Proposal for Master's Thesis: Accelerations for Training and Rendering Neural Reflectance and Radiance Fields

Ilia Mazlov
Mat. Nr.: 3056366

April 17, 2021

1 Introduction

The task of reconstructing a 3D scene from a set of 2D images is a central long-standing problem in computer graphics for quite a long time. The possibility to reproduce its appearance under novel lighting and viewing conditions makes the task even more complex. Some approaches to solve this task has already been proposed [Lombardi et al., 2019], [Mildenhall et al., 2020], [Bi et al., 2020], [Liu et al., 2020]. However, most of them struggle with low quality results, inability to use new lighting and viewing conditions, high time costs or low rendering speed.

In this thesis, I am going to develop a solution that overcomes or at least mitigates the aforementioned problems. The solution is going to model appearance with view- and light-dependant effects and imply colocated as well as non-colocated light sources. Some optimization techniques are going to be used in order to make the solution feasible within achievable hardware setup.

2 Related work

The vital part of achieving novel views for complex scenes is the scene representation, which consists of two main parts: scene geometry and light interaction.

Scene representation. There exist two main groups of methods for scene representation, namely: *explicit* and *implicit* representations. *Explicit* methods use geometric primitives to describe a scene. The most basic representations are voxel (occupancy) grids [Liu et al., 2020], point clouds [Qi et al., 2017] and meshes [Jack et al., 2018]. *Implicit* methods map points in space to some value, which implicitly gives knowledge about the scene. The most known representation is a signed distance field [Curless and Levoy, 1996].

Light interaction. The light interaction techniques determine the material properties on the surfaces. The bidirectional subsurface scattering reflectance distribution functions (BSSRDFs) are 7-D functions and are not commonly used in due to its complex nature. The most common way to describe the appearance of the surface are bidirectional reflectance distribution functions (BRDFs), which are 5-D functions. BRDFs are usually used within their analytical representations [Oren and Nayar, 1994], [Cook and Torrance, 1982], [Phong, 1975]. When a BRDF is changing across a surface, it is referred to as a spatially-varying BRDF (svBRDF).

Free-viewpoint rendering of real world scenes. [Lombardi et al., 2019] use memory inefficient voxel grid as scene geometry representation and only account for the scene color without any light interaction. However, the main idea of using neural networks (NN) as neural volumes became a key feature to a bunch of later works.

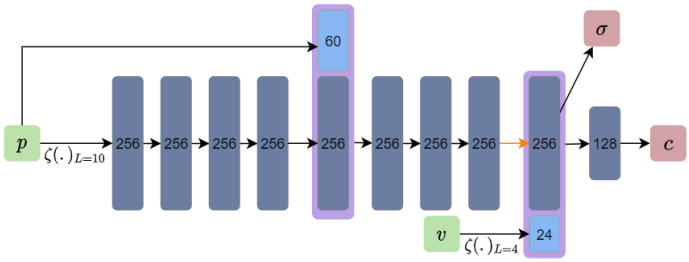

Figure 1: MLP used in NeRF [Mildenhall et al., 2020] Dark blue boxes represent hidden layers. Black arrows indicate FC-layers with sigmoid activation, orange arrows - FC-layers without activation. $\zeta(.)$ stays for positional encoding function. σ is 1D volume density, c is a 3D color value.

[Sitzmann et al., 2019] also use neural networks: scene representations networks (SRNs). The model appearance is not modelled explicitly (only implicitly in latent feature vectors). Use (in contrary to [Lombardi et al., 2019]) implicit geometry representation. Similar work is [Saito et al., 2019].

[Park et al., 2019] propose deep signed distance functions (DeepSDF), which stores scene information in the network. It can be extended to multi-shape version and is memory efficient.

[Mildenhall et al., 2020] reaches appealing results by combining (implicit) neural scene representation together with ray marching technique. Authors employ an MLP to implicitly represent the scene by taking positional inputs and returning the color at the given point. The scheme of used MLP is outlined in Figure 1. Authors also use some improvements such as: *positional encoding* for more detailed renders and *hierarchical volume sampling* for increasing efficiency of the approach. Although this method allows novel view synthesis, the scene appearance is not modeled. Another problem is an overall inefficiency of the approach.

