CS231A Project Midterm Milestone
Explorations and Extensions of NeRF

Colin Zheng
Department of Mechanical Engineering
Stanford University
colinz07@stanford.edu

Yiwen Jiang
Department of Electrical Engineering
Stanford University
yjiang98@stanford.edu

Tianheng Shi
Department of Mechanical Engineering
Stanford University
ts5@stanford.edu

Abstract

This paper is the project report for CS231a at Stanford University for student group of Colin Zheng, Yiwen Jiang, and Tianheng Shi.

This project focuses on NeRF: Neural Radiance Fields. The team first trains trained a model with provided dataset obtained from a Nerf open sources directory. With this trained model, a rendered video of a lego-truck, which is also a provided example, is used to test the model.

After replicating the results with provided dataset, the model was retrained with our own datasets. Different dataset under variant conditions were collected. NeRF's performance on those datasets was evaluated and compared to examine the applications for NeRF. Fine tuning was done for possible performance enhancement.

1. Introduction

Neural Radiance Fields, also known as NeRF, has been one of the most talked about topics in the field of computer vision in recent years. This rendering method was first proposed in a paper by Mildenhall et al. [4] The paper explored a way to synthesize a view of complex scenes. With given input images with known poses, NeRF can generate outputs as volume density and view-dependent emitted radiance. Then, with volume rendering techniques, the outputs can be synthesized into the image.

This topic is novel due to its unprecedented way to process volumetric rendering. It uses a similar structure as DeepSDF[5], but with density and color instead of a distance function. The synthesized views demonstrated impressive quality and detail.

In order to obtain camera pose estimates for the collected data, COLMAP is used. Using Structure-from-Motion algorithm as a foundation, COLMAP improves on the current method through methods such as augmenting scene graph and including filtering layer to provide a robust and accurate method to obtain camera poses from collected images.

1.1. NeRF: Neural Radiance Fields

Neural Radiance Field[4] is represented as a fully-connected neural network that maps 5D coordinates, 3 for positions and 2 for direction of scene, to the RGB value and single volume density. With the output color value and volume density, classical volume rendering techniques are used to reconstruct the 2D image that corresponds to the input 5D coordinates. The whole process of using 5D coordinate to render the 2D image is differentiable. Therefore, the whole model is optimized using gradient descent to minimize the loss between the reconstructed scene and the ground truth.

For the volume rendering techniques, the paper by Mildenhall et al. [4] used an approach based on the classical techniques by Kajiya, J.T.and Herzen, B.P.V. [1]. Instead of using continuous integral along the camera ray to calculate the expected color of the color ray, deterministic quadrature is used to discretize the integral. The resultant equation[4] for the color is:

$$\hat{\mathcal{C}}(\mathbf{r}) = \sum_{i=1}^{N} T_i(1 - \exp(-\theta_i\delta_i))\mathbf{c}_i, T_i = exp(-\sum_{j=1}^{i-1}\theta_j\delta_j) \tag{1}$$

Here, $\mathbf{c}$ and θ are the model output RGB color and density, with δ as the distance between adjacent samples along the camera ray.

The issue raises after using the neural radiance field representation and the volume rendering techniques: the

quality is not satisfactory for many high-resolution complex scenes. Two novel optimization methods are being applied[4]: positional encoding and hierarchical volume sampling.

Positional encoding is applied to solve the issue that the previously mentioned 5D input coordinates will only generate poor result in a scenario with high frequency variation in color and geometry. The idea is to map the 5D coordinates into a high-dimensional space with the following encoding function [4]:

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

Such an encoding function can project from $\mathbb{R}$ to $\mathbb{R}^{2L}$. Such function is applied to the previous input so that the neural network can be trained with a higher dimensional input and eventually yield solid result even under high-frequency variation in color and geometry cases.

In addition to the optimization with positional encoding, hierarchical volume sampling is also being used by Mildenhall et al.[4]. This optimization is necessary because the discretization approximation along the camera ray is often inefficient, especially given the cases with occlusion. The hierarchical volume sampling is based on the paper by Levoy, M.[2]. The idea is to sample along the ray proportionally with respect to the effect of sample towards the final rendering result. The method is to first train a "coarse" network using the original sampling method. With the trained "coarse" network, decompose the result with the following equation [4]:

$$\hat{\mathcal{C}}_c(\mathbf{r}) = \sum_{i=1}^{N_c} \omega_i c_i, \omega_i = T_i(1 - \exp(-\theta_i\delta_i)) \tag{3}$$

Then, normalize the w_i to generate a piecewise-constant PDF along the camera ray and resample based on inverse transform sampling. With the new samples N_f and the "coarse" samples N_c, a "fine" network is trained following eqn (1).

