# Neural Reflectance Decomposition

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

M.Sc. Mark Benedikt Boss
aus Münster

Tübingen
2022

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

| | |
|---|---|
| Tag der mündlichen Qualifikation: | 03.03.2023 |
| Dekan: | Prof. Dr. Thilo Stehle |
| 1. Berichterstatter: | Prof. Dr. Hendrik P. A. Lensch |
| 2. Berichterstatter: | Prof. Dr. Andreas Schilling |
| 3. Berichterstatter: | Dr. George Drettakis |

I would first like to thank my supervisor, Professor Hendrik Lensch, for the countless research conversations during coffee breaks and video calls. Your insightful feedback helped me in every project. I also thank my second supervisor, Professor Andreas Schilling, and the entire Ph.D. committee.

I also thank all my contributors. I especially want to thank Varun Jampani, whom I consider an additional supervisor. Similarly, Jonathan Barron provided many thoughtful insights into our collaborations. I want to thank every collaborator during my internships at Nvidia and Google.

For the publications, I also collaborated with Raphael Braun and Andreas Engelhardt. Thank you so much for your support. It was an amazing experience working with you on the NeRDy projects.

I also thank the entire computer graphics group for countless research conversations. I thank Benjamin Resch and Lukas Ruppert for keeping the compute resources alive. Without you, many deadlines would have failed. Similarly, I especially want to thank the colleagues in my office during the Ph.D.: Raphael Braun and Patrick Wieschollek, for keeping my sanity up during the deadlines mentioned earlier.

Lastly, I must express gratitude to my wife, Nadine Simon, and my parents for providing me with support and continuous encouragement throughout my years of study. This work would not have been possible without them.

Thank you.

# Abstract

Creating relightable objects from images or collections is a fundamental challenge in computer vision and graphics. This problem is also known as *inverse rendering*. One of the main challenges in this task is the high ambiguity. The creation of images from 3D objects is well defined as rendering. However, multiple properties such as shape, illumination, and surface reflectiveness influence each other. Additionally, an integration of these influences is performed to form the final image. Reversing these integrated dependencies is highly ill-posed and ambiguous. However, solving the task is essential, as automated creation of relightable objects has various applications in online shopping, Augmented Reality (AR), Virtual Reality (VR), games, or movies.

In this thesis, we propose two approaches to solve this task. First, a network architecture is discussed, which generalizes the decomposition of a two-shot capture of an object from large training datasets. The degree of novel view synthesis is limited as only a singular perspective is used in the decomposition. Therefore, the second set of approaches is proposed, which decomposes a set of 360-degree images. These multi-view images are optimized per object, and the result can be directly used in standard rendering software or games. We achieve this by extending recent research on *Neural Fields*, which can store information in a 3D neural volume. Leveraging volume rendering techniques, we can optimize a reflectance field from in-the-wild image collections without any Ground Truth (GT) supervision.

Our proposed methods achieve state-of-the-art decomposition quality and enable novel capture setups where objects can be under varying illumination or in different locations, which is typical for online image collections.

# Kurzfassung

Die Erstellung von fotorealistischen Modellen von Objekten aus Bildern oder Bildersammlungen ist eine grundlegende Herausforderung in der Computer Vision und Grafik. Dieses Problem wird auch als *inverses Rendering* bezeichnet. Eine der größten Herausforderungen bei dieser Aufgabe ist die vielfältige Ambiguität. Der Prozess Bilder aus 3D-Objekten zu erzeugen wird Rendering genannt. Allerdings beeinflussen sich mehrere Eigenschaften wie Form, Beleuchtung und die Reflektivität der Oberfläche gegenseitig. Zusätzlich wird eine Integration dieser Einflüsse durchgeführt, um das endgültige Bild zu erzeugen. Die Umkehrung dieser integrierten Abhängigkeiten ist eine äußerst schwierige und mehrdeutige Aufgabenstellung. Die Lösung dieser Aufgabe ist jedoch von entscheidender Bedeutung, da die automatisierte Erstellung solcher wieder beleuchtbaren Objekte verschiedene Anwendungen in den Bereichen Online-Shopping, Augmented Reality (AR), Virtual Reality (VR), Spiele oder Filme hat.

In dieser Arbeit werden zwei Ansätze zur Lösung dieser Aufgabe beschrieben. Erstens wird eine Netzwerkarchitektur vorgestellt, die die Erfassung eines Objekts und dessen Materialien von zwei Aufnahmen ermöglicht. Der Grad der Blicksynthese von diesen Objekten ist jedoch begrenzt, da bei der Dekomposition nur eine einzige Perspektive verwendet wird. Daher wird eine zweite Reihe von Ansätzen vorgeschlagen, bei denen eine Sammlung von 360 Grad verteilten Bildern in die Form, Reflektanz und Beleuchtung gespalten werden. Diese Multi-View-Bilder werden pro Objekt optimiert. Das resultierende Objekt kann direkt in handelsüblicher Rendering-Software oder in Spielen verwendet werden. Wir erreichen dies, indem wir die aktuelle Forschung zu *neuronalen Feldern* erweitern Reflektanz zu speichern. Durch den Einsatz von Volumen-Rendering-Techniken können wir ein Reflektanzfeld aus natürlichen Bildsammlungen ohne jegliche Ground Truth (GT) Überwachung optimieren.

Die von uns vorgeschlagenen Methoden erreichen eine erstklassige Qualität der Dekomposition und ermöglichen neuartige Aufnahmesituationen, in denen sich Objekte unter verschiedenen Beleuchtungsbedingungen oder an verschiedenen Orten befinden können, was üblich für Online-Bildsammlungen ist.

# Publications

This thesis is based on the following publications:

Two-shot Spatially-varying BRDF and Shape Estimation - **Mark Boss**, Varun Jampani, Kihwan Kim, Hendrik P. A. Lensch, Jan Kautz - *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* - 2020

NeRD: Neural Reflectance Decomposition from Image Collections - **Mark Boss**, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, Hendrik P. A. Lensch - *IEEE International Conference on Computer Vision (ICCV)* - 2021

Neural-PIL: Neural Pre-Integrated Lighting for Reflectance Decomposition - **Mark Boss**, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, Hendrik P. A. Lensch - *Advances in Neural Information Processing Systems (NeurIPS)* - 2021

SAMURAI: Shape And Material from Unconstrained Arbitrary Image collections - **Mark Boss**, Andreas Engelhardt, Abhishek Kar, Yuanzhen Li, Deqing Sun, Jonathan T. Barron, Hendrik P. A. Lensch, Varun Jampani - *Advances in Neural Information Processing Systems (NeurIPS)* - 2022

Chapters, which include passages or modified passages of these publications will be marked in the beginning of the corresponding chapter.

**Non-Incorporated Publications:**

Medicine quality screening: TLCyzer, an open-source smartphone-based imaging algorithm for quantitative evaluation of thin-layer chromatographic analyses using the GPHF Minilab - Cathrin Hauk, **Mark Boss**, Julia Gabel, Simon Schäfermann, Hendrik P. A. Lensch, Lutz Heide - *Scientific Reports* - 2022

Single Image BRDF Parameter Estimation with a Conditional Adversarial Network - **Mark Boss**, Hendrik P. A. Lensch - *ArXiv* - 2019

Deep Dual Loss BRDF Parameter Estimation - **Mark Boss**, Fabian Groh, Sebastian Herholz, Hendrik P. A. Lensch - *Workshop on Material Appearance Modeling* - 2018

# Contents

# Chapter 1

# Introduction

Inverse rendering is the task of decomposing a scene or object into its underlying physical properties, such as geometry, illumination, and materials. Recovering these properties is useful for several vision and graphics applications such as view synthesis [25, 26, 28, 29, 150, 185], relighting [25, 26, 28, 29, 94, 150, 186], object insertion [22, 58, 94] or automated asset creation for games or movies [7, 25, 26, 29, 121].

Digitizing real-world objects into 3D models gained a large following due to trends such as Augmented Reality (AR), Virtual Reality (VR), Mixed Reality (MR) or in general the Metaverse. For example, Microsoft created a digital version of the entire planet for the recent Flight Simulator [152]. Providing automatic methods for digitizing objects, rooms, or larger environments is a significant challenge in these fields. Furthermore, capturing these images of potential assets is also a time-consuming task, and extensive collections of online images exist that capture nearly every object. However, these collections are captured in varying illuminations or even locations. Converting these image collections into useful assets for games or other applications is challenging. Ideally, the assets need to be relightable as they have to integrate into various environments. This requirement is highly beneficial in AR applications.

The main challenge in converting images into relightable assets is the high ambiguity of the decomposition. A scene needs to be split into geometry, appearance, and illumination. The appearance is often described as the Bidirectional Reflectance Distribution Function (BRDF), defined in Sec. 2.1. As each of these elements influences the others, this problem is highly ill-posed. For example, a visible dark spot on an object can either appear due to a hole (shape), a darker texture (appearance), or no light being cast towards that spot (illumination). This challenging problem is explained in Sec. 2.3. Due to the high ambiguity, only partial decompositions are often addressed. For example, the shape [88, 89] or the illumination [27, 150] are expected to be Ground Truth (GT) and provided by a dedicated measurement processes. For example, the shape can be scanned using Light Detection And Ranging (LiDAR). In this thesis, all properties are estimated jointly without GT information.

Often the illumination is also controlled [27, 30, 88, 89, 122] by requiring a flash light source or fixed illumination patterns from a light stage [42]. Instead of these constrained setups, the highly casual capture setup is tackled in this thesis. Our objects are not in

a controlled environment but outside or in rooms under natural illuminations. These illuminations can also vary for each image. Combining the full decomposition with simple capture setups enables object reconstruction from online image collections.

This thesis presents two types of approaches for the decomposition of objects from images or image collections. Here, the distinction between a generalized and a per-object method is applied.

For a generalized method, a neural network is trained once on a large corpus of labeled data [27, 29, 30, 44, 45, 98]. For novel objects, the trained network weights are then evaluated. This enables a fast estimation of the physical properties of an object. Furthermore, only a single perspective is required. The resulting decompositions can be used for a small degree of novel view synthesis and easily relit. However, as existing weights are only evaluated after the training, no optimization for the specific instance is performed. When the object is not easily related to the corpus of the training data, the reconstruction quality is reduced significantly.

Additionally, as only a single perspective is optimized, no complete 3D object is created. The object cannot be visualized in 360 degrees, and fusing multiple estimations from a single perspective is challenging [50, 83, 178]. Furthermore, no actual global minimum is easily reached after the estimation, as each estimation is performed independently.

In the second case, a network is optimized for each object. However, no supervision is required in this case. We only require posed image collection for the optimization. For our contribution in Sec. 4.4, we even enable a reconstruction from a coarse posed image based on quadrants. Additionally, the optimization is performed from a larger set of images, which can be distributed 360 degrees around the object. Therefore, a complete 3D decomposition is performed. The resulting assets can be directly used in AR or VR applications and achieve higher quality due to the optimization for each dataset. The reconstructed object reaches a global minimum more easily as the decomposition has to explain all input images.

Overall, both approaches learn to decompose the input into geometry, Spatially-varying Bidirectional Reflectance Distribution Function (SVBRDF) (see Sec. 2.1.2), and the illumination. Due to the decomposition, the assets can be easily relit. We achieve this by optimizing the BRDF alongside the geometry and illumination. Our methods enable the automatic creation of 3D assets from arbitrary image collections or images from a singular perspective. The assets can be used in e-commerce, AR or VR applications, games, and movies. Enabling this from existing image collections instead of specifically captured ones can democratize the creation of 3D assets and even automate the creation of a large number of objects.

## 1.1 Main Contributions and Research Questions

The main objective of this thesis is to reconstruct an object model and decompose images or image collections into an explicit BRDF, shape, and illumination. Achieving this is essential for consistently relightable reconstructions. This problem is highly ambiguous and ill-posed. Therefore, a recurring theme is to steer the decomposition and constrain it to arrive at or strive toward a global optimum instead of getting stuck in local minima. Another key challenge is differentiable rendering and easily incorporating environment illumination to allow realistic real-world and unconstrained capture setups. Lastly, another goal is to enable the reconstruction of highly challenging in-the-wild image collections without any GT knowledge of pose, shape, illumination, or material.

Our main research question and contributions are as follows:

**Research Question 1:** *Can we leverage shading cues from flash and no-flash image pairs?*

Capturing images with a co-located camera flash reduces the ambiguity, as the predominant light source has nearly the same origin as the view. We define a *co-located light source* as a light close to the camera. The ray direction for the view is approximately the same as the light source. Moreover, we assume that the flash provides a powerful light source that predominantly lights the scene.

However, due to the flash, images have overexposed regions at the highlight, or the influence of the environment illumination is not easily visible. Providing information by taking a secondary image without a flash can unlock this information for the evaluation. By capturing this two-shot approach, additional information is available. For example, light decreases in strength over distance with the inverse square root. With both images, the visible illumination from the flash image can be leveraged to estimate the shape from the shading [99]. In our generalized decomposition method [29] in Sec. 3.2 we leverage this capture scenario with a hand-held burst capture setup. We devise a specialized network architecture to process the two-shot image in separate streams. We call these *MergeConvolutions*. We further show that capturing images in this two-shot approach improves the reconstruction quality drastically compared to a single shot under flash illumination.

**Research Question 2:** *Is a cascaded decomposition process beneficial with highly specialized networks for each task?*

In our work on generalized decomposition [29] in Sec. 3.2 we also compared the influence of creating a joint large network *vs*. several smaller networks for specific tasks. Both network architectures use a similar amount of weights to store a comparable amount of information. We found that creating highly specialized architectures for each task to be beneficial. Here, we split the decomposition into individual tasks. A network for shape estimation is followed by illumination estimation, and finally, the SVBRDF is

estimated. We then leverage the re-render of our initial estimation under the same setting as the flash image and use the difference to guide a final refinement stage. In this stage, the shape and BRDF are jointly optimized, which helps escape the local minima from the separate estimations. One side-advantage is that each network is smaller than a joint one, and each can be efficiently run on mobile hardware.

**Research Question 3:** *Can we reconstruct relightable 3D assets from image collections under varying environment illumination?*

Creating 3D assets from image collections is an inherently challenging problem. This problem is exacerbated considerably when each image has a different illumination. Each illumination needs to be optimized alongside the global shape and material model. We tackle this challenging task in three works: NeRD [25] in Sec. 4.2, Neural-PIL [28] in Sec. 4.3, and SAMURAI [26] in Sec. 4.4. In NeRD and SAMURAI, we propose an automated textured mesh extraction scheme. The resulting assets are easily useable in any game engine or rendering software. SAMURAI can create these assets from a highly challenging dataset based on online image collections.

**Research Question 4:** *What is an efficient way to render environment illumination in a differentiable renderer?*

During the per-object decomposition or the generalized training of networks, the current estimate is often rendered to compare it directly to the input image. This process has to be executed millions of times during the training. Having a fast rendering process is, therefore, a key challenge. The entire hemisphere of incoming light has to be integrated, which requires many evaluations and is therefore too expensive to perform during the optimization. Moreover, the entire process must be differentiable as described in Sec. 2.3.5. With classical Monte-Carlo integration, described in Sec. 2.1, the gradient is extremely noisy due to its stochastic nature, and providing a stable gradient requires many samples per pixel. We, therefore, apply methods to perform the integration of environment illumination more easily. We leverage Spherical Gaussian (SG) illumination models as described in Sec. 2.2.1 in our generalized decomposition method of Sec. 3.2 and NeRD of Sec. 4.2. As SG illuminations are mostly low-frequent we convert the pre-integration method of real-time rendering described in Sec. 2.2.2 into a neural network. This achieves higher quality and faster rendering. We leverage this approach in Neural-PIL of Sec. 4.3 and SAMURAI of Sec. 4.4.

**Research Question 5:** *How can we introduce priors in a per-object optimization process?*

Our generalized decomposition method of Sec. 3.2 is capable of learning from the statistical properties of the training dataset. The decomposition is mostly performed using data-driven priors. Is it possible to leverage similar priors in our per-object optimization methods? We introduced a sparse auto-encoder in NeRD and SAMURAI, which

enforces that similar BRDF materials are mapped to the same latent space. Then surface points of similar materials optimize the same underlying BRDF. This can aid in the decomposition, as the BRDF reconstruction is improved drastically.

Furthermore, we explore creating a general network trained on large quantities of synthetic data with specialized losses. The resulting *Smooth Manifold Autoencoder* (SMAE), is described in Sec. 4.3. The latent space from this network can then be used with frozen weights, as the gradient in the latent space is also smooth, and a valid manifold is defined. We employ this technique for the BRDF and illumination, which are then constrained to natural illumination and materials.

**Research Question 6:** *Is it possible to jointly recover poses and perform a decomposition in challenging datasets?*

Traditional 3D reconstruction and camera pose estimation methods often leverage correspondences. A popular example of this is COLMAP [145, 146]. Correspondences work well in scenes under single illumination and at a single location, where the background can also be leveraged for the pose recovery. This is especially critical when reconstructing objects, as often, objects are relatively homogeneous. When an object is under different illumination and in various locations, COLMAP often fails to recover the pose and shape. This is primarily due to inconsistent features from the varying illumination on the object. In Sec. 4.4 we tackle this problem using a combination of a *Camera Multiplex* inspired by Goel *et al*. [60] and a flexible camera parametrization and volume definition, which allows non-equidistant cameras. We even found that leveraging an explicit BRDF is beneficial compared to learning the plenoptic function defined in Sec. 2.3.3.

**Research Question 7:** *Can we reduce the influence of difficult-to-align images in highly challenging datasets?*

Even with the introduction of the novel contributions of **Research Question 6**, some poses are difficult to align. We also want to reduce the impact of poorly aligned poses on the global shape and BRDF reconstruction. This is achieved with *posterior scaling* as introduced in SAMURAI in Sec. 4.4. Here, we reweigh the losses for the network based on the current estimate. A poorly aligned pose has reduced influence compared to a well-aligned one. This improves the recovered shape and stability during training.

## 1.2 Outline

The thesis is structured as follows: Several fundamental areas useful for the thesis are discussed in Chapter 2. Here, the concept of rendering a 3D scene using Monte Carlo Integration is introduced in Sec. 2.1, and the BRDF is introduced in Sec. 2.1.2. Efficient environment illumination rendering used in the proposed methods is explained in Sec. 2.2. The process of inverse rendering, the challenges, and techniques are discussed

in Sec. 2.3. Lastly, the recent topic of *Neural Fields* is introduced in Sec. 2.4. Neural Fields and rendering of such fields are used in our per-object decomposition methods.

In Chapter 3, we introduce our generalized decomposition method, which given a pair of flash and no-flash images, estimates the shape, illumination, and BRDF. This is done using a novel dataset and a cascaded network architecture. These highly specialized networks of the cascaded structures are introduced in Sec. 3.2.3. As this method also leverages environment illumination, the SG rendering of Sec. 2.2.1 efficiently integrates the illumination. As no large-scale, GT labeled dataset exist to form our training corpus, we create a dataset in Sec. 3.2.4 using *domain randomization*. The dataset is generated from random objects, and the network must learn the concept of shape and shading without relying on semantic information. We show that our network can transfer the decomposition capabilities acquired from this random dataset to real-world objects.

The second part of our method covers the per-object decomposition methods in Chapter 4. Here, three methods are discussed. The first method, NeRD, extends on the recent NeRF [117] method and their use of Neural Fields (Sec. 2.4) with explicit BRDF decomposition and a SG illumination model (Sec. 2.2.1). An important distinguishing element is that we allow each image to be in a different illumination. This is especially relevant when in-the-wild online image collections are to be used. We achieve this using a specialized architecture for the decomposition described in Sec. 4.2.2.2 and a sparse auto-encoder for enforcing similar materials to occupy the same latent space. Furthermore, we aim to model the real-world camera processing in Sec. 4.2.2.5 to reduce the ambiguity due to white balance, tone mapping, and auto-exposure.

While NeRD achieved significant strides in decomposing in-the-wild image collections, several issues remain. While SG illumination models are fast to integrate, the resulting illumination is mostly low-frequency, and artifacts from the isotropic spherical lobes remain. This issue is also known in real-time rendering, and a pre-integrated illumination model described in Sec. 2.2.2 aims to solve this. However, an expensive pre-integration step is still required. Furthermore, in Chapter 3 we have shown that priors from datasets are effective in decomposing scenes. Therefore, we extend NeRD in Sec. 4.3 and propose Neural-PIL, which tackles both issues. We devise a way to learn smooth manifolds of BRDFs and natural illuminations using our proposed *Smooth Manifold Autoencoder* (SMAE) losses and architecture definition. With the smooth manifold, we can easily leverage the networks with frozen weights and only optimize the BRDF using the latent codes in the neural field or per-image for the illumination. Furthermore, we replace the expensive offline pre-integration with a network. This network is not only capable of performing the pre-integration but at the same time limiting our predictions to plausible natural illumination patterns. This significantly speeds up the rendering compared to SG and improves the quality.

Both previous methods require known camera poses from COLMAP [145, 146]. However, due to varying illumination and locations in online image collections, COLMAP fails in these cases. In our method, SAMURAI, we jointly decompose the scene and optimize the camera poses as described in Sec. 4.4. As the decomposition into shape,

material, and illumination is already difficult, introducing the added ambiguity from camera pose optimization requires several novel additions. Our method now only requires a rough quadrant-based initialization where each camera is placed in direction quadrants such as *Front vs*. *Back*, *Left vs*. *Right*, and *Top vs*. *Bottom* and the combinations of those. As the offset to the actual camera position is still large with these quadrant initializations, we employ a *Camera Multiplex*: Multiple random camera estimates around the quadrant are created and jointly optimized during the training. Unlikely camera estimates are phased out during the optimization. We reduce the influence of poorly aligned images with a posterior scaling, which reduces the influence of these poses on our shape, BRDF, and illumination estimation. Lastly, the camera is at varying distances from the object in real-world image collections. We solve this using a flexible camera parametrization alongside a Neural Field volume-bound definition. Given these novel additions, SAMURAI can run on highly challenging, in-the-wild image collections.

# Chapter 2

# Foundations

This chapter introduces general concepts from forward rendering used throughout the thesis. Additionally, it introduces the challenges and problems with inverse rendering.

## 2.1 Rendering

The rendering process generates a photorealistic 2D image from a 3D scene description. For this, the underlying physical process of light propagation must be followed. If this process is followed, the rendering is physically based, and the resulting image should be indistinguishable from a real image. Often this is achieved by ray tracing, where infinitesimally small rays are cast into a scene and reflected at surfaces. This surface reflection is performed based on the surface property and can follow a mirror direction, or it can be distributed in many possible directions. The rays can also intersect many objects along the paths and transport energy to and from each surface. Multiple jittered rays are often cast into the scene for each pixel in the image. Each ray then follows a separate path, and the result is aggregated. This mimics the physical behavior of light rays discretely.

In this section, the process of ray tracing (Sec. 2.1.1) and the behavior of surfaces (BRDF in Sec. 2.1.2) is explained, and the concepts of methods introduced will be used throughout the thesis.

### 2.1.1 Rendering Equation

The rendering equation [75] describes the image formation and light propagation of a 3D scene. It is defined as:

$$L_o(\boldsymbol{x}, \boldsymbol{\omega}_o) = L_e(\boldsymbol{x}, \boldsymbol{\omega}_o) + \int_\Omega f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n}) d\boldsymbol{\omega}_i \tag{2.1}$$

Here, the outgoing light $L_o$ for a location in space $\boldsymbol{x}$ is dependent on all incoming light $L_i$ and the BRDF $f_r$ (see Sec. 2.1.2), the *cosine shading term* $(\boldsymbol{\omega}_i \cdot \boldsymbol{n})$ and the self-emission $L_e$, where $\boldsymbol{\omega}_o$ describes the outgoing light direction, $\boldsymbol{\omega}_i$ the incoming one, and $\boldsymbol{n}$ the surface normal. The light direction and normal vectors are visualized in Fig. 2.1a.

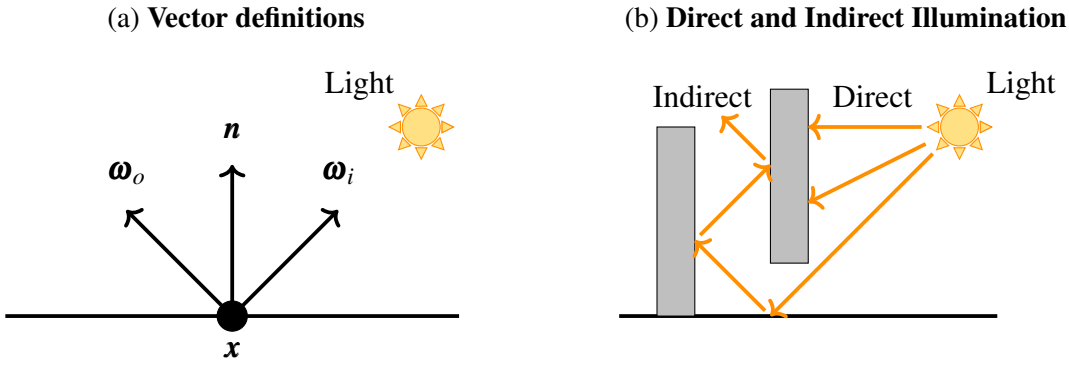



Figure 2.1: **Rendering definitions.** The definition of the surface normal, incoming, and outgoing light direction vector is shown in (a). In (b), the concept of direct and indirect illumination is visualized. In indirect illumination, the light ray bounces through the scene.

It is worth noting that the outgoing light $L_o$ of a surface point $\boldsymbol{x}$ also influences the incoming light of another point $L_i$. This recursive definition enables light to be reflected through the scene and enables indirect illumination effects. *E.g.* in Fig. 2.1b an example is shown where the light source can directly illuminate one wall but is also blocked by the same wall and indirectly lights everything behind the wall.

Additionally, the equation contains an integral over the hemisphere $\int_\Omega$, which evaluates all the incoming light. Monte-Carlo integration is often applied to render a scene, where individual rays are cast into the scene. The rays can reflect off various surfaces in the scene. A ray can either start at the camera or a light source. Both approaches have benefits and disadvantages, and hybrid methods exist, where rays from the light source and camera are traced and connected [73, 167]. For each method, energy is carried from and to each surface position. This way, indirect lighting is enabled. For example, a white surface next to a colored wall will be tinted in that color. As near-infinite directions are possible at every surface location, a single ray per pixel is not likely to produce the expected color, and the resulting image will be noisy. Therefore, often multiple rays are cast toward each pixel.

The Monte-Carlo integration then defines the expected output as the sum of random samples divided by the probability of each sample:

$$O = \int_a^b f(x)dx = \frac{1}{N}\sum_{i=0}^{N-1}\frac{f(x_i)}{\text{pdf}(x_i)} \tag{2.2}$$

where $\text{pdf}(x)$ describing the PDF of $f(x)$. If a sample is improbable, the PDF will be small. If the function $f(x)$ is at the same time unexpectedly large the influence of that sample will be large and result in artifacts called *Fireflies*. These fireflies are visible as

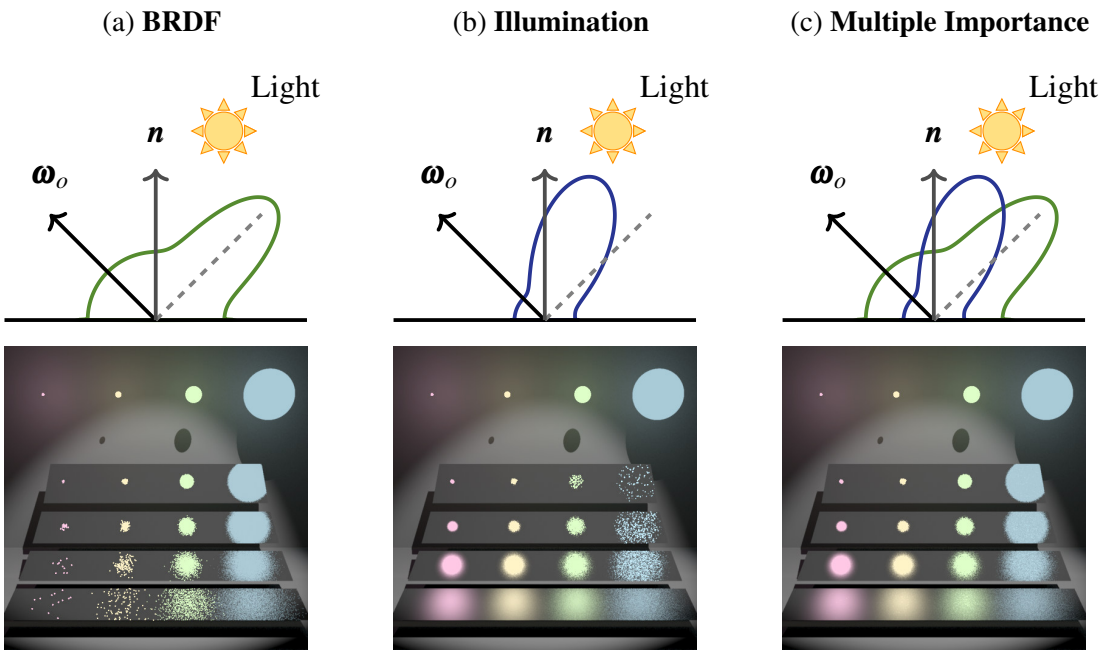



Figure 2.2: **Importance sampling.** Here, a visual representation of the different sampling methods alongside the rendered image is shown. In (a), the distribution of rays is shown if only the BRDF is considered when deciding the sampling direction. In (b), only the illumination is considered. Both methods cast a large portion of rays into improbable directions. Therefore, in (c), both distributions are sampled based on multiple importance sampling where the weighting is performed based on their Probability Density Functions (PDFs). The unlikely samples in either PDF are weighted down, and the resulting image has significantly less noise. Rendered images (bottom row) taken from Veach [167].

bright single pixels and are shown in Fig. 2.2a and Fig. 2.2b. In general the division by the PDF should account for the fewer samples placed in unlikely directions.

Sampling in more likely directions is therefore critical in rendering, and it is possible to consider the scene illumination and/or the surface properties to sample more efficiently. For example, in Fig. 2.2a the BRDF's PDF is mostly followed to distribute the ray directions. However, the light is not located in the mirror location. Most samples then never reach the light source. If instead only the illumination PDF is followed as in Fig. 2.2b, and the BRDF is nearly mirror-like most directions are unlikely. Both sampling strategies produce fireflies. An ideal solution is to consider both terms as in Fig. 2.2c and weight the individual distributions. This technique is called importance sampling or, in the joint case, multiple importance sampling [167]. Still, even with multiple importance sampling, multiple rays per pixel, and accurate knowledge about the location of light

sources and the BRDF, are required to render a non-noisy result.

#### 2.1.1.1 Approximations for inverse rendering.

In this work, rendering is used primarily in an inverse manner. *E.g.* estimating the shape, reflective properties, and illumination from images. This limits the scope of advanced rendering methods drastically, as these methods require accurate knowledge about the scene, but the scene needs to be optimized in inverse rendering. Additionally, the problem is challenging and highly ambiguous. As discussed in Sec. 2.3, simplifications of the rendering equation are often taken. One approach is to consider the illumination of the scene to be infinitely far away [24, 119]. Then the incoming light does not vary between different surface locations $\boldsymbol{x}$ and only depends on the direction $\boldsymbol{\omega}_i$. Additionally, one can ignore the light bounces and only consider the direct illumination from the infinitely far illumination. Then the incoming light is defined as $L_i(\boldsymbol{\omega}_i)$ and the rendering equation is not recursive anymore. Lastly, the self-emission term $L_e$ can be removed, resulting in the simplified rendering equation:

$$L_o(\boldsymbol{x}, \boldsymbol{\omega}_o) = \int_\Omega f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) L_i(\boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n}) d\boldsymbol{\omega}_i \tag{2.3}$$

These assumptions are taken throughout the entire thesis.

#### 2.1.1.2 Radiometry

Radiometry defines quantities, units, and techniques for measuring electromagnetic radiation. In the context of this thesis, the range will be limited to the visible light spectrum. It is closely related to *photometry* which is also limited to the visible light spectrum but factors in the human visual perception. In the context of this work, the following quantities can be easily linked to the rendering equation:

**Radiant energy** describes the energy of electromagnetic radiation with the unit: $J$ (joule)

**Radiant flux** describes the energy received per unit time $s$ with the unit: $W = \frac{J}{s}$ (watt)

**Radiance** describes the emitted, reflected, or transmitted radiant flux per unit solid angle (sr) per unit projected area $m^2$ with the unit: $W\,\mathrm{sr}^{-1}m^{-2}$

**Irradiance** describes the radiant flux received by a unit area with the unit: $\frac{W}{m^2}$

**Reflectance** describes the ratio of incoming to reflected radiant flux

In the context of the rendering equation, these quantities can then be linked as follows:

$$\underbrace{L_o(\boldsymbol{x}, \boldsymbol{\omega}_o)}_{\text{Radiance (Outgoing)}} = \int_\Omega \underbrace{f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)}_{\text{Reflectance}} \overbrace{\underbrace{L_i(\boldsymbol{\omega}_i)}_{\text{Radiance (Incoming)}} (\boldsymbol{\omega}_i \cdot \boldsymbol{n}) d\boldsymbol{\omega}_i}^{\text{Irradiance}} \tag{2.4}$$

### 2.1.2 Bidirectional Reflectance Distribution Function (BRDF)

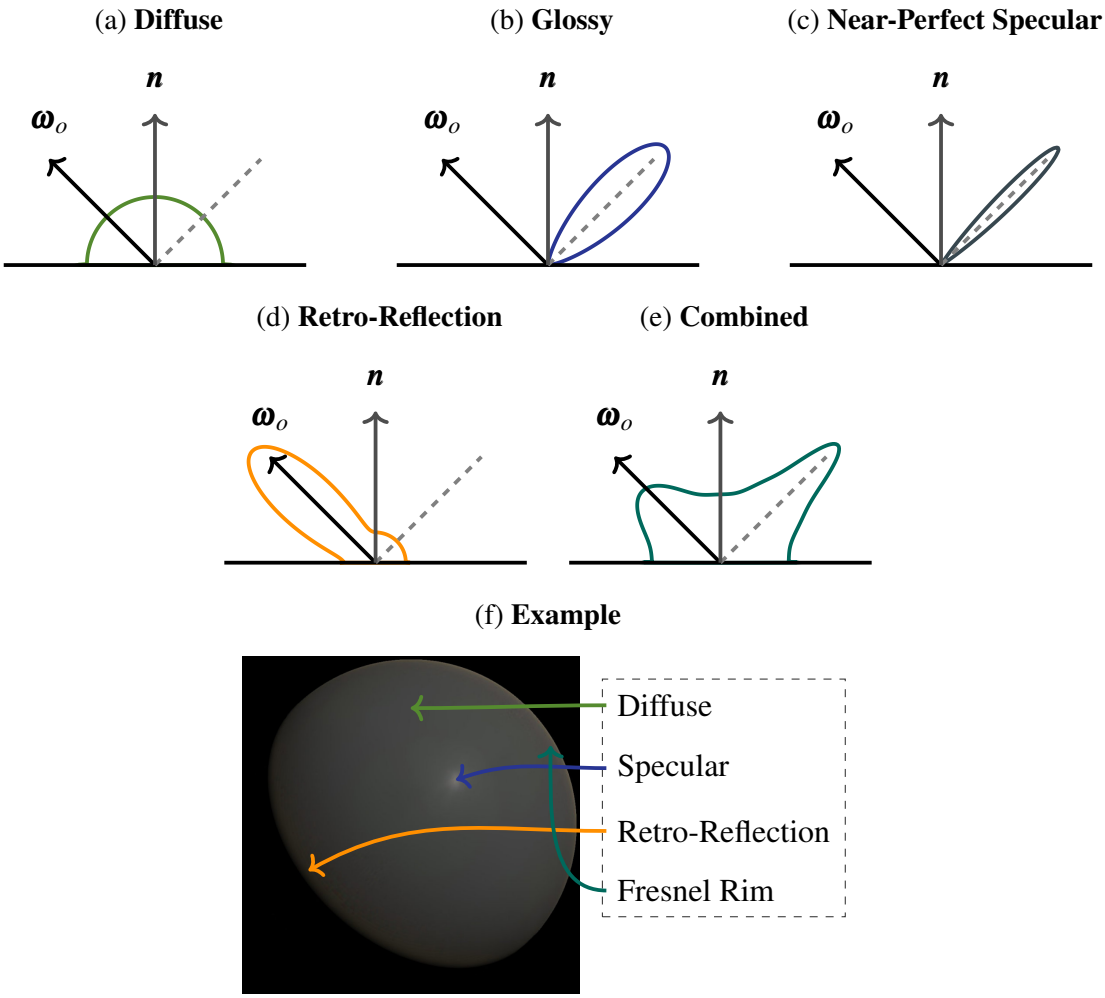



Figure 2.3: **Bidirectional Reflectance Distribution Function components.** A simple BRDF consists of multiple components. For example, in (a), a diffuse component is shown, where the scattering behavior is view direction independent. The incoming light is scattered in all directions equally. In (b) and (c), a glossy and specular BRDF is shown where the reflection mostly appears in the mirror direction (dashed line). In (d), retro-reflective behavior is visualized. In (e), a combination of all components is shown. Here, a large portion of reflected rays are cast into the mirror direction, but also some rays can be scattered in all directions. Lastly, in (f), a rendered example annotated with the components is shown.

The BRDF describes how light is reflected on a surface. This can be visualized in lobes, such as in Fig. 2.3. These lobes visualize the distribution of how light rays are reflected. For example, in Fig. 2.3a light is reflected in all directions, whereas in Fig. 2.3c

most rays are reflected in the mirror direction. In general a BRDF can be decomposed into diffuse (Fig. 2.3a), glossy (Fig. 2.3b), near-perfect specular (Fig. 2.3c) and retro-reflective (Fig. 2.3d) reflections. Additionally, most surfaces become more reflective at grazing angles due to the Fresnel reflection, as visible in Fig. 2.3f.

The BRDF is a function of the incoming $\boldsymbol{\omega}_i$ and outgoing $\boldsymbol{\omega}_o$ light angle and defined as: $f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$. One can define the incoming and outgoing light in spherical coordinates based on the surface normal $\boldsymbol{n}$. Then the BRDF is a 4-dimensional function, but it can be simplified to 3-dimensional by only assuming isotropic material. In isotropic materials, the reflective behavior remains the same while rotating around the surface normal $\boldsymbol{n}$, which results in a perfectly circular highlight. As the materials are changing over nearly every surface, the BRDF can be extended to be an SVBRDF, which extends the function $f_r$ by the surface location $\boldsymbol{x}$. As SVBRDFs can be encoded in 2D textures, the corresponding texture coordinate can be used, which adds two dimensions: $f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$.

For physical correctness, the BRDF also has to follow certain properties [52]. The BRDF is always positive: $f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) \geq 0$. It needs to follow the Helmholtz reciprocity, which means the behavior remains the same if the incoming and outgoing light directions are reversed: $f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o) = f_r(\boldsymbol{\omega}_o, \boldsymbol{\omega}_i)$. And lastly it has to be energy conserving, as a surface can not reflect more light than it has received: $\forall \boldsymbol{\omega}_i, \int_\Omega f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o)(\boldsymbol{\omega}_o \cdot \boldsymbol{n}) d\boldsymbol{\omega}_o \leq 1$.

There are two approaches to representing the BRDF. One is to measure the surface behavior under known light and view directions [51, 112, 180]. This measurement needs to be taken for every point. Due to the dense sampling required per point, most datasets only capture homogeneous materials. However, there also exists spatially-varying measured materials [180]. This *measured BRDF* can then be used as a lookup of the specific response for the given light and view direction during rendering. The other approach is to approximate the surface response using a parametric function. These models are then called analytical BRDF.

#### 2.1.2.1 Measured BRDF

The most accurate method to capture this reflective behavior is by measuring a surface for every incoming $\boldsymbol{\omega}_i$ and outgoing $\boldsymbol{\omega}_o$ light direction. In general, two approaches are used for measurements [51, 112, 113, 181]. One with a Gonioreflectometer such as in Fig. 2.4a and the other one being image-based such as in Fig. 2.4b. The main difference between the methods is a trade-off between speed and accuracy. In comparison, a Gonioreflectometer provides the highest level of accuracy but requires a long capture time, as multiple moving parts need to capture all incoming and outgoing angles. On the other hand, in the Merl Dataset of Matusik *et al*. [112], a spherical sample is used as shown in Fig. 2.4b. When a single image is taken, multiple viewing angles are captured. Therefore, the light source only rotates around the sample on a single axis to capture isotropic materials. Samples from this capture process are shown in Fig. 2.5.

Still, these methods only capture homogeneous materials. If a spatially-varying ma-

(a) **White *et al*. [181]** (b) **Matusik *et al*. [112]** (c) **McAllister [113]**

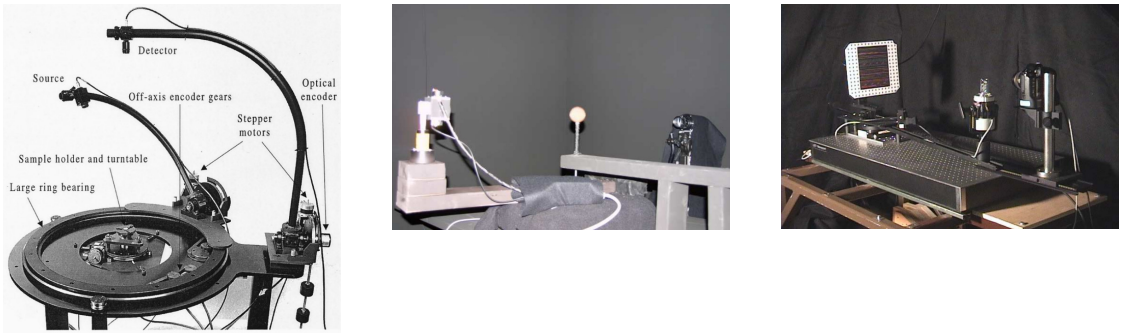



Figure 2.4: **BRDF measurement setups.** The setup in (a) shows a Gonioreflectometer setup, where a moveable, high-quality light source can be placed in all positions on the upper hemisphere. The detector can pivot on a single axis; therefore, three dimensions are captured, which is enough for isotropic BRDFs. In (b), the image acquisition setup from the Merl Database [112] is shown, which is image-based. The capture is sped up by applying the material to a sphere. Therefore, multiple light directions are captured in each image, and the light source only needs to rotate around one axis. Lastly, in (c), a setup for capturing spatially-varying, near-planar samples is shown, where the sample tray can also be moved. Images taken from respective works.

(a) **Bronze Aluminium** (b) **Dark Red Paint** (c) **Aluminium**

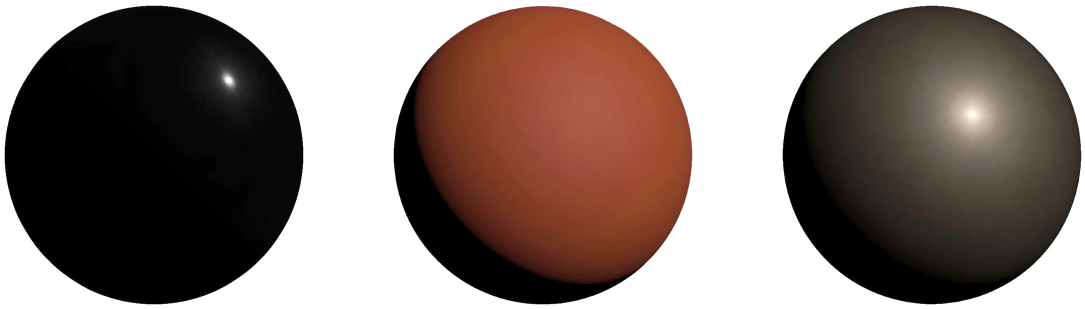


Figure 2.5: **Merl BRDF examples.** Examples of measured BRDFs from the Merl Dataset [112].

terial is captured, the sample tray itself needs to be moved, and for every point, the behavior of incoming and outgoing light must be recorded. Each material now requires storing multiple incoming and outgoing light directions per pixel. This results in quite large file sizes.

#### 2.1.2.2 Analytical BRDFs

As mentioned in Sec. 2.1.2.1, measured BRDFs are quite large in size and are therefore seldom used in actual productions. Instead, analytical models are used. These models are parametric functions that recreate the behavior of the captured materials. Usually, they consist of a color for the diffuse reflections, a scalar that models how reflective the material is — often called the roughness or glossiness —, and a potential specular reflection color, which expresses the tinting of highlights as often seen in metals. *I.e.* gold tints the specular highlight in a yellow to amber color. Compared to gold-colored plastics, the highlight will have the same color as the light source. Several of these models exist, which vary in accuracy and materials that can be expressed. In the following, some of these models are discussed.

**Lambertian Diffuse Model** is one of the simplest BRDF models and can only express materials that scatter light in all directions equally [85]. This is a strong assumption, and only a few materials roughly behave this way. *I.e.* wall paints often are roughly Lambertian. The BRDF is then defines as:

$$f_{\text{Lambert}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_d) = \frac{\boldsymbol{b}_d}{\pi} \tag{2.5}$$

Where $\boldsymbol{b}_d \in \mathbb{R}^3$ is the diffuse color, and the division by $\pi$ is done to ensure energy conservation as the outgoing radiance is integrated over the hemisphere.

**Cook-Torrance Model** is a physically-based, specular BRDF model [41]. It is modular by design, and most often, it models specular reflections realistically using microfacets. In this concept, a surface consists of microscopic irregularities, where each point behaves like a perfect mirror, as shown in Fig. 2.6a. Therefore, a secondary normal, the microfacet normal $\boldsymbol{m}$ is introduced. This normal does only exists conceptually and is instead modeled by a roughness parameter $\boldsymbol{b}_r$ and then defines the ratio of microfacet normals $\boldsymbol{m}$ being aligned with the surface normal $\boldsymbol{n}$. For a smooth surface, the roughness $\boldsymbol{b}_r$ is close to 0, and the majority of the microfacet normals $\boldsymbol{m}$ are aligned with the surface normal $\boldsymbol{n}$. The result is a sharp reflection. If the surface is rough, the roughness $\boldsymbol{b}_r$ is close to 1, and the microfacet normals $\boldsymbol{m}$ are scattered around the surface normal $\boldsymbol{n}$. The light is then reflected in random directions, which results in a blurry reflection.

Explicitly altering the geometry to accurately represent these micro structures is not feasible and, therefore this concept is approximated by a normal distribution function $D(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n})$. As the Cook-Torrance model is modular, varying normal distribution

(a) **Microfacet definition** (b) **Microfacet shadowing**


Figure 2.6: **Microfacets.** In (a), the microscopic surface irregularities are shown. In a microfacet model, a surface consists of tiny mirror-like patches defined by the microfacet normal $\boldsymbol{m}$. Furthermore, light can get absorbed due to the surface structure, as shown in (b). This is either due to self-occlusions or self-shadowing, where the structures block the reflected light or the incoming light casts a shadow.

functions can approximate this function. Often the GGX distribution [170] is used:

$$D(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n}) = \frac{\alpha}{\pi((\alpha-1)(\boldsymbol{n}\cdot\boldsymbol{h})^2+1)^2} \tag{2.6}$$

$$\alpha = {\boldsymbol{b}_r}^2 \tag{2.7}$$

$$\boldsymbol{h} = \frac{\boldsymbol{\omega}_i + \boldsymbol{\omega}_o}{\|\boldsymbol{\omega}_i + \boldsymbol{\omega}_o\|} \tag{2.8}$$

Due to the surface irregularities, light can also be blocked, and the surface should darken accordingly. An example is shown in Fig. 2.6b. It is again not feasible to model these irregularities and, therefore, this behavior is modelled by a geometric attenuation function $A(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n})$. As light can get blocked in the view or light directions by these irregularities, it is defined as:

$$A(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n}) = a(\boldsymbol{n}\cdot\boldsymbol{\omega}_i)a(\boldsymbol{n}\cdot\boldsymbol{\omega}_o) \tag{2.9}$$

$$a(d) = \frac{2d}{d + \sqrt{{\boldsymbol{b}_r}^+(1-\boldsymbol{b}_r)d^2}}$$

Lastly, as this is a specular model, the Fresnel effect must be accounted for. This term is defined as $F(\boldsymbol{\omega}_o, \boldsymbol{h})$. The common Schlick approximation [144] can be used to model this:

$$F(\boldsymbol{\omega}_o, \boldsymbol{h}; R_0) = R_0 + (1-R_0)(1-\boldsymbol{\omega}_o\cdot\boldsymbol{h})^5 \tag{2.10}$$

Here, $R_0$ describes the reflection coefficient for light incoming parallel to the surface normal $\boldsymbol{n}$. This $R_0$ coefficient is modelled by the indices of refraction of the two media $\eta_1$ and $\eta_2$ at the intersection point $\boldsymbol{x}$. It is then defined as: $R_0 = \left(\frac{\eta_1-\eta_2}{\eta_1+\eta_2}\right)^2$. For easier artistic control it can also be defined by an RGB value $\boldsymbol{b}_s$, where the first medium $\eta_1$ is always assumed to be air, which has an index of refraction of close to 1.

