SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY - INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Robotics, Cognition, Intelligence

Neural Radiance Field Reconstruction with Depth and Normal Constraints

Alexander Sheldrick

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY - INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Robotics, Cognition, Intelligence

Neural Radiance Field Reconstruction with Depth and Normal Constraints

Rekonstruktion mit neuronalen Strahlungsfeldern mit Tiefen- und Normaleneinschränkungen

Author:	Alexander Sheldrick
Supervisor:	Prof. Dr. Matthias Nießner
Advisor:	Prof. Dr. Matthias Nießner
Submission Date:	15.02.2023

I confirm that this master's thesis in robotics, cognition, intelligence is my own work and I have documented all sources and material used.

Munich, 15.02.2023 Alexander Sheldrick

Preface

It is with great excitement that I present this Master's thesis on the topic of novel view synthesis using Neural Radiance Fields (NeRFs) with depth and normal constraints. The field of computer vision continues to evolve rapidly, and view synthesis is an area that holds great potential for creating more realistic and interactive virtual environments, and for democratizing the creation of virtual assets. This thesis represents the culmination of several months of hard work, and I am proud to present it as my contribution to this field.

I hope that this thesis presents a useful study not only to the field of computer vision, but also to future researchers and students who are interested in exploring this area further. Through this work, I have learned a great deal about the theory and practices of view synthesis, as well as the challenges and limitations of this field. I believe that this thesis provides valuable insights and practical solutions that can help advance the state-of-the-art in this area.

I would like to express my deep gratitude to my thesis advisor, Prof. Dr. Matthias Nießner, for his guidance and support throughout this journey. His expertise and our insightful discussions have been invaluable to me, and I am grateful for the opportunities they have given me to learn and grow as a researcher. I would also like to extend my heartfelt thanks to my parents, George and Katharine, for their unwavering love and support throughout my academic career. Their belief in me has been a constant source of inspiration, and I am grateful for everything they have done for me.

Finally, I would like to express my profound gratitude to my partner, the love of my life, Chiara. Her love, support, and encouragement have been immeasurable throughout this journey, and she has been my rock through both the good times and the challenging ones. Her unwavering love has made me the best version of myself, and I am eternally grateful to have her by my side. Thank you, Chiara, for being my everything.

Alexander Sheldrick

Munich, February 2023

Abstract

This thesis addresses the problem of novel view synthesis from sparse-view supervision in the field of computer vision. Neural radiance fields (NeRFs) are a popular approach for this problem, but they rely heavily on a large dataset of images and precisely calibrated cameras. Motivated by recent advances in the area of monocular geometry prediction, which allow for cheap generation of depth- and normal maps, we systematically explore methods to incorporate these cues for the supervision of NeRFs. Our proposed method bounds the weights accumulated along rays using a Gaussian cumulative density function about the predicted depth. These bounds are directly derived from a Gaussian assumption on the likelihood of a ray being absorbed on its way through a neural volume. We show that our method, contrary to prior work, consistently improves reconstruction results for any number of training views, with photorealistic reconstructions being feasible with as few as three views. Our contribution to the field of computer vision is a flexible and easily implementable improvement to the performance of NeRFs for novel view synthesis.

Contents

1. Introduction

1.1. Motivation and Research Statement

View synthesis refers to the process of generating new views of a scene or an object from a given set of views or images. It is an important and active research area, with a rich body of work, lying at the intersection of the fields of computer vision and computer graphics, and has numerous applications in various domains, and areas such as virtual and augmented reality, robotics, and the entertainment industry.

In the field of virtual reality (VR), it is used to generate new views of a scene in real-time to match the viewpoint of the VR headset. This enhances the immersive experience of the user and trades the computational overhead of rendering new views in real-time for bandwidth and memory to store entire scenes in a photorealistic quality.

View synthesis can also be used in the field of robotics to generate new views of the environment from a robot's perspective, providing the robot with a more comprehensive understanding of its surroundings. In the entertainment industry, it is used to digitalize assets and environments from a collection of images, rather than having to expensively create them by hand by highly trained professionals. The traditional approach to view synthesis involves creating intermediate representations, i.e. 3D models of the scene or object, and then rendering new views by changing the viewpoint. However, with recent advancements in deep learning, the task of view synthesis can be performed more efficiently and realistically using generative models, such as Generative Adversarial Networks (GANs [Goo+20]) and Variational Autoencoders (VAEs [KW13]), and neural radiance fields (NeRFs [Mil+21]).

These recent approaches can generate novel views with high realism and quality, and their use case is generally distinguished by an important methodological difference: Generative models are trained on a large dataset of real-world images to generate completely new content to coherently match the input. On the other hand, Neural Radiance Fields (NeRFs) are trained as a per-scene representation, based on a collection of images that depict a specific scene only. As a result, NeRFs allow for the synthesis of accurate and high-quality novel views of said scene, but a trained model does not generalize to other scenes.

NeRFs have the potential to revolutionize the creation of realistic and interactive virtual environments and democratize the entertainment industry. By enabling users

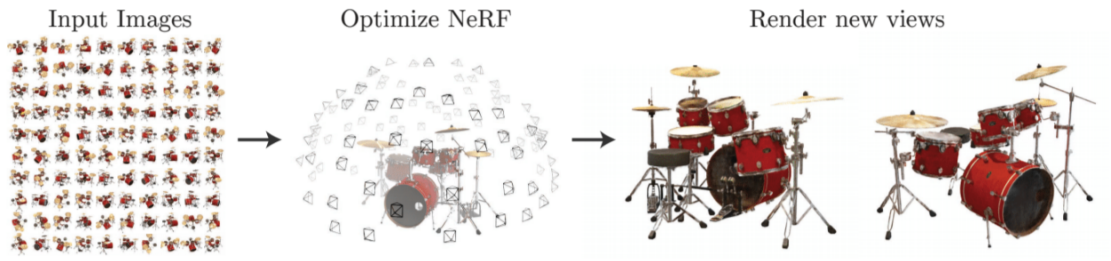

Figure 1.1.: Neural radiance fields (NeRFs [Mil+21] introduce a stochastic notion of visibility, to generate novel views, and to circumvent difficulties in differentiating through ray-triangle intersections. Using a fully differentiable volume rendering equation, and a dense set of input images with camera parameters, a neural network is regressed to predict a density and color at every point of a neural volume during training. This network is then queried during inference to generate photorealistic novel views, with full freedom over camera pose, lens properties, and image resolution, allowing for greater artistic freedom, and the creation of AR and VR media without the need for highly trained digital artists.

to automatically populate virtual environments from collections of images without the need for highly trained 3D artists, NeRFs have a significant impact on its business. However, despite these advancements, most current NeRF methods are still limited by their slow and complex training, and high computational costs, making them unsuitable for deployment on edge devices and hindering a real-time user experience. Furthermore, the training of NeRFs depends on the availability of a large collection of images with constant illumination, obtained from precisely calibrated cameras, for dense and accurate supervision.

Despite a plethora of literature addressing the speed, accuracy, and fundamental formulation of Neural Radiance Fields, there remains a gap in the field regarding the effective utilization of depth and normal data for supervision. This information can be obtained inexpensively from consumer-grade RGB-D cameras and from monocular depth and normal prediction networks, such as OmniData [Eft+21]. The incorporation of this data has the potential to significantly improve the accuracy and speed up the training of NeRFs, making it a crucial area for future research and development.

1.2. Research Objectives and Expected Outcomes

1. A comprehensive review and analysis of the state-of-the-art methods in this field, which could be valuable for researchers, practitioners, and students looking to explore this area.

2. Advancements in the understanding of the underlying principles and challenges of novel view synthesis with monocular inputs, which could lead to new insights and applications in related areas such as virtual and augmented reality, 3D reconstruction, and robotics.

3. Develop novel approaches to improving the quality and realism of synthesized views, while simultaneously reducing the number of views necessary for training, with the use of complementary monocular cues during training, i.e. color, depth, and normal maps, to increase the effectiveness of NeRFs in general.

4. Comparative evaluation of different deep learning models and approaches for novel view synthesis, including their performance on different types of images depicting synthetic and real-world scenes.

Expected Outcomes. I expect to gain a deep understanding of computer vision and image processing techniques. I will explore the state-of-the-art methods used to address this problem, including classical structure from motion algorithms generally, and suites of algorithms such as COLMAP specifically. I will discuss, compare and learn from recent developments in neural radiance fields (NeRFs) relevant to novel view synthesis.

I also anticipate gaining expertise in data preparation and augmentation, as well as in evaluating the performance of the models using metrics such as PSNR and SSIM. Additionally, I expect to develop skills in problem-solving, critical thinking, and experimental design, as I investigate novel approaches to improve the quality and realism of synthesized views.

Overall, my expected learning outcomes include an in-depth understanding of novel view synthesis with monocular inputs, advanced technical skills in computer vision and machine learning, and the ability to conduct independent research and contribute to the field's knowledge.

1.3. Research Contributions

In this work, we present a novel loss formulation derived from the fundamental principles of Neural Radiance Fields (NeRFs) and statistical considerations, which enhances the performance of novel view reconstruction. Our approach incorporates depth (and normal) constraints and provides a consistent improvement over previous methods. We demonstrate the limitations of prior work and emphasize the ease of implementation of our method in contemporary NeRF frameworks. Our contribution represents a significant advancement in the field of view synthesis and has the potential to be widely adopted.

1.4. Thesis Structure

In this thesis, we tackle the long-standing challenge of view synthesis from sparse-view supervision. This work consists of 6 chapters and follows the above-outlined research objectives.

Following the introduction and motivation, Chapter 2 provides a comprehensive review of the relevant literature in the field, including the state-of-the-art techniques and theories, and introduces the research question. In Chapter 3, the working principle of Neural Radiance Fields (NeRFs) is discussed in detail, as it is directly relevant to the proposed method and the main contribution of the thesis.

Chapter 4 describes the proposed novel view synthesis approach using NeRFs in detail, including the methodology, experiments, and evaluations. The results are presented and analyzed in Chapter 5, with a focus on comparing them with existing approaches. Finally, Chapter 6 concludes the thesis with a discussion of the findings and provides insights for future work in the field.

2. Related Works

The techniques presented in this thesis aim to directly extend NeRF, a highly influential technique of directly regressing a 3D scene from a collection of images, with the intent to synthesize novel photo-realistic views. This, essentially, is the task of novel view synthesis.

Before discussing the details and implementation, it is important to review the recent developments of the field, especially in the context of 3D representations used by computer graphics for view synthesis, and the recently introduced continuous implicit representations (neural fields).

2.1. Novel View Synthesis

Novel view synthesis is a long-standing computer vision task with a large body of work. The classical approaches consist of

1. Projection-based methods, i.e. projecting the texture of an object or scene onto a 3D geometry to synthesize new views.

2. Light field Rendering: Given a dense sampling of views one can employ Light field sample interpolation techniques [LH96; DLD12] that can synthesize photo-realistic novel views.

3. Multi-view Stereo (MVS): The reconstruction of a scene's 3D geometry by using multiple images captured from different viewpoints and the robust matching of photo-metric features. The position of image points in 3D space can then be triangulated from the intersection of their projection rays. Integral to the success of this technique are knowledge of the poses and precise calibration of the cameras from which the images were taken. MVS typically takes a set of ordered images as input and produces a dense and accurate point cloud or triangular mesh of the scene.

4. Structure from Motion (SfM), a technique complementary to MVS, is primarily used to recover the camera poses and 3D structure of a scene. The inputs to SfM are usually a set of unordered images from which it produces a sparse point

cloud, e.g. a set of key points, and camera poses. The resulting point cloud can be refined using various algorithms and robust matching procedures such as bundle adjustment to produce a more accurate and dense representation of the 3D structure. SfM is widely used in computer vision and photogrammetry and is still an active area of research and development, with ongoing efforts to improve the accuracy, speed, and scalability of its techniques.

One such suite of techniques, COLMAP [SF16], is a general-purpose SfM and MVS pipeline that is used by this work to generate the poses and camera parameters used in the experimental evaluations (chapter 5). For a brief illustration of COLMAP's reconstruction procedure see Figure 2.1.

