Depth Light Field Training (DeLFT)

NeRF as a rendering primitive

Mihnea Toader[1]

Supervisor(s): Elmar Eisemann[1]**, Petr Kellnhofer**[1]**, Michael Weinmann**[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Mihnea Toader
Final project course: CSE3000 Research Project
Thesis committee: Elmar Eisemann, Petr Kellnhofer, Michael Weinmann, Jan van Gemert

Project page: https://mihneatoader.github.io/Depth-Light-Field-Training

Abstract

Neural radiance fields (NeRF) based solutions for novel view synthesis can achieve state of the art results. Recent work proposes models that take less time to render, need less training data or take up less space. However, few papers explore the use of NeRFs in classic rendering scenarios such as rasterization, which could contribute to wider adoption. Our paper tackles the issue of shadow generation and proposes a deep residual MLP network with fast evaluation times, that generates view-dependent shadow maps. The network distills the knowledge of an existing NeRF model and achieves the speedup through the use of neural light fields, by only doing one network forward per ray.

1 Introduction

Many professional industries have used non-contact passive 3D scanning and photogrammetry technologies extensively. From designing medical prosthetics [5] and modelling existing buildings in architecture [9] to VFX in the movie industry [14] and cultural heritage preservation [17], it highlights the usefulness of creating 3D models from real world objects. However, the principal methods that are used struggle with issues such as: insufficient information leads to deformed meshes, sensor error can drastically alter the output and, most notably, they deal poorly with view-dependent effects, such as reflections, refractions and occlusions [18].

NeRF models [11] take a drastically different approach to this by using neural networks to estimate volumetric scene functions. They can generate photo-realistic results that represent the scene with a high degree of accuracy even from novel poses. Additionally, view-dependent effects are reproduced at state-of-the-art quality [11]. However, because of their novel architecture, integration into classic rendering techniques has been slow. Techniques such as inverse rendering or scene relighting have given developers some freedom, but not nearly enough for wide adoption [25]. By fully coupling NeRFs and rasterization, developers would be able to achieve advanced photorealism without the use of more computationally expensive rendering methods, such as ray tracing.

This paper offers a possible solution to one of the problems proposed by this transition: lighting scenes with NeRFs, more specifically shadows. The question we aim to answer is "*Given a NeRF, can we learn a more compact neural representation that can directly produce the depth map for any desired view angle without the need for expensive 3D integration of the classical volumetric NeRF? Can we use it to render shadows for a NeRF object in a simple CG scenario?*"

We propose[1] a deep neural light field network that learns the depth map of the object using an existing NeRF to generate training data. This approach proves to be significantly faster than rendering the NeRF (∼100x speedup) and quality evaluation shows that the results are comparable to both ground truth (∼28 PSNR) and the base model (∼32 PSNR).

2 Related work

Neural methods for Novel View Synthesis The field of Novel View Synthesis has enjoyed much recent development. This can be largely accredited to the publication of *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis* by Mildenhall et. al [11], which used a simple neural network and radiance fields as the scene representation function. Subsequent works tackled the issue of rendering time by using concepts such as efficient ray sampling [13; 10; 8], scene division [16; 24] or auxiliary meshes or voxel grids [6].

Instead of predicting the radiance for every point sampled along a ray and integrating the results, neural light field (NeLF) models directly predict the integrated radiance along a ray. Sitzmann et al. [19] introduced a full 360 degree NeLF model through the use of Plücker coordinates and meta-learning. This method turned out to be ∼15000x faster than NeRF for simple scenes. Finally, Wang et al. [21] propose using the outputs of an already-trained NeRF model to learn the light field function. This achieves 28-31x speedup compared to the teacher NeRF and 1.4-2.4 dB average PSNR improvement. Instead of Plücker coordinates, they employ an approach similar to the original NeRF, by concatenating sampled points along a ray and then feeding it into the network. Additonally, the results shown in the paper are consistent with complex scenes.

As fast as these methods have become, their purpose is to render a fully colored image. Since creating a silhouette or a depth map is a relaxation of the initial problem, it is evident that speedups can be achieved by creating a new model specific to this.

