	18.69	22.93	22.49	25.25	27.34	24.55
DietNeRF, $\mathcal{L}_{MSE}$ ft	20.029	**21.621**	**22.536**	20.940	24.311	25.595	26.794	26.626
CombiNeRF	**20.165**	20.73	21.392	**23.493**	**24.958**	**27.862**	**28.172**	**28.383**

Table 4.10: Per-scene quantitative results on NeRF-Synthetic dataset w.r.t. **PSNR**↑ metric.

NeRF-Synthetic 8-views	Drums	Material	Ship	Ficus	Lego	Chair	Mic	Hotdog
Vanilla NeRF	0.767	0.804	0.69	0.897	0.851	0.901	0.949	0.901
DietNeRF, $\mathcal{L}_{MSE}$ ft	**0.845**	**0.851**	**0.757**	0.874	0.875	0.912	0.950	0.924
CombiNeRF	0.838	0.838	0.754	**0.91**	**0.885**	**0.933**	**0.957**	**0.945**

Table 4.11: Per-scene quantitative results on NeRF-Synthetic dataset w.r.t. **SSIM**↑ metric.

NeRF-Synthetic 8-views	Drums	Material	Ship	Ficus	Lego	Chair	Mic	Hotdog
Vanilla NeRF	0.222	0.185	0.269	0.107	0.105	0.076	0.054	0.134
DietNeRF, $\mathcal{L}_{MSE}$ ft	**0.117**	**0.095**	0.193	0.094	0.096	0.077	0.043	0.067
CombiNeRF	0.14	0.114	**0.151**	**0.078**	**0.072**	**0.049**	**0.035**	**0.066**

Table 4.12: Per-scene quantitative results on NeRF-Synthetic dataset w.r.t. **LPIPS**↓ metric.

