THESIS

Accelerations for Training and Rendering Neural Reflectance and Radiance Fields

Ilia Mazlov

INSTITUTE OF COMPUTER SCIENCE II - COMPUTER GRAPHICS
UNIVERSITY OF BONN

First Reviewer: Prof. Dr. Reinhard Klein
Second Reviewer: Dr. Michael Weinmann

August 30, 2021

Statutory Declaration

I declare that I have developed and written the enclosed Thesis completely by myself, and have not used sources or means without declaration in the text. Any thoughts from others or literal quotations are clearly marked. The Thesis was not used in the same or in a similar version to achieve an academic grading or is being published elsewhere.

Location, Date	Signature

Summary

Photo-realistic scene reconstruction under novel viewing and illumination conditions is a challenging long-standing problem in computer graphics. Recent studies in this field have shown the applicability of deep neural networks to learn an implicit neural representation of the scene containing both geometric and appearance information about the scene.

Most works focus on extracting radiance fields under the static illumination of the scene. [Mil+20] presented the state-of-the-art approach NeRF, which learns continuous volumetric scene representation from the set of known 2D views. This allows to reconstruct the learned scene from the novel viewpoints, however, the proposed approach suffers from inefficiency. To improve NeRF's performance [Liu+21] show the application of octrees to learn Neural Sparse Voxel Fields (NSVF) that achieve up to 10 times better performance.

However, learned with NSVF radiance fields are still not implying any light interaction that allows to model light-dependent effects and reconstruct the scene under novel illumination conditions. [Bi+20a] propose Neural Reflectance Fields (NRF) that considers a single non-static point light illumination on the scene and implies additional light rays sampling, which drastically increases method complexity.

In this work, the performance limitation is addressed and several methods that allow increasing NRF's efficiency are offered. Explicit schemes *ExCol* and *ExBF* are meant to accelerate NRF approach using voxel octree structure, similarly to NSVF approach. Explicit scheme *ExVA* offers the in-voxel approximation, which makes the *ExBF* method more practical. Another *ImNRF* method is based on the original NSVF approach and implies learning an implicit neural reflectance representation of the scene.

Contents

1 Introduction

The task of reconstructing real 3D scene from a set of 2D captured images and modeling it under novel viewing and lighting conditions is a long-standing problem in computer graphics and vision. Possible complex scene geometries along with numerous appearance properties make this task a tremendously challenging problem. The real-world scenes contain a decent amount of complications that can come from both the environment (e.g. illumination is not completely static) and from the capturing hardware (e.g. distortions in camera lenses, low dynamic range (LDR) images).

The applications of realistic rendering include diverse fields, such as visualization, animation, virtual and augmented reality, visual effects, and even computer vision and robot navigation.

This task was traditionally addressed to the image-based rendering systems, which use vision-based scene geometry together with the image-based view interpolation [SK00]. However, recent works in this field make use of deep neural networks to learn implicit "neural" scene representation, containing both geometric and appearance information. [Tew+20]

[Mil+20] presented a state-of-the-art approach, which uses a deep neural network as an implicit continuous volumetric scene representation. At the core of this method lies the volume ray casting technique [DCH88], which implies such volume processes as absorption and emission while not accounting for scattering. View rays are traced through the 2D captures of the scene, sampled in a 3D space and then processed with the neural network, which predicts the volume density and color value at the given spatial point under a given viewing direction. These samples along the ray are used to predict volume density and color value, which are then gathered via the alpha-compositing process. This approach has quickly become a starting point for lots of other works [Gar+21; Liu+21; LMW21; Reb+20; Rei+21; Yu+21]. However, the main limitation for most of them is the lack of light interaction, which disables the capabilities for rendering the scene under novel light conditions.

[Bi+20a] propose the NRF approach that removes this limitation by reformulating the rendering equation in a way that it accounts for the *light transmittance*, i.e. the accumulated volume transmittance along corresponding light rays. This approach only considers a single (but not

static) point light source located on the scene. However, the proposed implementation is only practical for one specific case when the light source is co-located with the camera. Although this approach is already able to generalize for the relighting of the scene, the predictions are still affected by a very sparse light-view space sampling as there are only coincident light and view rays in the training dataset (e.g. the model never deals with shadows on the training sample images).

The goal of this thesis is to develop a solution that is able to model scene geometry and appearance with the view- and light-dependent effects implying both co-located and non-colocated light sources. This is done by utilizing the usage of neural sparse voxel fields (NSVF) [Liu+21] with such a problem formulation that considers a single non-static point light source illumination [Bi+20a]. This method is called *ExCol* (Explicit Colocated) and is only capable to be processed within the co-located setting as in NRF approach.

In order to extend this setup for the non-colocated light setting, the *ExBF* (Explicit Brute-Force) method is proposed, which traces and samples light rays for each of the view sample points to calculate light transmittance values. However, this approach is unrealistically slow and memory-demanding, which makes it impractical.

It is proposed to solve this problem by an approximation scheme *ExVA* (Explicit with in-Voxel Approximation), which leverages the usage of voxel octree structure and approximates light transmittance by the distance that light ray has traveled inside voxels. This method seems to be able to perform at the same level of quality as the initial *ExBF* scheme, though drastically outperforming it in terms of efficiency.

Additionally, the implicit neural reflectance representation *ImNRF* is proposed, which sacrifices the controlling capabilities of the neural representation for being the fastest implementation among presented schemes and having fewer limitations in the complexity of representing appearances.

To summarize, the contributions of this work are:

1. An acceleration technique *ExCol* for the NRF approach based on NSVF
2. The approximation scheme *ExVA* for the generalization *ExBF* of proposed *ExCol* that achieves comparable results with only a minor overhead
3. The implicit scheme *ImNRF* that extends NSVF to be able to learn light interaction
4. Extension for these methods from LDR to HDR data

2 Related work

This chapter reviews previous related works as well as concurrent to this thesis works.

2.1 Scene representation

The problem of scene reconstruction is an extensive challenge consisting of different aspects, one of which is scene representation. It defines the way how scene features are described. Different representations have their own properties and can be a better or worse fit depending on the task.

The scene features usually include geometry and illumination information and can be represented *explicitly* or *implicitly*.

2.1.1 Explicit geometry

Explicit methods use geometric primitives to describe scenes.

Voxel grids ([Lom+19]) is a very common way to describe the geometry of the scene mostly due to their simplicity. However, straightforwardly they are highly memory demanding. This can be handled applying techniques like multi-resolution approach ([HTM17]), octree hierarchies ([RUG17; TDB17]) or truncated signed distance fields ([CL96]), which in fact already represent geometry implicitly.

Another way to represent a geometry is using *point clouds* ([FSG16; Qi+17]). They can be easily retrieved using different sensors (e.g. depth cameras), therefore they are widely used in robotics and computer graphics fields. Point clouds require a complex post-processing step (e.g. [Ber+99]) in order to produce the mesh of the scene, which makes them quite tedious to work with.

Geometry can be represented using *meshes*, where corresponding edges and vertices form a graph ([Wan+18]). Although, meshes have advantages (e.g. they are good for forward rendering), the downsides of methods that are using meshes appear in high limitations (e.g. AtlasNet approach [Gro+18] tends to produce self-intersecting meshes or opened surfaces)

2.1.2 Implicit geometry

Implicit methods map points in space to some value, which implicitly gives knowledge about the scene.

The most common example of implicit geometry representation is the signed distance field (SDF) ([CL96; Lom+19]), which is basically a mapping $\mathbb{R}^3 \rightarrow \mathbb{R}$ defining the surface as a level-set (mostly zero-based).

Occupancy fields ([Mes+19]) are introduced as neural networks that directly learn continuous 3D occupancy functions. This allows achieving an arbitrary resolution and more compact and resolution-independent representation instead of learning voxelized representations that result only in fixed resolution.

2.2 Neural scene representation

Neural scene representations exploit the idea of employing deep neural networks in order to implicitly encode information about the scene. The network learns mapping between the spatial locations and feature representations of the scene. Different rendering techniques can be applied to get novel views of the scene but the volumetric rendering technique is the most commonly used with neural scene representations. To make training possible the rendering process has to be formulated in a differentiable way. This allows using loss functions that minimize the differences between generated views and ground truth 2D images.

[Nie+20] introduce a volumetric rendering method, that uses differentiable rendering functions, allowing the implicit neural geometry representations to be optimized with only 2D supervision instead of requiring 3D ground truth models. Using only 2D sample images the surface depth map is predicted and is then unprojected into 3D to be evaluated in the texture prediction field, which outputs color values. The appearance is not explicitly modeled, meaning that the view- and lighting-dependent effects are not taken into account.

The Scene Representations Networks (SRNs) are introduced by [SZW19]. Recurrent neural networks lie at the core of this approach. Camera extrinsics and intrinsics are passed as inputs into the network, which is then evaluated for all the samples along the ray, trying to decide where the surface is located. The network produces the feature vector which is then decoded into a single color value for that point on the surface. This method implies no explicit appearance modeling. The SRNs approach grows from the DeepVoxel method ([Sit+19]), where the feature vectors are integrated into a persistent cartesian 3D-voxel grid.

[Mil+19] propose to expand each sample view into local light field (LLF) using multiplane images (MPIs). The novel view is then rendered by projecting four nearest neighbors of the adjacent local light fields. This work is followed by Neural Radiance Fields (NeRF) [Mil+20]

proposed by same already introduced in this text earlier authors. NeRF extends the LLFs in a different way by leveraging volume rendering. The fully differentiable pipeline allows optimizing parameters of deep neural networks, which predict the volume density and the scene color at a given 3D point. The volume rendering technique is then used for gathering these sample points along the ray and calculating color values for the novel view. NeRF lies in the basis of the proposed in this work methods.

The downside of the NeRFs is the inefficiency of this method. [Liu+21] proposes to use the octree as an underlying structure for the scene, which allows increasing the efficiency of sampling at the cost of very efficient ray voxel intersections. The authors also propose to store learnable feature vectors in voxel corners, which allows decreasing the size of the network while keeping its capacity. The concept of using sparse voxel octrees is used in all of the proposed in this thesis methods.

In FastNeRF [Gar+21] propose factorization of deep radiance maps that can be cached and queried on high-end GPUs. This technique extends the original NeRF network structure by two separate MLPs for position-dependent and direction-dependent inputs. The evaluation time can be decreased by up to 4000 times. The method is not able to relight the scene.

[Rei+21] with the KiloNeRF approach employ thousand of tiny MLPs instead of one large MLP as proposed in NeRF. Each of these MLPs is responsible for its own part of the scene and all of them can be processed simultaneously on modern GPUs. This work reaches more than 2000 times faster in both training and inference stages, does not consider any light interaction though.

[Yu+21] propose PlenOctrees as another structure that can be used to increase the efficiency of the original NeRF method. Authors separate the original network to factorize the appearance using closed-form spherical basis functions - spherical harmonics (SH) [Moh97]. The SH coefficients together with volume density values are then used for building the PlenOctree. Although faster inference time is achieved, novel light conditions are not considered.

In the DeRF approach [Reb+20] the scene is separated into parts by applying spatial decomposition. These parts are then processed with smaller networks, which allows achieving near-constant inference time. The outputs of these networks are later composited using Painter's algorithm [08].

Another way to speed NeRFs up is to rethink the integral estimation using Monte Carlo sampling. AutoInt [LMW21] estimates the rendering integral from NeRF by building grad networks that can estimate any definite integral in two evaluations of the network. With this approach, the training time can be 10 times smaller comparing to the original NeRF.

[Zha+20] elaborate on shape-radiance ambiguity and propose the NeRF++ approach, which uses a separate NeRF to predict background, which increases performance for the unbounded scenes where the dynamic depth range can be very high.

[Lin+21] propose BARF approach that is aimed to solve the joint problem of learning neural 3D representations together with reconstructing unknown camera poses. This technique removes the limitation for the necessity of known camera views.

2.2.1 Relighting

Another branch of works focuses on extending the relighting limitations for NeRF-based approaches. [Mar+21] was one of the first works based on the original NeRF approach. The proposed technique with a more complex model structure that separately renders static and transient elements of the scene and then being composited. The key idea is to optimize the uncertainty image, which is used to weigh the target image. The difference between this target image and the composited outputs is minimized during training. The proposed solution is able to outperform the original NeRF in terms of quality when applied for the datasets with less controlled multi-view collections. This approach allows interpolating color and illumination between two training images, which add relighting capabilities, which are highly limited though.

[Bi+20a] reformulates the rendering equation for considering a single non-static light source on the scene. Although in general, this is highly impractical, one specific case when the light source is co-located with the camera can be evaluated without any sufficient overhead comparing to the NeRF work. This approach implies the appearance to be modeled using any differentiable BRDF model. In the former work [Bi+20b] the authors reconstruct discrete reflectance volumes, which results in its main limitation of fixed volume resolution. The formulation of rendering equation from NRF lies on the basis of all of proposed in this thesis methods except the implicit scheme *ImNRF*.

[Sri+21] with the NeRV approach addresses the relighting problem by extending the original NeRF method with a second MLP that predicts a visibility field. The 'one-bounce' indirect illumination is considered in this approach. This is concurrent with this thesis work, which results are presented in Figure 4.7 for performance comparison.

Neural Reflectance Decomposition (NeRD) [Bos+20] uses two networks: 'coarse' Sampling and 'fine' Decomposition networks to decompose the scene into spatially varying BRDF material properties. This concurrent approach considers uncontrolled illumination and is able to relight the scene under novel lighting conditions. The results of this work are shown in Figure 4.8 for comparison with the results of this work.

In NeRFactor [Zha+21] the surface representation is extracted from the NeRF scene representation. Scene geometry is refined simultaneously with the spatially-varying reflectance and environment lighting. This approach does not have a requirement for known illumination and allows for novel light-view synthesis. Some selective results of this concurrent approach are shown in Figure 4.9 and Figure 4.10

3 Method

In this chapter, I first describe techniques that lie at the core of the proposed solution. They then followed by a detailed explanation of the proposed solution. Finally, I give the implementation details and settings.