Knowledge distillation is a technique through which a model can transfer its knowledge to another. Buciluǎ et al. [2] describe this method and find no significant loss in performance when using data generated by the initial model to train a new one. Since then, this method was explored further and proved to be invaluable for many deep learning applications [7]. In particular, knowledge transfer between different architectures [4] and compression [15] interest us in the context of our problem. In order to transfer knowledge between an existing NeRF model and our new model, we will simply regress the output data from the NeRF model.

Shadow rendering As opposed to ray-traced shadows, which are inherently physically consistent, in rasterization, shadows are only clever approximations. Rasterized shadows are usually computed by rendering a depth buffer from the point of view of the light source and generating shadow maps from the output. This technique was introduced in 1978 [22] and is still commonly used to this day [20].

Simple depth buffers are, however, not able to simulate soft shadows. For this purpose, multiple occlusion textures rendered at different depths from the light position can be used [3]. Intuitively, NeRFs seem to be suitable for rendering these occlusion textures by sampling points along the generated rays at each depth slice. However, this would not result in any significant speedup, since we would still need to evaluate the network multiple times per ray. For this reason, this paper will focus on hard shadows generated using depth maps.

[1] https://mihneatoader.github.io/Depth-Light-Field-Training

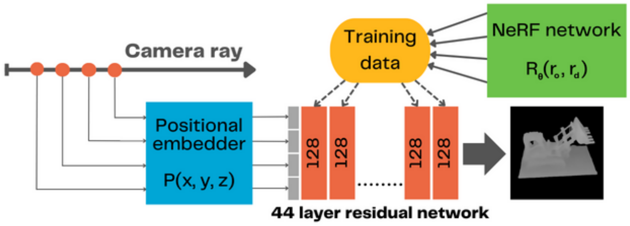


Figure 1: Sketch of the final network design

3 Depth Light Field Training (DeLFT)

In essence, the goal of our research is to explore the viability of extracting a subset of the information encoded in a neural scene model and training a new, faster model. In this context, we need to consider multiple influential factors: scene representation, training data, network design, and ray representation. Figure 1 shows a visualization of the final proposed network design.

For this purpose, Wang et al.'s approach [21] was invaluable in the development of a solution, since the issue they tackle is somewhat orthogonal to ours. Furthermore, they have made their source code available, which provided a strong baseline, without which the timeline of the project would have been infeasible.

Scene representation

As discussed previously, NeRF models without speedup structures suffer from slow render times because of multiple network forwards per ray. As such, light fields seem like a suitable candidate for our case, since they only require a single forward per ray. However, learning a light field representation from a sparse input set is a much more complex problem than it seems at first. Attal et al. [1] expands on this issue, stating that while NeRFs have the luxury of observing most points in the 3D scene multiple times, learning a 4D ray space from 2D images requires much more special consideration. In their paper, they employ a ray-space embedding to solve the issue of insufficient data. In our case, we have access to an already trained radiance field model which we can leverage to omit the issue of insufficient data.

Training data

Using a converged NeRF model of the scene, we can generate training data. In fact, no specific NeRF implementation is necessary, as long as the chosen model also outputs depth data. To do this, we sample rays within a user-defined scene-bounding sphere and save the outputted depth data, as well as the rays themselves. More specifically, let R_θ be the radiance field function and x_o, y_o, z_o, x_d, y_d, z_d the origin and direction of the sampled ray. Then,

$$R_\theta((x_o, y_o, z_o), (x_d, y_d, z_d)) = \hat{\lambda} \tag{1}$$

where $\hat{\lambda}$ is the depth at which the ray terminates. Let L_θ be the desired light field function, which we can then estimate by training it using the Mean Squared Error function:

$$\mathscr{L} = MSE(L_\theta((x_o, y_o, z_o), (x_d, y_d, z_d)), \hat{\lambda}) \tag{2}$$