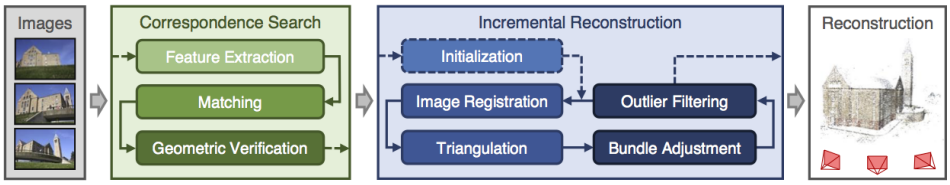

Figure 2.1.: Overview of COLMAP's incremental Structure-from-Motion pipeline. COLMAP extracts distinctive features from a collection of images and matches them amongst images to establish correspondences. Next, the intrinsic- and extrinsic- parameters of each camera are estimated and initialized. The parameters of initialized cameras are then iteratively refined via bundle adjustment, and key points that cannot be triangulated during optimization are rejected as erroneous matches. [SF16]

These traditional techniques use explicit representations of geometry such as point clouds, voxel grids, or triangular meshes, with very distinct advantages and disadvantages. While e.g. **Voxel Representations** are obvious candidates to describe space with much the same techniques used in the deep learning revolution to describe images, their cubically scaling memory footprint limits training to small batch-sizes and slow training, making them unsuitable to describe complex scenes. While their problems can be remedied by employing octree data structures, they have effectively fallen out of favor for more elegant descriptions of space.

Pointclouds are widely used in both robotics and the computer graphics communities, as they are relatively lightweight and able to model complex scene geometries, it has been difficult to incorporate them in convolution-based deep learning architectures, as they are usually unevenly sampled, cannot model empty space and have no obvious local order to perform convolutions on. Qi et al. [Qi+17a; Qi+17b] achieved permutation invariance by a applying fully connected neural network to each point independently, followed by a global pooling operation, pioneering point clouds for discriminative learning.

Triangular meshes are collections of interconnected triangles that approximate surfaces in 3D space. They are efficient representations of curved spaces compared to voxels and provide smooth and continuous surfaces and enclosed volumes compared to point clouds, but are typically much more difficult to handle than their alternatives, as their faces are implicitly described by their vertex positions in space.

2.2. Neural Field Scene Representation

Concurrent works in 2019 [Par+19; Mes+19] explored neural implicit representations (neural fields, coordinate networks) in which a Multi-Layer Perceptron (MLP) directly regresses a continuous scene description from a vector-valued input (usually coordinates) and is optionally conditioned on some latent code or image. The output scene description, i.e. SDF and Occupancy -fields, is provided at training time as supervision from ground truth meshes. The trained MLP thus learns to map input coordinates to Occupancy-/SDF- level-sets and thereby fully describes the boundaries of a watertight object in 3D space.

The advantages of these representations are an extremely low memory footprint, as the entire scene is encoded in the weights of the MLP, with no need to keep track of explicit representations of objects and volumes in the scenes. The major drawbacks however are the need for ground truth 3D data (e.g. ShapeNet [Cha+15]) as supervision, and expensive evaluations at test time, as the only way to extract the information from the MLP is through queries on a dense volumetric grid.

Subsequent works suggested optimizing directly from 2D images, via differential rendering approaches, to alleviate the need for ground truth 3D supervision, data which for real-world application is prohibitively expensive and cumbersome to generate. Niemeyer et al. [Nie+20] trained neural volumes in the form of occupancy fields, for which they introduced a differentiable rendering formula to alleviate restrictions necessitating the use of voxel- and mesh-based representations. Sitzmann et al. [Sit+20] instead proposed Scene Representation Networks, coordinate networks able to regress high-frequency 3D geometry from low dimensional coordinate inputs, by replacing typical ReLU activation functions with harmonic functions, creating similar capabilities to the later widely used Fourier embedding popularized by NeRF.

2.2.1. NeRF

NeRF [Mil+21] presented a novel method for generating photo-realistic novel views from camera parameters and 2D images alone. They proposed regressing a neural field from the 5D vector of input coordinates $\mathbf{x} = (x, y, z)$ and viewing direction $\mathbf{d} = (\theta, \phi)$ to output volume density $\sigma(\mathbf{x})$ and radiance $\mathbf{c} = (r, g, b)$. The central idea, to overcome the challenges of differentiating through ray-triangle intersections, was a probabilistic notion of visibility [TM22].

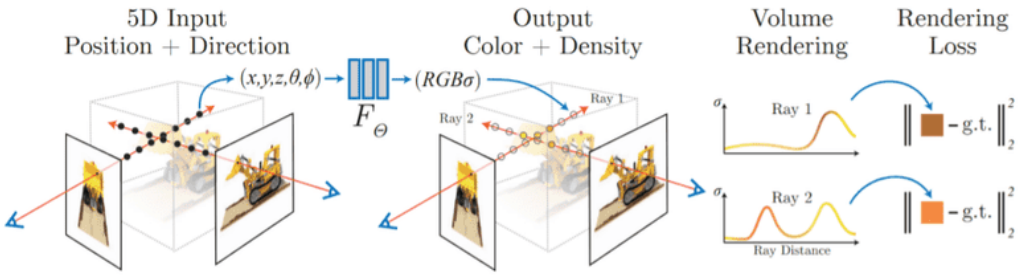

Figure 2.2.: An overview of NeRF's neural radiance field scene representation and differentiable rendering procedure. Images are synthesized by sampling 5D coordinates (location and viewing direction) along camera rays. (a) Samples from the reprojected rays are embedded (featurized) in a periodic, higher dimensional space, and fed into an MLP (b) to produce a color $\mathbf{c}$ and volume density $\sigma(\mathbf{x})$. NeRF's rendering function (c) is fully differentiable and thus NeRFs can be optimized from camera parameters and images alone (d), without need the for 3D supervision. [Mil+21].

The network, in summary, learns to predict the radiance of incoming light at each point in the scene, modeling occlusions as density (differential opacity, i.e. the differential probability of the light being absorbed over an interval) and encodes reflectance and view dependant effects via view-encoding of the camera from which the ray originates.

The network is trained on a large set of input views, minimizing the reconstruction error between the predicted and ground truth views. This allows the network to synthesize new views of the scene by evaluating the learned NeRF at arbitrary camera positions. From each image, a ray is projected into the neural volume and densely sampled at discrete points. The samples are then alpha-composited with their respective alpha-compositing weights (calculated from densities) that are inferred from the network at each point. See Figure 2.2 for a brief visualization of the technique.

Another significant contribution is the therein proposed effective coordinate embedding (Fourier features), motivated by neural tangent kernel theory, that map the low dimensional inputs to an orthogonal basis in a higher dimensional space. Without such

an embedding, the network fails to regress complex scene contents from low dimensional inputs, as the output images suffer from severe aliasing and over-smoothing.

NeRF started something akin to a gold rush in computer vision, as it was easily trained and featured photorealistic results. The technique however is not without drawbacks, it relies on dense supervision of equidistant and perfectly calibrated cameras with identical lighting conditions in each frame. Additionally, the first NeRFs took days to train, which started a wave of works addressing the various shortcomings, primarily relaxing the supervision and improving training times. Concurrent works, after only two years after the first NeRF publication, already feature near real-time rendering speeds and improved reconstruction results [Mül+22; Che+22].

2.2.2. NeRF variants

Some of the most influential follow-up works [Bar+21; Bar+22] are from the original authors. Mip-NeRF improves on NeRF with a multi-scale encoding that encodes featurized volumes rather than discrete points, thus allowing renderings at any resolution or scale. This ameliorates the severe aliasing exhibited by the original NeRF encodings while being simultaneously more accurate overall and faster to train.

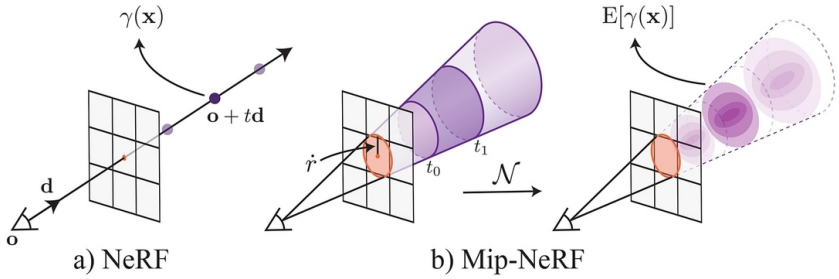

Figure 2.3.: NeRF (a) samples a neural field on discrete points along rays projected from the camera center through pixels. Mip-NeRF (b) instead reasons about conical frustums defined by the ray and pixel-radius $\dot{r}$. By featurizing conical frustums, approximated by multivariate Gaussians, the network is able to reason about the scale of inputs, ameliorating common aliasing problems of NeRF. [Bar+21]

In a subsequent publication, Mip-NeRF360°, the authors relaxed the restriction to bounded scenes by introducing a ray-warping function related to Kalman filters. They further proposed an improved sampling routine: by using a shallow proposal network to propose regions of interest, the number of samples required for inference can be reduced, yielding an increase in speed and accuracy. Finally, the results of this work largely build upon this model (Mip-NeRF360°, sans scene-wide warp) that has been

ported to PyTorch [Pas+19] from JAX [Bra+18] throughout the thesis.

2.2.3. Improving NeRF reconstructions with monocular cues

Successful NeRF reconstructions from RGB images require a large number of input views taken under static conditions, or else geometries will be incorrectly fitted. Depth as supervision is considered an easily available, cheap to generate, and proven signal to generate more robust 3D reconstructions. Different works [Den+22; Rem+22; Roe+22] have proposed their procedures and loss formulations and reported great success, i.e. faster training, more accurate results, and fewer images required.

One common observation however, is that when the number of images is large, depth supervision can hamper reconstruction quality, as it potentially constricts the neural radiance field into a local minimum with inferior PSNR[1], by enforcing the depth supervision constraints. The cautious advice is to rely solely on the RGB signal for maximum reconstruction quality if the collection of images is large enough.

Surface normals, of objects in the depicted scene, is another strong signal and previous works [Ver+22; Yu+22] have successfully used this cue to improve reflections in NeRF [Ver+22] and to improve 3D reconstruction quality of NeRFs regressing complex scenes [Yu+22] as a signed distance field (SDF).

Contrary to previous works, this work investigates the use of depth and normal cues to improve photo-metric reconstruction in all settings: with few, and with hundreds. See chapter 4 for details on the presented method, and chapter 5 for an extensive evaluation thereof on synthetic and real data.

[1] Peak Signal to Noise Ratio (PSNR) is a commonly used evaluation metric in computer vision and image processing that measures the similarity between two images. It provides a means of comparing the quality of reconstructed or restored image data to the original image data. The higher the PSNR value, the lower the level of distortions in the reconstructed image, indicating a higher degree of similarity with the original image. See Appendix A for details

3. Neural Radiance Fields: Theory

This work builds upon the digest published by Andrea Tagliasacchi in [TM22], whose clear descriptions, physical interpretation, and concise formalism have served as a significant inspiration for this study of neural fields. By first reiterating their work's principal statements, and by later expanding upon the ideas put forth, this work aims to make a meaningful contribution to the existing literature on the topic.

The following derivation of NeRF's underlying principles is closely related to the derivations presented in the original publication and the aforementioned digest [Mil+21; TM22].

3.1. NeRF: Probabilistic Interpretation

A scene in NeRF is represented as a 5D vector-valued function, whose inputs are a 3D location $\mathbf{x} = (x, y, z)$, and a 2D viewing-direction $\mathbf{d} = (\theta, \phi)$. To these inputs, the network maps to each location, conditioned on the respective viewing direction, an emitted color $\mathbf{c}(\mathbf{x}) = (r, g, b)$ and a volume density-field $\sigma(\mathbf{x})$. This neural network mapping of inputs to outputs is compactly written as $F_\theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$.

The scene is assumed to be comprised of a cloud of light-emitting particles. The neural volume is modeled to absorb and emit light but does not allow inter-particle scattering. For the sake of simplicity, the following derivation assumes that the emitted light from within the volume does not change as a function of viewing direction.

The density-field $\sigma(\mathbf{x})$ describes the volume and denotes the differential likelihood of a ray being absorbed by a particle over an infinitesimal distance. Since every point $\mathbf{x}$ along a given ray $\mathbf{r} = \mathbf{r}(\mathbf{o}, \mathbf{d}) = \mathbf{o} + t\mathbf{d}$ in the volume is fully characterized by the ray's origin $\mathbf{o}$, and the ray's direction $\mathbf{d}$, the vector-valued density field $\sigma(\mathbf{x})$ can be rewritten as a scalar-valued density field $\sigma(t)$, that depends only on the distance t traveled along the ray.