The results generated by NeRF is compared with Neural Volumes[3] and Scene Representation Networks[8]. The result is promising after computing the metrics of PSNR, SSIM, and LPIPS using multiple datasets. PSNR and SSIM will be explained in detail in technical approach section. In addition, qualitatively comparison is also made to show the outstanding performance of NeRF on different scenarios.

This unprecedented method successfully represents the complex continuous scenes of complex shape into a compact 5D neural radiance field, which is trained through a simple network without CNN getting evolved. Besides, the optimizations that the paper by Mildenhall et al. [4] proposed, hierarchical sampling and positional encoding, helped to generate high-quality results under circumstances that were processed poorly by other earlier works. All of these advantages made the NeRF the state-of-art view synthesis method.

The code we refered to about NeRF is a pytorch version, which can be found at the following link: https://github.com/yenchenlin/nerf-pytorch

1.2. COLMAP

COLMAP is a pipeline with Structure-from-Motion and Multi-View Stereo functions. For the purpose of this paper, only the structure from motion portion will be used. COLMAP implements a new Structure-from-Motion algorithm proposed by Schönberger and Frahm in 2016 [6].

This new Structure-from-Motion algorithm is based on the traditional Structure-from-Motion pipeline, which is consisted of two parts, correspondence search and incremental reconstruction.

The correspondence search section focuses on locating the overlap in input images and identifying correspondence points. This is achieved by first passing input image through feature extraction where local features that matches a certain appearance descriptor are being detected. Next, with the features a matching procedure is performed to locate overlapping scenes in the images. Finally, the potentially overlapping image pairs are being verified through projective geometry where a transformation estimation is generated and only be considered valid if it maps a sufficient number of features between the images.

With the verified correspondence pairs between images generated from the first part, the second part, incremental reconstruction, focuses on estimating pose as well as reconstructing scene structure. This portion of the algorithm starts with initializing a two-view reconstruction, and the later processes are based off of this initial reconstruction. Solving the Perspective-n-Point problem allows new images to be registered to the model, during the process the pose, P_c, of the newly registered image is being added to the overall set $\mathcal{P}$. New scene point X_k are being added using triangulation on the newly registered image. Bundle Adjustment is then performed to minimize the reprojection error and thus refining the parameters.

This new Structure from Motion algorithm aims in improving the algorithm's ability to handle diverse data collection and in its completeness and robustness. This new algorithm tackles existing challenges through the following adjustment on the current approach. In the geometric verification step, a multi-model strategy is used to augment the scene graph. A Next Best View Selection is being included such that images with more visible points and more uniform distribution are given higher score and in result registered first. Next, RANSAC is being included in the triangulation step to allow the algorithm to be robust even with contaminated data. To enhance the performance, a filtering

step is included to filter out observation with large errors, re-triangulation is performed when drift effect is detected before the bundle adjustment step. Lastly, redundant cameras are being clustered into groups of high scene overlap, this allows or higher computation efficiency.
https://colmap.github.io/.

2. Problem Statement

For this project, we will be focusing on taking a deeper look into different applications of NeRF: Neural Radiance Fields For View Synthesis and examine its performance under varying conditions: varying light conditions, background conditions, and image depth variations.

Given that the NeRF is not trained to be generalized to all inputs, a distinct dataset will be collected and processed for each condition. The results will be evaluated with metrics and compared with results from [4], as well as other conditions' results.

Detailed method for data collection and processing will be elaborated in the following section.

3. Technical Approach

The NERF architecture is referenced from https://github.com/yenchenlin/nerf-pytorch, and the SSIM calculations are referenced from https://github.com/Po-Hsun-Su/pytorch-ssim. To ensure the functionality of the code and model provided by the open source library, the team first replicated the lego results listed in [4]. To achieve that, the team set up a virtual machine and used the provided synthetic 360 dataset that represents the images and camera matrices as the input to the model. This serves the purpose of learning the model and the output format, and to perform qualitative and quantitative analysis on the model. The qualitative analysis is done by visually inspecting the mp4 files from the output of the model, and the quantitative analysis includes calculating the Peak Signal-to-Noise Ratio (PSNR) of the output. PSNR is calculated using the MSE of the ground truth image I and generated image K. Given an image size of $m \times n$, the MSE is calculated by:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \tag{4}$$