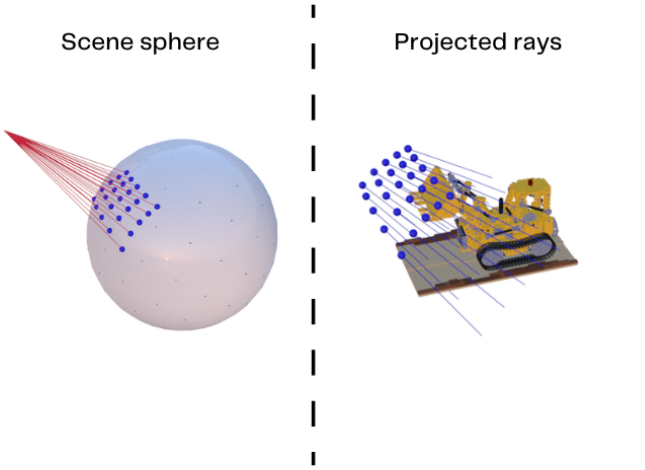


Figure 2: Illustration of ray reprojection onto the scene sphere

Network design

In order to fully leverage the amount of information made available by the existing NeRF model, a change in the network architecture is needed. Wang et al. [21] propose a deep residual network in their work, 256 nodes wide and 88 layers deep. Since our model will only estimate the depth at which the rays terminate instead of the RGB values, we employ a smaller network (128 nodes wide, 44 layers deep), which will lead to similar results. More information regarding this will be made available in the ablation study.

Ray representation

The proposed approach involves concatenating the spatial coordinates of K sampled points along a ray to construct an input vector (3K-d), which is then fed into a positional embedder and finally, the DeLFT network. The incorporation of this ray representation method within the DeLFT framework addresses the need for an effective representation scheme in capturing the complex information associated with light fields. By leveraging the spatial coordinates of multiple sampled points along a ray, the proposed approach offers a simple yet powerful solution for accurately encoding the underlying ray properties [21].

Ray space reprojection

Our proposed solution of sampling rays inside of the bounding sphere of the scene means that the neural network will not be able to correctly estimate the light field function outside of the specified area. In many cases of classical rendering, the light source is positioned far away from the position of the objects, which clearly indicates an issue.

Thankfully, most rendering scenarios do not position lights inside of objects. Having made this assumption, we discover that the position of the origin along the line that defines the ray does not change the outcome of the estimated depth. Additionally, all the learned rays inside of the bounding sphere define the entire ray space. This gives us two options to solve our problem: Plücker coordinates or reprojecting the rays on the bounding sphere. In the case of Plücker coordinates, Wang et al. [21] report that representing rays as such decreases the quality of the outputs. Therefore, we will simply reproject outside rays using ray-sphere intersections.

Figure 2 illustrates the ray reprojection process for a view outside of the bounding sphere. The red rays signify the initial projected rays from the viewport. Subsequently, the intersection points with the bounding sphere are calculated (blue points) and replace the ray origins. The offset between the initial origin and each intersection point are kept in memory and added to the output of the network to arrive at the real depth.

4 Experimental Setup and Results

4.1 Training setup

All models were trained on a GTX 1070, with 8GB of VRAM. Because of the lack of computing resources, all images are down-sampled by 2× (400x400) during training and testing.

Datasets In consideration of the project timeline and the primary objectives of our study, we have opted to focus our analysis on a single dataset, namely the original NeRF dataset. This decision is guided by both practical constraints (training time, available computing resources) and the research scope of the paper. While evaluating the efficacy and performance of our neural radiance field model on multiple datasets could offer broader insights, it is not central to the specific research questions we aim to answer. Additionally, the NeRF dataset contains both synthetic and real-world datasets, which together offer sufficient information in regards to the utility of the model.