This density is closely tied to the transmittance function $\mathcal{T}(t)$, which indicates the probability of a ray traveling over the interval $[0, t)$ without hitting any particles, and following the derivations of [TM22], we find that the probability of $\mathcal{T}(t + dt)$, i.e. the probability of not hitting a particle when taking a differential step dt through the volume, is equal to $\mathcal{T}(t) \cdot (1 - dt \cdot \sigma(t))$, i.e. the likelihood of the ray reaching t, multiplied with the probability of not hitting a light absorbing particle during the step.

Formally we are looking to solve the following differential equation:

$$\mathcal{T}(t + dt) = \mathcal{T}(t) \cdot (1 - dt \cdot \sigma(t)) \tag{3.1}$$

$$\frac{\mathcal{T}(t + dt) - \mathcal{T}(t)}{dt} \equiv \mathcal{T}'(t) = -\mathcal{T}(t) \cdot \sigma(t) \tag{3.2}$$

In a probabilistic interpretation, the function $1 - \mathcal{T}(t)$, i.e. the probability denoting the event that the ray gets absorbed by a particle before arriving at t, can be interpreted as the opacity of the medium along the ray, and as the cumulative distribution function (CDF) for the event of ray absorption. The corresponding probability density function (PDF), the derivative of the CDF, follows trivially from Equation 3.2:

$$(1 - \mathcal{T}(t))' = -(-\mathcal{T}(t) \cdot \sigma(t)) = \mathcal{T}(t) \cdot \sigma(t) \tag{3.3}$$

and indicates the likelihood that the ray stops at precisely t. Solving this differential equation with an exponential approach leads to the transmittance function $\mathcal{T}(a \to b)$ for a continuous interval $[a, b]$:

$$\mathcal{T}(a \to b) \equiv \frac{\mathcal{T}(b)}{\mathcal{T}(a)} = \exp\left(-\int_a^b \sigma(t)\, dt\right) \tag{3.4}$$

3.2. Volume Rendering with Radiance Fields

To render the resulting color for a ray passing through the neural volume (radiance field), the colors are inferred from the MLP $F_\theta : (\mathbf{x}, \mathbf{d}) \to (\mathbf{c}, \sigma)$ at discrete sampling points, conditioned on the rays respective (viewing-) direction. These colors can be calculated by solving the volume rendering integral Equation 3.5:

$$\boldsymbol{C}(t_{N+1}) = \sum_{n=1}^{N} \int_{t_n}^{t_{n+1}} \mathcal{T}(t) \cdot \sigma_n \cdot \mathbf{c}_n \, dt \tag{3.5}$$

With the assumption that the ray is traversing the volume through piece-wise constant densities, the volume rendering integral, and Equation 3.4, one recovers (without derivation) [Mil+21] [Section 4, Eq. 3.]:

$$\begin{aligned} \boldsymbol{C}(t_{N+1}) &= \sum_{n=1}^{N} \mathcal{T}_n \cdot (1 - \exp(-\sigma_n \delta_n)) \cdot \mathbf{c}_n \\ with \quad \mathcal{T}_n &= \exp\left(\sum_{k=1}^{n-1} -\sigma_k \delta_k\right) \end{aligned} \tag{3.6}$$

where $\delta_i = t_{i+1} - t_i$ is the distance between adjacent samples along the ray. Equation 3.6 is trivially differentiable and reduces to traditional alpha compositing of the transmission function with alpha weights $\alpha_i = 1 - exp(-\sigma_i \delta_i)$.

3.3. Sampling

Brute force dense evaluation of the neural radiance field on points along each camera ray is inefficient, as free and occluded regions that do not contribute to the rendered image are repeatedly sampled. This consideration is integral to most NeRF works, beginning from the first publication [Mil+21], and continuing with two of the authors' principal follow-up works [Bar+21; Bar+22]. Different strategies of performing intelligent sampling have been proposed, that in some form adhere to the "coarse to fine" strategy. This scheme draws inspiration from early work in volume rendering [Lev90] and aims to increase rendering efficiency by allocating samples proportionally to their expected effect on the final rendering.

To this effect, the "coarse" sampling entails collecting stratified samples in uniform intervals over the ray throughout the entire volume. A forward-pass through an MLP (initially a NeRF MLP, in the newest works a smaller proposal MLP) generates weights over intervals from the densities (differential opacity), which, when normalized, are treated as a piecewise-constant probability density function over the ray and its intervals: interpreting the weights as proxies for the differential probabilities for the event of ray termination. New samples are then drawn from this PDF for the "fine" stage via inverse transform sampling, which are finally passed to a "NeRF-MLP", that performs view dependant color and density inference for each point. See Figure 2.2 for an overview of the general idea and sampling mechanism.

One significant design choice is whether the densities are sampled from the same MLP as is the case in MipNeRF [Bar+21], or if the densities are generated by a smaller MLP as in MipNeRF360° [Bar+22], that for a (3x) increase of speed, leads to a (33%) increase in parameters, and a reported increase in accuracy, especially for fine structures.

3.4. Supervision

NeRFs are optimized from a collection of RGB images from well-calibrated cameras with given poses and extrinsic. For synthetic experiments, the experiments are directly trained with ground truth camera parameters. For real images, these parameters are generally inferred from SfM algorithms (usually with COLMAP).

For each image $n \in N$ from the dataset, a ray $\mathbf{r}_{n,i}(t)$ is projected into the neural volume from that camera's center $\mathbf{o}_n$. The ray goes through the center of its respective

pixel $px_{n,i}$ along the direction $\mathbf{d}$. In total, there are $i \in [1, H \cdot W]$ rays per image, and the resulting rays, $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, are fully defined by the camera parameters and their respective images. In the neural volume, the neural field is strategically (section 3.3) sampled along the path defined by $\mathbf{r}_{n,i}$. The colors and densities inferred at each sample point are alpha composited and denote the predicted color for the pixel $px_{n,i}$.

The principal supervision signal for rays in initial works is a gradient descent on a simple mean squared error loss function (MSE) between the predicted and ground truth pixel value, although recently published MipNeRF360° found the Charbonnier loss to lead to more stable results. This function was designed as a differentiable robust loss function and effectively interpolates between L1-loss for deviations larger than ϵ, and L2-loss for deviations smaller than ϵ (a parameter controlling the shape of the resulting function). The color loss for a mini-batch of N rays is calculated as follows:

$$\mathcal{L}_{chb}(\mathbf{c}, \hat{\mathbf{c}}) = \sum_{n \in N} \sqrt{(\mathbf{c}_n - \hat{\mathbf{c}}_n)^2 + \epsilon^2} \tag{3.7}$$

With ϵ being set to a small value, usually $\epsilon = 0.001$. Further monocular supervision signals relevant to this work are listed in Appendix B.

3.5. Positional Encoding

Finally, even though neural networks are universal function approximators [HSW89], it became evident that networks operating directly on low dimensional inputs are biased towards learning lower frequency functions [Rah+19]. Positional encoding is integral to the ability of an MLP to regress complex scenes with high-frequency details from low-dimensional inputs.

The core insight is that lifting inputs to an orthogonal basis in higher dimensions, with high-frequency functions as said basis, enables the network to better model high-frequency variations of the input signal. The first commonly used embedding for coordinate networks was the in NeRF proposed Fourier feature embedding $\gamma(p)$, which mapped inputs $p \in \mathbb{R}$ to a higher dimensional space $\mathbb{R}^{2L}$:

$$\gamma(p) = (\ \sin(2^0 \pi p),\ \cos(2^0 \pi p),\ \cdots,\ \sin(2^{L-1} \pi p),\ \cos(2^{L-1} \pi p)\ ). \tag{3.8}$$

This function $\gamma(p)$ is used to encode viewing directions and point coordinates and is applied separately to each of the three coordinates. The scene within the neural volume must be normalized to lie in [-1, 1], as the periodic basis only allows bijection from $\mathbb{R}^n$ to and from this interval.

Training an MLP with embedded point coordinates necessitates that every image is taken at an equal distance from the object that they depict, and only allows alias-free

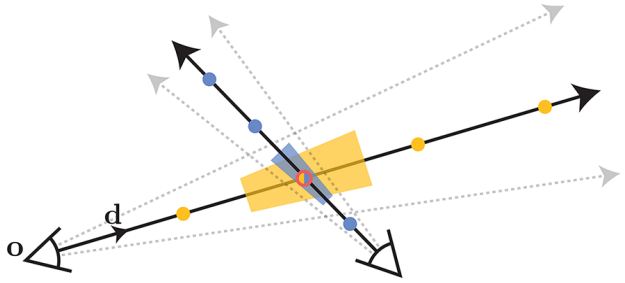

Figure 3.1.: NeRF samples and embeds discrete points (dots) along each pixel's ray, ignoring features such as ray interval length and the volume enclosed within, leading to significantly degraded performance. Mip-NeRF instead constructs conical sections from the ray intervals, which convey scale and enclosed volume to the MLP. Illustration copied from the original publication [Bar+21]

reconstructions for images from virtual cameras at said distance, as the neural volume is only trained on discrete paths and points. The obvious solution, i.e. super-sampling pixels, is computationally infeasible and instead has been resolved by taking inspiration from mipmaps in computer graphics.

The approach presented in [Bar+21], which they coined integrated positional encodings, encodes featurized volumes (ray cones) rather than point coordinates, and has shown to significantly reduce aliasing and increase accuracy at little computational cost. Essentially, using featurized conical sections as inputs, rather than just points, allows the network to reason about the size and scale of the inputs, resolving NeRF's insensitivity to scale and ameliorating most sources of aliasing.

4. Method

3D reconstruction from 2D images is a challenging problem as there can be numerous 3D configurations that can result in the same set of 2D images. In other words, the problem is under-constrained and lacks a unique solution. This is because 2D images only provide incomplete information about the 3D scene, limited to photometric traits like texture, shape, and lighting conditions. As a result, reconstructing the complete 3D geometry of a scene from 2D images can be challenging, especially when dealing with complex scenes, occlusions, or noisy input data.

The main objective of this study is to enhance novel view reconstruction outcomes by incorporating complementary monocular signals, such as depth- and normal maps, along with the photometric supervision signal. This helps to constrain the spatial positioning and surface orientation of objects in the depicted scene. To this end, we present a simple loss formulation that is derived from statistical considerations that can be easily tuned to work with synthetic and real data.

4.1. Depth Supervision with Statistical Weight Bounds

We assume a scene comprised of non-transparent objects. We model the scene with a neural field, $F_\theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$, that maps input coordinates $\mathbf{x}$ and viewing directions $\mathbf{d}$ to volume densities $\sigma(\mathbf{x})$ and colors $\mathbf{c}$. Into the scene, we project a camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, and we model the depth D at which the ray terminates, as the expectation of the distribution $f(t) = \mathcal{T}(t) \cdot \sigma(t)$, i.e. $D = \mathbb{E}[f(t)]$, with $f(t)$ being the likelihood that the ray stops at precisely t.

If we model $f(t)$ as a Gaussian distribution $f(t) = \mathcal{N}(\mu, \epsilon^2)$[1], then the expected value $\mathbb{E}[\mathcal{T}(t) \cdot \sigma(t)]$ is exactly the mode, i.e. the distribution is centered at $\mu = D$. The corresponding CDF $F(t)$ is easily found (in most statistics textbooks and also implemented in PyTorch). This CDF is the opacity function $F(t) = (1 - \mathcal{T}(t))$ and with this in mind, can be expressed as the cumulative sum of ray weights $\mathbf{w}$ in the discrete case. With $(1 - \mathcal{T})' = \mathcal{T}\sigma$ in mind we find:

[1] Usually this Gaussian scale parameter is symbolically represented by σ, but in this case, ϵ is chosen to distinguish it from the differential opacity (density) $\sigma(\mathbf{x})$ used to model the scene.

$$
\begin{aligned}
F(0 \to t_b) &= \int_0^{t_b} (1 - \mathcal{T}(t))' \, dt \\
&= (\mathcal{T}(0) - \mathcal{T}(t_b)) \\
&= (1 - \mathcal{T}(0 \ \to t_b)) \quad \text{constant density} \\
&= \sum_{i=1}^{N_b} w_i
\end{aligned}
\tag{4.1}
$$

In conclusion: With this assumption that $f(t)$ can be modeled as a Gaussian, and by knowing D and ϵ, the weights along the ray through the neural field are fully defined by the cumulative distribution function of the underlying (Gaussian) probability density function.