[Bi et al., 2020] introduce neural reflectance field (NRF), which is a neural scene representation that implicitly encodes scene geometry (in form of volume density), normal and reflectance properties of the scene. Authors extend the idea of [Mildenhall et al., 2020] by implying single-bounce single-point direct illumination from non-colocated light source by adding light transmittance term to the rendering equation. This requires using ray marching algorithm to the light ray as well. Although the same technique of *positional encoding* and *hierarchical volume sampling* has been used, the rendering is still highly time inefficient, especially under novel non-colocated light and view. Even precomputing the light transmittance volume does not help much. Due to these performance difficulties authors only consider colocated light source in their experiments. A big advantage of the algorithm is a possibility to use and arbitrary appearance model. In their work authors use microfacet BRDF that combines diffuse Lambertian term with specular term that uses GGX distribution [Walter et al., 2007]. For furry objects the fur reflectance model [Kajiya and Kay, 1989] is used.

[Liu et al., 2020] try to alleviate this problem by introducing neural sparse voxel fields (NSVFs), which leverage the usage of sparse voxel octrees for scene representation. The usage of classical ray marching approach together with combination of implicit NeRFs and explicit octree structure results in more than 10 times faster renders comparing to [Mildenhall et al., 2020]. Authors also use a bunch of techniques such as: *self-prunning* of octrees and *progressive training* for efficient handling essential and non-essential voxels. However, the light interaction is not modelled meaning that only global illumination is going to be implicitly represented in the model.

[Rebain et al., 2020] propose a framework to decompose the scene into multiple parts in order to increase the time efficiency of [Mildenhall et al., 2020]'s approach. This implies parallel evaluation of the NeRF networks, which are responsible for different parts of the scene decomposed using Voronoi

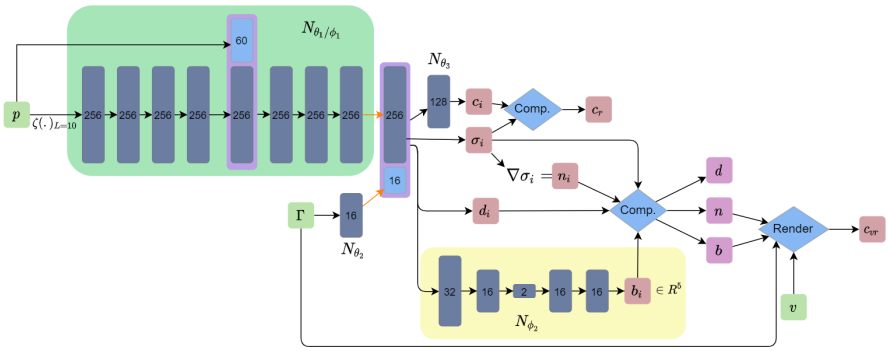

Figure 2: **NeRD Architecture** [Boss et al., 2020]. The architecture includes several networks. The training consists of two parts: training sampling network and training decomposition network. 1. Training sampling networks involves training the upper part: passing position p and Spherical Gaussians (SGs) Γ to the networks and evaluating c_i and σ_i, which are composed along the ray resulting in c_r. The MSE loss function is used on c_r to train sampling networks. 2. The second training part affects decomposition network (lower part). This time network N_{θ_1/ϕ_1} produces density σ_i and direct color d_i. Normal n_i is calculated based on gradient from the density. BRDF parameters b_i are the result of network N_{ϕ_2}. After composition along the ray the scene is rendered using SGs Γ and view direction v resulting in final color c_{vr}.

Diagrams [Aurenhammer, 1991] based method. The final image is composed using Painter's Algorithm [Newell et al., 1972]. This method does not model any light interaction thereby limiting scene reconstruction under novel light conditions.

[Boss et al., 2020] follows the NeRF architecture, but introduce explicit decomposition and rendering steps. The authors use 2 networks: first for sampling points along the ray and the second one for estimating color and normal information as well as BRDF parameters of the scene, which implies physically plausible output values. The big advantage of this decomposition approach is a textured mesh extraction. The overview of the MLP architecture is shown on Figure 2.

