RBE/CS : 549 Project 2
Building built in minutes - Neural Radiance Fields (NeRF)

Divam Trivedi
MS in Robotics
Worcester Polytechnic Institute
Email: dtrivedi@wpi.edu

Shreya Boyane
MS in Artificial Intelligence
Worcester Polytechnic Institute
Email: ssboyane@wpi.edu

***Abstract*—Neural Radiance Fields (NeRF) have emerged as a transformative framework for novel view synthesis by representing scenes as continuous volumetric functions parameterized by deep neural networks. This report explores the methodology introduced in the seminal paper "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," focusing on its underlying principles, implementation details, and performance in synthesizing photorealistic images. We propose a neural network that accepts five-dimensional input—comprising three-dimensional spatial coordinates (x, y, z) and two angular parameters (θ, ϕ)—and produces both RGB radiance and volumetric density (σ) outputs, enabling effective continuous scene representation.**

I. INTRODUCTION

Neural Radiance Fields (NeRF) have emerged as a groundbreaking method for 3D scene representation and novel view synthesis. Proposed by Mildenhall et al. [1], NeRF utilizes a fully-connected neural network to map five-dimensional inputs—comprising spatial coordinates (x, y, z) and viewing angles (θ, ϕ)—to outputs of volumetric density (σ) and RGB color (r, g, b). A highly abstract version of the process is shown in fig. 1

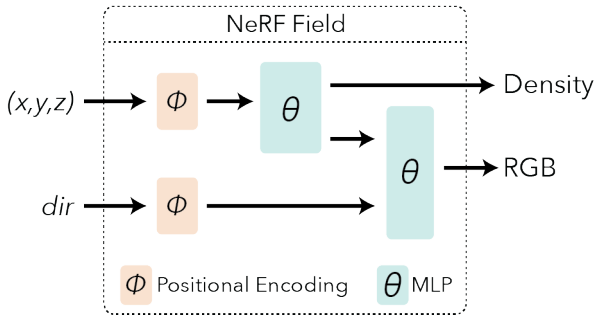

Fig. 1: NeRF Overview

NeRF synthesizes photorealistic images from arbitrary viewpoints by sampling points along camera rays and applying volume rendering techniques. This continuous scene representation enables advancements in virtual reality, augmented reality, and robotics by generating high-fidelity 3D reconstructions from sparse 2D images.

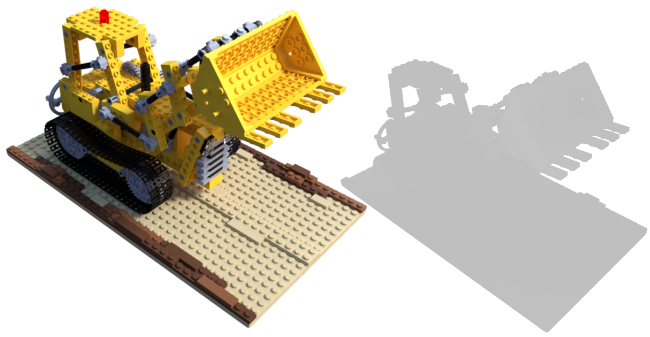

Fig. 2: RGB and Depth Image in Lego Dataset

NeRF's unified framework for modeling geometry and appearance makes it a pivotal innovation for immersive technologies, pushing the boundaries of scene rendering and visualization.

II. METHODOLOGY

In this section, we describe our overall approach for representing scenes as continuous neural radiance fields. Our methodology consists of three main components: (i) loading the datasets and associated camera parameters, (ii) defining the spatial bounds for the scene, and (iii) constructing the model architecture for mapping 5D input coordinates to volumetric density and view-dependent color.

A. Dataset

Firstly, we explore the dataset used, we use the standard NeRF synthetic dataset that consists of multiple static objects such as lego, ship, hotdog and many more. All the datasets consist of images for training, validation and testing along with their ground truth camera poses and camera extrinsics. Of all these, we take and validate our pipeline primarily on three datasets **lego**, **ship**, and **hotdog**. An example of the dataset is shown in fig. 2.

B. Setting Bounds for the Scene

Since our method represents scenes in continuous space, it is essential to define a bounded volume over which the network is queried. For synthetic scenes, we normalize the scene so that all content lies within a cube of side length 2 centered at the origin. This normalization can be expressed as

$$\mathbf{x}_{\text{norm}} = \frac{\mathbf{x} - \mathbf{x}_{\text{center}}}{s},$$

where $\mathbf{x}_{\text{center}}$ is the center of the scene and s is a scaling factor ensuring that the maximum extent of the scene is bounded.