With this knowledge, we set ϵ as a hyper-parameter in a Gaussian that slightly overestimates the scale of $\phi(\mu, \epsilon^2)$, which we are using to model $f(t)$, and assign $\Phi(\mu, \epsilon^2)$ to the corresponding continuous CDF. The ray interval is then divided into two regions for ray interval midpoints t_i. The near region is denoted $near : t_i < D$ and the far region denoted $far : t_i > D$. The loss for interval-midsections $t_i \in near$ is then a simple mean squared error evaluated on ray weights w_i that exceed the bound given by $\Phi(t_i \in near)$, and for interval-midsections $t_i \in far$ we penalize weights w_i which subceed $\Phi(t_i \in far)$.

Furthermore, if this assumption holds, then the empirical rule states that 99.97% of the opacity that a ray encounters traveling through the neural field should lie within a region of $\pm 3\epsilon_\Phi$, i.e. the probability that the ray has not been absorbed at a depth of $D + 3\epsilon_\Phi$ is under 0.03%. By setting ϵ to a small value, we can ensure concentrated and compact surface presentations in the neural field, but must be careful to not constrict the volume too much, as smaller ϵ_Φ leads to steeper gradients in the optimization, and further, we hypothesize that the expressiveness of neural fields stems partially from the volume rendering of expansive surfaces, rather than ones with δ-shaped densities.

For practical reasons it makes sense to divide the *near* region further, and here we take inspiration from priors introduced in [Rem+22], who themselves followed [CL96; Che+21; Mat+00], where we define $\{t_{empty} : t < D - 3\epsilon_\Phi\}$, and redefine $\{t_{near} : D - 3\epsilon_\Phi \ < t < D\}$. The Loss term $\mathcal{L}_{\mathcal{CDF}}$ for the statistical weight bounds for the optimization of parameters θ in $F_\theta : (\mathbf{x}, \mathbf{d}) \to (\mathbf{c}, \sigma)$ can then formally be written as:

$$
\mathcal{L}_{\mathcal{CDF}} = \lambda_{empty} \mathcal{L}_{empty} + \lambda_\Phi \mathcal{L}_\Phi \tag{4.2}
$$

This re-partition of intervals allows for individual tuning of λ_{empty} and λ_Φ. This allows setting $\lambda_{empty} >> \lambda_\Phi$ whereby densities in the $empty$ region are strongly penalized and without creating training instabilities from exploding gradients of loss

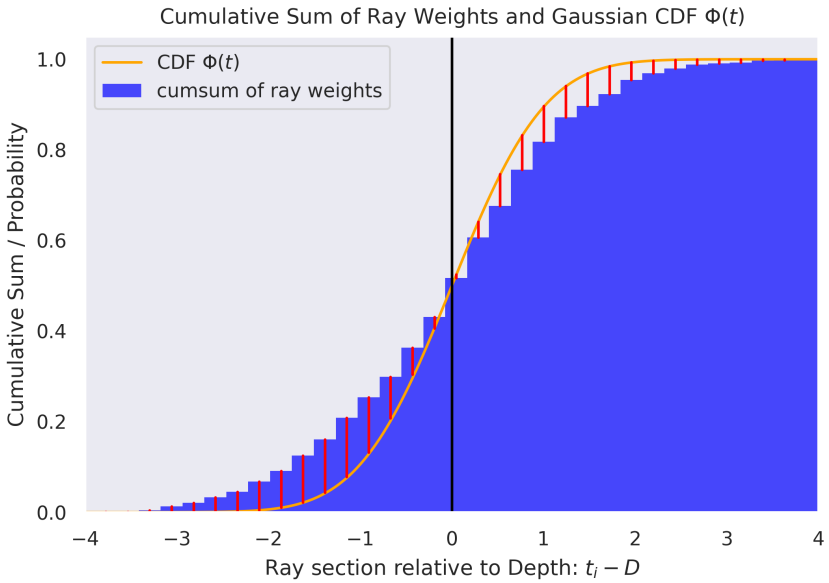

Figure 4.1.: Integrating over the likelihood $f(t)$, that a camera ray $\mathbf{r}(t)$ traversing the neural field F_θ along t terminates at exactly t, produces the Gaussian CDF $\Phi(t)$ if the densities along $f(t)$ are assumed to be normally distributed around D. The cumulative sum of ray weights can then be supervised by bounds given by $\Phi(t)$, where we penalize weights exceeding $\Phi(t)$ for $\{t_{near} : t - D < 0\}$, and for exceeding $\Phi(t)$ for $\{t_{far} : t - D > 0\}$.

incurred close to surfaces. With T_k as the number of indexes in each sum, the empty loss is defined as

$$\mathcal{L}_{empty} = \frac{1}{T_i} \sum_{t_i \in empty} w_i^2 \tag{4.3}$$

With $\mathcal{W}_i$ denoting the cumulative sum of weights w_i up to including i, the loss term for $\mathcal{L}_\Phi$ is then defined as follows:

$$\mathcal{L}_\Phi = \frac{1}{T_i} \sum_{t_i \in near} \max\{\mathcal{W}_i - \Phi(t_i), 0\}^2 + \frac{1}{T_j} \sum_{t_j \in far} \max\{\Phi(t_j) - \mathcal{W}_j, 0\}^2 \tag{4.4}$$

It should be noted that λ_Φ scales the loss contributed by each interval over the entire range linearly, while ϵ_Φ controls the slope of the Gaussian CDF in a non-linear way, and thereby the scale of how quickly deviations incur a penalty on the optimization.

As with most deep learning parameters, it is worth tuning ϵ_Φ to the application, as

setting it too low can lead to nonoptimal density distributions impacting performance, while setting it too high makes benefits from depth supervision weaker. Finally, far is set to being an open interval, as it encourages fully opaque surfaces in the vicinity of the depth measurement, but we found that the impact over limiting it to a multiple of ϵ_Φ was negligible.

4.2. Adaptation to Real Data

Real data generated through physical processes are never perfect, and as such, we introduce a slight modification of the method. To model measurement uncertainty, we introduce an offset β. Instead of evaluating the bounds with one Gaussian CDF centered at D, we evaluate for the bounds two separate Gaussian CDFs positioned at $D \pm 2\epsilon_\Phi$ as depicted in Figure 4.2, such that, within a tolerance of the estimated depth uncertainty, the model's predicted densities aren't erroneously constrained.

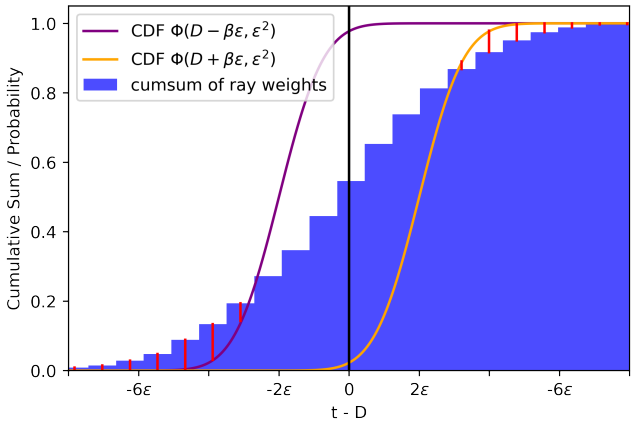

Figure 4.2.: To account for real-world sensor noise, and improper camera calibration, we adapt the loss presented in Figure 4.1 by introducing a parameter β, which acts as an X-Axis offset and describes the expected measurement error. We extend *near* and far regions by $\pm\beta\epsilon$ respectively. By supervising points in *near* with the upper bound given by $\Phi(D-\beta\epsilon,\epsilon^2)$, and points in far with the lower bound given by $\Phi(D+\beta\epsilon,\epsilon^2)$, the impact of measurement errors is effectively limited. In all real-world experiments we set $\beta = 2$ and for synthetic experiments, we set $\beta = 0$. Lower β generally leads to better results, if depth and camera data are well-calibrated and accurate.

4.3. Normal Supervision

We investigate methods to include supervision with normal maps as an additional, complementary monocular cue. We evaluate supervising the networks density normals $\mathbf{n}_\sigma(\mathbf{x})$ directly. To generate the density normals $\mathbf{n}_\sigma(\mathbf{x})$ we differentiate the densities, generated by the NeRF-MLP at sampling locations along a ray, in respect to their input coordinates $\mathbf{x}$. This is formally written as: $\mathbf{n}_\sigma(\mathbf{x}) = -\nabla_\mathbf{x}\sigma(\mathbf{x})$. For the direct supervision, we use the weighted euclidean distance between normalized normal vectors:

$$\mathcal{L}_{n_{dir}} = \sum_i w_i \|\mathbf{n}_\sigma(\mathbf{x}) - \mathbf{n}_{GT}(\mathbf{x})\|^2 \tag{4.5}$$

Note that minimizing the euclidean distance between $\mathbf{n}_\sigma(\mathbf{x})$ and $\mathbf{n}_{GT}(\mathbf{x})$, is related to minimizing the cosine similarity as follows:

$$\begin{aligned} \|A - B\|^2 &= (A - B) \cdot (A - B) && (4.6)\\ &= \|A\|^2 + \|B\|^2 - 2(A \cdot B) && \text{(polarization identity)}\\ &= 2(1 - \cos(A, B)) && \text{(unit norm)}\\ \|A - B\|^2 &= 2Dist_{cos}(A, B) \quad when \quad \|A\| = \|B\| = 1. && (4.7) \end{aligned}$$

We also investigate if indirect supervision has a tangible effect on PSNR, by placing a dense layer after the last layer of the NeRF-MLP to predict normals as an extra output, as introduced in Ref-NeRF [Ver+22], denoted $\mathbf{n}_\theta(\mathbf{x})$. This, according to the authors, *"..produces smoother normals than gradient density normals because the gradient operator acts as a high-pass filter on the MLP's effective interpolation kernel"*[Ver+22], and supervise:

$$\mathcal{L}_{n_{indir}} = \sum_i w_i \|\mathbf{n}_\theta(\mathbf{x}) - \mathbf{n}_\sigma(\mathbf{x})\|^2 + \sum_i w_i \|\mathbf{n}_\theta(\mathbf{x}) - \mathbf{n}_{GT}(\mathbf{x})\|^2 \tag{4.8}$$

4.4. Implementation details

Architecture and training. The architecture used for these experiments is a slightly modified Mip-NeRF, where "coarse-to-fine" sampling has been replaced by two rounds of sampling with a proposal MLP exactly as proposed in Mip-NeRF360° [Bar+22]. The proposal MLP has 4 layers and 256 Hidden units, and the NeRF MLP has 8 layers and 256 Hidden units, the model in total has 835K Parameters.

We train at half-precision with a batch-size of 8192 and the Adam optimizer [KB14] with standard PyTorch values for $\epsilon_{adam} = 1e^{-8}$ and $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate follows the schedule suggested in Mip-NeRF360°, i.e. 512 warm-up steps,

$2e^{-3}$ maximum learning rate, which is log-linearly annealed to $2e^{-5}$ over the course of training, with the maximum number of steps dependant on the number of training views. All training takes place on a GTX2070 consumer GPU and takes from 30 minutes to 12 hours depending on the scene.