The final specular formulation is then composed by the three functions $D$, $F$, and $A$ and normalized to ensure energy conservation:

$$f_{\text{Cook-Torrance}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_s, \boldsymbol{b}_r, \boldsymbol{n}) = \frac{D(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n}) F(\boldsymbol{\omega}_o, \boldsymbol{h}; \boldsymbol{b}_s) A(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n})}{4(\boldsymbol{\omega}_o \cdot \boldsymbol{n})(\boldsymbol{\omega}_i \cdot \boldsymbol{n})} \tag{2.11}$$

As this model by default does only model glossy reflection, it can be combined with the Lambertian reflection to model two lobes of the BRDF.

$$f_r(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_d, \boldsymbol{b}_s, \boldsymbol{b}_r, \boldsymbol{n}) = f_{\text{Lambert}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_d) + f_{\text{Cook-Torrance}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_s, \boldsymbol{b}_r, \boldsymbol{n}) \tag{2.12}$$

Instead of storing the specular and diffuse color as two individual values an alternative formulation from Burley [32] stores both colors in a single base-color $\boldsymbol{b}_b$ and interpolates the specular and diffuse color based on a metallic scalar $b_m$ as followed:

$$\begin{aligned} \boldsymbol{b}_d &= (1 - b_m)\boldsymbol{b}_b \\ \boldsymbol{b}_s &= (1 - b_m)0.04 + b_m \boldsymbol{b}_b \end{aligned} \tag{2.13}$$

A $b_m$ value near 1 results in a fully metallic material and near 0 in non-metal. The 0.04 is an often-used value to explain most specular behaviors of non-metals.

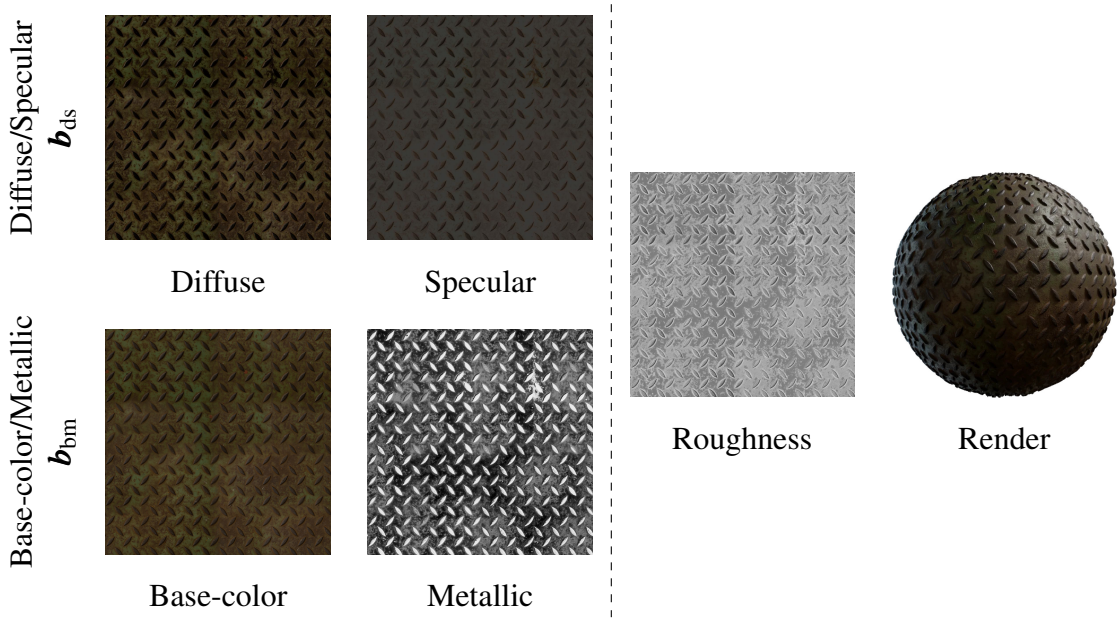



Figure 2.7: **Material Maps.** The material maps for a metal plate are shown for the Diffuse/Specular and Base-color/Metallic parametrization. The roughness parameter is shared for both formulations.

When the RGB and scalar coefficients $\boldsymbol{b}_d$, $\boldsymbol{b}_s$, and $\boldsymbol{b}_r$ or the equivalent $\boldsymbol{b}_b$, $b_m$, and

$\boldsymbol{b}_r$ are used the parameters can be stored easily in image file formats as material maps to build a SVBRDF. Sample material maps are shown in Fig. 2.7. We further define all parameters for the diffuse/specular and base-color/metallic model as $\boldsymbol{b}_{\text{ds}}$ and $\boldsymbol{b}_{\text{bm}}$, respectively.

## 2.2 Illumination Models

The rendering equation defined in Eq. (2.1) models the outgoing light of a singular point based on all incoming illumination. This problem is trivial to solve for a singular, known point light source without any indirect light, as the only light direction is known, and only a visibility check is required. However, casting a ray in every direction is infeasible for more complex illuminations where a point is illuminated from the entire environment. Here, the Monte-Carlo integration is often used to approximate the integration. This process is often too costly in real-time applications or differentiable optimization, and approximations are used. In Eq. (2.3), a simplified model is proposed, where the illumination is infinitely far away [24, 119]. With this simplified model, the illumination approximation of this section is used to speed up the rendering process drastically.

### 2.2.1 Spherical Gaussian Illumination

A regular 1D Gaussian is defined as: $ae^{\frac{-(x-b)^2}{2c^2}}$. Where $a \in \mathbb{R}$ defines the height or the amplitude, $b \in \mathbb{R}$ the central position of the peak, and $c \in \mathbb{R}_{>0}$ the width of the bell shape. An exemplary Gaussian is shown in Fig. 2.8a. This Gaussian is defined in a Cartesian coordinate system. However, the Gaussians can also be defined in a polar coordinate system, as shown in Fig. 2.8b. It is then defined as $ae^{c(\cos(\theta-b)-1)}$, where $\theta$ is now the coordinate to evaluate. It is worth noting that the lobe in Fig. 2.8b roughly resembles a glossy BRDF lobe as shown in Fig. 2.3b. This observation is the central concept behind this illumination model proposed by Wang *et al.* [171], where SG are used to express both the BRDF and the illumination.

A SG can express the three-dimensional BRDF and illumination. It is defined as:

$$G(\boldsymbol{x};\boldsymbol{\mu},\lambda,a) = ae^{\lambda(\boldsymbol{\mu}\cdot\boldsymbol{x}-1)} \tag{2.14}$$

The parameter $\boldsymbol{\mu} \in \mathbb{R}^3$ describes the *direction* of the peak of the lobe, the $\lambda \in \mathbb{R}_{>0}$ describes the *sharpness* or standard deviation of the lobe and $a \in \mathbb{R}$ the *amplitude*, which can also be extended to $\boldsymbol{a} \in \mathbb{R}^3$ to express even RGB colors. SG possesses several beneficial properties to express illuminations.

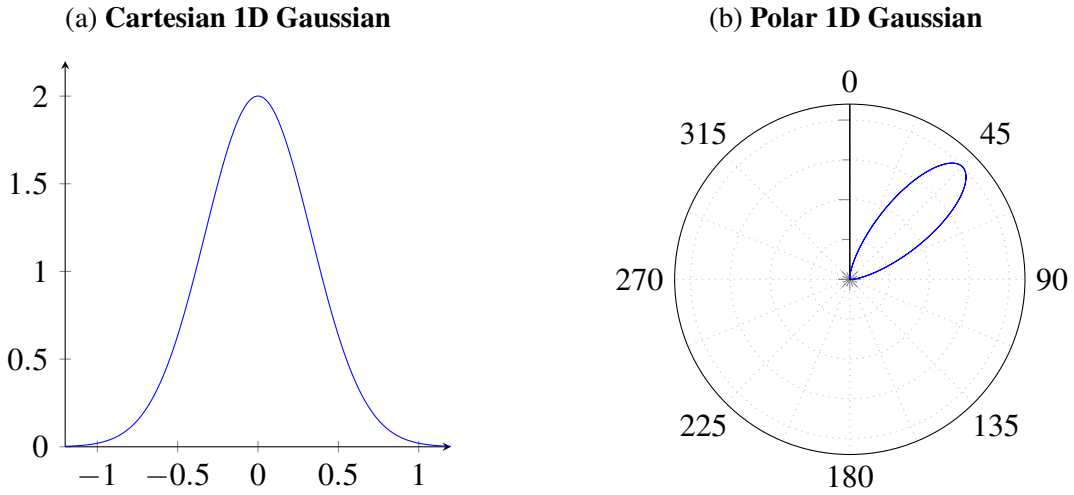



Figure 2.8: **One-dimensional Gaussians.** In (a), a Gaussian in a Cartesian coordinate system is shown. In (b), a Gaussian is shown in a polar coordinate system.

The product of two SGs is an SG. The product is defined as [171]:

$$
\begin{aligned}
G_1(\boldsymbol{x})G_2(\boldsymbol{x}) &= G(\boldsymbol{x}; \frac{\boldsymbol{\mu}_m}{\|\boldsymbol{\mu}_m\|}, \lambda_m\|\boldsymbol{\mu}_m\|, a_1 a_2 e^{\lambda_m(\|\boldsymbol{\mu}_m\|-1)}) \\
\lambda_m &= \lambda_1 + \lambda_2 \\
\boldsymbol{\mu}_m &= \frac{\lambda_1\boldsymbol{\mu}_1 + \lambda_2\boldsymbol{\mu}_2}{\lambda_1 + \lambda_2}
\end{aligned}
\tag{2.15}
$$

The integral of a Gaussian has a closed-form solution. This function is known as the error function. This also holds for SGs where the integral is taken over the entire sphere [161]:

$$
\int_\Omega G(\boldsymbol{x})d\boldsymbol{x} = 2\pi\frac{a}{\lambda}(1 - e^{-2\lambda}) \tag{2.16}
$$

This property can also be used to normalize an SG, as this can ensure that the SG integrates to 1. A normalized SG is equivalent to a von Mises-Fisher distribution in 3D.

Lastly, the inner product of two SGs is the integral of the product of two SGs. The operation is defined as [161]:

$$
\begin{aligned}
G_1(\boldsymbol{x}) \cdot G_2(\boldsymbol{x}) = \int_\Omega G_1(\boldsymbol{x})G_2(\boldsymbol{x})d\boldsymbol{x} &= {}^1/_{d_m}\left(2\pi a_1 a_2 e^{d_m - \lambda_m}(1.0 - e^{-2d_m})\right) \\
\lambda_m &= \lambda_1 - \lambda_2 \\
d_m &= \|\lambda_1\boldsymbol{\mu}_1 + \lambda_2\boldsymbol{\mu}_2\|
\end{aligned}
\tag{2.17}
$$

Especially the property that the inner product is the integral of two SGs is helpful if the illumination is approximated as an SG or a set of SGs and the reflective behavior of

the BRDF is also represented as an SG. The illumination can be naturally expressed as an SG, as the amplitude covers the illumination strength, the sharpness defines the size of the specific light source, and the direction describes the position of the infinitely far light source. On the other hand, a BRDF consists of multiple lobes, which are parametrized by diffuse, specular, and roughness.

(a) **Cosine Lobe** (b) **Evaluated SG illumiantion**

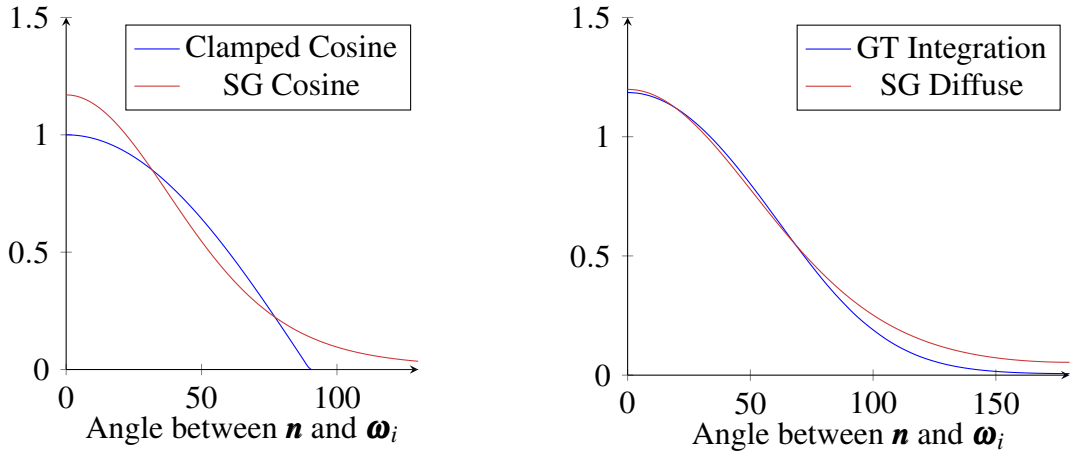



Figure 2.9: **Simple SG diffuse illumination fit.** In (a), the ground truth clamped cosine lobe is compared with the corresponding SG lobe. In (b), the effect of this approximation is shown. Here, a SG light is evaluated with the cosine lobe and compared to the ground truth numerical integration.

**Diffuse Illumination** The diffuse illumination is only parametrized by the diffuse color $\boldsymbol{b}_d$, and the Lambertian reflection model is defined in Eq. (2.5). Here, it is worth noting that the BRDF is a scalar. Therefore, the diffuse SG lobe mainly needs to approximate the cosine shading term. Wang *et al.* proposes a SG parametrized with amplitude $a = 1.17$ and sharpness $\lambda = 2.133$ [171]:

$$G_{\text{Cosine}}(\boldsymbol{n}, 2.133, 1.17) \tag{2.18}$$

This lobe is shown in Fig. 2.9a. It is worth noting that the lobe mostly follows the actual clamped cosine curve nicely, but the amplitude extends beyond 1 and also never reaches 0. This approximation affects the final rendered result, especially visible in areas facing away from the light source. Due to the cosine lobe never reaching 0, these areas are still dimly lit, and the error compared to ground-truth numerical integration can be seen in Fig. 2.9b. The final outgoing light per SG light source $G_i$ is then calculated as:

$$K_{\text{Diffuse}}(G_i) = \frac{\boldsymbol{b}_d}{\pi} \max\left(G_i \cdot G_{\text{Cosine}}, 0\right) \tag{2.19}$$

**Specular Illumination** For the specular part of the BRDF, the SG can also be fitted to the Cook-Torrance BRDF. This model consists of 3 parts: The normal distribution function $D$, the geometric self-attenuation $A$, and the Fresnel term $F$.

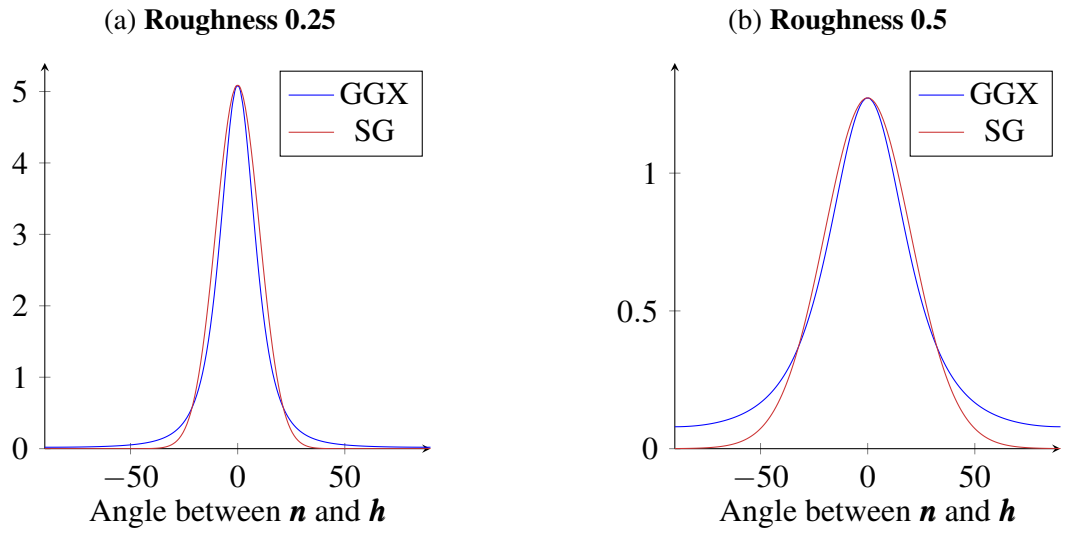



Figure 2.10: **Normal Distribution Function SG fit.** In (a), a relatively smooth surface is shown. The approximation follows the ground truth GGX function closely. In (b), a rougher surface is shown. Here, the approximation has a reduced accuracy if the half-vector $\boldsymbol{h}$ is not aligned with the normal $\boldsymbol{n}$ direction.

The normal distribution function has a close fit proposed by Wang *et al.* [171] with a Gaussian defined as:

$$D(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n}) = G_{\text{ndf}}(\boldsymbol{h}; \boldsymbol{n}, \frac{2}{{\boldsymbol{b}_r}^2}, \frac{1}{\pi {\boldsymbol{b}_r}^2}) \tag{2.20}$$

with $\boldsymbol{h}$ being defined in Eq. (2.8). The resulting fit for two roughness values is shown in Fig. 2.10. As seen, the SG amplitude also scales appropriately with the GGX function due to the scaling term $\frac{1}{\pi {\boldsymbol{b}_r}^2}$, which always ensures an amplitude of 1 over the integral.

However, the SG is now defined in the domain of the half vector $\boldsymbol{h}$ with the axis or direction of the SG pointing towards the normal $\boldsymbol{n}$. As the viewing angle shifts, so does the half vector $\boldsymbol{h}$. This view dependency must be introduced by warping the fitted normal distribution function to the light source domain. Wang *et al.* [171] propose the following warping operator applied to the normal distribution function fitted SG $G_{\text{ndf}}$:

$$
\begin{aligned}
G_w &= G(\boldsymbol{\mu}_w, \lambda_w, a_w) \\
\boldsymbol{\mu}_w &= 2\,(\boldsymbol{\omega}_o \cdot \boldsymbol{\mu}_{\text{ndf}})\,\boldsymbol{\mu}_{\text{ndf}} - \boldsymbol{\omega}_o \\
\lambda_w &= \frac{\lambda_{\text{ndf}}}{4\|\boldsymbol{\mu}_{\text{ndf}}\boldsymbol{\omega}_o\|} \\
a_w &= a_{\text{ndf}}
\end{aligned}
\tag{2.21}
$$

The remaining terms of the Cook-Torrance model, the geometric attenuation $A$ and the Fresnel term $F$, are not easily fitted to a SG. Here, Wang *et al*. [171] apply an assumption that these terms are constant over the entire lobe and can, therefore, be removed from the integration over the lobe. This assumption works well for smoother surfaces with a low roughness value but has reduced accuracy for rougher surfaces.

$$
K_{\text{Specular}}(G_i) = \frac{(G_w \cdot G_i)\,F(\boldsymbol{\omega}_o, \boldsymbol{\mu}_w; \boldsymbol{b}_s)A(\boldsymbol{b}_r, \boldsymbol{\mu}_w, \boldsymbol{\omega}_o, \boldsymbol{n})}{4(\boldsymbol{\mu}_w \cdot \boldsymbol{n})(\boldsymbol{\omega}_o \cdot \boldsymbol{n})}
\tag{2.22}
$$

**Joint Illumination** As light is additive, the final evaluation is then defined as:

$$
L_o(G_i) = K_{\text{Diffuse}}(G_i) + K_{\text{Specular}}(G_i)
\tag{2.23}
$$

Similarly, if multiple SG light sources are present, each light source is independently evaluated and then added:

$$
L_o = \sum_{G_i} K_{\text{Diffuse}}(G_i) + K_{\text{Specular}}(G_i)
\tag{2.24}
$$

### 2.2.2 Pre-integrated Illumination

While SG illumination models can be used to model ground truth illuminations and allow easy integration of all incoming light, the final rendering quality might be far off compared to ground truth Monte-Carlo integration. This is especially visible in high-frequency illumination patterns or highly specular materials. This problem is also well-known in real-time rendering. In Sec. 2.1.2.2, the Cook-Torrance model and the Lambertian BRDF are shown. These BRDFs split the rendering equation in a diffuse and specular formulation:

$$
L_o(\boldsymbol{x}, \boldsymbol{\omega}_o) = \underbrace{\frac{\boldsymbol{b}_d}{\pi}\int_\Omega L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\,d\boldsymbol{\omega}_i}_{\text{diffuse}} + \underbrace{\int_\Omega f_s(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_s, \boldsymbol{b}_r) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\,d\boldsymbol{\omega}_i}_{\text{specular}}
\tag{2.25}
$$

The central concept of pre-integrated rendering is to pre-compute some of these integrals for a given environment illumination and only perform minimal calculations during rendering [78]. The environment maps are often convolved or blurred, corresponding to the BRDF. For the diffuse materials, the entire hemisphere of incoming light needs to be integrated, as shown in Fig. 2.3a.

(a) **Environment Map** (b) **Diffuse Pre-Integration**

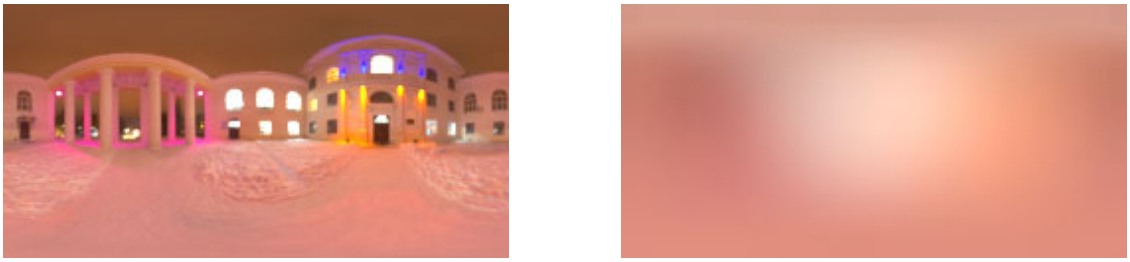


Figure 2.11: **Diffuse Pre-Integration.** In (a), the regular environment map is shown. Whereas (b) visualizes the pre-integrated diffuse illumination. The general structures of the environment still appear but are heavily blurred.

Each pixel corresponds to a latitude and longitude direction in an environment map such as Fig. 2.11a. This direction can also be thought of as the normal $\boldsymbol{n}$ of a surface. The hemisphere for this direction needs to be numerically integrated while considering the cosine term $\boldsymbol{\omega}_i \cdot \boldsymbol{n}$. The result is a severely blurred version of the environment illumination, such as in Fig. 2.11b. The final evaluation is then defined as follows:

$$k_{\text{Diffuse}} = \frac{\boldsymbol{b}_d}{\pi} \hat{L}_{\text{Diffuse}}(\boldsymbol{n}) \tag{2.26}$$

$$\hat{L}_{\text{Diffuse}}(\boldsymbol{n}) = \int_{\Omega} L_i(\boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n}) d\boldsymbol{\omega}_i \tag{2.27}$$

For the specular portion of the rendering equation, the dependence on roughness and the view direction needs to be minded. A mirror-like object achieves a sharp reflection, whereas a highly brushed metal object nearly behaves like a diffuse surface. Compared to the diffuse of Eq. (2.27), the specular pre-integration is now also dependent on the view direction and the roughness. However, similarly integrating the environment map for every roughness value, view, and normal direction is not feasible.

Here, Karis *et al*. [77] provide a reasonable approximation for the Cook-Torrance model defined in Eq. (2.11) of Sec. 2.1.2.2. The environment map lookup should only depend on the incoming illumination direction and the roughness value. However, due to the Fresnel effect, specular reflections are also view-dependent. Karis *et al*. propose a strong assumption that the view direction $\boldsymbol{\omega}_o$ is the same as the surface normal $\boldsymbol{n}$: $\boldsymbol{\omega}_o = \boldsymbol{n}$. The effect of this approximation is visible primarily at grazing angles and is visualized in Fig. 2.12.

(a) **"Correct" view direction** (b) $\boldsymbol{\omega}_o = \boldsymbol{\omega}_i = \boldsymbol{n}$ **view direction**

Figure 2.12: **View direction approximation.** In (a), the ground truth behavior of a specular reflection at a grazing angle is shown. In (b), the assumption of a similar view, light, and normal direction is shown. Notice the lack of stretched highlights. Image is taken from the Filament Documentation [63].

With this assumption, the environment map can be pre-integrated for varying levels of surface roughness. As shown in Eq. (2.2), some samples contribute less if they are highly unlikely. Therefore, the samples of the environment map should be distributed according to the distribution. Here, the GGX distribution $D(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n})$ defined in Eq. (2.7) is used, as it defines the scattering behavior of the individual reflected rays. Given two random variables from a uniform distribution $u_1$ and $u_2$, the half-vector $\boldsymbol{h}$ is then sampled with [77]:

$$
\begin{aligned}
\boldsymbol{s}(u_1, u_2, \boldsymbol{n}, \boldsymbol{b}_r) &= [\boldsymbol{t}\boldsymbol{r}_x + \boldsymbol{b}\boldsymbol{r}_y + \boldsymbol{n}\boldsymbol{r}_z]^T \\
\boldsymbol{t} &= \frac{[0,1,0]^T \times \boldsymbol{n}}{\|[0,1,0]^T \times \boldsymbol{n}\|} \\
\boldsymbol{b} &= \frac{\boldsymbol{n} \times \boldsymbol{t}}{\|\boldsymbol{n} \times \boldsymbol{t}\|} \\
\boldsymbol{r} &= [\cos(\theta)\sin(\phi), \sin(\theta)\sin(\phi), \cos(\phi)]^T \\
\theta &= 2\pi u_1 \\
\cos(\phi) &= \sqrt{\frac{1-u_2}{1+(\boldsymbol{b}_r{}^2 - 1)u_2}} \\
\sin(\phi) &= \sqrt{1-\cos(\phi)^2}
\end{aligned}
\tag{2.28}
$$

where $(\theta, \phi)$ define the spherical coordinates of $\boldsymbol{h}$. Here, $\theta \in [0, 2\pi)$ defines the azimuth and $\phi \in [0, \pi]$ defines the elevation.

The environment map is then integrated for $l$ different levels, and the final roughness value is then linearly interpolated between two adjacent levels. Furthermore, an image pyramid can be leveraged as the environment map will be increasingly blurred with each

consecutive level. The environment is then down-scaled by half in each level to ease the computation requirements for the increased integration area. Often $l = 5$ is used to distribute the roughness as $0, 0.25, 0.5, 0.75, 1$. The integrated illumination is now referred to as $\hat{L}_{\text{Specular}}(\boldsymbol{\omega}, \boldsymbol{b}_r)$. Here, $\boldsymbol{\omega}$ defines the sampling direction.

The specular incoming illumination is now pre-integrated. The specular term of the split equation in Eq. (2.25) can now be separated further [77]:

$$L_o^{\text{Specular}}(\boldsymbol{x}, \boldsymbol{\omega}_o) = \int_\Omega f_s(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_s, \boldsymbol{b}_r) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, d\boldsymbol{\omega}_i \tag{2.29}$$

$$= \underbrace{\int_\Omega L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) D(\boldsymbol{b}_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o, \boldsymbol{n})(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, d\boldsymbol{\omega}_i}_{\text{Incoming Illumination}} *$$

$$\underbrace{\int_\Omega f_s(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_s, \boldsymbol{b}_r)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, d\boldsymbol{\omega}_i}_{\text{BRDF and Cosine Term}} \tag{2.30}$$

The incoming illumination $L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)$ is now approximated by the pre-integrated environment map $\hat{L}_{\text{Specular}}(\boldsymbol{\omega}, \boldsymbol{b}_r)$. Here, the illumination is convolved with the GGX normal distribution function to account for the most likely sampling areas. Notice that the implementation of this does not integrate over the entire hemisphere but only performs a Monte Carlo integration based on the GGX importance sampling. Due to the view direction approximation, the normal $\boldsymbol{n}$ is expressed by $\boldsymbol{\omega}_o$. The specular portion is now expressed as:

$$L_o^{\text{Specular}}(\boldsymbol{x}, \boldsymbol{\omega}_o) \approx \hat{L}_{\text{Specular}}(\boldsymbol{\omega}, \boldsymbol{b}_r) \int_\Omega f_s(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_s, \boldsymbol{b}_r)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, d\boldsymbol{\omega}_i \tag{2.31}$$

The left part of the equation is pre-integrated based on the different roughness levels, and the correct incoming light can be gathered via a lookup. The right side still requires a complex integration. Karis [77] proposes to perform the pre-integration and store the integrals as a lookup table indexed by $(\boldsymbol{\omega}_o \cdot \boldsymbol{n})$ and $\boldsymbol{b}_r$. As the right-hand side is dependent on more parameters, Karis performs a reordering of the equation alongside further approximations. The first step is to extract the Fresnel term from the BRDF:

$$\int_\Omega f_s(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_s, \boldsymbol{b}_r)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, d\boldsymbol{\omega}_i \approx \int_\Omega \frac{f_s(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_r)}{F(\boldsymbol{\omega}_o, \boldsymbol{h}; \boldsymbol{b}_s)} F(\boldsymbol{\omega}_o, \boldsymbol{h}; \boldsymbol{b}_s)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, d\boldsymbol{\omega}_i \tag{2.32}$$

The term is then substituted with the Schlick approximation from Eq. (2.10):

$$\int_\Omega \frac{f_s(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}_r)}{F(\boldsymbol{\omega}_o, \boldsymbol{h}; \boldsymbol{b}_s)} (\boldsymbol{b}_s + (1 - \boldsymbol{b}_s)(1 - \boldsymbol{\omega}_o \cdot \boldsymbol{h})^5)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, d\boldsymbol{\omega}_i \tag{2.33}$$

The equation is then split over two integrals [77]:

$$\int_\Omega \frac{f_s(\boldsymbol{x},\boldsymbol{\omega}_i,\boldsymbol{\omega}_o;\boldsymbol{b}_r)}{F(\boldsymbol{\omega}_o,\boldsymbol{h};\boldsymbol{b}_s)}(\boldsymbol{b}_s(1-(1-\boldsymbol{\omega}_o\cdot\boldsymbol{h})^5))(\boldsymbol{\omega}_i\cdot\boldsymbol{n})\,d\boldsymbol{\omega}_i + \int_\Omega f_s(\boldsymbol{x},\boldsymbol{\omega}_i,\boldsymbol{\omega}_o;\boldsymbol{b}_r)(1-\boldsymbol{\omega}_o\cdot\boldsymbol{h})^5(\boldsymbol{\omega}_i\cdot\boldsymbol{n})\,d\boldsymbol{\omega}_i \quad (2.34)$$

As the specular color $\boldsymbol{b}_s$ is constant over the integral, it can be taken out of the integral. The two different terms highlighted as $B_0$ and $B_1$ now act as a scale and bias for the specular color $\boldsymbol{b}_s$. These can be seen as the integration of the BRDF for a constant environment illumination of 1:

$$\boldsymbol{b}_s \underbrace{\int_\Omega f_x^*(\boldsymbol{x},\boldsymbol{\omega}_i,\boldsymbol{\omega}_o;\boldsymbol{b}_r)(1-(1-\boldsymbol{\omega}_o\cdot\boldsymbol{h})^5)(\boldsymbol{\omega}_i\cdot\boldsymbol{n})\,d\boldsymbol{\omega}_i}_{B_0} + \underbrace{\int_\Omega f_s^*(\boldsymbol{x},\boldsymbol{\omega}_i,\boldsymbol{\omega}_o;\boldsymbol{b}_r)(1-\boldsymbol{\omega}_o\cdot\boldsymbol{h})^5(\boldsymbol{\omega}_i\cdot\boldsymbol{n})\,d\boldsymbol{\omega}_i}_{B_1} \quad (2.35)$$

Here, it is worth noting that $f_s$ regularly contains a Fresnel term as seen in Sec. 2.1.2.2. This term is left out as it would be canceled out, resulting in the modified specular BRDF $f_s^*$. The specular term also includes the normal distribution function, and in the implementation, we perform a Monte Carlo integration with the GGX importance sampling.

Both parts can also be pre-integrated with the importance sampling defined in Eq. (2.29) and stored in a lookup table, parametrized by $(\boldsymbol{\omega}_o\cdot\boldsymbol{n})$ and $\boldsymbol{b}_r$. The resulting values can be stored as an image in the red and green channels, respectively. The pre-integrated BRDF lookup table is shown in Fig. 2.13.

The joined diffuse and specular pre-integration is then defined as:

$$L_o(\boldsymbol{x},\boldsymbol{\omega}_o) \approx \underbrace{(\frac{\boldsymbol{b}_d}{\pi})\hat{L}_{\text{Diffuse}}(\boldsymbol{n})}_{\text{Diffuse}} + \underbrace{(F(\boldsymbol{\omega}_o,\boldsymbol{n};\boldsymbol{b}_s)B_0(\boldsymbol{\omega}_r\cdot\boldsymbol{n},\boldsymbol{b}_r)+B_1(\boldsymbol{\omega}_r\cdot\boldsymbol{n},\boldsymbol{b}_r))\hat{L}_{\text{Specular}}(\boldsymbol{\omega}_r,\boldsymbol{b}_r)}_{\text{Specular}} \quad (2.36)$$

The incoming illumination is then queried with the reflected ray of $\boldsymbol{\omega}_o$:

$$\boldsymbol{\omega}_r = -\boldsymbol{\omega}_o - 2(-\boldsymbol{\omega}_o\cdot\boldsymbol{n})\boldsymbol{n} \quad (2.37)$$

The specular pre-integrated environment $\hat{L}_{\text{Specular}}(\boldsymbol{\omega},\boldsymbol{b}_r)$ and the pre-integrated BRDF lookup table is then queried in the reflected view direction $\boldsymbol{\omega}_r$.

The entire evaluation is now expressed as a simple set of additions and multiplications. Also, notice that $\boldsymbol{h}$ can be replaced with $\boldsymbol{n}$, as the incoming light direction is always a perfectly reflected outgoing direction $\boldsymbol{\omega}_o$.

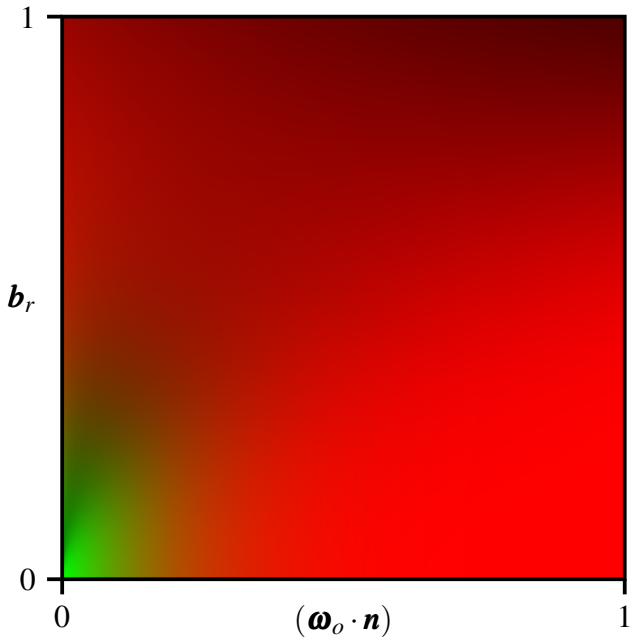



Figure 2.13: **Pre-Integrated BRDF.** The lookup table containing the integrated BRDF terms is indexed by $(\omega_o \cdot n)$ and $b_r$ and the scale and bias $B_0$ and $B_1$ is stored in the red and green channels, respectively.

## 2.3 Inverse Rendering

In rendering, a 2D image is generated from a 3D scene description. Path tracing performs this by tracing rays through the scene. Each ray can also reflect on surfaces. Each part of the scene can influence another part. An overview of this process is explained in Sec. 2.1.

In inverse rendering, the opposite task is performed — given a collection of images or even a single image of a scene, the shape, appearance, and illumination are estimated. The inverse rendering can be performed for all, or a select set of components can be considered. The remaining ones can be expected as GT. For example, the shape and illumination are given, and the appearance is optimized as in Lensch *et al.* [88, 89]. Solving all tasks jointly is highly ill-posed as each component affects the final output. However, only partially solving the tasks can result in local, inescapable minima. A metaphor for this highly ambiguous task is described in Sec. 2.3.1.

The decomposition can also be performed to different extents. The ambiguity is increased significantly when the decomposition is performed in finer granularity.

A common research area is to only estimate the shape. These methods and strategies are discussed in Sec. 2.3.2. Here, the shape can only entail depth information, which is a valid surface representation for the specific view of the 3D object. The number of images required for the approach can also contain either a single image, two images with rigidly defined and placed cameras, a multi-view approach, or even videos, where the temporal

information is leveraged.

Another group of research handles view interpolation between a set of sparse captures. Here, the methods disentangle the visible color from the shape with a light field in Sec. 2.3.3. The visible color is only interpolated between the multiple views. In other words, it remains the product of reflectance and incoming radiance. Intrinsic imaging described in Sec. 2.3.4 aims to split the color into shading and reflectance. However, the split is not physically motivated, and relighting the objects is limited.

A full decomposition of the rendering equation is required for a physically plausible split, as explained in Sec. 2.3.5. Recently, *neural fields* of Sec. 2.4 are often employed in the decomposition of either light fields or a full decomposition due to their simple volumetric shape representations.

### 2.3.1 Workshop Metaphor

Adelson and Pentland [4] introduce the workshop metaphor to explain the challenges of inverse rendering. A theatre workshop department is asked to replicate a given image in this metaphor. Each department branch starts to create the image given their respective specialization. The painter of the department paints on a flat surface to replicate the image in Fig. 2.14c. The lighting department, the gaffer, projects the image on a white surface by carefully shaping and blocking light sources in Fig. 2.14d. Lastly, the sculptor of the department bends metal to conform to a shape that replicates the image under a specific illumination direction in Fig. 2.14e. In other words, a given image can be either solved by altering the texture or appearance (painter), the illumination (gaffer), or the shape (sculptor). When a supervisor is present who distributes work in an ideal manner, all departments can work together. The sculptor creates an upright standing folded sheet, the painter adds a gray stripe running through its center, and the gaffer sets up a simple light source in front of the sheet. This solution is shown in Fig. 2.14b and is the solution where each department invests the least amount of work. It is also a plausible solution for the input image.

In Sec. 2.1 the process of generating images from a 3D scene description is described. For the inverse, this process is not known. So the aim is to model the supervisor of the metaphor. In other words, we decide the most likely explanation of the given image. This problem can be solved with multiple images from multiple views and even multiple illuminations or with priors that use hand-crafted expert priors or statistical distributions over large datasets. These priors can be learned with machine learning.

### 2.3.2 Shape Estimation

Traditionally, shape estimation is performed with two cameras, where the cameras are placed on a rigid setup and calibrated for the extrinsic and intrinsic camera parameters. Features [18, 105] are then extracted and matched between the images. From this, the ray geometry can be reconstructed, which can define the 3D position in relation to the

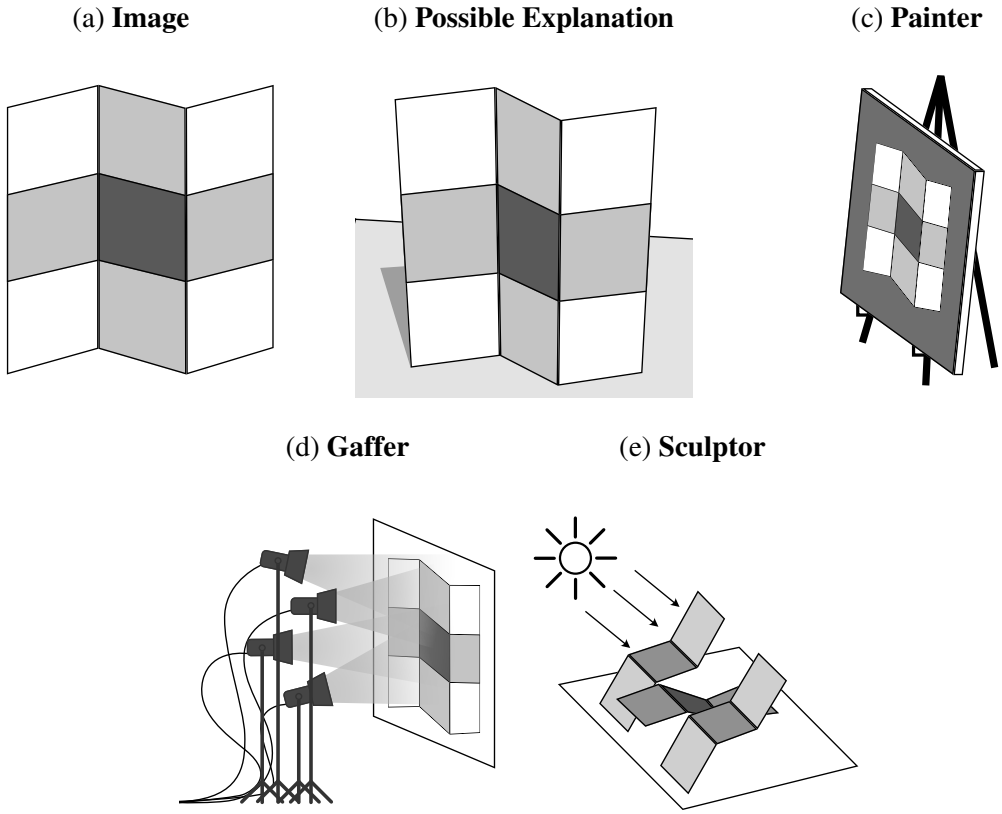



Figure 2.14: **Workshop metaphor.** This visualizes the workshop metaphor, where a theatre workshop supervisor is asked to recreate the image in (a). A possible explanation of how the image was taken is shown in (b). The different departments of the workshop then create the solution (c), (d), and (e) given their specializations. Image inspiration taken from Adelson and Pentland [4] and Barron [13].

cameras [143]. Extensions to this method have been created where wide baselines [151] or sparse correspondences [106, 159] are enabled.

By controlling the illumination, several improvements to the method can be made. One method is *structured light*, where a projector creates *spatial*y [72, 196] or *temporal* [142] patterns on the object's surface and correspondences between the projector and camera are found. Another way of leveraging controlled illumination is *active stereo*, where random patterns are projected on the objects [76, 165, 169]. These patterns can help to find camera-camera correspondences more easily. A combination of both approaches, *i.e*. finding camera-camera and camera-projector correspondences, is *structured light stereo* [71].

Another area of research to estimate a shape is *photometric stereo* [182]. The general task is to estimate the shape from multiple images of different illumination. In Sec. 2.1.2, the BRDF is defined, which defines the amount of reflected light for a direction based

on the incoming and outgoing light direction. When different light directions are measured, the surface orientation can be constrained, as only a few directions can explain the current shading. This can be constrained to a single possible surface orientation with enough light directions. However, highly reflective surfaces are not straightforward to optimize. This can be solved by additionally providing the light directions and optimizing the BRDF [67].

(a) **Ambigious shading** (b) **Clear shading**

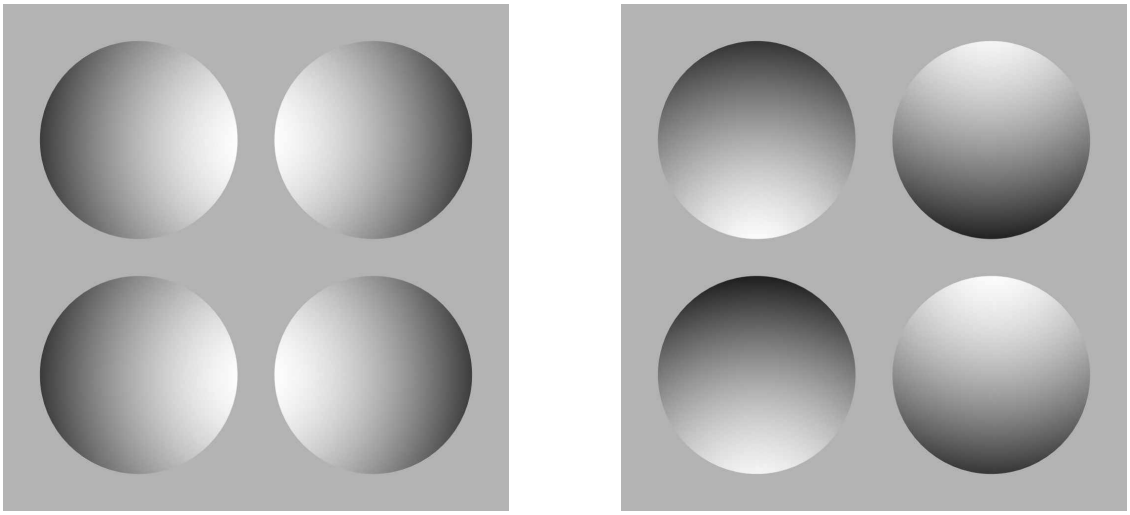


Figure 2.15: **Ambiguity for shape from shading.** When the light position is not known, the shading can become ambiguous. This is visible in (a), where it is not clear whether the shapes are convex or concave. In (b), the shape can be more easily inferred. Illustration inspired by Ramachandran [133].

A specialized case of photometric stereo is estimating the shape using *shape-from-X*. Here, X can express shading [68], silhouettes [17, 111, 153], *etc*. For the case of shape-from-shading, a single image is used under a Lambertian assumption (Sec. 2.1.2.2) [68, 197]. With the visible gradients, the shape can be recreated. Ideally, the position of the light source should be known, as otherwise, ambiguities exist. These ambiguities are visualized in Fig. 2.15.

Another case is shape-from-silhouettes, where a segmentation mask for each image is provided. This binary mask clearly defines which of the image belong to the object or the background. With sufficiently many of these masks, the 3D shape of convex objects can be reconstructed [17, 111, 153].

This can be extended to *multi-view stereo*, where the photometric reconstruction is also performed for multiple cameras. One of the main challenges is finding and matching the dense correspondences between images [146, 179]. If the images are also taken in an ordered manner with temporal information, *i.e*. videos, *structure from motion* can also be employed [134, 145]. Here, the information from motion parallax is especially

useful in the reconstruction. However, highly specular surfaces, varying illumination, and locations are challenging even with these techniques. With SAMURAI of Sec. 4.4, we provide a method to deal with these highly challenging cases.

With the advancements of neural networks in recent years, monocular depth estimation from general scenes became feasible [55, 86, 92, 101, 139, 175]. Here, statistical knowledge gained from large datasets is used to predict relative depth from single images. It is worth emphasizing that only a relative depth up to a scale and shift can be predicted from a single image. Furthermore, these networks are also trained on specific datasets, either indoor, outdoor, or for objects, and the trained network then also only estimate the depths correctly in these specific scenarios.