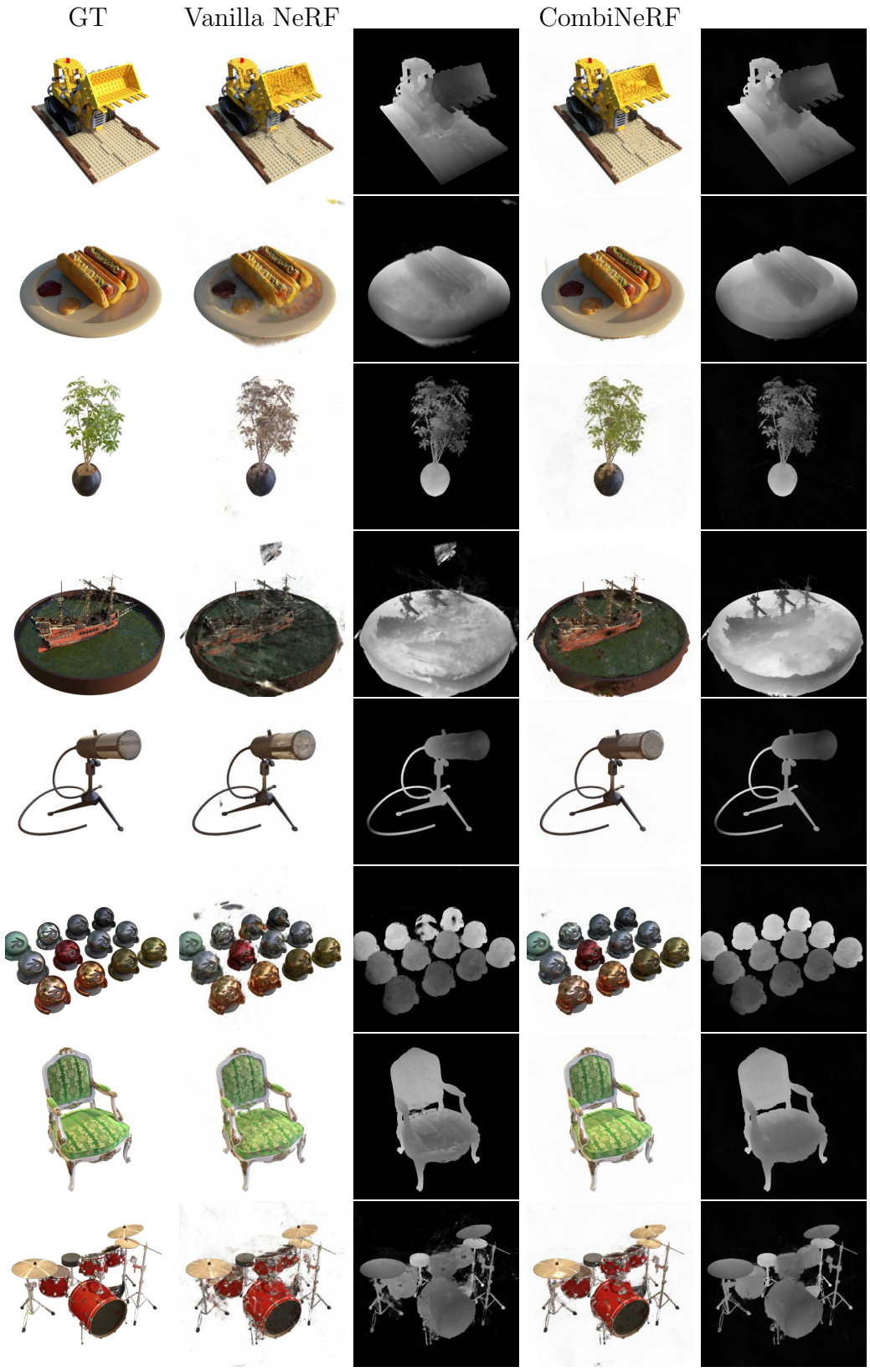


Figure 4.11: Additional qualitative results on all the scenarios of NeRF-Synthetic dataset with 8-view setting.

Chapter 5

Conclusion

In this thesis, we presented CombiNeRF, a combination of regularization techniques in a unified framework for few-shot Neural Radiance Field novel view synthesis. We regularize neighboring ray distributions, we also take into account the single ray distribution and a smoothness term is adopted to regularize near geometries. Besides these additional loss functions, we also consider Lipschitz regularization and an Encoding Mask to regularize high-frequency components.

CombiNeRF shows cutting-edge and consistent results in the quality of the reconstructions and it outperforms the SOTA methods in LLFF and NeRF-Synthetic datasets with few-shot settings. The increasing performance of the combination of multiple regularization techniques is validated through an ablation study that highlights the contribution of the CombiNeRF components and the differences with the previous implementations.

Additional techniques like Ray Entropy Loss, Adversarial Perturbation, and Space Annealing have been taken into account but not considered in CombiNeRF for their apparently poor contribution. More experiments are required as future works on these and other techniques in order to achieve a richer and more general-purpose NeRF framework.

The research activity performed within this thesis work has been submitted to a major conference in 3D and Computer Vision. At the moment of writing, the submitted paper is still under review and the final revision will be released upon acceptance. The open-source code implementation of CombiNeRF will also be released upon publication of this thesis document and the corresponding scientific paper.

5.1 Acknowledgment

This thesis was developed as part of the internship at FlexSight[1]. I thank everyone in the company who followed me, supported me and contributed to this work, especially Matteo Bonotto, Marco Imperoli, and Daniele Evangelista.

[1] https://flexsight.eu/

References

[1] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.

[2] Jiaxiang Tang. Torch-ngp: A pytorch implementation of instant-ngp, 2022.

[3] Michael Waechter, Nils Moehrle, and Michael Goesele. Let there be color! large-scale texturing of 3d reconstructions. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 836–850. Springer, 2014.

[4] C BUEHLER. Unstructured lumigraph rendering. In *Proc. SIGGRAPH'01*, pages 425–432, 2001.

[5] PE DEBEC. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *Proc. SIGGRAPH'96*, 1996.

[6] DN WOOD. Surface light fields for 3d photography. *SIGGRAPH2000, Jul.*, pages 287–296, 2000.

[7] M LEVOY. Light field rendering. In *Computer Graphics Proceedings, Annual Conference Series, Proc. SIGGRAPH'96 (New Orleans), August 1996*. ACM SIGGRAPH, 1996.