[Srinivasan et al., 2020] extend the original NeRF approach not only for novel view but also for novel direct and indirect (one bounce) light conditions. The authors exploit a quite simple idea of predicting visibility and surface fields for the scene along with other appearance properties. These fields are used later for illumination estimation, which is plugged into rendering integral. Authors use 2 networks instead of one used in NeRF for predicting the volume density and appearance parameters. The normal map is calculated as negative gradient vector of the volume density predicted by MLP. Another (the 3rd) network learns to predict visibility and max. depth information, which allows to efficiently model the scene under one-bounce indirect illumination. The scheme of used MLPs is shown on Figure 3

A few recent works also focus on improving performance of such a group of approaches as 3D scene reconstruction. [Lindell et al., 2020] employ neural networks for faster approximation of integral calculations in volume rendering tasks. [Tancik et al., 2020] improves the optimization trajectory of the model in coordinate-based neural representations such as NRF using so-called "Meta Initialization" instead of standard initialization of neural models.

[Guo et al., 2020] propose to adjust NeRF setup with two types of light-rays: intra-object (within the object) and inter-object (within the scene). The intra-object light is modeled implicitly with MLP (light scattering), but inter-object light rays are traced explicitly (shadows casting).

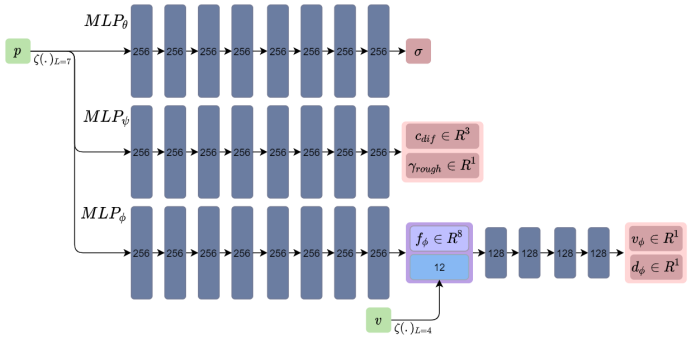

Figure 3: Schematics for NeRV [Srinivasan et al., 2020] approach. Two separate MLPs MLP_θ and MLP_ψ are used to predict volume density σ and appearance parameters diffuse albedo c_{dif} as well as roughness γ_{rough}. The 3[rd] MLP predicts visibility v_ϕ and max. depth d_ϕ values. Dark blue boxes represent hidden layers. Black arrows indicate FC-layers with ReLU activations.

3 Proposed Solution

The proposed solution involves NRF as a scene representation together with the ray marching framework as proposed in [Bi et al., 2020]. This allows not only to take into account view-dependant effects but also consider relighting re-rendered scene. In order to reduce the time consumption of the approach the effectiveness of sparse voxel octrees [Liu et al., 2020] can be leveraged. The extended NSVF architecture is then looks like as shown in Figure 4

Although this solution suppose to be more time effective, there is still a limitation for the colocated light source, because the light rays have to be sampled in a similar way the view rays are sampled (Figure 5, left). Assuming that the media inside the voxels is homogeneous this approach is going to be extended to non-colocated light sources by using approximation instead of real sampling for the light rays, as shown on Figure 5 (right).

Originally the approach implies the usage of explicit reflectance model. That means that the produced output is limited by used BRDF, e.g. the subsurface scattering effects are not modelled in BRDFs. However, the implicit neural representation of reflectance model can remove these restrictions allowing to reproduce richer light- and view-dependant surface effects. The Figure 6 shows the network architecture based on NSVF approach ([Liu et al., 2020]), which is extended to model single point light illumination.

In original approach the input view direction is defined in global coordinate system. In this case the MLP learns how the point looks like from the given view direction. However, the appearance at the point is highly dependant on the normal of the surface. That's why the network can be nudged to learn material appearance properties by using view and light directions transformed into local tangent frame. The normal direction can be calculated using negative normalized gradient vector of the σ.