### 2.3.3 Light Fields

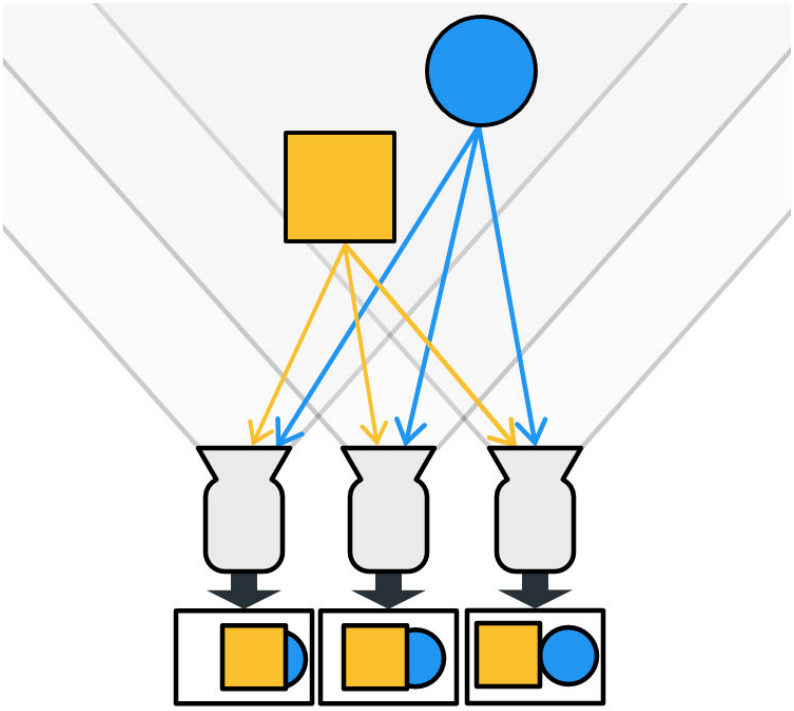


Figure 2.16: **Light Fields.** Traditionally, light fields are captured with an array of cameras. Each surface point is seen by different cameras and, with that, also different angles. This can be used to model view-dependent behavior (Not visualized here).

Michael Faraday first proposed to interpret light as a field [53] – similar to a magnetic field. The term *light field* was then coined by Arun Gershun in a work about radiometry (see Sec. 2.1.1.2) in the three dimensional room [59]. Here, the light field is defined as a five-dimensional function of the three-dimensional coordinate $(x, y, z)$ and the spherical orientation of the ray $(\theta, \phi)$ [3, 59]. This function is also known as the *plenoptic function*.

Capturing a light field is usually done with dense capture setups of either an array of cameras or a precisely controlled moving camera. In Fig. 2.16 a dense capture setup is shown. Here, the rays of two points from two objects are shown. Each camera can see the points with slightly different ray orientations, and the radiance for each direction can be optimized. Therefore, view-dependent effects can be captured with a light field.

The resulting captured light field can allow for several applications. For example, the rays can be refocused, and a synthetic aperture can either be used to increase a shallow depth of field or extend it drastically. Additionally, a degree of Novel View Synthesis (NVS) is possible as a virtual camera can be moved between the camera array in Fig. 2.16. The surface of the scene or object can also be reconstructed based on the parallax (see Sec. 2.3.2).

Recent techniques also allow for sparsely captured light fields. Here, only a collection of images is taken, and the light field is extracted from those [15, 117, 118, 189]. Often traditional cameras are used to capture the light fields, and here the method now learns from the pixel values of the images. These color values correspond to the tone mapped and compressed radiance.

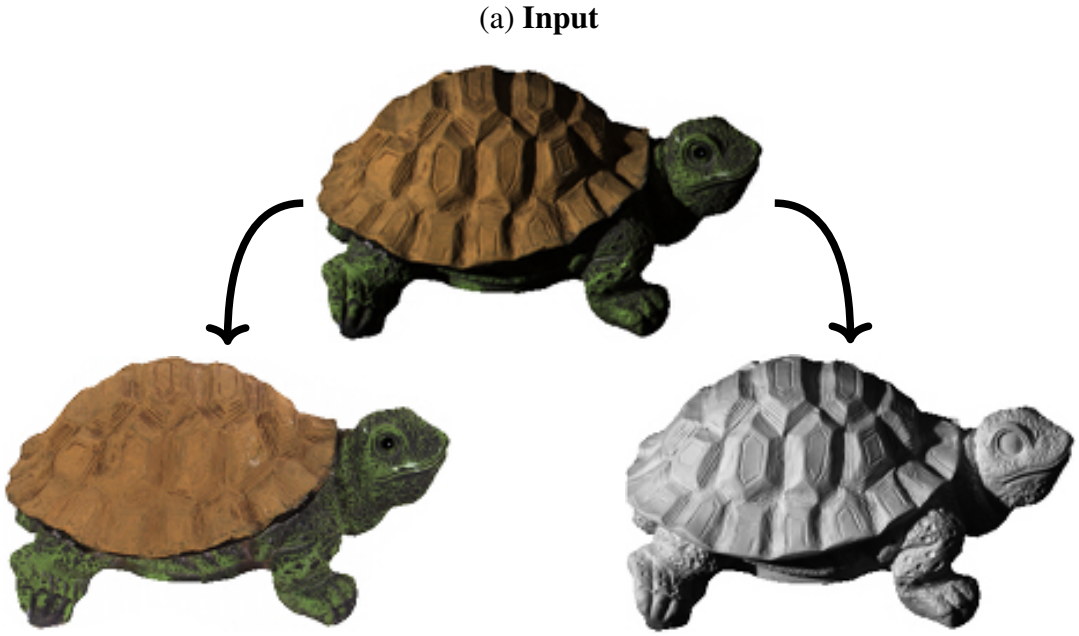
Compared to full decomposition (see Sec. 2.3.5), this partial decomposition only disentangles the shape from the shading and illumination. The appearance of the surface (reflectance) is still entangled with the illumination (incoming radiance). With this decomposition technique relighting a scene is not trivial as only the shape is disentangled from the rendering equation. Often techniques such as Generative Latent Optimization (GLO) embeddings are used to interpolate between multiple seen illuminations [110]. However, it is unlikely to be reconstructable when a specific illumination is not present in the dataset.

### 2.3.4 Intrinsic Imaging

The task of intrinsic imaging was first introduced by Barrow and Tenebaum [16]. The goal is to decompose a given image into separate layers of shape, reflectance, and illumination (irradiance). An exemplary decomposition from Rother *et al.* [137] is shown in Fig. 2.17. There is no physical mechanism underlying this decomposition. Instead, the layers are only multiplied together. Thus, they do not adhere to any physical mechanism that could be used for accurate relighting. For example, the reflectance might be off by a scale and bias.

Inverse rendering with a full decomposition (Sec. 2.3.5) on the other hand tries to solve the underlying physical effects defined in Sec. 2.1 with estimating the BRDF (Sec. 2.1.2) and the incoming illumination (Sec. 2.2). The result can be easily relighted under any illumination and used in computer games, moves, augmented reality, or virtual reality applications.

As an example of the difference in these approaches, the reflectance in intrinsic imaging is comparable to the diffuse albedo color of the BRDF. However, the diffuse albedo scales the BRDF per channel, which has a different meaning and effect than the intrinsic



(a) **Input**

(b) **Reflectance**

(c) **Shading**

Figure 2.17: **Intrinsic imaging.** Exemplary decomposition of (a) into reflectance (b) and shading (c). The reflectance (b) contains the texture with the illumination and shading removed and extracted to the shading (c). Decomposition result is taken from Rother *et al*. [137].

color, which might also contain color from lighting. Therefore the colors differ both by a scale and bias. The recovered illumination is also accurate up to a scale and bias. It is possible to reconstruct the input image with both terms, but relighting is not easily possible. The result may be too bright or dark. An example of an intrinsic *vs*. explicit BRDF decomposition is shown in Fig. 2.18. In inverse rendering, the albedo is an absolute unit, and any illumination can be applied, and the recovered material behaves accordingly.

### 2.3.5 Full Decomposition

During a full decomposition, not only the BRDF and shape are estimated, but also the incoming illumination. Here, the rendering equation is actually split into all parts. Optimizing for the components of the recursively defined integral in the rendering equation is ambiguous, as shown in Sec. 2.3.1 and therefore prone to getting stuck in local minima. Often simplifications and approximations of the rendering equation are taken as described in Sec. 2.1.1.1. Additionally, the incoming illumination from environments is highly costly to evaluate, as the entire hemisphere of incoming light needs to be integrated. During optimization, the image or parts are rendered in each of the thousands of

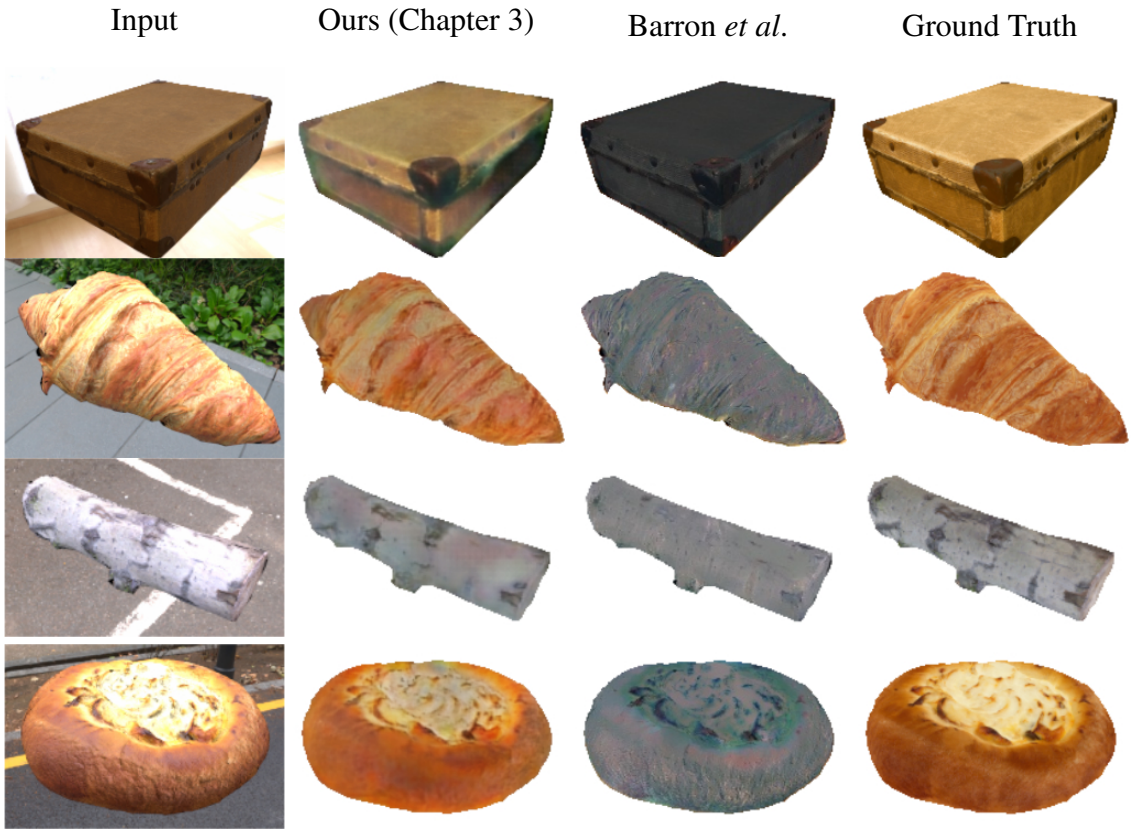



Figure 2.18: **Intrinsic imaging *vs*. BRDF decomposition.** Comparison of reflectance and diffuse composition our method of Chapter 3 and Barron *et al*. [14]. Here, our method performs a BRDF and illumination decomposition. Barron *et al*. perform an intrinsic imaging decomposition. The intrinsic decomposition exhibits a scale and shift in color, which is attributed here to the shading. A novel relighting with this decomposition would show an incorrectly colored object.

optimization steps. Often approximate illumination models from Sec. 2.2 are employed to reduce the computational cost during the optimization.

With these modifications a differentiable renderer is often implemented [25, 28, 29, 30, 45, 192]. Here, the rendering equation is either implemented in an automated differentiation framework, *e.g*. Tensorflow, Pytorch, Jax, or the gradients to each specific input are manually derived. Given these differentiable renderers, the decomposition task can be formulated as a gradient-based optimization, where the objective function compares a real image (GT) is compared to a rendered image. The loss can then be backpropagated through the rendering operation. This even allows for unsupervised decompositions of image collections. However, due to the ambiguity discussed in Sec. 2.3.1, the optimization often only reaches a local minimum. Priors or regularization are often employed to

guide the optimization to a more plausible solution.

Despite the significant ambiguities and the inherent challenge, this approach has several advantages. As everything is jointly optimized, it is possible to avoid local minima arising when each task is solved separately (see Sec. 2.3.1).

As discussed in Sec. 2.3.2, highly specular materials cannot be optimized easily with traditional shape estimation techniques. This challenging task can be solved when everything is jointly optimized, as shown in Chapter 4.

Additionally, a full decomposition allows relighting compared to the light fields of Sec. 2.3.3. Here, only novel views can be synthesized. Compared to intrinsic imaging from Sec. 2.3.4, the different properties of a BRDF model (see Sec. 2.1.2.2) are fully disentangled and allow for flexible relighting under any illumination.

Given the advantages and potential applications of the decompositions, we tackle the task of full reflectance, shape, and lighting decomposition in the thesis and provide techniques to reduce the ambiguities inherent to this task.

## 2.4 Neural Fields

A recent field in machine learning is the usage of neural networks to encode values associated with corresponding coordinates. For example, storing all RGB values $\boldsymbol{c} \in \mathbb{R}^3$ of an image for all corresponding coordinates $\boldsymbol{x} \in \mathbb{R}^2$. This can be extended to a volume where next to a color $\boldsymbol{c}$ a density $\sigma \in \mathbb{R}$ or signed distance to the encoded surface $d \in \mathbb{R}$ can also be stored.

### 2.4.1 Coordinate-based Multilayer Perceptrons

The values of images, volumes, or even temporally changing volumes can be encoded based on the coordinates in an Multilayer Perceptron (MLP). The MLP [136] then acts as compressed storage for the values and can perform an interpolated lookup of these values. The earliest works performing this are: [39, 116, 127]. Here, existing point clouds are encoded in the network architectures. The network can then be defined as $f(\boldsymbol{x};\theta) \rightarrow r$, with $\theta$ being the weights and biases of the MLP. Here, $r$ can encode the occupancy $o \in 0,1$, which decides in binary fashion if a coordinate is occupied or not [39, 116] or the Signed Distance Field (SDF) $d \in \mathbb{R}$. The distance $d$ is 0 on the surface. Outside the object, the SDF defines a positive value indicating the distance to the closest surface and a negative value for the closest distance to a surface inside an object. MLPs such as these provide an inductive bias for smooth surfaces and can fill in holes or gaps in the point cloud.

### 2.4.2 Neural Volume Rendering

Lombardi *et al*. [104] introduce volume rendering for a neural volume. This volume is created from features extracted from images using Convolutional Neural Networkss (CNNs). The main goal of synthesizing novel views can then be achieved by aggregating these features along a camera ray $r(t) = \boldsymbol{o} + t\boldsymbol{d}$, with ray origin $\boldsymbol{o} \in \mathbb{R}^3$ and view direction $\boldsymbol{d}$ as well as the distance along the ray $t \in \mathbb{R}$. Due to the volumetric nature, the samples need to be integrated. This can be approximated using numerical quadrature:

$$\tilde{\boldsymbol{c}}(\boldsymbol{r}) = \int_{t_n}^{t_f} T(t)\sigma(t)\boldsymbol{c}(t)\,dt \approx \sum_{i=1}^{N} T_i\left(1 - \exp(-\sigma_i\delta_i)\right)\boldsymbol{c}_i \tag{2.38}$$

$$T(t) = \exp\left(-\int_{t_n}^{t} \sigma(t)\,dt\right) \tag{2.39}$$

$$T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j\delta_j\right) \tag{2.40}$$

$$\delta_j = t_{i+1} - t_i \tag{2.41}$$

where the output feature or color $\tilde{\boldsymbol{c}} \in \mathbb{R}^3$ for the camera ray $r$ is aggregated using the volumetric density $\sigma \in \mathbb{R}$, and the specific feature or color at the location $\boldsymbol{c} \in \mathbb{R}^3$. The variables $t_n \in \mathbb{R}$ and $t_f \in \mathbb{R}$ express the near and far bounds of the ray, respectively.

### 2.4.3 Neural Radiance Fields

Mildenhall *et al*. propose the now seminal method NeRF [117]. The method spawned a large thread of follow-up work and extensions [15, 25, 37, 65, 81, 102, 102, 120, 120, 126, 141, 168, 172, 173, 189]. NeRF is a method that achieves photorealistic results for the task of novel view synthesis.

In NeRF, every new scene requires full training from scratch. No information is shared between different objects. However, specialized extensions exist that leverage Meta-learning to generate a specialized set of starting weights for a specific instance [154]. In NeRF the a *neural field* is created with a coordinate-based MLP and neural volume rendering. Each training step selects a stochastic batch of pixels and the corresponding camera rays. Samples are then placed along the rays. The sample positions are evaluated in the current field and aggregated with volume rendering. The resulting color for each ray can be compared with the input. This process is repeated over several hundred thousand steps, and the neural field starts to replicate the input scene slowly. This process is similar to the volumetric reconstruction of computer tomography.

More specifically, the method optimizes the plenoptic function from Sec. 2.3.3 from a set of posed images. No further supervision is required, and photorealistic novel views can be synthesized.

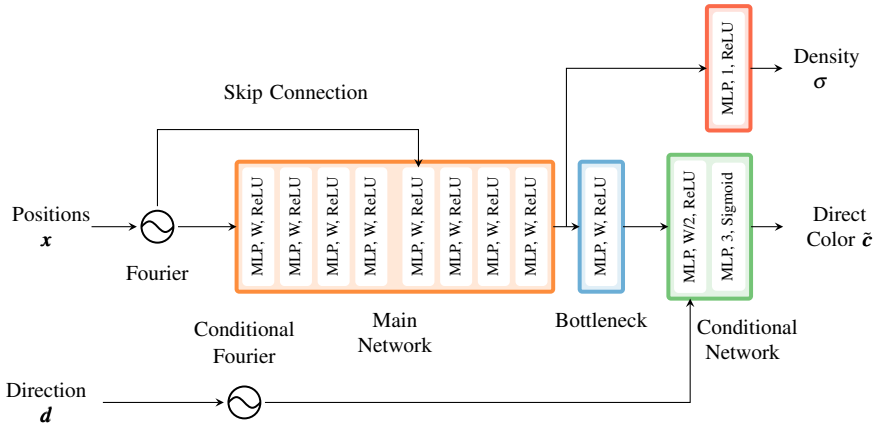



Figure 2.19: **Architecture.** The NeRF architecture shown is used in both networks. The main outputs include the Density $\sigma$, view dependent output color $\tilde{\boldsymbol{c}}$.

The method combines coordinate-based MLP with the positional encoding of transformers [166]. This positional encoding is also known as the Fourier Encoding due to sinusoidal activation of the input coordinates. The Fourier Encoding $\gamma : \mathbb{R}^3 \mapsto \mathbb{R}^{3+6L}$ used in NeRF [117] encodes a 3D coordinate $\boldsymbol{x}$ into $L$ frequency basis:

$$\gamma(\boldsymbol{x}) = (\boldsymbol{x}, \Gamma_1, \ldots, \Gamma_{L-1}) \tag{2.42}$$

where each frequency is encoded as:

$$\Gamma_k(\boldsymbol{x}) = \left[\sin(2^k \boldsymbol{x}), \cos(2^k \boldsymbol{x})\right] \tag{2.43}$$

Even though this encoding is low-frequent, the network can learn high-frequency information more easily. Tancik *et al.* investigate this property using methods from the neural tangent kernel (NTK) literature [155]. Here, it is shown that the spectral bias of the coordinate-based MLP is overcome with this simple encoding.

NeRF then leverages neural volume rendering to define the specific output color for the novel view. On a camera ray, multiple samples are evaluated and aggregated. Sampling a regular distance would introduce a bias for these fixed locations. Instead $N$ evenly spaced are sampled in uniform stratified strategy:

$$t_i \sim U\left[t_n + \frac{i-1}{N}(t_f - t_n), t_n + \frac{i}{N}(t_f - t_n)\right] \tag{2.44}$$

Resolving fine detail requires more samples. However, increasing the number of the $N$ sampling bins is highly inefficient, as most areas in a typical scene are empty. To overcome this, NeRF proposes a hierarchical sampling strategy, with two separate networks which are trained in conjunction. The first network is the coarse network and is sampled in the stratified strategy. With the knowledge about which areas are more likely

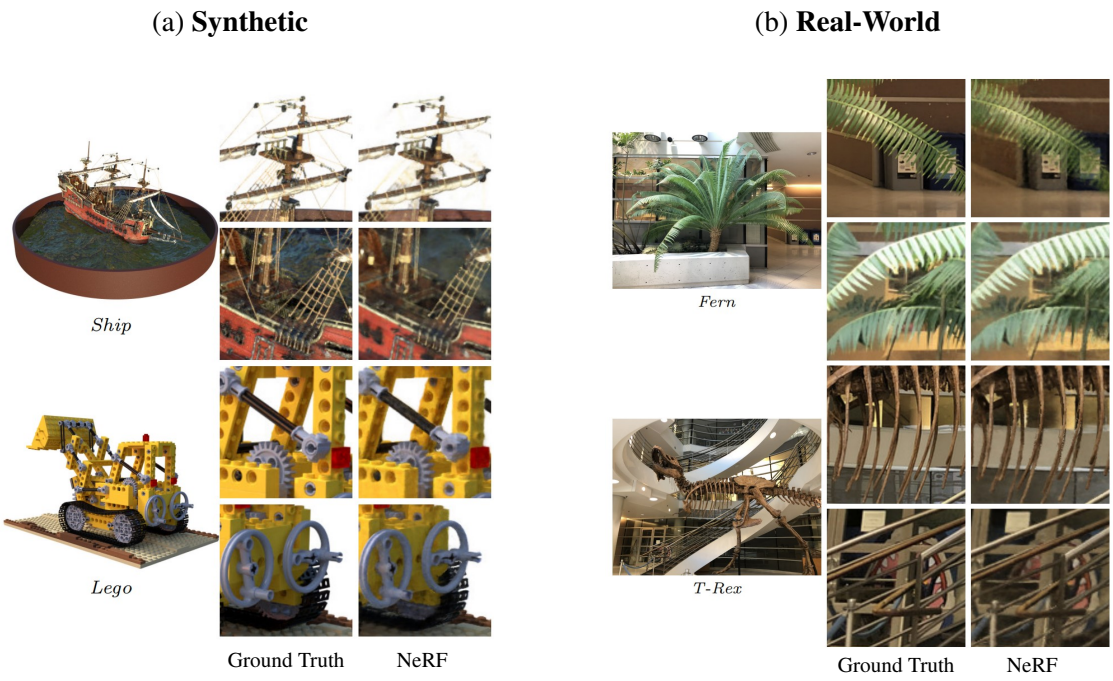



Figure 2.20: **NeRF results.** Shows results on synthetic and real-world scenes from NeRF [117].

to be occupied, more samples are placed closer to the occupied areas. This is achieved by rewriting the numerical quadrature of Eq. (2.38) as:

$$\tilde{\boldsymbol{c}}(\boldsymbol{r}) = \sum_{i=1}^{N} w_i \boldsymbol{c}_i \tag{2.45}$$

$$w_i = T_i \left(1 - \exp(-\sigma_i \delta_i)\right) \tag{2.46}$$

The weights can be normalized as:

$$\hat{w}_i = \frac{w_i}{\sum_{j=1}^{N} w_j} \tag{2.47}$$

A piecewise-constant PDF is created along the ray with the normalized weights. The second set of sampling locations is then drawn from this distribution using inverse transform sampling combined with the stratified samples of the coarse network. All samples are then passed to the second network for a finer rendering. The final synthesized novel view is only generated from the fine network. The coarse network's only role is to generate a more efficient sampling pattern.

Both networks leverage the same architecture. The structure is shown in Fig. 2.19. Here, the main network is trained only on the Fourier encoded positions. The result is a static encoding for each point, from which the density $\sigma$ is extracted. A simple bottleneck is introduced, which is then conditioned on the view direction. This direction is encoded with the same Fourier encoding but often of a lower frequency to enforce smoothness in the interpolation. The final output color per point is produced from the bottleneck encoding and the encoded directions.

Results of their method are shown in Fig. 2.20, where the close reproduction of the scene is visible. However, while the result shows a photorealistic reproduction, the method is not capable of relighting the scene. Only novel views can be estimated, as the shape of the scene is consistently stored in the neural field, and the view-dependent color can smoothly interpolate between the colors observed in the training views.

# Chapter 3

# Generalized Few-Shot Decomposition

In the context of this work, *generalized methods* are defined as methods that are trained once with direct supervision on large datasets, where GT information is available. For example, in this chapter, we have GT information about the BRDF, the illumination, and the shape – in the form of the depth and surface normal – for each image. The pre-trained weights are evaluated for novel scenes without any specialized re-training for the scene.

This enables fast inference times as no optimization is performed and specialized inference-only modes exist [2]. Here, no backpropagation has to be performed, drastically reducing memory consumption and computation times.

Often these pre-trained architectures are especially useful for decomposing a few images, where the training provides the statistical knowledge about the behavior of surfaces from the large-scale dataset. Without this knowledge from a dataset, decompositions into shape, BRDF, and illumination are nigh impossible from a few images. As shown in Sec. 2.1.2.1 a BRDF measurement alone requires a large number of various viewing angles.

However, as the decomposition from a few images is still highly ill-posed, the decomposition quality is more limited than optimization from a larger set of images. Methods that optimize the decomposition of each object are discussed in Chapter 4.

*This following sections are based on the publication:*

Two-shot Spatially-varying BRDF and Shape Estimation
Mark Boss, Varun Jampani, Kihwan Kim, Hendrik P. A. Lensch, Jan Kautz
*IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* - 2020

## 3.1 Related Work

The literature on object SVBRDF and/or shape estimation is vast. Here, we only discuss the representative works that are related to ours.

**BRDF Estimation.** An exhaustive sampling of each BRDF dimension demands long acquisition times. Several proposed methods focus on reducing acquisition time [6, 11,

27, 49, 87, 88]. These methods introduce capture setups and optimization techniques that reduce the number of images required to reconstruct high-quality SVBRDF. Recently, several attempts reconstruct the SVBRDF on flat surfaces with single-shot [44, 66, 97, 140], few-shot [5, 7] or multi-shot [9, 27, 45, 46, 56] estimation. These approaches leverage neural networks trained on large amounts of data and resolve the problem of ambiguity to some extent by learning the statistical properties of BRDF parameters.

For a joint estimation of shape and shading, separate optimization steps for shape and shading are common [14, 61, 89, 122]. Lensch *et al*. [89] introduce Lumitexels, which stack previously acquired shape information with the luminance information from the input images, to guide the BRDF estimation and to reduce ambiguities in the optimization. Compared to a joint estimation, fewer local minima are found, and the optimization is more robust. Recently, the task of predicting the shape and BRDF of objects or scenes is also addressed using deep learning models [98, 147]. Li *et al*. [98] predict the shape and BRDF of objects from a single flash image using an initial estimation network followed by several cascaded refinement networks. Here, the BRDF consists of diffuse albedo and specular roughness but lacks the specular albedo. Specularity is, however, essential in re-rendering metallic objects, for example.

Compared to Li *et al*. [98], our method described in Sec. 3.2 additionally estimates the SVBRDF with specular albedo. In comparison to flat surface SVBRDF estimation [5, 7, 44, 45, 97], our method handles full objects with shapes from any view position. Additionally, due to our unaligned two-shot setup, saturated flash highlights are better compensated while still providing the same one-button press capture experience for the user due to our mobile capture scenario.

**Intrinsic Imaging.** As described in Sec. 2.3.4, intrinsic imaging is the task of decomposing an image of a scene into reflectance (diffuse albedo), and shading [14, 16, 108, 156]. With the advance in deep learning, the problem of separating shape, reflectance, and shading is tackled from labeled data [90, 123, 148], unlabeled [96] and partially labeled data [19, 95, 124, 201]. Due to the simplistic rendering model, the use cases are limited compared to our SVBRDF estimation setup, which can be used for general re-rendering in new light scenarios.

**Shape Estimation.** One can obtain high-quality depth from stereo images, but the problem of monocular depth estimation is quite challenging. Monocular depth estimation is predominantly tackled with deep learning [55, 86, 92, 101, 139, 175] in recent years. This problem is challenging as no absolute scale is known from single images, and the depth cues need to be resolved by shading information such as the quadratic light fall-off [99].

**Placement in the literature.** Our technique of Sec. 3.2 can be seen in relation to other current literature in Tab. 3.1. The techniques enable the decomposition of objects instead

of flat surfaces as in previous research [44, 45].

Our method outperforms other methods aimed at object reconstruction, such as Li *et al*. [98] while enabling a more challenging decomposition with a spatially-varying specular color. This spatially varying property is essential when metals and non-metals are combined on a single object.

It also enables more casual capture setups [47] or has no constrained shape prior such as in Wu *et al*. [183], which only works on rotationally symmetric objects.

Our method also estimates environment illumination, which increases the accuracy in simple capture setups by enabling methods to explain illumination from environment light sources.

| Method | Number of Images | Target | BRDF-Model | Estimate Environment Illumination |
|---|---|---|---|---|
| Deschaintre[44] | 1 Flash | Flat Surface | Cook-Torrance | Not handled |
| Deschaintre[45] | 1-∞ Flash | Flat Surface | Cook-Torrance | Not handled |
| Deschaintre [47] | 3 Flash+Varying Polarization | Objects | Cook-Torrance | Not handled |
| Li [98] | 1 Flash | Objects | Cook-Torrance (No Specular) | Estimated |
| Wu [183] | 1 | Rotationally Symmetric Objects | Phong | Estimated |
| Ours [29] | 2 | Objects | Cook-Torrance | Estimated |

Table 3.1: **Literature overview.** Our work either enables a more challenging decomposition with a full Cook-Torrance model including spatially-varying specular albedo, solves the highly ambiguous decomposition of objects, and achieves a convenient, hand-held capture setup under environment illumination.

## 3.2 Two-shot Spatially-varying BRDF and Shape Estimation

In this work, we are interested in the automatic estimation of the shape and appearance of the object in a scene from only two images. In particular, we represent the shape of the object with a depth map and the appearance as a BRDF (Sec. 2.1.2). A BRDF describes the low-level material properties of an object that defines how light is reflected at any given point on an object's surface. One of the most popular parametric models [41] represents the diffuse and specular properties and the roughness of the surfaces. Since the material properties can vary across the surface, one has to estimate the BRDF at each image pixel for a more realistic appearance (i.e., spatially-varying BRDF (SVBRDF)).

As the BRDF is dependent on view and light directions and estimating depth from a single 2D image is an ambiguous task, multi-view setups improve the estimation accuracy of both shape [145] and BRDF [114]. Predicting shape and BRDF from only a few images is still very challenging. For shape estimation, the advances in deep learning-based depth estimation allow us to estimate the depth of a single [55, 86] or a pair of images [164] efficiently. As monocular depth estimation is not as accurate as multi-view approaches, we exploit shading cues on the surface to disambiguate the geometric

shape [10, 197] in our approach.

We propose a neural network-based approach to estimate the SVBRDF and shape of an object along with the illumination from given two-shot images: flash and no-flash pairs. Some recent deep learning approaches [44, 97, 98] for BRDF estimation use only a single flash image as input. Flash images often have harsh reflective highlights where the input pixel information is saturated in non-HDR images.

Li *et al*. [98] uses a single input image and estimates the shape and parts of the BRDF, such as diffuse albedo and the roughness, while ignoring the specular color. In this work, we use flash and no-flash image pairs as input allowing the network to access pixel information from the no-flash image when the corresponding pixels are saturated in the flash image. We focus on practical utility: Our input capture setup follows a real-world scenario where the two-shot images are consecutively taken using a mobile phone camera in burst capture. The system is designed to tackle the misalignment between the two-shot images due to camera shake.

A pivotal challenge for any learning approach is the need for training data. We tackle this issue by creating a large-scale synthetic dataset. Flash and no-flash images are rendered using high-quality, human-authored SVBRDF textures that are applied to synthetic geometry generated by domain randomization [158] of geometric shapes and backgrounds. Our networks trained on this synthetic data generalize well to real-world object images.

As briefly discussed in the Workshop Metaphor in Sec. 2.3.1, another key challenge in shape and SVBRDF estimation is the problem of ambiguity. We tackle this ambiguity by using a cascaded approach, where separate neural networks are used to estimate shape (depth), illumination, and SVBRDF. Fig. 3.1 shows an overview of our cascaded network. Specifically, we first estimate depth and normals using a geometry estimation network. Then the illumination is approximated, followed by SVBRDF reconstruction. The estimates of the previous networks guide each following step. Finally, shape and SVBRDF are optimized jointly using a refinement network. Specialized network architectures perform each task. Empirically, this cascaded regression approach works better than a single-step joint estimation. As a favorable side-effect of this cascaded approach, the size of each network is small compared to a large joint estimation network. This allows the inference networks to even operate on a mobile device. Coupled with two-shot mobile capturing, this presents a highly practical application.

Quantitative analysis based on a synthetic dataset comprising realistic object shapes and SVBRDFs demonstrates that our approach produces more accurate estimates of shape and SVBRDF than baseline approaches. We also qualitatively demonstrate the applicability of our approach on a real-world two-shot dataset [8].

### 3.2.1 Problem Setup

Our network takes two-shot object images (flash and no-flash) with the corresponding foreground object mask and estimates shape and SVBRDF. We also estimate illumina-

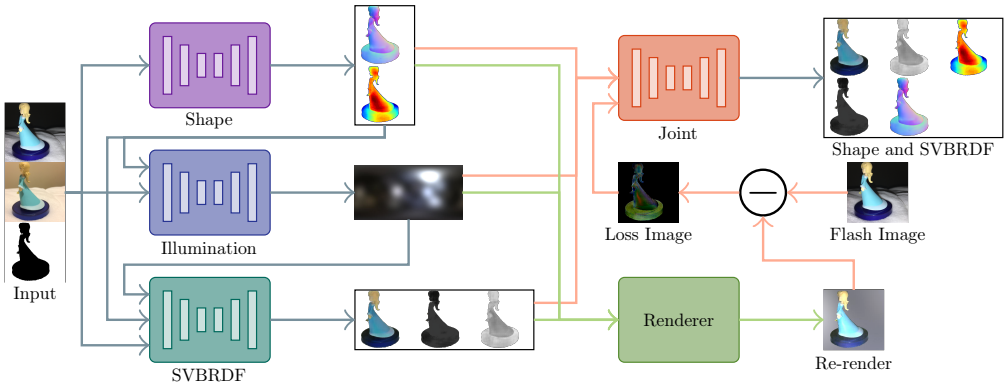



Figure 3.1: **Cascaded network.** Overview of the inference pipeline for shape, illumination, and SVBRDF estimation.

tion as a side-prediction to help shape and SVBRDF prediction. The two-shot images might be slightly misaligned to support practical image capture with a handheld camera. The object mask allows us to evaluate only the pixels of the object in the flash image and is easily generated with GrabCut [138]. The object shape is represented as depth and normal at each pixel. The depth map provides a rough shape of the object, while the normal map models local changes more precisely. This shape representation is commonly used in various BRDF estimation methods [98, 122]. We use the Cook-Torrence model [41] to represent the BRDF at each pixel with diffuse albedo (3 parameters), specular albedo (3), and roughness (1). The model is defined in Sec. 2.1.2.2. Similar to [91, 174], we estimate the environment illumination with 24 SGs, as described in Sec. 2.2.1.

### 3.2.2 Network Overview and Motivation

In order to tackle the shape/SVBRDF ambiguity, we take inspiration from traditional optimization techniques [89, 122], which iteratively minimize a residual and alternate between optimizing for shape and/or reflectance. Thus, separate networks are used for shape, illumination, and SVBRDF estimation in a cascaded as well as an iterative manner. Predictions from earlier stages of the networks in the cascade are used as inputs to later networks to guide network predictions to better solutions. In addition, the scene is re-rendered with the current estimates and refined further using the residual image.

Since flash and no-flash images are slightly misaligned, shape estimation is less challenging than the estimation of the SVBRDF. Mis-alignment in pixels and pixel differences between two-shot images [99] are good indicators of object depth. Thus, we first predict depth and normals using a specialized merge convolutional network followed by a shape-guided illumination estimation. Then, the SVBRDF is predicted with the current estimates of shape and illumination as additional input. Finally, we refine both shape and SVBRDF using a joint refinement network after computing a residual image. Refer to the appendix Sec. A.1 for network architecture details.

### 3.2.3 Network Architecture

Our proposed network architecture contains multiple specialized networks for each task. In Sec. 3.2.3.1 the shape prediction is explained. With the initial shape estimation, an illumination estimation is performed with the architecture described in Sec. 3.2.3.2. The next step in the pipeline is the SVBRDF estimation in Sec. 3.2.3.3, followed by a re-rendering with the initial prediction. This re-rendering allows for capturing an image for loss guidance for the joint refinement in Sec. 3.2.3.4.

#### 3.2.3.1 Shape Estimation with Merge Convolutions

Since the camera parameters are unknown and the two-shot images have a minimal baseline, traditional structure-from-motion or stereo solutions are not useful for dense depth estimation. The shape estimation needs to rely on the unstructured perspective shift and pixel differences between flash and no-flash images. In order to tightly integrate information from both images, we design a specialized convolutional network for shape estimation.

For depth and normal map prediction, we use a U-net-like encoder-decoder architecture [135]. Instead of standard convolution blocks, we propose to use novel merge convolution blocks (*MergeConv*). We concatenate the object mask with each of the two-shot input images as input to the network. Fig. 3.2 illustrates the MergeConv block. The input images and their intermediate features are separately processed by 2D convolutions (Conv2D). The outputs of each Conv2D operation are concatenated in channels with the merged output from the previous MergeConv layer and is processed with another Conv2D operation. Inspired by residual connections in ResNet [64], we add the Conv2D outputs as indicated in Fig. 3.2. We use 4 MergeConv blocks for the encoder and 4 for the decoder. During encoding, max pooling for $2\times$ spatial downsampling is used. For each MergeConv in the decoder, we use $2\times$ nearest neighbor upsampling. The final depth and normal map estimates are produced using a separate 2D convolution, followed by a sigmoid activation. The rationale behind this MergeConv architecture is to keep separating the process of pathways for both the input images while exchanging (merging) the information between them using a third pathway in the middle. We believe that information in both input images is essential for shape reasoning, and this architecture helps keep the features from each image intact throughout the network. Empirically, we observe reliably better shape predictions with this architecture than a standard U-net with a similar number of network parameters.

Training losses are based on the $\mathcal{L}_2$ distance between GT and predicted depths, ${\mathcal{L}_2}^{\text{depth}}$, as well as the angular distance between GT and predicted normals $\hat{\boldsymbol{n}}$:

$$\mathcal{L}_{angular}^{\text{normals}} = \frac{\arccos\left(\boldsymbol{n} \cdot \hat{\boldsymbol{n}}\right)}{\pi} \tag{3.1}$$

Besides, we use a new consistency loss between the predicted normal $\hat{\boldsymbol{n}}$ and a normal $\boldsymbol{n}^*$

derived from the depth information $d$, which enforces that the predicted normals follow the curvature of the shape:

$$\mathcal{L}_{consistency}^{\text{normals/depth}} = \frac{\hat{\boldsymbol{n}}}{\|\hat{\boldsymbol{n}}\|} - \frac{\boldsymbol{n}^*}{\|\boldsymbol{n}^*\|}, \tag{3.2}$$

$$\boldsymbol{n}^* = \begin{bmatrix} \nabla d & 2\frac{1}{\text{width}} \end{bmatrix}^T = \begin{bmatrix} \frac{\partial d}{\partial x} & \frac{\partial d}{\partial y} & 2\frac{1}{\text{width}} \end{bmatrix}^T, \tag{3.3}$$

The normal $\boldsymbol{n}^*$ is derived from the depth map using gradients from the horizontal ($x$) and vertical ($y$) directions. The $z$ component can be considered a scalar factor that is derived from the image width. The total loss is a weighted combination of the three losses: $\mathcal{L}_2^{\text{depth}} + \mathcal{L}_{angular}^{\text{normals}} + 0.5 \times \mathcal{L}_{consistency}^{\text{normals/depth}}$.

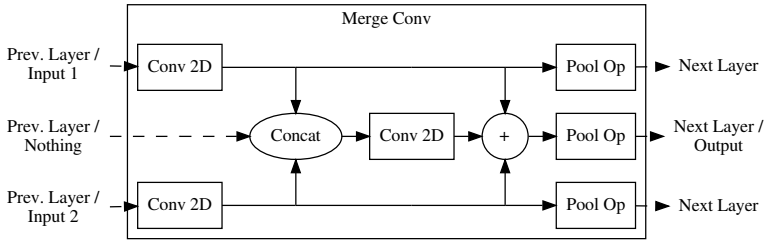



Figure 3.2: **Merge convolutions.** The merge convolution provides separate pathways for the two-shot inputs and merges the information in a third path.

#### 3.2.3.2 Shape Guided Illumination Estimation

We also estimate the environment illumination to guide SVBRDF predictions. Now, the BRDF prediction can consider environment light, reduce additional highlights, and improve the albedo colors and intensities. The illumination is represented with 24 SGs, where each SG is defined by amplitude, axis, and sharpness. However, we only estimate the amplitude and set the axis and sharpness to cover a unit sphere. The estimation thus only estimates the amplitudes of the SG resulting in 24 RGB values. As the environment illumination can reach very high values and the flash and no-flash input images are in Low Dynamic Range (LDR), SG amplitudes are constrained to values between 0 and 2. An overview of the visual quality of the SG illumination approximation is shown in Fig. 3.3. Here, the 24 SGs provides a balance between the number of parameters to optimize and the visual quality.

We use a small convolutional encoder network followed by an MLP for illumination estimation. The network receives input from the two-shot images, the object mask, and the previously predicted depth and normals. As illumination is reflected on the surface towards the viewer, the previously estimated shape information helps in the illumination

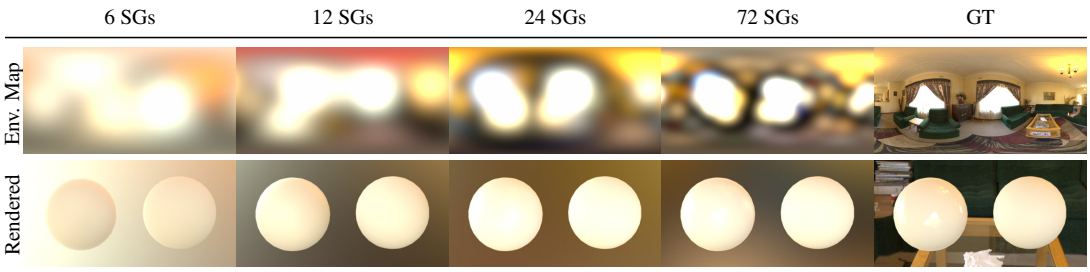



Figure 3.3: **SG illumination fits.** A varying number of SGs fitted to an environment illumination. Here, two spheres with different materials are lighted by the fitted SGs. The left sphere is highly reflective, and the right sphere uses a diffuse material. In this work, we use 24 SGs, as increasing the SGs further does not improve the lighting quality drastically.

estimations. To train the illumination network, we use the $\mathcal{L}_2$ distance between predicted and GT SGs as the loss function.

#### 3.2.3.3 Guided SVBRDF Estimation

SVBRDF estimation becomes a less ambiguous task when conditioned on known object shape and environment illumination. Thus, together with two-shot images, the previously estimated depth, normals, and illumination are used as input to the SVBRDF network to predict diffuse albedo and specular color as well as surface roughness at each pixel. Following recent work on BRDF estimation [44, 93, 97], the U-net architecture [135] is used in our SVBRDF network.

**Differentiable Rendering.** We develop a differentiable rendering module to re-render the object flash image from the estimated depth, normals, illumination, and SVBRDF. The renderer evaluates the direct light from the flash-light source and the estimated environment illumination at each surface point. It integrates it with the BRDF to compute the reflected light [75]. Fast evaluation of the environment illumination is achieved by representing the illumination as well as the BRDF model as SG [171]. Refer to Sec. 2.2.1 for a detailed description of the rendering approximation using SGs.

**Loss Functions for SVBRDF Network.** The SVBRDF network is trained using a combination of different loss terms: the Mean Absolute Error (MAE) between GT and the predicted SVBRDF parameters, as well as a loss between a synthetic direct illumination only flash GT image and re-rendered direct illumination flash image. The rendering loss is back-propagated through the differentiable renderer to update the SVBRDF network. As rendering can result in large values from specular highlights, the MAE loss is calculated on $\log(1+x)$, where $x$ refers to the direct light only synthetic input and the re-rendered image.

#### 3.2.3.4 Joint Shape and SVBRDF Refinement

In our cascaded network, we use the estimated depth to guide the SVBRDF prediction. Likewise, one can obtain better depth prediction with known SVBRDF. We jointly optimize depth, normals, and SVBRDF using a separate refinement network. All the earlier predictions and the residual loss image between the re-rendered previous result and the input flash image are used for this refinement. The network architecture is a small CNN encoder and decoder of 3 steps, each with 4 ResNet blocks [64] in-between. The loss function is an MAE loss between the predicted parameter maps and ground truth ones.

#### 3.2.3.5 Implementation

The cascaded networks and the differentiable renderer are implemented in Tensorflow [2]. The overall pipeline consists of 4 networks, as illustrated in Fig. 3.1.

**Runtime.** Each network is relatively small, and the inference pipeline takes 700ms on a 256×256 image on an NVIDIA 1080 TI, including the required rendering step. On a Google Pixel 4, the evaluation takes roughly 6 seconds. The rendering step is implemented in software and takes 220ms on a single-threaded desktop CPU (AMD Ryzen 7 1700) and similar speeds on a Google Pixel 4.

**Training.** All the networks are trained for 200 epochs with 1500 steps per epoch using the ADAM optimizer [82]. Here we use a learning rate of $2e{-}4$ at the beginning, which is reduced by half after 100 epochs. The networks are trained sequentially as each network in the cascade uses the result of earlier networks as input.

**Mobile Application for Scene Capture and Inference.** In addition to producing better results, another significant advantage of the cascaded network design compared to a single joint network is that each sub-network is small, and the overall network can fit on mobile hardware. We convert the network models to Tensorflow Lite, which runs on mobile hardware, and develop an efficient android application that can successively capture two-shot flash and no-flash images and runs the cascaded network to estimate SVBRDF and shape. We do not run the computations using quantization as the results degraded too harshly. A quantization-aware training could remedy this effect. A quantized output and model increases the prediction speed even further, but it is already reasonably fast. The mobile android application is written in kotlin and handles capturing objects, segmentation, and prediction. The capturing automatically takes the two-shot input pair. The segmentation is done using OpenCV's GrabCut [138] implementation on the device. In Fig. 3.10 a prediction from the mobile application is shown.

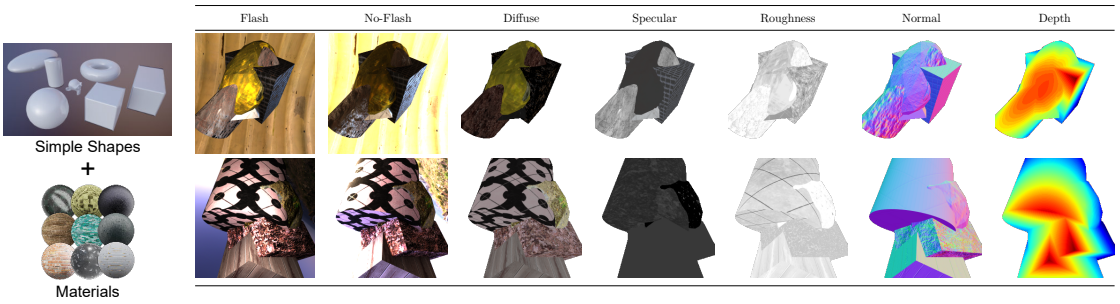



Figure 3.4: **Large-scale synthetic dataset.** (Left) Samples of primitive shapes and materials used for the dataset creation, (Right) The visualization of two examples with various properties.