[8] Pratul P Srinivasan, Tongzhou Wang, Ashwin Sreelal, Ravi Ramamoorthi, and Ren Ng. Learning to synthesize a 4d rgbd light field from a single image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2243–2251, 2017.

[9] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Transactions on Graphics (TOG)*, 32(3):1–12, 2013.

[10] Gaurav Chaurasia, Olga Sorkine, and George Drettakis. Silhouette-aware warping for image-based rendering. In *Computer Graphics Forum*, volume 30, pages 1223–1232. Wiley Online Library, 2011.

[11] Sudipta Sinha, Drew Steedly, and Rick Szeliski. Piecewise planar stereo for image-based rendering. In *2009 International Conference on Computer Vision*, pages 1881–1888, 2009.

[12] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *International journal of computer vision*, 38:199–218, 2000.

[13] Steven M Seitz and Charles R Dyer. Photorealistic scene reconstruction by voxel coloring. *International Journal of Computer Vision*, 35:151–173, 1999.

[14] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 517–524. IEEE, 1998.

[15] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2367–2376, 2019.

[16] Phlipp Henzler, Volker Rasche, Timo Ropinski, and Tobias Ritschel. Single-image tomography: 3d volumes from 2d cranial x-rays. In *Computer Graphics Forum*, volume 37, pages 377–388. Wiley Online Library, 2018.

[17] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. *Advances in neural information processing systems*, 30, 2017.

[18] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019.

[19] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):1–11, 2017.

[20] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 175–184, 2019.

[21] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2626–2634, 2017.

[22] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification. *ACM Transactions on Graphics*, 37(4):1–12, 2018.

[23] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[24] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[25] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *CVPR*, 2022.

[26] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5470–5479, June 2022.

[27] Jamie Wynn and Daniyar Turmukhambetov. DiffusioNeRF: Regularizing Neural Radiance Fields with Denoising Diffusion Models. In *CVPR*, 2023.

[28] Kevin Scaman and Aladin Virmaux. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Neural Information Processing Systems*, 2018.

[29] Hsueh-Ti Derek Liu, Francis Williams, Alec Jacobson, Sanja Fidler, and Or Litany. Learning smooth neural functions via lipschitz regularization. *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings*, Aug 2022.

[30] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[31] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5885–5894, October 2021.

[32] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2019.

[33] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, July 2019.

[34] Jae Shin Yoon, Kihwan Kim, Orazio Gallo, Hyun Soo Park, and Jan Kautz. Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020.

[35] Phong Nguyen-Ha, Animesh Karnewar, Lam Huynh, Esa Rahtu, and Janne Heikkila. Rgbd-net: Predicting color and depth images for novel views synthesis. In *Proceedings of the International Conference on 3D Vision*, 2021.

[36] Eric R. Chan, Koki Nagano, Matthew A. Chan, Alexander W. Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. GeNVS: Generative novel view synthesis with 3D-aware diffusion models. In *arXiv*, 2023.

[37] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022.

[38] Benjamin Attal, Jia-Bin Huang, Michael Zollhöfer, Johannes Kopf, and Changil Kim. Learning neural light fields with ray-space embedding networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[39] Liwen Wu, Jae Yong Lee, Anand Bhattad, Yu-Xiong Wang, and David Forsyth. Diver: Real-time and accurate neural radiance fields with deterministic integration for volume rendering. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2022.

[40] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P. Srinivasan, and Jonathan T. Barron. NeRF in the dark: High dynamic range view synthesis from noisy raw images. *CVPR*, 2022.

[41] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. P. Mildenhall, P. Srinivasan, J. T. Barron, and H. Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition*

(CVPR), pages 8238–8248, Los Alamitos, CA, USA, jun 2022. IEEE Computer Society.

[42] M. Piala and R. Clark. Terminerf: Ray termination prediction for efficient neural rendering. In *2021 International Conference on 3D Vision (3DV)*, pages 1106–1114, Los Alamitos, CA, USA, dec 2021. IEEE Computer Society.

