Tuned-PA-NeRF: Exploring the Impact of Positional Encoding and MLP Architectures on Neural Radiance Fields (NeRF)

u7802794 Guangxin Mi, u7889456 Angxuan Li, u7439262 Zezhou Wang, u7439250 Xinni Song

1 Introduction

In the domain of 3D computer vision, Neural Radiance Fields (NeRF) [3] has shown its capability in complex 3D representation and view synthesis. Through optimizing the weights of the specially designed MLP based on positional encoding and differentiable volume rendering, NeRF learns a volumetric representation of the scene, allowing the model to synthesize highly realistic views of different camera angles. NeRF has many variations and advanced versions, which expand the application fields of the model: PixelNeRF [9] integrates Convolutional Neural Networks (CNN) and reconstructs from a single or small picture, NeRF in the wild [2] enhances the reconstruction in uncontrolled environments, NeuS [8] focuses on precise geometric surface reconstruction, while Instant-NGP [4] significantly accelerate training and inference process. These subsequent works further indicate the promising performance of NeRF.

Building upon this progress and insights, we aim to explore and evaluate the impact of different selections of modules or strategies on NeRF, including positional encoding strategies and MLP architectures.

2 Aims and Objectives

NeRF relies heavily on positional encoding to learn rendering by mapping 3D coordinates into high-dimensional with high-frequency feature space [3]. We will investigate the impact of using different positional encoding schemes, such as hash-based encoding [4] and Fourier Features [6], on the model's performance to capture fine details and generalize novel viewpoints.

The original NeRF architecture consists of a simple 8-layers MLP that processes 5D coordinates and produces density and color values. We aim to investigate and evaluate the model's performance with different MLP structural order, number of layers, etc.

3 Proposed Method Explanation

3.1 Position Encoding Methods

Traditional Position Encoding: Given a 3D coordinate $p = (x, y, z)$, Positional Encoding maps it into a higher-dimensional space using sine and cosine functions of different frequencies.

For example, the positional encoding $\gamma(p)$ of the input p can be given by:

$$\gamma(p) = \left[\sin(2^0\pi p), \cos(2^0\pi p), \sin(2^1\pi p), \cos(2^1\pi p), \ldots, \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p)\right]$$

- L is a hyper-parameter that controls the number of frequency bands used.
- The sine and cosine functions apply different frequency scales to the input, allowing the network to learn both fine details (high-frequency) and smoother structures (low-frequency).

Hash Encoding: As shown in Figure 1, Multi-Resolution hash encoding converts the positional information of (x, y, z) into density information in multiple separate hash tables at that location and outputs an L×F dimensional vector as input to the network, where L refers to number of different resolutions and F is the dimension of density feature encoding.

3.2 NeRF Structure

Traditional NeRF Structure: NeRF uses a multi-layer perception (MLP) network to model the scene. The MLP receives the position (x, y, z) and viewing direction (θ, ϕ) as input and outputs the volume density σ and RGB color (r, g, b).

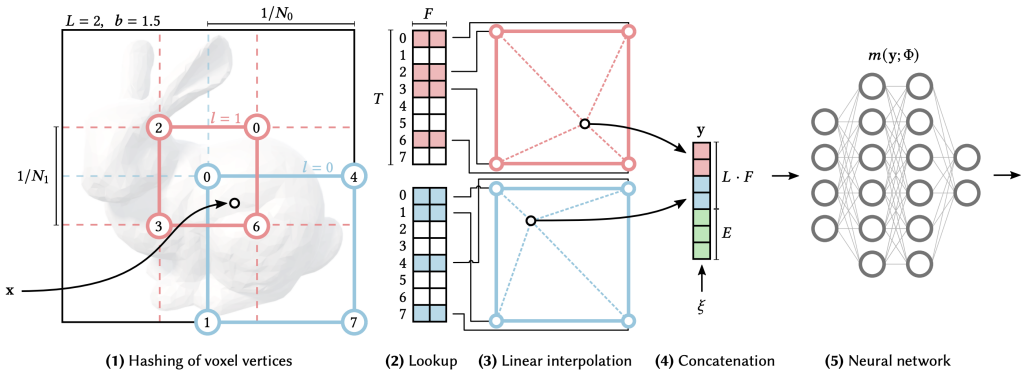

Figure 1: Multi-Resolution Hash Encoding

FastNerf Structure: FastNeRF [1] architecture splits the same task into two neural networks that are amenable to caching. The position-dependent network F_{pos} outputs a deep radiance map (u, v, w) consisting of D components, while the F_{dir} outputs the weights for those components $(\beta_1, \ldots, \beta_D)$ given a ray direction as input.