### 3.2.4 Large-scale SVBRDF & Shape Dataset

It is time-consuming and expensive to scan SVBRDF of real-world objects. Since we rely on deep learning techniques for SVBRDF and shape estimation, vast amounts of data are needed for network supervision. We create a large-scale synthetic dataset with realistic SVBRDF materials.

**High-quality Material Collection.** We gather a collection of publicly available human-authored, high-quality SVBRDF maps from various online sources [31, 43, 129, 157, 163, 191]. The parameterization of these collected SVBRDF maps is for the Cook-Torrence model [41], as described in Sec. 2.1.2.2. In total, the collection consists of 1125 high-resolution SVBRDF maps. To further increase the material pool, we randomly resize and take $768 \times 768$ crops of these material maps. We apply random overlays and perform simple contrast, hue, and brightness changes. The final material pool contains 11,250 material maps. Sample material maps are shown in Fig. 3.4.

**Domain Randomized Object Shapes.** One option for generating 3D objects is to gather realistic object meshes and apply materials to those. However, collecting large-scale object mesh data covering a wide range of object categories is challenging. Moreover, mapping the object meshes to the related materials (*e.g.*, using ceramic materials for teapots) would result in a small dataset. Thus, applying random materials to object meshes is a reasonable strategy. We notice that applying random material maps to complex-shaped object meshes would result in distorted texture or tiling artifacts. Because of these numerous challenges, we choose to randomize object shapes to synthesize large-scale data. Following Xu *et al*. [186], a randomly chosen material is applied to 9 different shape primitives such as spheres, cones, cylinders, tori, *etc*. We randomly choose 6 to 7 material-mapped primitive shapes and place them randomly to assemble a scene. Sample object shape primitives are shown in Fig. 3.4. This strategy is similar to Domain Randomization (DR) [158] that is shown to be useful in high-level semantic

tasks such as object detection [160]. Here, we demonstrate the use of DR for the low-level yet complex task of SVBRDF and shape estimation. For simplicity, we refer to our material-mapped and geometry randomized object shapes as DR objects. Fig. 3.4 shows sample primitive shapes, materials and resulting DR objects with GT shape and SVBRDF parameters.

**High Dynamic Range (HDR) Illumination.** For environment illumination, we collect 285 HDR illumination maps from [190]. These maps are images in latitude-longitude format wrapped on the inside of a sphere, which acts as a light source for the DR object.

**Rendering.** We use the Mitsuba [70] renderer to create two-shot flash and no-flash images of a DR object illuminated with a randomly chosen illumination. In total, the DR dataset contains 100K generated scenes. Note that each DR object consists of differently sampled primitive shapes, and the distance of the closest surface from the camera varies across different DR objects.

Rendering these domain randomized shapes in a realistic setup for real-world usage is crucial for a successful domain transfer. We achieve this by closely following the real-world scenarios in our rendering setup. The camera is positioned randomly on a sphere with a radius of 70cm from the origin. The objects are constructed at the origin, and due to the random translation, rotation and scaling can grow up to 17cm distance from the camera. For real-world capture, this is a reasonable distance between an object and a mobile phone. The camera view is rotated towards the origin always to have the object in focus.

The flash-light is approximated as a point light source and positioned in a 2cm radius around the camera with a flash strength of 45 Lumen, which are typical settings of smartphone cameras and flashes.

We separately rendered two HDR images with only flash and only environment illumination for the flash image and linearly combined these two HDR renderings to obtain the final flash image. This strategy allows us to randomly vary the flash strength by using randomly sampled weights for a linear combination of the ambient and flash rendering. As the network receives LDR input images, we perform a Saturation Based Sensitivity auto exposure calculation [1].

In addition to the two-shot flash and no-flash images, we also render another flash image with only direct illumination. This direct illumination flash image is used to additionally supervise the SVBRDF network after differentiable rendering (Sec. 3.2.3.3). This direct illumination-only image is solely used for training supervision and is not required for inference. Besides, we render GT depth, normals, diffuse albedo, specular albedo, and roughness maps using Mitsuba [70]. The maps are used for direct network supervision. Fig. 3.4 shows samples from this dataset.

### 3.2.5 Results

We evaluate our approach on synthetic and real-world datasets and compare it with several baseline techniques. In this section, we present quantitative and qualitative results and refer to the appendix A.2 for further visual results and comparisons.

**Test datasets.** We quantitatively validate the proposed method on synthetic data with realistic object shapes and SVBRDF and qualitatively on a real-world two-shot image dataset [8]. Images of both of these datasets are unseen during network training. For synthetic test data, we collected 20 freely available, fully textured 3D objects with realistic shapes and materials [35]. These objects are rendered using the Mitsuba renderer [70] with unseen HDR illumination maps. Fig. 3.5 and Fig. 3.6 show samples of two-shot input images of our synthetic test dataset.

For real-world evaluation, we use two-shot images from the recent 'flash and ambient illuminations dataset' from [8]. We have created foreground object masks on several samples from the 'Objects' and 'Toys' categories, as these fit the single object assumption. This dataset does not contain ground truth BRDF parameters. However, the visual quality can be inspected on the estimations and re-renderings with different camera views and illuminations.

**Metrics.** To evaluate the quality of the shape and SVBRDF predictions, we mainly use metrics that directly compare the GT and predictions. A Mean Squared Error (MSE) is a fitting candidate for the depth and normal estimations. For methods that produce relative depth or depth in a different scale, we enable fair comparisons with a Scale-Shift Invariant Metric as in [86]. The exact loss is also applied for intrinsic image decomposition methods, as the predicted diffuse color is not subject to an absolute scale as the diffuse albedo parameter is. To achieve the scale and shift in-variance we define it as:

$$\mathcal{L}(x, x^{\mathrm{gt}}) = \arg\min_{\alpha,\beta} \frac{1}{2D} \sum_{i=1}^{D} (\alpha x_i + \beta - x_i^{\mathrm{gt}})^2 \tag{3.4}$$

where $\alpha$ accounts for the scale and $\beta$ for the shift, $D$ for the image dimension, $x$ for the predicted result and $x^{\mathrm{gt}}$ for the corresponding ground truth. For the scale-invariant loss, only the $\alpha$ is optimized.

For SVBRDFs, no clear metric aligns with the human perception of materials. Following previous works, we also use the MSE metric on SVBRDF prediction maps.

#### 3.2.5.1 Ablation Study

Within our framework, we empirically evaluate different choices we make in our network design.

| Method | Diffuse | Specular | Roughness | Normal | Depth |
|---|---|---|---|---|---|
| MiDaS [86] | NA | NA | NA | NA | [0.006] |
| SIRFS [14] | [0.033] | NA | NA | 0.089 | [0.021] |
| RAFII [124] | **[0.018]** | NA | NA | NA | NA |
| Li *et al*. [98] | 0.160/[0.019] | NA | 0.072 | 0.034 | [0.024] |
| Ours-JN | 0.065/[0.022] | 0.053 | 0.064 | 0.025 | [0.005] |
| Ours-CN | **0.060/[0.018]** | **0.047** | **0.061** | **0.021** | **[0.004]** |

Table 3.2: **State-of-the-art comparison.** The MSE on a sample dataset of 20 unseen objects. Scale and shift-invariant metrics are shown in [ ] brackets where it applies. For the diffuse color, this metric is only scale-invariant.

**Cascade *vs*. Joint Network.** We compare our cascaded network with a single large joint network that estimates all the shape and SVBRDF parameters. For a fair comparison, we design a joint (JN) that has a comparable number of network parameters as our cascaded network (CN) ('Ours-CN' *vs*. 'Ours-JN'). The JN follows the U-Net [135] architecture. Tab. 3.2 shows the quantitative comparisons between them. Results indicate that the CN consistently outperforms JN on a significant margin of both SVBRDF and shape estimations. This empirically underlines the usefulness of our guided stage-wise estimation and joint refinement compared to using a single large network for joint SVBRDF and shape estimation.

**Merge *vs*. Standard Convolutions for Shape Estimation.** Another technical innovation in this work is using MergeConv blocks (Sec. 3.2.3.1) in the shape estimation network instead of standard convolution. Overall the depth estimation error decreased from a MSE of 0.021 to 0.016 and the normal MSE from 0.026 to 0.021.

#### 3.2.5.2 Comparisons with state-of-the-art

As per our knowledge, we are the first work that uses two-shot images as input and does complete SVBRDF estimation, including specular color and shape estimation for objects. Most existing closely related techniques usually use a single flash image as input and either work only on flat surfaces [44, 45, 93, 97] or do not estimate the specular color [98]. Although our approach features a unique setting, we perform the comparisons with SIRFS [14], Li *et al*. [98], and RAFII [124] on SVBRDF and shape estimation. SIRFS [14] uses a no-flash single image as input and predicts diffuse albedo, shading, and shape using an optimization-based approach. RAFII [124] uses a single non-flash image to perform the intrinsic decomposition. Based on a single flash image Li *et al*. [98] is a recent deep learning approach that predicts diffuse albedo, roughness, normal, and depth maps.

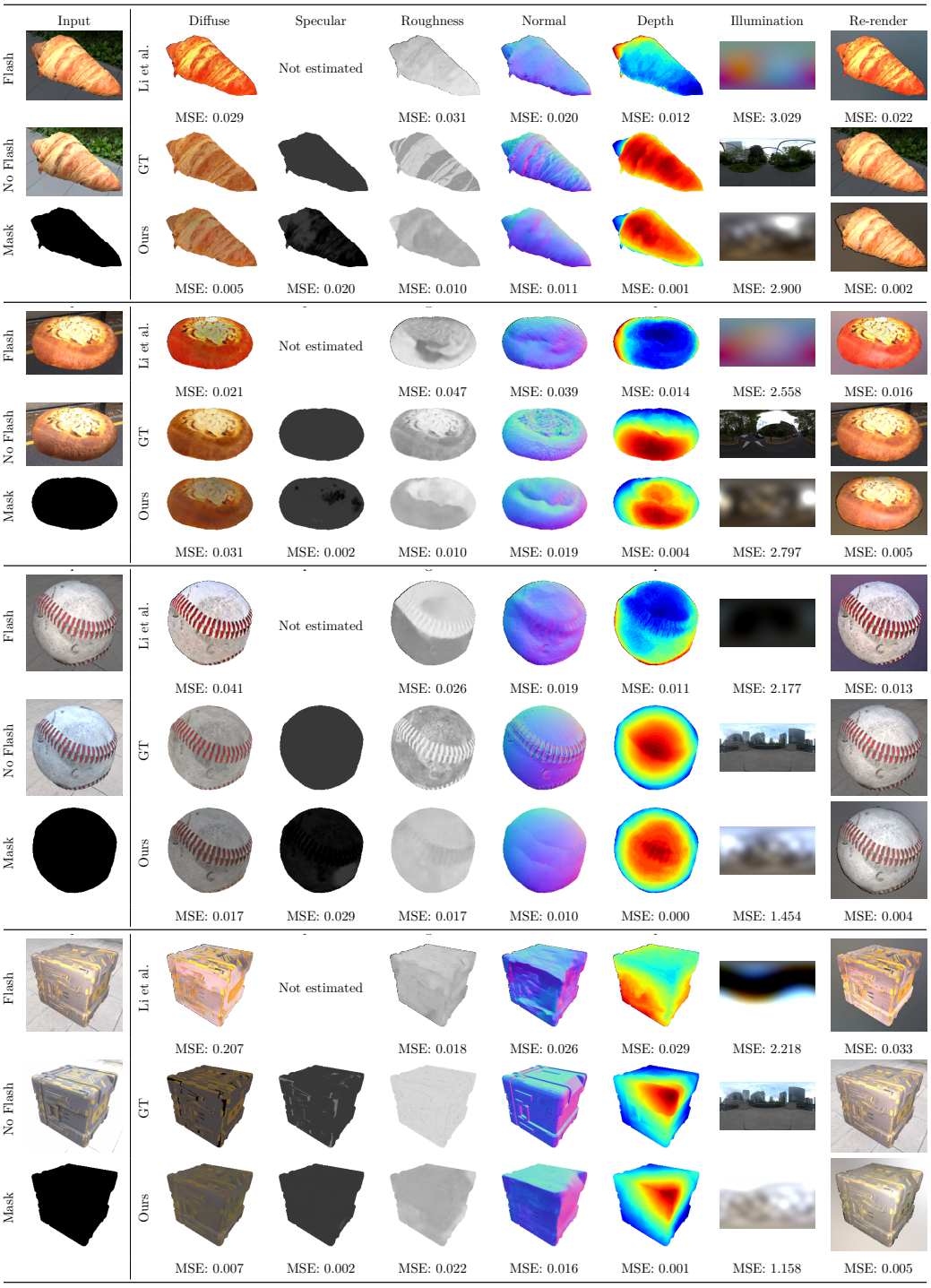



Figure 3.5: **Comparison with Li *et al.* [98].** Ours estimates the diffuse, depth and normal more accurately in particular.

Quantitative results on the 20 objects synthetic test dataset shown in Tab. 3.2 demonstrate the superior performance of our approach (Ours-CascadeNet) compared to both SIRFS [14] and Li *et al*. [98]. Since SIRFS predicts diffuse albedo only up to a scale factor, we also report scale-invariant MSE scores on diffuse albedo. Fig. 3.5 shows a visual comparison with Li *et al*. [98]. Our estimations are also visually closer to GT. Primarily, we can observe apparent visual differences in predicted diffuse albedos where the light information is separated much better in our result. Furthermore, the general shape of the object in the normal map of our method follows the contour of the croissant, while the method of Li *et al*. predicts a mostly flat shape. The details in the roughness and normal map, on the other hand, are not perfectly predicted by either method.

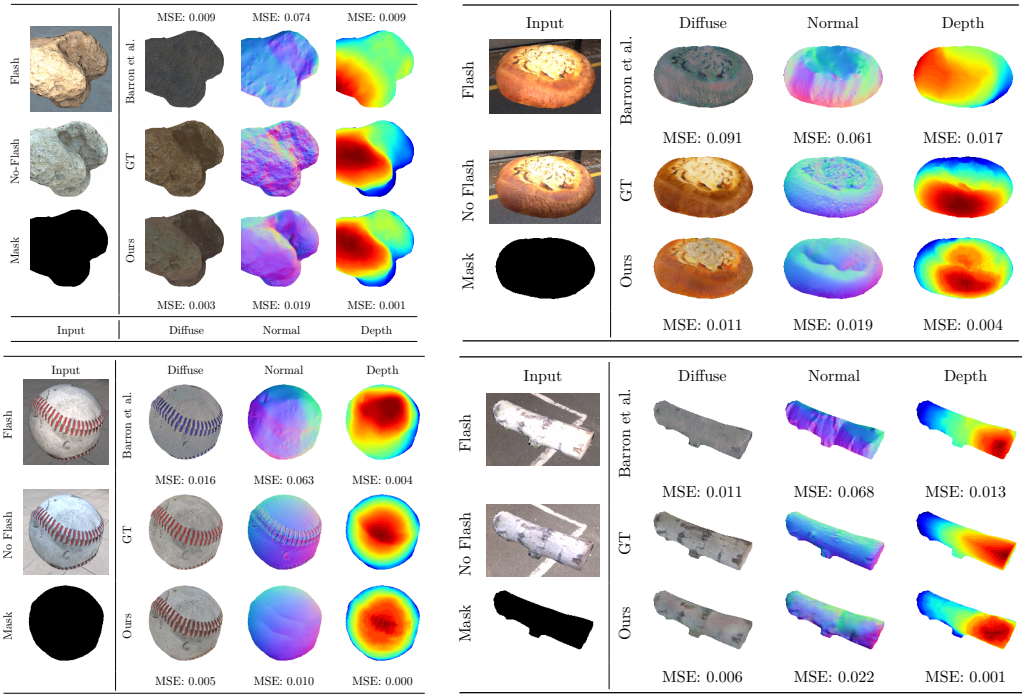



Figure 3.6: **Comparison with Barron *et al.* [14] (SIRFS).** Barron *et al*. does not estimate specular and roughness parameters.

Fig. 3.6 shows a visual comparison with SIRFS, where we again observe our method predictions to be closer to GT. Here, the improvements in the diffuse and normal maps are apparent. The SIRFS method fails in this example to separate shape from shading.

Comparisons with Nestmeyer *et al*. [124] are shown in Fig. 3.7. Here, we want to highlight that our method tackles a complex BRDF model, which is more difficult to disentangle than the intrinsic imaging model of Nestmeyer *et al*. Due to this, we only compare, similar to Barron *et al*., with the scale-invariant diffuse color. Our method can reconstruct the diffuse color either with equal quality or surpass the prediction quality of Nestmeyer *et al*. Especially texture details are preserved better in our method.

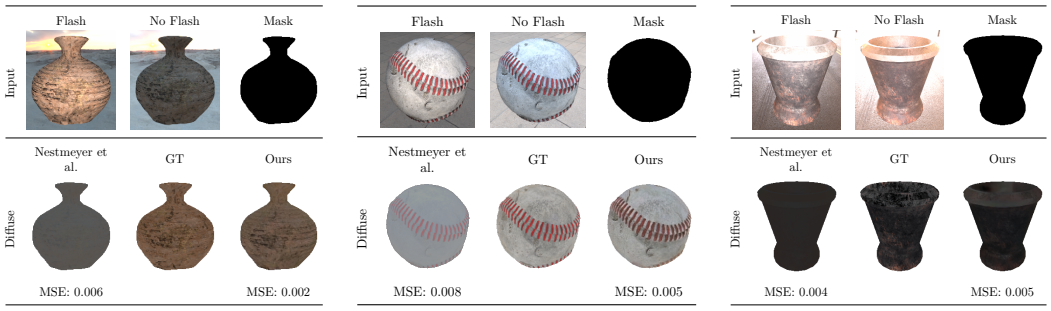



Figure 3.7: **Comparison with Nestmeyer *et al.* [124] (RAFII).** Nestmeyer *et al.* only estimate the diffuse albedo. Our prediction estimate the diffuse component more accurately.

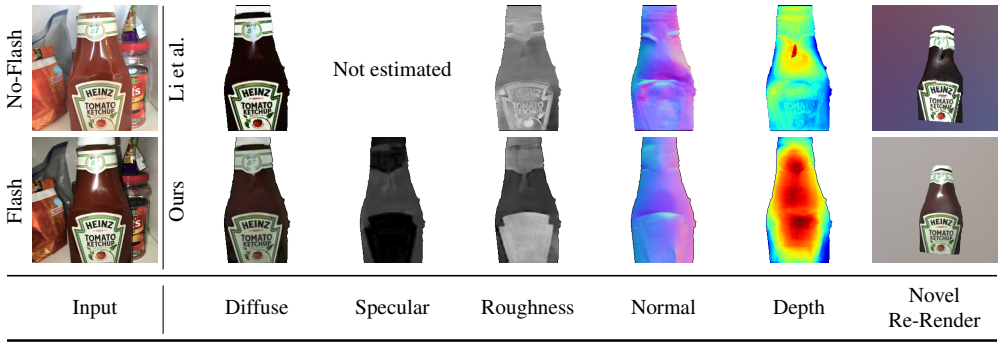



Figure 3.8: **Real-world comparison.** Comparison with Li *et al.* [98] on a real-world sample from [8].

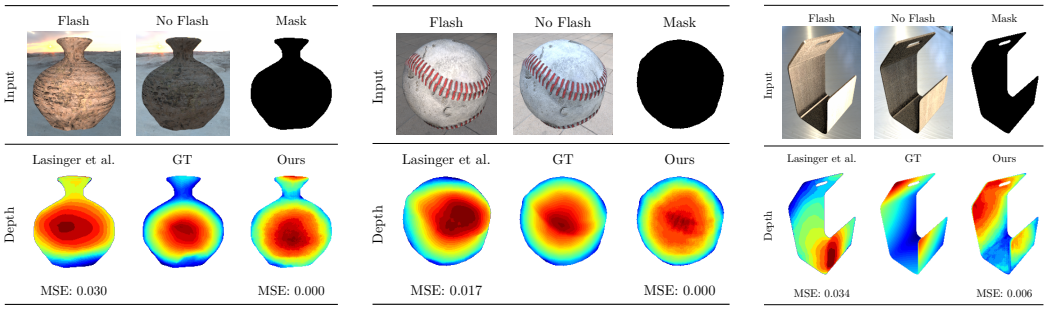



Figure 3.9: **Comparison with Lasinger *et al.* [86] (MiDaS).** Lasinger *et al.* only estimate the depth from a single monocular image. Our method captures the general shape in greater detail and more accurately.

For evaluating depth prediction, we compare our depth estimates against those from a new state-of-the-art monocular depth network of MiDaS [86]. MiDaS is trained with several existing depth datasets and is quite robust to different scene types. As described in Sec. 2.3.2, monocular depth estimation can only predict a relative depth. For compar-

isons, a scale-shift invariant MSE metric is used. Tab. 3.2 shows the results indicating better depth estimations using our approach and a visual comparison is shown in Fig. 3.9.

A visual comparison between Li *et al*. [98] on a real-world example from Yagiz *et al*. [8] is shown in Fig. 3.8. Our method seems to capture the object's color and shape better. The shape from Li *et al*. is predicted as a nearly flat surface. This is apparent in the novel re-rendering. Our predicted normal map is smoother with fewer artifacts and closely follows the bottle shape.

Further visual comparisons are available in the appendix A.2.


Figure 3.10: **Mobile capture and inference.** A result of our mobile application that performs two-shot image capture followed by SVBRDF and shape estimation.

#### 3.2.5.3 Mobile capture and inference

To further showcase our real-world performance, Fig. 3.10 presents an example captured with our mobile application. As seen, most parameters are plausible. However, the lid on top of the electric kettle is estimated slightly too far away in the depth map. This can be attributed to the 'deep is dark' ambiguity. Here, we want to point out an additional challenge from an unknown mobile camera capture pipeline. A RAW image capture would avoid most unknown image pre-processing in modern cameras.

## 3.3 Future Work

In the future, we would like to tackle the SVBRDF estimation of more complex mirror-like objects by incorporating reflection removal techniques. Another interesting research topic is accounting for the effect of inter-reflections or shadowing. If these effects are handled accurately, an increase in the decomposition quality is highly likely. While our BRDF can reproduce a wide range of natural objects, anisotropic or subsurface scattering BRDF models can enable even more objects such as brushed metals, human skin, marble, wax, or leaves. These extensions to the BRDF model introduce several new research challenges, and especially for subsurface scattering, a 3D representation is required as the volume has to be known besides the depth. This creation of a 3D representation is also a crucial topic, as it enables a higher degree of novel view synthesis than the depth-based shape of this chapter. We propose a potential method to create this 3D

reconstruction in the Chapter 4. However, the method does not leverage statistical priors from large datasets as in this chapter. An interesting approach would be fusing multiple views from a video similar to Luo *et al*. [107]. They propose to use a pre-trained network as a starting point and further optimize it per scene to be consistent with other views. A similar process for our two-shot method would increase the decomposition quality. This increase is due to more measurements from different viewing angles, and the depth information can be fused to a single object representation.

# Chapter 4

# Per-Object Multi-Shot Decomposition

Compared to methods discussed in Chapter 3, this chapter introduces methods trained for each scene or object. Here, the setup is more akin to traditional optimization methods than deep learning, where statistical priors are extracted from large datasets and condensed into network weights.

One advantage of these optimization-based methods is that no ground truth supervision is required. The previous chapter leveraged large datasets with known BRDF, shape, and illumination. Creating these datasets is challenging, even for synthetic datasets. As discussed in Sec. 2.1.2.1, BRDF measurements are incredibly complex and time-consuming to create for real-world objects.

The input of the methods in this chapter is multiple images of objects. These images can even be under varying illuminations. The result is a 3D reconstruction of the object with BRDF texture maps. As this problem is highly ill-posed and underconstrained, we solve this by requiring around 50-100 images, and the optimization has to create a shape, BRDF, and per-image illumination to fulfill all observations as close as possible. This also means that each method is trained for every new scene in this chapter.

Novel views and illuminations can be generated by rendering the optimized volume or leveraging the extracted mesh. With the extracted meshes, real-time rendering is easily possible without losing quality.

## 4.1 Related Work

This chapter is based and linked to several fields such as *Neural Fields*, camera pose optimization or estimation, and BRDF and illumination estimation. As these fields are extensive, we discussed the related works in this chapter.

**Neural Fields** allow spatial information to be stored within the weights of a neural network, thereby allowing the retrieval of information solely by querying coordinates [39, 116, 127, 155]. The general concept of these networks is described in detail in Sec. 2.4.1. These methods have been combined with neural volume rendering [104] from Sec. 2.4.2 to enable photorealistic results on novel view synthesis, well-exemplified

by NeRF [117]. In NeRF, a coordinate-based model is used to model a field of volumetric density and color, and renderings are produced by ray-marching through that neural volume. A detailed explanation of this method is given in Sec. 2.4.3. Recent works leverage this neural volume rendering to achieve photorealistic view synthesis results with view-dependent appearance variations. Rapid research in neural fields followed, which alternated the surface representations [126, 172], provided general improvements to the method [15, 168], reduced the long training times [37, 102, 120, 141, 173] and inference times [25, 65, 81, 102, 120, 189], enabled extraction of 3D geometry and materials [25, 121, 193], added generalization capabilities [36, 173, 199] or enabled relighting of scenes [21, 84, 110, 121, 150, 193, 194, 198].

Bi *et al*. [21] enable a decomposition from images with a co-located camera flash in a perfectly dark environment. While the results are convincing, the capture setup is highly limiting. More practical capture setups exists such as PhySG [194], NvDiffRec [121], NeRS [193], NeRFactor [198], NeRV [150]. However, they only allow a decomposition under a single illumination or even require known illumination (NeRV). On the other hand, our methods also allow varying illuminations besides the fixed illumination capture setup. Enabling varying illumination is critical for online image collections, as each image is under different illumination. In SAMURAI of Sec. 4.4, we even extend the decomposition quality to datasets in various locations. This is highly challenging, as all previous techniques – except NeRS [193] – require near-perfect camera poses. Often COLMAP [145, 146] is leveraged to estimate the poses, but we found that it is not capable of estimating the poses in these challenging conditions. SAMURAI can decompose these datasets due to a joint camera optimization from quadrant-based coarse poses.

**Joint camera and shape estimation** is a complex task. An accurate shape reconstruction is only possible with accurate poses and vice versa. Often techniques rely on correspondences across images to estimate camera poses [145, 146]. Recently, several methods combined camera calibration with a joint neural volume training. Jeong *et al*. [74] (SCNeRF) rely on correspondences. BARF [100] proposes a coarse to fine optimization using a varying number of Fourier frequencies during the NeRF rendering of Sec. 2.4.3. Additionally, the optimization requires rough camera poses, which are not too far from the actual position. NeRF-- [176] requires training the neural volume twice while keeping the previous camera parameter optimization. GNeRF [115] proposes using a discriminator on randomly sampled views to learn a pose estimation network on synthesized views jointly. Over time the pose estimation network can estimate the real camera poses, which can then be used for the full neural volume training. NeRS [193] deforms a sphere to a specific shape using coordinate-based MLPs and converts the deformation field to a mesh; while also optimizing camera poses and single illumination.

**BRDF estimation** is a challenging research problem that aims to estimate the appearance of a material. A general overview of techniques in this field is given in Sec. 3.1.

The presented methods do not estimate the shape or only estimate the depth for the specific perspective. This is limiting, as only a small degree of novel view synthesis can be performed.

These capture setups can be extended to estimating the BRDF and shape of objects [21, 22, 23, 29, 79, 122, 140, 192] or scenes [94, 147]. Most of the methods are based on known active illumination. A limited number of light sources – often a single – are assumed to be responsible for the majority of illumination in a scene. Relying on only natural and uncontrolled illumination adds additional challenges due to the drastically increased ambiguity across the shape, illumination, and BRDFs. Often these challenges are reduced by keeping the specular albedo non-spatially-varying or by removing it entirely [93, 188, 194]. Other approaches require temporal traces and limit the casual capture setup [48, 184].

In comparison, our works estimate the full shape from 360-degree images under varying illuminations with a full SVBRDF model. The results of our works can be directly leveraged in traditional rendering methods and accurately capture the objects.

**Illumination estimation** from a single image is an inherently challenging problem. The task is inherently linked to BRDF estimation, as illumination affects the appearance and is only indirectly observable from its interactions with surface materials.

We propose to solve the two tasks in conjunction in this chapter. In NeRD of Sec. 4.2, we decompose a single object into shape, reflectance, and a global set of SGs. NeRD is similar to PhySG [194], which performs this under a single illumination.

Chen *et al*. [38] leverage a deep prior of environment maps with homogeneous materials, using an invertible neural BRDF model. Li *et al*. [94] decompose an entire scene into a simplified BRDF model with hemispherical SGs per point in the scene. The image of the environment in the background may be incorporated into the prediction, shifting the problem to completion of the HDR environment map from sparse observations [58, 149, 177]. In Sec. 4.3, we not only learn a deep prior but a rendering-aware network capable of integrating the environment illumination for a specific surface roughness enabling rendering the entire hemisphere of incoming light with a single evaluation.

**Placement in literature.** The area of neural fields, BRDF estimation, and camera pose optimization is extensive. We compare our approaches with the recent literature in Tab. 4.1.

Here, we specifically compare in the context of relighting, training on in-the-wild datasets with various illumination, camera pose initialization, optimization capabilities, and mesh and texture extraction for fast real-time rendering. We propose one of the few methods capable of working with in-the-wild datasets under varying illumination and enabling textured mesh extraction. Furthermore, SAMURAI even allows camera pose optimization.

Our methods also enable relighting under any illumination compared to NeRF-w [110],

| Method | Relightable | Varying illumination | Initial Poses | Pose optimization | Mesh | Textures | Note |
|---|---|---|---|---|---|---|---|
| NeRF [117] | ✗ | ✗ | COLMAP | ✗ | ✓ | ✗ | |
| Instant-NGP [120] | ✗ | ✗ | COLMAP | ✗ | ✓ | ✗ | Fast training & Inference |
| TensoRF [37] | ✗ | ✗ | COLMAP | ✗ | ✓* | ✗ | Fast training & Inference |
| NeRF-w [110] | ✓(within datasets) | ✓ | COLMAP | ✗ | ✓* | ✗ | |
| BaRF [100] | ✗ | ✗ | Quadrant/Origin | ✓ | ✓* | ✗ | Mostly for forward facing scenes |
| GNeRF [115] | ✗ | ✗ | Random | ✓ | ✓* | ✗ | |
| NeRF-- [176] | ✗ | ✗ | Origin | ✓ | ✓* | ✗ | Mostly for forward facing scenes |
| SCNeRF [74] | ✗ | ✗ | Origin | ✓ | ✓* | ✗ | Requires correspondences |
| NeRV [150] | ✓ | ✗ | COLMAP | ✗ | ✓* | ✗ | Handles shadowing |
| NeRS [193] | ✓ | ✗ | Quadrant-based | ✓ | ✓ | ✓ | |
| PhySG [194] | ✓ | ✗ | COLMAP | ✗ | ✓* | ✓* | |
| NeRFFactor [198] | ✓ | ✗ | COLMAP | ✗ | ✓* | ✗ | Requires NeRF training |
| NVDiffRec [121] | ✓ | ✗ | COLMAP | ✗ | ✓ | ✓ | |
| NeRD [25] (Sec. 4.2) | ✓ | ✓ | COLMAP | ✗ | ✓ | ✓ | |
| Neural-PIL [28] (Sec. 4.3) | ✓ | ✓ | COLMAP | ✗ | ✓* | ✓* | |
| SAMURAI [26] (Sec. 4.4) | ✓ | ✓ | Quadrant-based | ✓ | ✓ | ✓ | |

Table 4.1: **Literature overview.** Compared with recent literature, our proposed methods enable easy relighting under any illumination and training on varying illumination datasets. SAMURAI can even decompose an object with coarse initialized camera poses. We also enable the extraction of relightable meshes, which can be rendered in real-time. If a method does not propose an extraction, we mark it with * to indicate if it is potentially possible.

which only can interpolate illuminations based on the seen lighting conditions in the dataset. Our explicit BRDF decomposition can be placed in any illumination and behaves plausibly.

## 4.2 NeRD: Neural Reflectance Decomposition

*This section is based on the publications*:

NeRD: Neural Reflectance Decomposition from Image Collections

Mark Boss, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, Hendrik P. A. Lensch

*IEEE International Conference on Computer Vision (ICCV)* - 2021

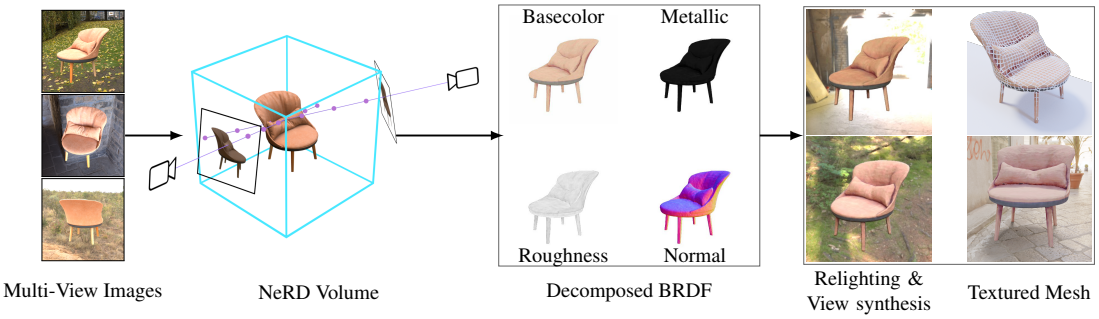



Figure 4.1: **Neural reflectance decomposition for relighting.** We encode multiple views of an object under varying or fixed illumination into the NeRD volume. We decompose each given image into geometry, spatially-varying BRDF parameters, and a rough approximation of the incident illumination in a globally consistent manner. We then extract a relightable textured mesh that can be re-rendered under novel illumination conditions in real-time.

Traditional SVBRDF estimation techniques involve capturing images using a light-stage setup where the light direction and camera view settings are controlled [11, 27, 87, 88, 89]. In Chapter 3 a method for reconstruction objects under environment illumination is introduced. This approaches and several other recent ones enable SVBRDF estimation from more practical capture setups [21, 22, 23, 56, 122]. However, often the illumination is limited to a single dominant source (*e.g.*, a flash attached to a camera), or they assume that the flash predominantly lights the scene. In Sec. 3 this assumption is true to a lesser extent due to our two-shot capture scenario. When the illumination is either known or severely constrained, the ambiguity of shape and material estimation is reduced. However, the practical utility is also limited to laboratory settings or flash photography in dark environments.

Additionally, in our previous work of Sec. 3, only a singular viewpoint is considered. As only the shape for that particular perspective is estimated, only a small degree of novel view synthesis is possible.

In contrast to standard SVBRDF and shape estimation techniques, recently introduced coordinate-based scene representation networks [110, 117, 195], can directly perform high-quality view synthesis without explicitly estimating shape or SVBRDF. An overview of these techniques is given in Sec. 2.4.

They represent the radiance field of the scene using a neural network trained specifically for a single scene, using as input multiple images of that scene. These neural networks directly encode the geometry and appearance as volumetric density and color functions parameterized by 3D coordinates of query points in the scene. Realistic novel views can be generated by raymarching through the volume. Though these approaches can reproduce view-dependent appearance effects, the radiance of a point in a direction is "baked in" to these networks, making them unusable for relighting. Even if such techniques could be extended to relighting, the rendering speed of these methods limits their practicality — the time required by NeRF to generate a single view is about 30 seconds [117].

In this section, we combine the SG illumination of Sec. 3 with these recent *Neural Fields*. This work presents a shape and SVBRDF estimation technique that allows for a more flexible capture setting while enabling relighting under novel illuminations.

Our key technique is an explicit decomposition model for shape, reflectance, and illumination within a NeRF-like coordinate-based neural representation framework [117]. Compared to NeRF, our volumetric geometry representation stores SVBRDF parameters at each 3D point instead of radiance. Each image is then differentiably rendered with a jointly optimized spherical Gaussian illumination model (see Figure 4.1). Shape, BRDF parameters, and illumination are optimized simultaneously to minimize the photometric rendering loss w.r.t. each input image. We call our approach "Neural Reflectance Decomposition" (NeRD).

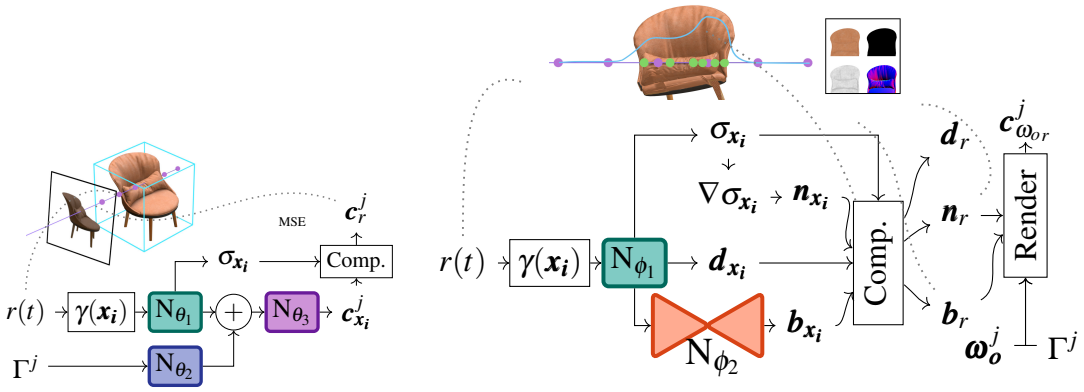
NeRD not only enables simultaneous relighting and view synthesis but also allows for a more flexible range of image acquisition settings: Input images of the object need not be captured under the same illumination conditions. NeRD supports both camera motion around an object as well as captures of rotating objects. All NeRD requires as input is a set of images of an object with known camera pose (computed for *e.g.* using COLMAP [145, 146]), where a foreground segmentation mask accompanies each image. Besides the SVBRDF and shape parameters, we also explicitly optimize the illumination corresponding to each image for varying illuminations or globally for static illumination.

As a post-processing step, we propose a way to extract a 3D surface mesh along with SVBRDF parameters as textures from the learned coordinate-based representation network. This allows for a highly flexible representation for downstream tasks such as real-time rendering of novel views, relighting, 3D asset generation, *etc*.

Our method optimizes a model for shape, BRDF, and illumination by minimizing the photometric error to the input image collection of an object captured under fixed or different illuminations.

### 4.2.1 Problem Setup

Our input consists of a set of $q$ images with $s$ pixels each, $I_j \in \mathbb{R}^{s\times 3}; j \in 1,...,q$ potentially captured under different illumination conditions. We aim to learn a 3D volume $\mathcal{V}$, where at each point $\boldsymbol{x} = (x,y,z) \in \mathbb{R}^3$ in 3D space, we estimate BRDF parameters




(a) **Sampling network.** The main task of the coarse sampling network is to generate a finer distribution for sampling in the decomposition network. The color prediction needs to account for the illumination to match the input during training. We combine a compacted $\Gamma^j$ from $N_{\theta_2}$ with the latent color output of $N_{\theta_1}$ to generate the illumination-dependent color in $N_{\theta_3}$.

(b) **Decomposition network.** With the sampling pattern generated from the coarse network, we perform SVBRDF decomposition at each point in the neural volume. The density, $\sigma$, and direct RGB color $\boldsymbol{d}$ is queried from the $N_{\phi_1}$. Additionally, a vector is passed to $N_{\phi_2}$, which decodes it to the point's BRDF parameters $\boldsymbol{b}_{\text{bm}}$. By compressing the BRDF to a low-dimensional latent space, all surface points contribute to training a joint space of plausible BRDFs for the scene. Each point still interpolates its parameters in this space. The gradient from the density forms the normal $\boldsymbol{n}$ and is passed with the BRDF and SGs $\Gamma^j$ to the differentiable renderer.

Figure 4.2: **NeRD architecture.** The architecture consists of two networks. Here, $N_{\theta_1}/N_{\phi_1}$ denote instances of the main networks which encodes the neural volume. $r(t)$ defines a ray with sampling positions $\boldsymbol{x_i}$, $\gamma(\boldsymbol{x})$ is the Fourier Embedding [117], and $\boldsymbol{\Gamma}^j$ denotes the SG parameters per image $j$. $\boldsymbol{c}$ is the output color and $\sigma$ is the density in the volume. The individual samples along the ray need to be alpha composed based on the density $\sigma$ along the ray. This is denoted as "Comp.".

$\boldsymbol{b}_{\text{bm}} \in \mathbb{R}^5$, surface normal $\boldsymbol{n} \in \mathbb{R}^3$ and density $\sigma \in \mathbb{R}$. The environment maps are represented by SG mixtures with parameters $\Gamma \in \mathbb{R}^{24\times 7}$ (24 lobes). Here, we follow the rendering formulation of Sec. 2.2.1 and leverage the volumetric rendering described in Sec. 2.4.3.

## 4.2.2 Network Architecture

Compared to NeRF described in Sec. 2.4.3, the architecture of NeRD mainly differs in the second fine network. NeRD uses the *decomposition network* as the fine network, which stores the lighting-independent reflectance parameters instead of the direct view-dependent color. Also, the sampling network in NeRD differs from NeRF's coarse network as we learn illumination-dependent colors as NeRD can work with differently illuminated input images. An overview of both networks is shown in Fig. 4.2. The parameters of the networks and the SGs are optimized by backpropagation informed by comparing the output of a differentiable rendering step to each input image $I_j$ for individual rays across the 3D volume.

### 4.2.2.1 Sampling network

The *sampling network* directly estimates a view-independent but illumination dependent color $\boldsymbol{c}^j$ at each point, which is optimized by a MSE: $\frac{1}{s}\sum^s(\hat{\boldsymbol{c}}_r^j - \boldsymbol{c}_r^j)^2$. The sampling network's primary goal is to establish a useful sampling pattern for the *decomposition network*. The sample network structure is visualized in Fig. 4.2a. Compared to NeRF, our training images can have varying illuminations. Therefore, the network needs to consider the illumination $\Gamma^j$ to create the illumination-dependent color $\boldsymbol{c}^j$ that should match image $I_j$. The density $\sigma$ is not dependent on the illumination, which is why we extract it directly as the side-output of $\text{N}_{\theta_1}$. Here, we follow a concept from NeRF-w [110] that combines an embedding of the estimated illumination with the latent color vector produced by $\text{N}_{\theta_1}$. As the dimensionality of the SGs can be large, we add a compaction network ($\text{N}_{\theta_2}$), which encodes the $24 \times 7$ dimensional SGs to 16 dimensions. The compacted SG's embedding is then appended to the output of the last layer of $\text{N}_{\theta_1}$ and jointly passed to the final estimation network $\text{N}_{\theta_3}$ that outputs color values. Without the illumination-dependent color prediction, several floaters would appear in the volume estimate, introducing the wrong semi-transparent geometry to paint in highlights for individual views (see Fig. 4.3b). The resulting 3D volume is sparser and more consistent by introducing the illumination-dependent branch.

### 4.2.2.2 Decomposition network

After a ray has sampled the *sampling network*, additional $m$ samples are placed based on the density $\sigma$. This is visualized in Fig. 4.2b as the additional green points on the ray. The decomposition network is trained with the same loss as the *sampling network*.

However, we introduce an explicit decomposition step and a rendering step in-between. Our decomposition step estimates view and illumination independent BRDF parameters $\boldsymbol{b}_{\text{bm}}$ and a surface normal $\boldsymbol{n}$ at each point. The popular Cook-Torrance analytical BRDF model [41] is used for rendering. Here, we choose the Disney BRDF basecolor and metallic parametrization [32] instead of independently predicting the diffuse and specular color, as it enforces physical correctness. The illumination $\Gamma^j$, in the form of SGs, is jointly optimized. After rendering the decomposed parameters, the final output is a view and illumination-dependent color $\boldsymbol{c}^j_{\omega_{or}}$.

By keeping the rendering differentiable, the loss from the input color $\hat{\boldsymbol{c}}^j_r$ can be back-propagated to the BRDF $\boldsymbol{b}_{\text{bm}}$, the normal $\boldsymbol{n}$, and the illumination $\Gamma^j$. Our rendering step approximates the general rendering equation:

$$L_o(\boldsymbol{x}, \boldsymbol{\omega}_i) = \int_\Omega f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n}) d\boldsymbol{\omega}_i \tag{4.1}$$

using a sum of 24 SG evaluations. The $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_o$ define the incoming and outgoing ray direction, respectively. The reflectance due to diffuse and specular lobes is separately evaluated by functions $\rho_d$ and $\rho_s$, respectively [171]. A detailed explanation of rendering with SGs is shown in Sec. 2.2.1. Overall, our image formation is defined as:

$$L_o(\boldsymbol{x}, \boldsymbol{\omega}_i) \approx \sum_{m=1}^{24} \rho_d(\boldsymbol{\omega}_o, \Gamma_m, \boldsymbol{n}, \boldsymbol{b}_{\text{bm}}) + \rho_s(\boldsymbol{\omega}_o, \Gamma_m, \boldsymbol{n}, \boldsymbol{b}_{\text{bm}}) \tag{4.2}$$

Our differentiable rendering implementation follows Sec. 3.2.

The overall network architecture is shown in Fig. 4.2b. Especially in the early stages of the estimation, joint optimization of BRDF and shape proved difficult. Therefore, we estimate the density $\sigma$ and, in the beginning, a view-independent color $\boldsymbol{d}$ for each point in $\text{N}_{\phi_1}$. The direct color prediction $\boldsymbol{d}$ is compared with the input image, and the loss is faded out over time when the rough shape is established.

The surface normal is required to compute the shading. One approach could be to learn the normal as another output [21]. However, this typically leads to inconsistent normals that do not necessarily fit the object's shape (Fig. 4.3c). Specific reflections can be created by shifting the normal instead of altering the BRDF. As seen in Sec. 3.2.3.1, coupling the surface normal to the actual shape can resolve some ambiguity. In coordinate-based volume representations like NeRF [117], we can establish this link by defining the normal as the normalized negative gradient of the density field:

$$\boldsymbol{n} = -\frac{\nabla_{\boldsymbol{x}}\sigma}{\|\nabla_{\boldsymbol{x}}\sigma\|} \tag{4.3}$$

While the density field defines the surface implicitly, the density in the 3D volume changes drastically at the boundary between non-opaque air to the opaque object. Thus, the gradient at a surface will be perpendicular to the implicitly represented surface. This is a similar to the normal reconstruction from SDFs of Yariv *et al*. [187].

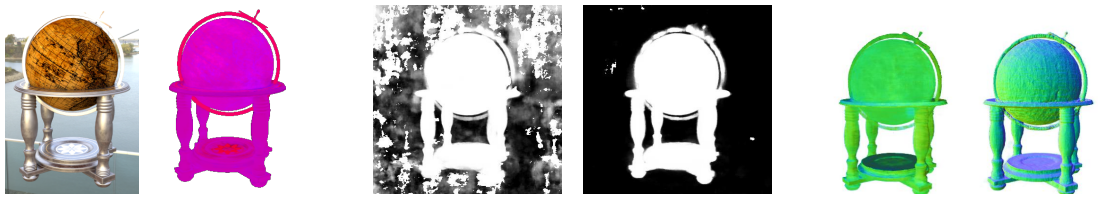



Figure 4.3a: **Compressed BRDF Space.** Instead of directly estimating the BRDF, we learn a 2D embedding per scene which clusters similar materials. As several points jointly estimate BRDFs, this stabilizes the decomposition and improves quality. Notice how similar materials are identified across the surface in the resulting material map.

Figure 4.3b: **SGs dependent Sampling Network.** $N_{\theta_1}$ can to some extent model view-dependence by composition along the ray. However, this is too weak to deal with varying illumination. Apparent highlights introduce spurious geometry that mimics the effect for individual views. We can obtain a better shape by estimating the illumination-dependent radiance with SGs.