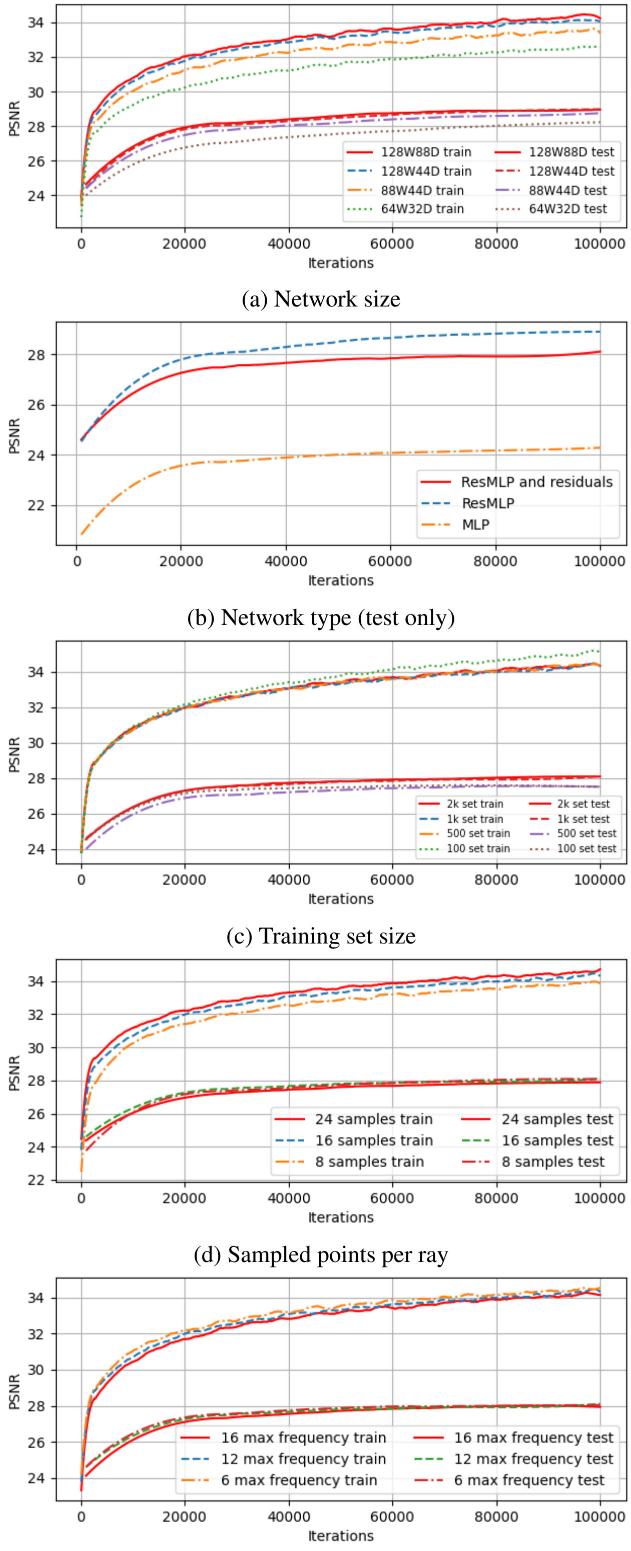
Implementation details As mentioned previously, our solution would work regardless of the NeRF implementation used, as long as it also outputs depth information. Since both implementations of it and pretrained models are publicly available online (nerf-pytorch [23]), we decided to use the standard NeRF model introduced by Mildenhall et al. [11]. Integration of new models with significant speedups, such as Instant-NGP [12], would result in much faster data acquisition. This was, however, outside the scope of our project.

In the upcoming sections we will discuss the performance of the trained models, how different network configurations compare to each other, the evaluation metrics used and whether the results are accurate enough for use in classical rendering.

4.2 Ablation study

In order to obtain the best performance at the lowest computational cost, we need to analyze the quality of our model given parameters that concern both the network and the training setup. This section analyzes the convergence of the Peak Signal to Noise Ratio (PSNR) over time, during the training of the models.

Starting with the size of our deep network, we tested multiple different width and depth configurations. Figure 3a shows the training progress at every 1000 iterations on an interval of 100.000 iterations. As would be expected, larger networks perform better (higher PSNR cap). However, the architecture



(a) Network size

(b) Network type (test only)

(c) Training set size

(d) Sampled points per ray

(e) Maximum frequency for positional encoding

Figure 3: PSNR during training. All models are trained for 100k iterations on the lego synthetic scene.

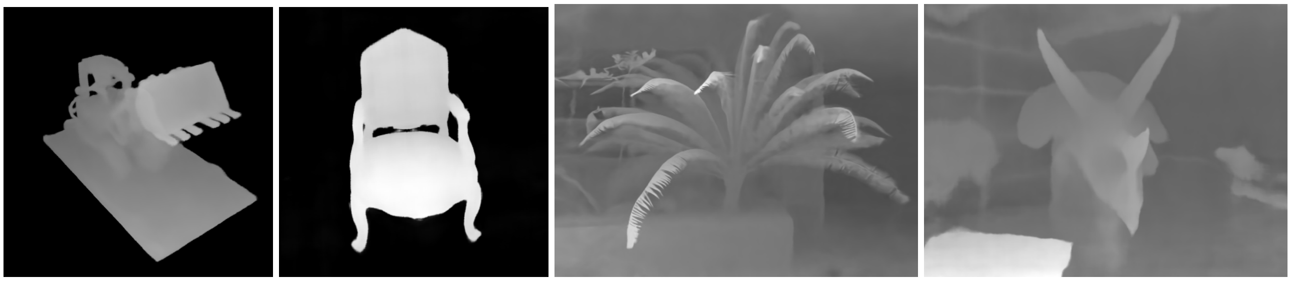


Figure 4: Rendered DeLFT novel poses for different synthetic (left) and LLFF scenes (right)