3.1 Neural Radiance Fields

The Neural Radiance Fields approach ([Mil+20]) lies directly in the heart of the proposed solutions. The continuous 5D function

$$F : (p, v) \rightarrow (c, \sigma) \tag{3.1}$$

is required to be achieved in order to solve the novel view synthesis problem. Given the spatial location $p \in \mathbb{R}^3$ and the 2D viewing direction (θ, ϕ) (or its equivalent 3D Cartesian representation $v \in \mathbb{R}^3$), one has to obtain the RGB color value c and the volume density σ at this exact point p under the viewing direction v. To obtain the image of the scene, one can take a pin-hole camera at position p_0 and cast rays to the scene: $p(z) = p_0 + z \cdot v$. The final visible color value $C(p, v)$ can then be evaluated using volume rendering approach [Nie+20; Nov+18] along the ray p:

$$C(p, v) = \int_0^\infty \omega(p(z)) \cdot c(p(z), v) dz, \tag{3.2}$$

$\omega(p(z))$ is a probability weight function

$$\omega(p(z)) = \tau_c(p(z))\sigma(p(z)), \tag{3.3}$$

where $\tau_c(p(z)) = e^{\int_0^z \sigma(p(s))ds}$ denotes the accumulated transmittance along the ray up to the point $p(z)$, $\sigma(p(z))$ is the volume density at point $p(z)$ and the probability property is held: $\int_0^\infty \omega(p(z))dz = 1$

[Mil+20] propose to use a Multi-Layer Perceptron (MLP) as a representation of an implicit

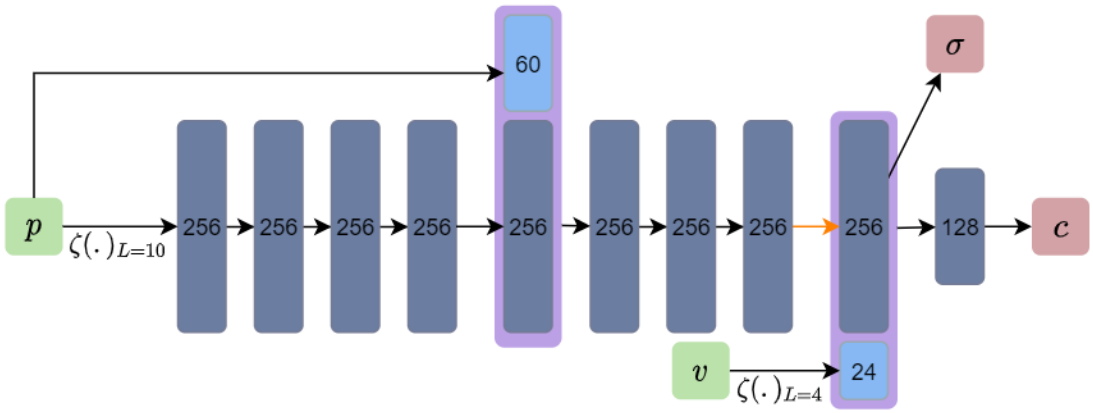


Figure 3.1: MLP used in NeRF [Mil+20] Dark blue boxes represent hidden layers. Black arrows indicate FC-layers with sigmoid activation, orange arrows - FC-layers without activation. Green boxes are inputs: p is a 3D sample point, v is a 3D vector of viewing direction. Red boxes are the outputs: σ is 1D volume density, c is a 3D color value. $\zeta(.)$ stands for positional encoding function that maps inputs into higher dimensional space $\mathbb{R}^{2L}$ (more in Section 3.1.1

function $F = F_\Theta$ with parameters Θ. The scheme structure of the original NeRF's MLP is outlined in Figure 3.1. The position $p(z)$ is first encoded using position encoding function $\zeta(\cdot)$ (positional encoding is described in section 3.1.1) and then is processed with 8 fully-connected neural layers with ReLU activations. The resulting feature vector is then concatenated with positionally encoded vector direction v before the volume density $\sigma(p(z))$ and the color value $c(p(z), v)$ is output. The important concept here is that σ only depends on point $p(z)$ whilst the color value c is view dependant as well. This setting is required to encourage the MLP to be multi-view consistent.

In order to perform the volume rendering, the continuous integral from equation Equation (3.2) has to be solved. One can estimate this integral using a discrete set of densely sampled points as proposed by [Mil+20]. Separating each ray into N bins and drawing random samples from each bin makes the representation able to learn continuous function while only using a finite number of samples. The estimate for the integral $C(p, v)$ will have a form ([Max95; Mil+20]):

$$C(p, v) \approx \sum_{i=1}^{N} \tau_c(p_i)(1 - \exp(-\sigma(p_i)\delta_i))c(p_i, v), \tag{3.4}$$

where $p_i = p(z_i)$ for brevity, $\tau_c(p_i) = \exp(-\sum_{j=0}^{i} \sigma(p_j)\delta_j)$ is the accumulated transmittance and $\delta_i = z_{i+1} - z_i$ is the distance between adjacent sample points.

3.1.1 Positional encoding

Neural networks are known as a highly representative class of functions ([HSW89]). However, recent works ([Rah+19; Tan+20]) demonstrate the tendency of biasing towards low-frequency functions during the training of deep neural networks. The results can be smoothed and blurry as frequency-dependent learning speed is much slower for high-frequency parts of the scene. [Rah+19] show that one can highly improve the quality of the MLP outputs by using an embedding, which maps inputs to a higher dimensional space before passing them to the MLP.

The positional encoding function $\zeta(p) : \mathbb{R} \rightarrow \mathbb{R}^{2L}$ as proposed by [Mil+20; Vas+17] is represented by Fourier series in the following form:

$$\zeta(x) = (\sin(2^0\pi x), \cos(2^0\pi x), ..., \sin(2^{L-1}\pi x), \cos(2^{L-1}\pi x)) \tag{3.5}$$

The 3D Cartesian coordinates of p and v are then separately transformed using $\zeta(\cdot)$ and concatenated.

3.1.2 Hierarchical sampling

The proposed approach is ineffective in a way that it tends to generate a lot of unimportant samples that lie in free space and do not contribute much. To solve that authors propose the hierarchical volume sampling algorithm (similar to [Lev90]) when another "coarse" neural network is used in order to roughly estimate volume densities along the ray. This insight is then used to perform a more informed sampling of the main ("fine") network. The "coarse" network is being optimized along with the main network, which can be considered as an overhead. However, taking into account the increase of training convergence the overall training gets more efficient.

3.1.3 Optimization

The fully differentiable formulation of the problem makes it possible to optimize network parameters Θ without any 3D supervision. This is done using the back-propagation approach from the differences between rendered predictions and ground truth images. This task, therefore, is reduced to the minimization problem of the loss function. The original method by [Mil+20] for loss function uses the following formulation:

$$\mathcal{L} = \sum_{(p_0,v)\in\mathcal{B}} \|C_c(p,v) - C^*(p,v)\|_2^2 + \left\|C_f(p,v) - C^*(p,v)\right\|_2^2, \tag{3.6}$$

where $\mathcal{B}$ is the set of sampled rays in each batch, C^* is the target color value of the view ray v and $C_c(p,v)$, $C_f(p,v)$ - are the predicted colors for the view ray v of the coarse and fine networks respectively.

Another regularization term can be added to this loss function as proposed by [Lom+19]. The beta-distribution regularizer $\Omega(\cdot)$ forces the ray transmittance from the fine network τ_c to be close to 0 or 1:

$$\Omega(\tau_c(p_i)) = \log(\tau_c(p_i)) + \log(1 - \tau_c(p_i)) \tag{3.7}$$

This helps the network to better handle background colors and get cleaner rendering after all. This regularization term is used in NSVF work ([Liu+21]) that is described in Section 3.1.4 and NRF work ([Bi+20a]) that is described in Section 3.1.6

3.1.4 Sparse Voxels Octree

The described approach is already able to represent scenes with high accuracy. However, the very inefficient way of sampling points along rays makes the training very slow and tough, since the model has to be repeatedly evaluated on unimportant samples of free space or occluded regions that do not make a worthwhile contribution to the finally rendered color value. The *hierarchical sampling* strategy alleviates the problem but does not solve it completely. The method in its formulation, therefore, is still barely practical.

[Liu+21] propose another framework that is able to increase the efficiency of this method and make the training time up to an order of magnitude faster. The main contribution is the usage of sparse voxel octrees ([LK11]) to divide space into voxels that are efficiently used for training MLPs with shared parameters.

This approach requires a slight elaboration on the problem formulation. Let the whole scene be fully contained in a set of K sparse voxels: $\mathbb{V} = \{V_k\}_{k=1}^{K}$. The scene can then be represented as a set of voxel-bounded implicit functions

$$F(p, v) = F_\Theta^i(g_i(p), \zeta(v)), \forall p \in V_i, \tag{3.8}$$

where each function F_Θ^i is modeled with an MLP with shared parameters Θ:

$$F_\Theta^i : (g_i(p), \zeta(v)) \rightarrow (c, \sigma), \forall p \in V_i \tag{3.9}$$

Here $g_i(p)$ is the representation of p that is defined as:

$$g_i(p) = \zeta(\chi(\{\tilde{g}_i(p_j^*)\}_{j=1}^{8})), \tag{3.10}$$

where $\{p_j^*\}_{j=1}^{8}, p_j^* \in \mathbb{R}^3$ is the set of eight vertices of voxel V_i, $\tilde{g}_i \in \mathbb{R}^d$ represent the feature vectors that are stored for each vertex of voxel V_i, $\chi(\cdot)$ stands for trilinear interpolation, and $\zeta(\cdot)$ is a post-processing function, which corresponds to positional encoding function explained in 3.1.1.

The proposed network structure is outlined in Figure 3.2. The encoder part that is denoted with

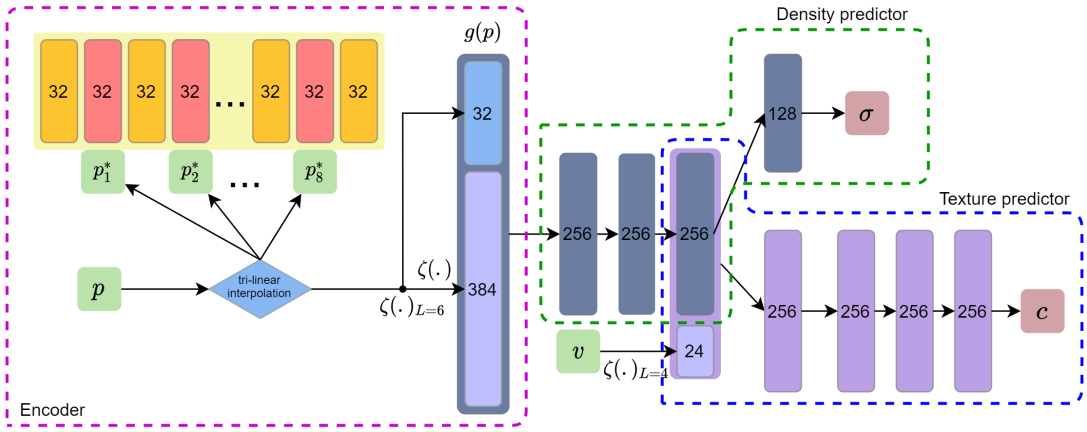


Figure 3.2: Network structure of the NSVF approach (the figure is adapted from [Liu+21]). Same notation as for Figure 3.1 is used. Dotted lines denote semantic parts of the presented structure: pink, green and blue dotted lines stand for encoder, density predictor and texture predictor respectively. *Trilinear interpolation* block denotes interpolation of the sample point p between voxel embeddings.

a pink dotted line consists of only one layer comparing to the original NRF network where the encoder part consists of 5 deep layers. The density predictor (green dotted line) remains the same. The texture predictor (blue dotted line) is deeper comparing to the NeRF model. The overall model structure is slightly shallower comparing to the original NeRF network, which is still sufficient due to the distributed way of storing voxel embeddings in the voxel vertices. Without voxel embeddings, the model contains approximately 0.5M learnable parameters, which is lower than in the original NeRF model.

The volumetric rendering itself does not undergo any significant changes except for some additional steps that have to be performed. The sampling of the ray is only performed at the intervals of intersection with voxels V_i. This allows rejecting a decent amount of unimportant samples. A very efficient Axis Aligned Bounding Box (AABB) ray intersection algorithm ([Hai89]) is applied here to extract the above-mentioned intervals. Another step is *early termination*, which reduces the number of samples that have to be processed. Since solid surfaces do not imply color values being dispersed along the ray, the sample points that are behind the surface would be needless. The technique here is to straightforwardly stop evaluating points, that have an accumulated transmittance lower than some threshold.

3.1.5 Voxels pruning and refinement

The training on the scene begins with the octree, which consists of $N = N_0^3$ voxels that fit in the bounding box of the whole scene and cover all parts of the scene. One can get rid of non-essential voxels by performing a *self-pruning* procedure. This technique leverages the property of neural radiance fields to extract some coarse scene geometry during early training iterations.

The decision if voxel is going to be pruned is made by uniformly sampling G points inside the voxel and comparing the minimum transmittance value with some threshold value γ or more formally when:

$$\min_{j=1\ldots G} \exp(-\sigma(g_i(p_j))) > \gamma, p_j \in V_i, V_i \in \mathbb{V}, \tag{3.11}$$

where $\sigma(g_i(p_j))$ is the model prediction for σ when processing $g_i(p_j)$. The pruning procedure has to be performed regularly as the accuracy of the model increases as it gets more and more trained, which means that more voxels can be pruned and therefore rejected from processing.

As the model learns a more detailed representation of the scene it becomes necessary to refine the voxel structure. During the refinement procedure, each voxel is being subdivided into 2^3 subvoxels by halving the voxel size. Ray sampling steps are also being halved at this point in order to correspond new voxel size value. The feature representation vectors $\tilde{g}_i$ of the new vertices are initialized similarly to calculating $G_i(p)$, namely using trilinear interpolation function $\chi(\cdot)$. This scheme allows to achieve more detailed scene representation and also increases the model capacity.

3.1.6 Neural Reflectance Fields

Although the neural radiance fields approach is able to produce realistic renderings under novel view conditions, one of the main limitations of this method is the inability to handle any kind of visual effects related to dynamic light conditions. The core assumption of these approaches is static illumination, which is implicitly presented in the color values of the rendered images. This in turn does not allow to change illumination on the rendering phase.

[Bi+20a] propose the framework for eliminating this exact limitation. Authors introduce the neural reflectance field (NRF), which is the neural scene representation that in addition to scene geometry also encodes normal and reflectance properties of the scene. This approach allows the evaluation of an explicit reflectance model at the selected point after all. Although [Bi+20a] focus on the microfacet bidirectional reflectance distribution function (BRDF) ([Wal+07]), any other reflectance model can be used within this method as well (i.e. authors also show some results of furry objects using hair reflectance model [KK89]).

This method uses the same rendering framework as proposed by [Mil+20]. Assuming that the volume containing the scene to be only scattering volume (i.e. there are no emission or absorption processes), the corresponding rendering equation Equation (3.2) ([Nov+18]) looks similarly except for operating with radiance values L instead of color values c:

$$L(p, v) = \int_0^\infty \omega(p(z)) \cdot L_s(p(z), v) dz = \int_0^\infty \tau_c(p(z)) \sigma(p(z)) \cdot L_s(p(z), v) dz, \tag{3.12}$$

where $L_s(p(z), v)$ denotes the scattered light at point $p(z)$ along direction v.

In general case L_s is calculated by integrating incident radiance along the whole unit sphere $\mathbb{S}$:

$$L_s(p, v) = \int_{\mathbb{S}} f_p(p, v, l) L_i(p, l) dl, \tag{3.13}$$

where l is the 3D Cartesian representation of directions of the incident radiance $L_i(p, l)$. $f_p(p, v, l)$ is the phase function, which controls how the light that comes from l scatters at point p if viewing from v.

[Bi+20a] only assume single light source setting and the reflectance function as the phase function, which allows simplifying Equation (3.13):

$$L_s(p, v) = f_r(p, v, l, n(p), R(p)) L_i(p, l), \tag{3.14}$$

where f_r denotes a differentiable reflectance function with parameters R. Normal vector n is also required for evaluation of reflectance model f_r. $L_i(p, l)$ is the radiance that arrives from direction l and can be calculated using light intensity $L_l(p)$ and the transmittance value $\tau_l(p, l)$, which indicates the loss of light from the light source to the sample point p along light ray l, as follows: $L_i(p, l) = \tau_l(p, l) L_l(p, l)$. The light attenuation coefficient $f_{att} = \frac{1}{k_c + k_l d + k_q d^2}$ [Mad11a; Mad11b] is considered inside the L_l term: $L_l(p, l) = f_{att} \cdot I$, where k_c, k_l and k_q are attenuation factors, I is the point light intensity. In my experiments the attenuation coefficients $k_c = k_l = 0$ and $k_q = 1$ were used, so the attenuation coefficient is inverse proportional to squared distance between light sample point and light source: $f_{att} = \frac{1}{d^2}$

Combining these equations and applying the same technique for numerical integral estimation as used by [Mil+20] and [Liu+21] gives the following formulation of Equation (3.4) for NRF:

$$L(p, v, l) \approx \sum_{i=1}^{N} \tau_l(p_i, l) L_l(p_i, l) f_r(p_i, v, l, n(p_i), R(p_i)) \tau_c(p_i) (1 - \exp(-\sigma(p_i)\delta_i)) \tag{3.15}$$

Here τ_l plays the same role for light ray l as τ_c operates for view ray v. It is evaluated similarly along the light ray l: $\tau_l(p'_i, l) = \exp(-\sum_{j=1}^{M} \sigma(p'_j)\delta'_j)$, where $p'(z') = p'_0 + z' \cdot l$ are sample points that are additionally sampled along the light ray l and $\delta'_j = z'_{j=1} - z'_j$.