3.1 Road map

1. *Prepare synthetic dataset* The most of relevant works do not principally model light sources, which means that these datasets are not useful for the stated task. [Bi et al., 2020] use datasets with colocated light, however authors did not expose any implementation as well as datasets from their work. [Srinivasan et al., 2020] also use suitable synthetic datasets, but at the moment of writing they are not

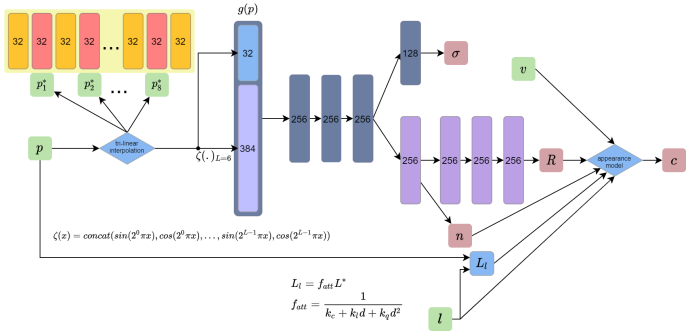

Figure 4: The green boxes indicate inputs, the red boxes indicate outputs. Conceptually the architecture repeats the one proposed in [Liu et al., 2020]. However, instead of color the network outputs appearance vector R, which is then used (together with light and view directions, normal and light intensity) to evaluate appearance model. The L_l denotes the light radiance considering the point light falloff.

publicly available. This is the reason to create a synthetic dataset that contains the scene rendered from different viewpoints with colocated and non-colocated light sources.

2. *Sparse voxel octrees* Combine sparse voxel octrees optimization technique with the NRF. NSVF provide decent implementation which is going to be extended with appearance model from NRF. This also requires implementation of the proper rendering procedure. The model from [Bi et al., 2020] outputs not color values but the reflectance parameters. This requires to apply the BRDF in order to get the color values of the rendering.

3. *Training/Evaluation stage* The training of the model is going to be done using prepared synthetic dataset consisting of samples with colocated light sources.

    - The "test dataset" has to be chosen in a way it does not have any close-neighbour points, in order to test the model for generalization.
    - The evaluation is to be compared with results of NRF-based approach [Bi et al., 2020]

4. *Neural reflectance model* Adjust the model such that it also implicitly represents the reflectance of the scene. This is going to be done by changing the structure of the neural reflectance field such that it outputs the color values instead of the reflectance parameters. This means that the reflectance model is implicitly modelled in the neural reflectance field. The model capacity might need to be adjusted in order to correspond to increased complexity of the model.

5. *Training/Evaluation stage* Repeat training with the neural reflectance representation.

    - The evaluation involves comparison with base-line methods (more on that in Section 4)

6. *Non-colocated light extension* Extension for non-colocated light sources requires an approximation step in order to make the approach feasible. The offered technique is to approximate light transmittance only according to depth inside the sparse voxels instead of applying the whole ray marching framework for the light rays.

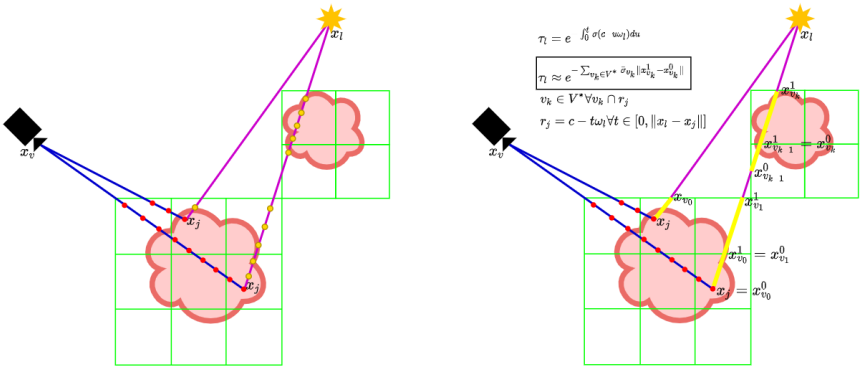

Figure 5: The sketch of how the view and light rays are sampled on the scene. The left part shows the same idea as used proposed by [Bi et al., 2020] with the difference of using sparse voxel octree optimization (SVO), proposed by [Liu et al., 2020]. A lot of samples have to be sampled on light rays, which is very computationally demanding even with using SVO approach. The right part shows the proposed approximation for sampling the light rays. Assuming that the density inside each voxel is homogeneous, the collective transmittance of the voxels intersected by the ray is dependant on the distance the ray travelled inside the voxel. The transmittance decay outside the voxels is assumed negligible.