Then, the PSNR(dB) can be calculated by

$$PSNR = 10log_{10}(\frac{MAX_I^2}{MSE})^2 \tag{5}$$

where MAX_I^2 is the maximum pixel value of the reference image. In the colored image case which has three channels representing RGB, the pixel value is the mean of the RGB values of each pixel. Another image quality metric we plan to use is Structural Similarity (SSIM), which is also a widely used image quality metric. SSIM takes a sliding window of size $N \times N$ from images x and y, and calculates the mean μ_x μ_y, variance σ_x σ_y, and covariance σ_{xy} for each window. Then, the luminance, contract and structure values could be calculated by:

$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \tag{6}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \tag{7}$$

$$s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3} \tag{8}$$

where $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$ are constants to prevent zero-division. L is the range of pixel values. Normally, $c_3 = c_2/2$. Then, SSIM is calculated by

$$SSIM(x,y) = [l(x,y)^\alpha \times c(x,y)^\beta \times s(x,y)^\gamma] \tag{9}$$

By setting α, β and γ to 1, the above equation becomes

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{10}$$

Finally, the mean of SSIM calculated for all $N \times N$ windows is taken for the SSIM for whole images.

For data collection process, a cellphone iPhone 13 pro max was used. For each condition, a distinct dataset was collected. 56 pictures was taken for each scene and 1/8 of them was used as test sets. Each image's resolution is determined to be 378 x 504, which is 1/8 of the 3024 x 4032 of the iPhone camera resolution.

The collected data was passed through COLMAP to obtain estimated camera poses for each set of images. It should be noted that COLMAP have certain requirements including having good texture and similar illumination for the input images to ensure the accuracy of the result. The function outputs a 17x1 vector for each camera, this vector includes a 15x1 camera pose value, close and inf depth. For n input images, the output of the COLMAP function will be a nx17 matrix. This matrix was passed directly into the nerf function.

We focused on analyzing Nerf's performance under the following three conditions: one with similar object and background colors, and one with high-contrast object and background colors; one with strong direct light condition, one with normal sunlight, and one with dim lighting condition; one with relative close scene depth, and one with relative large scene depth.

To determine whether the background colors affect the rendering results, a fake hand was used as the object in the scene. Two datasets were taken with similar and different background color. A screenshot of the dataset is shown in Fig. 1.

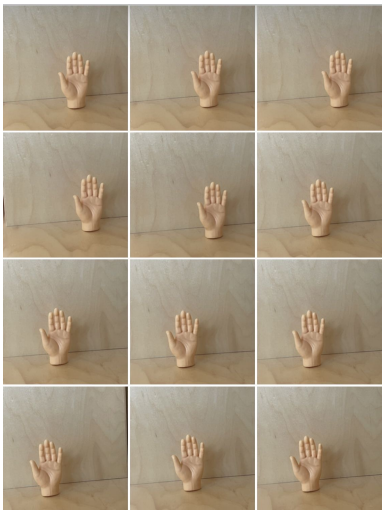

Figure 1. Fake Hand Scene Dataset

For the variation in light condition dataset, the team used a model of a car as the scene. Given the reflective surface of the car, the team thought this object could be sensitive to light. A screenshot of the dataset is shown as in Fig. 2.

Figure 2. Car Scene Dataset

For the exploration on the effect of scene depth, the team chose a scene of flower that has complex geometry and variation in color. The dataset is shown as in Fig. 3.

After exploring on the previously mentioned variation on dataset conditions, fine tuning was performed on the flower scene, which has the most noise-free and reasonable results, to explore possible performance improvements. In particular, the close scene depth flower dataset rendering

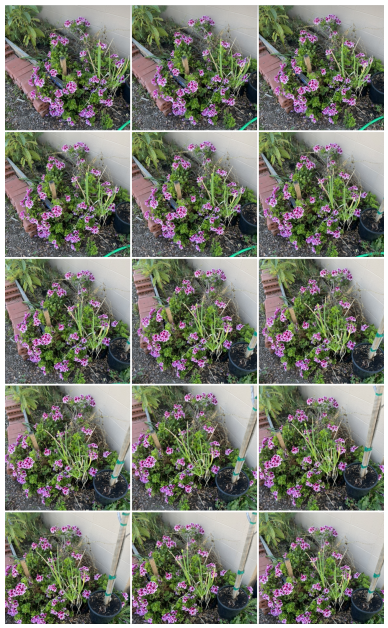

Figure 3. Flower Scene Dataset

results served as the baseline. The detailed results for different scenes are presented in the next sections. For the fine tuning, additional fine samples added, batch size, and network layer numbers are being altered. One parameter was changed at a time.