The architecture of the MLP is outlined in Figure 3.3. The deeper structure is used as the model capacity has to be increased due to the higher complexity of the task. For each view sample point p the model outputs both normal vector n and parameters R, which are required by the selected BRDF model. For the microfacet BRDF ([Wal+07]) R consists of a 3-dimensional albedo value and 1-dimensional roughness value.

This formulation covers the general case when the light source can be located at an arbitrary position ('arbitrary setting'). Since the l and v vectors are different, the additional sampling of light ray for each view sample point is required. It makes this problem impractical. However,

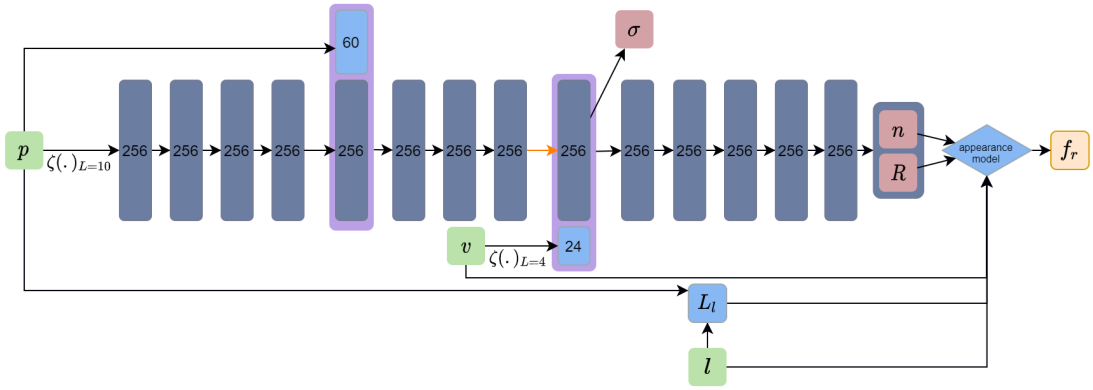


Figure 3.3: MLP structure of the NRF [Bi+20a] approach. The overall network structure echoes the original NeRF model. 14 deep layers are used (instead of 10 in NeRF). The appearance model block represents BRDF f_r that is evaluated for each sample point p. To this end network outputs normal $n \in \mathbb{R}^3$ and BRDF parameters $R \in \mathbb{R}^m$ ($m = 4$ for microfacet BRDF). L_l denotes light intensity with consideration to distance attenuation.

[Bi+20a] show that one specific light setting exists, which can be used to handle this problem. When the light ray l coincide with the view ray v, then sampling points $p(z)$ and $p'(z)$ would be the same, which in turn means that $\tau_c(p_i) = \tau_l(p_i)$ ('co-located setting'). In this case, there is no need to additionally sample light rays, which leave the processing time requirements on the same level as the original work by [Mil+20]. Another advantage of this setting is that it has an accessible real-world analogy. In particular, all modern cellphones are equipped with cameras and flashlights that are placed close to each other. On the scale of the scene, this displacement can be neglected and the setting can be considered to be co-located.

The co-located setting results in a very sparse sampling of the scene in terms of light-view directions. During training, there are no other samples where the angle between light and view rays is greater than zero. This fact negatively affects the overall training performance, although [Bi+20a] claim that even with these restrictions the network is still able to generalize and reconstruct the scene appearance in high quality. The authors also show some renders where the 'arbitrary setting' had been used during the prediction phase.

3.2 Explicit Neural Reflectance Field

The described above methods have already achieved some appealing results. However, all of them have their own limitations that in either event reduce the performance either qualitatively (no light interaction) or quantitatively (too slow). The explained below solution is constructed to take advantage of both NRF and NSVF approaches. This method is later referred to as the 'brute-force scheme' as it performs the light rays sampling without any approximation techniques.

Since the NRF approach [Bi+20a] is based on NeRF, it inherits its poor performance behavior and forces it to be only applicable for the 'co-located setting'. Although the authors are even able to render the scene within the 'arbitrary setting' when only being trained on the co-located dataset, the sparsity of the training dataset still highly affects the network's generalization opportunities.

[Liu+21] in turn propose the technique for increasing the effectiveness of the training for the color-based original NeRF approach. The usage of gradually refined octrees gives more control over the sampling process and makes the whole training phase faster.

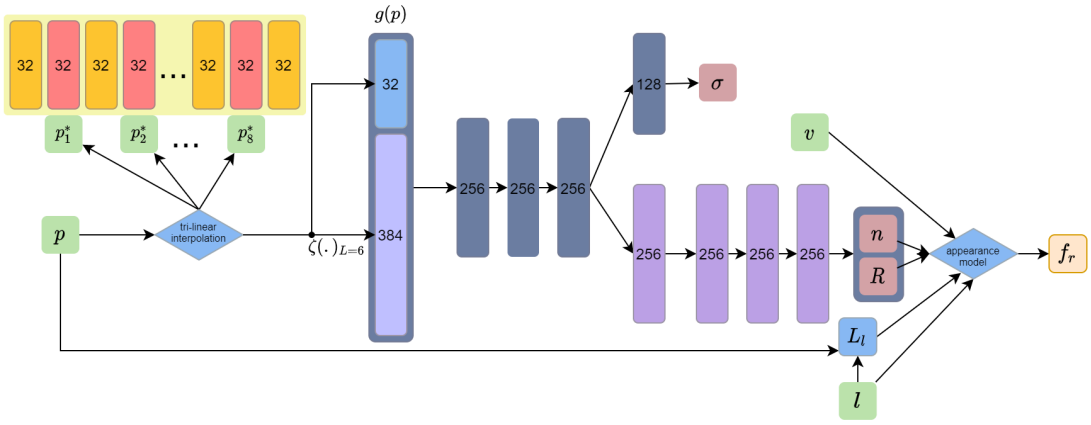


Figure 3.4: Model architecture for the 'explicit scheme' (Section 3.2). The overall structure follows one from NSVF work. The outputs of the network are similar to the NRF model. BRDF evaluation ('appearance model' block) is required for achieving f_r value.

Since both works NSVF and NRF use the original NeRF framework, in particular similar numerical integral estimation, the process of reformulating NSVF problem to work with light radiances instead of color values is fairly straightforward. For the integral estimation Equation (3.4) is directly replaced with Equation (3.15). The network architecture follows the guidelines from the NSVF work, namely the encoder is shallow while the texture predictor is deeper comparing to the original NeRF method. Appearance model block (right blue rhombus) includes BRDF evaluation, which results in f_r value required for final radiance calculation (using integral estimation formula equation (3.15))

In order to train the model on arbitrary light dataset an additional step of light rays sampling and volume density prediction is needed. This requires some nested model evaluations in order to retrieve σ values at light sample points. This is quite a memory intense part, however, the lower model size comparing to the NRF approach makes it more accessible. Another keynote here is that there is no need in getting texture field predictions, as for light rays the only valuable information is the volume density. This allows to only evaluate the corresponding part of the model.

Another key point here is the format of the dataset. The NeRF and NSVF methods only work with color values, which allows them to use the LDR images in the datasets (i.e. PNG, JPG

image formats and others). However, the NRF approach implies light intensity values that are integrated into the final radiance calculations. The LDR images are usually lacking the information of radiances and only operate with color values. All these facts make the LDR images bad candidates for the training dataset. One should instead consider using HDR images (i.e. HDR, EXR formats).

The point light intensity value I that takes part in the calculation of $L_l(p, l) = f_{att} \cdot I$ represents the radiation power of the light source. This value is known when processing artificially generated scenes and thus can directly be used as the value of I. The real-world scenes (e.g. captured with the cellphone and flashlight) on the contrary do not have the exact light intensity value. One could estimate the power of the flashlight, but this would give only a rough estimation. To make the method robust to this kind of estimations, the light intensity value I is passed to the optimizer (along with other model parameters Θ). This technique requires some robust estimation of I as an initialization value as well as an increased learning rate.

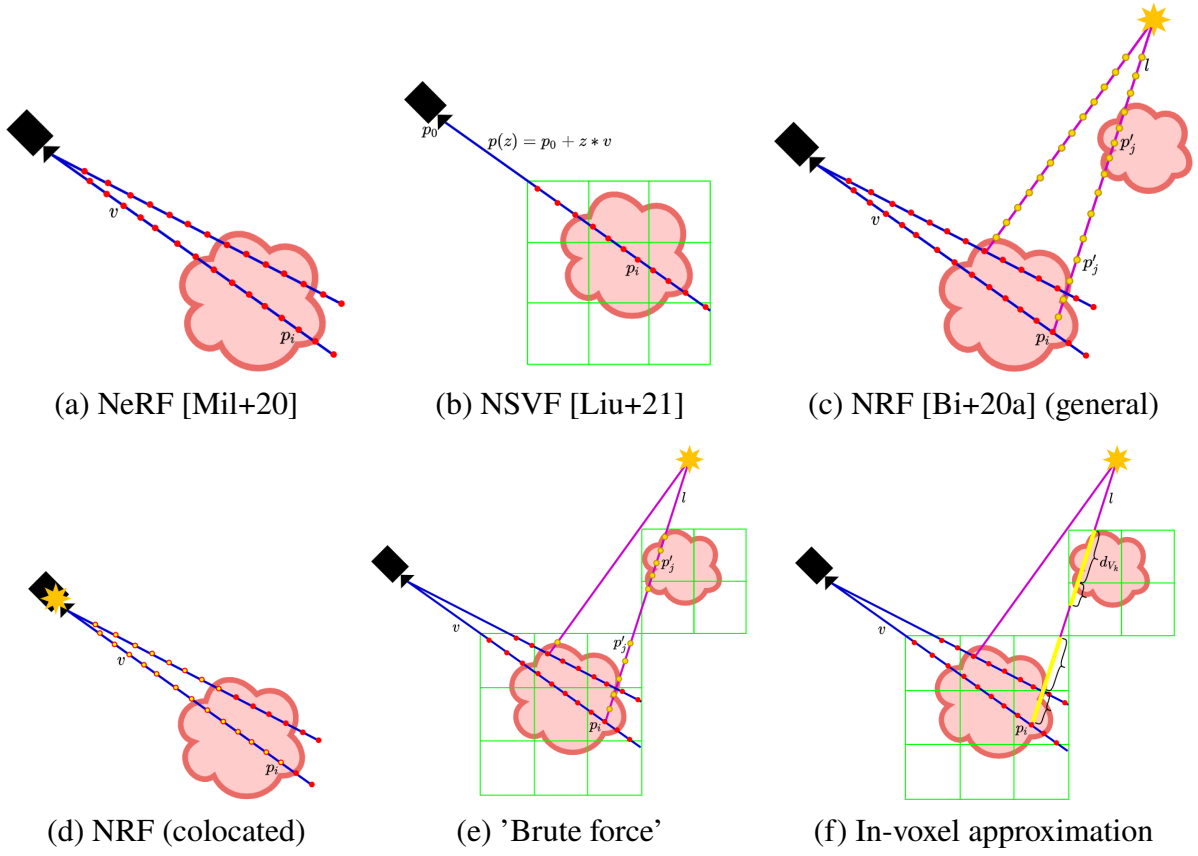
3.3 In-Voxel Light Approximation

Using sparse octree voxels for increasing the efficiency of the method makes the whole problem feasible within some commonly accessible hardware. However, the training procedure still remained highly computationally expensive and memory demanding and therefore very slow in training or still even intractable for high-resolution scenes.

The main load of this approach still remains in the additional sampling of light rays and the consecutive model evaluation for calculating $\tau_l(p', l)$. This fact leads to the task of minimizing the overhead that is caused by calculating light ray transmittances.

Predictions of volume densities at view ray sample points have primary importance as they contribute to the final radiance value the most. This assumption brings to the fact that light rays are not required to be sampled as densely as view rays to remain the same level of accuracy. Thus the lower number of points can be sampled on the light rays, which would result in lower memory and time consumption. However, this approach only alleviates the problem while not solving its root cause.

The octree structure that is used to encode feature vectors $\tilde{g}_i$ in the corners of voxels represents itself a coarse geometry of the scene. Starting with only a few voxels that cover the whole bounding box around the scene, the voxels undergo the *self-pruning* and *refinement* procedure (explained in Section 3.1.5), which increases the detailing capabilities of the octree by increasing the number of voxels, lowering its sizes and rejecting empty voxels. After only a few steps of the aforementioned operations, the overall voxel structure repeats scene geometry with a considerable amount of accuracy. These voxels that intersect with the light rays can be used as an estimation of volume densities.



(a) NeRF [Mil+20] (b) NSVF [Liu+21] (c) NRF [Bi+20a] (general)

(d) NRF (colocated) (e) 'Brute force' (f) In-voxel approximation

Figure 3.5: Sampling techniques used in different approaches. The 2D sketch schemes are used instead of 3D visualizations for simplicity. Blue lines represent view rays $p(z)$ with direction vector v, purple lines stand for light rays $p'(z)$ with light direction vector l (only two of those are shown for brevity). Red dots represent sampled view points p_i, yellow dots stand for light sample points p'_i. Scheme (a) shows the original NeRF sampling technique without *hierarchy sampling* (Section 3.1.2). Scheme (b) correspond to the NSVF sampling technique that only occurs inside intersected voxels (represented by green boxes). Scheme (c) shows sampling of the general case of the NRF approach when light source is placed at arbitrary position. Scheme (d) displays the 'colocated setting' when point light source is positioned at the same location as camera. In this case light sample points (yellow dots) coincide with the view sample points (red dots). Scheme (e) correspond to the proposed in this work 'brute-force' approach when light rays sampling is reduced by using the same voxel structure as in NSVF (fig. (b)). Scheme (f) shows sampling proposed in this work in-voxel approximation scheme. View rays are sampled equally as in NSVF approach while light ray transmittances are estimated using travelled distances d_{V_i} of the light ray inside voxels.

There are plenty of techniques that can be applied here in order to calculate light transmittances more efficiently. The main proposed technique implies the assumption that the volume density is homogeneously distributed inside the voxel. This assumption is certainly wrong for the general case but should give a fair estimation for the light ray transmittances. In this case, the traveled distance d_{V_k} of the ray inside voxel V_k can be used to weigh some value $\hat{\sigma_{V_k}}$ (Figure 3.5 f):

$$\tau_l(p_i, l) = \exp \sum_{V_k \in \mathbb{V}^*} -\hat{\sigma}_{V_k} \cdot d_{V_k}, \tag{3.16}$$

where $\tau_l(p_i, l)$ is the light transmittance value at the view sample point p_i and $\mathbb{V}^* \subset \mathbb{V}$ is the set of voxels V_k that have been intersected by the light ray $p'(z') = p'_0 + z' \cdot l$.

The model can be periodically evaluated to predict the values of $\hat{\sigma}_{V_k}$ and consequently calculate τ_l. The period of evaluation can be similar to periods for *self-pruning* and *refinement* procedures as some coarse value would already give an acceptable level of certainty.