- Figure 5 (right) shows formulas for calculating light transmittance inside voxels. The $\bar{\sigma}_{v_k}$ indicates an average transmittance inside the voxel. For efficiency purposes it has to be precomputed and recalculated throughout the training.

7. *Training/Evaluation stage* Training the model on synthetic dataset with the non-colocated light sources.
   - Compare results with the results from the previous step and with base-line methods as well
8. *Real-world dataset* [Bi et al., 2020] use some real-world scenes captured with a cell phone. Although this dataset contains only colocated light source (cell phone flashlight), it can still be used for training the model. Another source of real-world data can be captured with TAC7 scanning device [X-Rite,], which uses sophisticated camera and light setup.
   - Since TAC7 extracts the height map of the scene, it can be used for increasing performance of the method. Namely such technique as "self-pruning" for sparse voxel octrees is not needed anymore, because the information from the height map can be used to filter out empty voxels.
9. *Training/Evaluation stage* Training and evaluation of the model on real-world dataset.

4 Evaluation

A vast majority of related works do not model light interaction, which makes them being as not good candidates to compare the results with.

[Bi et al., 2020]'s approach is a basement of the proposed solution, therefore it is considered as a baseline method.

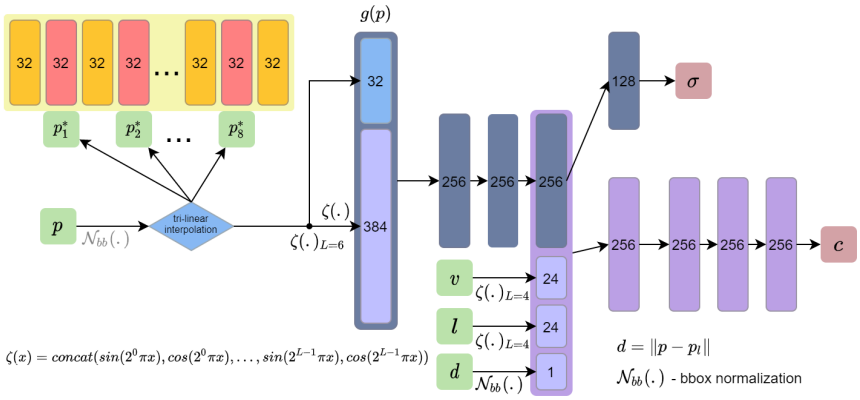

Figure 6: The original NSVF ([Liu et al., 2020]) network is extended for modelling single point light illumination. The positionally encoded light direction l is being concatenated with the distance to the point light source d, view direction v and 3rd hidden layer of denisity predictor to form a first layer of the Texture predictor that outputs color value c. Only 4 hidden layers of Texture predictor might not be enough and might have to be increased.

Although [Boss et al., 2020] do not use light source in their setup, but they still extract complex appearance parameters, which makes the results to be comparable with those from the proposed solution.

[Srinivasan et al., 2020] model both direct and indirect illumination and consequently use datasets captured with light positions. Since this work is one of the closest works to the proposed approach, it should definitely be compared qualitatively as well as quantitatively (evaluation timings) with the results to be obtained.