shows diminishing returns around the 128 width, 44 depth model; the next model is twice as large, but the difference in PSNR cap is minimal (29.17 vs 29.13 PSNR). For this reason we chose the 128 width, 44 depth model as the standard model for all our future experiments. Regarding training time, the chosen architecture takes approximately 11 hours to reach 100.000 iterations on the described setup.

Wang et al. [21] found significant performance improvements using residuals in their network (∼20 vs ∼16 test PSNR). Since the purpose of our network differs from theirs, we wanted to validate this finding in our model as well. Additionally, we also included a simple MLP with no skip connections in our experiments. Figure 3b shows that while both residual MLPs outperform the one with no skip connections, including residuals shows a decrease in the quality of the model.

The size of the training set is important considering both model performance and data generation time. The more training rays, the better the quality of the outputs, however, every 400x400 batch of rays takes ∼20 seconds to compute from the original NeRF model. With the next experiment, we would like to find a good compromise between the two metrics. Figure 3c shows similar test performance between the 2000 and 1000 sized training sets, as well as between the 500 and 100 sized training sets. Based on this observation, we will train all subsequent models on training sets of size 1000.

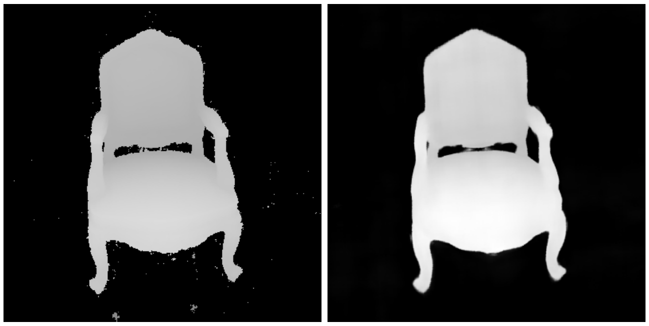
The final considerations for our model involved the amount of points samples per ray and the maximum frequency of our positional embedder. Figures 3d and 3e show that these parameters do not significantly impact the performance of the model if chosen between the specified intervals, since the network performances converge similarly for both training and testing. It is important to note that the ”x max frequency” mentioned in the figure refers to the $\log_2$ of the maximum frequency for positional encoding. Given these results, we will use 16 sampled points per ray and a maximum encoding frequency of 12 in our final networks.

4.3 Model quality

Using the final configuration discussed in the previous sections, we then trained multiple models on the NeRF dataset, using both synthetic and local light field fusion (LLFF) scenes. The final evaluation results of these models are presented in Table 1. All network forwards take approximately 0.2 seconds. The results for the synthetic scenes are gathered from novel poses with respect to the ground truth. The LLFF scenes do not have available ground truth depth data, so no comparison could be made. Instead, we display the evaluation metrics as compared to the training data. Figure 4 shows the output of the network on novel poses in the synthetic and LLFF scenes.

Scene		PSNR	MSE	SSIM
Synthetic	Lego	28.3468	0.0015	0.9821
	Microphone	26.5106	0.0022	0.8832
	Chair	24.9331	0.0032	0.8855
LLFF*	Fern	29.8775	0.0010	0.9925
	Horns	27.0494	0.0020	0.9635

Table 1: Evaluation metrics for multiple converged scenes, trained for 100k iterations. (*) LLFF scenes do not have ground truth depth data available, so train PSNR is displayed.