Some other options can be applied here such as:

1. calculate $\hat{\sigma}_{V_k}$ using points sampled in centers of voxels $V_k \in \mathbb{V}^*$
2. calculate mean $\hat{\sigma}_{V_k}$ value from points of intersection of light ray with voxels $V_k \in \mathbb{V}^*$ (requires handling of edge cases)
3. assign $\hat{\sigma}_{V_k} = \text{const}, \forall V_k \in \mathbb{V}^*$
4. use $\hat{\sigma}_{V_k}$ that was calculated during pruning stage

3.4 Implicit Neural Reflectance Field

The aforementioned methods that are proposed within this work extend existing approaches in terms of efficiency for the problem of reproducing the scene under novel view and light conditions. The novel light conditions are the essential requirement here as it expects the method to not only account for color values but also care about the reflectance of the surface. Although the NRF approach is formulated in general form when any differentiable appearance model can be used for final radiance evaluation, the result is still limited by the chosen BRDF.

To avoid these limitations one can shift the appearance evaluation burden to the deep neural network, which is able to get trained for very complex appearance representations. This 'implicit scheme' would have a simpler neural network structure, which is shown in Figure 3.6. The light interaction is performed here implicitly by providing the network with the light direction l as well as the distance from the sample point to the light source $d_i = \left\| p_k^l - p_i \right\|$ as denoted in the figure. Light ray direction l is also positionally encoded with the same L as view ray direction

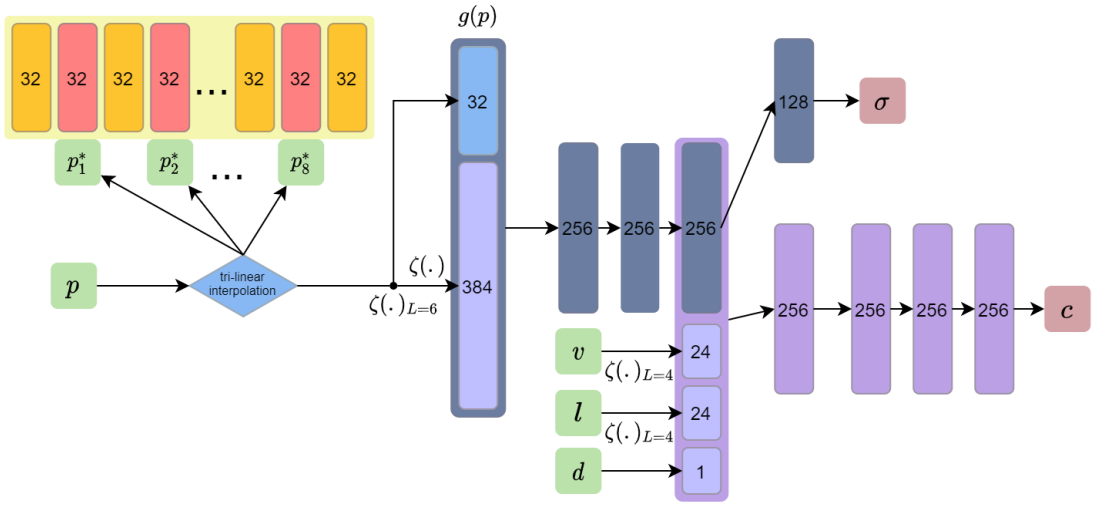


Figure 3.6: Network architecture for the implicit scheme. The structure of the network mostly remains the same as in original NSVF work. Some additional parameters such as light ray direction l and distance d from view sample point p to the point light are provided in advance into texture predictor.

v is. Although the shadowing is not explicitly modeled, the network is expected to learn this effect by itself.

The main downside of this scheme comes from its advantage. Implicit representation significantly decreases the amount of control over the learned field. In other words, bigger amount of parameters that are responsible for different effects on the scene are hidden in the weights of the model and cannot be easily attained.

Another characteristic of this scheme is that it does not imply any kind of regularization. This means that the network is not guaranteed to be able to extract the appearance of the scene. This behavior can be clearly seen from the experiments of the *ImNRF* on the colocated light dataset Section 4.4.1.

4 Evaluation

This chapter addresses the implementation details, datasets, metrics and experiments with the demonstration of the achieved results.

4.1 Training and validation data

Each training session consists of optimizing parameters of a separate neural representation from one of the proposed methods. The dataset for each training session is created on one specific scene and with one specific light setting. All of the used datasets have been created in an artificial environment using Blender [Com18]. Each dataset consists of rendered RGB images of the scene. Each image is an HDR image (the OpenEXR [KBH04] format is used to store HDR data on disk). The dataset samples are the aforementioned images that correspond to a view from a pin-hole camera with intrinsic parameters $\mathcal{I}$ (same for the whole dataset). The camera is placed on a sphere around the scene, up-vector is always positive in Z axis, the orientation is always aimed to the center of the scene. The position of the camera is described with two angles: azimuthal angle θ and inclination angle ϕ. For each view, the values of (θ, ϕ) are sampled uniformly on the surface of the sphere. In the experiments, the inclination angle is bounded between 10 and 100 degrees. Camera extrinsics (in form of a transformation matrix) are exported for each dataset sample along with the HDR image. Additionally to that, the bounding box is also exported as it is required to initialize the octree for the NSVF method and all of the proposed methods as well. 200 sample images are used in datasets with 'static' and 'colocated' light settings. For the 'arbitrary' light setting each dataset consists of 500 sample images for having a more dense light-view sampling distribution. These sample images are then split into training (75%) and validation (25%) sets.

4.1.1 Light settings

Different light settings are required due to a diversity of tasks that different methods are to solve. The illumination of the scene is set using a point light source. The intensity of the light source varies for different scenes and is chosen empirically. The positioning of the light source is

performed according to the chosen light setting. Three different light settings are used in order to create three versions of each dataset (Figure 4.1). This is done based on the fact that different methods can only handle datasets with different settings:

1. **Static setting**

   In this case the light source is positioned at the same global location for all of the dataset samples. This allows producing renders of the scene with static illumination. The motivation for this type of datasets comes from the limitation of the NSVF method as it does not imply dynamic illumination. Datasets with static setting are made for 'Vanilla NSVF' ([Liu+21]) and can also be processed with 'Brute-force' (Section 3.2) and both explicit and implicit schemes (Figure 3.6) with in-voxel approximation (Section 3.3).

2. **Colocated setting**

   This setting implies the light source to be co-located with the camera. In this case, there is no displacement between the camera and the light source, which allows the NRF ([Bi+20a]) approach to reuse the view volume transmittances in calculations as transmittances of the light rays. The 'Brute-force' scheme as well as both explicit and implicit schemes with in-voxel approximation are also able to be trained on this type of datasets.

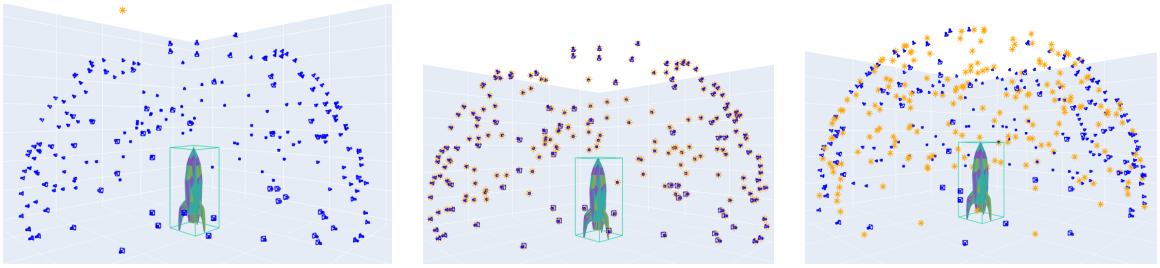
3. **Arbitrary setting**

   Datasets that have been created with this setting consist of renders where both camera and light positions are sampled uniformly on a sphere around the scene. The positioning of the light source follows the same scheme as for camera position. Another constraint is applied here that forces the angle between view and light rays to the center of the scene to be smaller than some value α_{max}. In the experiments, the value $\alpha_{max} = 120$ degrees has been used.

   This setting is focused on the general case of the NRF approach and can be handled by proposed methods, such as: 'Brute force' scheme and both explicit and implicit schemes with in-voxel approximation.

4.1.2 Scenes

Creating the datasets from scratch is motivated by the fact that the vast majority of related works do not consider any light interaction. This results in such a problem that the datasets that have been used in these works do not include any necessary light information (global position). Another limitation of these datasets is that the light source is static upon the dataset samples, which results in a sparse light-view sampling of the scene. To the date of writing this thesis, the datasets from corresponding related works [Sri+21] still are not publicly available. Four different scenes have been used for the evaluation (Figure 4.2).



(a) Static (b) Colocated (c) Arbitrary

Figure 4.1: Types of datasets that are used in the experiments. Cameras are denoted with blue icons, point light sources are represented by yellow stars. In static light setting (a) light source in all of created sample images are placed at the same point while camera locations are spread on a clipped sphere around the scene. Colocated light setting (b) imply point light sources to be placed at the same position with the camera for each sample view. In arbitrary light setting (c) both point light source and camera locations are spread on a clipped spheres around the scene.

(a) Rocket (b) Guitar (c) Lego (d) Hotdog

Figure 4.2: Preview samples of the created datasets. All the datasets are created from the very beginning. (a) [Bou19] and (b) [Leg20] are created from publicly available 3D models for Blender [Com18]. (c) and (d) are created from 3D models that are publicly available withing NeRF work [Mil+19].

Rocket

The *rocket* scene (Figure 4.2 (a)) contains a single object, which exhibits geometry of medium complexity and realistic non-Lambertian materials. This scene contains global geometry complications (rocket wings) and some fine details (rivets on the rocket shell). The surface of the rocket shows a moderate level of specularities (in comparison with sphere scene)

Guitar

The *guitar* scene (Figure 4.2 (b)) is created for testing the model's ability to reproduce complex geometrical features on different scales. For example, strings can be barely seen on renders and

they would presumably be the most troublesome geometrical part of the whole scene. A fairly specular surface appearance also allows assessing how well does it model surface reflectance behavior.

Lego

Lego scene (Figure 4.2 (c)) is the reproduction of the model that has been used in a former NeRF work [Mil+20]. The dataset is recreated under a specific illumination, which consists of only one point light source. The object exhibits a very complex geometry on all scales. The appearance of the surface does not show any complications.

Hotdog

Hotdog scene (Figure 4.2 (d)) is also a reproduction of the model from the NeRF work with only one point light source. The reflectance behavior of objects on this scene is more sophisticated comparing to the *lego* scene while the geometry is simpler.

4.1.3 Real-world dataset handling

For the group of methods that only consider static illumination of the scene, the dataset acquisition can be performed in different ways. One of these ways is to use a cell phone with its camera to capture the scene. The acquired images can then be processed with such techniques as structure-from-motion ([JP11; MMM12; SF16]) in order to retrieve the camera intrinsics and extrinsics in correspondence with captured images. Another way is to use some special hardware setups with calibrated camera positions such as gonioreflectometers.

However, for the methods that model light interaction the light extrinsics information is lacking when using the aforementioned techniques. The acquisition of the dataset consisting of captures from real-world scenes for this group of methods is a fairly hard task that most probably cannot be performed without any hardware assistance. Even for the 'colocated' case of the NRF method cellphones could only have been used together with the robotic arm holding the cellphone ([Bi+20a]). Some hardware setups such as X-Rite tac7 [MWK17] are equipped with static cameras and light sources and a turntable, which allows to rotate the object and thereby acquire varying viewing and illumination directions.

Another key point is the background that has to be as homogeneous as possible. The quality of final renders is sensitive to the too complex background behind the object on the scene.

4.2 Implementation details

All of the proposed methods are implemented on Python using PyTorch [Pas+19] framework for neural fields and ray casting procedure. The Fairseq [Ott+19] framework is used over the

Pytorch, which provides easier training, evaluation and validation pipelines as well as extends the initial functionality. During training 80 pixels are first sampled from each of 1 or 2 images forming a batch that is used for one training iteration. These pixels are used to cast rays through the focal point of the corresponding cameras. These rays then intersect the octree, which is initialized with $N_0^3 = 4^3$ voxels (4 voxels for each axis). These rays are then sampled on the voxel intersection intervals with the step size 0.125. The following procedure differs depending on the method and includes additional light ray casting, intersecting and sampling for the 'brute-force' scheme and BRDF evaluation for the 'explicit schemes'. Alpha compositing is performed after the color (or radiance) values are achieved. These values are then being supervised with the ground truth values using the fine-network term from loss function from the Equation (3.6) with the beta-distribution regularizer from the Equation (3.7):

$$\mathcal{L} = \sum_{(p_0,v)\in\mathcal{B}} \|C(p,v) - C^*(p,v)\| + \beta(\log(\tau_c(p,v)) + \log(1-\tau_c(p,v))), \tag{4.1}$$

with $\beta = 10^{-3}$ that is used in the experiments.

Adam optimizer is used as an optimization algorithm for learning parameters (Θ, I) (I is light intensity value). The learning rate for the optimizer is initialized with value 10^{-4} and linearly decays to value 10^{-6} over 100k iterations. Octree refinement procedure as well as reducing ray casting step size are performed on iterations 5k, 25k and 50k. Self-pruning is performed every 5k iterations. Tone-mapping using scaling and gamma-correction (differ depending on the scene) is performed to visualize HDR images (both inputs and outputs).

All of the experiments are held on NVIDIA GTX 1080Ti GPU with 11GB of memory. Dataset sample images are originally created in 1024x1024px resolution and then individually down-sampled to some lower values for each experiment to fit the GPU memory limitations. The resolutions from 64x64px (for *brute-force scheme*) up to 512x512px (for methods with lower memory load, such as Vanilla NSVF) are used. The *log-transform* is applied at the pre-processing phase to prepare the HDR data before it is processed by the model:

$$c = \log(c^{HDR} + 1), \tag{4.2}$$

where c^{HDR} is the color value of the HDR input and c is the color value that is used for model supervision. The inverse-log-transform is performed on the model outputs at the post-processing stage.

4.3 Metrics

For evaluation of the results the following metrics have been used:

1. *PSNR*↑ [HZ10] (Peak Signal to Noise Ratio) is a traditional estimator for image comparison that is based on MSE and concentrates on the pixel-by-pixel comparison. Higher value is better.

2. *SSIM*↑ [HZ10; NA20; Wan+04] (Structural Similarity Index Measure) is a perceptual metric that assesses image quality based on perception of the human visual system. Higher value is better.

3. *LPIPS*↓ [Zha+18] (Perceptual Similarity) is the metric that utilizes deep features of the pre-trained deep neural networks (Alex-net [KSH12] in my experiments) to provide an evaluation that agrees well with humans' assessments. Lower value is better.

4. *HDRFlipLoss*↓ [And+20; And+21] is the metric that is aimed to work with HDR images and produce better assessment comparing to other metrics that were originally designed to work with LDR images. Lower value is better.

Additionally to the above-mentioned metrics, empirical evaluations have to be made to assess the quality of the reproduced appearance. This can be done by evaluating the model on a given trajectory for camera and light sources. This produces continuous changes in appearance and reveals all of the artifacts of the produced renders. The usage of artificial datasets (Section 4.1) allows generating the ground truth images even for novel light and view conditions, which are not contained in training or validation datasets. This cannot be done for real-world images, which makes the whole task of quality assessment more difficult for such datasets.

4.4 Experiments

Following list overviews schemes that are compared in this section:

1. Vanilla *NSVF* method [Liu+21] (described in Section 3.1.4). The fastest implementation (among other listed methods), however, can only be processed with the static light dataset.