5 Implementation details

Implementation details of the proposed solution may undergo some changes during the working process, however the rough plan consists of the following:

- *Neural scene representation frameworks* [Liu et al., 2020] accompany their paper with the implementation, which is going to be used as a base implementation of the project and includes:
  - Python3
  - PyTorch (using Fairseq framework [Ott et al., 2019])
  - CUDA
  - Nvidia Apex library
- *Dataset-related tools* Tools:
  - OpenGL-based rendering framework [Thies et al., 2019] is going to be used for creating synthetic dataset
  - Blender (alternative rendering option)

  Datasets:

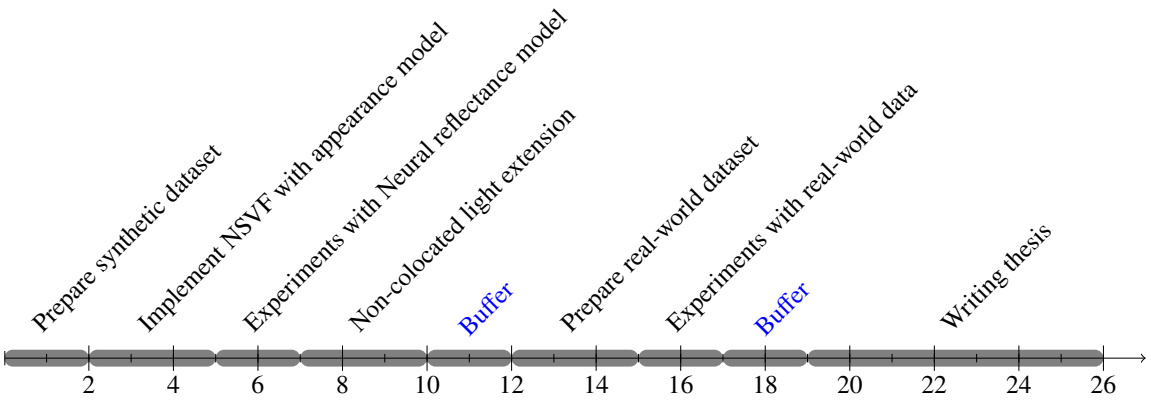

Figure 7: Timeline in weeks since the registration.

- Datasets captured by [Bi et al., 2020] (when they will be available)
- TAC7 Appearance Scanner [X-Rite,] for capturing real-world dataset

6 Extension points

Another way to increase the speed of the approach (2-3 times faster) is to decompose it into sub-scenes using Voronoi-based decomposition and employ multiple networks to these parts as proposed by [Rebain et al., 2020]. As soon as these implementations become available it can be plugged into the proposed method.

Auto integration technique from [Lindell et al., 2020] has good prerequisites to be adopted to the proposed method and can also be used to make the method up to 10 times faster.

The proposed solution considers light ray to be represented the same way the view ray is represented using 3D vector. However, light ray can also be parametrized using half-vector, which allows to drop one dimension down assuming isotropic reflection of the scene. Experiments with this kind of parametrization are planned as an extension point of this work.

In [Bi et al., 2020] the GGX distribution is used for specular term of microfacet BRDF model. However, the better results might be achieved by using symmetric variant of distribution - SGGX [Heitz et al., 2015].

7 Timeline

Figure 7 sketches how I plan to use the 26 weeks from the registration of the thesis to its deadline. The timeline echoes the road map described in Section 3.1. It contains in addition two extra buffer periods, which are supposed to help me to stay on or ahead of the schedule in case of any unexpected complications. The last weeks are reserved for writing the thesis text. Note that this is only a rough plan on how the process is going to work out. The phases are usually blending into each other.

References

[Aurenhammer, 1991] Aurenhammer, F. (1991). Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405.

[Bi et al., 2020] Bi, S., Xu, Z., Srinivasan, P., Mildenhall, B., Sunkavalli, K., Hašan, M., Hold-Geoffroy, Y., Kriegman, D., and Ramamoorthi, R. (2020). Neural reflectance fields for appearance acquisition.

[Boss et al., 2020] Boss, M., Braun, R., Jampani, V., Barron, J. T., Liu, C., and Lensch, H. P. A. (2020). Nerd: Neural reflectance decomposition from image collections.

[Cook and Torrance, 1982] Cook, R. L. and Torrance, K. E. (1982). A reflectance model for computer graphics. *ACM Trans. Graph.*, 1(1):7–24.

[Curless and Levoy, 1996] Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, page 303–312, New York, NY, USA. Association for Computing Machinery.