Figure 4.3c: **Surface Normal Estimation.** Instead of directly predicting the normal as another output of the $N_{\theta_1}$, the normal in our approach is calculated from the gradient of the density $\nabla\sigma$. Photometric information thus influences both $\boldsymbol{n}$ and $\sigma$ during training.

Figure 4.3: **Novel techniques.** An overview of three selected novel additions.

By calculating the gradient inside the optimization and allowing the photometric loss from the differentiable rendering to optimize the normal, we optimize the $\sigma$ parameter in the second order. Therefore, the neighborhood of surrounding points in the volume is smoothed and made more coherent with the photometric observations. As a more densely defined implicit volume allows for a smoother normal, we additionally jitter the ray samples during training. Each ray is now cast in a subpixel direction, and the target color is obtained by bilinear interpolation.

For the BRDF estimation, we use the property that often real-world objects consist of a few highly similar BRDFs which might be spatially separated. To account for this we introduce an additional network $N_{\phi_2}$ which receives the latent vector output of $N_{\phi_1}$. This autoencoder creates a severe bottleneck, a two-dimensional latent space, which encodes all possible BRDFs in this scene. As the embedding space enforces a compression, similar BRDFs will share the same embedding. This step couples the BRDF estimation of multiple surface points, increasing the robustness. The assignment to various BRDFs is visualized in Fig. 4.3a, which can be utilized for material-based segmentation.

The approach will converge to a globally consistent state, as the underlying shape and BRDF is assumed to be the same for all input images. The SGs are estimated for each input image, but we can force them to be the same or a rotated version of a single SG in case of static illumination.

#### 4.2.2.3 Detailed Architecture

The main network $N_{\theta_1}/N_{\phi_1}$ uses 8 MLP layers with a feature dimension of 256 and ReLU activation. The input coordinate $\boldsymbol{x}$ is transformed by the Fourier output $\gamma(\boldsymbol{x})$ with 10 bands to 63 features. For the *sampling network*, the output from the main network is then transformed to the density $\sigma$ with a single dense layer without any activation. The flattened 192 SGs parameters $\Gamma^j$ are compacted to 16 features using a fully connected layer ($N_{\theta_2}$) without any activation. As the value range can be extensive in real-world illuminations, the amplitudes $\alpha$ are normalized to $[0, 1]$. The main network output is concatenated with the SGs embeddings and passed to the final prediction network. Here, an MLP with ReLU activation first reduces the joined input to 128 features. The final color prediction $c^j$ is handled in the last layer without activation and an output dimension of 3.

The *decomposition network* uses the output from $N_{\phi_1}$ and directly predicts the direct color $d$ and the density in a layer without activation and 4 output dimensions (RGB+$\sigma$). The main network output is then passed to several ReLU activated layers which handle the BRDF compression $N_{\theta_2}$. The feature outputs are as followed: 32, 16, 2 (no activation), 16, 16, 5 (no activation). The final five output dimensions correspond to the number of parameters of the BRDF model. The compressed embedding with two feature outputs is regularized with a $\mathcal{L}_2$ norm with a scale of 0.1 and further clipped to a value range of $-40$ to 40 to keep the value ranges in the beginning stable.

#### 4.2.2.4 Environment representation and rendering.

We follow the rendering definition with SGs defined in Sec. 2.2.1. SGs are useful for representing low-frequency approximations of the environment. Our differentiable rendering step follows the rendering implementation of Sec. 3.2.

#### 4.2.2.5 Dynamic range, tonemapping and whitebalancing

As most online image collections consist of LDR images with at least an sRGB curve and white balancing applied, we have to ensure that our rendering setup's linear output recreates these mapping steps before computing a loss. However, rendering can produce an extensive value range depending on the incident light and the object's specularity. Real-world cameras also face this problem and tackle it by changes in aperture, shutter speed, and ISO. Based on the meta-data information encoded in JPEG files, we can reconstruct the input image's exposure value and apply this to our re-rendering. NeRD is then forced to always work with physically plausible ranges. For synthetic examples, we calculate these exposure values based on Saturation Based Sensitivity auto exposure calculation [1] and apply an sRGB curve.

Cameras also apply a white balancing based on the illumination, or it is set by hand afterward. This can reduce some ambiguity between illumination and material color

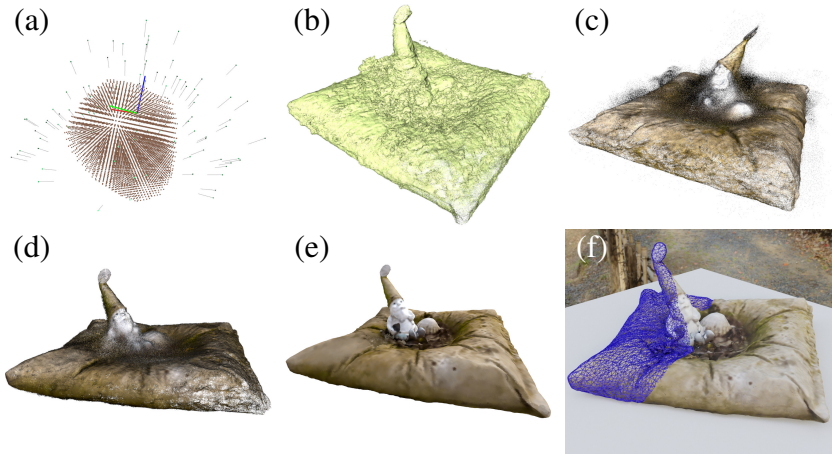



Figure 4.4: **Mesh generation.** Shows the stages of the mesh generation. In (a), the cameras and view frustum are shown. From there, in (b) points are sampled where $\sigma$ was high. The resulting point cloud with the basecolor is then shown in (c). Outliers in the point cloud are then removed in (d). The mesh is then created in (e) where the texture information is applied with vertex colors. Lastly, the final Low-resolution mesh with material textures is shown in (f).

and, in particular, fix the illumination's overall intensity. For synthetic data, we evaluate a small spot of material with 80% gray value in the environment. We assume a perfect white balancing and exposure on real-world data, at least for one of the input images. The RGB color ($\boldsymbol{w}$) of the white point is stored. After each training step a single-pixel with a rough 80% gray material is rendered in the estimated illumination and a factor $\boldsymbol{f} = \frac{\boldsymbol{w}}{\boldsymbol{b}_{\text{bm}}}$ is calculated. This factor is then applied to the corresponding SG. As the training will adopt the BRDF to the normalized SG, a single white-balanced input can implicitly update and correct all other views. In practice, the calculated factor $\boldsymbol{f}$ could abruptly change the SGs in one step, causing unstable training. Therefore, we clip the range of $\boldsymbol{f}$ to $[0.99; 1.01]$ to spread the update over multiple training iterations.

#### 4.2.2.6 Mesh extraction

The ability to extract a consistent textured mesh from NeRD after training is one of the key advantages of the decomposition approach and enables real-time rendering and relighting. This is impossible with NeRF-based approaches where the view-dependent appearance is directly baked into the volume. We use the following four general steps to extract textured meshes:

1. A very dense point cloud representation of the surface is extracted. This step utilizes the same rendering functions used during training, ensuring that the resulting 3D coordinates are consistent with the training. To generate the rays for the rendering step, we sample the *decomposition network* for $\sigma$ in a regular grid within

the view volume determined by the view frustums of the cameras. We construct a discrete PDF from this grid, which is then sampled to generate about 10 million points where $\sigma$ is high. The rays are constructed by following the normals at those points to get the ray-origins. We use the slightly jittered inverted normals as ray directions. See Fig. 4.4 (a) to (c) for visualizations of this step.

2. For meshing, we use the Open3D [200] implementation of the Poisson surface reconstruction algorithm [80] using the normals from NeRD. Before meshing, we perform two cleanup steps: First, we reject all points where the accumulated opacity along a ray is lower than 0.98. Secondly, we perform statistical outlier removal from Open3D. Those steps are visualized in Fig. 4.4 (d) and (e).

3. We use Blender's [40] *Smart UV Project* to get a simple UV-unwrapping for the mesh. Reducing the mesh resolution beforehand is an optional step that reduces the computational burden for using the mesh later. This is also done using Blender via *Decimate Geometry* or the *Voxel Remesher*.

4. We bake the surface coordinates and geometry normals into a floating-point texture of the desired resolution. The textures are generated by generating and rendering one ray per texel to look up the BRDF parameters and shading normals with NeRD. A result is show in Fig. 4.4 (f).

#### 4.2.2.7 Training and losses

The estimation is driven by a MSE loss between the input image and the results of evaluating randomly generated rays. For the *sampling network* this loss is applied to the RGB prediction and for the *decomposition network* to the re-rendered result $\boldsymbol{c}^j$ and the direct color prediction $\boldsymbol{d}$. The loss for the color prediction based on $\boldsymbol{d}$ is exponentially faded out. Additionally, we leverage the foreground/background mask as a supervision signal, where all values along the ray in background regions are forced to 0. This loss is exponentially reduced throughout the training to reduce optimization instabilities. By gradually increasing this loss, the network is forced to provide a more accurate silhouette, which prevents the smearing of information at the end of the training. The networks are trained for 300K steps with the Adam optimizer [82] with a learning rate of 5e$-$4. Per batch, 1024 rays are cast into a single scene. For adjusting the losses and learning rate during the training, we use exponential decay: $p(i;v,r,s) = vr^{\frac{i}{s}}$. The learning rate then uses $p(i;0.000375,0.1,250000)$, the direct color $d$ loss is faded out using $p(i;1,0.75,1500)$ and the alpha loss is faded in using $p(i;1,0.9,5000)$. During the first 1000 steps, we also do not optimize the SGs parameters and first use the white balancing only to adjust the mean environment strength. This step also sets the illumination strength per image based on the exposure values. On 4 NVIDIA 2080 Ti, the training takes about 1.5 days. The final mesh extraction takes approximately 90 minutes.

### 4.2.3 Results

The proposed method recovers shape, appearance, and illumination for relighting in unconstrained settings. Our reconstruction and relighting performance on synthetic sets is measured against ground truth images and known BRDF parameters. We present novel relit views and compare the renderings with validation images excluded from training for real-world examples. If the environment map for the validation image is known, we directly use this for relighting. Otherwise, we recover the unseen illumination by optimizing the SGs through the frozen network in 1000 steps using stochastic gradient descent with a learning rate of 0.1. One-to-one comparisons with previous methods are challenging, as most use different capturing setups. We can, however, compare the outcome of NeRF when trained on a similar scene. NeRF cannot relight the object under novel illumination, and even NeRF-w and our simplified NeRF-A baseline can only interpolate between seen illuminations.

We also perform an ablation study on the influence of our novel training techniques.

#### 4.2.3.1 Datasets

We use three synthetic scenes to showcase the quality of the estimated BRDF parameters. We use three textured models (Globe [162], Wreck [33], Chair [34]) and render each model with a varying environment illumination per image. For a fixed illumination synthetic dataset, we use the Lego, Chair, and Ship scenes from NeRF [117].

We also evaluate using two real-world scenes from the British Museum's photogrammetry dataset: an Ethiopian Head [130] and a Gold Cape [131]. These scenes feature an object in a fixed environment with either a rotating object or a camera. Additionally, we captured our own scenes under varying illumination at various times of day (Gnome, MotherChild). An overview of all datasets and visual examples are shown in the appendix Sec. B.1.

#### 4.2.3.2 BRDF decomposition results

Fig. 4.5 shows exemplary views and decomposition results of the synthetic Car Wreck and Chair scenes. In all cases, the estimated re-renderings are similar to GT. The estimated BRDF parameters may not match perfectly in some places compared to the GT, but given the purely passive unknown illumination setup, they still reproduce the GT images. Causes for deviations are the inherent ambiguity of the decomposition problem and the differences in shading based on SG *vs*. the high-resolution GT environment map.

#### 4.2.3.3 Relighting and novel view synthesis

In Fig. 4.6, novel views and plausible relighting in unseen environments are shown for our real-world data sets. The relighted images are visually close to the held-out validation images. Furthermore, a novel view can be relighted with the lighting from a different


Figure 4.5: **Decomposition on synthetic examples.** Two scenes are highlighted to show the decomposition performance of our method. Notice the accurate performance in relighting with unseen illuminations.

| | | Fixed Illumination | | Varying Illumination | |
|---|---|---|---|---|---|
| | Method | PSNR↑ | SSIM↑ | PSNR↑ | SSIM↑ |
| Syn. | NeRF | **34.24** | **0.97** | 21.05 | 0.89 |
| | NeRF-A | 32.44 | **0.97** | **28.53** | 0.92 |
| | Ours | 30.07 | 0.95 | 27.96 | **0.95** |
| Real | NeRF | 23.34 | 0.85 | 20.11 | 0.87 |
| | NeRF-A | 22.87 | 0.83 | **26.36** | 0.94 |
| | Ours | **23.86** | **0.88** | 25.81 | **0.95** |

Table 4.2: **Novel view synthesis.** Comparison with NeRF and NeRF-A on novel view synthesis (with relighting in varying illumination). Notice NeRF and NeRF-A cannot relight in unseen illuminations, nor is an extraction of a textured mesh from the network easily possible.

view. Note that some fine details are missing in the reconstructions of the Gold Cape, which is caused by minor inaccuracies in the camera registration. Also, the MotherChild model is missing some highlights, especially at grazing angles, which can be attributed to the limitations of the SG based rendering model.

While no ground truth BRDF exists, the estimated parameters for the Gnome (Fig. 4.6)

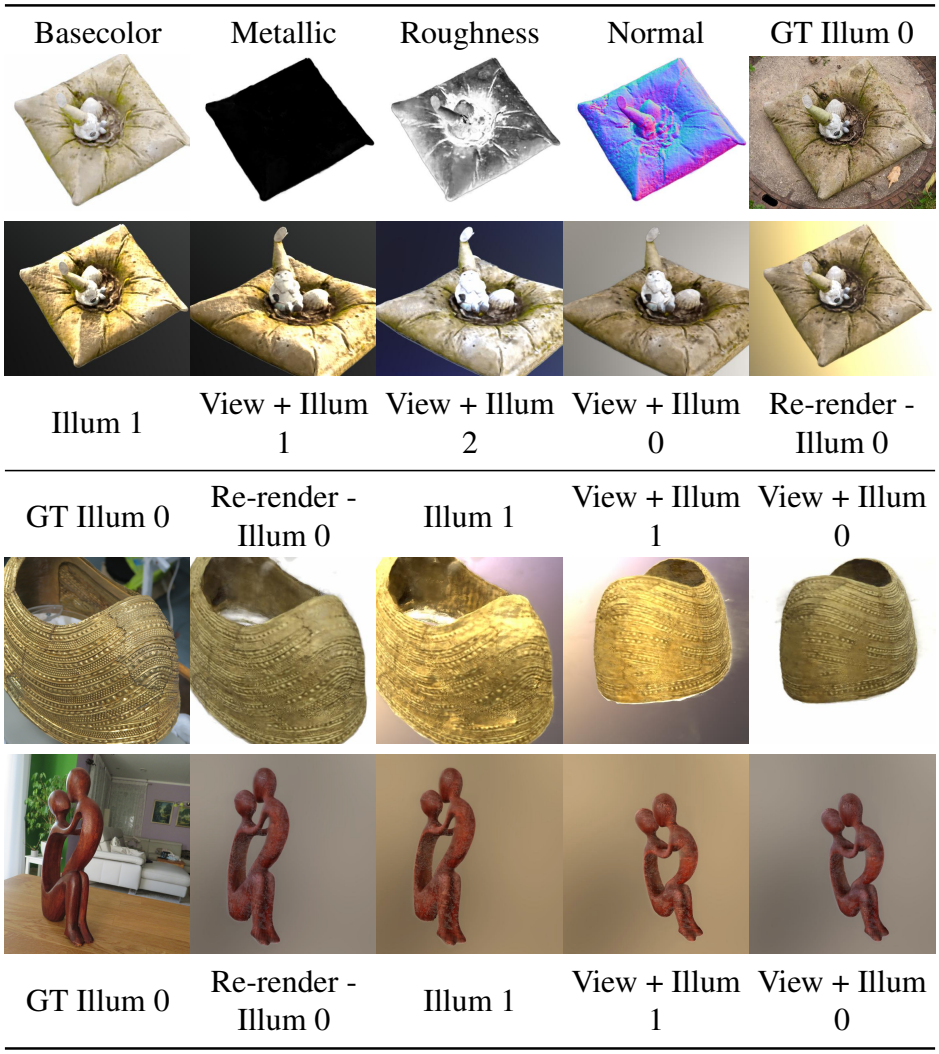
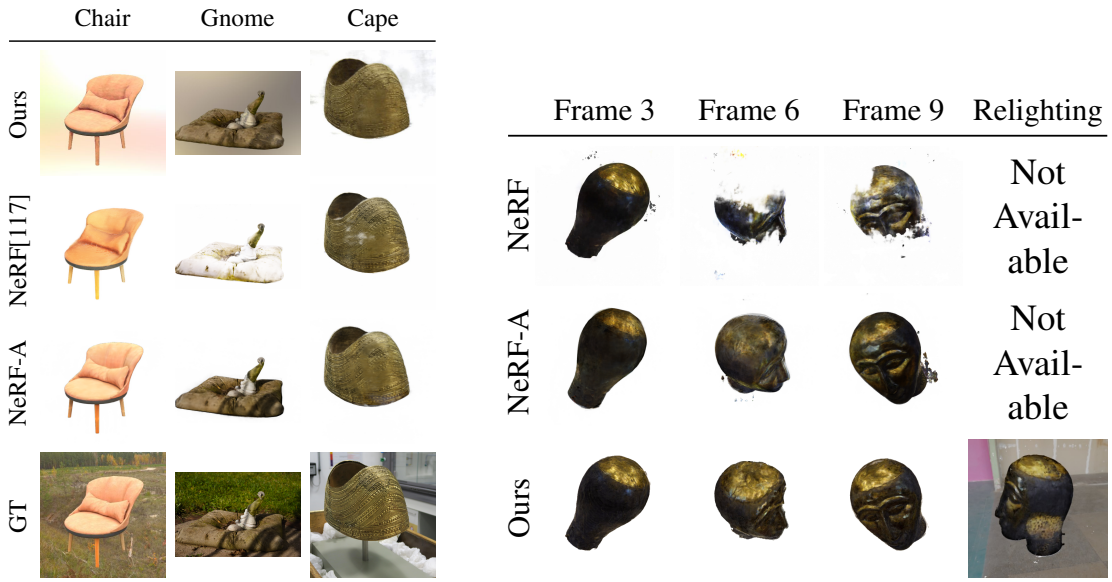



Figure 4.6: **Real world BRDF decomposition and relighting.** The decomposition produces plausible BRDFs, and re-rendered images are close to the ground truth input images. Note that the estimated parameters are hardly affected by the shadows visible in the input of the gnome scene. The appearance is well reproduced when relit with the estimated SG of a validation view. Even in a different perspective or under completely novel, artificial illumination, the recovered BRDF parameters result in compelling renderings.

seems plausible. The material is correctly classified as non-metallic (black metalness map), has a higher roughness, and the normal also aligns well with the shape. In the central valley, where dirt is collected, the BRDF parameters increase in roughness compared to the clean, smooth concrete pillow surface. The color is also captured well, and the similarity to the GT is evident in re-rendering.

Another evaluation focuses on using our method purely for novel view synthesis, with implicit relighting. In this setting, our method can be compared with NeRF [117] and




(a) **Influence of varying illumination.** (b) **Novel view comparison.**

Figure 4.7: **Comparison with NeRF and NeRF-A.** We show the quality of NeRF, NeRF-A, and NeRD on scenes under varying and fixed illumination in (a). Here, it is evident that NeRF fails as expected in scenes with varying illumination (Chair, Gnome). In (b), we highlight the improved consistency in our method. NeRF introduces highlights as floaters in the radiance volume that inconsistently occlude the scene geometry in other views. Additionally, we showcase the quality of relighting the Head with our method.

an extension to NeRF, called NeRF-A, which is inspired from [110]. NeRF-A models the appearance change per image in a 48-dimensional latent vector. It is worth noting that NeRF-A is a strong baseline as the task is simpler compared to NeRD, and it is only capable of relighting within known scenes. On fixed illumination scenes, NeRF-A is not capable of relighting. Tab. 4.2 shows the quantitative results over multiple datasets, real-world (Real) and synthetic (Syn.), on the test views wherein the "Fixed Illumination" case only novel view synthesis is performed, and in the "Varying Illumination" case, novel view synthesis and relighting. Here, "Varying Illumination" also refers to the case where the object is rotating w.r.t the camera, and therefore the relative illumination varies. The corresponding datasets for the fixed and synthetic cases are from NeRF (Ship, Lego, Chair), and for varying, we use ours (Globe, Wreck, Chair). For the real-world comparison, Cape provides fixed illumination, and the Gnome, Head, and MotherChild scenes are recorded in varying environments. Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) results show that NeRF performs relatively poorly in varying illumination cases. NeRF-A, on the other hand, is a strong baseline in the varying illumination case, which we mostly match or surpass while solving a more challenging problem which allows for more flexible relighting use cases.

(a) **COLMAP reconstruction.** COLMAP fails to recreate plausible geometry.

(b) **Ours.** Our reconstruction can handle this complex scene.

Figure 4.8: **Geometry reconstruction.** Comparison of a COLMAP reconstruction in our globe scene with varying illuminations.

Fig. 4.7b shows the novel view synthesis results of NeRF, NeRF-A, and NeRD (Ours) on the Ethiopian Head real-world scene. The object rotates in front of the camera. We, therefore, compose the Head on a white background in Fig. 4.7b as NeRF cannot handle a static background with a fixed camera. During training, both models recreate the input quite closely. However, in the test views, NeRF added spurious geometry to mimic highlights for specific camera locations, which are not seen by other cameras in the training set. NeRF-A can express the relative illumination change in the appearance embedding and improve the reconstruction quality compared to NeRF. However, as only a single illumination type is seen, NeRF-A is still not capable of relighting under arbitrary illumination. Due to our physically motivated setup with the explicit decomposition of shape, reflectance, and illumination, these issues are almost completely removed. Our method creates convincing object shapes and reflection properties, which also allow for relighting in novel settings.

Overall, it is evident that NeRF will not work with varying illuminations, clearly demonstrating the advantage of our more flexible decomposition. This is shown in Fig. 4.7a. Here, especially in the scenes with varying illumination (Chair and Gnome), NeRF fails as expected. After rendering the view, our method decomposes the information, and the synthesis is close to GT. In the scenes with fixed illumination (Head and Cape), the performance between NeRF and our method is on par in most parts. The main difference in MSE is due to the baked-in highlights of NeRF. Our physically grounded design using rendering reduces these artifacts drastically. We also want to point out that relighting a scene is not possible in NeRF. It is also worth noting that even if an appearance embedding as in NeRF-w [110] is used in our simplified NeRF-A baseline, the method can only interpolate between seen illuminations. Still, in a dataset, NeRF-A can capture the illumination conditions as shown in Fig. 4.7a and provides similar quality to ours. Our model can also relight even if the scene is only captured in a single fixed illumination.

#### 4.2.3.4 Results from partial estimation techniques

Several sub-tasks of the unconstrained shape and BRDF decomposition problem have been addressed by earlier works. Unfortunately, trying to recover parameters separately or sequentially, *e.g.* geometry, BRDF, or illumination, often fails in challenging scenes. We show that COLMAP fails to reconstruct a plausible geometry for some of our data sets in Fig. 4.8. If the following stages rely on accurate geometry, the pipeline cannot recover meaningful material properties from the inaccurate shape.

| Method [PSNR↑] | Diffuse | Specular | Roughness |
|---|---|---|---|
| Li *et al.* | 1.06 | — | 17.18 |
| Li *et al.* + NeRF | 1.15 | — | **17.28** |
| Ours | **18.24** | **25.70** | 15.00 |

Table 4.3: **BRDF estimation.** Comparison with a recent state-of-the-art method in BRDF decomposition under environment illumination [98]. Li *et al.*: directly on test images, Li *et al.* + NeRF: NeRF trained on BRDFs from [98].

We also tried recovering the BRDF parameters (diffuse and roughness) for each image separately using the work of Li *et al.* [98]. Li *et al.* [98] is a method that decomposes objects illuminated by environment illumination only into diffuse and specular roughness. However, no novel views cannot be synthesized, and single image decomposition is highly ill-posed. In Fig. 4.9 results are shown. The diffuse parameter in our method is more consistent compared to Li *et al.*, and it is apparent that Li *et al.* failed to factor out the illumination from the diffuse. However, the roughness is slightly better for Li *et al.* but is not as consistent, and the roughness is highly correlated with the texture of the globe, which is incorrect. Our method is biased towards the extreme roughness value range but is more consistent. It is also worth noting that the roughness parameter plays a minor role compared to the diffuse color during re-rendering. If the color of an object is wrong, the error is more visible than slight alterations in how reflective it is. Additionally, our method can estimate the specular color, which is a challenging task and allows our method to render metals correctly.

Li *et al.* [98] is a method that decomposes objects illuminated by environment illumination only into diffuse and specular roughness. However, no novel views cannot be synthesized, and single image decomposition is highly ill-posed.

As Li *et al.* do not allow for a significant degree of novel view synthesis – except slightly based on the estimated depth map – one approach to solve this is to use NeRF on top. By running Li *et al.* on the train set and then constraining NeRF not to use view-dependent effects and extending the RGB space to RGB + roughness, we can try to join the distinct images in a volumetric model. In Fig. 4.9 it is visible that this method fails, as each image is quite different from the other, and NeRF cannot place the varying information at the correct locations.

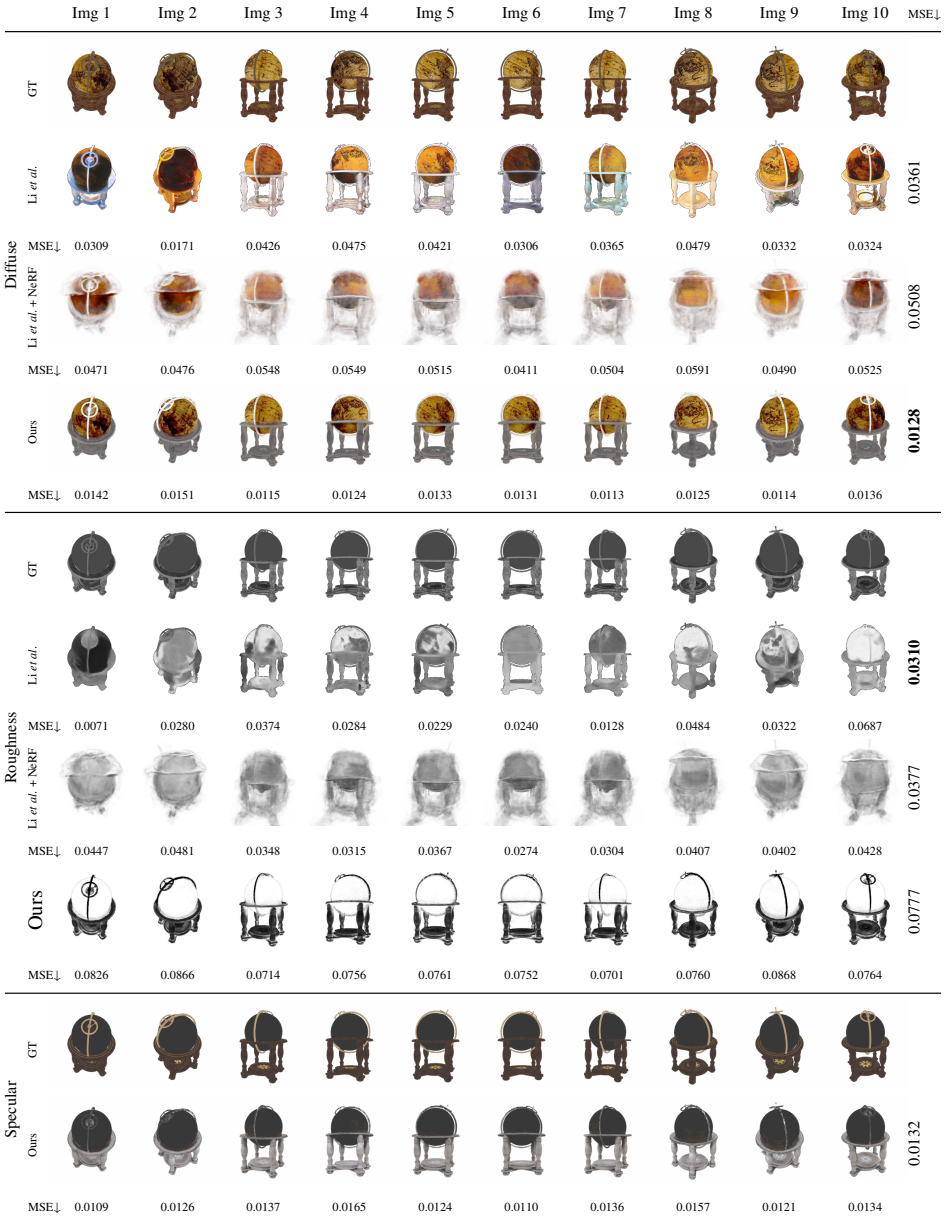



Figure 4.9: **Partial estimation techniques.** Comparison with partial estimation methods. Here, we compare our method with Li *et al.* [98] and Li *et al.* + NeRF. Li *et al.* is a method that decomposes a single image of an object illuminated by environment light into diffuse and roughness parameters. For the Li *et al.* + NeRF baseline, we first decomposed the training dataset with Li *et al.* and then trained a NeRF with disabled view conditioning on top. We then generate the novel test set views. For Li *et al.*, no view synthesis takes place, and the method is run on the test set directly. Notice how our method generates consistent results on all test views.

We, therefore, conclude that joint optimization of shape and SVBRDF is essential for this highly ambiguous problem. Quantitative comparisons with Li *et al*. are shown in Tab. 4.3. These are average PSNR results over our synthetic datasets (Globe, Wreck, and Chair). We decompose our basecolor into *diffuse* and specular to enable comparison with Li *et al*., which uses a *diffuse* and roughness parameterization. It is worth noting that Li *et al*. is a weak baseline but the closest available, as their method expects a flash in conjunction with the environment illumination. However, as most scenes are captured with an outside environment illumination, the flash will be barely noticeable due to the strong sunlight.

| Method | Base Color | Metalness | Roughness | Normal | Re-Render |
|---|---|---|---|---|---|
| w/o Grad. Normal | 0.1264 | 0.1203 | 0.3192 | 0.1664 | 0.0893 |
| w/o Com. BRDF | 0.1828 | 0.2496 | 0.2827 | 0.0089 | 0.0759 |
| w/o WB | 0.1059 | 0.0870 | 0.2754 | 0.0087 | 0.0655 |
| Full Model | **0.0796** | **0.0784** | **0.2724** | **0.0084** | **0.0592** |

Table 4.4: **Ablation study.** The MSE loss on 10 test views with ablation of gradient (grad.) normals, compressed (Com.) BRDF and white balancing (WB) on the globe dataset.

#### 4.2.3.5 Ablation study

In Tab. 4.4, we ablate the gradient-based normal estimation, the BRDF interpolation in a compressed space, and incorporating the white balancing in the optimization. We perform this study on the Globe scene as it contains reflective, metallic, and diffuse materials and fine geometry. One of the largest improvements stems from the addition of gradient-based normals. The coupling of shape and normals improves the BRDF and illumination separation. Normals cannot be rotated freely to mimic specific reflections. The compressed BRDF space also improves the result, especially in the metalness parameter estimation. This indicates that the joint optimization of the encoder/decoder network $N_{\phi_2}$ optimizes similar materials across different surface samples. The white balancing fixes the absolute intensity and color of the SGs, which indirectly forces the BRDF parameters into the correct range.

## 4.3 Neural-PIL: Neural Pre-integrated Lighting for Reflectance Decomposition

*This section is based on the publications*:

Neural-PIL: Neural Pre-Integrated Lighting for Reflectance Decomposition
Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, Hendrik P. A. Lensch
*Advances in Neural Information Processing Systems (NeurIPS)* - 2021

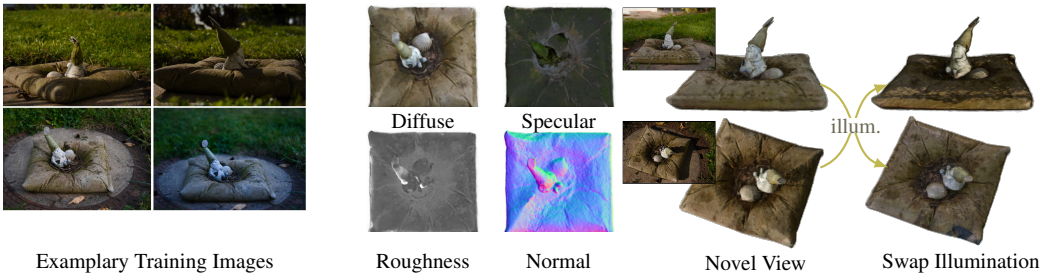



Figure 4.10: **Problem setting.** Our Neural-PIL-based technique decomposes images observed under unknown illumination into high-quality BRDF, shape, and illuminations. This allows us to synthesize novel views (targets shown in insets) and perform relighting or illumination transfer.

We introduced reflectance fields in the previous Sec. 4.2. A collection of posed images is used to perform a complete *inverse rendering* by creating a reflectance field. It decomposes the images into an object's 3D shape and SVBRDF under different illumination conditions. The illumination is also jointly recovered for each image or globally if only a single illumination is present. This approach offers easy relighting and is often tackled in recent literature [25, 150, 193, 194]. A key component in learning these neural SVBRDF decomposition networks is the differentiable rendering [25, 29, 194] that generates images and gradients for the estimated lighting and SVBRDF parameters. These methods leverage traditional rendering techniques within modern deep learning frameworks to enable backpropagation. This is often expensive, as rendering requires computing integrals over the incoming light at each 3D location in the scene. In our previous work NeRD of Sec. 4.2, we leverage SG illumination as described in Sec. 2.2.1 to accelerate the illumination integration. Concurrently Zhang *et al*. propose to leverage SGs to approximate illumination. However, they only tackle objects under a fixed illumination. However, these SG representations lack the capacity required to model or recover the shape and material properties of highly reflective objects or images in complex natural environments. One downside of SG illumination is that the illumination created by those SGs is mostly of low frequency, and even if the number of SGs is increased, the individual SG would still produce visible spherical artifacts due to their spherical definition.

This work aims to replace the costly illumination integration step within these rendering approaches with a learned network. Inspired by the real-time graphics literature on image-based lighting [77], we propose a novel pre-integrated lighting (PIL) network that converts the illumination integration process used in rendering into a simple network query. Our Neural-PIL takes as input a latent vector representation for the environment map, the surface roughness, and an incident ray direction, and from them predicts an integrated illumination estimate. This query-based approach for light integration results in efficient rendering and simplifies and accelerates rendering and optimization. This neural light representation is also significantly more expressive than the more commonly used SG representation, thereby enabling higher-fidelity renderings. The architecture of our Neural-PIL uses a conditional MLP with FiLM layers [36]. Fig. 4.11 illustrates this illumination pre-integration for different surface roughness levels.

Additionally, our previous work NeRD of Sec. 4.2 is unconstrained in the BRDF and illumination reconstruction. However, a manifold of physically plausible illuminations and BRDF appearing in the real world exists. We present a smooth manifold auto-encoder (SMAE), based on interpolating auto-encoders [20], that can learn these effective low-dimensional manifolds. This learned low-dimensional space serves as a strong regularizer or prior for constraining the solution space of BRDFs and illumination. These constraints are critical due to the ill-posedness of our problem setting. The smoothness of this manifold enables stable and effective gradient-based optimization of BRDF and light parameters. The Neural-PIL, light-SMAE, and BRDF-SMAE networks are pre-trained on a dataset with high-quality environment maps (illumination) and materials (BRDFs). We integrate these component networks into our decomposition framework, optimizing a 3D neural volume with shape and SVBRDF while also optimizing per-image illuminations.

We perform an empirical analysis on synthetic datasets and qualitative and quantitative visual results on real-world datasets. We demonstrate that using our Neural-PIL decomposition network, the shape and material properties estimation is more accurate than the prior art. The 3D assets with material properties produced by our model can be used to generate high-quality relighting and view-synthesis results with finer details compared to existing approaches.

### 4.3.1 Problem Setup

Given an image collection of an object captured under varying illumination conditions and camera viewpoints, we aim to jointly estimate the object's 3D shape and spatially-varying BRDF, as well as the illumination conditions of each image. Our input consists of a set of $q$ images with $s$ pixels each: $C_j \in \mathbb{R}^{s\times 3}; j \in \{1,\dots,q\}$ along with per-pixel masks $M_j \in \{0,1\}^{s\times 1}$ indicating which pixels belong to the object. Our goal is to learn a neural 3D volume $\mathcal{V}$ where, at each point $\boldsymbol{x} \in \mathbb{R}^3$, we estimate the BRDF parameters for the Cook-Torrance model [41] $\boldsymbol{b} \in \mathbb{R}^7$ (diffuse $\boldsymbol{b}_d \in \mathbb{R}^3$, specular $\boldsymbol{b}_s \in \mathbb{R}^3$, roughness $b_r \in \mathbb{R}$), unit-length surface normal $\boldsymbol{n} \in \mathbb{R}^3$ and optical density $\sigma \in \mathbb{R}$. In addition, we

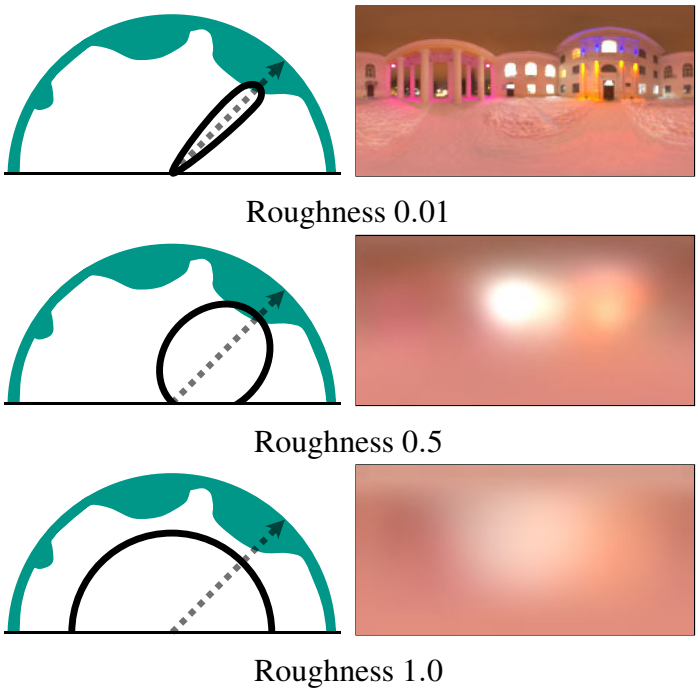



Figure 4.11: **Pre-integrated lighting.** As the roughness of the material increases, the reflected radiance depends on a larger region of the environment map. Brute-force integrals over the environment map are expensive. Hence we propose a coordinate-based MLP that is trained to directly output the integrated illumination values conditioned on the surface roughness and view direction.

also estimate latent vectors representing per-image illumination $\boldsymbol{z}^l \in \mathbb{R}^{128}$. This problem statement corresponds to the practical application of recovering a 3D model of some real-world object (e.g., a statue or landmark) that different people have photographed at different times.

### 4.3.2 Image formation and image-based lighting

NeRF directly models view-dependent color $\hat{\boldsymbol{c}}$ at each 3D location. Thus, a simple image formation process that integrates the color information along camera rays is sufficient to render images. In contrast, we want to explicitly estimate an object's material decomposition at each 3D location. We, therefore, must use a more explicit rendering formulation that relates image formation to BRDFs and illumination. The rendering equation [75] estimates the radiance $L_o \in \mathbb{R}^3$ at $\boldsymbol{x}$ along the outgoing view direction $\boldsymbol{\omega}_o \in \mathbb{R}^3$ ($\boldsymbol{\omega}_o = -\boldsymbol{d}$), where $\boldsymbol{d}$ is the ray direction: $L_o(\boldsymbol{x}, \boldsymbol{\omega}_o) = \int_\Omega f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o; \boldsymbol{b}) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i)(\boldsymbol{\omega}_i \cdot \boldsymbol{n})\, d\boldsymbol{\omega}_i$, where $f_r$ is the BRDF evaluation, $L_i \in \mathbb{R}^3$ is incoming light, and $\boldsymbol{\omega}_i \in \mathbb{R}^3$ is the incoming light direction. Using this single bounce rendering formulation and ignoring exposure varia-

tion and tone-mapping, $L_o$ is equivalent to NeRF's color $\hat{\boldsymbol{c}}$.

We follow the concept of pre-integrated illumination as discussed in Sec. 2.2.2. The formulation for rendering is then defined in Eq. (2.36). However, we further simplify it as follows:

$$L_o(\boldsymbol{x}, \boldsymbol{\omega}_o) \approx \underbrace{(\boldsymbol{b}_d/\pi)\tilde{L}_i(\boldsymbol{n}, 1)}_{\text{diffuse}} + \underbrace{\boldsymbol{b}_s(F_0(\boldsymbol{\omega}_o, \boldsymbol{n})B_0(\boldsymbol{\omega}_o \cdot \boldsymbol{n}, b_r) + B_1(\boldsymbol{\omega}_o \cdot \boldsymbol{n}, b_r))\tilde{L}_i(\boldsymbol{\omega}_r, b_r)}_{\text{specular}} \tag{4.4}$$

The illumination is now pre-integrated as: $\tilde{L}_i(\boldsymbol{\omega}_r, b_r) = \int_\Omega D(b_r, \boldsymbol{\omega}_i, \boldsymbol{\omega}_r) L_i(\boldsymbol{x}, \boldsymbol{\omega}_i) d\boldsymbol{\omega}_i$ which only depends on the mirrored view direction $\boldsymbol{\omega}_r$ (which subsumes the surface normal $\boldsymbol{n}$) and the roughness $b_r$, where $D$ describes the microfacet distribution [170]. This light pre-integration is illustrated in Fig. 4.11. Note that compared to Eq. (2.36), the same pre-integrated $\tilde{L}_i(\boldsymbol{\omega}_r, b_r)$ is queried twice: the diffuse part captures the entire hemisphere and therefore is parameterized by the surface normal $\boldsymbol{n} \in \mathbb{R}^3$ and a diffuse roughness of 1. The specular part looks up $\tilde{L}_i(\boldsymbol{\omega}_r, b_r)$ for the reflected view direction $\boldsymbol{\omega}_r \in \mathbb{R}^3$ and the specular roughness $b_r$.

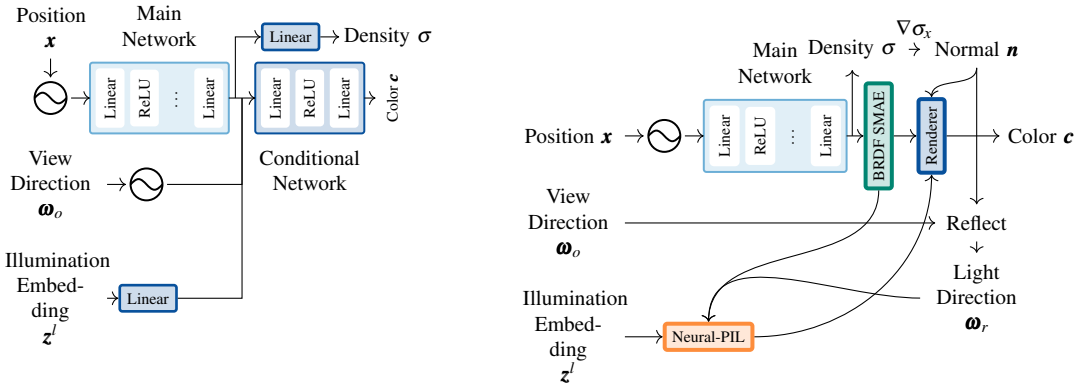
This pre-integration approach replaces the complex integration during shading with a set of simple additions and multiplications. We have integrated the core idea of this approach into an efficient differentiable neural rendering framework, which allows for the optimization of geometry, BRDF, and illumination simultaneously via standard back-propagation. We further reduce the computational complexity by mimicking the pre-integration of $\tilde{L}_i(\boldsymbol{\omega}_r, b_r)$ with a simple query through our Neural-PIL that operates directly on a neural representation of the illumination, as we will now explain.

## 4.3.3 Network Architecture

Fig. 4.12 shows the neural decomposition architecture which closely follows the architectures of NeRF [117] and NeRD as explained in Sec. 4.2.2, but with some key differences. The coarse network learns a view and illumination-dependent color, whereas the fine network decomposes the scene into BRDF parameters.

### 4.3.3.1 Coarse network

Like in NeRF [117], the coarse network aims to obtain a rough point density that helps in finer sampling for the following decomposition network. As illustrated in Figure 4.12a, the coarse network takes 3D location $\boldsymbol{x}$, view direction $\boldsymbol{\omega}_o$ and illumination embedding $\boldsymbol{z}^l$ as input and predicts point density $\sigma$ and color $\boldsymbol{c}$ at $\boldsymbol{x}$. In contrast to NeRF, which estimates view-conditioned colors, we estimate both view and illumination-conditioned colors, as our input images can be captured under varying illumination.




(a) **Coarse network.** (b) **Decomposition network.**

Figure 4.12: **Decomposition with the Neural-PIL architecture.** A architecture similar to NeRD (Sec. 4.2) and NeRF-w [110] is used for our coarse network in (a), where a latent illumination code is estimated to predict a view-dependent color and density. In (b), pre-trained networks restrict the possible BRDF representation (BRDF-SMAE) and the incident lighting (PIL) to lower-dimensional spaces. A single evaluation of Neural-PIL returns a pre-filtered illumination cone according to surface roughness. Using that, the BRDF estimate, and a surface normal (the unit-norm gradient of our density estimate), we render the shaded color $\boldsymbol{c}$.

#### 4.3.3.2 Decomposition network

The decomposition network estimates density $\sigma$ and BRDF embedding $\boldsymbol{z}^b \in \mathbb{R}^4$ at each 3D location $\boldsymbol{x}$ in the implicit volume. As illustrated in Figure 4.12b, the conditional network in the coarse network is replaced by explicit rendering in the decomposition network. There are two key innovations in the decomposition network: 1) Use a novel pre-integrated light (PIL) network that results in efficient rendering while also representing the illumination with high fidelity. 2) We learn smooth low-dimensional manifolds to represent illumination and BRDF parameters, which serve as strong priors. We will explain our rendering process, the Neural-PIL, and the smooth manifold auto-encoders (SMAE).

#### 4.3.3.3 Rendering process

For rendering, we use the rendering formulation in Eq. (4.4). We estimate normal $\boldsymbol{n}$ at $\boldsymbol{x}$ by computing the gradient $\nabla\sigma_x$ of the density w.r.t. the input position (by passing gradients through the decomposition network). We also convert the BRDF embedding $\boldsymbol{z}^b$ into BRDF parameters $\boldsymbol{b}$ with our BRDF-SMAE. Unlike NeRF [117] of Sec. 2.4.3, which integrates sample colors along the camera ray, we first compute the expected termination of each ray (similar to a depth map) with the sample densities along the camera ray and then do the rendering only at that point along each camera ray. At the ray termination po-

sitions, we compute the integrated illumination $\tilde{L}_i(\boldsymbol{\omega}_r, b_r)$ and use Eq. (4.4) for rendering with the estimated BRDF $\boldsymbol{b}$ and normal $\boldsymbol{n}$.