Sampling. We collect 64 stratified samples linearly spaced in z-space, and the inferred densities are normalized and re-sampled by inverse transform sampling (2 proposal rounds). From the final round of proposal probabilities, 32 samples are again generated for inference by the NeRF-MLP. The weights $\hat{w}^k$ in intervals $\hat{t}^k$ from both proposal-rounds $k \in \{1, 2\}$ are supervised by thresholding the histograms as proposed in Mip-NeRF360°. The bound function computes the sum of all proposal weights $\hat{w}^k$ in proposal intervals $\hat{t}^k$ that overlap with interval T:

$$\text{bound}(\hat{\mathbf{t}}, \hat{\mathbf{w}}, T) = \sum_{j:T \cap \hat{T}_j \neq \emptyset} \hat{w}_j \tag{4.9}$$

Any interval that violates the bound given by $w_i \leq \text{bound}\left(\hat{\mathbf{t}}, \hat{\mathbf{w}}, T_i\right)$, i.e. that has surplus histogram mass in a given interval-range, is penalized. We bound for all intervals and weights (T,w_i) that are the result of the final forward pass through NeRF MLP:

$$\mathcal{L}_{\text{prop}}\left(\mathbf{t}, \mathbf{w}, \hat{\mathbf{t}}, \hat{\mathbf{w}}\right) = \sum_i \frac{1}{w_i} \max\left(0, w_i - \text{bound}\left(\hat{\mathbf{t}}, \hat{\mathbf{w}}, T_i\right)\right)^2 \tag{4.10}$$

Loss. We supervise the network with the following total loss:

$$\mathcal{L} = \mathcal{L}_{Chb} + \lambda_\Phi \mathcal{L}_\Phi + \lambda_{empty} \mathcal{L}_{empty} + \lambda_n \mathcal{L}_n + \lambda_{prop} \sum_i^k \mathcal{L}_{prop} \tag{4.11}$$

The hyper parameters for synthetic experiments are $\lambda_{prop} = 1.0$, $\epsilon_\Phi = 0.03$, $\lambda_{empty} = 1.0$ and $\lambda_\Phi = 0.1$ for few views ($n \leq 12$) and $\lambda_\Phi = 0.01$ for many views ($n > 12$). For experiments investigating the benefits of normal map supervision we set $\lambda_n = 1e^{-4}$, and unless otherwise mentioned, set $\lambda_n = 0$ in all other experiments.

For real-world experiments, ϵ_Φ is set to be a subtle overestimate of the error in registering the depth map, i.e. 0.5% - 1.5% of the depth maps value. Finally, lower values for ϵ_Φ generally lead to more accurate depth inference, unless it underestimates the measurement errors. A good rule of thumb is to increase or decrease λ_{near} and ϵ_Φ together as they figuratively act like slope and bias.

4.5. Datasets and Metrics

Synthetic experiments. We use the Blender dataset with ground truth camera parameters and constant illumination introduced in NeRF[Mil+21]. The dataset is re-rendered at 400x400 resolution with scripts, provided by NeRF's authors, to generate ground-truth depth- and normal- maps. We evaluate the average PSNR of novel views (test views) in dependence on the number of views supplied at training (train views).

Real-world data experiments. We use scenes from ScanNet [Dai+17]. Scenes are trained with 15 to 25 images per scene. As data preparation, we center-crop dark borders (results from un-distorting images) and infer camera parameters (extrinsic and intrinsic) via COLMAP.

We use the RGB-D ground truth data for pixels wherever available, and replace missing measurements by predicting monocular depth- and normal- maps with the help of a pre-trained monocular depth and normal estimator model: OmniData [Eft+21]. We fit and scale those depth maps by solving a least squares regression system for valid pixels in the sensor's depth map and update invalid sensor readings with the transformed inferred values.

Evaluation and Metrics. For the evaluation of synthetic data, we compare the PSNR (Peak Signal to Noise Ratio) of the novel views to the ground truth using common methods that incorporate depth maps as a supervision signal for training neural fields, as well as a baseline model without our proposed statistical weight bounds (SWB). When evaluating the real-world dataset ScanNet, we calculate the mean square root error (MSRE) of the reconstructed depth maps on valid pixels and assess the PSNR and SSIM (structural similarity index measure) of predicted views in comparison to previous studies. To address varying lighting conditions (as shown in Figure 4.3) we also present our color-corrected reconstruction metrics.

4.6. Global Latent Optimization

The real-world scenes in ScanNet can show great photometric variations from image to image, even when depicting the same objects (see Figure 4.3). This most likely stems from the camera's auto exposure or auto white balance being allowed to vary between images. This makes photometric reconstruction an ill-posed task, as we are not interested in modeling the most likely exposure of each pose, but rather want to synthesize novel views that look consistent, pleasing, and believable.

To allow the model to disentangle viewing direction and camera exposure, we pass the camera index into an $Nx4$ dimensional embedding layer during training (N is the number of total training views). During the evaluation, we set this embedding to zero,

Figure 4.3.: Real-world datasets can exhibit strong photometric variation when depicting the same region in the same scene, leading to difficulties in synthesizing consistent novel views and lower PSNR scores for the reconstruction of test views. To remedy the impact of this effect, we embed a per-camera optimizable embedding at train time, to disentangle an image's camera exposure and the general scene illumination, from the camera's viewing direction. In addition, we additionally perform a least squares linear fit from predicted images to ground truth images, to give a better indication of reconstruction quality, and list them with the "color corrected" tag.

which gives us consistent lighting conditions and a pleasant appearance across all validation views, that are close to the average lighting conditions.

Generating predictions like this produces pleasing and consistent results, but perform poorly on PSNR. We adopt the strategy proposed in [Bar+22] and additionally evaluate our "color-corrected"[2] novel views. We solve the least squares fit of RGB values from predicted images to ground truth images, to better match the output images, and to give a fairer comparison. Note that if we were interested in retrieving the exact lighting conditions at the time the data was recorded, we could just optimize the camera's latent embedding at test time, but that is not the goal of this technique. We interpret the color-predicted PSNR and SSIM results as a lower bound on the model's true reconstruction capabilities, as it is nothing more than an affine transformation of the output images.

[2]Unless otherwise indicated the results presented are generated without color-correction.

5. Experimental Results

In this chapter, we evaluate the qualitative and quantitative benefits of training neural radiance fields on RGBD(-N) data, i.e. to what extent the training of neural fields can be enhanced by incorporating depth- and normal-map supervision as additional monocular cues during training.

We demonstrate superior results and faster training speeds by incorporating depth with the novel loss formulation derived in chapter 4. We first evaluate the method on the synthetic Blender dataset presented in the original NeRF paper [Mil+21], and then contrast it to popular methods for dealing with depth data in neural radiance fields, specifically rendered depth, and depth carving from Urban Radiance Fields [Rem+22] (see loss Appendix B for their respective definitions).

Further, we investigate methods to include supervision with normal maps as a complementary monocular cue. We evaluate supervising the network density normals, i.e. $\mathbf{n}_\sigma(\mathbf{x}) = -\nabla\sigma(\mathbf{x})$ directly, and indirectly with an intermediate MLP, as introduced in Ref-NeRF [Ver+22], denoted $\mathbf{n}_\theta(\mathbf{x})$ (see Equation 4.8).

5.1. Synthetic Scenes

5.1.1. Investigating Depth Supervision

For initial tests, we use the synthetic Blender dataset and evaluate the average PSNR of novel views (test views) in dependence on the number of views supplied at training (train views). We demonstrate (Figure 5.1) that the presented method improves results of novel views, **regardless of the number of train views**, with diminishing returns as the number of train views becomes very large. Further, it allows for higher learning rates and thus converges much faster, taking as little as 15 minutes on a consumer GPU to achieve 23.8 Test-PSNR on the Lego Blender scene with only 3 input training views, and using only a slightly modified Mip-NeRF.

To compare the presented method with other works, we perform a binary random search to find a suitable configuration for loss hyper-parameters λ_i for the few view setting ($n \leq 12$). For the simple Rendered Depth, we only have to find an acceptable λ_D to which we commit 5 trials, for Urban Radiance Fields we commit 40 trials, as it

features a plethora of hyperparameters that all interact[1]. For our presented method, we find that the parameters are fairly insensitive and set them as described in chapter 4.

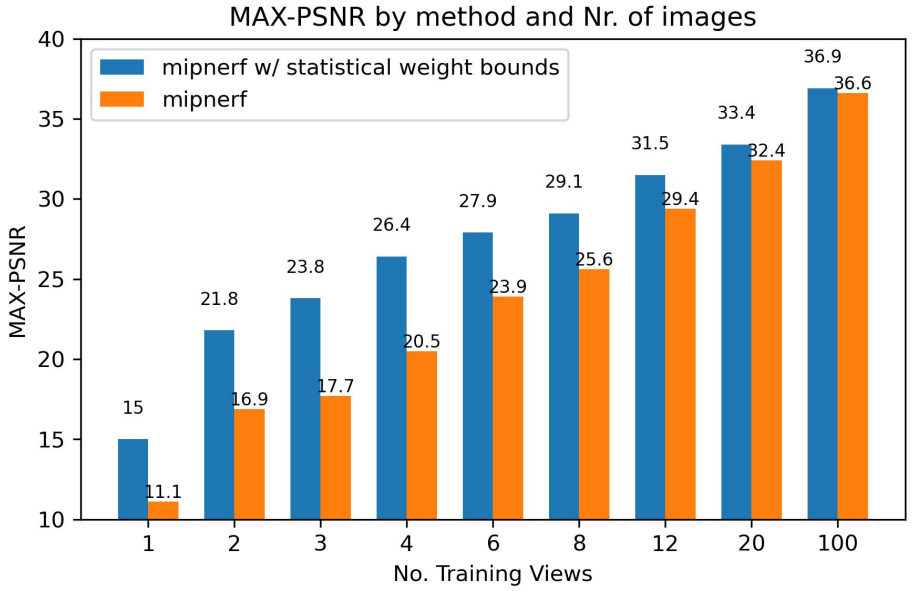

Figure 5.1.: Contrary to other common methods for using depth data in neural fields, our proposed bounded weight loss improves reconstruction results for any number of training views. Even with as few as two views we can attain a satisfying 3D consistent result.

We demonstrate the advantages of our method on synthetic data and show that it quantitatively (Table 5.1) and qualitatively (Figure 5.2, Figure 5.3) outperforms prior work aiming to incorporate depth map data in the training of neural fields. We find that ours is the only method that consistently improves test PSNR with RGB-D data, contrary to previous works that improve PSNR of reconstruction at fewer views and negatively impact PSNR of reconstructions when the scene is well defined.

Figure 5.2 provides insight into failure modes of current approaches. Supervision with the rendered depth leads to nonsensical predictions, i.e. volumes of white densities, floaters and, generally in areas that do not contain meaningful information, to minimize the loss incurred when summing over the weighted intervals. URF reduces the solution space too much and does not consider that objects in 3D are expansive, leading to inconsistent geometries in novel views when supervision is sparse.

[1] ϵ_{max}, ϵ_{min}, λ_{empty}, λ_{near}, and $\dot{\epsilon}$, $\ddot{\epsilon}$, where the latter two signify the rate and mode (linear, exponential) of annealing the ϵ-interval

	PSNR		
No. Views	3	6	100
Mip-NeRF	17.73	23.87	36.65
Rendered	20.32	25.90	35.09
URF	20.09	26.12	36.36
Ours	23.81	27.91	36.96

Table 5.1.: Comparison of model performance. While depth supervised methods (Rendered depth, Urban Radiance Fields [Rem+22]) improve results over the base model for few views, they adversely affect reconstruction quality when supervision is dense. Our loss by contrast consistently improves reconstruction results.

We hypothesize that our method encourages correctly initialized densities in regions that are multiview consistent, while simultaneously discouraging floaters and density in general in regions that we know to be empty. The key advantage over URF's formulation is the consideration of bounded intervals that let the network retain expansive volumes that are expressive and multiview consistent, rather than constricting entire volumes, supervised by discrete pixel-wise quantities, to predict δ-shaped densities.

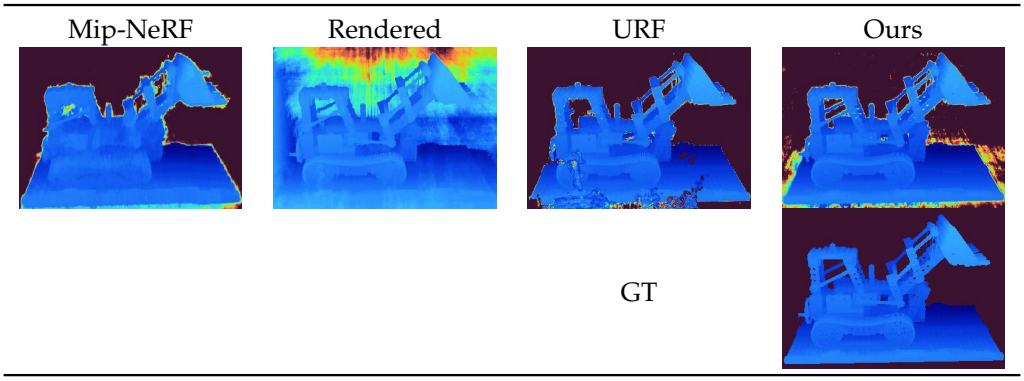

Figure 5.2.: Qualitative inspection of predicted depth maps during 3-view training shows how different densities are enforced. Mip-NeRF assigns irregular densities that purely minimize reconstruction loss from 3 views, and therefore don't have to be constrained to surfaces. Rendered Loss assigns semi-transparent and white densities everywhere to ensure the weighted training rays sum to their respective depth values. URF overfits to training views and enforces δ-shaped densities, while ours models very fine geometry (see holes in the Lego model) with minimal supervision.