For real scenes, where the depth may span a large range, we transform the depth values into *normalized device coordinates* (NDC). This transformation maps the ray origins to the near plane and converts perspective rays into parallel rays within the bounded interval $[-1, 1]$. In practice, we use disparity (inverse depth) to maintain a bounded representation:

$$d_{\text{NDC}} = \frac{1}{d} \quad \text{with} \quad d \in [d_{\min}, d_{\max}],$$

ensuring that all 3D coordinates fall within a finite range during optimization.

C. Model Architecture

The core of our method is a fully-connected neural network that represents the scene as a continuous function

$$F_\Theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma),$$

where $\mathbf{x} \in \mathbb{R}^3$ is a 3D point, $\mathbf{d} \in \mathbb{S}^2$ is a 2D viewing direction (encoded as a 3D unit vector), $\mathbf{c} \in \mathbb{R}^3$ is the emitted RGB color, and $\sigma \in \mathbb{R}^+$ represents the volume density. The architecture of the model (same for both fine and coarse model) is as shown in fig. 3.

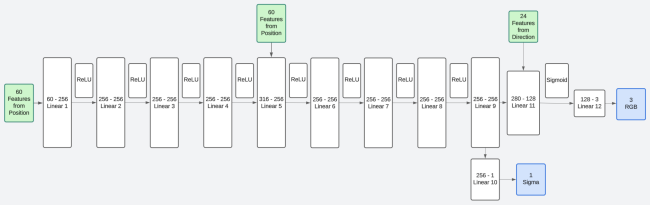

Fig. 3: Architecture of NeRF model

a) Positional Encoding.: Directly regressing from raw input coordinates often limits the network's ability to capture high-frequency details. Therefore, both the spatial coordinates and viewing directions are first passed through a positional encoding function $\gamma(\cdot)$ defined as:

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

which is applied independently to each component of $\mathbf{x}$ and $\mathbf{d}$. In our experiments, we set $L = 10$ for spatial coordinates and $L = 4$ for viewing directions.

b) Network Structure.: The network is divided into two stages:

- **Coarse Network:** The coarse network is an 8-layer multilayer perceptron (MLP) with 256 hidden units per layer. A skip connection is introduced at the 4th layer by concatenating the original encoded spatial coordinates with the intermediate feature representation. This network outputs:
  - σ (volume density), obtained via a linear layer followed by a $\mathrm{softplus}$ activation to enforce non-negativity.
  - A feature vector that is further processed to predict view-dependent color.
- **RGB Prediction:** The 256-dimensional feature vector is concatenated with the encoded viewing direction and passed through a two-layer MLP. The final output is an RGB color vector $\mathbf{c}$, with values bounded between 0 and 1 using a sigmoid activation.

c) Volume Rendering.: For a given camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ with near and far bounds t_n and t_f, the expected color is computed as

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

where the accumulated transmittance $T(t)$ is given by

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

In practice, this integral is approximated using stratified sampling along each ray, followed by quadrature to obtain the final rendered color.

d) Optimization and Scheduling.: The network parameters, including those of both the coarse and (optionally) fine networks, are optimized using the Adam optimizer. An exponential learning rate scheduler is employed to decay the learning rate over time:

$$\text{lr}_t = \text{lr}_0 \times \gamma^{\frac{1}{T}},$$

where T is the maximum number of iterations and γ is set such that the learning rate decays from the initial value to a fraction of it by the end of training.

This integrated approach—combining effective dataset management, appropriate scene bounds, and a robust network architecture with hierarchical volume sampling and adaptive optimization—enables our model to capture high-frequency details and synthesize photorealistic novel views.

III. Working of NeRF

Neural Radiance Fields (NeRF) reconstruct 3D scenes from 2D images with known camera positions and orientations. NeRF represents a scene as a continuous function that outputs color and volume density for each 3D point and 2D viewing direction. The process involves casting rays from the camera through each image pixel, tracing light's path from the scene to the sensor. Multiple 3D points are sampled along each

ray, and the NeRF model predicts color and density for each point. This is shown pictorially in fig. 4.