#### 4.3.3.4 Neural-PIL

The integration of the incoming light is traditionally approximated by Monte Carlo sampling, in which illumination contributions from many directions are numerically integrated. The computation of the pre-integrated $\tilde{L}_i(\boldsymbol{\omega}_r, b_r)$ also involves either this costly numerical accumulation or a convolution performed on the complete environment map — though neither approach is practical within a differential rendering engine. We, therefore, learn a network that performs this light pre-integration, thereby converting the costly integral computation into a simple network query. The architecture of the Neural-PIL is visualized in Fig. 4.13. The Neural-PIL takes as input the illumination embedding $\boldsymbol{z}^l$, the incoming light direction $\boldsymbol{\omega}_r$ and the roughness $b_r$ at a point, and directly predicts the pre-integrated light $\tilde{L}_i(\boldsymbol{\omega}_r, b_r)$. The aim of the Neural-PIL is to first decode the illumination along the incoming mirror direction $\boldsymbol{\omega}_r$ from the given embedding $\boldsymbol{z}^l$ and then mimic the light pre-integration process for the surface roughness $b_r$. Following this general intuition, the Neural-PIL takes $\boldsymbol{\omega}_r$ as input, and we condition the first few layers of the network with illumination $\boldsymbol{z}^l$, and condition a later layer with roughness $b_r$. The first few layers are intended to decode all required illumination information for the given direction. The later layers are intended to perform light integration conditioned on the material roughness.

We leverage a pi-GAN-like [36] architecture for the Neural-PIL design with FiLM-SIREN layers. Each FILM-SIREN layer [36] takes the modulating parameters (a scalar $\lambda_0$ and two vectors $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$) to modulate the output $\boldsymbol{y}$ of the earlier linear layer followed by sine computation as follows:

$$\phi(\boldsymbol{y}) = \sin(\lambda_0 \boldsymbol{\gamma} \odot \boldsymbol{y} + \boldsymbol{\beta}) \tag{4.5}$$

where $\odot$ denotes a Hadamard product. In our Neural-PIL, we employ two mapping networks to predict modulating parameters($\boldsymbol{\gamma}$, $\boldsymbol{\beta}$) for the FiLM-SIREN layers. The first mapping network generates the modulating parameters for the first layers from the illumination embedding $\boldsymbol{z}^l$. The second performs the same for the penultimate layer with the roughness $\boldsymbol{b}_r$.

In addition to converting a costly light integration process in the rendering into a simple network query, our Neural-PIL has another key advantage. State-of-the-art differentiable rendering frameworks [25, 57, 94, 194] that work with BRDFs use either SG – as seen in our previous approach NeRD of Sec. 4.2 – or spherical harmonic (SH) light representations, which both suffer from lack of fine details. Although one could represent fine illumination details with a large number of SG or SH bands, this parameter increase would also slow the rendering with high memory costs. In contrast, our Neural-PIL is an MLP that directly produces the pre-integrated light required for the rendering. Our ex-

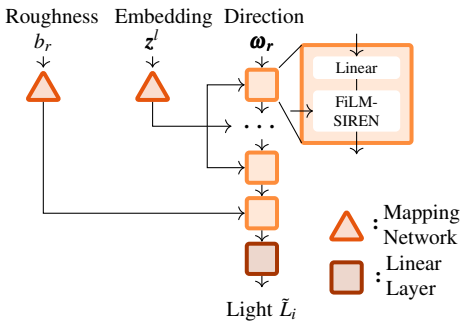



Figure 4.13: **Neural-PIL.** A coordinate-based MLP returns the pre-integrated radiance for the query direction, where roughness determines the integration footprint.

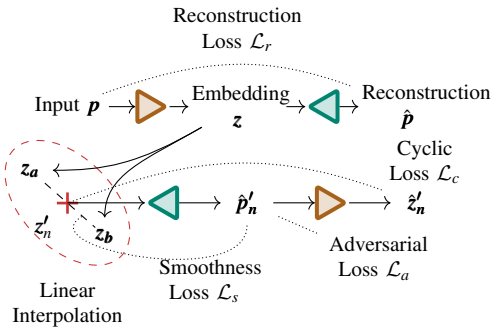



Figure 4.14: **Smooth manifold auto-encoder.** By imposing specific losses on interpolations between input samples, our Smooth Manifold Autoencoder encourages a smooth embedding space.

periments also demonstrate that our Neural-PIL can represent finer details in illumination compared to SG representation (Sec. 4.3.4 - Fig. 4.15).

#### 4.3.3.5 Smooth manifold auto-encoder (SMAE)

Joint estimating 3D shapes, materials, and lighting is a highly underconstrained problem. Therefore, we must regularize optimization towards likely illuminations and BRDFs to converge to plausible solutions. For this, we learn low-dimensional smooth manifolds that capture the data distribution of BRDFs and illuminations. In addition to acting as strong priors, optimizing on smooth manifolds (as opposed to directly optimizing on standard BRDF space, which need not be smooth) allows for more effective gradient-based optimization of reflectance decomposition for a given scene. Fig. 4.14 illustrates our SMAE that we use to learn separate low-dimensional manifolds to represent BRDF and illumination embeddings. Specifically, we use Interpolating Autoencoders [20] with several additional loss functions. The encoder network $E$ takes input $\boldsymbol{p}$ (either BRDF or light environment map) and generates the latent embedding $\boldsymbol{z}$, which is then passed onto the decoder network $G$ that generates an input reconstruction $\boldsymbol{p}'$. We then randomly sample two latent vectors from the mini-batch: $\boldsymbol{z}_a$ and $\boldsymbol{z}_b$; followed by sampling $m \in \mathbb{N}$ linearly interpolated embeddings that are uniformly spaced between $\boldsymbol{z}_a$ and $\boldsymbol{z}_b$: $\{\boldsymbol{z}'_n\}|n = 1,2,\ldots,m$. We pass each of these interpolated latents $\boldsymbol{z}'_n$ through the decoder $G$ and the encoder $E$ to obtain $\hat{\boldsymbol{p}}'_{\boldsymbol{n}}$ and $\hat{\boldsymbol{z}}'_{\boldsymbol{n}}$, respectively. Using the four losses depicted in Fig. 4.14 the encoder and decoder networks are trained jointly. One is the standard reconstruction loss $\mathcal{L}_r$ between input $\boldsymbol{p}$ and reconstruction $\boldsymbol{p}'$. In addition, we add a discriminator network on $\hat{\boldsymbol{p}}'_{\boldsymbol{n}}$ and use the standard adversarial loss $\mathcal{L}_a$ used in LS-GAN [109], which ensures that the interpolated latent vectors can generate plausible data. We enforce a bijective mapping of the encoder and the decoder with a cyclic loss

$\mathcal{L}_c$ which is the $L_2$-loss between the interpolated latents $\{\boldsymbol{z}'_n\}$ and their re-estimated counterparts $\{\hat{\boldsymbol{z}}'_n\}$. Lastly, to ensure that the learned embedding space is smooth, we impose a smoothness loss $\mathcal{L}_s$ on the gradient of decoder $G$ w.r.t. the interpolating scalar value $\alpha$: $\mathcal{L}_s = 1/m \sum_n (\nabla_\alpha G(\boldsymbol{z}'_n))^2$. The total loss to train SMAE is a combination of the 4 losses:

$$\mathcal{L} = \mathcal{L}_r + \lambda_1 \mathcal{L}_a + \lambda_2 \mathcal{L}_c + \lambda_3 \mathcal{L}_s \tag{4.6}$$

Despite being only 7-dimensional, the space of the Cook-Torrence BRDF representation [41] that we use is too unconstrained for our task and imposes strong correlations between the diffuse and specular terms of real-world materials. We therefore train a BRDF-SMAE with an MLP encoder and decoder that maps these 7D parameters into 4D latent embeddings $\boldsymbol{z}^b \in \mathbb{R}^4$ using our dataset of real-world BRDF material collections from Sec. 3.2.4. Similarly, real-world illuminations exhibit significant statistical regularities: lights are more likely to be tinted blue or yellow, and a brighter light is more likely to come from above than below. To capture this regularity, we train a Light-SMAE with CNN encoder and decoder on a dataset of 320 environment maps from [190]. We then map the $128 \times 256$ 2D environment maps onto a 128-dimensional smooth latent space, $\boldsymbol{z}^l \in \mathbb{R}^{128}$.

#### 4.3.3.6 Architecture and training details

For this method, several networks are trained separately and combined. The following explains the training schedule for each sub-network, and the architectural details are provided.

**BRDF-SMAE.** In our decomposition network, a specific BRDF embedding is stored in a neural volume at each point. Our BRDF-SMAE should therefore be able to encode a single BRDF. Each point in the neural volume can have a different embedding, so the resulting decomposition has a spatially varying BRDF. To encode this singular BRDF per point, we leverage a MLP network architecture. We use three MLP layers with 32 output features for the encoder, decoder, and discriminator. We train our SMAE to create a smooth latent space on the material dataset from Sec. 3.2.4. We set the SMAE loss weighting to $\lambda_1 = 0.01$, $\lambda_2 = 0.01$ and $\lambda_3 = 0.001$. We use 64 interpolation steps in the latent space and a MAE on the BRDF parameters for the reconstruction loss. We perform 1.5 million training steps with a batch size of 256 for training on a single NVIDIA 1080 TI GPU. This roughly takes 3.5 hours to converge. We use the Adam optimizer with a learning rate of 1e$-$4.

**Light-SMAE.** As our dataset only consists of 320 environment maps, we augment the dataset by randomly rotating each environment map ten times, and during training, we randomly blend two environment maps. Additionally, we downscale the environment maps to $128 \times 256$. We set the SMAE loss weighting to $\lambda_1 = 0.01$, $\lambda_2 = 0.0001$ and

$\lambda_3 = 0.05$ with 5 interpolation steps between each of the batch halves. Due to the high dynamic range, we found that specific care is required to ensure smooth training. The input to the encoder is transformed from HDR to LDR by $\log(1+x)$ and the output from LDR to HDR with $\exp(x-1)$. We further calculate the loss on a logarithmic scale using the MALE loss: $|\log(1+x^*) - \log(1+\hat{x})|$.

Our networks are all based on CNNs, whereas the encoder and discriminator leverage CoordConvs [103]. The encoder and discriminator do not use padding, whereas the decoder uses the "same" padding. The overall architecture is shown in Tab. C.1 in the appendix Sec. C. We run 4 million steps with a batch size of 24 to train on a single NVIDIA 1080 TI GPU, which takes five days to train. The Adam optimizer with a learning rate of 5e−4 is used for the training.

**Neural-PIL.** We train the Neural-PIL using the same environment maps dataset for training the Light-SMAE. Here, the encoder of the Light-SMAE is used for defining the smooth latent space. The network is comprised of MLPs with the FiLM-SIREN conditioning [36]. The first portion of the network is comprised of three layers with 128 features. The $\beta$ and $\gamma$ conditioning parameters are generators from a mapping network with an MLP of two layers with 128 ELU activated features. An additional dense output layer produces 768 features, corresponding to 128 $\beta$ and 128 $\gamma$ features per layer. The penultimate layer is then conditioned on the roughness $b_r$, which is parametrized by a mapping network with the first ELU activated layer outputting 32 features and the second 256 for the $\beta$ and $\gamma$ mapping parameters of the respective layer. Finally, a final MLP in the main network generates the final output color with three features output and $\exp(x-1)$ as the activation to generate HDR values.

For training, we first try to encode the full environment map with a roughness of 0. In addition, we perform reconstruction with 8192 random directions and roughness levels. Both perform both of these simultaneously with a batch size of 8. We perform 4 million steps to train on a single NVIDIA 1080 TI GPU, which takes 4.5 days. We use the Adam optimizer with a learning rate of 5e−4.

**Decomposition.** The coarse and fine networks consist of 8 MLP layers with 256 ReLU activated features. We randomly sample 4096 ray directions per image for training. The ray directions are also jittered as described in our previous work NeRD Sec. 4.2.2.2. We sample the ray in disparity rather than linear in depth for real-world data. This places more samples close to the camera. We train 400,000 steps on 4 NVIDIA 2080 TI GPU, which takes about 22 hours. We use an Adam optimizer with a learning rate of 4e−4. The pre-trained BRDF-SMAE's decoder and the Neural-PIL remain with frozen weights during the training. For stability, we only optimize the illumination embedding $z^l$ via the decomposition network, and we do not backpropagate the loss signal onto illumination in the coarse network.

We employ additional exponentially decaying losses over 10,000 steps. We use the

background segmentation loss, similar to NeRD (Sec. 4.2.2.7), which ensures rays that do not hit the object do not contribute and additionally add a BRDF priming loss. This loss initially sets the diffuse color to the actual image color and the roughness to 0.3 using a MSE. The background segmentation loss fades in over the duration, and the BRDF priming loss fades out. Our main reconstruction loss is an MSE between the rendered color $\boldsymbol{c}$ and the corresponding pixel in the input image. This loss is then exponentially faded over 100,000 steps to a cosine weighted MSE: $(\boldsymbol{x}^* \boldsymbol{\omega}_o \cdot \boldsymbol{n} - \hat{\boldsymbol{x}} \boldsymbol{\omega}_o \cdot \boldsymbol{n})^2$. This weighting tends to achieve better BRDF fitting results [54] as harsh grazing highlights from the Fresnel effect is not factored as much as regular samples, as well as our approximated rendering model is the least accurate in the grazing angles. The reason for this fading loss scheme is that the normals $\boldsymbol{n}$ are unreliable in the early stages of the training.

## 4.3.4 Results

We evaluate our approach w.r.t. different baselines on the aspects of BRDF and light estimation, view synthesis, and relighting.

### 4.3.4.1 Baselines

The closest work to ours is our previous method NeRD (Sec. 4.2), which forms our primary comparison across different evaluations. To our knowledge, no other published work tackles the same problem of estimating shape, illumination, and BRDF from images of varying illumination. For view synthesis, we also compare with NeRF [117]. For BRDF evaluations, we also compare with Li *et al*. [98] which does BRDF decomposition from a single image. In addition, we combine Li *et al*. [98] with NeRF [117] to create a baseline that is closer to our problem setting.

For the evaluation, we leverage the same datasets described in Sec. 4.2.3.1.

| Roughness | MC | SGs | Neural-PIL (Ours) |
|---|---|---|---|
| 0.2 | 34.88 | 31.57 | **35.76** |
| 0.5 | 35.14 | 28.98 | **35.28** |

Table 4.5: **Better illumination estimates with Neural-PIL.** Average PSNR with 6 rendered spheres shows that Neural-PIL achieves better PSNR over the SG and Monte-Carlo integration (MC) baselines. More accurate illuminations enable improved BRDF decomposition and relighting.

### 4.3.4.2 Fildelity of Neural-PIL

Since Neural-PIL forms the key component of our decomposition framework, we evaluate its learned light representation against a more commonly used SG representation.

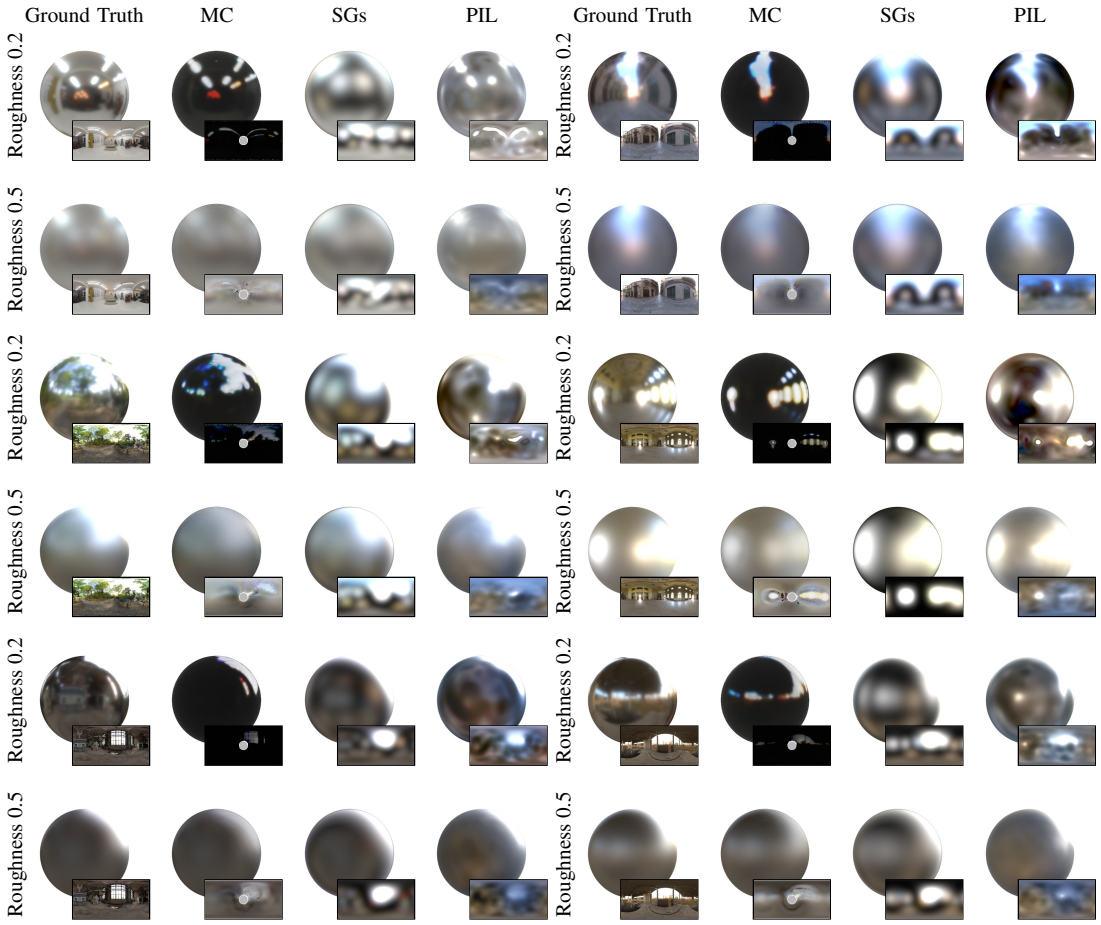



Figure 4.15: **Neural-PIL *vs*. Spherical Gaussian (SG) *vs*. Monte-Carlo (MC) integration renderings.** With known geometry and reflectance, we optimize using MC integration for the direct illumination, SGs as well as our latent illumination via Neural-PIL. This figure shows final renderings with the optimized light parameters, while the recovered illumination is shown in the insets. Despite Neural-PIL having fewer parameters, it can recover more detailed environment maps and produce accurate renderings.

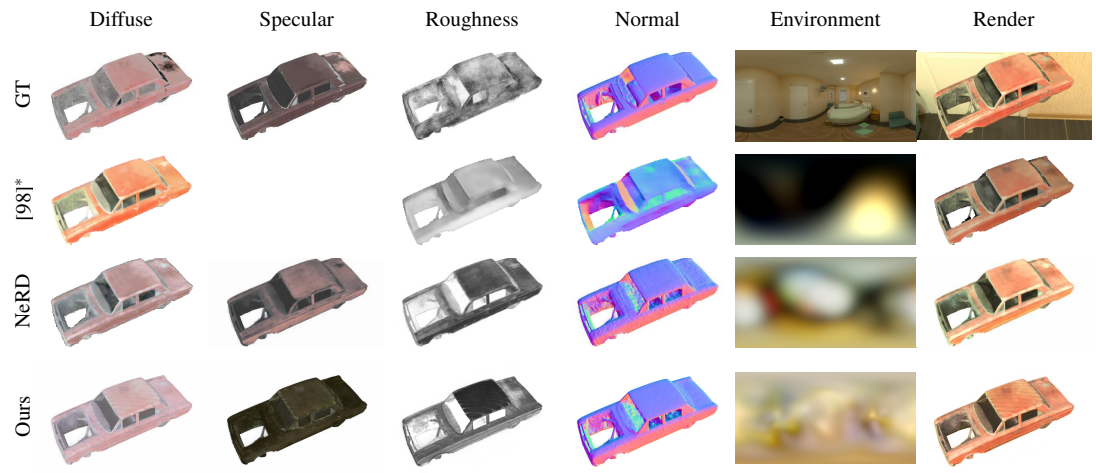
Additionally, we add a baseline that directly optimizes an environment map using Monte-Carlo (MC) integration. We render a simple metallic sphere using an unseen environment, as shown in Fig. 4.15, with two different roughness levels, 0.2 and 0.5. Assuming known roughness and shape, we optimize for SG illumination using the SG-based differentiable rendering used in NeRD. Similarly, we optimize the latent illumination representation using our Neural-PIL-based renderer. For the MC baseline, we leverage BRDF importance sampling, which describes how the rays would likely scatter based on the surface roughness. Here, we cast 128 samples-per-pixel (spp) based on the BRDF towards the environment map with a resolution of $128 \times 256$. Overall, 1000 optimization steps are performed for each method. The resulting estimated MC, SG illumination and Neural-PIL illuminations are shows in Fig. 4.15. Compared to the SG illumination model with 24 lobes and 168 parameters, our recovered illumination vector $\boldsymbol{z}^l$ with only 128 dimensions captures more details, especially in the high-frequency areas. This leads to a significantly reduced rendering error for both roughness values, even though the illumination prediction is more ambiguous for rougher materials. While the MC integration could easily recover detailed highlights, the remaining areas are not recovered well. Besides the improved quality, Neural-PIL-based rendering is also much faster. Rendering million samples with our Neural-PIL network takes just 1.86 ms compared to 210 ms rendering with 24 SGs. Tab. 4.5 shows average PSNR on 6 rendered spheres with more visual results similar to Fig. 4.15 in the supplements. Our method outperforms both baselines in reconstruction quality.

| Parameter | w/o BRDF SMAE | Ours |
|---|---|---|
| Diffuse | 11.87 | **20.22** |
| Specular | 9.24 | **16.84** |
| Roughness | 16.51 | **24.82** |

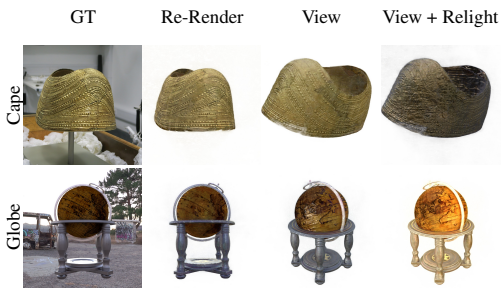
Table 4.6: **Ablation study.** Average PSNR of BRDF estimation on 3 synthetic scens under varying illumination demonstrates the positive influence of using the BRDF-SMAE to constrain the BRDF parameter space.

#### 4.3.4.3 Ablation study

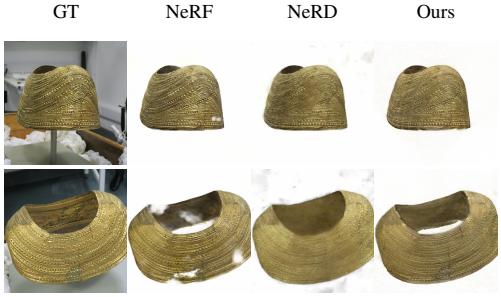
To showcase the effectiveness of our novel additions, we perform an ablation of the BRDF-SMAE. Tab. 4.6 shows the influence of the BRDF-SMAE on material estimation. These are the PSNR values on the three synthetic scenes under varying illumination. It is clear from the table that, especially in estimating the specular parameter, using BRDF-SMAE improves the results drastically. As this parameter is also tied to the diffuse color, degradation in performance is expected. Even though it is uncorrelated to diffuse and specular, the roughness parameter is most likely improved due to improved color param-




(a) **BRDF and light estimation.**




(b) **View synthesis and relighting.**




(c) **View synthesis with NeRF, NeRD and our approach.**

Figure 4.16: **Visual comparisons.** In (a), our model produces more accurate BRDF and illumination estimates, which results in more accurate rendering results. To evaluate view synthesis and relighting, we show in (b) the influence of keeping the camera and light fixed (col 2), then moving the camera (col 3), and then adjusting the lighting (col 4). In (c), we show that even when using a single illumination (the problem setting used by NeRF), our method produces shape estimates with fewer artifacts and more detail than NeRF or NeRD.

eters. For the ablation of the Neural-PIL network, one can refer to NeRD as a baseline that neither uses BRDF-SMAE nor Neural PIL. PSNR metrics in Tab. 4.7a shows that our method can result in better decomposition compared to our previous method NeRD.

#### 4.3.4.4 BRDF evaluations

Tab. 4.7a shows the BRDF estimation metrics for different techniques computed on the scenes Globe, Car and Chair. Compared with NeRD, our approach resulted in better diffuse and roughness parameters. Only the prediction of the specular parameter is worse

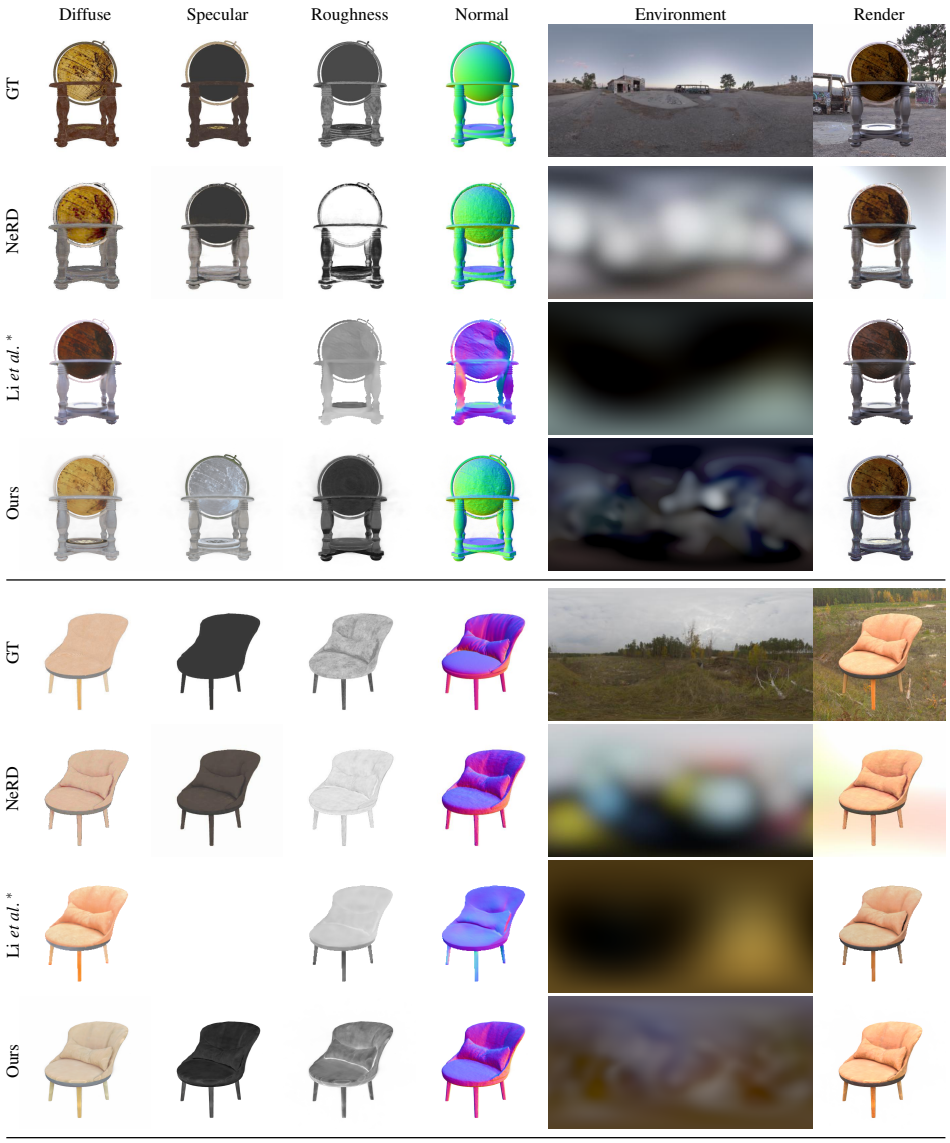



Figure 4.17: **Additional BRDF decomposition results.** Comparison with NeRD and Li *et al*. [98] on two synthetic scenes.

compared to NeRD. This may be due to NeRD's basecolor-metallic parameterization, which can reduce some ambiguity but also limits the space of expressible materials. A visual comparison is shown in Fig. 4.16a demonstrating clear visual improvements w.r.t. [98]. Further results are visible in Fig. 4.17. One can observe higher frequency details in the environment map using our approach compared to NeRD, and the final renderings also show that our result is closer to GT rendering (top-right). Refer to the supplementary material for more visual results.

#### 4.3.4.5 View synthesis and relighting

On the datasets with fixed illumination (Cape, NeRF-Ship, NeRF-Chair, NeRF-Lego), we can directly compare our renderings with existing novel view synthesis techniques (both NeRF [117] and our previous work NeRD). Tab. 4.7b shows novel view evaluation metrics on these datasets with fixed illumination. Results show that our results are better than NeRD, showing the improved capture of view-dependent effects. NeRF still outperforms NeRD and our method in the synthetic fixed illumination setting but is outperformed in the real-world fixed illumination dataset. However, the fixed illumination might, in general, limit the decomposition capabilities, as shadows always appear at the exact surface locations and, therefore, might not be correctly disentangled from the BRDF.

On datasets with varying illumination across images (Gnome, MotherChild, Chair, Car, Globe, Head), we need to do both view synthesis and relighting to generate novel unseen test views. Tab. 4.7c displays the results on these datasets. NeRF [117] can not do relighting and is included as a weak baseline. The results are significantly better than NeRD and show that our method can more faithfully estimate the underlying parameters resulting in better relighting under novel illumination conditions.

Fig. 4.16b shows a couple of results with view synthesis and relighting. The renderings demonstrate realistic view synthesis, including relighting. Fig. 4.16c shows a novel view synthesis comparison with NeRF and NeRD on the Cape scene captured with fixed illumination. A comparison for all datasets on an exemplary novel view is shown in Fig. 4.18. Here, we highlight the detail recovered by our method. We further show the relighting performance of our method in Fig. 4.19, which demonstrates that our method can produce a convincing relight of the scenes. For the single illumination scene (Cape), we show the effect of a rotating environment illumination. Despite NeRF being a strong baseline, it could not recover the entire surface due to reflectiveness. On the other hand, our view synthesis results are closer to the GT on unseen views than NeRF and NeRD.

(a) **BRDF decomposition**

| PSNR↑ | [98] | [98]+[117] | [25] | Ours |
|---|---|---|---|---|
| Diffuse | 1.06 | 1.15 | 18.24 | 20.22 |
| Specular | — | — | 25.70 | 16.84 |
| Roughness | 17.18 | 17.28 | 15.00 | 24.82 |

(b) **View synthesis**

| | Synthetic | | Real-World | |
|---|---|---|---|---|
| Method | PSNR↑ | SSIM↑ | PSNR↑ | SSIM↑ |
| NeRF | 34.24 | 0.97 | 23.34 | 0.85 |
| NeRD | 30.07 | 0.95 | 23.86 | 0.88 |
| Ours | 30.08 | 0.95 | 23.95 | 0.90 |

(c) **View synthesis and relighting**

| | Synthetic | | Real-World | |
|---|---|---|---|---|
| Method | PSNR↑ | SSIM↑ | PSNR↑ | SSIM↑ |
| NeRF | 21.05 | 0.89 | 20.11 | 0.87 |
| NeRD | 27.96 | 0.95 | 25.81 | 0.95 |
| Ours | 29.24 | 0.96 | 26.23 | 0.95 |

Table 4.7: **Comparisons with baselines.** In (a), a comparison against methods for BRDF decomposition under unknown illuminations, where we see that our model performs well (consistent with our improved relighting performance). An evaluation of view-synthesis (without relighting) under a single illumination is performed in (b), where our model performs well despite this not being our primary task. In (c), input images are taken under different illumination conditions, so joint relighting and view synthesis are required. Our model outperforms both baselines by a significant margin: NeRF (which is not intended to address this task) and NeRD (which targets this same problem statement).


Figure 4.18: **Results from each scene.** Comparison with NeRF, NeRD and Neural-PIL for every scene.


Figure 4.19: **Additional relighting results.** Relighting of various scenes under the source illumination is shown in the insets. For the last row, the illumination is rotated.

## 4.4 SAMURAI: Shape And Material from Unconstrained Arbitrary Image collections

*This section is based on the publications*:

SAMURAI: Shape And Material from Unconstrained Arbitrary Image collections
Mark Boss, Andreas Engelhardt, Abhishek Kar, Yuanzhen Li, Deqing Sun,
Jonathan T. Barron, Hendrik P. A. Lensch, Varun Jampani
*Advances in Neural Information Processing Systems (NeurIPS)* - 2022

In the previous sections, we leverage COLMAP [145, 146] to reconstruct the camera poses from image collections. However, in highly challenging datasets, COLMAP can fail. COLMAP leverages correspondences to match the different images and drive the reconstruction. If an object is captured in different locations and with highly varying illuminations, finding correspondences that can match is near impossible. Most traditional camera reconstruction methods will fail in these conditions. We propose SAMURAI to enable a reconstruction from these challenging image collections. We achieve this by creating a *Camera Multiplex*. Here, we optimize several camera assumptions for each image simultaneously.

In our previous works, we also used the assumption of NeRF [117] (Sec. 2.4.3) that all images are captured at a similar distance to the object. We enable a non-equidistant camera by introducing a volume-bound definition alongside a flexible camera parametrization. With these novel additions, we can reconstruct 3D relightable assets from arbitrary online image collections.

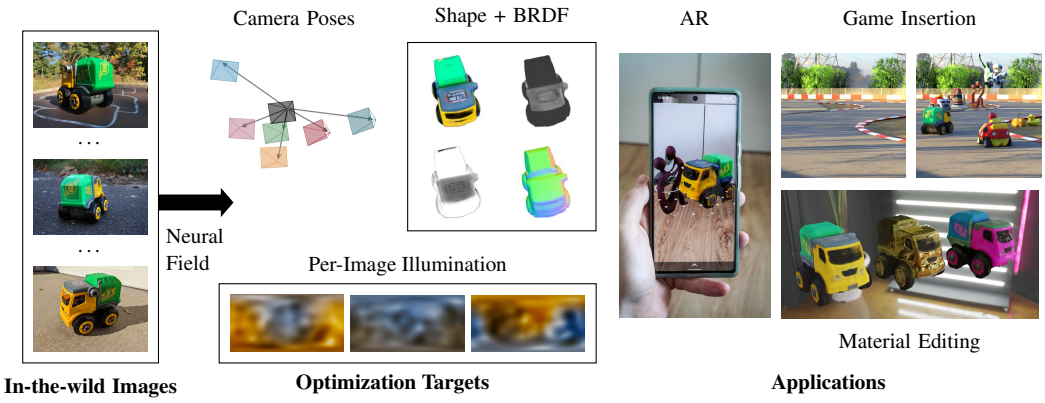



Figure 4.20: **Sample SAMURAI outputs and applications.** Sample input collection and the outputs on a challenging real-world unconstrained image collection. We extract meshes with material properties from the learned volumes enabling several applications in AR/VR, material editing *etc*.

Capturing high-quality 3D shapes and materials of real-world objects is essential for many graphics applications in AR, VR, games, movies, *etc*. Using active multi-view object capture setups can provide high-quality 3D assets [22, 122] but cannot scale to a large-scale set of objects present in the world. By contrast, image collections provided by image search results or product review images exist for nearly every object. In this work, we propose a category-agnostic technique to estimate the 3D shape and material properties of objects from such Internet image collections. Estimating 3D shapes and materials from Internet object image collections poses several challenges as the images are highly unconstrained with varying backgrounds, illuminations, and camera intrinsics. Fig. 4.20 (left) shows a sample image collection of an object which forms the input to our technique.

Concretely, we estimate the 3D shape and BRDF material properties [41] while estimating per-image illumination, camera poses, and intrinsics. Our previous works NeRD (Sec. 4.2) and Neural-PIL (Sec. 4.3), alongside several contemporary works on shape and material estimation [22, 150, 194] assume constant camera intrinsics, near-perfect segmentation masks as well as almost-correct camera poses given by COLMAP. However, manually annotating object masks in the input images is tedious. We observe that COLMAP can often fail on highly-challenging image collections, as COLMAP leverages correspondences to match the different images and drive the reconstruction. If an object is captured in different locations and with highly varying illuminations, finding correspondences that match is near impossible. Most traditional camera reconstruction methods will also fail in these conditions. Instead of COLMAP, we use a rough quadrant-based pose initialization, *e.g*. (Front, Above, Right), (Front, Below, Left), *etc*., as in NeRS [193], which usually takes only a few minutes of annotation time per image collection.

We base our technique on our previous Neural-PIL method of Sec. 4.3 that proposes to learn illumination priors along with a novel pre-integration illumination network for estimating a neural volume with 3D shape, BRDF, and per-image illumination. Neural-PIL assumes perfect camera poses and the same camera intrinsics across images. Given that traditional camera pose estimations like COLMAP may fail on in-the-wild images, we propose SAMURAI to jointly optimize camera poses and intrinsics with a carefully designed optimization protocol (Fig. 4.20). Furthermore, Neural-PIL requires perfect object masks, whereas we leverage automatically estimated object masks and deal with noisy masks using a posterior scaling loss.

Some key distinguishing features of SAMURAI include:

- *Flexible camera parametrization for varying distances.* Standard techniques such as NeRF [117] assume fixed near/far clipping planes with equidistant cameras to the object. In contrast, we define the neural volume in global coordinates and propose to learn clipping planes per image.

- *Camera multiplex optimization.* Optimizing a single camera per image is prone to getting stuck in local minima. We propose using a multiplex camera where we

optimize several cameras poses per image and then phase out the incorrect poses throughout the optimization. Although camera multiplexes are previously used in mesh optimization [60], optimizing camera multiplex with neural volumes is challenging due to inefficient ray-based neural volume rendering.

- *Posterior scaling of input images.* As input images have different noise characteristics (*e.g.*, noisy masks), some images would be more beneficial for the optimization. We propose to use posterior scaling of input images which weighs the influence of different images on the optimization.

- *Mesh extraction.* We extract explicit meshes with BRDF texture from the learned neural volume making the resulting 3D models readily usable in existing graphics engines.

We observe that existing datasets such as the one gathered for NeRD (Sec. 4.2.3.1) do not capture the variations present in in-the-wild image collections. For instance, NeRD-dataset images have non-varying background making it easier for COLMAP to work. In addition, the illumination variations are more drastic in internet images captured by different people/cameras and at different times. To evaluate the practical in-the-wild setting, we collected image collections with eight objects in which each image is captured under unique background and illumination conditions. In addition, we also vary the cameras used for capturing the images. Experiments on our new and existing datasets demonstrate better view synthesis and relighting results with SAMURAI compared to existing works. In addition, explicit mesh extraction allows for seamless use of learned 3D assets in graphics applications such as object insertion in AR or games and material editing *etc*. Fig. 4.20 (right) shows some sample application results with 3D assets estimated using SAMURAI.

### 4.4.1 Problem setup

The input is a collection of $q$ object images $C_j \in \mathbb{R}^{s_j \times 3}; j \in \{1, \ldots, q\}$ captured with different backgrounds, cameras and illuminations; and can also have varying resolutions. We denote the value of a specific pixel as $\boldsymbol{C}^s$. In addition, we roughly annotate camera pose quadrants with three simple binary questions: Left *vs*. Right, Above *vs*. Below, and Front *vs*. Back. We automatically estimate foreground segmentation masks $M_j \in \{0,1\}^{s \times 1}$ using U$^2$-Net [132], which can be imperfect. Given these, we jointly optimize a 3D neural volume with shape and BRDF material information along with per-image illumination, camera poses, and intrinsics. This practical capture setup allows the conversion of most 2D image collections into a 3D representation with little manual work. The rough pose quadrant annotation takes about a few (3-5) minutes for a typical collection with 80 images. At each point $\boldsymbol{x} \in \mathbb{R}^3$ in the 3D neural volume $\mathcal{V}$, we estimate the BRDF parameters for the Cook-Torrance model [41] $\boldsymbol{b} \in \mathbb{R}^5$ (basecolor $\boldsymbol{b}_c \in \mathbb{R}^3$, metallic $\boldsymbol{b}_m \in \mathbb{R}^1$, roughness $b_r \in \mathbb{R}$), unit-length surface normal $\boldsymbol{n} \in \mathbb{R}^3$

and volume density $\sigma \in \mathbb{R}$. We also estimate the latent per-image illumination vectors $\boldsymbol{z}_j^l \in \mathbb{R}^{128}; j \in \{1,\ldots,q\}$ used in our previous work Neural-PIL of Sec. 4.3. We also estimate per-image camera poses and intrinsics, which we represent using a 'look-at' parameterization that we explain later.

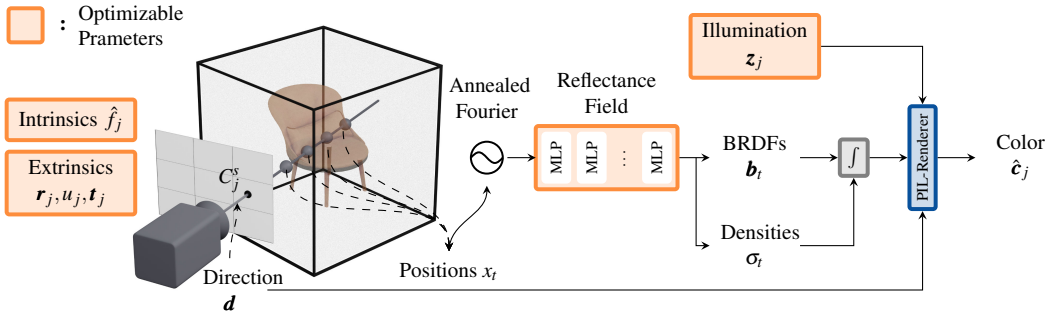



Figure 4.21: **Overview.** We jointly optimize the intrinsic ($\hat{f}_j$) and extrinsic camera ($\boldsymbol{r}_j, u_j, \boldsymbol{t}_j$) parameters alongside the shape ($\sigma$) and BRDF ($\boldsymbol{b}$) in a *Reflectance Field* and per-image illumination ($\boldsymbol{z}_j$). The shape is encoded in the density $\sigma$ and used to integrate all BRDFs along a ray in the direction $\boldsymbol{d}$. The composed BRDF is then rendered using our previous work Neural-PIL (Sec. 4.3) in a deferred rendering style.

## 4.4.2 Network Architecture

The main limitations of Neural-PIL include the assumption of near-perfect camera poses and the availability of perfect object segmentation masks. We observe that COLMAP either produces incorrect poses or completely fails due to an insufficient number of correspondences across images when the backgrounds and illuminations are highly varying across the image collection. In addition, camera intrinsics could vary across image collections, and the automatically estimated object masks could also be noisy. We propose a technique (we refer to as 'SAMURAI') for joint optimization of 3D shape, BRDF, per-image camera parameters, and illuminations for a given in-the-wild image collection. This is a highly under-constrained and challenging optimization problem when only image collections and rough camera pose quadrants are given as input. We address this highly challenging problem with carefully designed camera parameterization and optimization schemes.

In this work, we follow the image formation as described in Sec. 4.3.2 for the rendering.

### 4.4.2.1 Architecture overview

A high-level overview of SAMURAI architecture is shown in Fig. 4.21, which mostly follows the architecture of our previous sections: NeRD (Sec. 4.2) and Neural-PIL

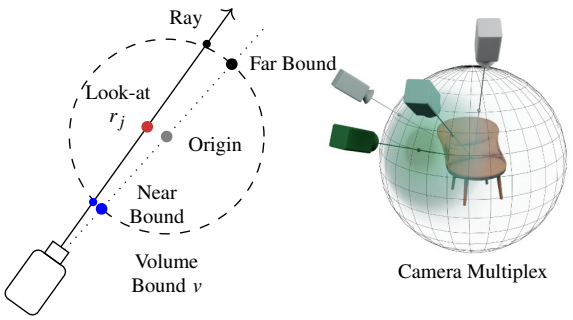



Figure 4.22: **Ray parametrization and Camera Multiplex.** (Left) A world space sphere defines our ray bounds. The distance from the origin to the near and far points of the sphere defines the sampling range. Our cameras can be placed at arbitrary distances by defining a globally consistent sampling range. (Right) When optimizing multiple camera hypotheses, only the best camera should optimize the shape and appearance, here visualized in a deep green. Cameras that are not aligned well are visualized in off-white and cannot influence the reflectance field. When the non-aligned camera poses improve during the training, they may become applicable for the network optimization.

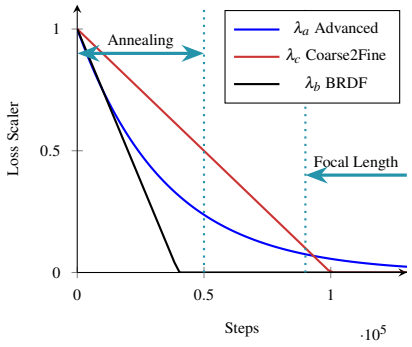



Figure 4.23: **Optimization scheme.** Our method performs a smooth flow of optimization parameters using three $\lambda$ variables for loss scaling. Additionally, we perform a Fourier frequency annealing in the first phase of the training and delay the training of the focal length for later stages. The $\lambda_b$ parameter mainly regulates the BRDF estimation.

(Sec. 4.3. However, we do not use a coarse network for efficiency reasons and only use the fine or decomposition network with an MLP with eight layers of 128 features. Overall we sample 128 points along the ray in fixed steps. A layer with one feature output follows the base network for the density, and an MLP with a hidden layer for the view and appearance conditioned radiance. We also leverage a BRDF decoder similar to the one of NeRD in Sec. 4.2.2.2, which first compresses the feature output of the main network to 16 features and expands them again to the BRDF (base color, metallic, roughness). We encourage sparsity in the embedding space using: $\mathcal{L}_{\text{Dec Sparsity}} = {}^1/_N \sum_i^N |\boldsymbol{e}_i|$, an $\mathcal{L}_1$-Loss on the BRDF embedding $\boldsymbol{e}$ and a smoothness loss $\mathcal{L}_{\text{Dec Smooth}} = {}^1/_N \sum_i^N |f(\theta;\boldsymbol{e}_i) - f(\theta;\boldsymbol{e}_i + \boldsymbol{\varepsilon})|$, where $N$ denotes the number of random rays, $f(\theta)$ the BRDF decoder with the weights $\theta$ and $\varepsilon$ is normal distributed Gaussian noise with a standard deviation of 0.01. Similar to NeRD, we also predict a regular direction-dependent radiance $\tilde{\boldsymbol{c}}$ in the early stages of the training. This is mainly used for stabilization in the early stages. As this direct color prediction is only used in the early stages, we omitted it for clarity in Fig. 4.21. Inspired by Tancik *et al.* [155] we add a Gaussian distributed noise to the Fourier embedding. The Fourier Encoding $\gamma : \mathbb{R}^3 \mapsto \mathbb{R}^{3+6L}$ used in NeRF [117] encodes a 3D coordinate $\boldsymbol{x}$ into $L$

frequency basis:

$$\gamma(\boldsymbol{x}) = (\boldsymbol{x}, \Gamma_1, \ldots, \Gamma_{L-1}) \tag{4.7}$$

where each frequency is encoded as:

$$\Gamma_k(\boldsymbol{x}) = \left[\sin(2^k \boldsymbol{x}), \cos(2^k \boldsymbol{x})\right] \tag{4.8}$$

BARF [100] and Nerfies [128] introduced an annealing of the Fourier Frequencies using a weighting:

$$\Gamma_k(\boldsymbol{x}; \alpha) = w_k(\alpha) \left[\sin(2^k \boldsymbol{x}), \cos(2^k \boldsymbol{x})\right] \tag{4.9}$$

$$w_k(\alpha) = \frac{1 - \cos\left(\pi \operatorname{clamp}(\alpha - k, 0, 1)\right)}{2} \tag{4.10}$$

where $\alpha \in [0, L]$. This can be seen as a truncated Hann window. One downside of this form of encoding is that all frequencies are axis-aligned. In Tancik *et al*. [155] the benefits of adding random frequencies are demonstrated. However, combining this with the sliding cosine window is not easily possible. Therefore, we propose to add random Gaussian offsets $\boldsymbol{R} \in \mathbb{R}^{L \times 3}$ to the frequencies. The offsets $\boldsymbol{R}$ are sampled from $N(0, 0.1)$. This can be thought of randomly rotating each frequency band:

$$\Gamma_k(\boldsymbol{x}; \alpha) = w_k(\alpha) \left[\sin(2^k \boldsymbol{x} + 2^k \boldsymbol{R}_k), \cos(2^k \boldsymbol{x} + 2^k \boldsymbol{R}_k)\right] \tag{4.11}$$

#### 4.4.2.2 Rough camera pose quadrant initialization

We observe that camera pose optimization is a highly non-convex problem and tends to get stuck in local minima quickly. To combat this, we propose to annotate camera pose quadrants with three simple binary questions: Left *vs*. Right, Above *vs*. Below, and Front *vs*. Back. This only takes about 4-5 minutes for a typical collection with 80 images. Note that our pose quadrant initialization is much noisier than adding some noise around GT camera poses as in some related works such as NeRF-- [176]. This rough pose initialization is in line with recent works such as NeRS [193] that also use rough manual pose initialization.