[43] Thomas Neff, Pascal Stadlbauer, Mathias Parger, Andreas Kurz, Joerg H. Mueller, Chakravarty R. Alla Chaitanya, Anton S. Kaplanyan, and Markus Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 40(4), 2021.

[44] Sara Fridovich-Keil and Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022.

[45] Peng Wang, Yuan Liu, Zhaoxi Chen, Lingjie Liu, Ziwei Liu, Taku Komura, Christian Theobalt, and Wenping Wang. F2-nerf: Fast neural radiance field training with free camera trajectories. *CVPR*, 2023.

[46] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[47] Yihao Zhi, Shenhan Qian, Xinhao Yan, and Shenghua Gao. Dual-space nerf: Learning animatable avatars and scene lighting in separate spaces. In *International Conference on 3D Vision (3DV)*, September 2022.

[48] Matteo Poggi, Pierluigi Zama Ramirez, Fabio Tosi, Samuele Salti, Luigi Di Stefano, and Stefano Mattoccia. Cross-spectral neural radiance fields. In *Proceedings of the International Conference on 3D Vision*, 2022. 3DV.

[49] Michael Niemeyer and Andreas Geiger. Campari: Camera-aware decomposed generative neural radiance fields. In *International Conference on 3D Vision (3DV)*, 2021.

[50] Christopher Xie, Keunhong Park, Ricardo Martin-Brualla, and Matthew Brown. Fig-nerf: Figure-ground neural radiance fields for 3d object category modelling. In *International Conference on 3D Vision (3DV)*, 2021.

[51] Yuan-Chen Guo, Di Kang, Linchao Bao, Yu He, and Song-Hai Zhang. Nerfren: Neural radiance fields with reflections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18409–18418, June 2022.

[52] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021.

[53] Barbara Roessle, Jonathan T Barron, Ben Mildenhall, Pratul P Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12892–12901, 2022.

[54] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[55] Tianlong Chen, Peihao Wang, Zhiwen Fan, and Zhangyang Wang. Aug-nerf: Training stronger neural radiance fields with triple-level physically-grounded augmentations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[56] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33, 2020.

[57] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. PhySG: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[58] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022.

[59] Jingwen Wang, Tymoteusz Bleja, and Lourdes Agapito. Go-surf: Neural feature grid optimization for fast, high-fidelity rgb-d surface reconstruction. *2022 International Conference on 3D Vision (3DV)*, Sep 2022.

[60] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2020.

[61] Stefan Lionar, Lukas Schmid, Cesar Cadena, Roland Siegwart, and Andrei Cramariuc. Neuralblox: Real-time neural representation fusion for robust volumetric mapping. *2021 International Conference on 3D Vision (3DV)*, Dec 2021.

[62] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *CoRR*, abs/2106.10689, 2021.

[63] Yiqun Wang, Ivan Skorokhodov, and Peter Wonka. Hf-neus: Improved surface reconstruction using high-frequency details. *Advances in Neural Information Processing Systems*, 35:1966–1978, 2022.

[64] Fuqiang Zhao, Yuheng Jiang, Kaixin Yao, Jiakai Zhang, Liao Wang, Haizhao Dai, Yuhui Zhong, Yingliang Zhang, Minye Wu, Lan Xu, and Jingyi Yu. Human performance modeling and rendering via neural animated mesh. *ACM Trans. Graph.*, 41(6), nov 2022.

[65] Radu Alexandru Rosu and Sven Behnke. Permutosdf: Fast multi-view reconstruction with implicit surfaces using permutohedral lattices. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[66] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[67] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[68] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

Appendix A

Additional Implementations

In the following sections, we describe three additional implementations considered during the empirical selection of the best methods used in CombiNeRF. The following implementations were not included in CombiNeRF for their apparently poor contribution with respect to Vanilla NeRF [1]. Due to a lack of time for further testing, we were only able to partially evaluate these methods, which will eventually be considered in future CombiNeRF extensions. As done for the other considered techniques, we initially tested their performance on the LLFF dataset with the 3-view setting, comparing their results against Vanilla NeRF. For completeness, we provide below a brief summary of these techniques.