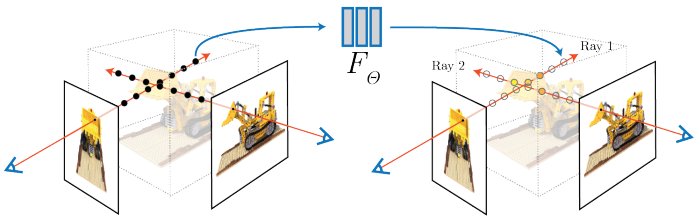

Fig. 4: Working principle of NeRF

Volume density indicates light absorption, with high density suggesting solid objects and low density indicating empty space. Color depends on position and viewing angle, capturing non-Lambertian effects like reflections and transparency. Combining predicted colors and densities for all points along a ray produces the final pixel color. This process is repeated for all pixels in all images.

NeRF's power lies in its differentiability, allowing it to learn the correct scene representation by comparing rendered images to input images and minimizing the difference through gradient descent. After training, NeRF can generate new views of the scene from unseen angles, producing photorealistic renderings with intricate details and realistic lighting.

IV. RESULTS

Over here we present different metrics of calibration and training of the model along with the vuews rendered from our pipeline.

A. Loss Curves

We first present the training loss curves for the pipeline, fig. 6 shows the loss curve for the coarse network whereas fig. 7 shows the loss curve for the fine network. In fig. 5 we can see the overall loss for the model.

Fig. 5: Loss curve for training

B. Novel Views

In this section, we present novel views from which the GIF was created. We present here, 3 datasets from the NeRF Synthetic datasets, These are for Lego, Ship and Hotdog as shown by fig. 8, fig. 9 and fig. 10

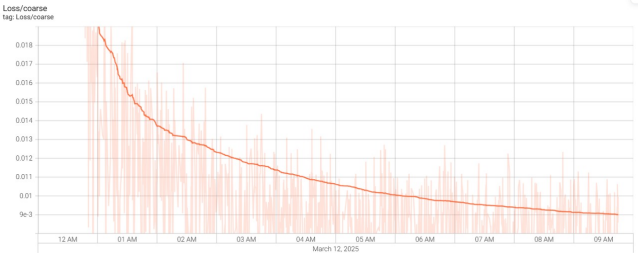

Fig. 6: Loss curve for Coarse model

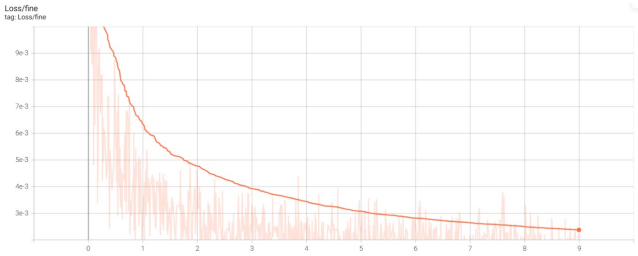

Fig. 7: Loss curve for Fine-tuning of model

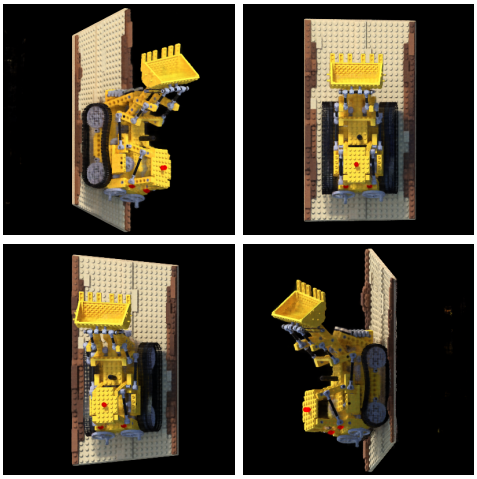

Fig. 8: Novel Views for Lego Dataset

C. Iteration Performance

Given here, is an overview of how iterations affect the performance of the NeRF model and how over iterations, the rendering gets clearer. These are shwon in fig. 11, fig. 12 and fig. 13. We can see in fig. 14 the effect of iterations on various datasets. The Htdog dataset due to its lower complexity was able to render good views in less than 250000 iterations. On the other hand, more complicated datasets such as Lego or Ship can even take upto 200000 iterations.

D. Positional Encoding

To analyze the impact of positional encoding (PE), we visually compare the reconstructed images generated by the model with and without PE against the ground truth. Fig. 15 illustrates the qualitative differences.