4. Results

The team spent a long time setting up the environment and virtual machine for high efficiency during training process. A GPU Tesla V-100 is being used. The training time for the dataset of Lego from Diffuse Synthetic 360◦[7] is around 6 hours, which is significantly faster comparing the 30 hours using a CPU. Firstly, the team successful replicated the results for Lego as mentioned before. A screenshot of the rendered .mp4 file is shown in Fig. 4. The metric PSNR is used as evaluation. The result after training is 32.50 which matches the result from [4]. The result mp4 file can be provided upon request by teaching staffs.

In addition, the team obtained the 6 DoF image poses and near/far depth bounds for a separate dataset of ferns that is provided by[4], which has unknown image poses. The team followed the procedure mentioned in the technical approach section. Similar process was done towards our own datasets for training and evaluation through NeRF. Screenshots of synthesized videos are provided in the remaining part of the result section. Videos can be found via the github link https://github.com/ColinZ07/nerf-pytorch.

For the fake hand scene, the result from the dataset which has similar background color is presented in Fig. 5. The result from the dataset which has different background color

Figure 4. Screenshot from Rendered Lego Scene

is presented in Fig. 6. Table 1. provides the PSNR and SSIM metric evaluations on the two results.

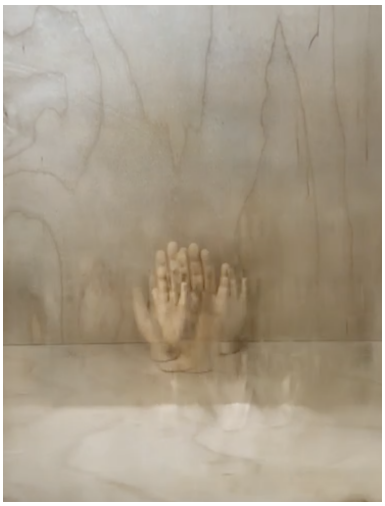

Figure 5. Screenshot from Rendered Fake Hand Scene with Similar Background Color

Figure 6. Screenshot from Rendered Fake Hand Scene with Different Background Color

For the flower scene, the result from the dataset which has relatively close scene depth is presented in Fig. 7. The result from the dataset which has relatively far scene depth is presented in Fig. 8. Table 2. provides the PSNR and SSIM metric evaluations on the two results.

Condition	PSNR	SSIM
Similar Background Color	39.1506	0.9995
Different Background Color	35.4633	0.9992

Table 1. Final Results Evaluation for Fake Hand with Similar and Different Background Color.

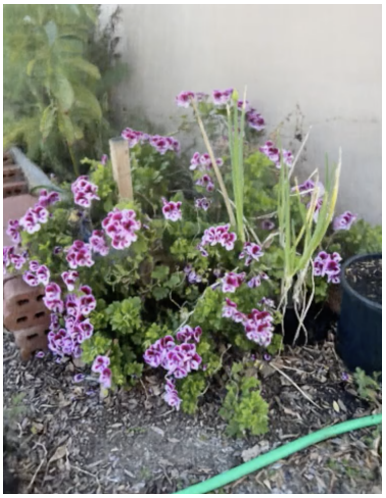

Figure 7. Screenshot from Rendered Flower Scene with Close Scene Depth

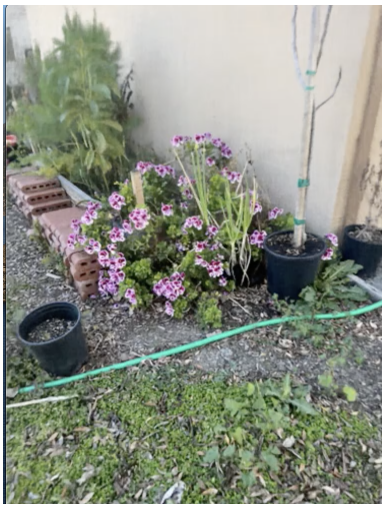

Figure 8. Screenshot from Rendered Flower Scene with Far Scene Depth

Condition	PSNR	SSIM
Close Scene Depth	22.6020	0.9765
Far Scene Depth	23.1733	0.9807

Table 2. Final Results Evaluation for Flower Scene with Close and Far Depth.