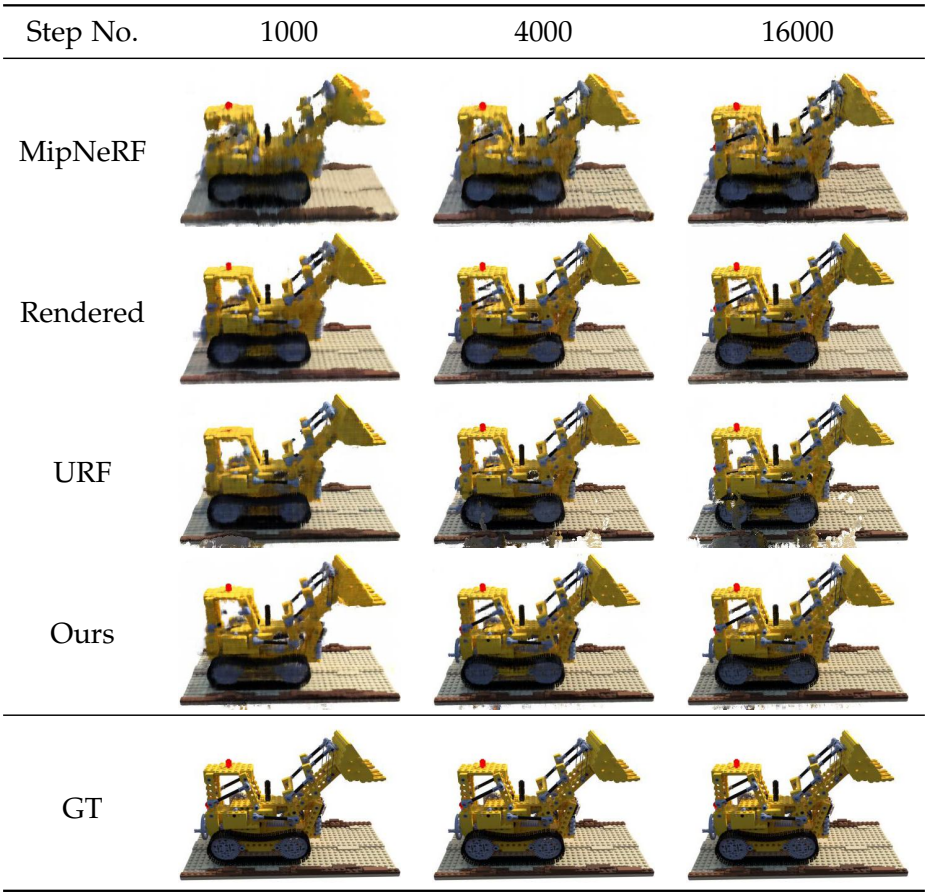

Figure 5.3.: Qualitative inspection of validation views during training with 3 views shows major improvements when training with depth maps for all of the studied methods. While Mip-NeRF struggles to constrain densities to sharp regions, the alternatives present richer details. When trained with the Rendered Depth as supervision, the model cannot reconstruct fine geometries (e.g. holes in the tractor arms), while training with depth carving (URF) enforces δ-shaped densities and leads to visible artifacts in novel views. Our method creates plausible and multiview consistent novel views with only 3 training views as supervision, by correctly localizing densities in space and allowing for expansive surfaces. Even with expansive surfaces it is able to model fine details (holes in the tractor) and shows a massive increase in convergence speed.

5.1.2. Benefits of Normal Supervision

With our architecture, a slightly modified Mip-NeRF, we find little evidence that supervising with normal maps in addition to depth consistently increases test-PSNR, but note a 50% increase in training time, from calculating an additional backward-pass through the network to compute $\mathbf{n}_\sigma(\mathbf{x})$. Supervising the gradient normals indirectly, by training a network to predict the density normals, $\mathbf{n}_\theta(\mathbf{x})$, leads to a stronger degradation of performance than supervising $\mathbf{n}_\sigma(\mathbf{x})$ directly.

It is possible, that changing the base model to the architecture presented in Ref-NeRF, where $\mathbf{n}_\theta(\mathbf{x})$ is physically motivated and used to correctly model reflections in shiny blender, a dataset populated with non-Lambertian materials, could make use of said normals to enhance reconstruction performance. [Yu+22] has found normal maps to be a useful monocular cue that improved reconstruction quality, but they model the scene as an SDF field, with δ-thickness surfaces, where the normals directly correspond to the gradient of the underlying field. It is possible that this, or other architectural innovations, extracts greater benefit from the information provided by normal maps.

	PSNR		
No. Views	3	6	100
RGB	17.73	23.87	36.65
RGB-D	23.81	27.91	36.96
RGB-N_σ	17.62	23.99	-
RGB-D-N_θ	21.31	27.52	36.70
RGB-D-N_σ	22.12	28.86	36.88

Table 5.2.: Comparison of model performance. Our proposed depth loss does not convincingly profit from an additional term for normal maps when views of the scene densely overlap but shows a slight increase in performance for semi-sparse view-overlap. Training on RGB-N inputs alone is inferior to RGB-D but slightly better than the just RGB baseline. Supervising volume density with normal maps could offer downstream benefits if the normals predicted are used for inference of secondary effects (e.g. reflections).

While not directly beneficial to novel view synthesis, supervising the model to predict normals n_θ with normal maps lets the model learn accurate surface normals[2] at low penalty ($< 1\%$ PSNR). These normals might be interesting for downstream tasks like reflections and relighting of neural scenes.

[2] see Appendix C for an example of MLP-predicted surface normals

5.2. Real World Scenes

There are several differences between real and synthetic datasets that are important to consider when training a neural radiance field, which could impact performance negatively left unaccounted.

- Diversity: Real images can have large variability in terms of lighting conditions, background, and object appearance.
- Quality: Real images may contain noise, blur, or other artifacts.
- Inaccurate: Camera parameters retrieved with COLMAP may be inaccurate and lead to visual artifacts in NeRF.
- Defective: Depth sensors are imperfect devices and are usually accurate up to a fraction ($\approx 1\%$) of the measured value (see Figure C.5).
- Incomplete: Sensors are bounded to a minimum and maximum possible distance they can measure. Pixels depicting non-Lambertian materials, transparent media, and out-of-range objects give invalid readings and lead to an incomplete depth map.

5.2.1. Ablations

With this in mind, we adapt our method to account for inaccurate camera initializations and sensor error in depth map measurements by introducing the parameter $\beta = 2\epsilon_\Phi$ (Figure 4.2). Invalid depth pixels are replaced by the affine-transform of a depth map predicted from a Monocular Depth Estimation network [Eft+21](see Appendix C for an example), which we refer to as "Stitching". Further, we follow the example of other works [Bar+22; Mar+21] and introduce a low-dimensional per camera embedding to model varying camera exposure and white balance (see section 4.6 for details). Further, we list metrics for "color-corrected" images as described in section 4.6. We ablate and compare our design choices in Table 5.3.

Our proposed method generates consistent and pleasing novel views and scores high on all metrics, especially those less sensitive to matching the lighting conditions of the validation image (RGB-Normalized and Depth RSME).

No-GLO performs best for SSIM and uncorrected PSNR, but qualitative inspection of the generated views reveals rapid lighting changes when panning the room, as the model has learned to correlate viewing direction and most likely illumination, effectively gaming the validation metrics. Such a model seems of little use for our application, even if it performs better on color-dependant metrics because it produces

Method	PSNR ↑	SSIM ↑	Depth RSME ↓	RGB-Normalized PSNR ↑	RGB-Normalized SSIM ↑
No-Depth	19.50	0.6898	0.6307	21.40	0.6926
Rendered Depth	20.57	0.7341	0.0691	23.20	0.7368
No-GLO	22.10	0.7632	0.0803	23.99	0.7704
No-β	21.06	0.7486	0.0699	23.89	0.7524
No-Stiching	21.21	0.7524	0.0453	23.98	0.7523
No-Empty	21.18	0.7532	0.1652	24.03	0.7565
Ours	21.18	0.7524	0.0489	24.08	0.7574

Table 5.3.: Ablation of model design and their effects on performance. Color-legend: red/best, orange/second best, and yellow/third best-performing model. Our proposed method performs well across all metrics, especially metrics less sensitive to the lighting conditions of the test set. Evaluation on ScanNet-Scene0710_00.

unappealing and inconsistent novel views (see Appendix C for examples of failure cases for No-GLO and No-Stitching).

Other well-performing variants are more difficult to train and often exhibit some mode of failure. Without stitching of depth maps (**No-Stitching**), scenes with very incomplete depth maps, or with objects in front of distant backgrounds, do not converge, as densities are not correctly initialized. The **No-Empty** variant is littered with floaters that negatively affect its depth RMSE.

We want to emphasize that the normalized test metrics should be seen as a lower bound on true model performance and that test-time optimization of the GLO embedding would perform even better. Indeed while the gaming of test metrics is of little practical use, the idea itself allows for artistic expression, as we can effectively change the scene's lighting conditions and the illumination of objects depicted by interpolating in the camera's embedding space, which concurrent works do explore [Mar+21], and which might be an attractive direction for future work.

5.2.2. ScanNet: Novel View Synthesis Results

We evaluate our best model on ScanNet with identical data and poses that were provided by Dense Depth Priors [Roe+22]. This allows us to directly compare it to their tabulated benchmark, although we have to discern the relative improvements stemming from using Mip-NeRF over NeRF (see Table 5.3 for the impact on scores from architectural improvements), and from our novel loss formulation over prior work. We demonstrate that our model outperforms prior work by a large margin, especially in metrics insensitive to photometric variations between test and training views. A core difference however is that most of these prior methods use sparse depth as input,

although this anecdotally only presents a marginally more challenging setting, because we only need a scale and offset parameter to initialize a good proxy via "Stitching". The normals and depth maps inferred by our model are multi-view consistent and extremely accurate (Appendix C for visualizations), the calculated RMSE on valid depth map pixels is close to the sensor's relative inaccuracy ($\approx 1\%$) and, could be useful for downstream tasks.

Method	PSNR ↑	SSIM ↑	Depth RSME ↓	RGB-Normalized PSNR ↑
NeRF[Mil+21]	19.03	0.670	1.163	
DS-NeRF[Den+22]	20.85	0.713	0.447	
NerfingMVS[Wei+21]	16.29	0.626	0.482	
DDP[Roe+22]	20.96	0.737	0.236	22.30[3]
Ours	21.33	0.7923	0.0753	23.89

Table 5.4.: Evaluation of our model against the tabularized benchmark of methods on the ScanNet dataset provided by [Roe+22](includes Scenes:0710_00, 0758_00, 0781_00). Our model outperforms prior work across all metrics, but especially for depth prediction.

5.3. limitations

Our proposed method does little to compensate for noisily initialized cameras and in that regard fails in the same way previous NeRF works do. A promising future research direction is the simultaneous optimization of cameras and scene as presented in other works [Lin+21]. In addition, specular reflections, bright light sources, and non-Lambertian materials are poorly reconstructed and often show up as artifacts on validation views. This usually requires dense supervision to be accurately modeled, and a change of the architecture, on which other works have shown tremendous progress [Ver+22]. We also note that the performance of models can be ambiguous when optimizing for color-sensitive metrics, but it is a flaw that can be remedied by careful modeling of camera behavior during training or dataset acquisition.