[Guo et al., 2020] Guo, M., Fathi, A., Wu, J., and Funkhouser, T. (2020). Object-centric neural scene rendering. *arXiv preprint arXiv:2012.08503*.

[Heitz et al., 2015] Heitz, E., Dupuy, J., Crassin, C., and Dachsbacher, C. (2015). The sggx microflake distribution. *ACM Trans. Graph.*, 34(4).

[Jack et al., 2018] Jack, D., Pontes, J. K., Sridharan, S., Fookes, C., Shirazi, S., Maire, F., and Eriksson, A. (2018). Learning free-form deformations for 3d object reconstruction.

[Kajiya and Kay, 1989] Kajiya, J. T. and Kay, T. L. (1989). Rendering fur with three dimensional textures. In *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '89, page 271–280, New York, NY, USA. Association for Computing Machinery.

[Lindell et al., 2020] Lindell, D. B., Martel, J. N. P., and Wetzstein, G. (2020). Autoint: Automatic integration for fast neural volume rendering.

[Liu et al., 2020] Liu, L., Gu, J., Lin, K. Z., Chua, T.-S., and Theobalt, C. (2020). Neural sparse voxel fields. *NeurIPS*.

[Lombardi et al., 2019] Lombardi, S., Simon, T., Saragih, J. M., Schwartz, G., Lehrmann, A. M., and Sheikh, Y. (2019). Neural volumes: Learning dynamic renderable volumes from images. *CoRR*, abs/1906.07751.

[Mildenhall et al., 2020] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*.

[Newell et al., 1972] Newell, M., Newell, R. G., and Sancha, T. L. (1972). A new approach to the shaded picture problem.

[Oren and Nayar, 1994] Oren, M. and Nayar, S. K. (1994). Generalization of lambert's reflectance model. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, page 239–246, New York, NY, USA. Association for Computing Machinery.

[Ott et al., 2019] Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

[Park et al., 2019] Park, J. J., Florence, P., Straub, J., Newcombe, R. A., and Lovegrove, S. (2019). Deepsdf: Learning continuous signed distance functions for shape representation. *CoRR*, abs/1901.05103.

[Phong, 1975] Phong, B. T. (1975). Illumination for computer generated pictures. *Commun. ACM*, 18(6):311–317.

[Qi et al., 2017] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation.

[Rebain et al., 2020] Rebain, D., Jiang, W., Yazdani, S., Li, K., Yi, K. M., and Tagliasacchi, A. (2020). Derf: Decomposed radiance fields.

[Saito et al., 2019] Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., and Li, H. (2019). Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. *CoRR*, abs/1905.05172.

[Sitzmann et al., 2019] Sitzmann, V., Zollhöfer, M., and Wetzstein, G. (2019). Scene representation networks: Continuous 3d-structure-aware neural scene representations. *CoRR*, abs/1906.01618.

[Srinivasan et al., 2020] Srinivasan, P. P., Deng, B., Zhang, X., Tancik, M., Mildenhall, B., and Barron, J. T. (2020). Nerv: Neural reflectance and visibility fields for relighting and view synthesis.

[Tancik et al., 2020] Tancik, M., Mildenhall, B., Wang, T., Schmidt, D., Srinivasan, P. P., Barron, J. T., and Ng, R. (2020). Learned initializations for optimizing coordinate-based neural representations. In *arxiv*.

[Thies et al., 2019] Thies, J., Zollhöfer, M., and Nießner, M. (2019). Deferred neural rendering: Image synthesis using neural textures.

[Walter et al., 2007] Walter, B., Marschner, S. R., Li, H., and Torrance, K. E. (2007). Microfacet models for refraction through rough surfaces. In *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, EGSR'07, page 195–206, Goslar, DEU. Eurographics Association.

[X-Rite,] X-Rite. Tac7-scanner; x-rite. `https://www.xrite.com/categories/appearance/total-appearance-capture-ecosystem/tac7`. Accessed: 2020-12-09.

Student: _____ _____

Location, Date Signature

First Examiner: _____ _____

Location, Date Signature