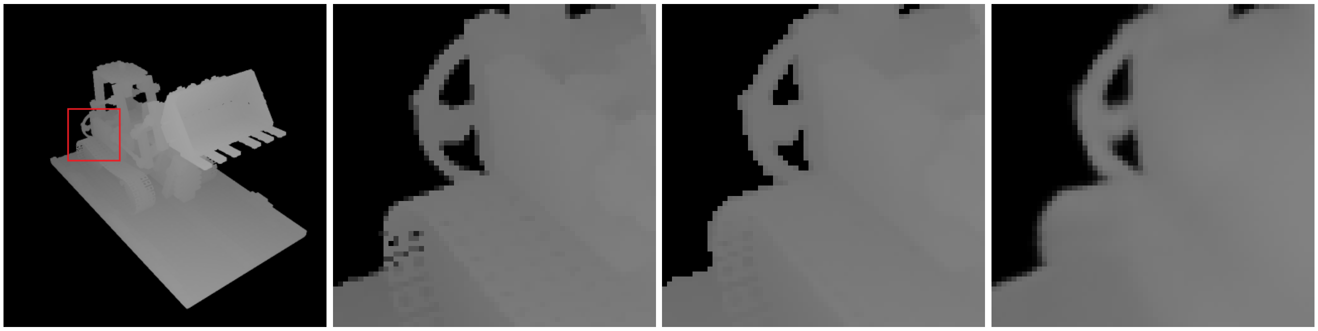


(a) NeRF

(b) DeLFT

Figure 5: Comparison between NeRF and DeLFT noise

Figure 6 serves as a qualitative comparison between ground truth, NeRF generated and DeLFT generated depth maps. A few observations are note-worthy: compared to the original NeRF model, the DeLFT renders show less granularity for small features, such as the holes in the tracks. However, silhouette edges are approximated better than in the NeRF render. Additionally, NeRF depth outputs have a tendency to be noisy, as can be seen in Figure 5. By observing the ray termination depth from different viewpoints, the DeLFT model is able to suppress the noise and generate clean


(a) Ground truth full image

(b) Ground truth

(c) NeRF

(d) DeLFT

Figure 6: Visual comparison between ground truth, original NeRF and DeLFT depth maps

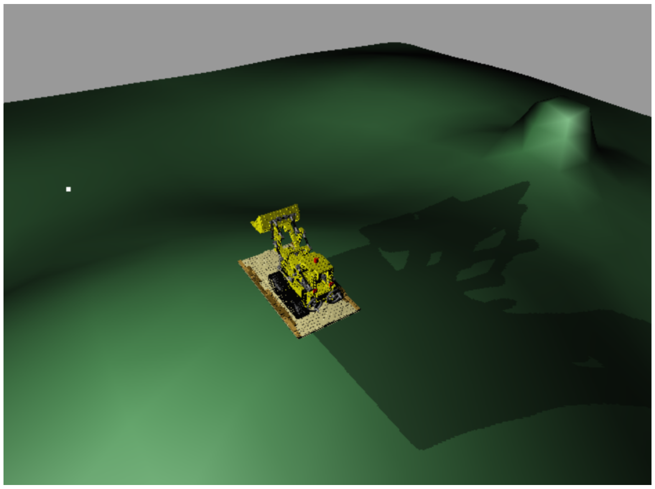


Figure 7: Render of shadow demo

depth maps.

4.4 Shadow demo

To prove the utility of our solution, we developed a simple shadow mapping demo that makes use of the generated depth maps. This demo serves as a tangible showcase of the utility and versatility of our solution in real-world scenarios and highlights the potential applications in computer graphics, gaming, or virtual reality. Figure 7 shows a rendered still-frame from this demo. The shadows are computed using depth maps generated by the model using the light position as the rays' origin. Visually, the results are adequate and offer sufficient information.

5 Responsible Research

In any field, regardless of apparent innocuousness, responsible research practices should be carefully considered. By embracing these practices, researchers in the field can foster trust and enhance the quality and impact of their work. The realm of neural scene representations is no different; in fact, transparent reporting and reproductibility has contributed to the ever faster development of NeRF technology. In this sense, our contribution should strive to reach the same standards put forth by the community. This section will touch upon the collection and usage of data, reproductibility of our findings and the review mechanism our paper went through.

In alignment with responsible research practices, we utilize existing data rather than collecting it ourselves. By utilizing well-established datasets that have undergone rigorous scrutiny and approval processes, we minimize the potential privacy concerns that arise from data collection. Additionally, using these datasets gives way for our findings to be verified by independent researchers, promoting transparency and fostering a collaborative scientific community.

As many papers before have done, we also make our source code publicly available, in order to aid reproductibility. In fact, much of our code is based on public solutions, which highlights why responsible research practices are important.