Finally, we see less beneficial impact of normal map supervision than concurrent work [Yu+22], which might be due to architectural differences[4], but this fact leaves

[3]Dense Depth Priors optimize a per camera latent embedding at test time

[4]The authors of mono-SDF regress an SDF instead of a neural radiance field. Additionally, they regularize that SDF to have unit gradient norm with the Eikonal equation, where they can directly supervise both predicted surface normals and norm with normal supervision. SDFs are also a smoother approximation than NeRFs, which might lead to more stable training

room to speculate that there may be some way of making use of normal supervision for neural radiance fields.

6. Conclusion

In this thesis, we introduce a novel loss formulation for RGB-D data, based on the principles of neural radiance fields and a Gaussian modeling of the likelihood for the ray termination in neural volumes. Our work directly addresses deficiencies of prior work that use depth maps as supervision for the training of neural radiance fields. The formulation is easy to understand, the code is simple to implement, and the hyper-parameters are effortlessly and robustly tuned.

We found spurious evidence that combined normal- and depth map supervision provides benefits for sparse-view supervision, but note that the performance penalty for normal supervision is typically insignificant. It may be worthwhile to incorporate normal supervision if downstream applications or architectural changes can benefit from predicted normals, but find little use for Mip-NeRFs theoretical framework.

We find that our presented method addresses a particular subset of issues with real-world data, i.e. depth uncertainty and expansive volumes, but that there is still much to do to make NeRFs robust to real-world data. We think that room-scale novel view reconstruction would benefit greatly from more accurate camera initializations, or a more robust framework that jointly optimizes cameras and neural radiance field. NeRFs are also ill-suited for the photometric variations common to indoor datasets, and we hypothesize that frameworks predicting or modeling camera behavior could be beneficial to modeling real-world scenes.

We hope that our findings here may be of use for future work aiming to improve the training of NeRFs with additional monocular cues.

A. Evaluation Metrics

A.1. MSE and RMSE

Mean Squared Error (MSE) is a widely used loss function for regression problems in machine learning and computer vision. MSE measures the difference between the true target and predicted values by computing the mean of the squared differences between the two sets of values. The MSE loss is used to quantify the quality of the predictions and to guide the optimization process in machine learning models.

Mathematically, given a set of true target values t_i with $i \in [1, .., N]$ and corresponding predicted values y_i the MSE is defined as follows:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (t_i - y_i)^2$$

The use of MSE as a loss function is motivated by its simplicity and its ability to provide a single scalar value that summarizes the quality of the predictions. The MSE loss is sensitive to outliers and is often used in combination with other loss functions to balance the trade-off between robustness and sensitivity.

Similarly the RMSE is defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (t_i - y_i)^2}$$

The use of RMSE as a performance metric is motivated by its ability to provide a more interpretable measure of the prediction ,error in terms of the original units of the target values, making it easier to compare the performance of different models on different datasets.

A.2. PSNR

PSNR (Peak Signal-to-Noise Ratio) is a widely used quality metric for image and video compression and restoration. PSNR measures the similarity between a reference and a distorted image by computing the mean squared error between the two images and then transforming this error into decibels (dB) for its logarithmic scale. The higher the PSNR value, the more similar the two images are, and the better the quality of the distorted image. Mathematically, given two images A and B, PSNR is calculated as follows:

$$\text{PSNR}(A, B) = 10 \log_{10} \left(\frac{MAX^2}{\text{MSE}} \right)$$

where MAX is the maximum possible pixel value for the given image format (for example, 255 for 8-bit grayscale images).

The use of PSNR as a quality metric for image and video compression is motivated by its ability to provide a simple, easily interpretable measure of image quality that is widely used in both academia and industry. Its popularity is due in part to its simplicity and its ability to provide a single scalar value that summarizes the quality of an image or video.

A.3. SSIM

SSIM (Structural Similarity Index) is another commonly used image quality metric that measures the similarity between two images by comparing the structural information in the images. SSIM takes into account not only the mean and variance of the image pixels, but also their cross-correlation, which helps to capture the spatial relationships between pixels. Mathematically, given two images A and B, SSIM is calculated as follows:

$$\text{SSIM}(A, B) = \frac{(2\mu_A\mu_B + c_1)(2\sigma_{A,B} + c_2)}{(\mu_A^2 + \mu_B^2 + c_1)(\sigma_A^2 + \sigma_B^2 + c_2)}$$

where μ_A, μ_B, σ_A, σ_B, and $\sigma_{A,B}$ are the local mean, standard deviation, and cross-covariance between the two images, respectively, and c_1 and c_2 are constants to ensure that the SSIM index is well-behaved.

The use of SSIM as a quality metric for image and video compression is motivated by its ability to provide a more accurate and perceptually meaningful measure of image quality than simple mean squared error metrics like PSNR. It is widely used in academic research and industry applications and has been shown to provide a better correlation with human perception of image quality than PSNR.

B. Loss Functions

B.1. MSE RGB Loss

Standard NeRF pipelines use a MSE loss on predicted pixel colors $\hat{\mathbf{c}}_i$ and ground truth pixel colors $\mathbf{c}_i$ for every ray i in a mini-batch of N rays:

$$\mathcal{L}_{L2-color}(\mathbf{c}, \hat{\mathbf{c}}) = \sum_{i=1}^{N} (\mathbf{c}_i - \hat{\mathbf{c}}_i)^2 \tag{B.1}$$

B.2. Rendered Depth

The rendered depth $\hat{\mathbf{d}}$ is calculated as the dot product between ray intervals $\mathbf{t_i}$ and ray weights $\mathbf{w_i}$ for every ray i in a mini-batch of N rays. The loss is then the MSE loss over all rays in the batch:

$$\mathcal{L}_{L2-depth}(\mathbf{d}, \hat{\mathbf{d}}) = \sum_{i=1}^{N} (d_i - \mathbf{w}_i \mathbf{t}_i)^2 \tag{B.2}$$

B.3. Urban Radiance Field: Depth Carving

$$\mathcal{L} = \lambda_{L2-depth}\mathcal{L}_{L2-depth} + \lambda_{near}\mathcal{L}_{near} + \lambda\mathcal{L}_{empty} \tag{B.3}$$

$$\begin{aligned} \mathcal{L}_{empty} &= \int_{t_n}^{z-\epsilon} w(t)^2 dt \\ \mathcal{L}_{near} &= \int_{z-\epsilon}^{z+\epsilon} (w(t) - \mathcal{G}(t-z))^2 dt \\ &\text{with } \mathcal{G}(t)\text{: Gaussian}(0,\ (\epsilon/3)^2) \end{aligned} \tag{B.4}$$

C. Visualizations

Figure C.1.: Failure mode: Inconsistent looking views without per-camera embedding. The algorithm games the train PSNR by predicting illumination conditioned on viewing direction. Also visualizes an incorrectly initialized camera from COLAMP, leading to artifacts in the reconstruction.

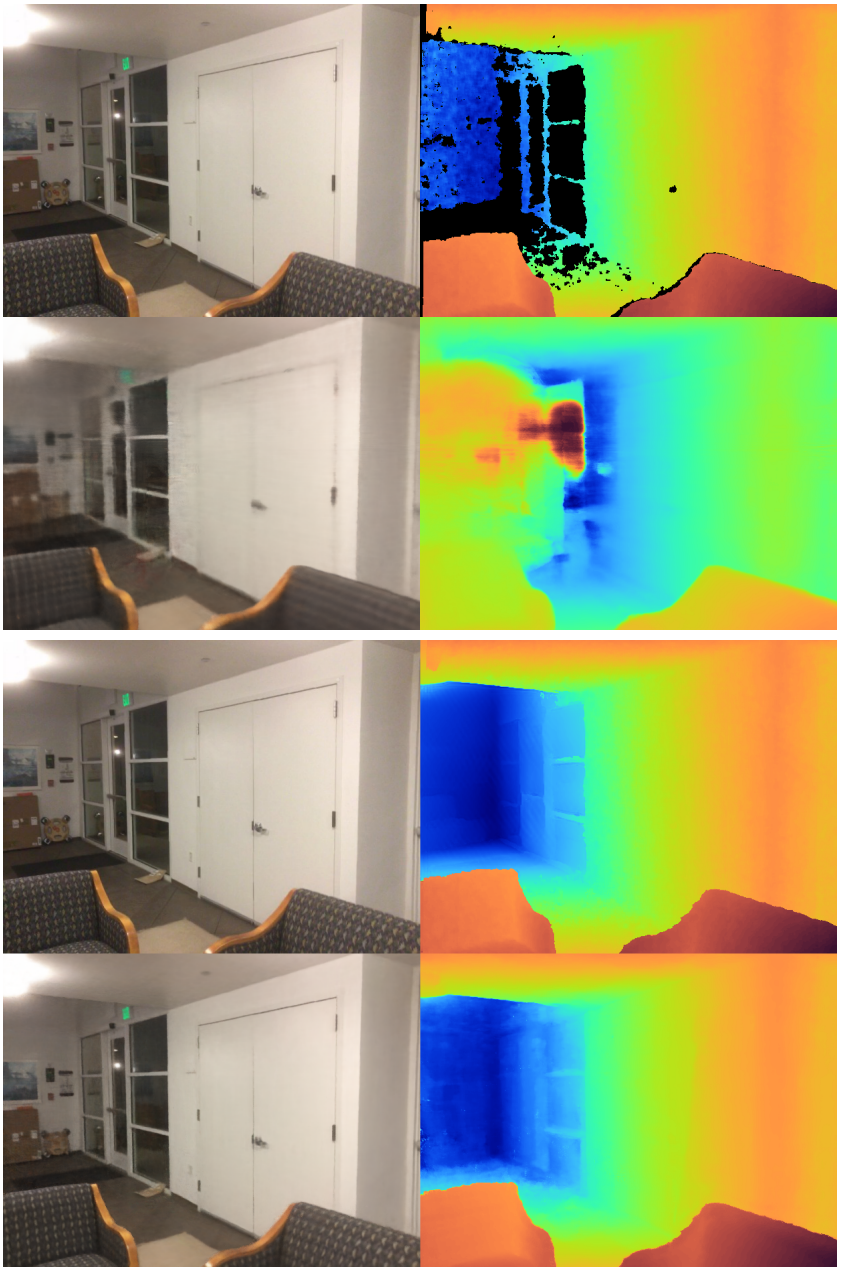

Figure C.2.: Failure mode: Without stitching (top) the network fails to reconstruct scenes with high z-axis differences. Stitching (bottom) allows for correct initialization and helps the model to predict accurate geometries. From top to bottom: GT, Pred, GT, Pred.

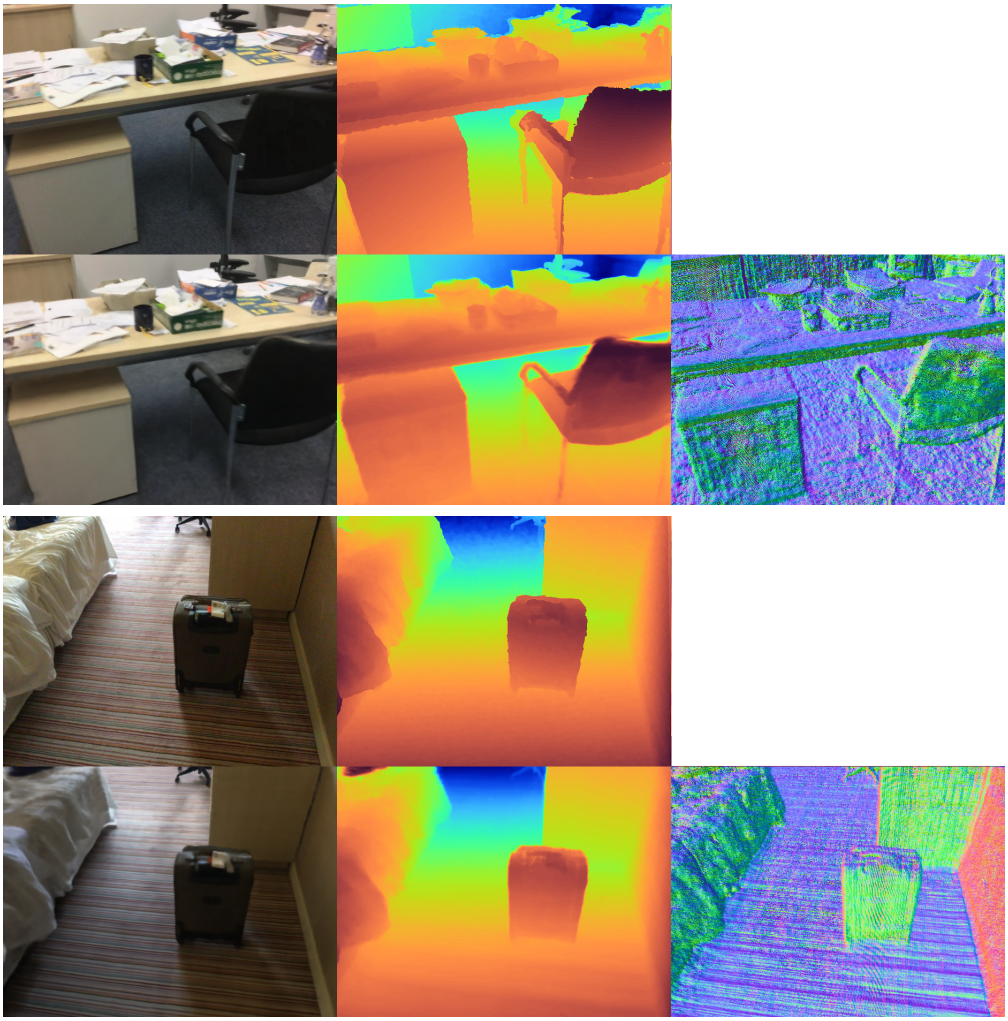

Figure C.3.: Algorithm predicts extraordinary high precision depth and normal maps that could be used for downstream tasks or that could serve as better depthmap for iterative training. From top to bottom: GT, Pred, GT, Pred.