2. **Ex**plicit **Col**ocated scheme (*ExCol*) [Bi+20a; Liu+21], which is explained in details in Section 3.1.6. This is a 'lighter' version of the *explicit brute-force scheme* due to the reuse of viewing sample points. The second fastest implementation as it only contains a small overhead of evaluating BRDF instead of directly predicting colors. However, similarly to *NSVF* can only be trained on one type of dataset: with the colocated light setting.

3. **Ex**plicit **B**rute-**F**orce scheme (*ExBF*) (described in Section 3.2). The generalization of *ExCol*. The slowest implementation with the highest memory demand, highly impractical especially with the aforementioned hardware setup.

4. **Ex**plicit **V**oxel-**A**pproximated scheme (*ExVA*), which is the *explicit brute-force scheme* that uses the in-voxel approximation from Section 3.3. Implementation involves similar complexity as *ExCol* as it also includes light rays octree intersection overhead, which makes it slower than *ExCol*.

5. **Im**plicit **N**eural **R**eflectance **F**ield scheme (*ImNRF*) (described in Section 3.4). This method is based on the Vanilla NSVF approach. Light direction vectors and distances to the light source are also provided along with view direction and sample point location. This scheme is considered to implicitly represent the appearance of the scene and can handle all of the listed dataset settings.

	NSVF [Liu+21]	ExCol (ours)	ExBF (ours)	ExVA (ours)	ImNRF (ours)
Static light	✓	X	✓	✓	✓
Colocated light	X	✓	✓	✓	✓
Arbitrary light	X	X	✓	✓	✓
Novel views	✓	✓	✓	✓	✓
Novel lights	X	✓	✓	✓	✓*

Table 4.1: The applicability of different types of datasets to methods. As can be seen from the experiments section (Section 4.4.1), *ImNRF's* ability for view synthesis under novel light conditions is highly sensitive to the type of training dataset.

Please refer to Table 4.1 for the visual representation of the applicability of different types of datasets to the methods mentioned above methods.

4.4.1 Colocated light setting

Datasets with the colocated light setting can be evaluated by all of the listed models except *NSVF*. The fact that a very fast *ExCol* scheme can be evaluated on these datasets makes them very advantageous for comparing against other methods. Since the *ExCol* scheme is essentially a special case for the *ExBF* scheme they perform the same way as long as the sampling for light rays is performed the same way as for view rays. On the contrary, the performance of the *ExBF* scheme is drastically worse comparing to the *ExCol* scheme. The *ImNRF* scheme is expected to perform at the same efficiency level as the Vanilla *NSVF* method. This is due to the fact, that light rays coincide with view rays and the scheme does not consider any kind of regularization for the model to distinguish light rays from view rays. The light distance is only applicable for light attenuation, which is not expected to make a very noticeable effect. Hence the main focus here is to compare colocated scheme *ExCol* with the in-voxel approximation scheme *ExVA*.

An overview of quantitative results of evaluation methods on colocated datasets is presented in Table 4.2. Four aforementioned metrics are calculated on renders over the validation dataset

Dataset	Method	PSNR↑	SSIM↑	LPIPS↓	HDRFlip↓	Iters.	Res.	Time
Rocket	ExCol	27.69	0.94	0.088	0.053	50k	256px	2h30m
	ExVA	29.14	0.94	0.083	0.058	50k	256px	3h
	ImNRF	**29.87**	**0.95**	**0.061**	**0.049**	50k	256px	2h
	ExBF	26.80	0.94	0.030	0.090	50k	64px	6h30m
Guitar	ExCol	29.40	0.96	0.042	0.057	150k	256px	10h30m
	ExVA	29.22	0.96	0.043	0.059	150k	256px	20h
	ImNRF	**32.56**	**0.98**	**0.017**	**0.042**	150k	256px	20h'
	ExBF	25.28	0.93	0.030	0.120	30k	64px	8h
Lego	ExCol	25.57	0.91	0.090	0.110	50k	256px	3h
	ExVA	25.95	0.92	0.090	0.110	50k	256px	4h30m
	ImNRF	**26.99**	**0.93**	**0.066**	**0.099**	50k	256px	2h30m
	ExBF	20.76	0.85	0.053	0.179	50k	64px	6h
Hotdog	ExCol	32.80	0.96	0.040	0.075	70k	256px	6h
	ExVA	32.57	0.96	0.040	**0.073**	70k	256px	25h*
	ImNRF	**33.90**	**0.97**	**0.026**	0.089	70k	256px	11h'
	ExBF	24.68	0.88	0.059	0.150	70k	64px	16h

Table 4.2: Quantitative comparison of evaluations on colocated datasets (25% of the whole dataset or 50 sample views). *ExBF* scheme is only trained on 64px images due to hardware limitations while other methods proceed with 256px images (except *ImNRF* on rocket dataset, which metrics are given for reference only). Time consumption involves training time as well as regular validation time (performed every 1k iterations). Reported time' for *ImNRF* method is generally non-demonstrative due to the unreasonable usage of too low chunking rates, which led to unrealized potential. *ExCol* and *ExVA* schemes generally produce very close results, although *ExVA* is an approximation of the general case of *ExCol* method. The overhead of additional calculations of *ExVA* results in more time consumption. *ImNRF* stands out from other methods by achieving better results. Time measurement for *ExVA* method on Hotdog dataset is not plausible due to adjacent circumstances limiting the hardware performance for this experiment.

(25% of the whole dataset, 50 images for these experiments). *ImNRF* is the fastest scheme since it almost completely shares complexity from Vanilla NSVF. The *ExCol* scheme is the second fastest since it does not evaluate any overhead related to sampling light rays as the view samples are simply reused as samples for light rays. The overhead of the *ExCol* method comparing with *ImNRF* comes from the additional evaluation of the BRDF model before alpha compositing of view samples. The *ExVA* scheme is an approximation of the *ExCol* method that uses the distance of travel of light rays inside the voxels scaled by some constant sigma value. The implementation implies light rays intersection with the octree, which is a fairly fast task, although it still substantially drops the performance. The approximation scheme *ExVA* is from 1.4 to 2 times slower than the fastest *ExCol* depending on the scene.

The *ExCol* scheme is similar to *ExBF* results, which concurs with the expectations. However, the

ExBF method has a massive overhead in sampling light rays and evaluating the model on these samples, which makes it naturally impractical. Training of *ExBF* model even on 64px images takes 1.5 times more time than training approximation *ExVA* scheme on 256px images. In these experiments, the *ExBF* scheme did not manage to converge completely due to hardware memory limitations. The *ImNRF* scheme shows the best performance on this dataset. This is presumably due to the fact that it is not forced to perform any kind of light interaction generalization. As can be seen later (Figure 4.5) the model mostly does not change predictions for non-coincident view and light rays. For these experiments, the *ImNRF* is believed to perform in a similar way Vanilla *NSVF* would have performed if it used consideration of light interaction.

Some selected views from these experiments are presented for comparison in Figure 4.3. As was already said the *ExBF* did not show high-quality results. The main limitation is that this scheme is very memory-demanding and fairly slow. Images in 64px resolution have been used for the *ExBF* scheme. The synthesized views are also in the same resolution and scaled up for clearer comparison. For Lego and Hotdog datasets *ExBF* failed some parts of scene geometry (the corresponding voxels have been removed during the pruning stage). When comparing *ExCol*, *ExVA* and *ImNRF* schemes, the results are very similar especially for *ExCol* and *ExVA* methods. The *ImNRF* scheme generally showed the best results (fine details on lego dataset, specularities on hotdog and guitar datasets), which confirms the tendency from Table 4.2. Both *ExCol* and *ExVA* schemes are occasionally lacking specularities (rocket left) and failing to reproduce very fine details of the Lego dataset. The very bright spot on the Lego renders (bottom) is also reproduced incorrectly, however, this is likely due to the models being underfitted.

ExCol vs. NRF

The *ExCol* scheme is originally based on the NRF framework [Bi+20a], which extends the problem of novel view rendering to 'novel view-light rendering' by considering reflectance of the objects in the scene. However, the *ExCol* method in turn uses the NSVF framework for increasing the efficiency of sampling. Although [Bi+20a] do not provide the datasets from their experiments, some rough time estimation comparison can be done. It takes approximately 48 hours for the NRF model to get trained on 4 GPUs NVIDIA RTX 2080Ti. This hardware setup is much more powerful than the hardware used in this thesis. Nevertheless, as it can be seen from Table 4.2 the *ExCol* can get trained in approximately 12 hours on a single GPU NVIDIA 1080Ti, which is 4 times faster than the original NRF approach. Even considering the differences in the datasets and quantitative differences in achieved results, the efficiency improvement is fairly distinct. Unfortunately, [Bi+20a] do not provide any metrics values to compare our results with.

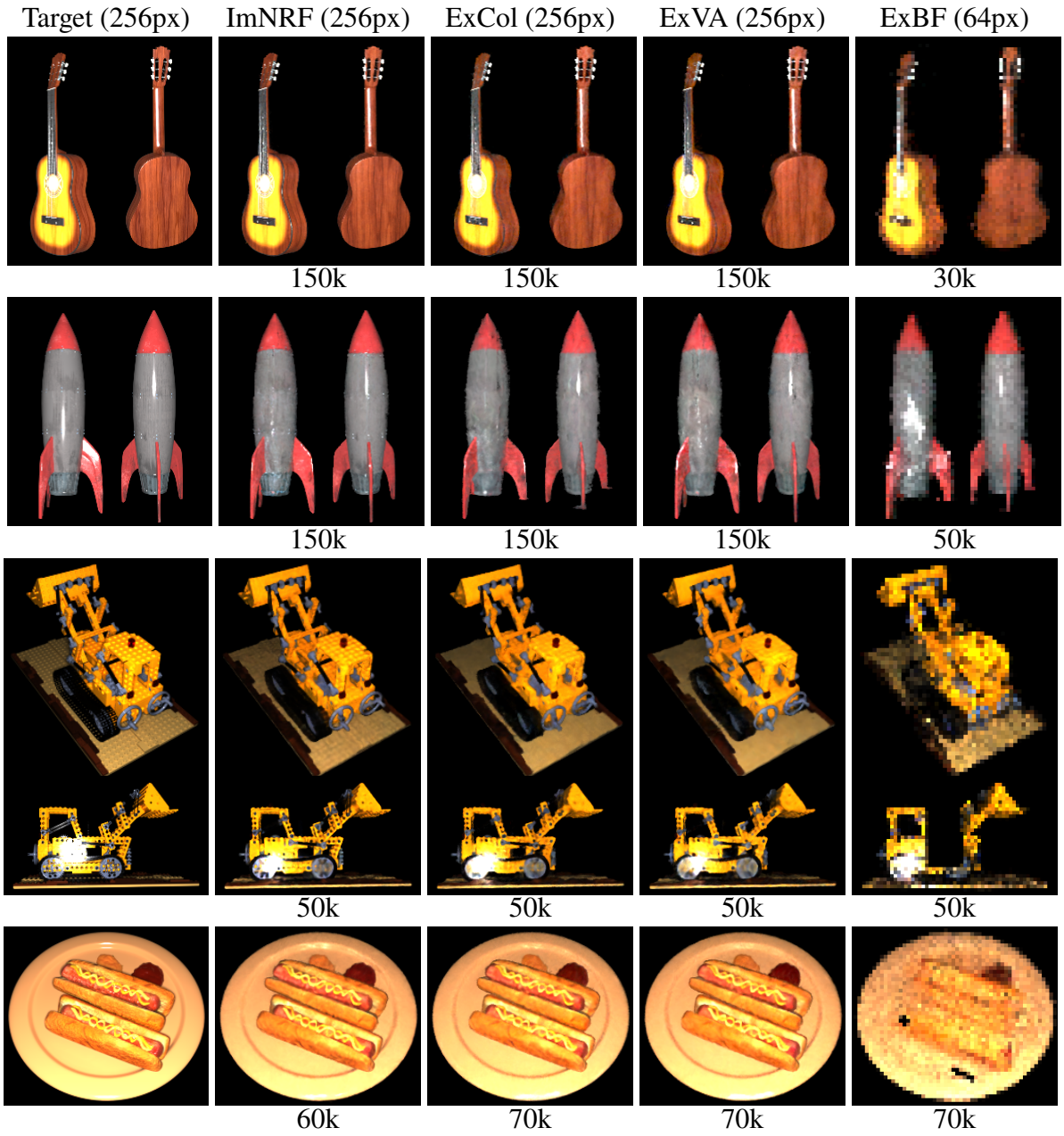


Figure 4.3: The overview of some novel view synthesis results achieved with our methods trained on colocated light setting datasets. Columns correspond to the schemes that have been used to obtain results. The number of training iterations is denoted below each result to have better understanding about model convergence. *ExBF* has only been trained on 64px images due to hardware limitations. *ExBF* poorly handles geometry of Lego and Hotdog datasets, which results in some lacking parts of the scene (corresponding voxels have been pruned). The *ExBF* are generally underfitted as training is very slow and is hard in terms of hardware. The *ExCol* and *ExVA* methods generally produce similar results. However, *ImNRF* produce results on higher level of quality by better handling fine details (lego, rocket) and specularities.

4.4.2 Arbitrary light setting

Datasets with the arbitrary setting cannot be processed with *ExCol* scheme, since the light source is not co-located with the camera. The *ExBF* method is an impractically slow generalization for the *ExCol*. Thus *ExVA* and *ImNRF* are the main methods for achieving renders on this type of datasets. Although *ExVA* is an approximation, which theoretically produces worse results than the 'brute-force' or 'colocated' schemes, experiments with the colocated datasets (Section 4.4.1) showed that evaluations are very similar between *ExVA* and *ExCol* by assessing both using metrics (Table 4.2) and empirically (Figure 4.3). The implicit scheme is not restricted to represent scene appearance with some specific model. On the contrary, it is remained to implicitly learn scene appearance by including light ray directions and distances along with a viewing direction.

Figure 4.4 gives an overview of the results of the two mentioned above methods trained on the Lego dataset with an arbitrary light setting. Images marked with *HDRFlip* are the difference images produced within HDRFlipLoss [And+20; And+21] between target images and corresponding renders. It can be seen that both models reproduced the scene with fairly high quality, although they produce more errors on darker views with some highly specular flares (3rd row, HDRFlip images show strong deviation from target images). Please note that model predictions, as well as target image on the 3rd row, are post-processed by scaling and applying gamma-correction ($\gamma = 2.5$) and in reality, they contain a very dark object with a very bright spot of light reflection. *ImNRF* shows higher overall performance than *ExVA*.

Novel light synthesis

Although [Bi+20a] claim that deep neural network from their framework can generalize to some arbitrary light position while only being trained on coinciding light-view positions, the quality of this generalization is still doubtful. Datasets that were created with the arbitrary light setting contain denser light-view sample space than those created with the colocated setting. This results in a higher quality of novel-light synthesized views. Figure 4.5 contains five columns, which contain images evaluated from the same novel (has not been used during training) camera position: $\phi = 0$ and $\theta = 0$. Each of these images is lit from the light source, which novel locations are different along different columns. The light source's location is defined by two angles: inclination angle θ = const and azimuthal angle ϕ that takes values from -90°to 90°. In the central column, the light source is co-located with the camera, which completely coincides with the light setting used for training *ExCol* (although this view is novel for the method). Both camera and light positions are the same for renders inside each column. Please note that for this experiment *ExCol* has been trained on the dataset with colocated light setting whilst *ExVA* has been trained on the dataset with the arbitrary light setting.