A.1 Regularization Losses

A.1.1 Entropy Loss

The aim is to introduce a sparsity constraint to focus only on the scene of interest. This constraint can be achieved by minimizing the entropy of each sampled ray density function, as done in [25], in order to force the rays to have a few high peaks, i.e. the peaks should only represent the hit surfaces.

To minimize the entropy of a ray $\mathbf{r}$, we start defining the normalized ray density $q(\mathbf{r})$ as:

$$q(\mathbf{r}_i) = \frac{\alpha_i}{\sum_j \alpha_j} = \frac{1 - \exp(-\sigma_i \delta_i)}{\sum_j 1 - \exp(-\sigma_j \delta_j)}, \tag{A.1}$$

where $\mathbf{r}_i$ $(i = 1, ..., N)$ is a sampled point in the ray, σ_i is the observed density at $\mathbf{r}_i$, δ_i is a sampling interval around $\mathbf{r}_i$ and α_i is the opacity at $\mathbf{r}_i$.

Starting from the Shannon Entropy, we can define the entropy of a discrete

ray density function:

$$H(\mathbf{r}) = -\sum_{i=1}^{N} q(\mathbf{r}_i) \log q(\mathbf{r}_i), \tag{A.2}$$

where the calculation of $q(\mathbf{r}_i)$, which includes σ_i and δ_i, is already given thanks to volume rendering computation.

A mask M is applied to filter the rays in the entropy defined in Eq. (A.2) with the aim of discarding too low-density rays not intersecting or hitting any object in the scene:

$$M(\mathbf{r}) = \begin{cases} 1 & \text{if } Q(\mathbf{r}) > \epsilon \\ 0 & \text{otherwise} \end{cases}, \tag{A.3}$$

where

$$Q(\mathbf{r}) = \sum_{i=1}^{N} 1 - \exp(-\sigma_i \delta_i), \tag{A.4}$$

is the cumulative ray density.

Because unobserved viewpoints can help generalization, for the entropy loss computation both rays from training views and from unseen images can be considered, whose sets are denoted respectively by R_t and R_u. The final entropy loss is defined as:

$$\mathcal{L}_{entr} = \lambda_{entr} \cdot \frac{1}{|R_t| + |R_u|} \sum_{\mathbf{r} \in R_t \cup R_u} M(\mathbf{r}) \odot H(\mathbf{r}), \tag{A.5}$$

where $\odot$ denotes the element-wise multiplication and λ_{entr} defines the contribution of this loss.

A.1.2 Perturbation

Previous works demonstrate how neural networks benefit from random or learned data augmentation. Generalization can be achieved by injecting noise during the training phase, as done in [55]. Here, noise is introduced in the form of worst-case perturbations applied on the input coordinates, into the features of the network, and on the pre-rending output of the network. The worst-case perturbation can be formulated as a min-max problem defined as follows:

$$\min_{\Theta} \max_{\delta} \left\| C^{\dagger}(r|\Theta, \delta) - C^{*}(r) \right\|_2^2, \tag{A.6}$$

where

$$\delta = (\delta_p, \delta_f, \delta_r) \in \mathcal{S}_p \times \mathcal{S}_f \times \mathcal{S}_r, \tag{A.7}$$

and $\delta_p, \delta_f, \delta_r$ are the learned perturbation, $\mathcal{S}_p, \mathcal{S}_f, \mathcal{S}_r$ are the corresponding search range and $C^\dagger(r|\Theta, \delta)$ is the perturbed rendered color.

The input coordinates perturbation $\delta_p = (\delta_t, \delta_{xyz}, \delta_\theta)$ consists of along-ray perturbation $\delta_t \in \mathbb{R}^3$, point position perturbation $\delta_{xyz} \in \mathbb{R}^3$ and view-direction perturbation $\delta_\theta \in \mathbb{R}^3$ (hence $S_p \subseteq \mathbb{R}^6$). The feature perturbation δ_f consists of perturbing a D-dimensional features vector, hence $\mathcal{S}_f \subseteq \mathbb{R}^D$. Finally, a pre-rendering output perturbation is applied to RGB color and density values ($\mathcal{S}_r \subseteq \mathbb{R}^4$).