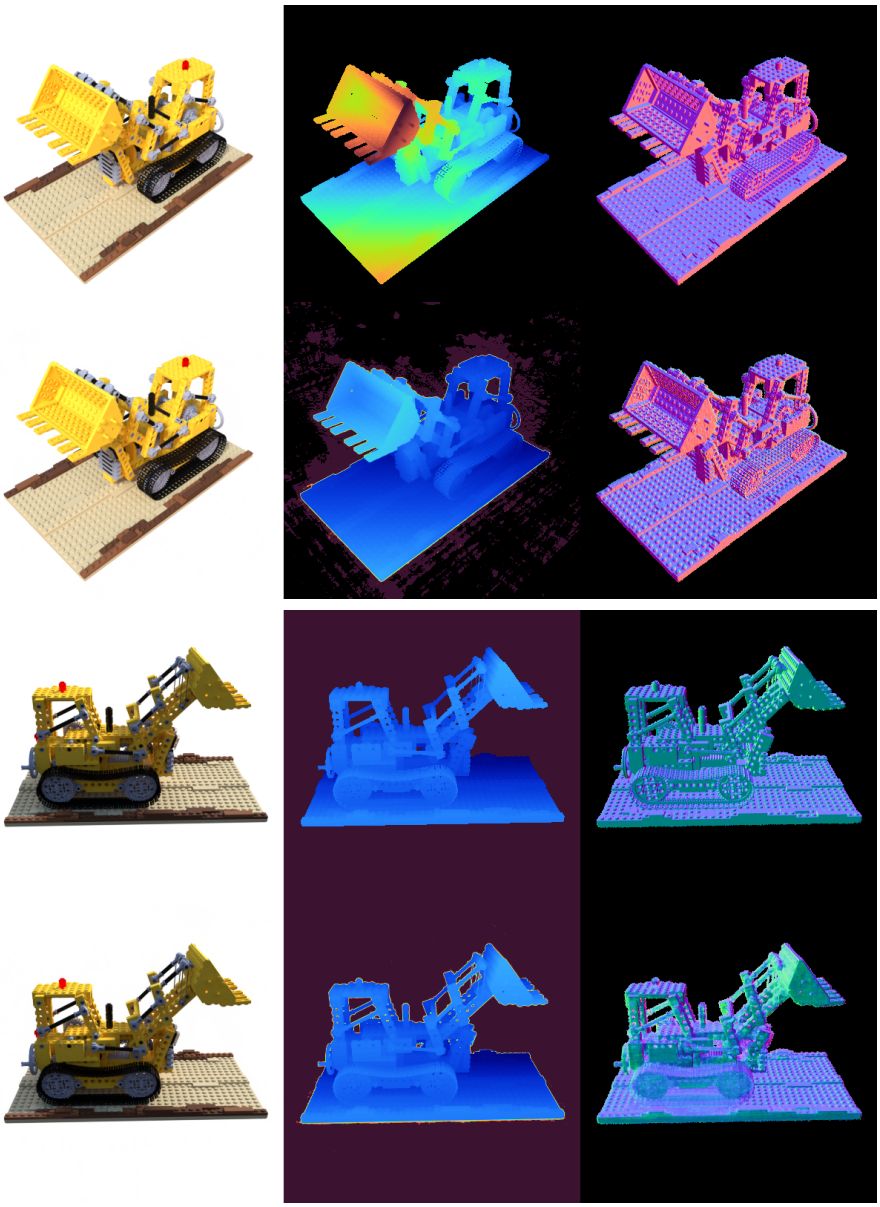

Figure C.4.: Training on 100 views for Lego Blender for RGB-D-N methods. Top: density normals (n_σ) directly supervised. Bottom: Density normals indirectly supervised via MLP (n_θ). From top to bottom: GT, Pred, GT, Pred.

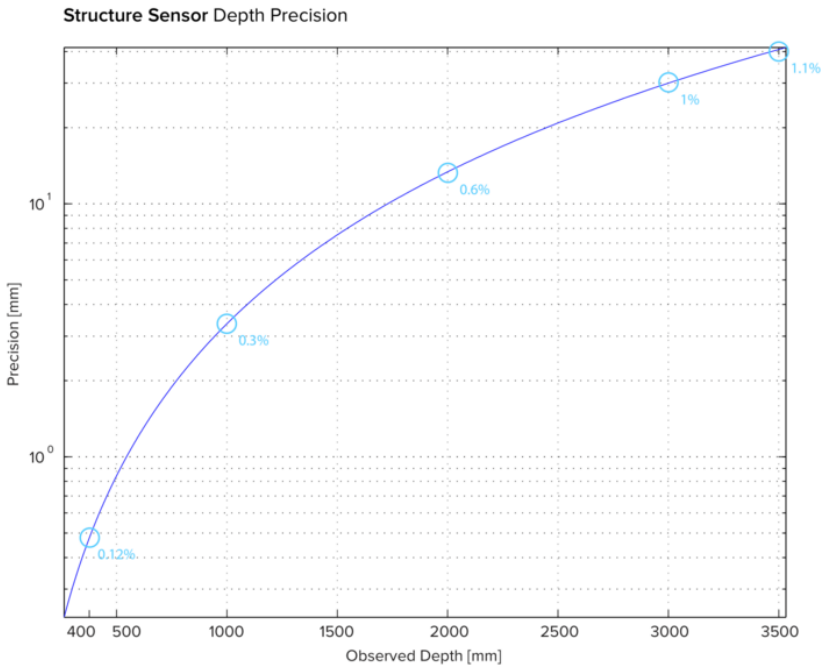

Figure C.5.: Sensor precision as a function of distance according to manufacturer specifications [MOc14]. This is the sensor used in the creation of the ScanNet dataset, error values should be interpreted as a lower bound on measurement error and don't consider distortion effects or user inexperience.

List of Figures

List of Tables

Bibliography

[Bar+21] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. "Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 5855–5864.

[Bar+22] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. "Mip-nerf 360: Unbounded anti-aliased neural radiance fields." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 5470–5479.

[Bra+18] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. *JAX: composable transformations of Python+NumPy programs*. Version 0.3.13. 2018.

[Cha+15] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. "Shapenet: An information-rich 3d model repository." In: *arXiv preprint arXiv:1512.03012* (2015).

[Che+21] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su. "Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 14124–14133.

[Che+22] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. "Tensorf: Tensorial radiance fields." In: *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXII*. Springer. 2022, pp. 333–350.

[CL96] B. Curless and M. Levoy. "A volumetric method for building complex models from range images." In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 1996, pp. 303–312.

[Dai+17] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. "Scannet: Richly-annotated 3d reconstructions of indoor scenes." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5828–5839.

[Den+22] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan. "Depth-supervised nerf: Fewer views and faster training for free." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 12882–12891.

[DLD12] A. Davis, M. Levoy, and F. Durand. "Unstructured light fields." In: *Computer Graphics Forum*. Vol. 31. 2pt1. Wiley Online Library. 2012, pp. 305–314.

[Eft+21] A. Eftekhar, A. Sax, J. Malik, and A. Zamir. "Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 10786–10796.

[Goo+20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. "Generative adversarial networks." In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[HSW89] K. Hornik, M. Stinchcombe, and H. White. "Multilayer feedforward networks are universal approximators." In: *Neural networks* 2.5 (1989), pp. 359–366.

[KB14] D. P. Kingma and J. Ba. "Adam: A method for stochastic optimization." In: *arXiv preprint arXiv:1412.6980* (2014).

[KW13] D. P. Kingma and M. Welling. "Auto-encoding variational bayes." In: *arXiv preprint arXiv:1312.6114* (2013).

[Lev90] M. Levoy. "Efficient Ray Tracing of Volume Data." In: *ACM Trans. Graph.* 9.3 (1990), pp. 245–261. ISSN: 0730-0301. DOI: 10.1145/78964.78965.

[LH96] M. Levoy and P. Hanrahan. "Light field rendering." In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 1996, pp. 31–42.

[Lin+21] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. "Barf: Bundle-adjusting neural radiance fields." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 5741–5751.

[Mar+21] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. "NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections." In: *CVPR*. 2021.

[Mat+00] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. "Image-based visual hulls." In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. 2000, pp. 369–374.

[Mes+19] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. "Occupancy networks: Learning 3d reconstruction in function space." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 4460–4470.

[Mil+21] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. "Nerf: Representing scenes as neural radiance fields for view synthesis." In: *Communications of the ACM* 65.1 (2021), pp. 99–106.

[MOc14] I. MOccipital. *structure sensor precision Published by Manufacturer*. Accessed: 2023-02-13. 2014. URL: `https://s3.amazonaws.com/io.structure.assets/structure_sensor_precision.pdf`.

[Mül+22] T. Müller, A. Evans, C. Schied, and A. Keller. "Instant neural graphics primitives with a multiresolution hash encoding." In: *ACM Transactions on Graphics (ToG)* 41.4 (2022), pp. 1–15.

[Nie+20] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger. "Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 3504–3515.

[Par+19] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. "Deepsdf: Learning continuous signed distance functions for shape representation." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 165–174.

[Pas+19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. "PyTorch: An Imperative Style, High-Performance Deep Learning Library." In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[Qi+17a] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. "Pointnet: Deep learning on point sets for 3d classification and segmentation." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[Qi+17b] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space." In: *Advances in neural information processing systems* 30 (2017).

[Rah+19] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. Hamprecht, Y. Bengio, and A. Courville. "On the spectral bias of neural networks." In: *International Conference on Machine Learning*. PMLR. 2019, pp. 5301–5310.

[Rem+22] K. Rematas, A. Liu, P. P. Srinivasan, J. T. Barron, A. Tagliasacchi, T. Funkhouser, and V. Ferrari. "Urban radiance fields." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 12932–12942.

[Roe+22] B. Roessle, J. T. Barron, B. Mildenhall, P. P. Srinivasan, and M. 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*. 2022, pp. 12892–12901.

[SF16] J. L. Schönberger and J.-M. Frahm. "Structure-from-Motion Revisited." In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[Sit+20] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein. "Implicit neural representations with periodic activation functions." In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 7462–7473.

[TM22] A. Tagliasacchi and B. Mildenhall. "Volume Rendering Digest (for NeRF)." In: *arXiv preprint arXiv:2209.02417* (2022).

[Ver+22] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan. "Ref-nerf: Structured view-dependent appearance for neural radiance fields." In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2022, pp. 5481–5490.

[Wei+21] Y. Wei, S. Liu, Y. Rao, W. Zhao, J. Lu, and J. Zhou. "Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 5610–5619.

[Yu+22] Z. Yu, S. Peng, M. Niemeyer, T. Sattler, and A. Geiger. "Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction." In: *arXiv preprint arXiv:2206.00665* (2022).