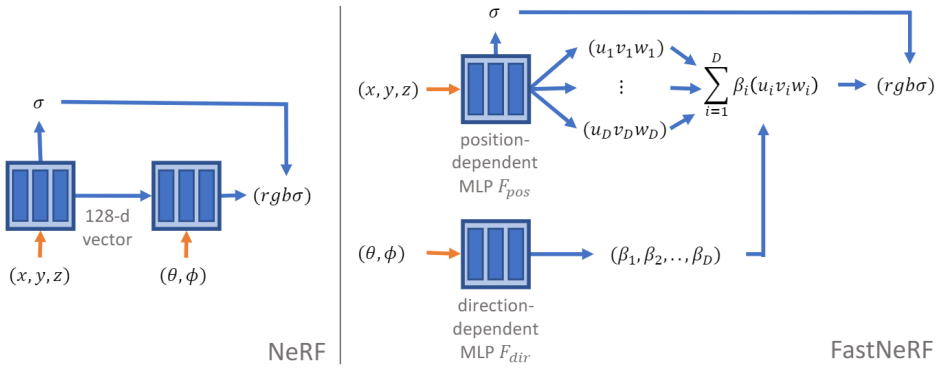

Figure 2: NeRF neural network structure(left) and FastNeRF [1] structure(right)

4 Experimental Setting, Software, and Dataset

Since the 3D computer vision will potentially consume extensive computational resources, we plan to use the Gadi platform provided by National Computing Infrastructure (NCI) to accelerate the training process with the help of powerful GPUs like A100. We will employ Nerfstudio's API [7] for training and fine-tuning. Nerfstudio provides an end-to-end API that simplifies the entire process of NeRF from creation, training to testing [7].

For the training, we plan to mainly utilize one primary dataset: the synthetic Lego dataset [5], employed in the work of NeRF [3]. The Lego dataset consists of multiple rendered images of a 3D Lego model captured from different viewpoints, which is an ideal dataset for the experiment.

5 Project Plan

The project will be divided into several phases, with clear timelines and milestones to track progress:

1. Literature Review, Dataset Preparation, and Environment Setting (Week 7: 17/09/2024 to 22/09/2024),
2. Model Design and Implementation (Week 8: 23/09/2024 to 29/09/2024),
3. Training and Testing (Week 9 - Week 10: 30/09/2024 to 13/10/2024),
4. Evaluation and Comparison (Week 11: 14/10/2024 to 20/10/2024),
5. Final Report and Presentation (Week 12: 21/10/2024 to 27/10/2024).

6 References

[1] Stephan J Garbin et al. "Fastnerf: High-fidelity neural rendering at 200fps". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 14346–14355.

[2] Ricardo Martin-Brualla et al. "Nerf in the wild: Neural radiance fields for unconstrained photo collections". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 7210–7219.

[3] Ben Mildenhall et al. "Nerf: Representing scenes as neural radiance fields for view synthesis". In: *Communications of the ACM* 65.1 (2021), pp. 99–106.

[4] Thomas Müller et al. "Instant neural graphics primitives with a multiresolution hash encoding". In: *ACM transactions on graphics (TOG)* 41.4 (2022), pp. 1–15.

[5] *Synthetic Lego Dataset*. `https://drive.google.com/drive/folders/128yBriW1IG_3NJ5Rp7APSTZsJqdJdfc1`. Accessed: [2024.09.19].

[6] Matthew Tancik et al. "Fourier features let networks learn high frequency functions in low dimensional domains". In: *Advances in neural information processing systems* 33 (2020), pp. 7537–7547.

[7] Matthew Tancik et al. "Nerfstudio: A Modular Framework for Neural Radiance Field Development". In: *ACM SIGGRAPH 2023 Conference Proceedings*. SIGGRAPH '23. 2023.

[8] Peng Wang et al. "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction". In: *arXiv preprint arXiv:2106.10689* (2021).

[9] Alex Yu et al. "pixelnerf: Neural radiance fields from one or few images". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 4578–4587.