#### 4.4.2.3 Flexible object-centric camera parameterization for varying camera distances

We define the trainable per-image camera parameters using a 'look-at' parameterization with a 3D look-at vector $\boldsymbol{r}_j \in \mathbb{R}^3; j \in \{1, \ldots, q\}$, a scalar up rotation $u_j \in \mathbb{R}[-\pi, \pi]$ and a 3D camera position $\boldsymbol{t}_j \in \mathbb{R}^3$ as well as a focal length $f_j \in \mathbb{R}$. Furthermore, these are

stored as offsets to the initial parameters, enabling easier regularization. We additionally store the offset vertical focal lengths $f_j$ in a compressed manner similar to NeRF--: $\hat{f}_j = \sqrt{f_j/h}$ [176], where $h$ is the image height in pixels. The cameras are initialized based on the given pose quadrants and an initial field of view of 53.13 degrees. We optimize a perspective pinhole camera with a fixed principal point but per-image focal lengths. The cameras are not always equidistant to the object for the in-the-wild image collections. To account for variable camera-object distances, we do not set fixed near and far bounds for each ray which is a standard practice in neural volumetric optimizations such as NeRF [117]. Instead, we define a sampling range based on the camera distance to origin, *e.g.* the near bound is $|\boldsymbol{o}| - v$ and the far bound is $|\boldsymbol{o}| + v$, where $v$ is defined as our sampling radius with a diameter of 1. We illustrate this sphere with near and far bounds in Fig. 4.22. This explicit computation of near and far bounds for each ray enables placing the cameras at arbitrary distances from the object. This is not possible with the existing neural volume optimization techniques that use fixed near and far bounds for each camera ray. The cameras are then placed based on the quadrants and at a distance to make the entire neural volume $v$ visible. This look-at parameterization is more flexible for optimizing object-centric neural volumes than more commonly used 3D rotation matrices.

#### 4.4.2.4 Camera multiplexes

We observe that camera pose optimization gets stuck in local minima even with rough quadrant pose initialization. To combat this, inspired by the mesh optimization works of Goel *et al.* [60], we propose to optimize a camera multiplex with four randomly jittered poses around the quadrant center direction for each image. Optimizing multiple cameras per image would reduce the number of rays we can cast in a single optimization step due to memory and computational limitations. This makes camera multiplex optimization noisy and challenging in learning neural volumes. We propose techniques to make camera multiplex learning more robust by dynamically re-weighing the loss functions associated with different cameras in a multiplex during the optimization. This process is visualized in Fig. 4.22. Specifically, we compute the mask reconstruction loss $\mathcal{L}_{\text{Mask}}{}^{i}_{j} \in \mathbb{R}$ associated with each camera $i$ and image $j$. We then re-weigh each camera loss in a multiplex with $S_j = softmax(-\lambda_s \mathcal{L}_{\text{Mask}}{}^{i}_{j})$, where $S_j \in \mathbb{R}^4$ and $\lambda_s$ is a scalar that is gradually increased during the optimization. That is, we re-weigh the loss with $\mathcal{L}_{\text{Network}\,j} = \sum_i S^i_j \mathcal{L}_{\text{Network}}{}^{i}_{j}$. This dynamic re-weighing reduces the influence of bad camera poses while learning the shape and materials. As we stochastically sample points for each batch, a potential bad camera can have favorable samples and outperform a better camera. We alleviate this issue by storing the weights for each of our optimization images in a memory bank $\boldsymbol{W} \in \mathbb{R}^{j\times 4}$. These can then be updated during the optimization and reduce the impact of the sample distributions. Furthermore, we store a memory bank of velocities $\boldsymbol{V} \in \mathbb{R}^{j\times 4}$ to speed up the selection of the best camera pose. The weight matrix is then updated with the new camera multiplex weights $S_j$ using:

$$\boldsymbol{W}_j^* = \max(\boldsymbol{W}_j + m\boldsymbol{V}_j^* + g, 0) \tag{4.12}$$

$$\boldsymbol{V}_j^* = m * \boldsymbol{V}_j + \boldsymbol{g} \tag{4.13}$$

$$\boldsymbol{g} = s(\boldsymbol{S}_j - \boldsymbol{W}_j) \tag{4.14}$$

Where the new weights $\boldsymbol{W}_j^*$ and velocities $\boldsymbol{V}_j^*$ replace the old ones, the parameters $m$ represent the momentum and $s$ the learning rate. The values for these are 0.75 and 0.3, respectively.

#### 4.4.2.5 Posterior scaling of input images

Some images are blurrier than others (*e.g.*, due to camera shake) or noisy object masks $M_j$. To be robust against such noisy data, we propose to re-weigh images in the given collection. We keep a circular buffer of around 1000 elements with the recent mask losses and rendered image losses with multiplex scaling applied. We use this buffer to calculate the mean $\mu_l$ and standard deviation $\sigma_l$ of these losses. Given the recent loss statistics we also create a loss scalar using:

$$s_{\mathrm{p}\,j} = \max(\tanh\left(\frac{\mu_l - (\mathcal{L}_{\mathrm{Mask}\,j} + \mathcal{L}_{\mathrm{Image}\,j})}{\sigma_l}\right) + 1, 1) \tag{4.15}$$

In a similar way to the camera poserior scaling, we employ it on a per-image basis using: $\mathcal{L}_{\mathrm{Network}\,j} = s_{\mathrm{p}\,j}\,\mathcal{L}_{\mathrm{Network}\,j}$.

#### 4.4.2.6 Mesh extraction

Similar to NeRD, we perform a mesh extraction from the learned reflectance neural volume. The process for NeRD is described in Sec. 4.2.2.6, but we alter the extraction method. In NeRD, the object is placed depending on the poses. COLMAP can place the actual objects at arbitrary positions in the volume. In SAMURAI, we have an explicitly defined volume bound.

In the first step, we perform a marching cubes extraction step similar to the one proposed in NeRF [117]. However, as the naive marching cube algorithm can have block artifacts, we sample 2 million points on the mesh surface and cast rays towards the surface. The resulting point cloud is converted to a refined mesh using Poisson reconstruction. This refined mesh provides more details and smoother surfaces. We then UV unwrap the resulting mesh in Blender's [40] automatic UV unwrapping tool and bake the world space positions into the texture map. We can then query all surface locations in our fine network and compute the BRDF texture maps. We then save the textured mesh in the GLB format for easy deployment. The extraction of a mesh takes around 3-5 minutes.

#### 4.4.2.7 Losses and Optimization

Our loss formulation consists of several for the direct image and segmentation mask reconstruction, regularization, and BRDF regularization and initialization losses. Furthermore, we introduce a smooth optimization schedule to transition to different optimization targets throughout the training.

**Image reconstruction loss.** The image reconstruction loss is a Chabonnier loss:

$$\mathcal{L}_{\text{Image}}(\tilde{\boldsymbol{c}}, \hat{\boldsymbol{c}}) = \sqrt{(\tilde{\boldsymbol{c}} - \hat{\boldsymbol{c}})^2 + 0.001^2} \tag{4.16}$$

between the input color from $C$ for pixel $s$ and the corresponding predicted color of the networks $\tilde{\boldsymbol{c}}$. We additionally calculate the loss with the rendered color $\hat{\boldsymbol{c}}$.

**Mask losses.** Mask losses consist of two terms. One is the binary cross-entropy loss $\mathcal{L}_{\text{BCE}}$ between the volume-rendered mask and estimated foreground object mask. The second one is the background loss $\mathcal{L}_{\text{Background}}$ from NeRD Sec. 4.2.2.7, which forces all samples for rays cast towards the background to be 0. We combine these losses as the mask loss: $\mathcal{L}_{\text{Mask}} = \mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{Background}}$.

**Regularization losses.** We compute the gradient of the density to estimate the surface normals. We use the normal direction loss $\mathcal{L}_{\text{ndir}}$ from Verbin *et al.* [168] to constrain the normals to face the camera until the ray reaches the surface. This helps in providing sharper surfaces without cloud-like artifacts.

**BRDF losses.** The joint estimation of BRDF and illumination is quite challenging. For example, the illumination can fall into a local minimum. The object is then tinted in a bluish color, and the illumination is an orange color to express a more neutral color tone. As our image collections have multiple illuminations, we can force the base color $\boldsymbol{b}_c$ to replicate the pixel color from the images. This way, a mean color over the dataset is learned and prevents falling into the local minima. We leverage the MSE for this: $\mathcal{L}_{\text{Init}} = \mathcal{L}_{\text{MSE}}(\boldsymbol{C}^{\boldsymbol{s}}, \boldsymbol{b}_c)$. Additionally, we find that a smoothness loss $\mathcal{L}_{\text{Smooth}}$ for the normal, roughness and metallic parameters similar to the one used in UNISURF [126] further regularizes the solution.

**Overall network and camera losses.** The final loss to optimize the decomposition network is then defined as:

$$\mathcal{L}_{\text{Network}} = \lambda_b \mathcal{L}_{\text{Image}}(\boldsymbol{C}^{\boldsymbol{s}}, \tilde{\boldsymbol{c}}) + (1-\lambda_b)\mathcal{L}_{\text{Image}}(\boldsymbol{C}^{\boldsymbol{s}}, \hat{\boldsymbol{c}}) + \mathcal{L}_{\text{Mask}} + \lambda_a \mathcal{L}_{\text{Init}} + \lambda_{\text{ndir}}\mathcal{L}_{\text{ndir}} + \lambda_{\text{Smooth}}\mathcal{L}_{\text{Smooth}} + \lambda_{\text{Dec Smooth}}\mathcal{L}_{\text{Dec Smooth}} + \lambda_{\text{Dec Sparsity}}\mathcal{L}_{\text{Dec Sparsity}} \quad (4.17)$$

Here, the optimization scheduling variables are $\lambda_b$ and $\lambda_a$. Furthermore, the camera posterior scaling is applied to these losses. For the camera optimization, we leverage the same losses described in $\mathcal{L}_{\text{Network}}$. However, as the camera should constantly be optimized, we do not apply posterior scaling to the losses when optimizing cameras. This enables cameras that are not aligned well to be optimized and allows each camera to leave the local minima. Here, we only calculate the mean loss. Therefore, poorly initialized camera poses can still recover over the training duration. Additionally, we define an $\mathcal{L}_1$ loss on the look-at vector $\boldsymbol{r}_j$ to constrain the camera pose to look at the object. The loss is defined as $\mathcal{L}_{\text{lookat}}$. We also use a volume padding loss, which prevents cameras from going too far into our volume bound $v$: $\mathcal{L}_{\text{Bounds}} = \max((v - |\boldsymbol{t}_j|)^2, 0)$.

#### 4.4.2.8 Optimization scheduling

Fig. 4.23 shows the optimization schedule of different loss weights. We use three fading $\lambda$ variables to transition the optimization schedule smoothly. The $\lambda_c$ is mainly used to increase image resolution and reduce the number of active multiplex cameras. The direct color $\tilde{\boldsymbol{c}}$ optimization is faded to the BRDF optimization using $\lambda_b$ and some losses are scaled by $\lambda_a$ as defined earlier. Furthermore, we perform the BARF [100] frequency annealing in the early stages of the training and delay the focal length optimization to the later stages.

The input images for our network are used without cropping. We sample the foreground area thrice as often as the background regions to circumvent the potential large background areas. As the resolution varies drastically and can be large, we further resize the images so that the largest dimension is 400 pixels.

We use two different optimizers. ADAM [82] optimizes the networks with a learning rate of 1e$-$4 and exponentially decayed by an order of magnitude every 300k steps. The camera optimization is performed with a learning rate of 3e$-$3 and exponentially decayed by an order of magnitude every 70k steps.

The detailed configuration of our network is shown in Fig. 4.24. We use 10 Random Offset Annealed Fourier Frequencies for the positional encoding. These are annealed over 50000 steps using Eq. (4.10). The directions are encoded using 4 non-annealed and non-offset Fourier frequencies. The losses in Eq. (4.17) are weighted with the following scalars besides the optimization schedule scalars $\lambda_b$, $\lambda_a$:

| $\lambda_{\text{ndir}}$ | $\lambda_{\text{Smooth}}$ | $\lambda_{\text{Dec Sparsity}}$ | $\lambda_{\text{Dec Smooth}}$ |
|---|---|---|---|
| 0.005 | 0.01 | 0.01 | 0.1 |

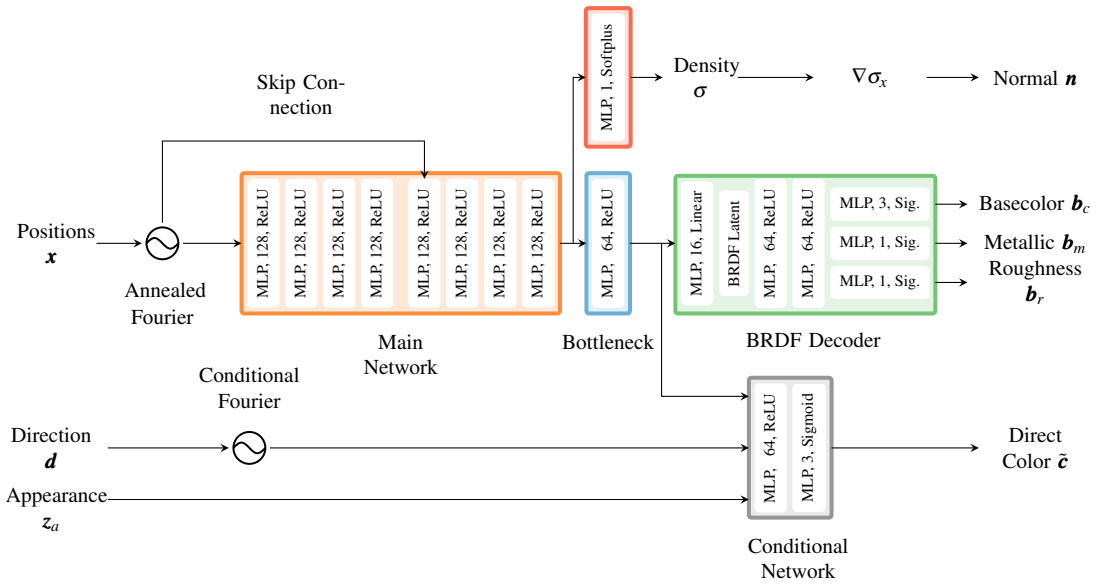



Figure 4.24: **Architecture.** The detailed architecture of our network. Note that the conditional network and the direct color is only used in the early stages of the training for stabilization. It does not contribute to the final decomposition result. Our main outputs include the Density $\sigma$, the normal $\boldsymbol{n}$ and the BRDF ($\boldsymbol{b}_c$, $\boldsymbol{b}_m$, $\boldsymbol{b}_r$), which are used for rendering our actual output color $\hat{\boldsymbol{c}}$ with Neural-PIL of Sec. 4.3.3.6.

The coarse to fine optimization is further governed by $\lambda_c$. This parameter mainly interpolates between the available resolution of the largest dimension from 100 to 400 pixels and the number of cameras from 4 to 1. The softmax scalar $\lambda_s$ is also driven by $\lambda_c$ and fades from a scalar of 1 to 10.

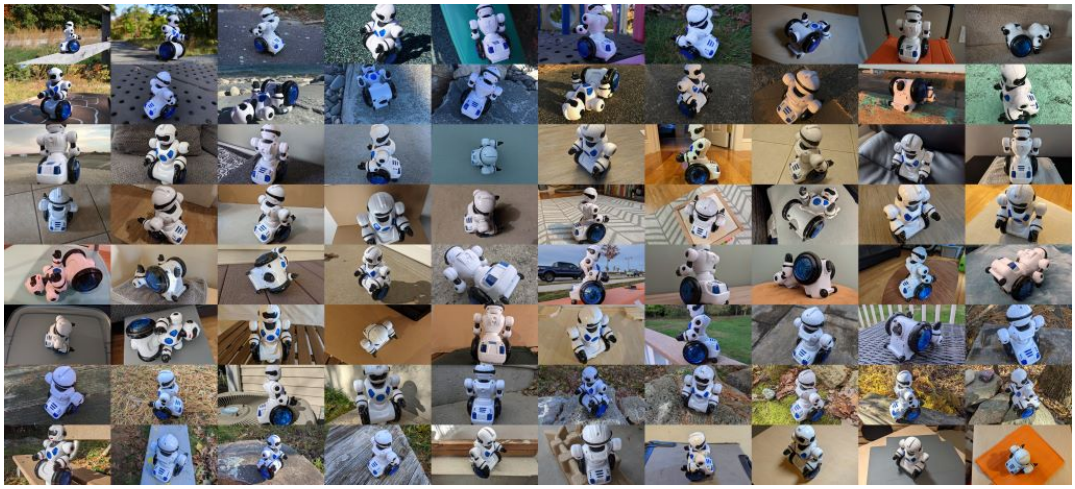
Furthermore, we apply gradient scaling to the gradients for the network by the norm of 0.1. The camera gradients are neither clipped nor scaled.

## 4.4.3 Results

We evaluate our approach w.r.t. different baselines on the aspects of BRDF, view synthesis, relighting, and camera pose estimation.

### 4.4.3.1 Datasets

For evaluations, we created new datasets of 8 objects (each with 80 images) captured under unique illuminations and locations and a few different cameras. We refer to this dataset as the SAMURAI dataset. We tried to replicate the online collection setting as much as possible by using different cameras (Pixel 4a, iPhone 7 Plus, Sony alpha 6000), capturing the objects in different unique environments, and replicating the



(a) Robot

(b) Fire Engine

Figure 4.25: **Dataset overview.** Notice the complex illumination conditions and the drastically varying locations. Also, the distances vary quite severely.

hand-held capture setup with varying distances. Even with an extensive manual tuning of parameters, we cannot estimate the camera poses in traditional methods such as COLMAP [145, 146]. Fig. 4.25 shows an overview of the images in two image collections. Common methods such as COLMAP fail to estimate correspondences and camera poses for this dataset. Therefore, we cannot run methods that require poses on this dataset. Additionally, we evaluate 2 CC-licensed image collections from online sources of the statue of liberty and a chair. We also use the three synthetic and two real-world datasets of NeRD under varying illumination, where poses are available. Lastly, to showcase the performance with other methods, we use the two real-world datasets from NeRD (Sec. 4.2.3.1), which are taken under fixed illumination. In total, we evaluate SAMURAI

on 17 scenes. A list of our datasets used in the evaluation is shown in Table 4.8.

| Scene | Multi-Illumination | Known Poses | Notes |
|---|---|---|---|
| Gold Cape | ✗ | ✓ | From Sec. 4.2.3.1 |
| Head | ✗ | ✓ | From Sec. 4.2.3.1 |
| Syn. CarWreck | ✓ | ✓ | Synthetic from Sec. 4.2.3.1 |
| Syn. Globe | ✓ | ✓ | Synthetic from Sec. 4.2.3.1 |
| Syn. Chair | ✓ | ✓ | Synthetic from Sec. 4.2.3.1 |
| Mother Child | ✓ | ✓ | From Sec. 4.2.3.1 |
| Gnome | ✓ | ✓ | From Sec. 4.2.3.1 |
| Statue of Liberty | ✓ | ✗ | Online collection |
| Chair | ✓ | ✗ | Online collection |
| Duck | ✓ | ✗ | Self-collected |
| Fire Engine | ✓ | ✗ | Self-collected |
| Garbage Truck | ✓ | ✗ | Self-collected |
| Keywest | ✓ | ✗ | Self-collected |
| Pumpkin | ✓ | ✗ | Self-collected |
| RC Car | ✓ | ✗ | Self-collected |
| Robot | ✓ | ✗ | Self-collected |
| Shoe | ✓ | ✗ | Self-collected |

Table 4.8: **List of datasets.** List of all datasets and the classification into multi-illumination and known poses.

#### 4.4.3.2 Baselines

Currently, no prior art can tackle varying illumination input images while jointly estimating camera poses. So, we compare with a modified BARF [100] technique, which can store per-image appearances in a latent vector. We call this baseline BARF-A. Additionally, on scenes with fixed illumination, we can compare with GNeRF [115], the regular BARF, and a modified version of NeRS [193] (details below). On the datasets where poses are easily recovered or given, we can also compare with NeRD (Sec. 4.2) and Neural-PIL (Sec. 4.3), which require known, near-perfect camera poses. We supply BARF, BARF-A, and NeRS with the same pose initialization used in SAMURAI.

**NeRS modifications.** The default implementation of NeRS [193] does not implement mini-batching. This means all images are optimized simultaneously in a resolution of $256 \times 256$. This works well for a few images, but the GPU memory runs out with larger

image collections. We have created a modified version that implements mini-batching for a fair comparison. Still, NeRS is only capable of working on single illumination datasets.

#### 4.4.3.3 Evaluation

We perform novel view synthesis using learned volumes and use standard PSNR and SSIM metrics w.r.t. ground-truth views. We held out every 16th image for testing. We optimize the cameras and illuminations on the test images for evaluation purposes but do not allow the test images to affect the main decomposition network or camera training.

We evaluate the quality of the reconstructed camera poses against a reference obtained from COLMAP [145, 146]. References are only available for the scenes of the NeRD dataset. First, we align the camera locations using Procrustes analysis [62] as in [100]. The rotation error is reported as a mean deviation in degrees, while the translation error is computed as the mean difference in scene units of the reference scene. In contrast to the evaluation of the view synthesis and rendering performance, we here use all cameras from the training data for comparison like it has been done in concurrent works [100, 115].

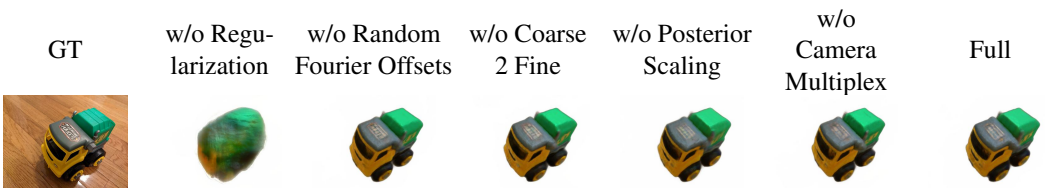



Figure 4.26: **Visual ablation.** Each of our novel additions improves the reconstruction. In this particular scene, the regularization is critical for the decomposition. The coarse to fine optimization, posterior scaling, and camera multiplex ablations mainly have a reduced sharpness in the sticker on the top. Without the random Fourier offset, striping patterns are apparent in some areas, which are alleviated with our full model.

#### 4.4.3.4 Ablation Study

We perform an ablation study where we ablate different aspects of the SAMURAI model to analyze their importance. Table 4.9 shows the novel view synthesis average metrics on the Garbage Truck collection from the SAMURAI dataset and the synthetic car dataset from Sec. 4.2.3.1. Metrics show that the regularization and the coarse to fine optimization are the most significant contributing factors to the final reconstruction quality. The multiplex cameras and the posterior scaling also improve the reconstruction quality, stabilizing the training and preventing cameras from getting stuck in local minima. In Fig. 4.26 the result of our ablation study is shown. The benefit of our regularization is easily apparent in this scene. Furthermore, our coarse-to-fine, posterior scaling, and

| Method | PSNR↑ | SSIM↑ |
|---|---|---|
| w/o Camera Multiplex | 23.01 | 0.87 |
| w/o Posterior Scaling | 23.51 | 0.88 |
| w/o Coarse 2 Fine | 22.73 | 0.83 |
| w/o Random Fourier Offsets | 24.01 | 0.91 |
| w/o Regularization | 21.77 | 0.86 |
| **Full** | 24.31 | 0.92 |

Table 4.9: **Ablation study.** view synthesis and relighting results on two scenes (Garbage Truck and NeRD car) show that ablating any of the proposed aspects of SAMURAI can results in worse results demonstrating their importance. This can be seen in Fig. 4.26.

camera multiplex help recover slightly sharper details but significantly help stabilize the optimization. The random Fourier offsets also alleviate some slight striping artifacts.

| | Poses Not Known (10) | | Poses Available (5) | | | |
|---|---|---|---|---|---|---|
| Method | PSNR↑ | SSIM↑ | PSNR↑ | SSIM↑ | Translation↓ | Rotation °↓ |
| BARF-A | 16.9 | 0.79 | 19.7 | 0.73 | 23.38 | 2.99 |
| **SAMURAI** | 23.46 | 0.90 | 22.84 | 0.89 | 8.61 | 0.86 |
| NeRD [25] | — | — | 26.88 | 0.95 | — | — |
| Neural-PIL [28] | — | — | 27.73 | 0.96 | — | — |

Table 4.10: **Novel view synthesis on varying illumination datasets.** We split our datasets into those where we have poses, and highly challenging ones where the poses were not recoverable with classical methods. SAMURAI achieves considerably better performance compared to BARF-A. For reference, we also show the metrics from our previous works (NeRD and Neural-PIL), which require GT poses and do not work on images with unknown poses.

#### 4.4.3.5 Results on varying illumination datasets

We divide the varying illumination datasets into those without GT poses, and those with accurate camera poses (either via GT or COLMAP). However, we do not grant our method access to the COLMAP, or GT poses. Since these datasets have varying illumination, we need to perform both view synthesis and relight the objects to obtain target views. Table 4.10 shows the metrics computed w.r.t. test views. Results show that we considerably outperform BARF-A in both PSNR and SSIM metrics while at the same time solving a more challenging BRDF decomposition task. Visual results


Figure 4.27: **Comparison with BARF-A.** When comparing novel view synthesis and relighting results of SAMURAI (top) with BARF-A (bottom), SAMURAI produces more accurate camera poses and captures the object better. BARF-A is sometimes unable to recover the shape and poses.

in Fig. 4.27 also clearly demonstrate better view synthesis and relighting results than BARF-A. BARF-A fails to align the camera poses, whereas SAMURAI achieves more accurate camera poses and drastically improved reconstruction quality. Only slightly perturbed poses are leveraged as starting positions in the original BARF method. Our coarse pose initialization is too noisy for the method to work accurately. SAMURAI overcomes this issue with the camera multiplex and other optimization strategies.

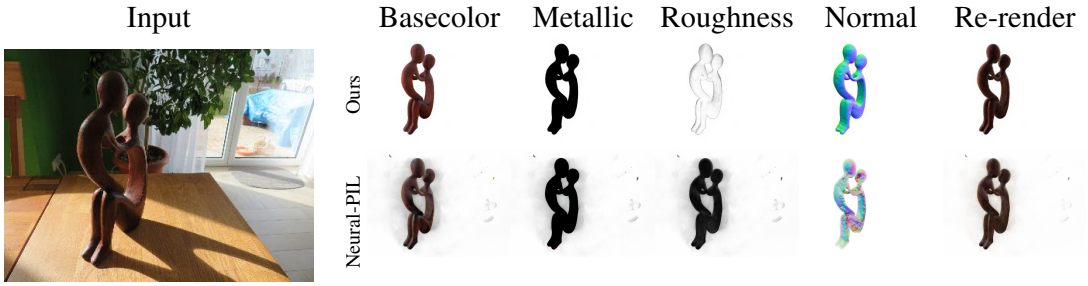



Figure 4.28: **Comparison with the Neural-PIL decomposition.** Notice SAMURAI's accurate pose alignment, plausible geometry, reduced floaters and accurate BRDF decomposition, compared to Neural-PIL, even without relying on near perfect poses.

Fig. 4.28 shows the visual comparison of BRDF decompositions on MotherChild dataset from NeRD of Sec. 4.2 along with the corresponding results from Neural-PIL of Sec. 4.3. Our method can generally decompose the scene even with unknown camera poses. SAMURAI also produces fewer floating artifacts and creates a more coherent surface. The roughness parameter is also more plausible in our result, as the object is rough, whereas Neural-PIL estimated a near mirror-like surface.

Further results from the SAMURAI dataset are shown in Fig. 4.29. We show novel views and relighting results w.r.t. the target test views. The visual comparisons clearly show that SAMURAI can recover the pose and provide a consistent illumination w.r.t the ground-truth target views. Even most fine details like the RC Car's antenna are preserved well. Only the legs of the chair object are not reproduced well. However, the legs are also not detected well by our automatic segmentation with U2-Net.


Figure 4.29: **Novel view synthesis results.** Renderings with camera poses and illumination from test views demonstrate plausible novel view synthesis and relighting on various datasets.

| Method | Pose Init | PSNR↑ | SSIM↑ | Translation↓ | Rotation °↓ |
|---|---|---|---|---|---|
| BARF [100] | Directions | 14.96 | 0.47 | 34.64 | 0.86 |
| GNeRF [115] | Random | 20.3 | 0.61 | 81.22 | 2.39 |
| NeRS [193] | Directions | 12.84 | 0.68 | 32.27 | 0.77 |
| **SAMURAI** | Directions | 21.08 | 0.76 | 33.95 | 0.71 |
| NeRD [25] | GT | 23.86 | 0.88 | — | — |
| Neural-PIL [28] | GT | 23.95 | 0.90 | — | — |

Table 4.11: **Novel view synthesis on single illumination datasets.** For two scenes under single illumination, the poses are easily recoverable. Furthermore, we can now compare with GNeRF which does not require pose initialization. As seen our method achieves a good performance. The view synthesis metrics with NeRD and Neural-PIL that use GT known poses are also shown for reference.

#### 4.4.3.6 Results on fixed illumination datasets

For image collections captured under fixed illumination, we can compare with more techniques. We compare with GNeRF, the default BARF, and NeRS. We additionally can compare with our previous work Neural-PIL and NeRD on the near-perfect camera poses recovered from COLMAP. Results in Table 4.11 show that SAMURAI outperforms the baselines BARF, GNeRF, and NeRS and is also close to Neural-PIL and NeRD that uses GT camera poses. GNeRF does not require a rough pose initialization. Overall, our method achieves a good pose recovery, where NeRS only slightly outperforms our method in the translational error due to some outliers in our case. These outliers do not degrade our reconstructions due to our image posterior loss.


Figure 4.30: **Comparison with baselines.** When comparing SAMURAI with the baselines (GNeRF, BARF, and NeRS), ours outperforms all methods in reconstruction quality and pose estimation.

Fig. 4.30 shows sample view synthesis results of SAMURAI, BARF, GNeRF, and NeRS on single illumination datasets. Visuals indicate better results with SAMURAI compared to GNeRF and BARF. NeRS seems to capture more apparent detail, but the general decomposition quality is significantly better in our method, where the cape gold material is represented more accurately. NeRS also introduces a misaligned face texture in the Head scene, where two faces are visible. Furthermore, NeRS is not capable of perfectly optimizing the poses. This is visible in Cape 1 and Head 1.

#### 4.4.3.7 View direction Radiance Conditioning Entanglement in BARF

Based on the view direction, BARF conditions the output radiance similar to NeRF [117]. In Fig. 4.31 we show the effect of a fixed camera with varying directional conditioning of the radiance. The texture starts to shift on the surface. BARF reduced the photometric training error of unaligned poses by slightly shifting the texture for these views. This can result in shifting textures in novel view synthesis. With the shifting textures, the shape representation worsens, and the pose alignment quality is further limited.

We found that this is also true for our modified BARF-A baseline. The view direction embedding is also shifting the texture on the surface, as seen in Fig. 4.31. However, in BARF-A, the illumination is modeled as an appearance embedding. This results in an even higher ambiguity in the representation of high-frequent surface details. The ambiguity is visible when we interpolate between two appearance embeddings in Fig. 4.32. The texture is now also shifting based on the appearance embedding.

Our BRDF decomposition creates a view-independent texture representation, and our illumination is global and does not influence the static texture information. With this representation shifting the textures is impossible, and the texture can only remain static. Especially in pose alignment, this is highly beneficial as each camera pose has to align to a globally static model. This also explains our improvement in quality in novel view synthesis in Table 4.10 and Table 4.11.

#### 4.4.3.8 Applications

One of the contributions of this work is the extraction of explicit mesh with material properties from the learned neural reflectance volume. The process is described in the supplements. The resulting mesh can be realistically placed in an Augmented Reality (AR) scene or a 3D game. In addition, one could edit the BRDF materials on the recovered mesh. See Fig. 4.20 for sample results of these applications, where our recovered 3D assets blend well in a given 3D scene.

#### 4.4.3.9 Limitations

SAMURAI achieves large strides in decomposing in-the-wild image collections compared to the prior art. However, we still rely on rough pose initialization. GNeRF

proposes a reconstruction technique without pose initialization, but it fails on the challenging in-the-wild datasets. Furthermore, SAMURAI produces slightly blurry textures. This is especially noticeable in the cape scene in Fig. 4.30. Here, the cape has a repeating, high-frequent texture. Reconstruction of this high-frequency texture requires near-perfect camera poses. Since this dataset is in a single location and illumination, COLMAP-based pose estimation outperforms SAMURAI-based pose alignment. However, SAMURAI enables the reconstruction of highly challenging datasets of online image collections where COLMAP completely fails. Our BRDF and illumination decomposition are also incapable of modeling shadowing and inter-reflections. As we mainly tackle object decomposition, the shadows and inter-reflections are not crucial. Removing the need for pose initialization and modeling shadows and inter-reflections form an important future work.

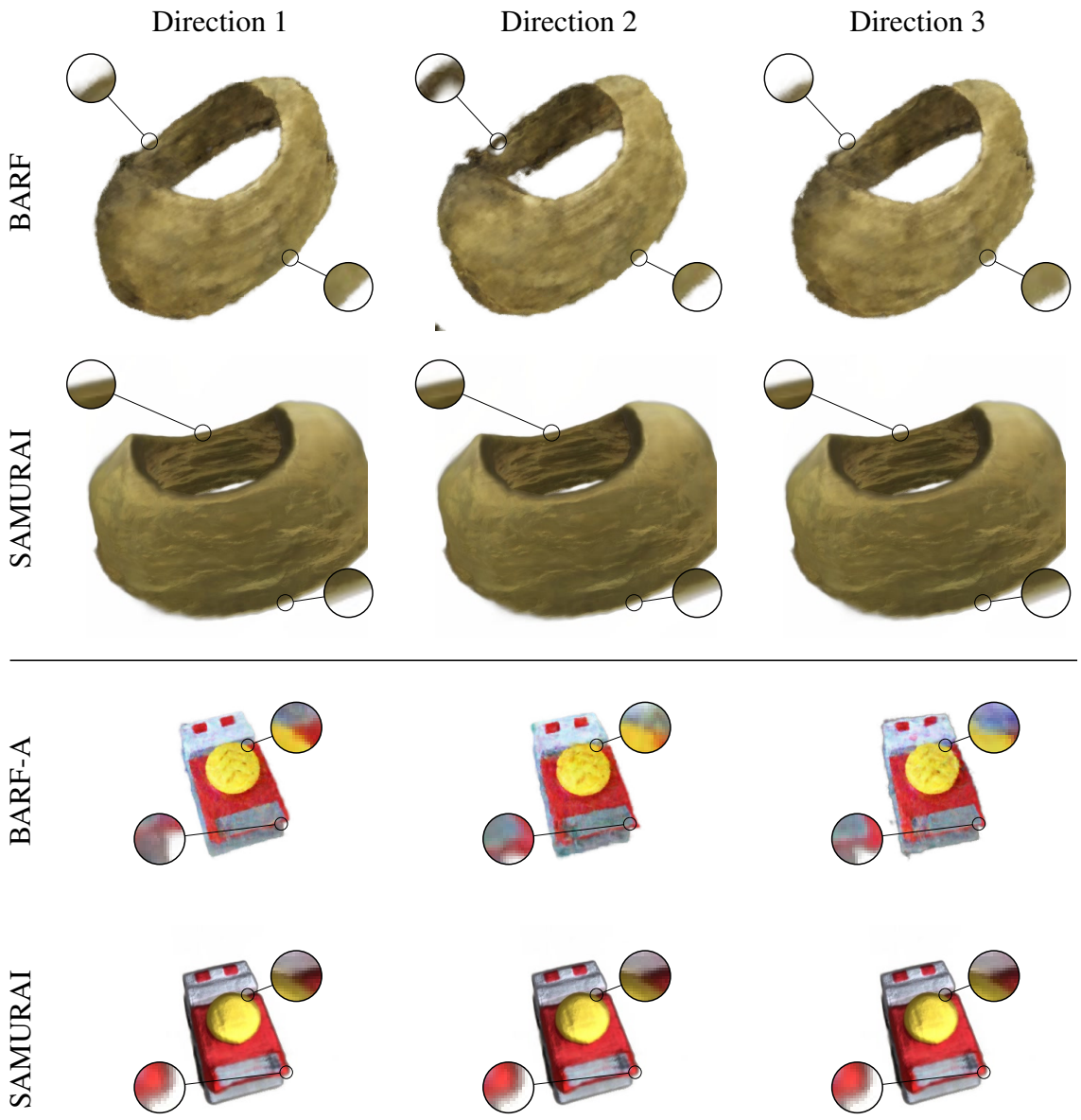



Figure 4.31: **BARF directional conditioning entanglement.** BARF's radiance output is conditioned based on the view direction. When we manipulate the view direction while keeping the camera static, we can see that the embedding is still entangled with the pose. Here, we provide magnifications on edges where this entanglement is noticeable. In a good decomposition, only highlights should be moving, and the texture remains static. However, by shifting the texture, BARF improves the photometric training error of unaligned poses. We also show this for BARF-A, where the radiance is also conditioned on a trainable appearance embedding. Due to different camera coordinate systems and optimizations, we provide similar views from SAMURAI. Here, it is clear that the texture remains static.

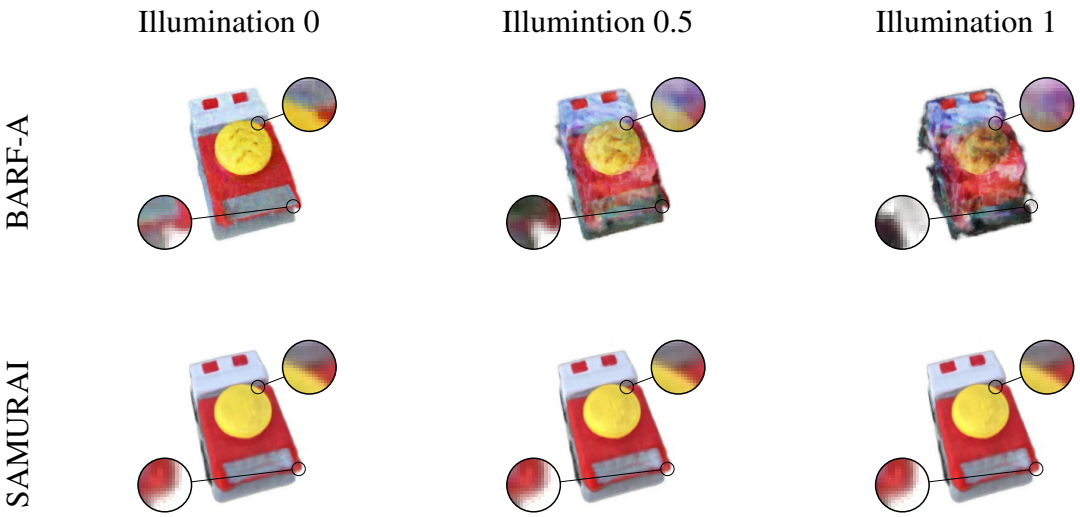



Figure 4.32: **BARF-A illumination conditioning entanglement.** BARF-A's radiance output is also conditioned based on the appearance embedding. When we interpolate between appearances while keeping the camera and view embedding static, we can see that the embedding is also entangled with the pose. Here, we provide magnifications on edges where this entanglement is noticeable. Only the model should be relit, and the texture should remain static. However, by shifting the texture, BARF improves the photometric training error of unaligned poses. The results from SAMURAI remain with a static texture.

## 4.5 Future Work

While significant strides towards automatic decompositions of online image collections are achieved, several challenges remain for future work. Our methods still require a large number of images. For several specific objects, only a few images exist. The introduction of strong priors might be an interesting addition to enable few-shot object decomposition. Furthermore, objects often have different colors, decals or prints, or slight geometric changes, such as open doors for cars. Dealing with the variations is a further highly relevant research direction.

Static objects capture many potential targets for our method, but many objects, even humans and animals, can deform and are dynamic. Solving this task can enable many applications such as AR-based telepresence, where the human avatars can be relit in a way consistent with the environment.

Our methods currently also only consider global illuminations without inter-reflections or shadowing. These challenging additions are also crucial for a successful and accurate decomposition. This is essential when extending the proposed methods from objects to rooms, buildings, or even cities.

While SAMURAI enables decomposition from coarse quadrant-based poses, an annotation is still required. Removing these constraints is an interesting future work direction.

Lastly, our decomposition requires a significant amount of training time. Recent methods such as Instant-NGP [120] or TensoRF [37] achieve training times of radiance fields in minutes or seconds. A combination of SAMURAI with these approaches would enable a decomposition in a time shorter than the pose estimation of COLMAP.

# Chapter 5

# Conclusion

Inverse rendering is a challenging but essential task and, combined with a flexible capture setup, can enable automated 3D reconstructions from online image collections. In this thesis, significant strides in solving this highly ill-posed problem are taken. We propose two categories of approaches to solving this task.

In Chapter 3, we propose a novel cascaded network design coupled with guided prediction networks for SVBRDF and shape estimation from two-shot images. Our key insight is that separating tasks and leveraging a stage-wise prediction can lead to significantly better results than joint estimation with a single large network. We use a two-shot capture setting, which is practical and helps estimate higher quality SVBRDF and shape compared to existing works. Our image capture, network inference, and rendering can be easily implemented on mobile hardware. Another contribution is creating the large-scale synthetic training dataset with domain-randomized geometry and carefully collected materials. We show that networks trained on this data can generalize well to real-world objects.

In Chapter 4, we discuss three novel techniques on BRDF, shape, and illumination decomposition. In Sec. 3 we leveraged the SG illumination rendering from Sec. 2.2.1 to model the incoming light. We extend this approach to handle 360-degree asset creation in NeRD of Sec. 4.2. Here, we extend the recent *Neural Fields* of Sec. 2.4 to handle *reflectance fields* and enable the decomposition of highly challenging scenes under multiple illuminations. NeRD achieved state-of-the-art decomposition quality. However, the method showed limitations from the spherical nature of the SGs illumination model. Additionally, in Sec. 3.2, we found that leveraging priors from datasets decomposes novel scenes effectively. Therefore, we propose Neural-PIL in Sec. 4.3, which leverages priors and uses an improved pre-integrated illumination rendering from Sec. 2.2.2. Here, we replace the brute force pre-integration with a general neural network. While Neural-PIL achieved state-of-the-art decomposition quality, it still required camera poses. These can be hard to obtain in in-the-wild datasets where the objects are in varying locations and under drastically different illumination conditions. We enable these image collections in SAMURAI of Sec. 4.4. Here, we propose a posterior loss scaling, a *Camera Multiplex*,

and a flexible camera and volume parametrization for non-equidistant cameras. With these novel additions, SAMURAI is capable of decomposing online image collections.

Compared to the current methods in the field, our methods lean strongly into this in-the-wild capture scenario. Our methods in the Chapter 4 build towards this goal. Even if methods enable relighting, our methods are the only ones capable of processing varying illumination datasets. This is crucial for processing online image collections, as most objects are captured under various illuminations or even in various places. We found that traditional pose estimation methods (*e.g.* COLMAP) fail in these conditions and propose SAMURAI of Sec. 4.4 to enable these compositions. With this method, we achieved the goal of decomposing online image collections with minimal user interactions. In our work in Chapter 3, we also achieve highly practical capture setups with a complex BRDF model. This is in contrast to more constrained capture setups or limited BRDF models in other methods in the field. To showcase our unconstrained capture setup, we introduced a mobile phone application, which captures the images and estimates the decomposition on the device. Our method is the only method that achieves this convenient setup.

The task of full inverse rendering might also be beneficial to other research fields. For example, in Sec. 4.4, we found that our decomposition into a global illumination model and a static BRDF is beneficial for a camera pose estimation. Compared to other methods based on radiance fields, the degree of freedom to express high-frequent details is too large. The texture was capable of shifting on the surface to force a sharp texture for misaligned cameras. However, these misaligned cameras then do not receive a strong signal that they are misaligned, and the optimization stays in this local minimum. With an explicit decomposition, this problem cannot occur, as the texture is defined only based on the 3D location in the volume and not conditioned on anything else.

This explicit decomposition might also benefit other tasks such as scene understanding, where removing illumination might provide more insides into the potential material. Disentangling illumination from the appearance might be beneficial for detecting the material class. *E.g.* detecting if the material is either wood, stone, or plastic might be simpler if the influence of illumination is removed.

The performance of relighting is also improved drastically with a disentangled approach. In this thesis, we have shown that an explicit decomposition is indeed capable of relighting under any illumination plausibly and consistently. For example, in NeRF-W [110] the radiance is conditioned on the illumination embedding and the view direction. Even a separation with a neural BRDF might be beneficial for this task and enable the method to escape the manifold created during the training.

While the proposed methods enable the decomposition of in-the-wild image collections, several topics remain for future research. In Chapter 3 we introduce a method that leverages large datasets to generate an understanding of the interplay of shape, shading, and illumination. Here, we demonstrate that this is possible even when the training set

consists of random shapes and materials without semantic meaning. While Neural-PIL of Sec. 4.3 includes some of these priors from datasets, the shape is still optimized per scene, and the BRDF prior is only valid for singular points. The general texture information for patches is not included in the prior, and many materials, *e.g.* wood, have distinctive patterns. Introducing patch and shape priors in the optimization can lead to faster training and enable sparser image collections [69, 125]. In Neural-PIL, the per-point prior of natural BRDFs also enables more accurate decompositions by building a manifold of possible BRDFs. An extension to patches might also lead to reduced ambiguities and increased recovered details. For example, NeRF-Tex [12] leverages NeRFs to generate highly detailed fur-like textures on flat surfaces. If this can be applied with pre-trained networks, a smooth shape can be reconstructed, and apparent detail is added on top.

Furthermore, the success of Transformers [166] can also be applied to volume rendering similar to IBRNet [173]. Here, the explicit volume rendering is replaced by a Transformer, which converts a sequence of image features from a CNN to the volume density along the ray. This way, IBRNet can skip the training per scene and directly perform novel view synthesis based on images alone.

While SAMURAI enables camera optimization from coarsely posed images, these poses still need to be provided by the user to reduce the risk of falling into local minima due to cameras being on the wrong side of an object. If a video is used for the decomposition, this problem is alleviated. A camera pose is only offset by a small degree between frames; therefore, the optimization can be constrained.

Lastly, several techniques such as TensoRF [37] or Instant-NGP [120] enable fast training of NeRFs. This is due to explicit information storage in a grid and only relying on a small MLP for interpretation. This speed-up will be highly beneficial for our proposed decomposition methods of Chapter 4.

Still, our proposed methods enable highly practical asset creation with minimal user interaction. Each asset is directly usable in standard rendering software or games and can be easily included in AR or VR applications. Our flexible, unconstrained capture setups democratize the creation of these assets compared to traditional laboratory condition acquisition methods. Even with these highly flexible capture setups, we achieve state-of-the-art results in relightable asset creation, enabling real-time rendering even on mobile devices.

# Appendix A

# Two-shot spatially-varying BRDF and Shape Estimation

## A.1 Network Architecture

The proposed cascaded network architecture uses four distinct network architectures. In the following we will denote a regular 2D convolution with a kernel size of 4, a stride of 2, InstanceNorm, ReLU activation and `k` filters as `c-k`. A transposed convolution is called `ct-k` with the same kernel size, stride, and activations.