Finally, we strive to report transparently on our findings, with no personal biases or vested interests. For this to happen, our paper went through both a peer review, as well as a review from our supervisors and responsible professor before finalisation.

6 Conclusions and Future Work

We introduced a deep residual network with fast evaluation times ($\sim$0.22 seconds) that learns a subset of the data encoded within a NeRF model, that is, ray termination depth. By using output data from a converged NeRF model to train this new network, we address the issues of unavailable depth data within non-contact passive 3D scanning and insufficient training data for neural light field scene representations. The light position is not restricted to an area around the rendered object due to our use of ray reprojection, making this representation more effective for common rendering scenarios. Therefore, the outputs of the network prove to be useful when rendering convincing rasterized shadows on non-NeRF targets. Casting shadows on the NeRF object itself requires scene relighting, so self-shadows are only possible if the object is converted to a mesh.

On an outdated system configuration, it takes around 6 hours to generate the training data and 11 hours for the network to converge. Further speedup might be achieved in both training and evaluation times by investigating recent speedup structures discussed in papers the likes of Instant-NGP[12]. Additionally, generating training data and training the network simultaneously could be possible with modern systems, which would greatly reduce overhead.

References

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

[2] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 535–541, New York, NY, USA, 2006. Association for Computing Machinery.

[3] Elmar Eisemann and Xavier Decoret. Plausible image based soft shadows using occlusion textures. In *2006 19th Brazilian Symposium on Computer Graphics and Image Processing*, pages 155–162, 2006.

[4] Krzysztof J. Geras, Abdel rahman Mohamed, Rich Caruana, Gregor Urban, Shengjie Wang, Ozlem Aslan, Matthai Philipose, Matthew Richardson, and Charles Sutton. Blending lstms into cnns, 2016.

[5] Abid Haleem and Mohd. Javaid. 3d scanning applications in medical field: A literature-based review. *Clinical Epidemiology and Global Health*, 7(2):199–210, 2019.

[6] Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5875–5884, October 2021.

[7] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[8] David B. Lindell, Julien N. P. Martel, and Gordon Wetzstein. Autoint: Automatic integration for fast neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14556–14565, June 2021.

[9] Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floornet: A unified framework for floorplan reconstruction from 3d scans. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[10] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15651–15663. Curran Associates, Inc., 2020.

[11] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

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

[13] T. Neff, P. Stadlbauer, M. Parger, A. Kurz, J. H. Mueller, C. R. A. Chaitanya, A. Kaplanyan, and M. Steinberger. Donerf: Towards real-time rendering of compact neural radiance fields using depth oracle networks. *Computer Graphics Forum*, 40(4):45–59, 2021.

[14] Jeffrey A. Okun and Susan Zwerman. *The VES Handbook of Visual Effects: Industry Standard VFX Practices and Procedures*, chapter Chapter 3: Acquisition/Shooting. Taylor & Francis, 2020.

[15] Antonio Polino, Razvan Pascanu, and Dan Alistarh. Model compression via distillation and quantization, 2018.

[16] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14335–14345, October 2021.

[17] Fabio Remondino. Heritage recording and 3d modeling with photogrammetry and 3d scanning. *Remote Sensing*, 3(6):1104–1138, 2011.

[18] Fabio Remondino and Sabry El-Hakim. Image-based 3d modelling: a review. *The photogrammetric record*, 21(115):269–291, 2006.

[19] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 19313–19325. Curran Associates, Inc., 2021.

[20] Naty Hoffman Tomas Akenine-Möller, Eric Haines. *Real-Time Rendering, Fourth Edition*, chapter Chapter 7: Shadows. CRC Press, 2018.

[21] Huan Wang, Jian Ren, Zeng Huang, Kyle Olszewski, Menglei Chai, Yun Fu, and Sergey Tulyakov. R2l: Distilling neural radiance field to neural light field for efficient novel view synthesis. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 612–629, Cham, 2022. Springer Nature Switzerland.

[22] Lance Williams. Casting curved shadows on curved surfaces. *SIGGRAPH Comput. Graph.*, 12(3):270–274, aug 1978.

[23] Lin Yen-Chen. Nerf-pytorch. https://github.com/yenchenlin/nerf-pytorch/, 2020.

[24] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *Proceedings of the*

IEEE/CVF International Conference on Computer Vision (ICCV), pages 5752–5761, October 2021.

[25] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting, 2021.