It can be clearly seen that *ImNRF* trained on the colocated light dataset, did not manage to properly generalize for the non-colocated light-view case. There also are no shadows on the

predictions as *ImNRF* does not consider any shadowing. In general *ExVA* method outperforms *ExCol*. The major difference is in the capability of rendering shadows (non-zero angles) while for the zero-angle view both methods perform with approximately the same quality. *Note:* that for evaluation under novel light-view conditions the original NRF method used light transmittance caching (i.e. finding nearest sampled points and interpolating transmittance value between those), which allowed for realistic shadowing effects. In *ExCol no interpolation is performed*, which reasons the lack of shadows on the presented render. Another key observation is that the sigma field of *ExVA* did not converge completely on the front part of the base of the model. *ExCol* renders this part without any difficulties. This difference is most likely due to the more consistent lighting of this part captured in the colocated light dataset comparing to those in the arbitrary light dataset. Another noticeable difference is that *ExVA* better handles fine details on the scene. This can be explained by the fact that these fine features appear in the arbitrary light dataset with its own shadows casted in different directions (depending on light source location), which encourages the density predictor to better distinguish geometrical features from appearance effects. *ImNRF* (trained on the arbitrary light dataset) managed to reproduce the scene even better than *ExVA*. The same tendency as in Figure 4.4 can be noticed that *ImNrf* is able to recover more details comparing to the *ExVA*.

4.4.3 Static light setting

Static light setting is the only type of datasets that can be used for training the original NSVF method, due to the limitation of the illumination to be static. However, this type of dataset appears to be challenging for the novel light-view synthesis task. Although, the sampling in the light-view space is denser comparing with the colocated setting where no other than coincided light and view rays exist, the scene is lit with the light from only one side. This poorly affects the shadowed side as it always appears in the dataset darkened. The underlying effect is the worse capabilities of the model to converge on the sigma field, which plays a significant role in the final appearance estimation. This problem is not method-dependant and is going to appear on all of the presented results. However, the incorrect scene geometry will seriously affect the scene under novel lighting as the shadows will move revealing incorrect parts of the geometry prediction.

This problem is exaggerated by the consideration of using a single light source to illuminate the scene that comes with formulating the rendering equation (equation (3.15)). In the NeRF and NSVF methods, the global static illumination is considered, which is not limited by only one light source and can be very complex.

Some comparisons of evaluations of these two methods are presented in Figure 4.6. The training was performed on static light datasets. Each row consists of three sub-rows that contain evaluation outputs for *ExVA*, *NSVF* and *ImNRF* methods. Columns correspond to different

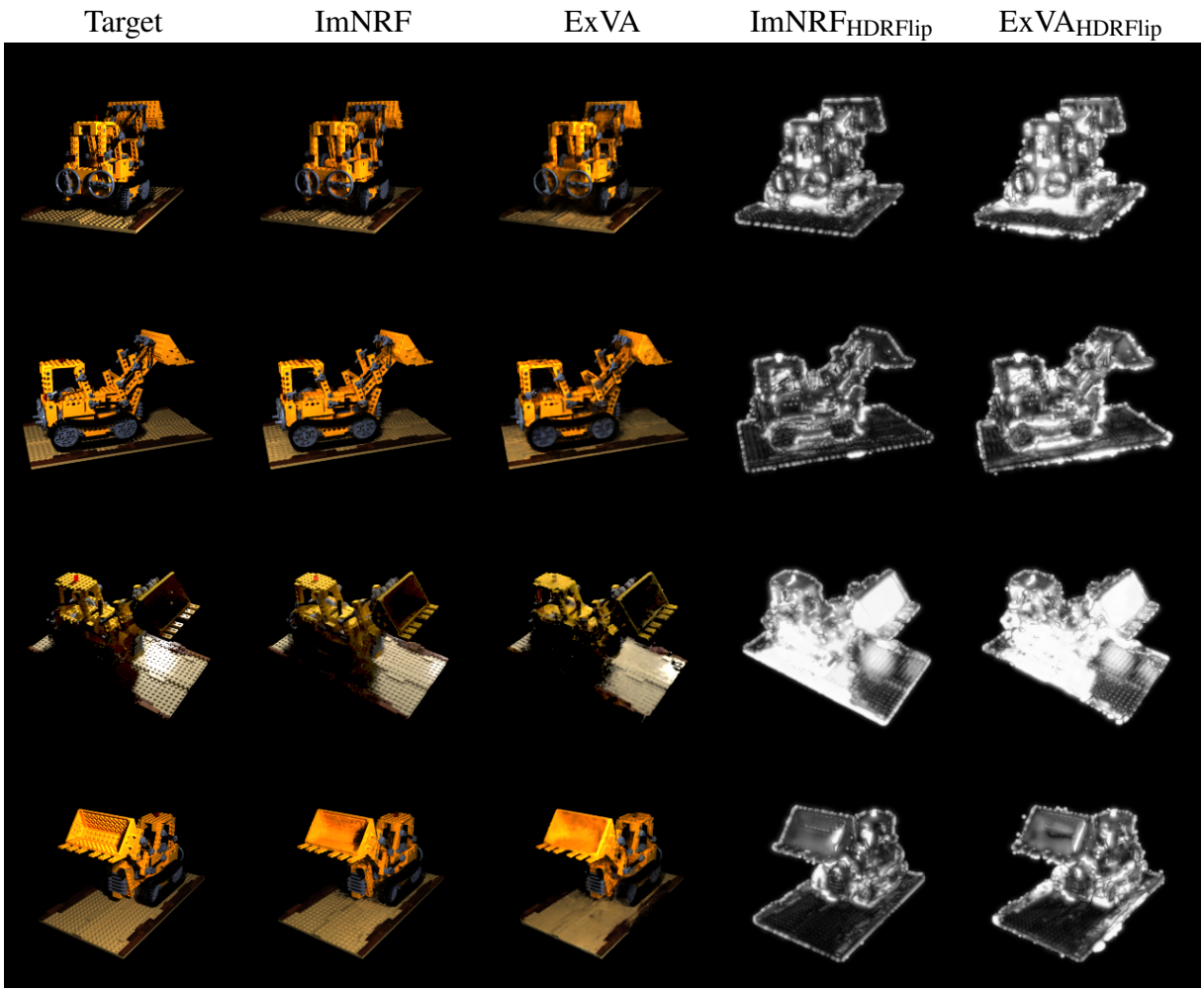


Figure 4.4: The overview of some novel light-view synthesis results achieved with our methods trained on arbitrary light setting Lego dataset for 150k iterations using 256px images. Each row correspond to one specific novel light-view location, second and third columns show synthesized images, last two columns contain difference images calculated between corresponding predictions and target images using HDRFlipLoss (Section 4.3). Renders from the third row have been post-processed (gamma-correction with $\gamma = 2.5$ and scaling) since the original view is very dark with very bright spot from the reflection of the light source. Note that *ImNRF* was able to better reproduce fine details. Both methods fail to predict highly specular spot on the third row.

types of evaluation outputs. *Normal* is the normal map (in the viewing coordinate frame) that is calculated from the sigma field. *Voxel* map is the visualization of the octree for the presented view rendering.

It can be easily seen that all methods struggle with the correct prediction of shadowed parts of the scenes (holes in hotdog dataset, missing part of the base in lego dataset bottom subrow). Although the geometry fails, the produced renderings of the *NSVF* and *ImNRF* method match

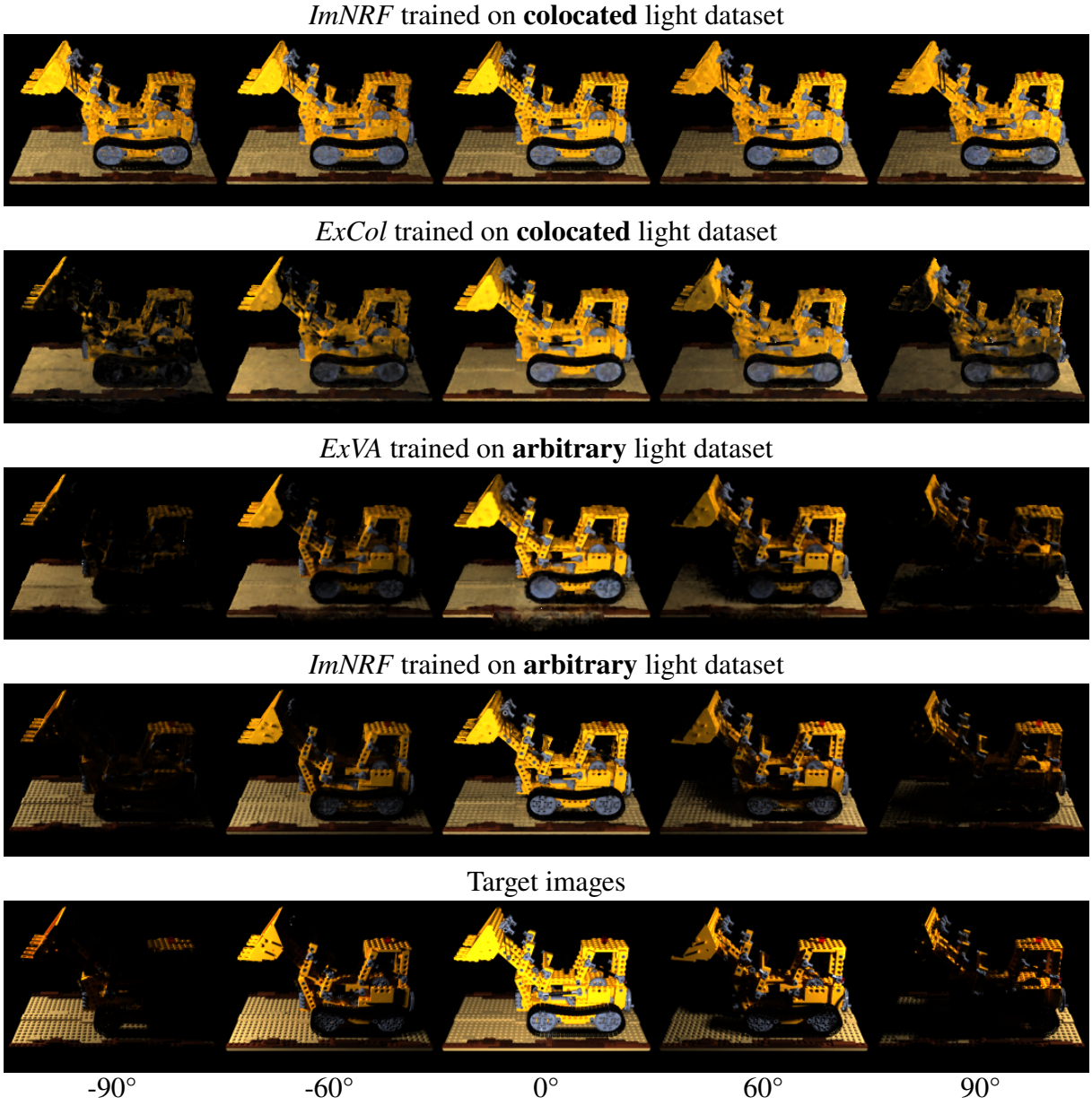


Figure 4.5: Novel view-light synthesis results. The *ExCol* model is trained on dataset with colocated light setting while *ExVA* mode is trained on dataset with arbitrary light setting. Two separate experiments of the *ImNRF* method are presented: using colocated and arbitrary light settings. Both camera view as well as light source location are novel for all models. Camera is fixed at the same position for all of renders. Point light source location is changing along rows: inclination angle θ = const, azimuthal angle ϕ is changing from -90° to 90°. On central column light source is co-located with camera.

target images with fairly high accuracy. The reason for that is the static lighting, which is 'baked' into the scene appearance. On contrary, the *ExVA* method implements the BRDF model, which relies on geometry and is especially sensitive to correct normal estimation on spots with high

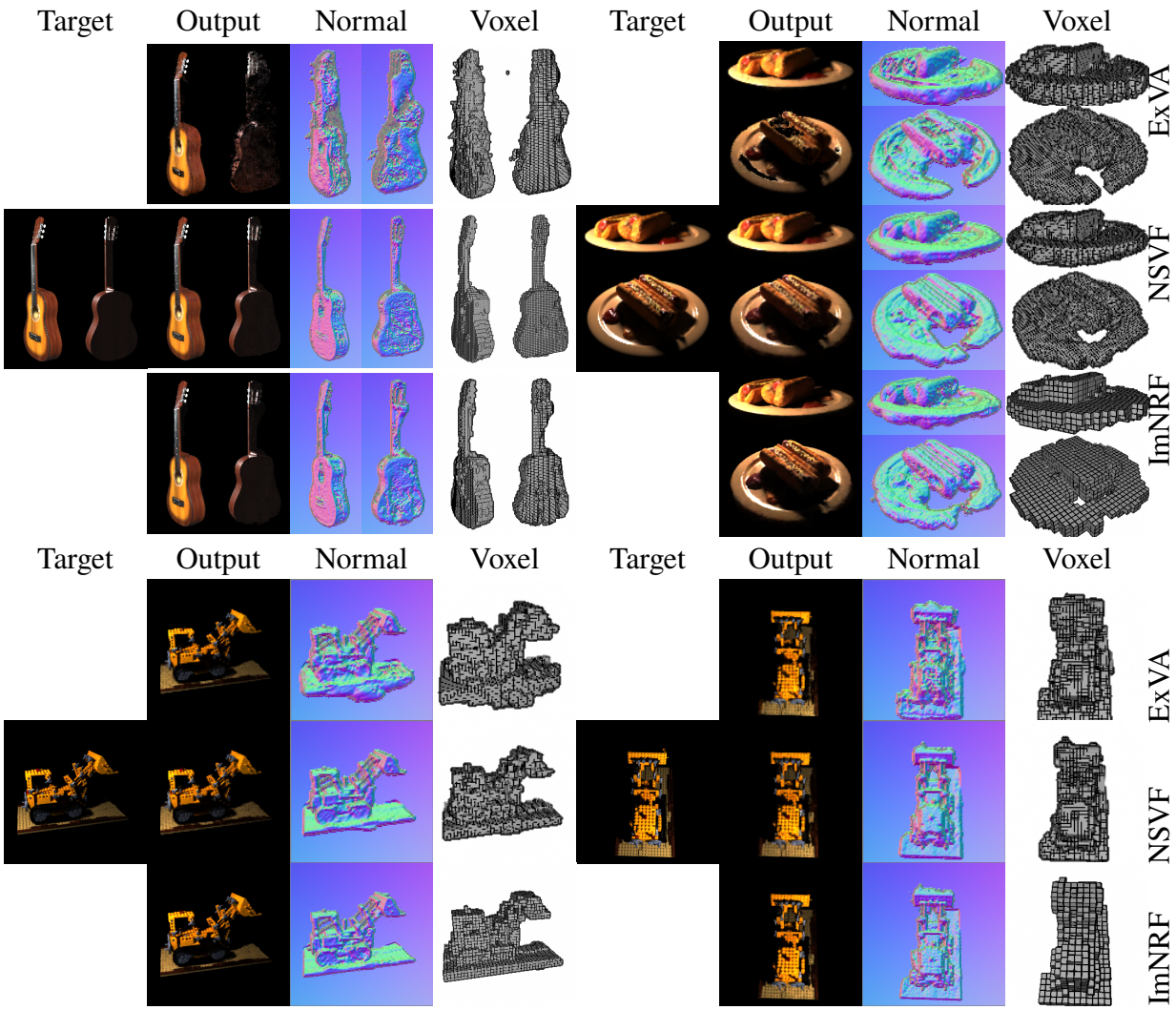


Figure 4.6: The overview of some novel view synthesis results achieved with *ExVA*, *NSVF* and *ImNRF* methods trained on static light datasets. Columns correspond to the output maps. Normal map is calculated from the sigma-field predicted by the model. Note how all schemes fail to learn geometry in the shadowed regions. It can also be seen how more the *ExVA* method is sensitive to inconsistent geometry comparing with the *NSVF* renders. The *ImNRF* and *NSVF* methods shows almost equal performance.

light contribution (i.e. flares from the light source). This behavior can be seen on the hotdog dataset and guitar dataset. In general, both *NSVF* and *ImNRF* outperform *ExVA* on this dataset light setting.