For the car scene, the screenshots for each light conditions are presented in Fig. 9, Fig. 10, and Fig. 11. Table 3.

Condition	PSNR	SSIM
Dim Light Condition	30.3618	0.9793
Natural Light Condition	21.3376	0.9717
Strong Light Condition	22.0771	0.9737

Table 3. Final Results Evaluation for Car with Dim/Natural/Strong Light Condition.

contains the PSNR and SSIM evaluation for each case.

Figure 9. Screenshot from Rendered Car Scene with Dim Light Condition

Figure 10. Screenshot from Rendered Car Scene with Natural Light Condition

Next, as mentioned, the close scene depth dataset of the flower scene is used for parameter tuning. For the result in Fig. 7 and Table 1., the parameters are: 64 additional fines samples, 1024 batch size, and 8 layers in the networks. Changing 8 layers to 12 layers led to the rendered result as

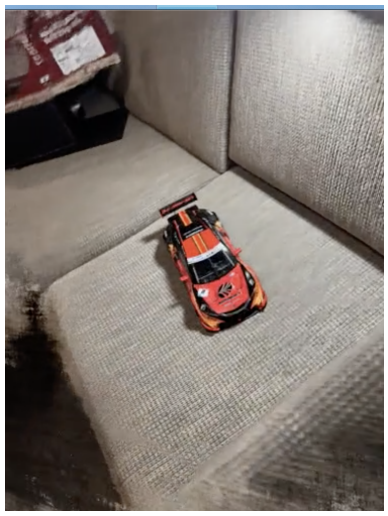

Figure 11. Screenshot from Rendered Car Scene with Strong Light Condition

Condition	PSNR	SSIM
8 Layers	22.6020	0.9765
12 Layers	22.2820	0.9738

Table 4. Final Results Evaluation for Close Flower Scene with Different Model Layers.

in Fig 12. The metric results are shown in Table 4. Using 128 additional fines samples instead of 64 yielded the rendered result as in Fig 13. Corresponding metric results are shown in Table 5. Finally, after setting the batch size to 2048, results shown in Fig. 14. were obtained, with Table 6. as metric evaluation.

Figure 12. Screenshot from Rendered Flower Scene with 12 Network Layers

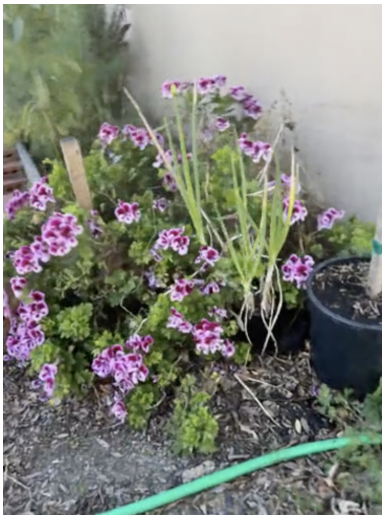

Figure 13. Screenshot from Rendered Flower Scene with 128 Additional Fine Samples

Condition	PSNR	SSIM
64 Additional Fine Samples Per Ray	22.6020	0.9765
128 Additional Fine Samples Per Ray	22.7472	0.9766

Table 5. Final Results Evaluation for Close Flower Scene with Different Additional Fine Samples Per Ray.

Figure 14. Screenshot from Rendered Flower Scene with 2048 Batch Size

Condition	PSNR	SSIM
1024 Batch Size	22.6020	0.9765
2048 Batch Size	22.8715	0.9774

Table 6. Final Results Evaluation for Close Flower Scene with Different Batch Size

5. Discussion

The quantitative and qualitative rendering results generated with the three sets of data collected under various conditions are being examined.

To start, results from the fake hand set where the object was placed in front of two backgrounds, one with contrasting color and one with similar color as the object are being evaluated. While Nerf was able to produce a nice rendering of the fake hand object placed in front the white background, one thing that was noticeable in the video was the inconsistency in light shadow. In contrast, for the set where the background had similar color as the object, it was evident that the rendering output was erroneous and presented a significant deviation from the expected result. This showed that when the object and background are alike, the Nerf algorithm has difficulty distinguishing the two. In addition, the qualitative results demonstrate that Nerf is also very sensitive to light shadows that are present in the input data. Specifically, the existence of inconsistent light shadow in the input data would adversely impact the quality of the results.