In order to estimate the worst-case perturbation of the injected noise, [55] proposes a multi-step Projected Gradient Descent (PGD) approach. Given l_{inf} as the norm ball defining all search spaces $\mathcal{S}_p, \mathcal{S}_f, \mathcal{S}_r$ and the radius ϵ as the maximum magnitude of the perturbation, a PGD step is defined as:

$$\delta^{(t+1)} = \prod_{\|\delta\|_\infty \leq \epsilon} \left[\delta^{(t)} + \alpha \cdot \text{sgn}(\left\| C^\dagger(r|\Theta, \delta) - C^*(r) \right\|_2^2) \right], \tag{A.8}$$

where α is the step size, $\prod[\cdot]$ is a projection operator and $\text{sgn}(\cdot)$ is the sign function.

Following this procedure, the adversarial perturbation loss is obtained as:

$$\mathcal{L}_{adv} = \lambda_{adv} \cdot \sum_{r \in \mathcal{R}} \left\| C^\dagger(r|\Theta, \delta) - C^*(r) \right\|_2^2, \tag{A.9}$$

where λ_{adv} defines the contribution of this loss.

A.2 Sample Space Annealing

When few images are provided, NeRF could converge to undesired solutions. In this case, high-density values are concentrated close to ray origins. Annealing the sampled scene space over the early iterations of the training [24] can be a solution to avoid this specific problem: the idea is to restrict the sampling space to an initial smaller region where the scene is centered.

Given the camera's near and far plane t_n and t_f, let t_m be a point between them, likely their midpoint, and define the new near and far plane per-iteration

LLFF 3-views	PSNR ↑	SSIM ↑	LPIPS ↓	Average ↓
Vanilla NeRF	17.71	0.544	0.303	0.155
$\mathcal{L}_{adv}$	17.75	0.541	0.300	0.155
$\mathcal{L}_{entr}$	17.34	0.526	0.312	0.161
Anneal	18.04	0.562	0.284	0.147
CombiNeRF	**20.37**	**0.686**	**0.191**	**0.101**

Table A.1: Quantitative results comparison among the other three regularization techniques on LLFF dataset with 3-view setting.

as:

$$\begin{aligned} t_n(i) &= t_m + (t_n - t_m)\eta(i) \\ t_f(i) &= t_m + (t_f - t_m)\eta(i) \\ \eta(i) &= \min(\max(i/N_t, p_s), 1), \end{aligned} \tag{A.10}$$

where i is the current training iteration, the parameter N_t represents the maximum number of iterations in which to perform the space annealing and p_s indicates the starting range. These parameters should be tuned according to the particular scene under consideration.

Annealing the sampled space, especially during the first iteration of training, may allow NeRF to focus on the right region of interest, avoiding degenerate solutions.

A.3 Results

We evaluated the three implementations presented above on the LLFF dataset with a 3-view setting. We compared the obtained results with the results obtained by Vanilla NeRF to see if a specific technique should be considered for further experiments or should be revised or better fine-tuned.

In Tab. A.1 we show the quantitative results of this study. CombiNeRF remains the best-performing method, outperforming all the other methods by a large margin. $\mathcal{L}_{entr}$ is not able to improve NeRF generalization and, on the contrary, degrades its performance. $\mathcal{L}_{adv}$ slightly improves some metrics but it doesn't affect the overall score. Moreover, the time complexity required to perform PGD for worst-case perturbation considerably increases the total computa-

tional time. Instead, the Space Annealing technique outperforms Vanilla NeRF in all metrics. This result would lead us to focus on the latter as a promising method to be tested and eventually integrated into CombiNeRF. However, we argue that the contribution provided by Space Annealing could not offer additional benefits when paired with the distortion loss of Eq. (3.4), as both methods try to remove high-density values on points close to the camera.