4.5 Concurrent works results

Some results of the concurrent works are shown below in order to make a comparative understanding of our methods' quality.

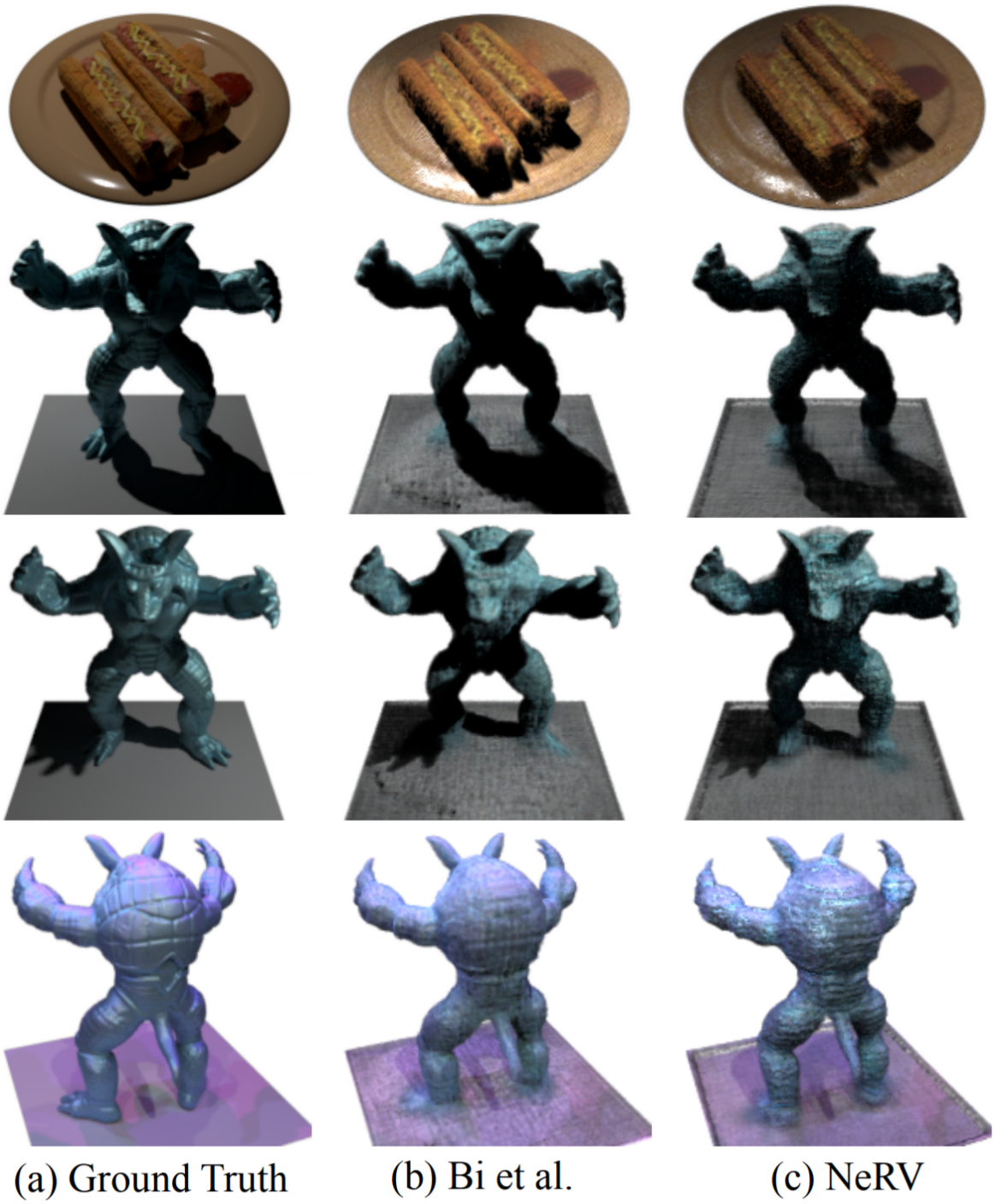


Figure 4.7: Novel view renders with a single point light source illumination of (c) NeRV approach [Sri+21] in comparison with (a) Ground Truth images and renders of (b) NRV approach [Bi+20a]. Images are directly used from NeRV paper [Sri+21].

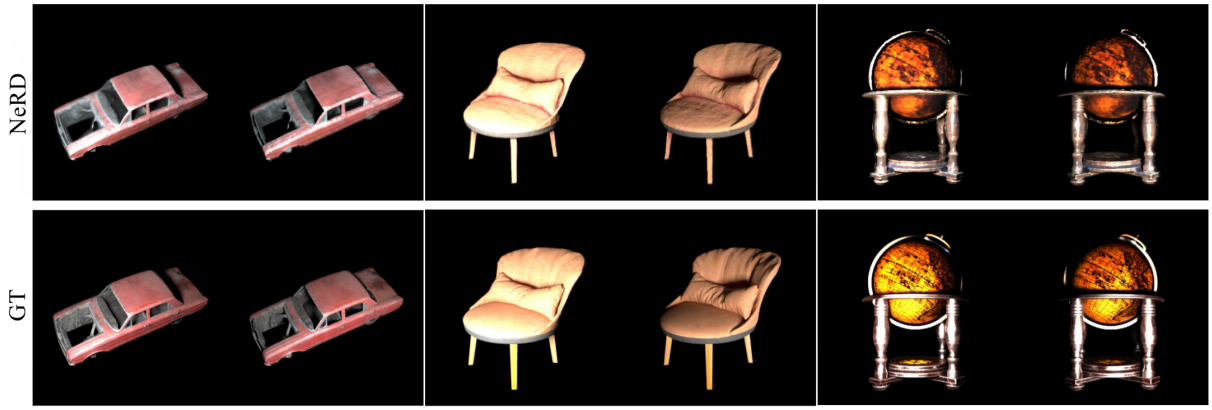


Figure 4.8: Novel view renders with a single point light source illumination of NeRD approach [Bos+20] in comparison with (a) Ground Truth images. Images are directly used from NeRD paper [Bos+20].

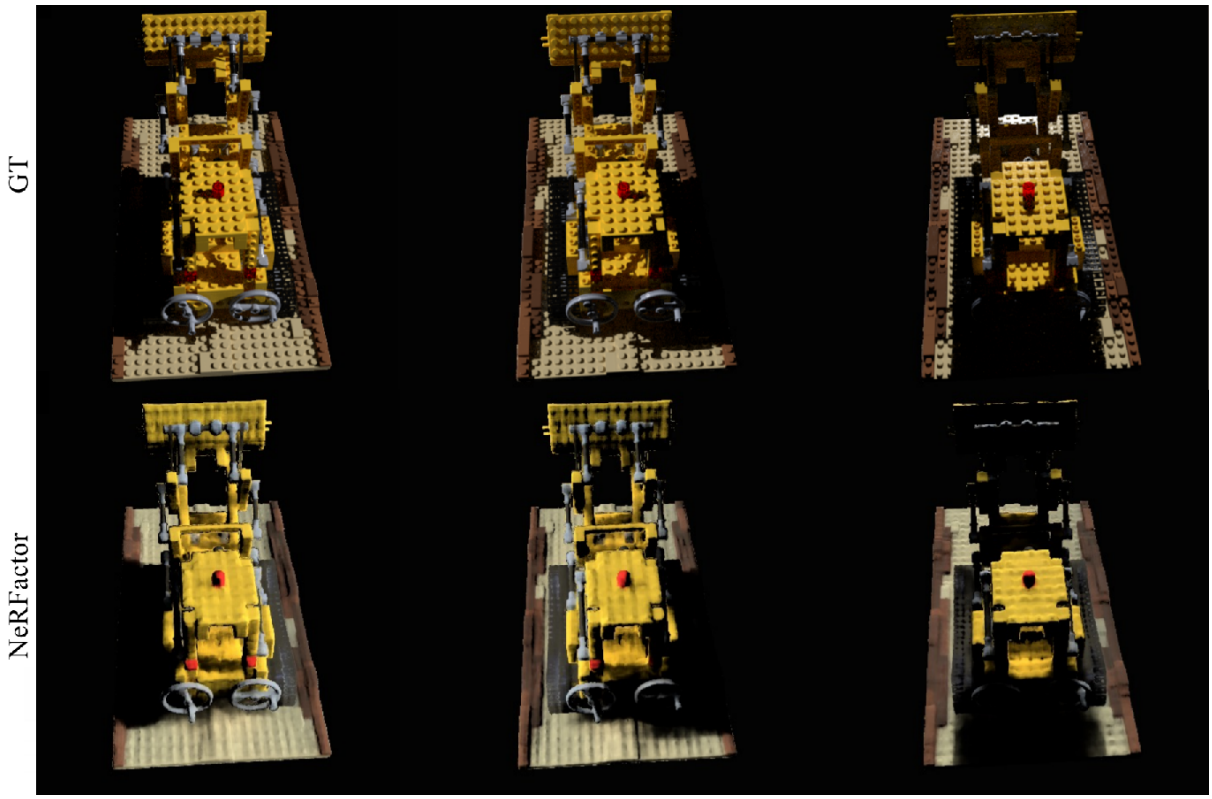


Figure 4.9: Novel view renders with a single point light source illumination of NeRFactor approach [Zha+21] in comparison with (a) Ground Truth images. Images are directly used from NeRFactor paper [Zha+21].

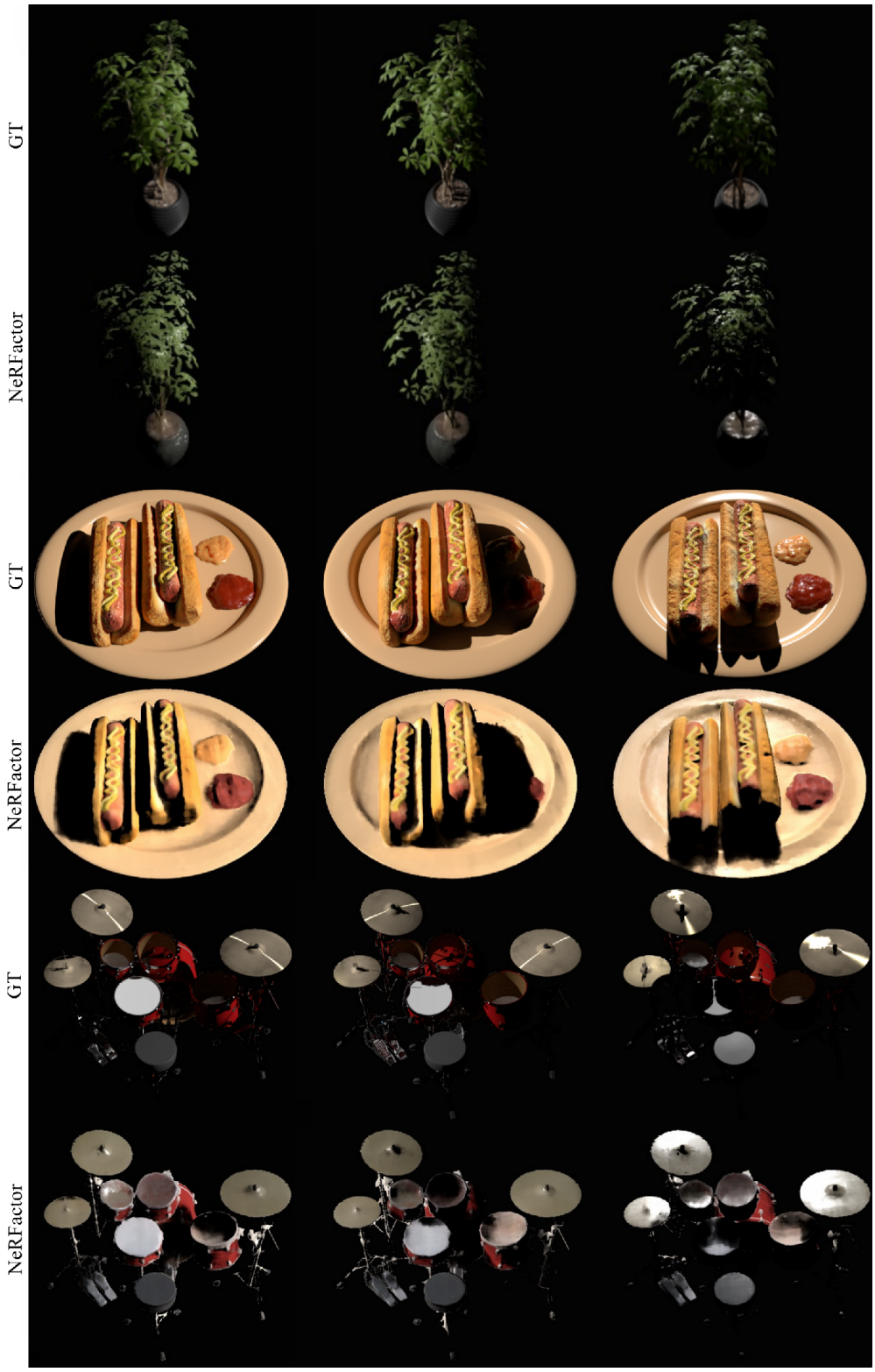


Figure 4.10: Novel view renders with a single point light source illumination of NeRFactor approach [Zha+21] in comparison with (a) Ground Truth images. Images are directly used from NeRFactor paper [Zha+21].

5 Conclusion

This thesis focuses on a comprehensive problem of 3D scene reconstruction. The existing prior works propose different approaches to handle this challenging task, however, most of them do not give enough quality and performance to be practical. This work directly addresses the limitations of NeRF-based [Mil+20] methods and elaborates on their extension to also handle Reflectance Fields similarly as proposed by [Bi+20a].

5.1 Contribution

The main efficiency improvement is connected with using the Neural Sparse Voxel Fields [Liu+21], i.e. voxel octree structure together with the encoding based on the feature vectors that are stored in voxel corners and corresponding framework consisting of such procedures as *self-pruning* and *refinement*. This approach is fused with the reformulation of the rendering equation (Equation (3.1)) to consider a single point light source that illuminates the scene as proposed by [Bi+20a]. This method is called *ExCol* and is the fastest implementation (among other proposed solutions) that is able to reconstruct the scene under novel light-view conditions. This scheme nonetheless retains the limitation of the dataset that can be used for training: it should only consist of co-located light sources (e.g. sharing the same location with the camera in each image sample).

The 'brute-force scheme' (*ExBF*) is proposed as an *ExCol* generalization that is able to handle arbitrary light training data (i.e. when the light source is not restricted to be located at the same position with the camera). However, this method implies casting many light rays that in turn have to be sampled and evaluated by the model. This method can be considered applicable, especially on some powerful hardware setups with multiple high-end GPUs, and comparing with the general formulation of the NRF method [Bi+20a], the complete impracticality is alleviated and better results can be achieved. However, it is still highly memory-exhausting, slow when using within accessible hardware setups and thus fairly unstable on training.

Therefore, the approximation to this scheme that leverages the usage of the voxel octree structure is proposed. The 'brute-force scheme' with the in-voxel approximation is referred to as *ExVA*.

The most computationally expensive part of *ExBF* is the process of obtaining volume densities for the light rays. The entire approximation idea is driven by the assumption that light rays play a secondary role in the contribution to the finally observed value. Under this assumption, it is claimed that volume transmittance for the light rays can be estimated by the distance of travel of the light rays inside the octree voxels (Equation (3.16)). This allows to almost completely eliminate any overheads comparing to *ExCol*. Only light rays intersection with the octree is left, which is efficiently handled using AABB ray intersection algorithm.