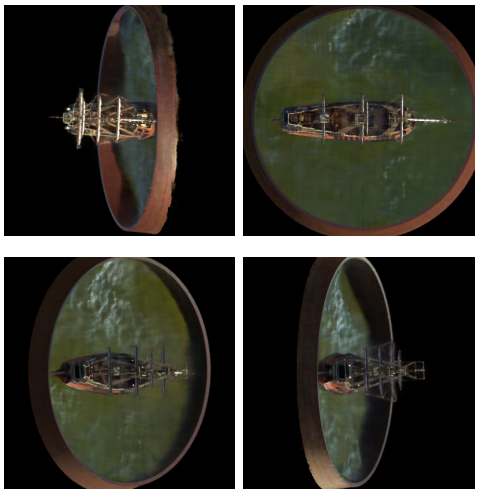

Fig. 9: Novel Views for Ship Dataset

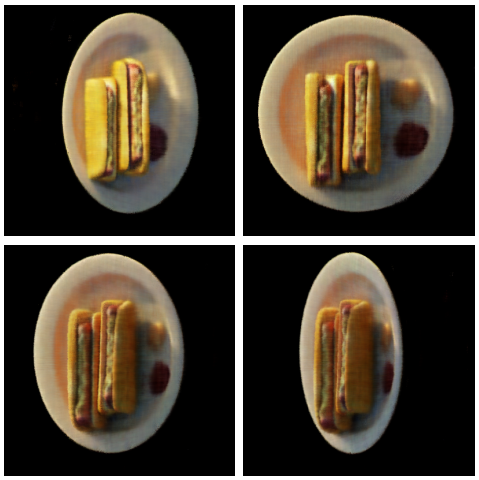

Fig. 10: Novel Views for Hotdog Dataset

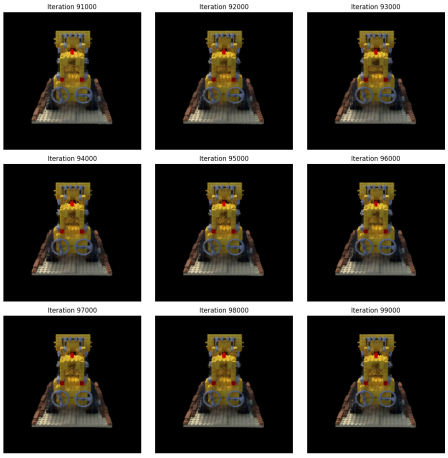

Fig. 11: Performance over iterations for Lego Dataset

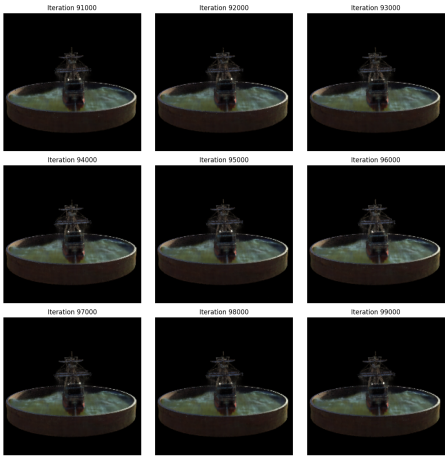

Fig. 12: Performance over iterations for Ship Dataset

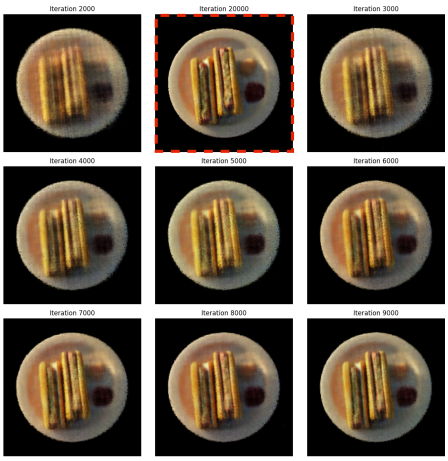

Fig. 13: Performance over iterations for Hotdog Dataset

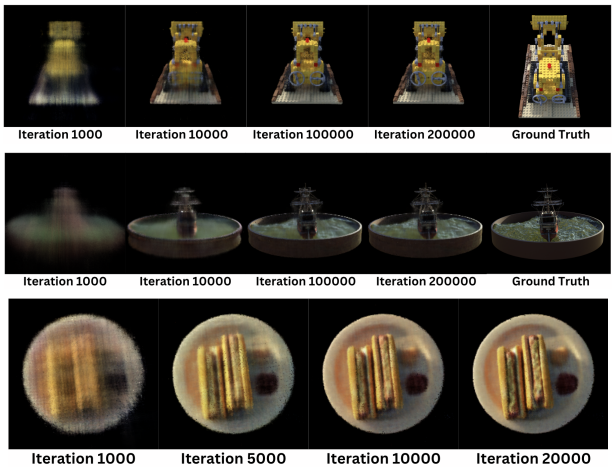

Fig. 14: Comparison of Rendered views over various iterations

Positional encoding plays a crucial role in capturing high-frequency details and improving spatial representation. Without positional encoding, the model struggles to learn fine-grained structures, resulting in blurry reconstructions with