In Fig. A.1 and Fig. A.2 are shown qualitative results of Vanilla NeRF against the other three implementations. In Space Annealing results, the depths are smoother, and the reconstruction quality increases as visible in the white wall and the light on top of "T-rex", in the leaves' details on "Fern" and in the overall object reconstruction of "Fortress". With $\mathcal{L}_{adv}$ is very difficult to see improvements in the resulting renderings, which reflects the quantitative results seen in Tab. A.1. In $\mathcal{L}_{entr}$ results, both depth and RGB images are less accurate, containing more artifacts (e.g., in "Fortress") and noise, particularly visible in the white wall of "T-rex".

Figure A.1: RGB comparison of Vanilla NeRF against Space Annealing, Adversarial Perturbation and Entropy loss on LLFF with 3-view setting.

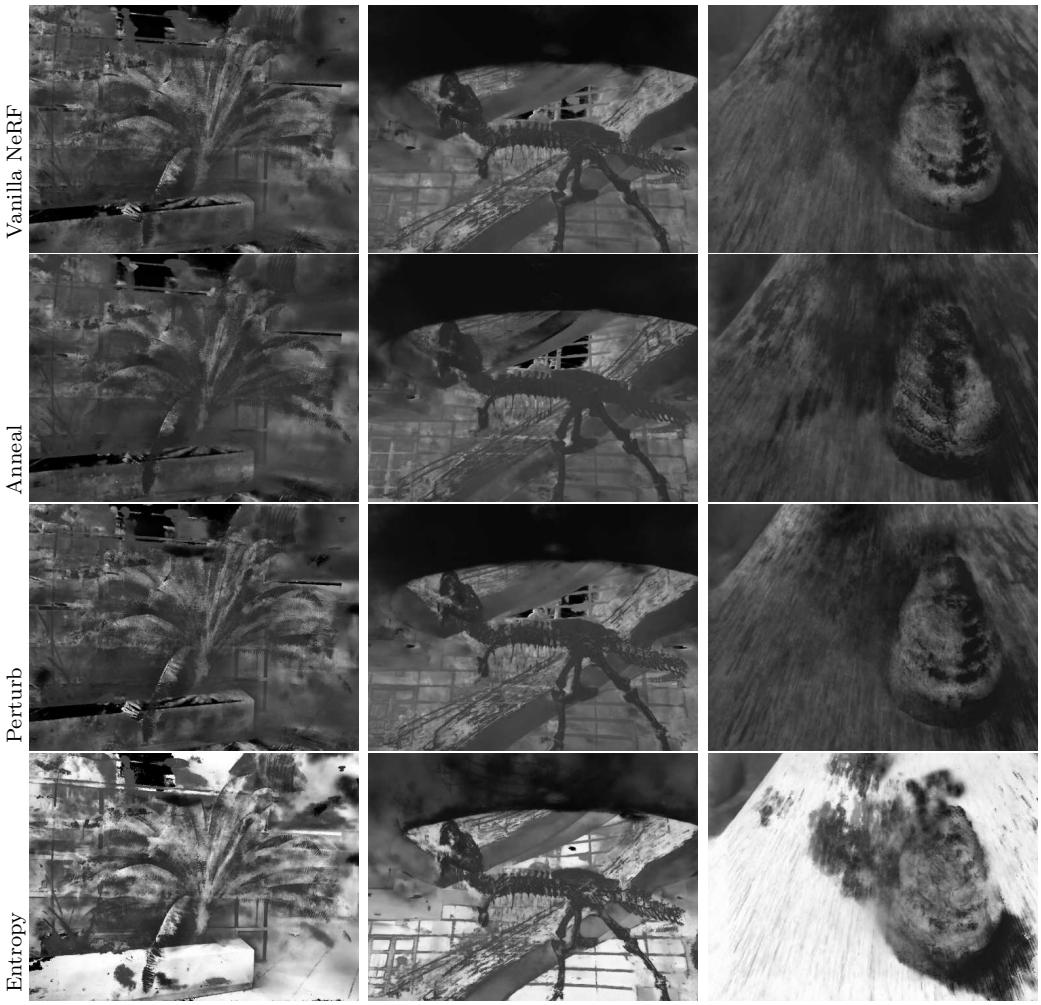


Figure A.2: Depth comparison of Vanilla NeRF against Space Annealing, Adversarial Perturbation and Entropy loss on LLFF with 3-view setting.