The implicit scheme *ImNRF* follows the Vanilla NSVF structure and learns an implicit representation of the scene during the training phase. It is generally able to produce better predictions when comparing with concurrent *ExCol* and *ExVA* methods as well as showing an outstanding efficiency due to the lack of complications (i.e. BRDF model evaluation or light rays sampling). However, *ImNRF* is not regularized to extract appearance from the scene. This means that *ImNRF* is highly sensitive to the training data and how dense is the light-view space in it. Another drawback is the lack of control over the implicit neural representation.

The experiments were held on synthetic datasets listed in Section 4.1.2. The possibility to use the real-world scene in the experiments included the usage of measurements from X-Rite tac7 [MWK17]. However, there were no acquisitions available with calibrated camera poses and the reconstruction of those produced not accurate enough camera calibrations. The BARF approach [Lin+21] could have presumably been better a method for this specific case. The experiments were hence held on synthetic datasets to compare the performance of corresponding schemes. Although *ExBF* scheme is the most general and considered to produce the most accurate results, it appears with questionable applicability and does not impress with the achieved results. The *ExVA* scheme shows appealing results with approximately the same quality as the accurate and second fastest *ExCol*. However, *ExVA* implies no limitation for the training dataset, which is a severe drawback of the *ExCol* method. In general case comparison of the novel light-view synthesis *ExVA* even outperforms *ExCol* by leveraging denser light-view space of the arbitrary light dataset. The *ImNRF* scheme shows the best efficiency and performance comparing to all other methods. However, it is lacking regularization of appearance extraction (e.g. inability to generalize from the colocated light training data to the arbitrary light validation inputs) and control over learned representation.

5.2 Extension points

The proposed solutions already achieve considerable improvements in quality and performance. It can be further developed with a perspective for accelerating training and inference phases as well as for improving the quality of the predictions.

One way is to adopt the technique proposed by [Reb+20], which applies Voronoi-based decomposition for splitting the scene into sub-scenes and then applying multiple networks on these

parts. With this done training time is expected to decrease by 2-3 times.

Another extension can be performed by incorporating the auto integration technique proposed by [LMW21], which increases the speed of the rendering integral estimation. In original work, the increase of efficiency is up to a factor of 10.

The usage of different BRDF models can be considered for achieving better results. For example, in this work, the GGX distribution [Wal+07] is used for the specular term of the microfacet BRDF model. However, better results might be obtained by using the symmetric variant of distribution - SGGX [Hei+15].

5.3 Acknowledgements

I thank Prof. Dr. Reinhard Klein for steering this work in the right direction and my thesis advisor Sebastian Merzbach for a consistent dedicated involvement throughout the entire process I would also like to acknowledge Dr. Michael Weinmann as a second reader of this thesis. The University of Bonn provided computer pools with GPUs that have been used to manage experiments. I thank Oleg Kosenko for comments and CGTrader and Blend Swap users for the 3D models that have been used to create realistic synthetic datasets: Heinzelnisse (lego), vernenb (rocket), AshMesh (guitar) and erickfree (hotdog).

Bibliography

[08] "Binary Space Partitions". In: *Computational Geometry: Algorithms and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 259–281. URL: https://doi.org/10.1007/978-3-540-77974-2_12.

[And+20] P. Andersson, J. Nilsson, T. Akenine-Möller, M. Oskarsson, K. Åström, and M. D. Fairchild. "FLIP: A Difference Evaluator for Alternating Images". In: *Proc. ACM Comput. Graph. Interact. Tech.* 3.2 (Aug. 2020). URL: https://doi.org/10.1145/3406183.

[And+21] P. Andersson, J. Nilsson, P. Shirley, and T. Akenine-Möller. "Visualizing Errors in Rendered High Dynamic Range Images". In: *Eurographics 2021 - Short Papers*. Ed. by H. Theisel and M. Wimmer. The Eurographics Association, 2021.

[Ber+99] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. "The Ball-Pivoting Algorithm for Surface Reconstruction". In: *Visualization and Computer Graphics, IEEE Transactions on* 5 (Nov. 1999), pp. 349–359.

[Bi+20a] S. Bi, Z. Xu, P. Srinivasan, B. Mildenhall, K. Sunkavalli, M. Haan, Y. Hold-Geoffroy, D. Kriegman, and R. Ramamoorthi. *Neural Reflectance Fields for Appearance Acquisition*. 2020. arXiv: 2008.03824 [cs.CV].

[Bi+20b] S. Bi, Z. Xu, K. Sunkavalli, M. Haan, Y. Hold-Geoffroy, D. Kriegman, and R. Ramamoorthi. *Deep Reflectance Volumes: Relightable Reconstructions from Multi-View Photometric Images*. 2020. arXiv: 2007.09892 [cs.CV].

[Bos+20] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. P. Lensch. *NeRD: Neural Reflectance Decomposition from Image Collections*. 2020. arXiv: 2012.03918 [cs.CV].

[Bou19] V. Boucher. *Low Poly Rocket Simple Free low-poly 3D model*. Mar. 2019. URL: https://www.cgtrader.com/free-3d-models/space/spaceship/low-poly-rocket-simple.

[CL96] B. Curless and M. Levoy. "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. New York, NY, USA: Association for Computing Machinery, 1996, pp. 303–312. URL: https://doi.org/10.1145/237170.237269.

[Com18] B. O. Community. *Blender - a 3D modelling and rendering package*. Blender Foundation. Stichting Blender Foundation, Amsterdam, 2018. URL: http://www.blender.org.

[DCH88] R. A. Drebin, L. Carpenter, and P. Hanrahan. "Volume Rendering". In: *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '88. New York, NY, USA: Association for Computing Machinery, 1988, pp. 65–74. URL: https://doi.org/10.1145/54852.378484.

[FSG16] H. Fan, H. Su, and L. Guibas. *A Point Set Generation Network for 3D Object Reconstruction from a Single Image*. 2016. arXiv: 1612.00603 [cs.CV].

[Gar+21] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin. *FastNeRF: High-Fidelity Neural Rendering at 200FPS*. 2021. arXiv: 2103.10380 [cs.CV].

[Gro+18] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. *AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation*. 2018. arXiv: 1802.05384 [cs.CV].

[Hai89] E. Haines. "Essential Ray Tracing Algorithms". In: *An Introduction to Ray Tracing*. GBR: Academic Press Ltd., 1989, pp. 33–77.

[Hei+15] E. Heitz, J. Dupuy, C. Crassin, and C. Dachsbacher. "The SGGX Microflake Distribution". In: *ACM Trans. Graph.* 34.4 (July 2015). URL: https://doi.org/10.1145/2766988.

[HSW89] K. Hornik, M. Stinchcombe, and H. White. "Multilayer feedforward networks are universal approximators". In: *Neural Networks* 2.5 (1989), pp. 359–366. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208.

[HTM17] C. Häne, S. Tulsiani, and J. Malik. *Hierarchical Surface Prediction for 3D Object Reconstruction*. 2017. arXiv: 1704.00710 [cs.CV].

[HZ10] A. Horé and D. Ziou. "Image quality metrics: PSNR vs. SSIM". In: Aug. 2010, pp. 2366–2369.

[JP11] M. Jancosek and T. Pajdla. "Multi-view reconstruction preserving weakly-supported surfaces". In: *CVPR 2011*. IEEE, June 2011. URL: https://doi.org/10.1109/cvpr.2011.5995693.

[KBH04] F. Kainz, R. Bogart, and D. Hess. "OpenEXR image file format". In: (Jan. 2004).

[KK89] J. T. Kajiya and T. L. Kay. "Rendering Fur with Three Dimensional Textures". In: *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '89. New York, NY, USA: Association for Computing Machinery, 1989, pp. 271–280. URL: https://doi.org/10.1145/74333.74361.

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'12. Lake Tahoe, Nevada: Curran Associates Inc., 2012, pp. 1097–1105.

[Leg20] A. Legalov. *Game Ready Acoustic Guitar Free low-poly 3D model*. Oct. 2020. URL: https://www.cgtrader.com/free-3d-models/sports/music/game-ready-acoustic-guitar-baefe8c6-5190-4de0-814f-ff1fd4840df2.

[Lev90] M. Levoy. "Efficient Ray Tracing of Volume Data". In: *ACM Trans. Graph.* 9.3 (July 1990), pp. 245–261. URL: https://doi.org/10.1145/78964.78965.

[Lin+21] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. *BARF: Bundle-Adjusting Neural Radiance Fields*. 2021. arXiv: 2104.06405 [cs.CV].

[Liu+21] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt. *Neural Sparse Voxel Fields*. 2021. arXiv: 2007.11571 [cs.CV].

[LK11] S. Laine and T. Karras. "Efficient Sparse Voxel Octrees Analysis, Extensions, and Implementation". In: 2011.

[LMW21] D. B. Lindell, J. N. P. Martel, and G. Wetzstein. *AutoInt: Automatic Integration for Fast Neural Volume Rendering*. 2021. arXiv: 2012.01714 [cs.CV].

[Lom+19] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh. "Neural volumes". In: *ACM Transactions on Graphics* 38.4 (July 2019), pp. 1–14. URL: http://dx.doi.org/10.1145/3306346.3323020.

[Mad11a] T. Madams. *Improved light attenuation*. Feb. 2011. URL: https://imdoingitwrong.wordpress.com/2011/02/10/improved-light-attenuation/.

[Mad11b] T. Madams. *Light attenuation*. Jan. 2011. URL: https://imdoingitwrong.wordpress.com/2011/01/31/light-attenuation/.

[Mar+21] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. *NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections*. 2021. arXiv: 2008.02268 [cs.CV].

[Max95] N. Max. "Optical models for direct volume rendering". In: *IEEE Transactions on Visualization and Computer Graphics* 1.2 (1995), pp. 99–108.

[Mes+19] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. "Occupancy Networks: Learning 3D Reconstruction in Function Space". In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[Mil+19] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. *Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines*. 2019. arXiv: 1905.00889 [cs.CV].

[Mil+20] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. 2020. arXiv: 2003.08934 [cs.CV].

[MMM12] P. Moulon, P. Monasse, and R. Marlet. "Adaptive Structure from Motion with a Contrario Model Estimation". In: *Proceedings of the Asian Computer Vision Conference (ACCV 2012)*. Springer Berlin Heidelberg, 2012, pp. 257–270.

[Moh97] M. J. Mohlenkamp. "A Fast Transform for Spherical Harmonics". UMI Order No. GAX97-33952. PhD thesis. USA, 1997.

[MWK17] S. Merzbach, M. Weinmann, and R. Klein. "High-Quality Multi-Spectral Reflectance Acquisition with X-Rite TAC7". In: *Proceedings of the Workshop on Material Appearance Modeling*. MAM '17. Helsinki, Finland: Eurographics Association, 2017, pp. 11–16. URL: https://doi.org/10.2312/mam.20171325.

[NA20] J. Nilsson and T. Akenine-Möller. *Understanding SSIM*. 2020. arXiv: 2006.13846 [eess.IV].

[Nie+20] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger. *Differentiable Volumetric Rendering: Learning Implicit 3D Representations without 3D Supervision*. 2020. arXiv: 1912.07372 [cs.CV].

[Nov+18] J. Novák, I. Georgiev, J. Hanika, and W. Jarosz. "Monte Carlo Methods for Volumetric Light Transport Simulation". In: *Computer Graphics Forum (Proceedings of Eurographics - State of the Art Reports)* 37.2 (May 2018).

[Ott+19] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli. "fairseq: A Fast, Extensible Toolkit for Sequence Modeling". In: *Proceedings of NAACL-HLT 2019: Demonstrations*. 2019.

[Pas+19] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. "PyTorch:

An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[Qi+17] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*. 2017. arXiv: 1612.00593 [cs.CV].

[Rah+19] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. A. Hamprecht, Y. Bengio, and A. Courville. *On the Spectral Bias of Neural Networks*. 2019. arXiv: 1806.08734 [stat.ML].

[Reb+20] D. Rebain, W. Jiang, S. Yazdani, K. Li, K. M. Yi, and A. Tagliasacchi. *DeRF: Decomposed Radiance Fields*. 2020. arXiv: 2011.12490 [cs.CV].

[Rei+21] C. Reiser, S. Peng, Y. Liao, and A. Geiger. *KiloNeRF: Speeding up Neural Radiance Fields with Thousands of Tiny MLPs*. 2021. arXiv: 2103.13744 [cs.CV].

[RUG17] G. Riegler, A. O. Ulusoy, and A. Geiger. *OctNet: Learning Deep 3D Representations at High Resolutions*. 2017. arXiv: 1611.05009 [cs.CV].

[SF16] J. L. Schönberger and J.-M. Frahm. "Structure-from-Motion Revisited". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4104–4113.

[Sit+19] V. Sitzmann, J. Thies, F. Heide, M. NieSSner, G. Wetzstein, and M. Zollhöfer. "DeepVoxels: Learning Persistent 3D Feature Embeddings". In: *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*. 2019.

[SK00] H. Shum and S. B. Kang. "A Review of Image-Based Rendering Techniques". In: vol. 4067. May 2000, pp. 2–13.

[Sri+21] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron. "NeRV: Neural Reflectance and Visibility Fields for Relighting and View Synthesis". In: *CVPR*. 2021.

[SZW19] V. Sitzmann, M. Zollhöfer, and G. Wetzstein. "Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations". In: *Advances in Neural Information Processing Systems*. 2019.

[Tan+20] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng. "Fourier Features Let Networks

Learn High Frequency Functions in Low Dimensional Domains". In: *NeurIPS* (2020).

[TDB17] M. Tatarchenko, A. Dosovitskiy, and T. Brox. *Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs*. 2017. arXiv: 1703.09438 [cs.CV].

[Tew+20] A. Tewari, O. Fried, J. Thies, V. Sitzmann, S. Lombardi, K. Sunkavalli, R. Martin-Brualla, T. Simon, J. Saragih, M. NieSSner, R. Pandey, S. Fanello, G. Wetzstein, J.-Y. Zhu, C. Theobalt, M. Agrawala, E. Shechtman, D. B. Goldman, and M. Zollhöfer. *State of the Art on Neural Rendering*. 2020. arXiv: 2004.03805 [cs.CV].

[Vas+17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. *Attention Is All You Need*. 2017. arXiv: 1706.03762 [cs.CL].

[Wal+07] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance. "Microfacet Models for Refraction through Rough Surfaces". In: *Proceedings of the 18th Eurographics Conference on Rendering Techniques*. EGSR'07. Grenoble, France: Eurographics Association, 2007, pp. 195–206.

[Wan+04] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612.

[Wan+18] P. Wang, Y. Gan, P. Shui, F. Yu, Y. Zhang, S. Chen, and Z. Sun. *3D Shape Segmentation via Shape Fully Convolutional Networks*. 2018. arXiv: 1702.08675 [cs.CV].

[Yu+21] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa. *PlenOctrees for Real-time Rendering of Neural Radiance Fields*. 2021. arXiv: 2103.14024 [cs.CV].

[Zha+18] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". In: *CVPR*. 2018.

[Zha+20] K. Zhang, G. Riegler, N. Snavely, and V. Koltun. *NeRF++: Analyzing and Improving Neural Radiance Fields*. 2020. arXiv: 2010.07492 [cs.CV].

[Zha+21] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron. "NeRFactor: Neural Factorization of Shape and Reflectance Under an Unknown Illumination". In: *arXiv preprint arXiv:2106.01970* (2021).