**Shape Estimation with Merge Convolutions** The input of the shape estimation network is the two-shot input images and the segmentation mask. We use MergeConv blocks in an encoder-decoder architecture. Refer to the paper for details about a MergeConv block. We use four MergeConv blocks for encoding and decoding in a U-net-inspired architecture [135]. The initial input of each of the pathways is one of the two-shot input images channel stacked with the segmentation mask.

To denote the network architecture, we use the following naming scheme. A MergeConv block with a kernel size of 4, a stride of 2, InstanceNorm, and ReLU activation is denoted as `mo-k`. Here, `k` defines the number of output filters for the merged and input pathways. Upsampling or downsampling is denoted in `o`, where `d` is used for the downsampling operation and `u` for the upsampling. A regular convolution with a kernel size of 5 and a stride of 1 is denoted as `c-k`. The `k` parameter also defines the number of output features, and a sigmoid activation function is used. The network architecture is described as:

```
md-32, md-64, md-128, md-256, mu-256, mu-128, mu-64, mu-32, c-4
```

**Shape Guided Illumination Estimation** The input for this network architecture now consists of the two-shot input images, the segmentation mask, and the previous shape estimation (Normals and Depth). Here, we do not employ the merge convolutions, and all inputs are channel stacked. As the network output is 24 RGB values, we only employ an encoder, followed by fully connected blocks. An additional convolution operation is

denoted as `cn-k`, with a kernel size of 3, a stride of 2, and a ReLU activation. Lastly, a fully connected layer is referred to as `f-k`. The architecture is then denoted as:

```
c-16, c-32, c-64, c-128, c-256, c-256, cn-256, cn-512, f-256, ReLU,
                                f-72, Sigmoid
```

The last fully-connected layer consists of 72 outputs corresponding to 24 RGB values for the spherical Gaussian amplitudes.

**Guided SVBRDF Estimation** For BRDF prediction, we stack the channels of the previous predictions and the two-shot input images. The illumination prediction is here appended to each pixel of the input images. An additional output convolution is referred to as `co-k` with a kernel size of 5, a stride of 1, and sigmoid activation. The network architecture is defined as:

```
c-32, c-64, c-96, c-128, c-160, c-192, ct-192, ct-160, ct-128,
                    ct-96, ct-64, ct-32, co-7
```

**Joint Shape and SVBRDF Refinement** Similar to the BRDF estimation, we stack each previous prediction in the channel dimensions. We also add the residual loss image between the input flash image and the re-rendered initial predictions. A ResNet block [64] here consists of two pre-activated 2D convolutions with a kernel size of 3, a stride of 1, InstanceNorm, and ReLU activation. The shortcut connection is added from the input to the final block output. The block is denoted as `r-k`. A final output convolution is denoted as `c0-k` with a kernel size of 5, a stride of 1, and a sigmoid activation. The overall network is described as:

```
c-64, c-128, c-256, r-256, r-256, r-256, r-256, ct-256, ct-128,
                            ct-64, co-11
```

The final output consists of 11 channels corresponding to diffuse (3), specular (3), roughness (1), depth (1), and normal (3).

## A.2 Results

We provide additional visual comparisions with Li *et al*. [98], Nestmeyer *et al*. [124], and Lasinger *et al*. [86] in Fig. A.1, A.2, and A.3, respectively. In these additional examples, our method's general trend for more accurate decomposition remains.

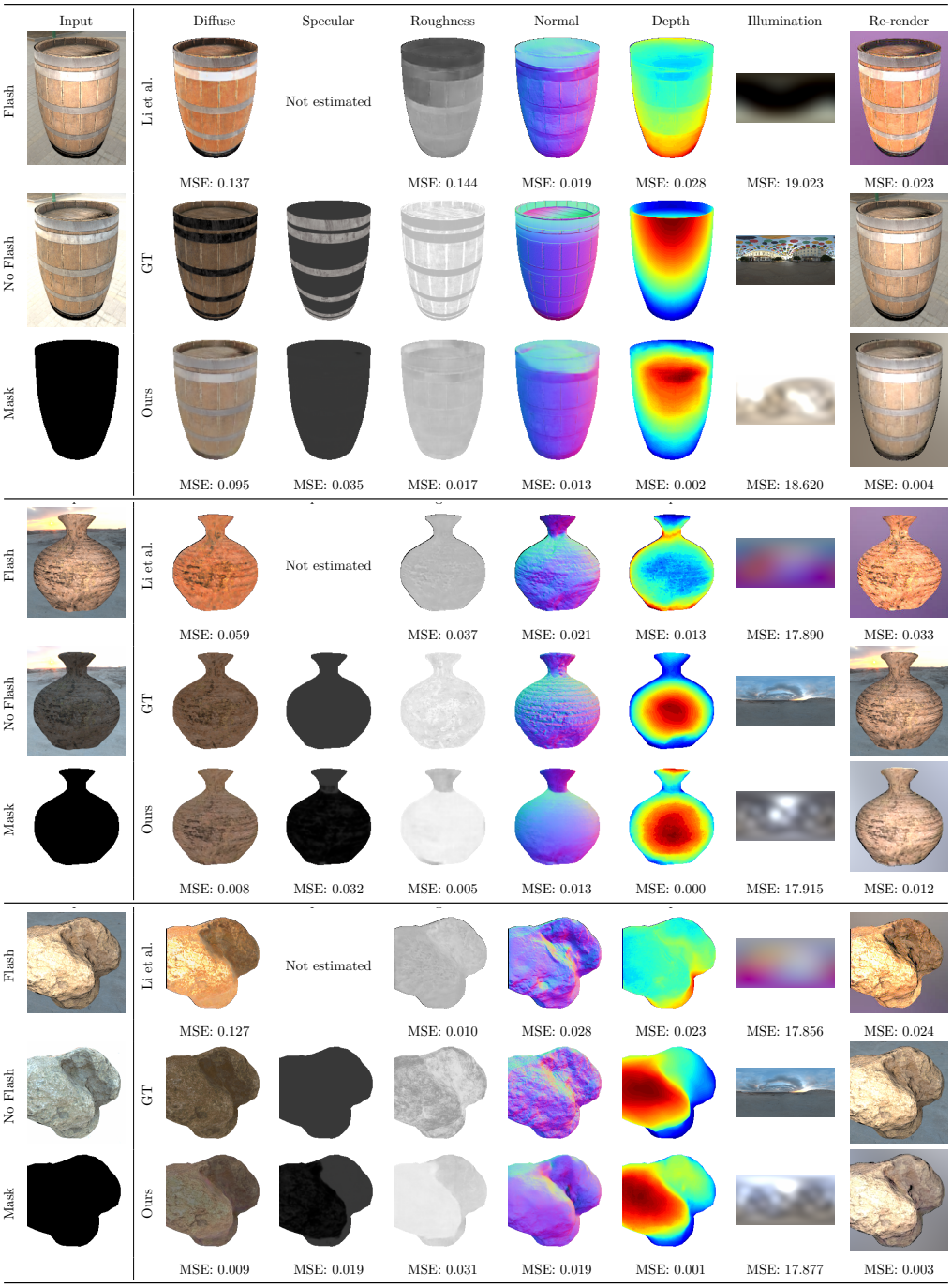



Figure A.1: **Comparison with Li *et al.* [98].** Further comparisons with Li *et al.* On all scenes, our decomposition outperforms the state-of-the-art.

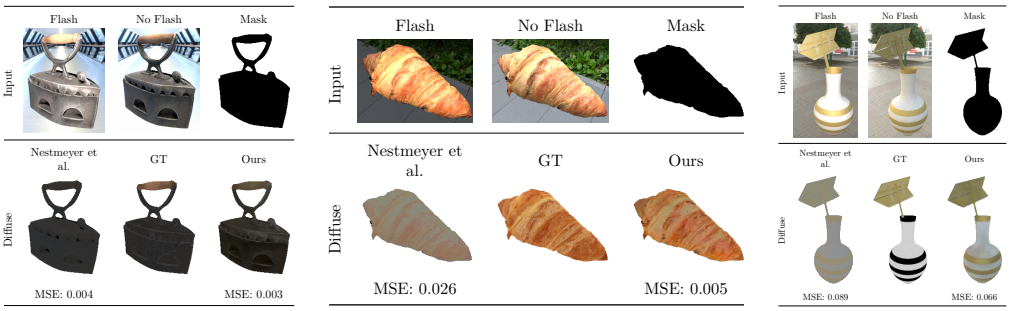



Figure A.2: **Comparison with Nestmeyer *et al.* [124] (RAFII).** Nestmeyer *et al*. only estimate the diffuse albedo. Our prediction estimate the diffuse component more accurately.

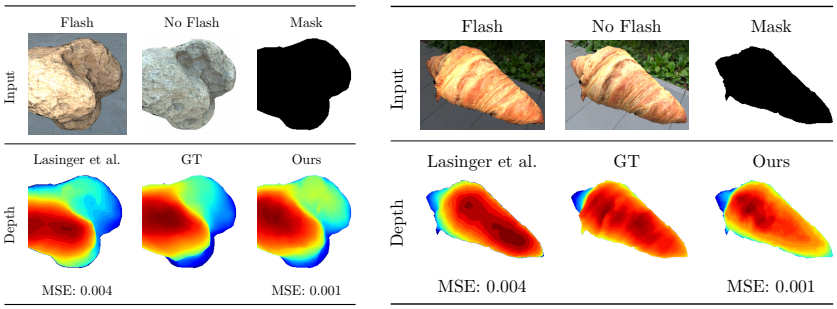



Figure A.3: **Comparison with Lasinger *et al.* [86] (MiDaS).** Lasinger *et al*. only estimate the depth from a single monocular image. Our method captures the general shape in greater detail and more accurately.

# Appendix B

# NeRD: Neural Reflectance Decomposition from Image Collections

## B.1 Dataset details

In Tab. B.1, we list the trained resolution, the number of total images, and the test train split for each dataset. Exemplary images of the real-world datasets are shown in Fig. B.1.

| Dataset | Resolution (W×H) | #Images | #Train | #Test |
|---|---|---|---|---|
| Globe | $400 \times 400$ | 210 | 200 | 10 |
| Car Wreck | $400 \times 400$ | 210 | 200 | 10 |
| Chair | $400 \times 400$ | 210 | 200 | 10 |
| Ethiopian Head | $500 \times 500$ | 66 | 62 | 4 |
| Gold Cape | $456 \times 456$ | 119 | 111 | 8 |
| Gnome | $752 \times 502$ | 103 | 96 | 7 |
| MotherChild | $864 \times 648$ | 104 | 97 | 7 |

Table B.1: **Dataset overview.** Overview of the resolution and number of images used for training.

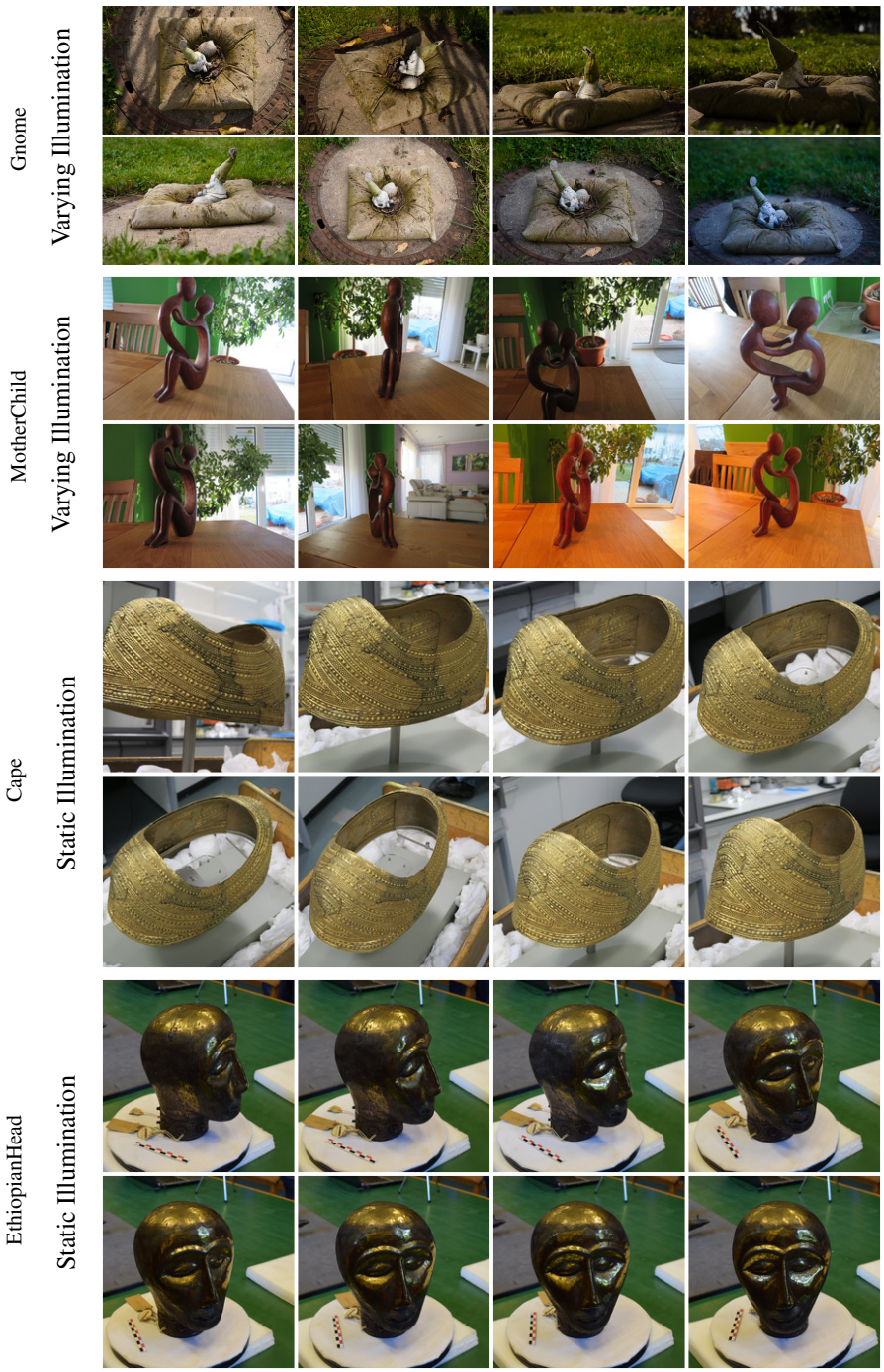



Figure B.1: **Datasets.** Exemplary images of our real-world datasets. Notice the challenging environment illumination in the varying illumination scenes. The gnome dataset even features shadows from the environment.

# Appendix C

# Neural-PIL: Neural Pre-integrated Lighting for Reflectance Decomposition

(a) **Encoder**

| Type | Size | Stride | Features | Activation |
|---|---|---|---|---|
| CoordConv | 3 | 1 | 8 | elu |
| CoordConv | 4 | 2 | 21 | elu |
| CoordConv | 3 | 1 | 21 | elu |
| CoordConv | 4 | 2 | 42 | elu |
| CoordConv | 3 | 1 | 42 | elu |
| CoordConv | 4 | 2 | 64 | elu |
| CoordConv | 3 | 1 | 64 | elu |
| Flatten | | | | |
| MLP | | | 128 | Linear |

(b) **Decoder**

| Type | Size | Stride | Features | Activation |
|---|---|---|---|---|
| ConvT | (1,2) | 1 | 64 | elu |
| ConvT | 4 | 2 | 58 | elu |
| Conv | 3 | 1 | 58 | elu |
| ConvT | 4 | 2 | 52 | elu |
| Conv | 3 | 1 | 52 | elu |
| ConvT | 4 | 2 | 45 | elu |
| Conv | 3 | 1 | 45 | elu |
| ConvT | 4 | 2 | 39 | elu |
| Conv | 3 | 1 | 39 | elu |
| ConvT | 4 | 2 | 32 | elu |
| Conv | 3 | 1 | 32 | elu |
| ConvT | 4 | 2 | 26 | elu |
| Conv | 3 | 1 | 26 | elu |
| ConvT | 4 | 2 | 20 | elu |
| Conv | 3 | 1 | 20 | elu |
| Conv | 1 | 1 | 3 | $\exp(x-1)$ |

(c) **Discriminator**

| Type | Size | Stride | Features | Activation |
|---|---|---|---|---|
| CoordConv | 3 | 1 | 8 | relu |
| CoordConv | 4 | 2 | 32 | relu |
| Conv | 3 | 1 | 32 | relu |
| CoordConv | 4 | 2 | 32 | relu |
| Conv | 3 | 1 | 32 | relu |
| CoordConv | 4 | 2 | 32 | relu |
| Conv | 3 | 1 | 32 | relu |
| CoordConv | 4 | 2 | 32 | relu |
| Conv | 3 | 1 | 32 | relu |
| Flatten | | | | |
| Dense | | | 1 | Linear |

Table C.1: **Light-SMAE architecture.** Details for the architecture used for each network. Conv denotes a regular 2D conv, ConvT a transposed 2D convolution and CoordConv uses a 2D convolution with the coordinates as described in [103].

# Notations

## Symbol Styles

| Style | Description |
|---|---|
| $x$ | Scalars are defined as lower case non-bold symbols. |
| $\boldsymbol{x}$ | Vector are defined as bold lower case symbols. |
| $\boldsymbol{A}$ | Matrices are defined as bold upper case symbols. |
| $f(\ldots)$ | Functions are defined as non-bold symbols with brackets and parameters. |

## Operators

| Operator | Description |
|---|---|
| $\lvert x\rvert$ | The absolute value of a scalar. |
| $\lVert\boldsymbol{x}\rVert$ | The euclidean norm of vector $\boldsymbol{x} = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}$. |
| $\boldsymbol{a} \cdot \boldsymbol{b}$ | The dot or inner product of two vectors $\boldsymbol{a} \cdot \boldsymbol{b} = \sum_{i=1}^{n} a_i b_i + a_2 b_2 + \ldots + a_n b_n$. |
| $\boldsymbol{a} \times \boldsymbol{b}$ | The cross product of two vectors. |
| $\int f(\ldots) * \int b(\ldots)$ | Convolve the function $f$ with $b$. |
| $\nabla_x f(x,y)$ | Partial derivative of the function $f$ to the parameter $x$. |

# List of Symbols

| Symbol | Description | $\in$ |
|---|---|---|
| $\boldsymbol{x}$ | Position in space. | $\mathbb{R}^3$ |
| $\boldsymbol{\omega}_o$ | Outgoing light direction. | $\mathbb{R}^3$ |
| $f_r(\boldsymbol{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$ | The Spatially-varying Bidirectional Reflectance Distribution Function. | $f(\mathbb{R}^3, \mathbb{R}^3, \mathbb{R}^3) \mapsto \mathbb{R}^3$ |
| $\boldsymbol{\omega}_i$ | Incoming light direction. | $\mathbb{R}^3$ |
| $\boldsymbol{n}$ | The surface normal. | $\mathbb{R}^3$ |
| $\boldsymbol{b}_d$ | The diffuse color of the analytical BRDF. | $[0,1] \subset \mathbb{R}^3$ |
| $\boldsymbol{m}$ | The microfacet normal. | $\mathbb{R}^3$ |
| $\boldsymbol{b}_r$ | The roughness scalar of the analytical BRDF. | $[0,1] \subset \mathbb{R}$ |
| $\boldsymbol{h}$ | The half-vector between $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_o$. $\boldsymbol{h} = \frac{\boldsymbol{\omega}_i + \boldsymbol{\omega}_o}{\|\boldsymbol{\omega}_i + \boldsymbol{\omega}_o\|}$. | $\mathbb{R}^3$ |
| $\boldsymbol{b}_s$ | The specular color of the analytical BRDF. | $[0,1] \subset \mathbb{R}^3$ |
| $\boldsymbol{b}_b$ | The basecolor of the analytical BRDF. | $[0,1] \subset \mathbb{R}^3$ |
| $b_m$ | The metallic scalar of the analytical BRDF. | $[0,1] \subset \mathbb{R}$ |
| $\boldsymbol{b}_{\text{ds}}$ | All BRDF parameters of the diffuse-specular parametrization. | $[0,1] \subset \mathbb{R}^7$ |
| $\boldsymbol{b}_{\text{bm}}$ | All BRDF parameters of the basecolor-metallic parametrization. | $[0,1] \subset \mathbb{R}^5$ |
| $\boldsymbol{\omega}_r$ | The reflected outgoing light direction. Now pointing in the incoming light direction. | $\mathbb{R}^3$ |
| $\mathcal{L}_2$ | The sum of squared vector elements with $\mathcal{L}_2(\boldsymbol{v}) = \sum_i^d v_i^2$. | $f(\mathbb{R}^d) \mapsto \mathbb{R}$ |
| $L_o(\boldsymbol{x}, \boldsymbol{\omega}_i)$ | The amount of outgoing light in the specified direction. | $f(\mathbb{R}^3, \mathbb{R}^3) \mapsto \mathbb{R}^3$ |

# List of Abbreviations

**AR** Augmented Reality

**BRDF** Bidirectional Reflectance Distribution Function

**CNN** Convolutional Neural Networks

**DR** Domain Randomization

**GLO** Generative Latent Optimization

**GT** Ground Truth

**HDR** High Dynamic Range

**LDR** Low Dynamic Range

**LiDAR** Light Detection And Ranging

**MAE** Mean Absolute Error

**MLP** Multilayer Perceptron

**MR** Mixed Reality

**MSE** Mean Squared Error

**NVS** Novel View Synthesis

**PDF** Probability Density Function

**PSNR** Peak Signal-to-Noise Ratio

**SDF** Signed Distance Field

**SG** Spherical Gaussian

**SSIM** Structural Similarity Index Measure

**SVBRDF** Spatially-varying Bidirectional Reflectance Distribution Function

**VR** Virtual Reality

# Bibliography


[1] T. C. I. 42. Photography — Digital still cameras — Determination of exposure index, ISO speed ratings, standard output sensitivity, and recommended exposure index. Standard, International Organization for Standardization, 2019.

[2] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[3] E. Adelson and J. Bergen. The plenoptic function and the elements of early vision. *Computation Models of Visual Processing*, 1991.

[4] E. Adelson and A. Pentland. *The perception of shading and reflectance*. Cambridge University Press, 1996.

[5] M. Aittala, T. Aila, and J. Lehtinen. Reflectance modeling by neural texture synthesis. In *ACM Transactions on Graphics (ToG)*, 2018.

[6] M. Aittala, T. Weyrich, and J. Lehtinen. Practical SVBRDF capture in the frequency domain. In *ACM Transactions on Graphics (SIGGRAPH)*, 2013.

[7] M. Aittala, T. Weyrich, and J. Lehtinen. Two-shot SVBRDF capture for stationary materials. In *ACM Transactions on Graphics (ToG)*, 2015.

[8] Y. Aksoy, C. Kim, P. Kellnhofer, S. Paris, M. Elgharib, M. Pollefeys, and W. Matusik. A dataset of flash and ambient illumination pairs from the crowd. In *European Conference on Computer Vision (ECCV)*, 2018.

[9] R. Albert, D. Y. Chan, D. B. Goldman, and J. F. O'Brian. Approximate svBRDF estimation from mobile phone video. In *Eurographics Symposium on Rendering*, 2018.

[10] N. G. Alldrin, T. Zickler, and D. Kriegman. Photometric stereo with non-parametric and spatially-varying reflectance. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[11] L.-P. Asselin, D. Laurendeau, and J.-F. Lalonde. Deep SVBRDF estimation on real materials. In *International Conference on 3D Vision (3DV)*, 2020.

[12] H. Baatz, J. Granskog, M. Papas, F. Rousselle, and J. Novák. Nerf-tex: Neural reflectance field textures. In *Eurographics Symposium on Rendering*, 2021.

[13] J. Barron. *Shapes, Paint, and Light*. PhD thesis, EECS Department, University of California, Berkeley, Aug 2013.

[14] J. T. Barron and J. Malik. Shape, illumination, and reflectance from shading. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2015.

[15] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[16] H. Barrow and J. M. Tenenbaum. Recovering intrinsic scene characteristics from images. *Computer Vision Systems*, 1978.

[17] B. G. Baumgart. *Geometric Modeling for Computer Vision*. PhD thesis, Stanford University, 1974.

[18] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding CVIU*, 2008.

[19] S. Bell, K. Bala, and N. Snavely. Intrinsic images in the wild. In *ACM Transactions on Graphics (SIGGRAPH)*, 2014.

[20] D. Berthelot, C. Raffel, A. Roy, and I. Goodfellow. Understanding and improving interpolation in autoencoders via an adversarial regularizer. *International Conference on Learning Representations (ICLR)*, 2019.

[21] S. Bi, Z. Xu, P. Srinivasan, B. Mildenhall, K. Sunkavalli, M. Hašan, Y. Hold-Geoffroy, D. Kriegman, and R. Ramamoorthi. Neural reflectance fields for appearance acquisition. *ArXiv e-prints*, 2020.

[22] S. Bi, Z. Xu, K. Sunkavalli, M. Hašan, Y. Hold-Geoffroy, D. Kriegman, and R. Ramamoorthi. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. In *European Conference on Computer Vision (ECCV)*, 2020.

[23] S. Bi, Z. Xu, K. Sunkavalli, D. Kriegman, and R. Ramamoorthi. Deep 3d capture: Geometry and reflectance from sparse multi-view images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[24] J. F. Blinn and M. E. Newell. Texture and reflection in computer generated images. In *Communications of ACM*, 1976.

[25] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. P. Lensch. NeRD: Neural reflectance decomposition from image collections. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[26] M. Boss, A. Engelhardt, A. Kar, Y. Li, D. Sun, J. T. Barron, H. P. Lensch, and V. Jampani. SAMURAI: Shape And Material from Unconstrained Real-world Arbitrary Image collections. In *ArXiv e-prints*, 2022.

[27] M. Boss, F. Groh, S. Herholz, and H. P. A. Lensch. Deep Dual Loss BRDF Parameter Estimation. In *Workshop on Material Appearance Modeling*, 2018.

[28] M. Boss, V. Jampani, R. Braun, C. Liu, J. T. Barron, and H. P. Lensch. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[29] M. Boss, V. Jampani, K. Kim, H. P. Lensch, and J. Kautz. Two-shot spatially-varying BRDF and shape estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[30] M. Boss and H. P. Lensch. Single image brdf parameter estimation with a conditional adversarial network. In *ArXiv e-prints*, 2019.

[31] Brian. freepbr, 2019. `https://freepbr.com`.

[32] B. Burley. Physically based shading at disney. In *ACM Transactions on Graphics (SIGGRAPH)*, 2012.

[33] cgtrader. Carwreck. `https://www.cgtrader.com/free-3d-models/vehicle/other/car-wreck-pbr-game-asset`.

[34] cgtrader. Chair. `https://www.cgtrader.com/free-3d-models/furniture/chair/freifrau-easy-chair-pbr`.

[35] CgTrader. Free 3d models, 2019. `www.cgtrader.com`.

[36] E. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein. pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[37] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. TensoRF: Tensorial radiance fields. In *ArXiv e-prints*, 2022.

[38] Z. Chen, S. Nobuhara, and K. Nishino. Invertible neural BRDF for object inverse rendering. In *European Conference on Computer Vision (ECCV)*, 2020.

[39] Z. Chen and H. Zhang. Learning implicit fields for generative shape modeling. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[40] B. O. Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018.

[41] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Transactions on Graphics (ToG)*, 1982.

[42] P. Debevec, A. Wenger, C. Tchou, A. Gardner, J. Waese, and T. Hawkins. A lighting reproduction approach to live-action compositing. *ACM Transactions on Graphics (ToG)*, 2002.

[43] L. Demes. Cc0 textures, 2019. `https://cc0textures.com/`.

[44] V. Deschaintre, M. Aittalla, F. Durand, G. Drettakis, and A. Bousseau. Single-image SVBRDF capture with a rendering-aware deep network. In *ACM Transactions on Graphics (ToG)*, 2018.

[45] V. Deschaintre, M. Aittalla, F. Durand, G. Drettakis, and A. Bousseau. Flexible SVBRDF capture with a multi-image deep network. In *Eurographics Symposium on Rendering*, 2019.

[46] V. Deschaintre, G. Drettakis, and A. Bousseau. Guided fine-tuning for large-scale material transfer. In *Eurographics Symposium on Rendering*, 2020.

[47] V. Deschaintre, Y. Lin, and A. Ghosh. Deep polarization imaging for 3d shape and svbrdf acquisition. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15562–15571, 2021.

[48] Y. Dong, G. Chen, P. Peers, J. Zhang, and X. Tong. Appearance-from-motion: Recovering spatially varying surface reflectance under unknown lighting. *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2014.

[49] Y. Dong, J. Wang, X. Tong, J. Snyder, Y. Lan, M. Ben-Ezra, and B. Guo. Manifold bootstrapping for SVBRDF capture. In *ACM Transactions on Graphics (SIGGRAPH)*, 2010.

[50] S. Donne and A. Geiger. Defusr: Learning non-volumetric depth fusion using successive reprojections. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[51] J. Dupuy and W. Jakob. An adaptive parameterization for efficient material acquisition and rendering. In *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2018.

[52] B. Duvenhage, K. Bouatouch, and D. G. Kourie. Numerical verification of bidirectional reflectance distribution functions for physical plausibility. In *South African Institute for Computer Scientists and Information Technologists Conference (SAICSIT)*, 2013.

[53] M. Faraday. Thoughts on ray vibrations. *Philosophical Magazine*, 1846.

[54] A. Fores, J. Ferwerda, and J. Gu. Toward a perceptually based metric for BRDF modeling. In *Color and Imaging Conference CIC*, 2012.

[55] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[56] D. Gao, X. Li, Y. Dong, P. Peers, and X. Tong. Deep inverse rendering for high-resolution SVBRDF estimation from an arbitrary number of images. In *ACM Transactions on Graphics (SIGGRAPH)*, 2019.

[57] M.-A. Gardner, Y. Hold-Geoffroy, K. Sunkavalli, C. Gagne, and J.-F. Lalonde. Deep parametric indoor lighting estimation. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[58] M.-A. Gardner, K. Sunkavalli, E. Yumer, X. Shen, E. Gambaretto, C. Gagné, and J.-F. Lalonde. Learning to predict indoor illumination from a single image. *ACM Transactions on Graphics (ToG)*, 2017.

[59] A. Gershun. The light field. *Journal of Mathematics and Physics*, 1936.

[60] S. Goel, A. Kanazawa, and J. Malik. Shape and viewpoint without keypoints. In *European Conference on Computer Vision (ECCV)*, 2020.

[61] D. B. Goldman, B. Curless, A. Hertzmann, and S. M. Seitz. Shape and spatially-varying BRDFs from photometric stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2009.

[62] J. C. Gower and G. B. Dijksterhuis. *Procrustes problems*. OUP Oxford, 2004.

[63] R. Guy and M. Agopian. Physically based rendering in filament, 2022. `https://google.github.io/filament/Filament.md.html`.

[64] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision (ECCV)*, 2016.

[65] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec. Baking neural radiance fields for real-time view synthesis. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[66] P. Henzler, V. Deschaintre, N. J. Mitra, and T. Ritschel. Generative modelling of BRDF textures from flash images. *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2021.

[67] M. Holroyd, J. Lawrence, G. Humphreys, and T. Zickler. A photometric approach for estimating normals and tangents. In *ACM Transactions on Graphics (SIGGRAPH)*, 2008.

[68] B. K. P. Horn. *Obtaining Shape from Shading Information*. MIT Press, 1989.

[69] A. Jain, M. Tancik, and P. Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[70] W. Jakob. Mitsuba - physically based renderer, 2018. https://www.mitsuba-renderer.org/.

[71] W. Jang, C. Je, Y. Seo, and S. W. Lee. Structured-light stereo: Comparative analysis and integration of structured-light and active stereo for measuring dynamic shape. *Optics and Lasers in Engineering*, 2013.

[72] C. Je, S. W. Lee, and R.-H. Park. High-contrast color-stripe pattern for rapid structured-light range imaging. In *European Conference on Computer Vision (ECCV)*, 2004.

[73] H. W. Jensen. Global illumination using photon maps. *Rendering Techniques*, 1996.

[74] Y. Jeong, S. Ahn, C. Choy, A. Anandkumar, M. Cho, and J. Park. Self-calibrating neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[75] J. T. Kajiya. The rendering equation. In *ACM Transactions on Graphics (SIGGRAPH)*, 1986.

[76] S. B. Kang, J. A. Webb, C. L. Zitnick, and T. Kanade. A multibaseline stereo system with active illumination and real-time image acquisition. In *IEEE International Conference on Computer Vision (ICCV)*, 1995.

[77] B. Karis. Real shading in unreal engine 4. Technical report, Epic Games, 2013.

[78] J. Kautz, P.-P. Vázquez Alcocer, W. Heidrich, and H.-P. Seidel. A unified approach to prefiltered environment maps. *Eurographics Symposium on Rendering*, 2000.

[79] B. Kaya, S. Kumar, C. Oliveira, V. Ferrari, and L. Van Gool. Uncalibrated neural inverse rendering for photometric stereo of general surfaces. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[80] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006.

[81] P. Kellnhofer, L. Jebe, A. Jones, R. Spicer, K. Pulli, and G. Wetzstein. Neural lumigraph rendering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[82] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ArXiv e-prints*, 2014.

[83] J. Kopf, X. Rong, and J.-B. Huang. Robust consistent video depth estimation. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[84] Z. Kuang, K. Olszewski, M. Chai, Z. Huang, P. Achlioptas, and S. Tulyakov. NeROIC: Neural object capture and rendering from online image collections. *ArXiv e-prints*, 2022.

[85] J. H. Lambert and E. Anding. *Lamberts Photometrie. (Photometria, sive De mensura et gradibus luminis, colorum et umbrae)*. Leipzig, W. Engelmann, 1760.

[86] K. Lasinger, R. Ranftl, K. Schindler, and V. Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2020.

[87] J. Lawrence, S. Rusinkiewicz, and R. Ramamoorthi. Efficient BRDF importance sampling using a factored representation. *ACM Transactions on Graphics (ToG)*, 2004.

[88] H. P. Lensch, J. Lang, M. S. Asla, and H. Seidel. Planned sampling of spatially varying BRDFs. In *Computer Graphics Forum*, 2003.

[89] H. P. A. Lensch, J. Kautz, M. Gosele, and H.-P. Seidel. Image-based reconstruction of spatially varying materials. In *Eurographics Conference on Rendering*, 2001.

[90] L. Lettry, K. Vanhoey, and L. Van Gool. DARN: a deep adversarial residual network for intrinsic image decomposition. In *IEEE International Conference on Computer Vision (ICCV)*, 2018.

[91] M. Li et al. Deep spherical Gaussian illumination estimation for indoor scene. In *ACM Multimedia Asia Conference (MM Asia)*, 2019.

[92] R. Li, K. Xian, C. Shen, Z. Cao, H. Lu, and L. Hang. Deep attention-based classification network for robust depth prediction. In *Asian Conference on Computer Vision (ACCV)*, 2019.

[93] X. Li, Y. Dong, P. Peers, and X. Tong. Modeling surface appearance from a single photograph using self-augmented convolutional neural networks. In *ACM Transactions on Graphics (ToG)*, 2017.

[94] Z. Li, M. Shafiei, R. Ramamoorthi, K. Sunkavalli, and M. Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and SVBRDF from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[95] Z. Li and N. Snavely. Cgintrinsics: Better intrinsic image decomposition through physically-based rendering. In *European Conference on Computer Vision (ECCV)*, 2018.

[96] Z. Li and N. Snavely. Learning intrinsic image decomposition from watching the world. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[97] Z. Li, K. Sunkavalli, and M. Chandraker. Materials for masses: SVBRDF acquisition with a single mobile phone image. In *European Conference on Computer Vision (ECCV)*, 2018.

[98] Z. Li, Z. Xu, R. Ramamoorthi, K. Sunkavalli, and M. Chandraker. Learning to reconstruct shape and spatially-varying reflectance from a single image. In *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2018.

[99] M. Liao, L. Wang, R. Yang, and M. Gong. Light fall-off stereo. In *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[100] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. Barf: Bundle-adjusting neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[101] F. Liu, C. Shen, G. Lin, and I. D. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2016.

[102] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt. Neural sparse voxel fields. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[103] R. Liu, J. Lehman, P. Molino, F. Petroski Such, E. Frank, A. Sergeev, and J. Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[104] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Transactions on Graphics (ToG)*, 2019.

[105] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision - IJCV*, 2004.

[106] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision - IJCV*, 2004.

[107] X. Luo, J. Huang, R. Szeliski, K. Matzen, and J. Kopf. Consistent video depth estimation. In *ACM Transactions on Graphics (ToG)*, 2020.

[108] R. Maier, K. Kim, D. Cremers, J. Kautz, and M. Nießner. Intrinsic3D: High-quality 3D reconstruction by joint appearance and geometry optimization with spatially-varying lighting. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[109] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[110] 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. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[111] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. Image-based visual hulls. In *ACM Transactions on Graphics (SIGGRAPH)*, 2000.

[112] W. Matusik, H. Pfister, M. Brand, and L. McMillan. A data-driven reflectance model. *ACM Transactions on Graphics (ToG)*, 2003.

[113] D. McAllister. *A Generalized Surface Appearance Representation for Computer Graphics*. PhD thesis, University of North Carolina, 2002.

[114] A. Meka, M. Maximov, M. Zollhoefer, A. Chatterjee, H.-P. Seidel, C. Richardt, and C. Theobalt. Lime: Live intrinsic material estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[115] Q. Meng, A. Chen, H. Luo, M. Wu, H. Su, L. Xu, X. He, and J. Yu. GNeRF: GAN-based Neural Radiance Field without Posed Camera. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[116] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[117] B. Mildenhall, P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, 2020.

[118] 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. *ACM Transactions on Graphics (TOG)*, 2019.

[119] G. S. Miller and C. R. Hoffman. Illumination and reflection maps: Simulated objects in simulated and real environments gene. In *ACM Transactions on Graphics (SIGGRAPH)*, 1984.

[120] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 2022.

[121] J. Munkberg, J. Hasselgren, T. Shen, J. Gao, W. Chen, A. Evans, T. Mueller, and S. Fidler. Extracting Triangular 3D Models, Materials, and Lighting From Images. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[122] G. Nam, D. Gutierrez, and M. H. Kim. Practical SVBRDF acquisition of 3d objects with unstructured flash photography. In *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2018.

[123] T. Narihira, M. Maire, and S. X. Yu. Direct intrinsics: Learning albedo-shading decomposition by convolutional regression. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[124] T. Nestmeyer and P. V. Gehler. Reflectance adaptive filtering improves intrinsic image estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[125] M. Niemeyer, J. T. Barron, B. Mildenhall, M. S. M. Sajjadi, A. Geiger, and N. Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[126] M. Oechsle, S. Peng, and A. Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

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

[128] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Deformable neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[129] J. Paulo. 3d textures, 2019. `https://3dtextures.me/`.

[130] D. Pett. Ethiopian Head, 2016. `https://github.com/BritishMuseumDH/ethiopianHead`.

[131] D. Pett. Mold Gold Cape, 2017. `https://github.com/BritishMuseumDH/moldGoldCape`.

[132] X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. Zaiane, and M. Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. In *Pattern Recognition*, volume 106, 2020.

[133] V. S. Ramachandran. Perceiving shape from shading. *Scientific American*, 1988.

[134] B. Resch, H. P. A. Lensch, O. Wang, M. Pollefeys, and A. Sorkine-Hornung. Scalable structure from motion for densely sampled videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[135] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015.

[136] F. Rosenblatt. *The Perceptron, a Perceiving and Recognizing Automaton Project Para*. Report: Cornell Aeronautical Laboratory. Cornell Aeronautical Laboratory, 1957.

[137] C. Rother, M. Kiefel, L. Zhang, B. Schölkopf, and P. Gehler. Recovering intrinsic images with a global sparsity prior on reflectance. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[138] C. Rother, V. Kolmogorov, and A. Blake. Grabcut -interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (SIGGRAPH)*, 2004.

[139] A. Roy and S. Todorovic. Monocular Depth Estimation Using Neural Regression Forest. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[140] S. Sang and M. Chandraker. Single-shot neural relighting and SVBRDF estimation. In *European Conference on Computer Vision (ECCV)*, 2020.

[141] Sara Fridovich-Keil and Alex Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[142] K. Sato and S. Inokuchi. Three-dimensional surface measurement by space encoding range imaging. *Journal of Robotic Systems*, 1985.

[143] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision - IJCV*, 2000.

[144] C. Schlick. An inexpensive BRDF model for physically-based rendering. In *Computer Graphics Forum*, 1994.

[145] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[146] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[147] S. Sengupta, J. Gu, K. Kim, G. Liu, D. W. Jacobs, and J. Kautz. Neural inverse rendering of an indoor scene from a single image. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[148] J. Shi, Y. Dong, H. Su, and S. X. Yu. Learning non-lambertian object intrinsics across shapenet categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[149] S. Song and T. Funkhouser. Neural illumination: Lighting prediction for indoor environments. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[150] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[151] C. Strecha, R. Fransens, and L. Van Gool. Wide-baseline stereo from multiple views: A probabilistic account. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[152] M. X. G. S. A. Studio. Microsoft flight simulator. `https://www.flightsimulator.com`.

[153] R. Szeliski. Rapid octree construction from image sequences. *CVGIP: Image Understanding*, 1993.

[154] M. Tancik, B. Mildenhall, T. Wang, D. Schmidt, P. P. Srinivasan, J. T. Barron, and R. Ng. Learned initializations for optimizing coordinate-based neural representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[155] 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. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[156] M. F. Tappen, W. T. Freeman, and E. H. Adelson. Recovering intrinsic images from a single image. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2005.

[157] S. Textures. Share textures, 2019. `https://sharetextures.com`.

[158] J. Tobin, R. Fong, A. Ray, J. Schneider, Z. Wojciech, and P. Abeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent RObots and Systems IROS*, 2017.

[159] A. Toshev, J. Shi, and K. Daniilidis. Image matching via saliency region correspondences. 2007.

[160] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 969–977, 2018.

[161] Y.-T. Tsai and Z.-C. Shih. All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation. *ACM Transactions on Graphics (ToG)*, 2006.

[162] TurboSquid. Standing Globe. `https://www.turbosquid.com/3d-models/3d-standing-globe-1421971`.

[163] R. Tuytel. Texture haven, 2019. `https://texturehaven.com`.

[164] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. DeMoN: depth and motion network for learning monocular stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[165] C. W. Urquhart, J. P. Siebert, J. P. Mcdonald, R. J. Fryer, and G. House. Active animate stereo vision. In *British Machine Vision Conference (BMVC)*, 1993.

[166] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[167] E. Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford University, 1997.

[168] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan. Ref-neRF: Structured view-dependent appearance for neural radiance fields. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[169] D. Viejo, J. Saez, M. Cazorla, and F. Escolano. Active stereo based compact mapping. In *IEEE/RSJ International Conference on Intelligent RObots and Systems IROS*, 2005.

[170] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance. Microfacet models for refraction through rough surfaces. In *Eurographics Symposium on Rendering*, 2007.

[171] J. Wang, P. Ren, M. Gong, J. Snyder, and B. Guo. All-frequency rendering of dynamic, spatially-varying reflectance. In *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2009.

[172] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[173] Q. Wang, Z. Wang, K. Genova, P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[174] T. Y. Wang et al. Joint material and illumination estimation from photo sets in the wild. In *International Conference on 3D Vision (3DV)*, 2018.

[175] X. Wang, D. Fouhey, and A. Gupta. Designing deep networks for surface normal estimation. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[176] Z. Wang, S. Wu, W. Xie, M. Chen, and V. A. Prisacariu. NeRF−−: Neural radiance fields without known camera parameters. *ArXiv e-prints*, 2021.

[177] H. Weber, P. Donald, and J. Lalonde. Learning to estimate indoor lighting from 3d objects. In *International Conference on 3D Vision (3DV)*, 2018.

[178] S. Weder, J. L. Schonberger, M. Pollefeys, and M. R. Oswald. Neuralfusion: Online depth fusion in latent space. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[179] J. Wei, B. Resch, and L. H. P. A. Multi-view depth map estimation with cross-view consistency. In *British Machine Vision Conference (BMVC)*, 2014.

[180] M. Weinmann, J. Gall, and R. Klein. Material classification based on training data synthesized using a btf database. In *European Conference on Computer Vision (ECCV)*, 2014.

[181] D. White, P. Saunders, S. Bonsey, J. Ven, and H. Edgar. Reflectometer for measuring the bidirectional reflectance of rough surfaces. *Applied optics*, 1998.

[182] R. J. Woodham. Photometric Method For Determining Surface Orientation From Multiple Images. *Optical Engineering*, 1980.

[183] S. Wu, A. Makadia, J. Wu, N. Snavely, R. Tucker, and A. Kanazawa. De-rendering the world's revolutionary artefacts. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6334–6343, 2021.

[184] R. Xia, Y. Dong, P. Peers, and X. Tong. Recovering shape and spatially-varying surface reflectance under unknown illumination. In *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2016.

[185] Z. Xu, S. Bi, K. Sunkavalli, S. Hadap, H. Su, and R. Ramamoorthi. Deep view synthesis from sparse photometric images. *ACM Transactions on Graphics (ToG)*, 2019.

[186] Z. Xu et al. Deep image-based relighting from optimal sparse samples. *ACM Transactions on Graphics (ToG)*, 2018.

[187] L. Yariv, Y. Kasten, D. Moran, M. Galun, M. Atzmon, B. Ronen, and Y. Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[188] W. Ye, X. Li, Y. Dong, P. Peers, and X. Tong. Single image surface appearance modeling with self-augmented cnns and inexact supervision. *Computer Graphics Forum*, 2018.

[189] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[190] G. Zaal. Hdri haven, 2019. `https://hdrihaven.com/`.

[191] D. Zgraggen. cgbookcase, 2019. `https://cgbookcase.com/`.

[192] J. Zhang, G. Chen, Y. Dong, J. Shi, B. Zhang, and E. Wu. Deep inverse rendering for practical object appearance scan with uncalibrated illumination. In *Advances in Computer Graphics*, 2020.

[193] J. Zhang, G. Yang, S. Tulsiani, and D. Ramanan. NeRS: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[194] K. Zhang, F. Luan, Q. Wang, K. Bala, and N. Snavely. PhySG: Inverse rendering with spherical Gaussians for physics-based material editing and relighting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[195] K. Zhang, G. Riegler, N. Snavely, and V. Koltun. NeRF++: Analyzing and improving neural radiance fields. *ArXiv e-prints*, 2020.

[196] L. Zhang, B. Curless, and S. M. Seitz. Rapid shape acquisition using color structured light and multi-pass dynamic programming. In *Symposium on 3D Data Processing, Visualization, and Transmission*.

[197] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah. Shape from shading: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 1999.

[198] 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 *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2021.

[199] Y. Zhang, W. Chen, H. Ling, J. Gao, Y. Zhang, A. Torralba, and S. Fidler. Image GANs meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *International Conference on Learning Representations (ICLR)*, 2021.

[200] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *ArXiv e-prints*, 2018.

[201] T. Zhou, P. Krähenbühl, and A. A. Efros. Learning data-driven reflectance priors for intrinsic image decomposition. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

# Contributions

Except otherwise stated, all mathematical formulations, algorithms, implementations, and evaluations were performed by the author of this thesis. The experimental evaluations of other methods were created by the implementations of the authors or modifications from the thesis author.

The Gnome dataset of NeRD in Sec. 4.2 was captured by Raphael Braun and the MotherChild from Hendrik Lensch. The Cape and Head datasets were created by 'The British Museum'. Raphael Braun implemented and performed the mesh extraction for NeRD.

Varun Jampani created the SAMURAI dataset, and Yuanzhen Li gathered the online image collection for SAMURAI. Andreas Engelhardt performed the Procrustes analysis and ran the evaluation with BaRF, GNeRF and BaRF-A. Andreas Engelhardt also created the implementation of BaRF-A.