Another thing worth noting from the results of this data set is the mismatch between the qualitative and quantitative results. As discussed above, the quantitative result for object placed in front of a similar background is significantly worse than the one placed in front of contrasting background. However, the qualitative result indicated the opposite. Referring to Table 1, the similar background rendering result had higher values in both of the metrics, namely PSNR and SSIM. The low resolution of input data used can partially account for this disparity. The reduction of our original sized input data by a factor of 8 blurred the boundary between background and the object. Moreover, the low color gradient nature of our object and its similar colored background further obscured the edges. Both of these contributed to the increase in difficulty in the process of discerning pixels of data with very similar color. However, this still brings the question in whether or not PSNR and SSIM would serve as fair and ideal evaluation metrics for the purpose of evaluating Nerf results.

Looking into the effect of light conditions of dataset on results, light intensity had no significant impact on the object, which is the bright-colored car shown in the screenshots. However, under dim light conditions, the details of the sofa was much less compared to natural and strong light conditions. This can be seen in both qualitative and quantitative results: the sofa had much less textural details in the dim light setting that also led to higher PSNR and SSIM scores. This is consistent with the mismatch between the metric scores and qualitative results discussed above. The dim light condition data contains large areas with low color gradient resulted from less details, the MSE that was used to calculate both PSNR and SSIM is small, so that the metric

scores are higher but less representative of the actual results compared to other lighting conditions. Hence, it can be concluded that NERF is not suited for dim lighting conditions because of the loss of details.

For the flower scene with different scene depths, the synthesized videos of the two datasets have no visible change of quality. Also, from both the PSNR and SSIM comparisons, only negligible changes in numbers can be observed. It can be concluded that the scene depth changes have little impact on the result quality qualitatively and quantitatively.

Synthesized videos across different scenes are being compared. It can be observed clearly that the flower scene yielded videos with best quality. Potential reasons include, firstly, the dataset was captured in an outdoor environment where the light source is not limited to certain incoming angles. Therefore the dataset itself is not compromised by inconsistent shadows. Secondly, the flower scene has the most complex features and textures, which makes it much easier to be tracked.

Using the result from flower scene with close depth as the baseline, the fine tuning didn't lead to significant performance enhancement. The resultant videos can hardly be distinguished by quality. For the metric evaluations, it can be seen that changing 8 layers to 12 led to minor decrease on metrics. For the other two parameters, tuning led to trivial increases in metrics. This result is accepted and expected since NeRF most likely has already been tested with a wide range of parameter tuning, and the current setting of parameters has to be the optimal.

6. Conclusion and Future Work

With the goal of exploring potential application and the existing limitation of Nerf, we collected three sets of data that aim to investigate Nerf's performance under different conditions. The rendering and fine-tuning results of these data are being evaluated and analyzed. These results indicated that the depth of scene has no influence on the results. On the other hand, the closeness in color of an object to its background can adversely impact the quality of output. As for lighting conditions, NERF is not suited for dim-light applications as it would lead to unrepresentative PSNR and SSIM scores resulting from loss of image details. Moreover, the change in parameters also resulted in insignificant improvement in the final output, indicating the optimality of the current model. Limitations of the algorithm including its sensitivity to shadows and movement in the data acquisition process also became evident through examining the rendering results. Lastly, it can be seen that the results presented relatively low resolution which contributed to increasing mismatch between quantitative and qualitative results, as MSE calculations would be less representative because of the loss of details. As the input images now had been downsampled to $\frac{1}{8}$ for faster training, possible future work would be to evaluate the conditions evaluated in this project with higher image resolution to compensate for the negative effect of downsampling.

References

[1] B. Kajiya, J. T.and Herzen. Ray tracing volume densities. computer graphics. 1984. 1

[2] M. Levoy. Efficient ray tracing of volume data. acm transactions on graphics. 1990. 2

[3] T. S. J. S. G. L. A. S. Y. Lombardi, S.and Simon. Neural volumes: Learning dynamic renderable volumes from images. acm trans- actions on graphics (siggraph). 2019. 2

[4] B. Mildenhall, P. P. Srinivasan, M. Tancik, R. R. Barron, J. T., and R. Ng. Nerf. *Communications of the ACM*, 65(1):99–106, 2022. 1, 2, 3, 4

[5] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[6] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[7] J. H. F. N. M. W. G. Z. M. Sitzmann, V.and Thies. Deepvoxels: Learning persistent 3d feature embeddings. in: Cvpr. 2019. 4

[8] M. W. G. Sitzmann, V.and Zollhoefer. Scene representation networks: Con- tinuous 3d-structure-aware neural scene representations. in: Neurips. 2019. 2