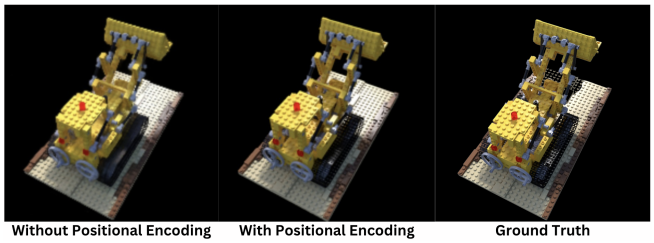

Fig. 15: Effect of Positional Encoding on Rendered Outputs

washed-out textures. This happens because standard neural networks tend to favor learning low-frequency patterns, making it difficult to represent complex scene details.

With positional encoding, the input coordinates are transformed into a higher-dimensional space with a series of sinusoidal functions. This allows the network to learn intricate spatial variations, leading to sharper edges, more defined object boundaries, and improved texture reproduction. As a result, the reconstructed images closely resemble the ground truth, maintaining structural integrity and preserving high-frequency components such as fine textures and small details.

E. PSNR/SSIM

To evaluate the quality of novel view synthesis, we use two standard image quality metrics: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM).

a) Peak Signal-to-Noise Ratio (PSNR): PSNR is a widely used metric for measuring image reconstruction quality. It is defined as:

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

where MAX is the maximum pixel intensity (typically 1 for normalized images), and MSE represents the mean squared error between the predicted and ground truth images. Higher PSNR values indicate better image quality, with values above 30 dB typically considered high quality.

b) Structural Similarity Index (SSIM): SSIM measures the perceptual similarity between two images by considering luminance, contrast, and structural information. It is computed as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{3}$$

where μ_x, μ_y are the mean intensities, σ_x^2, σ_y^2 are variances, and σ_{xy} is the covariance. SSIM values range from 0 to 1, with higher values indicating better structural similarity to the reference image.

TABLE I: Effect of Positional Encoding on PSNR and SSIM

Method	PSNR (dB)	SSIM
With Positional Encoding(Lego)	24.2953	0.9043
Without Positional Encoding(Lego)	3.6639	0.0306
With Positional Encoding(Ship)	27.1775	0.7812
Without Positional Encoding(Ship)	4.2831	0.0412

As seen in Table I, the inclusion of positional encoding significantly improves the model's performance, demonstrating its effectiveness in capturing high-frequency details and preserving structural similarity. These results highlight the crucial role of positional encoding in enabling the network to learn complex scene representations.

V. Conclusion

This report explored Neural Radiance Fields (NeRF) for novel view synthesis. We detailed a pipeline that includes dataset preparation, scene normalization, and a neural network mapping 5D inputs to volumetric density and view-dependent color, using volume rendering and hierarchical sampling to generate photorealistic images from sparse views.

Positional encoding enhances reconstruction by transforming coordinates into a higher-dimensional space, enabling the capture of high-frequency details and resulting in sharper, more accurate outputs, as supported by both visual comparisons and quantitative metrics.

Future work will focus on improving training efficiency and scalability for complex scenes. Overall, NeRF represents a significant advance in neural scene representation, with promising applications in virtual reality, augmented reality, and computer graphics.

References

[1] 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 *European Conference on Computer Vision (ECCV)*, 2020, pp. 405–421.

[2] J. T. Barron, B. Mildenhall, M. Tancik, P. P. Srinivasan, and R. Ramamoorthi, "Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields," in *International Conference on Learning Representations (ICLR)*, 2021.

[3] R. Martin-Brualla, E. M. Sitzmann, D. Wetzstein, and P. Mildenhall, "NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[4] A. Pumarola, T. Agustsson, A. Newell, and R. Theis, "D-NeRF: Neural Radiance Fields for Dynamic Scenes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.